MULTI-RADIO COEXISTENCE AND COLLABORATION ON AN SDR PLATFORM

Tommi Zetterman, Antti Piipponen, Kalle Raiskila, Sverre Slotte
(Nokia Research Center, Helsinki, Finland)
{tommi.zetterman, antti.piipponen, kalle.raiskila, sverre.slotte}@nokia.com;

ABSTRACT

In order to support the simultaneous use of both legacy and new radios in a multi-radio handset, the SDR platform needs to offer co-existence mechanisms and services for radios. We propose an SDR control framework to provide the coexistence services, and common interfaces how these services are used. By using our SDR technology demonstrator, we show how multiple simultaneously active radios are controlled dynamically, and how the coexistence mechanism can be used to provide tangible benefits to the SDR modem user.

1. INTRODUCTION

The growing number of wireless standards and the trend of using multiple concurrently active radios in a handset bring new challenges and possibilities for multi-radio capable Software Defined Radio (SDR) platforms. Future cognitive radios that exploit the spectrum holes left unused or underused by the primary radios benefit from an agile platform, which supports run-time opportunity detection and exploitation in a dynamically changing environment with varying combination of other active co-located radios.

To speed up the time to market of new mobile equipment, the development of the modem platform and the radios running on it need to be parallel tasks. This requires a strict definition of how the radios and the SDR modem see each other and how they behave. Moreover, the dependencies between individual radios must be kept minimal in order to develop them in parallel or to acquire them from multiple sources. Implementing a dedicated coexistence solution for every problematic pair of radios is not feasible when the number of radio pair combinations grows or when the coexistence issues include more than two radios.

This necessitates an SDR modem architecture where things that affect multiple radios are implemented as standard services provided by the platform. This was a major design principle in our multi-radio SDR platform architecture design [1].

Besides design time benefits, our architectural approach provides run-time benefits like bringing temporal and spectral agility to all radios which could benefit from it. For example, the platform allows two independent radios unaware of each other having mutual interference to operate simultaneously or share common hardware resources. Coexistence problems are likely to appear more frequently in the future as the number of radios increases, while at the same time trying to utilize all possible parts of the spectrum. In addition, the increasing cost pressure for mobile devices motivates the SDR modem resources to be shared between the radios in a controlled way.

In chapter three, we present our proposed multi-radio control framework and relevant interfaces. In chapter four, we describe the control mechanisms used to establish the coexistence service for radios. In chapter five, we show two examples how the new coexistence services are used.

Both examples are implemented by just using the well-defined platform services and interfaces. There is no need for radio-pair specific coexistence solutions. This means that any radio can connect to this coexistence scheme, even a new one created after the device is manufactured and shipped.

2. RELATED WORK

The primary task of the Multi-radio Controller (MRC) is to solve the coexistence problems caused by radio frequency interference between radios co-located in a small device. Instead of using radio pair specific solutions like the Packet Traffic Arbitration for Bluetooth and 802.11 Wireless LAN [2], the MRC was introduced to provide a generic co-existence mechanism for all radios in our SDR architecture [1]. A similar approach is also described in [3]. We have also showed how our multi-radio control framework can be used to share baseband computation resources between radios [4] as well as radio frequency (RF) hardware [5].

3. MULTI-RADIO COEXISTENCE FRAMEWORK

3.1. SDR Functional Architecture (SDR-FA)

Our mobile device SDR functional architecture has been described in [1] and a technology demonstrator with a prototype implementation of the functional entities in [4]. The architecture presents the SDR subsystem as a set of management, control, and user plane services, that the network layer entities (e.g. Internet Protocol stack, mobility connectivity manager) can access using the Multi-radio Interface.

The services provided by the SDR subsystem are radio (i.e. waveform) independent, following the *radio computer* paradigm supporting the installation of new radio software unknown at compilation time. The SDR-FA is divided into a common control framework (Figure 1) and unified radio applications, which can be any radios whose behavior from the radio computer viewpoint is strictly specified. For multi-radio coexistence the SDR-FA requires the radios to access all platform resources and the spectrum through the common control framework.

The SDR-FA does not specify how the services are implemented. Contrary to for example the Software Communications Architecture (SCA), which describes how software components connect to the middleware and to each other, our SDR-FA specifies how the system behaves on a more abstract level. One could implement the functional architecture entities and their communication using SCA and CORBA middleware, for instance.

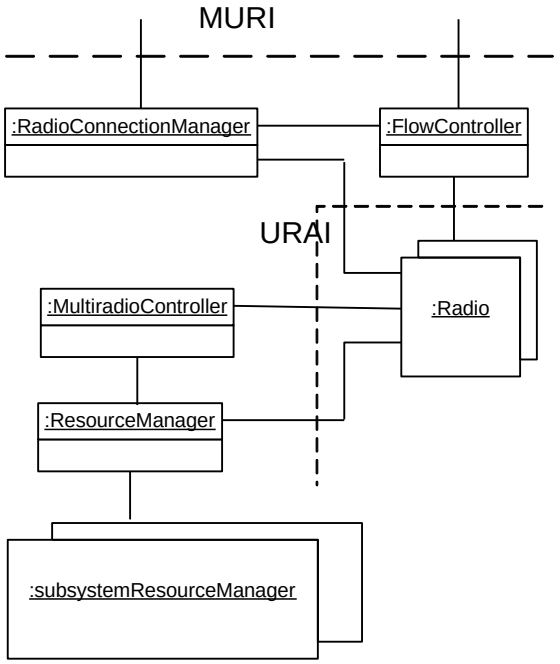

Figure 1 Multi-radio control architecture and interfaces

3.2. Multi-radio Interface (MURI)

The MURI is drawn so that all radio dependent functionality fall within the SDR subsystem. With the traditional Open Systems Interconnection (OSI) model this is roughly between the network and data link layers. The main purpose of the MURI definition is twofold: 1) to provide a generic interface for the radio services in order to ease the development of cross-technology cognitive functionality like intelligent radio selection, and 2) to enable installation of new radio software unknown at compilation time into the SDR subsystem. In addition to the installation and un-installation of radio software, the MURI provides services for connection handling (e.g. scanning for networks, establishing communication data flows, moving data flows from one radio connection to another) and user data transfer.

The interface assumes that all the cross-technology intelligence resides on the network layer, outside of the SDR subsystem, which in turn provides the information needed for decision making. The SDR control framework then handles the coexistence and collaboration between the active radios. In particular, the connection handling services allow the network layer to set up priorities and Quality of Service requirements for the established radio links, which the SDR control framework takes into account when allocating resources.

ETSI Reconfigurable Radio Systems technical committee identified MURI as a potential standardization topic between stakeholder boundaries [6].

3.3. Unified Radio Application Interface (URAI)

The URAI separates the common SDR control framework from the specific radio access technologies. The interface has two parts: 1) the behavior towards the user in the form of services provided by the radios, and 2) the SDR platform resource access in the form of services provided by the control framework to the radios. As described in [4], in addition to the URAI functional specification a radio must adhere to the programming interface of the implementation platform (e.g. SCA, or the interface specification to the reconfigurable radio frequency part such as what has been specified in [7]).

When a radio is activated, it begins to use the SDR platform's computation, memory, communications and RF circuitry resources. Also the radio spectrum can be considered a shared resource, especially in the case where two radios operate on the same frequency band. The use of these resources is requested from the control framework through URAI. The Resource Manager (RM) and the Multi-radio Controller provide semi-static resource allocation and

dynamic resource scheduling in order to keep the radios independent of each other.

3.4 Multi-Radio Controller (MRC)

The Multi-radio Controller was introduced to protect active radios from spectral interference caused by other simultaneously active co-located radios. This is done either by changing the behavior of the victim or the aggressor radio, or if that was not possible, temporally preventing the lower priority radio from operating. To be able to perform its task, the MRC monitors the behavior of the radios dynamically at run-time. Because it has the collected up-to-date information about the current and predicted future states of all radios, it is a natural control point for resource sharing functionality.

The MRC scheduler needs to provide a fast real-time response to radios. All the data needed to make the scheduling decisions are collected into the MRC beforehand. Besides the estimated behavior of radios, the information comprises both static rules, which are defined during design time (e.g. spectral interference rules), and changeable rules, which can be altered when device is used.

The communication between the radios and the MRC is very simple: radios send their estimated future behavior pattern as scheduling requests to the MRC. After that, the MRC compares the scheduling requests of different radios, finds interoperability conflicts using both static and changeable rules, solves conflicts, and sends scheduling decisions back to radios for execution.

The scheduling is done by putting the radios' requests into a common timeline and processing the timeline in small slices, called *scheduling window*. The selection of the length of the scheduling window depends on the set of concurrently active radios. The selection has two effects on the performance of the scheduler. First, the scheduling window sets the upper limit to the length of radio operation time slot processed (and granted) in one go. Longer time slots are scheduled and reported in multiple pieces. Because of scheduling efficiency and to ease the writing of the radio protocol code, it is advisable to make scheduling window at least as long as is the common length of activity slots of frequently used radios (e.g. 1 ms for an LTE sub-frame, 625 µs for a Bluetooth timeslot, and 577 µs for a GSM timeslot). Second, increasing the scheduling window also tightens the deadline determining how much beforehand the radios must send their scheduling requests to the MRC.

Besides the scheduling window, the actual deadline for the scheduling requests also depends on the type of requests needed by radios, and the additional delay of the platform to realize the control and re-configuration. In our SDR demonstrator, the minimum deadline is three times the scheduling window length.

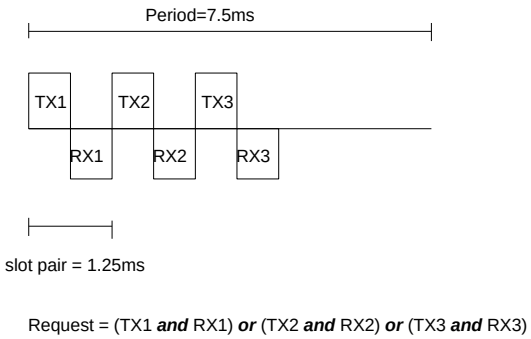

Figure 2 Flexible scheduling operation for Bluetooth radio

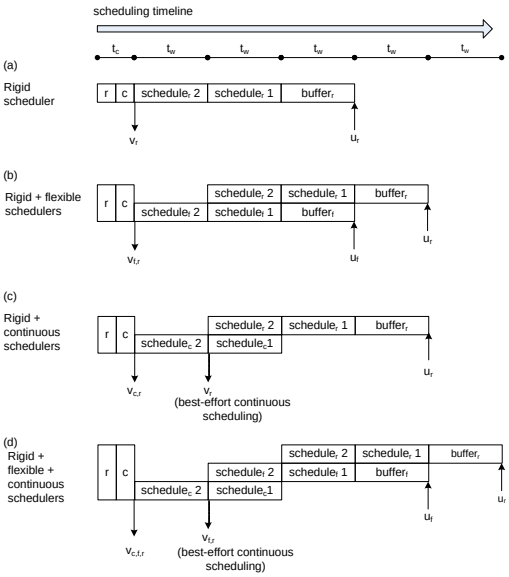

Figure 3 Scheduling timelines

The scheduling tasks, called *operations*, are divided into three different categories based on their semantics. Also, there are three separate schedulers for these different operations.

Rigid operations are individual operation time slots with length not exceeding a scheduling window. From the scheduler point of view, they do not have any relations to other operations, so they are scheduled one-by-one. For example, cellular time-division duplex radios which have their behavior dictated by the base station are well suited to be scheduled as high-priority rigid operations.

Flexible operations consist of multiple rigid operations with boolean relations between them. An example of flexible operation is shown in Figure 2.

Continuous operations are the ones that exceed the scheduling window length, so they are processed (and reported back) piecewise as the scheduling proceeds. The continuous scheduler in our demonstrator supports two modes of operation: conflicting period reporting and conflict-free time slot reporting. The first mode can be used

when a continuously-on radio is expected to have short and relatively rare breaks (either caused by interoperability or resource conflict with another radio), and can use that information to improve its performance (e.g. GPS and DVB-H receivers). The second mode can be used to find time slots long enough for a radio to perform its operation without disturbing other radios.

Figure 3 shows the scheduling timelines implemented in our demonstrator. Each timeline has a time constant added for communication delay *c* and reconfiguration delay *r*. With all three scheduling requests in use, the fastest response time out demonstrator can provide is 9.5 ms.

3.5. Resource Manager (RM)

When the radios use the scheduling services, they already assume that they have the computation, memory, communications and RF circuitry resources available. This is trivially true with traditional implementations using dedicated hardware. In our proposed SDR architecture, the radios share these resources for a more cost-efficient implementation [1].

This resource sharing is based on each radio having its execution split up into long-term behavioral patterns called *operational states*. Examples of these are maintaining a link to the base station, actively transmitting data, or scanning for peers. When the radio must change its operational state, it requests admission from the RM.

The division between admission control and scheduling is motivated by the computation complexity and real-time requirements. The scheduling, in general, is a lightweight computation with strict hard-real time requirements, whereas the admission control is computationally intensive and has more relaxed timing requirements. Therefore, the division of radio behavior to different operational states needs to be coarse enough to avoid unnecessarily frequent operational state changes.

An implementation of this kind of resource sharing for baseband computation was introduced in [4] and an RF platform allowing similar sharing is described in [5].

4. COEXISTENCE MECHANISMS

4.1 Starting and Activating a New Radio

The request to activate a new radio is handled by the Radio Connection Manager (RCM) as depicted in Figure 4. First, a new instance of the radio application is created and connected to the control framework (using the URAI interface). When a radio registers itself, the MRC creates an internal timer to hold a copy of the radio's time. Before scheduling is possible, the radio must synchronize with the MRC's timer, and later update synchronization if needed (for example, when the uncertainty of the clock drift

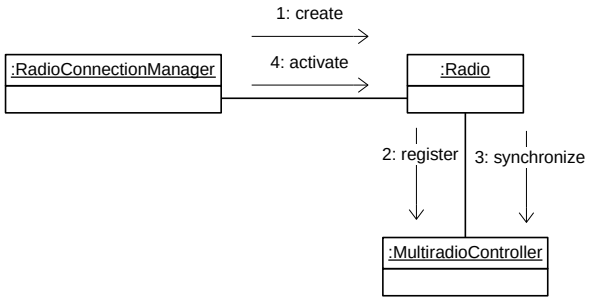

Figure 4 Starting and activating a new radio

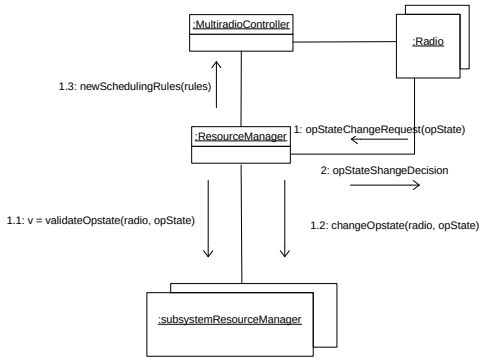

Figure 5 Admission control and operational state change

becomes large enough with respect to the scheduling accuracy). When this is done, the radio is ready to be scheduled and it could be activated to perform scanning, establish a connection, and transfer data.

4.2 Admission Control

The admission control is shown in figure 5. An operational state change request initiates new resource allocation for radios based on their demands associated to their new estimated future behavioral pattern. The request is sent to the RM, which then performs the admission check, asserting that each radio can get its requested share of resources. It further calls all implementation-specific sub-system resource managers to delegate them the sub-system admission control.

When calculating the new resource allocation, the RM may decide to share some of the required resources between multiple radios. This may bring limitations to the simultaneous use of these radios. To be able to dynamically solve possible resource conflicts, the RM communicates these limitations to the MRC as additional scheduling rules. Our demonstrator supports mutual exclusion type of rules for resources, which make the MRC to guarantee the higher priority radio the access to the shared resources in case of conflict.

After the new admission table is ready and the MRC has the new scheduling rules, the operational state change completion is informed back to the radio. Admission check failures are not processed by the SDR control architecture but signaled via MURI to upper layer control entities (e.g. mobility connectivity manager) which have the knowledge about the applications using the radios and can solve the problem for example by stopping one of the low priority radio links.

4.3 Flow Control and Association

After the activation and first admission check, the radio is able to perform a scan to discover communication peers (e.g. base stations). Before user data transfer can occur, the RCM needs to create a new data flow and associate it with the radio. The flow controller takes care of active data flows between user (URI interface) and radios. It buffers the incoming and outgoing data and can also move flows from one radio to another to support intersystem handover based on RCM's instruction. The number of flows connected to a radio is not limited to one. However, the architecture does not specify how a radio uses multiple flows, and the radio application designer is responsible for implementing for example traffic priority handling correctly in case of multiple flows.

5 EXAMPLES

5.1 Fine-grained Spectrum Hole Usage

In the first example, our demonstrator runs three radios simultaneously. Two of them are operating on the 2.4 GHz ISM band (Bluetooth and 802.11 Wireless LAN), and the third one is a cellular LTE radio located near the ISM band (Band VII or Band 40). The use case is defined as:

1) Bluetooth radio is used to connect to an audio headset by extended synchronous connection-oriented link (BT-eSCO) with 2+2 re-transmission slots [8].

2) LTE radio is used to connect to the internet. It is assumed that the cellular base station (eNodeB) takes in use discontinuous reception (DRX) [9] to allow power saving on terminal side.

3) WLAN is used to share the internet connection locally, e.g. the SDR demonstrator provides an internet access point. The WLAN is run in infrastructure mode, and the demonstrator buffers downlink traffic for stations in power-save mode. It is also assumed that most of the traffic is downlink (e.g. data download or data streaming).

Furthermore, it is assumed that if one of the radios is transmitting while either one of the two others is receiving, the receiving radio suffers from interference [10], [11]. In typical cases, not all frequencies may be affected by interference, but here a worst-case scenario is assumed, i.e. LTE at the cell edge, WLAN channel at the closest channel, and so on.

The LTE radio is given the top priority in scheduling, so in case of a scheduling conflict it is favored over the other radios. Its temporal and spatial resource allocation is fully dictated by the eNodeB, so its schedule is naturally presented as series of high-priority rigid requests. With the DRX mode used, communication is bursty with free subframes between bursts.

The Bluetooth radio is given the medium priority. It uses the BT-eSCO link allowing it to use one of the re-transmission slot pairs if the reception or transmission fails. This feature is exploited by precluding the use of primary TX/RX slot pair if that conflicts with the higher priority LTE radio, and delaying the transmission to the first available re-transmission slot pair. Bluetooth schedule is presented as periodical flexible requests defining optional transmission pairs. While there are two possible retransmission slot pairs in the used Bluetooth radio, a single request looks like the one illustrated in figure 2.

The WLAN radio is the lowest priority one and it is granted the conflict-free periods to transmit while the other two radios are not receiving. This is done by using the continuous scheduler, which finds periods long enough to be used for transmission.

In addition to data length, the time needed for WLAN packet transmission depends on network congestion. In case a packet is lost in a congested network, the backoff timer (which is used to define the random contention window, i.e. the period the medium is listened to before retransmission) is increased [12]. However, requiring conservative conflict-free periods causes potential transmission periods to be missed in good network conditions. One possibility is to dynamically adjust the minimum slot length granted by the scheduler, but for simplicity we chose to use a fixed minimum slot length of 1 ms. In case of consecutive errors the backoff timer will gradually increase so that the contention window may increase the total length of MAC frame exchange to exceed the free slot. In that case, the transmission is delayed until a slot long enough is granted, or the contention window drawing gives a period short enough to fit in to the granted period. Our simulations show that with these settings, the WLAN radio is able to utilize the free time slots quite well in different network conditions, offering throughput rate approximately proportional to the time share of the WLAN radio.

Figure 6 illustrates the scheduling results. The last signal *WLAN_TX_GRANTED* shows the interference-free time slots longer than 1 ms found by scheduler. For example, near the cursor the top priority LTE radio causes all primary and backup Bluetooth radio slots to be discarded. Because of that, the scheduler is able to stretch the time periods allocated to the low priority WLAN radio until the LTE radio starts it activity.

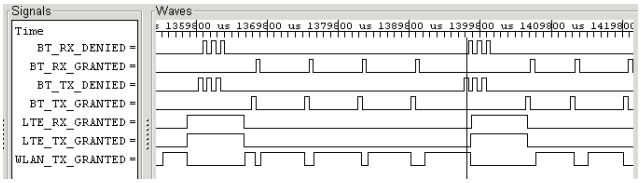

Figure 6 MRC scheduling decisions for finding conflict-free operation slots for WLAN

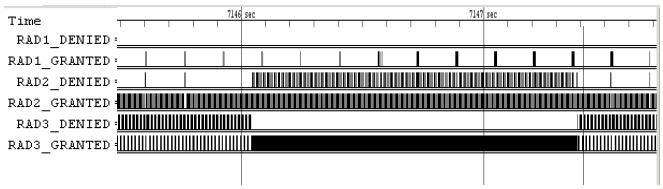

Figure 7 MRC scheduling decisions for traffic offloading

5.2. Traffic Offloading

In this example there are three active radio transmitters called RAD1-3. RAD1 has the highest priority and RAD3 the lowest priority. RAD1 is in a link-keeping operational state, requiring resources only infrequently, RAD2 is used to initially transfer medium-priority stream traffic, and RAD3 to transfer best effort data traffic. Figure 7 shows what happens when the throughput of RAD2 decreases under the limit needed for data stream. This could be e.g. a scenario where RAD2 is a secondary user in a cognitive white space, and the primary user has become active. (Technically, in our demonstrator this is done by blocking scheduling requests so that the remaining bandwidth is not enough for the stream).

The control framework detects the problem and moves the stream data flow from RAD2 to RAD3, which is defined as a secondary radio able to carry the traffic. With the flow movement RAD3 gets also the higher priority from RAD2 to be able to overrun the RAD2 and lower priority radios in scheduling (this is shown by the 'denied' requests for RAD2).

While the traffic is offloaded to the RAD3, RAD2 continues to monitor the status of connection and when the bandwidth is restored, it informs RCM to be able to continue with the stream. In the example RCM then initiates the flow movement back to RAD2.

This offloading does not necessarily need operational state change if the RAD3 has enough bandwidth for stream data in its current operational state (as in the example). This allows the offloading mechanism to respond very quickly to the changing condition in radio throughput.

6. CONCLUSIONS

We presented a multi-radio SDR control architecture with generic interfaces to use, control and dynamically schedule radios. In addition, our platform can provide cognitive features like the ability to use short-time spectrum holes for legacy radios like WLAN. New radios can be connected to the control framework causing the existing radios to adapt their behavior to the new combination of simultaneously active radios. The capabilities of the platform were demonstrated by showing how the radios are scheduled when three conflicting radios are used simultaneously, and how the quality of high priority traffic is maintained while using multiple alternative connections.

Future work and open research questions are:

- Making the MRC aware of the environment surrounding the device by using spectrum sensor…
- …and utilizing that information in scheduling.
- Standardization and optimization of interfaces
- Shortening the reaction time of dynamic scheduler

7. REFERENCES

[1] A.Ahtiainen et al, "Multi-radio Scheduling and Resource Sharing on a Software Defined Radio Computing Platform", In Proc. of the SDR Forum 2008

[2] "Coexistence of Wireless Personal Area Networks with Other Wireless Devices Operating in Unlicensed Frequency Bands", IEEE Std.802.15.2.

[3] J. Zhu, A. Waltho, X. Yang, and X. Guo. “Multi-Radio Coexistence: Challenges and Opportunities”, In Proceedings of ICCCN’07

[4] K. van Berkel et al, "A Multi-Radio SDR Technology Demonstrator", in Proc. of the SDR Forum 2009

[5] A. Immonen et al, "A Reconfigurable Multi-standard radio Platform", accepted for publication in EETR'10, September 2010

[6] ETSI TR 102 680 “Reconfigurable Radio Systems; SDR Reference Architecture for Mobile Device”, V1.1.1, March 2009

[7] Wireless Innovation Forum, Transceiver Facility Specification, V1.0.0, June 2009

[8] Bluetooth SIG, "Specification of the Bluetooth System Version 4.0", 2009

[9] Chandra S Bontu, Ed Illidge, “DRX Mechanism for power saving in LTE”, IEEE Communications Magazine, Volume 47, Issue 6 (June 2009)

[10] H. Holma, A. Toskala, “LTE for UMTS: OFDM and SC-FDMA Based Radio Access”, ch. 11.6, John Wiley & Sons Ltd., 2009

[11] Bluetooth SIG, “Filter Recommendations for Coexistence with LTE and WiMAX”, 2010

[12] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification", IEEE Std.802.11, 1999.