Software Defined Radio: Challenges and Opportunities

1
Author(s)

Tore Ulversoy

View All Authors

Abstract

Abstract: Software Defined Radio (SDR) may provide flexible, upgradeable and longer lifetime radio equipment for the military and for civilian wireless communications infrastructur... **View more**

Document Sections

I. Introduction

Metadata

II. SDR and the Software Communications Architecture
III. SW Architectural Challenges
IV. Challenges and Opportunities Related to Computational Requirements of SDR
V. Security Related Challenges
Authors
Figures
References
Citations
Keywords
Metrics
Related Articles
Show Full Outline

Abstract:
Software Defined Radio … longer lifetime radio equipment for the … ns infrastructure. SDR may also provide … ard-terminals for end users. It is also impo… future context-sensitive, adaptive and l… dios. SDR also poses many challenges, … ve slower than otherwise anticipated. Tr… ze, weight and power issues such as th… rchitectural challenges such as wave… ding implications for regulators, security o…

Published in: IEEE Con… 2 , Issue: 4 , Fourth Quarter 2010)

Page(s): 531 - 550 …ber: 11641603

Date of Publication: 10… .032910.00019

ISSN Information:

…mmunications

Contents

SECTION I.
Introduction

The term ‘Software Radio’ was coined by Joseph Mitola III to signal the shift from HW design dominated radio systems to systems where the major part of the functionality is defined in software. He described the concept in [1].

Software Defined Radio (SDR) has no single, unified, globally recognized definition. Slightly different interpretations exist among the actors in the field. Adding to this, a variety of related terms has been proposed and is used to variable degrees. These include Software Based Radio [2], Reconfigurable Radio, Flexible Architecture Radio [3]. The main parameter in the various interpretations and the related terms, is how flexibly the radio waveform can be changed through changing software (SW) and without modifying the SDR Platform (the combination of hardware and operating environment where the waveform application is running). Obviously the ideal however unrealizable goal is to be able to communicate at any desirable frequency, bandwidth, modulation and data rate by simply loading the appropriate SW. Usually SDR is given a more practical interpretation, implying simply that large parts of the waveform are defined in SW, giving the flexibility to change the waveform within certain bounds as given by the actual system. The flexibility is commonly assumed to extend at least to multi-band and multi-modulation. Examples of specific definitions are those provided by the Software Defined Radio Forum (SDR Forum) [4] and by the Federal Communications Commission (FCC) [5].

The evolution towards SDR systems has been driven in part by the

evolution of the enabling technologies, first and foremost the DA and AD converters and the Digital Signal Processors (DSPs), but also that of the General Purpose Processors (GPPs) and the Field Programmable Gate Arrays (FPGAs). A major driving force has also been the demand for more flexible and reconfigurable radio communication solutions, in particular from the military sector. This demand has resulted in several major governmental development programmes, e.g. in the US the SpeakEasy I, the SpeakEasy II [6] and the ongoing Joint Tactical Radio System (JTRS) [7]–[8][9] programme. Examples from Europe are the "Software Radio Architecture" Plan d'Etude Amont (PEA) [10], [11] in France, the Terminal Radio Software (TERSO) programme [12] in Spain, the Finnish Software Radio Programme (FSRP) [13], the Swedish Common Tactical Radio System (GTRS) [14], [15] and the European Secured Software Defined Radio Referential (ESSOR) [16] programme. The latter is the result of cooperation between several European countries under the banner of the European Defence Agency (EDA).

There are many motivations for utilizing SDR solutions. For the military sector, where communication systems need to have a longer service life time than in the commercial sector, SDR helps to protect investments by prolonging the useful service life of communication systems. This is facilitated through SDR allowing the possibility to change waveforms and/or load new waveforms on already acquired SDR equipment. It also allows SDR applications (waveforms) that are already invested in to be ported to new and more capable SDR platforms. SDR furthermore provides the flexible asset suited for the changing environments of coalition and Network Centric Operations (NCO).

A major motivation within the commercial communications arena, is the rapid evolvement of communications standards, making SW upgrades of base stations a more attractive solution than the costly replacement of base stations.

Common for both the military and the commercial sector, is that SDR opens up a range of possibilities by making existing types of radio applications easier to implement, and by allowing new types of applications. In particular the computing capacity and the flexibility of the SDR may be exploited to develop Cognitive Radios (CR), context-sensitive and adaptive units that may also learn from their adaptations. As an example, the SDR unit may adapt to harsh interference and noise conditions by instantly changing parts of the waveform processing through loading different SW modules, in order to still maintain adequate bit error rates. The cognitive functionality may also be used for improved spectrum utilization, e.g. through the coexistence of cognitive systems with legacy systems.

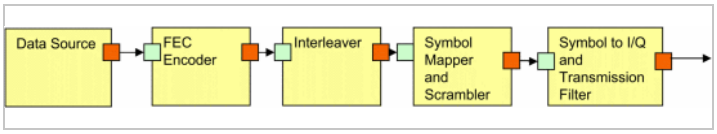

Fig. 1.
An example application component structure, the TX baseband processing of a relatively simple Stanag 4285 waveform. The processed data is sent in packets from the output port of one component to the input port of the next component.

SDR is also beneficial for space applications as it provides the flexibility that will allow deployed satellite communication equipment to be SW upgraded according to advances in algorithms and communication standards. This will allow communication functionality changes and

multiple uses during the lifetime of the satellite [17].

The exploitation of SDR technology in actual products has evolved more slowly than what was anticipated some years ago. As late as in 2002 [2] it was predicted that by 2006 the adoption of SDR in commercial mobile terminals would have “widespread adoption and movement to SDR as baseline design”, something which certainly has not happened. Also the US government JTRS programme has at various times since its initialization in 1997 “experienced cost and schedule overruns and performance shortfalls” [7]. However, lately there have also been several positive signs and accomplishments within SDR, which indicates that we are getting closer to a largerscale adoption of SDR in commercial products. Examples are Vanu Inc.'s ‘Anywave’ base station approved by the FCC [18], the first ‘SCA approved with waivers’ military communications product Thales AN/PRC-148 JEM, and the first ‘SCA approved with no waivers’ Harris Falcon III(TM) AN/PRC-152(C). Also an increasing number of SDR research and prototyping platforms are being offered on the market, along with SDR development tools.

This tutorial will review the fundamental challenges that SDR imposes on the various actors within the field, i.e. developers, security organizations, regulators, business managers, and users. It will further review part of the important past and ongoing work, when available, that has contributed to deal with these challenges. During the discussion of each topic there will be a summary of the remaining open items and/or the projections.

The tutorial will start with SDR SW architecture. As a background to this discussion, Software Communications Architecture (SCA) is briefly reviewed. Then there is a review of the challenges and existing/ongoing work within application portability, application development, the underlying middleware platform and alternative architectures.

A fundamental challenge of SDR is to provide the necessary computational capacity to process the waveform applications, in particular the complex and high data rate waveforms and especially for units with strict power- and size limitations. The computational requirements and the available computing elements required to handle them will be reviewed.

Further the implications for security, regulations and for the radio manufacturer business structure will be discussed and the remaining challenges and/or future projections will be commented on.

SDR poses severe challenges also in analogue RF hardware design and the conversion between the analogue and digital domains, particularly in wideband implementations. In order to limit the scope of this tutorial, and as these topics are not unique to SDR alone, these topics have not been discussed here. A recommended source of information on these topics is the recent work by Kenington [3]. The aspects of AD conversion are also excellently treated in [19] which occurs in what is considered a landmark special issue on Software Radio [20]. Additional recommended sources on SDR receiver front-end technology are [17], [21]–[22][23][24].

SECTION II.
SDR and the Software

Communications Architecture

The most widely used software architecture for SDR is the Software Communications Architecture (SCA). SCA is published by the Joint Program Executive Office (JPEO) for JTRS, with SDR Forum having assisted the JPEO with the development of SCA. SCA is considered a ‘de facto’ standard for the military domain, and has been implemented within the SDR industry, research organizations and at universities.

SCA together with available SCA-based tools allow designers to build component-based SDR applications, as assemblies of components and logical devices. The components are SW processing modules with input and output ports, context dependencies in the form of processing element dependencies, and with settable properties. The logical devices are abstractions for HW modules that in some way process the information stream.

The component-based approach makes the reuse of parts of applications easier as the components have clearly defined inputs, outputs and context requirements, and are deployable units. The component-based approach also promotes a separation of roles in the development. Thus, a radio systems engineer could assemble an SDR application based on preprogrammed components without having to be a SW specialist, whereas a signal processing implementation specialist could concentrate on the processing code of a component.

The SCA is a distributed systems architecture, allowing the various parts of applications to run on different processing elements, i.e. each component is deployed on one of a set of available processors. The communication between the components, and between components and devices, is based on using the Common Object Request Broker Architecture (CORBA) middleware. For communication with specialized processors, e.g. DSPs or FPGAs, SCA advises the use of adapters between CORBA and these units.

The SCA defines a protocol and an environment for the application components. SCA does this by defining a set of interfaces and services, referred to as the Core Framework (CF) [25], by specifying the information requirements and formats for the Extensible Markup Language (XML) descriptions for components and applications, termed the “Domain Profile”, and by specifying underlying middleware and standards. By providing a standard for instantiation, management, connection of and communication between components, the SCA contributes to providing portability and reusability of components.

The CF interfaces are grouped in four sets:

- The Base Application Interfaces provide management and control interfaces for the application components, and they are implemented by these application components.

- The Base Device Interfaces allow management and control of hardware devices through their software interface, and are implemented by the logical device components.

- The Framework Control Interfaces control the instantiation' management and removal of software from the system, and these interfaces are implemented by software modules that form part of the system platform.

- The Framework Services Interfaces provide file functions, and are implemented by software modules that form part of the system platform.

SCA is a general architecture, targeted for but not limited to SDR systems. SCA has similarities with other distributed component architectures, e.g. the CORBA Component Model (CCM). A conceptual difference relative to CCM is the support for system components or 'devices'.

SECTION III.
SW Architectural Challenges

Since the SCA is the dominant SDR architecture, the SCA-related challenges will be focused on first. Then the more general SW architectural challenges and alternatives to SCA will be discussed. The remaining open issues will be highlighted.

A. Portability of SDR SCA-Based Applications

A fundamental challenge of SDR is to provide an ideal platform to application separation, such that waveform applications can be moved from one SDR platform to be rebuilt on another one without having to change or rewrite the application. Such waveform portability is highly desirable, particularly in the military sector, for example in order to achieve interoperability in coalitions by exchanging waveforms.

SCA contributes to such application portability by providing a standard for deployment and management of SCA-based applications [25]. It also standardizes the interconnection and intercommunication both between the components of the application, and between components and system devices. Using the SCA Application Environment Profile (AEP), SCA also standardizes the minimum subset of operating system capabilities that must be available for the applications, and hence the limited subset that applications may use.

The SCA compliance of an application is not sufficient to cover all aspects of portability. Significant pieces that are not standardized by the SCA itself are the APIs to the services and devices of the system platform (see Figure 2). Since these are linked to the actual implementation of the system platform, they are supposed to be standardized per system or domain, as is clearly pointed out in the SCA 2.2.2 specification [25].

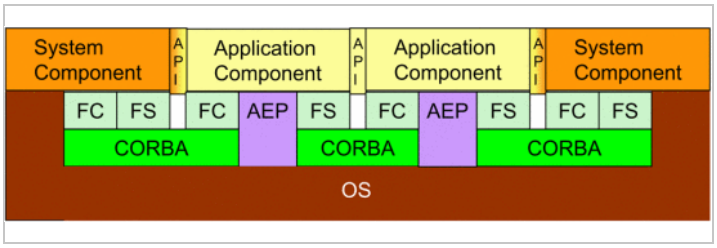

Fig. 2.
A visualization of the layers of the SCA. 'FC' is an implementation of the Framework Control Interfaces, and 'FS' an implementation of the Framework Services Interfaces. 'AEP' is the Application Environment Profile, that limits the applications access to the Operating System (OS).

Within JTRS, a number of such APIs have been developed [8]. Although previously not publicly accessible from the SCA website, 14 APIs were made available in April 2007 [26] and as of February 2009, 18 were available. Presumably this is not a complete set of APIs. Also for some APIs there may be strategic reasons for not wanting to release them from a particular domain, examples are the security-related APIs.

In order for portability to extend across domains, the APIs to the services and devices will need to be standardized across domains as well. With the JTRS APIs now being available, these may be one option for such standardization, particularly for military domains. There are also several other initiatives in this area, including one from the Object Management Group (OMG) Software Based Communications Domain Task Force [27]. Another example is the ESSOR project, which aims at giving “European industry the capability to develop interoperable SDR” [16]. It remains to be seen if ESSOR will develop standards to be used by a European military domain only, or whether this initiative could also contribute to providing inter-domain waveform portability.

An alternative and equivalent approach to that of standardizing APIs to system components is providing abstraction layers between the platform and the application components. An example is a proposal for a “Transceiver Facility Platform Independent Model” [28].

Another related portability issue is the various alternatives for transport mechanisms for the communication with components deployed on DSPs and FPGAs. SCA 2.2.2 prescribes adapters between CORBA and the DSP and FPGA components as the primary means of communication with these elements. The JTRS has standardized a specific adapter referred to as the Modem Hardware Abstraction layer (MHAL) for this purpose [8], [26]. Other similar solutions exist, e.g. Spectrum Signal Processing's ‘QuicComm’ [29]. In recent years, Object Request Broker (ORB) implementations have also been made available on DSPs and FPGAs, making CORBA communication possible also to these components [30]. The fact that various messaging protocols are currently used implies, however, that communication with DSPs and FPGAs will remain a portability issue until one standard or another has become the de facto standard.

Furthermore there are some minor portability issues related to differences in ORB implementations [31].

Lastly, portability obviously requires that the component code is interpreted correctly on the platform. This again has two aspects, language compatibility issues and target processor functionality compatibility. Since SCA is based on CORBA which has support for several programming languages, using different code languages will be possible as long as the appropriate compilers and libraries are available. However, different processing elements, in particular different types of DSPs and FPGAs, support different functionalities and features. This either requires several component implementations, one for each family of processing elements, resulting in an overhead of work-hours used. The other approach is to have the component functionality defined in a high-level language, which is compiled to create a correct code image for the actual processing element to be used. Obviously such a compiler may become very complicated. The resulting target code or image may also become less optimal than a target code written specifically for the target processor.

Further information on portability issues may be found in recent

publications, including API standardization [32], lessons learned from porting a soldier radio waveform [33], SCA aspects in heterogeneous environments [34] and the trade-off between portability and energy efficiency of the processing [35].

The portability issues with SCA-based applications are summarized in Table I. As is evident from the table, important challenges remain in this area.

Table I A Summary of Portability Issues for SCA-Based Applications

Portability aspect:	Standardized through SCA?
Environment and protocol for the installation, instantiation, control, connection and inter-communication of application components	Yes. (But: SCA Security requirements not public)
Defined allowed Operating-System access	Yes, through AEP
APIs to system units (devices) and system services	No. (SCA states this is to be handled per domain.)
Communication (message transport) with specialized processors (DSPs, FPGAs)	No. Multiple solutions available. JTRS has standardized on MHAL.
ORB	SCA specifies CORBA. There are however some minor differences between ORB-implementations.
Programming language	No, but this merely presumes availability of compilers, libraries etc.
Target processor compatibility of the code	No. Different DSPs and FPGAs may support different features.

B. Challenges related to SCA Application Development

For the traditional communications equipment design engineer, with a communications or radio engineering and less of a SW distributed systems background, SCA may appear challenging to learn and understand.

Even for embedded systems engineers without a CORBA and Object Oriented Programming background, according to [36] ‘it could take several months to fully understand SCA'.

SCA tools help abstract away some of the difficulties in SCA. Commercial tools are available from various sources, e.g. Zeligsoft [37], PrismTech [38] and Communications Research Canada (CRC) [39], and there is also a University Open Source initiative, the Ossie Waveform Developer [40] from VirginiaTech. While defining SCA components manually is tedious work involving a lot of XML and CORBA details, the tools allow the SDR designers to define the components through user-friendly tool interfaces. The tools also allow applications to be formed by making connections between the various components, and between the components and the devices of the platform. Still, even with the tools being of significant help in the development process, concluding for example from SCA-based development efforts within own organization, detailed SCA knowledge is still needed and in a starting phase a lot of time is spent on non-signal-processing issues, particularly on a heterogeneous platform.

The tools typically generate the necessary XML and the SCA infrastructure part of the components [41], while functional processing code needs to be added by the designer, either coded manually or using

her/his favourite tools. A more unified higher-level design approach possibly could improve productivity. An approach where the functional skeleton code is imported into a Unified Modelling Language (UML) to allow higher abstraction level modelling of the functional behaviour, is described in [41]. It is envisioned that SDR-design will increasingly be performed at higher abstraction levels, eventually using fully integrated Model-Driven Development (MDD) [42] tools with automatic transformation from model level to any specific heterogeneous platform.

In summary, a further enhancement of the efficiency of designing SCA-based applications, as well as a general availability of MDD tools with fully automated conversion to code level for any given HW platform are important remaining challenges.

C. CORBA Related Challenges

CORBA is demanded by SCA as a middleware platform. The use of CORBA, however, has known challenges in the form of implications on communication throughput, latency and latency variation, as well as an overhead of consumed computation and memory resources. Another issue is that CORBA has lost its popularity in some application domains, which naturally raises the question of whether an alternative middleware is also needed for SDR.

Throughput is a factor of both CORBA and the underlying transport used by CORBA. In [43] throughput of CORBA one-way invocations has been measured using a TAO 1.2 ORB, TCP/IP and 100MBit/sec Ethernet, and compared to using TCP/IP socket programming directly. The results show that the CORBA throughput is highly dependent on message size. With a message size of 64 bytes the CORBA TAO throughput was only a few MBit/sec, whereas with TCP/IP socket programming above 90 Mbit/sec. For message sizes above 8K bytes the throughput came close to that of using socket programming directly. These results show that avoiding too small packet sizes in SCA-based applications is important in order to keep the throughput optimized.

Where the throughput is limited mainly by the underlying transport, other transport mechanisms than TCP/IP may be used with CORBA. “CORBA over RapidIO” [44] and CORBA over a PCI bus [45] are recently described examples.

Latency implies processing delays in the SDR, and tolerable level is application and waveform dependent. As with throughput, latency is a function of both CORBA and the underlying transport. Average latency tends to show a linear relation with message size [44], [46], and with a significant non-zero latency for message size 0. As an example, latency with CORBA over a RapidIO transport between GPPs [44] is measured at 114 μs for size 32 bytes and 180 μs for 4096 bytes. Latency variation may be reduced [47] by using Real-Time CORBA features.

Measurements of CORBA computation overhead for a GPPsystem with the Ossie CF [40] and OmniORB have been provided in [48]. Figure 3 shows an example of how the computation overhead is increased when an application is split into more and more components, and which in this case is dominated by a CORBA-related overhead [48]. The application waveform was a Stanag 4285 TX base band waveform running at an increased symbol rate of 25600 symbols/sec, with frames of 256 symbols being processed at a time.

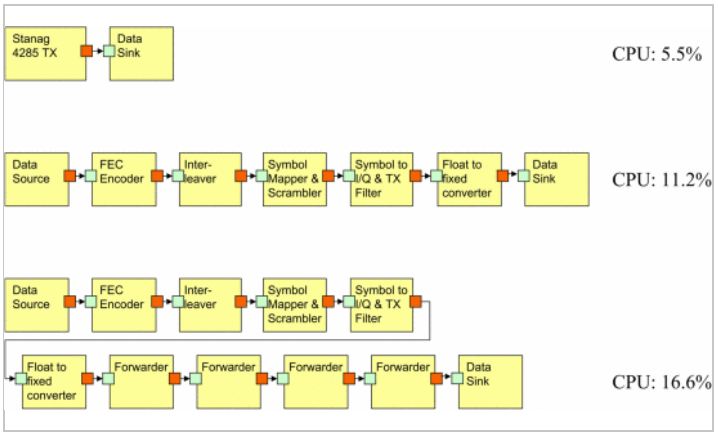

Fig. 3.
Processor workload (user applications + system) in % when running a Stanag 4285 TX waveform, at an increased symbol rate of 25600 symbols/sec, using the Ossie CF on a GPP, and with having the waveform application implemented as one SCA component (top), 6 components (middle) and 6 components with 4 additional (no-functionality) forwarding components (bottom). With the useful signal processing being identical in all three cases, the processor workload is seen to increase significantly as the processing is split into more components.

The memory footprint may be significant on a general fullfeature ORB [49] but slim implementations have been made needing less than 100 kBytes of memory [44], [50].

In order to eliminate CORBA latency, throughput and footprint implications, and in particular when the processing is done on DSPs and FPGAs, data transfers are done on many current SDR systems on specific high-speed connections with low-level formatting [50], [51] instead of using CORBA. A downside of this approach is that it makes portability more difficult, unless the high-speed connections and formatting are standardized. Adapters may be used for control and data from CORBA-enabled processors.

While CORBA used to be limited to GPP processors only, ORBs for specialized processing elements such as DSPs [50] and FPGAs [30], [44], [50], [52] have been developed in recent years. The ORB on an FPGA may be put in an embedded microprocessor [52], or (as a CORBA subset) directly implemented at native gate level [44], [50], where the latter has a processing speed advantage. This theoretically facilitates CORBA communication between all typical processing elements on an SDR platform, which is excellent for portability. The downside of the approach is the amount of resources occupied by these ORBs on the processing elements, and the latency and throughput implications. As for FPGA ones, latency numbers published in [44] and statements in [50] that an FPGA native level ORB may process a message in ‘a few hundred nanoseconds’ indicate that FPGA native level ORBs can now be made very effective in terms of performance, but further public domain results are needed on this subject.

CORBA has its popularity in embedded and real-time applications, but has become less popular in general businessoriented applications. This naturally raises the question of whether there are other likely candidates to take over from CORBA also in SCA and SDR applications. In [53], CORBA has been compared to two main competitors in the businessoriented domain, Enterprise Java Beans (EJB) and Web Services. CORBA was found to be the most mature and better performing technology, 7 times faster than Web Services in a specific single-client evaluation test, but also by far the most complex one. Web

Services was the worst performer but the simplest one and the one that tackled Internet applications best. Web Services is popular in this domain, which illustrates that it is not because it is being outcompeted on performance that CORBAs popularity has diminished, but rather due to other technologies meeting a weighted set of requirements for this type of application better. CORBA's standing in this domain is thus not directly transferrable to the SDR domain, as the SDR domain has more focus on performance issues.

The Data Distribution Service (DDS) has been suggested as an alternative middleware in an SCA context. DDS is an OMG-managed standard for publish-subscribe communication for real-time and embedded systems [54]. DDS belongs to the group of Message-Oriented Middleware (MOM). MOM provides a looser coupling between senders and receivers than in CORBA, messages are sent to an intermediary layer from which the message is received by the addressee [55].

In summary, there are exciting ongoing CORBA activities, such as enabling ORBs to work with fast transports and new ORBs for FPGAs. In the near term, it is anticipated that this migration of CORBA onto specialized processors and faster transports will continue, but that low-level non-CORBA data connections will still be used where they are advantageous. So far there is not a clear path for a middleware that is less complex yet better performing, to potentially take over the role of CORBA in SDR.

D. SCA Challenges and Alternative Architectures

Several technical- and complexity-related SCA challenges have been reviewed in the previous subsections. A further political argument against SCA is that it is also not an open standard as it is directly managed under the supervision of the JPEO. With these issues as a background, it is interesting to explore the alternatives.

A closely related alternative architecture specification for SDR, and derived from the SCA, is OMG's “PIM and PSM for Software Radio Components Specification” [56]. Its current Platform Specific Model (PSM) utilizes CORBA-interfaces [56], but the division in a Platform Independent Model (PIM) and a PSM makes it easier to substitute CORBA with some other middleware, if more suitable middleware platforms were to emerge in the future. OMG's standards are open ones also in the sense that all members have an equal vote on the final content of a standard. OMG's specification is used by the WINTSEC project [57] in Europe. It has been put forward as a promising candidate for future use in the commercial domain [58].

Of particular interest for resource-constrained systems is NASA's ‘Space Telecommunication Radio System’ (STRS) [59]–[60][61] architecture. Electronic devices used in space require radiation hardening [59], and processors are hence slower than terrestrial equivalents, which places further requirements on reduced resource consumption on the application and runtime environment. STRS has many characteristics in common with SCA, such as the separation of waveform applications from hardware, but there are also differences. No particular communication mechanism is described, i.e. CORBA is neither mandated nor precluded. Likewise, an XML parser is not part of the STRS infrastructure, XML files may be pre-processed prior to deployment [59]. STRS does not have the notion of ports but rather optional source and sink APIs [60]. The standard is a NASA managed one, but it is influenced through collaboration with OMG and SDR

Forum [61]. An open-source implementation, “Open Space Radio” [62], has been made available by Virginia Tech.

The GNU radio architecture [63] is an open-source initiative, where the signal processing is carried out on GPP computers. GNU radio is adapted to the Universal Serial Radio Peripheral (USRP) which converts between base band and RF signals. Radio applications are formed as graphs where the vertices represent signal processing blocks and the edges are the data flow, and where the blocks have input and output ports. The signal processing blocks are written in C++ and the graph is connected using the Python programming language. A comparison between GNU radio and a OSSIE SCA-based system has been recently published [64].

Since part of the features of MOM fit well with SDR's needs, there could be a potential for MOM-based architectures as future alternatives for SDR, however this has to be demonstrated.

With the evolution towards cognitive radios which requires the radio to have reasoning capability and adaptivity there will probably be a need for architectural features beyond the present SCA. As an example, with cognitive radios it is beneficial to have convenient framework functions to be able to swap a component from/to a running application in close to real-time. Also, although the cognitive functionality itself, e.g. adaptation of the waveform application to external conditions, may be implemented as application components, it may be beneficial to partly support this functionality through middleware. An example of a middleware-based approach to system self-adaptation is provided in [65].

Concluding on the outlook for SDR architectures, it is expected that the SCA will remain a dominating architecture in the military segment, due to its momentum and the high importance of portability in this domain. In the commercial civilian segment, where there is less focus on portability and more on hardware cost and low power consumption, it is expected that a significant portion of designs will use dedicated and proprietary lighter-weight architectures. In a longer time perspective, with decreasing hardware cost and increasing performance, it is expected that open and standardized architectures such as the OMG one will gain wider acceptance in this sector.

SECTION IV.
Challenges and Opportunities Related to Computational Requirements of SDR

A. Computational Requirements

A fundamental challenge with SDR is how to achieve sufficient computational capacity, in particular for processing wide-band high bit rate waveforms, within acceptable size and weight factors, within acceptable unit costs, and with acceptable power consumption.

This is particularly challenging for small handheld units, e.g. multi mode terminals. The power consumption must be below certain limits to keep the battery discharge time within acceptable limits, and with the smallest handheld units it will also be an issue of not causing the surface

temperature of the device to become unpleasantly high for the user.

For base stations like cellular network infrastructure stations, and for vehicular mounted stations, the power, size and weight factors are easier to accommodate, however performance versus these parameters and cost may still be challenging for complex high bit rate waveforms.

SDR applications perform processing of the various stages of receive and transmit signals, but they also perform protocol handling, application control activities, user interaction and more. Conceptually, as an abstraction, we can consider SDR applications to consist of two main groups of components, (1) Data Processing Components (DPCs) and (2) Event Driven, Administrative and Control Components (EDACCs). DPCs typically have deterministic behaviour, in the form of processing a package of data according to its defined algorithm. DPCs typically also have a high degree of inherent parallelism that may be exploited, an example being a Finite Impulse Response (FIR) filter where a number of additions and multiplications may be carried out in parallel, and the deterministic behaviour also allows low-level optimization of implementations, see, for example [66]. EDACCs depend on events, on data content or user interaction, or perform various administrative and control tasks and are less predictable in their path of execution. Also they typically have far less inherent parallelism.

The SDR components may to a large degree run in parallel, e.g. a decimator component may run in parallel with a filter component and a Fast Fourier Transform (FFT) component, since they work on different stages of the processed data. The Software Communications Architecture (SCA) facilitates this type of parallel processing as it is a distributed systems architecture, where processes may run on several processing elements while exchanging processed data. A good exploitation of this parallelism depends on a well devised component structure of the waveform application, along with optimized deployment of the components on the available processing elements.

With complex waveforms, the DPCs will be the components that require the most computing capacity, and the ones that drive the processing requirements of the SDR computing platform. Table II lists some computational complexity numbers for some key algorithms of the 802.11a waveform at 24 Mbps, as provided in [67]. The calculated complexity numbers are in the form of gigacycles per second referenced to an Alpha GPP. The complexity is seen to be overwhelming for such a single GPP, as the required cycle rate is far above achievable rates.

Table II Computational complexity for some key algorithms of 802.11a

802.11a signal processing operation at 24 Mbps	Gigacycles per second, Alpha GPP
FFT	15.6
FIR (Rx)	6.08
Viterbi decoder	35.0

B. Processing Options

There is a large variety of available processing elements, each with their associated strong and weak points. For the DPCs, processing elements that are able to exploit regular structures, deterministic flows of instructions and internal parallelism will be beneficial from a performance point of view. In the following some of the most important processing alternatives will be reviewed. The alternatives will be listed

according to reconfiguration time, starting with the least configurable options and proceeding to the real-time configurable, as the ability of SDRs to be reconfigured or reloaded with new waveform components or applications is one of its most essential properties. For some SDR applications, it will be sufficient to be able to reconfigure the unit at a maintenance site, and it will not be a problem if it takes a few minutes to load a new application. For other applications, reconfiguration will need to happen while switching from one service, network or waveform standard to another, e.g. while switching from GSM to WiFi, and the reconfiguration should then typically be done in less than a few tenths of a second. For other applications, e.g. a fully context-adaptive SDR, reconfiguration will need to be done in real-time without disturbing any operation of the radio system.

1. Static Processing Elements and Tailored Functional Arrays
In a non-SDR device, the computationally demanding and often highly parallel parts of an algorithm would typically be implemented as logical circuitry in an Application-Specific Integrated Circuit (ASIC). This is often regarded as the optimum solution for computation efficiency and power consumption, and is typically regarded as a reference solution, which the reconfigurable solutions may be compared against. ASIC solutions may provide low unit costs for very high production volumes, but with high development costs.

ASIC implementations are static and as such not usable in an SDR. SDR approaches that focus on the advantages of ASIC implementations, and on the fact that waveforms tend to have common modules, have however been suggested. It has been pointed out [68] that CMOS ASIC devices with more total logic than alternative Field Programmable Gate Arrays (FPGAs), have significantly less quiescent power and dynamic power consumption. Taking this into account, it is further suggested [68] it is beneficial for any design to evaluate the waveforms and determine which functions are common across waveforms, which then could be hosted in an ASIC to allow a low power implementation.

Related suggestions are also discussed in [69], where the processing platform is constructed of interconnected application-domain tailored functional units.

Along the same line, some commercial signal processors are having coprocessors specifically tailored for specific functions [70].

The functional units may be parameterized to adapt to the specific waveform application. By changing parameters and switching functional units in and out, fast reconfiguration within the solution space provided by the functional units is available.

It can be argued that such ASIC-hosted modules or tailored functional units will have a negative impact on portability of waveform applications, limiting portability to platforms that have the necessary ASIC implementations or functional units. Against this it can be argued that having alternative software implementations of the same functionality for more general processing elements, and by allowing the deployment manager to make intelligent decisions on whether to utilize the ASIC-hosted modules, tailored functional units, or more general processing elements, portability will still be achieved. Still, for part of the SDR community, SDR platforms with ASIC-hosted processing modules will not be considered true SDR ones.

2. Reconfigurable Processing Elements

The Field Programmable Gate Array (FPGA) is the reconfigurable alternative to the ASIC. At the expense of higher power consumption and circuit area than the corresponding ASIC solution, an FPGA can be field-programmed with the specific code needed for the specific waveform application. Reconfiguration times may become as low as fractions of a second or just some milliseconds, and hence may allow reconfiguration of the SDR unit to connect to a network via a different waveform standard, for example.

FPGAs have become computationally powerful circuits, and come in many variants. An additional advantage of the FPGAs is the rich availability of toolsets. Further, the amount of designs done for FPGAs and the amount of know-how about this type of designs in typical electronics development organizations' make designs for FPGAs easily planned development tasks with low or moderate risk. Also, compilers are being promoted that make it possible to generate FPGA code directly from Matlab or Simulink, or directly from c-code. This type of compilers can be used for applications such as accelerating bottleneck pieces of c-code running on a GPP or DSP, by converting them to FPGA code. An example is Altera's Nios II C2H compiler which is described in [71].

3. Fast Reconfigurable Units
Configurable Computing Machines (CCM) offer shorter reconfiguration times than FPGAs, and for some types close-to real-time reconfiguration. CCMs are ‘customizable FPGAs with a coarser granularity in its fundamental composition that is better suited for signal processing or wireless applications’ [72]. CCMs have application-domain tailored processing units, connected via a highly flexible and fast reconfigurable fabric.

There is a huge variety of proposed CCMs, both academic initiatives and commercial products. [72] and [3] provide overviews of different CCMs.

CCMs may seem ideally suited for SDRs that need high performance and fast reconfiguration. A disadvantage, however, is the diversity in approaches, which makes efforts to use them very much a unique effort for each type. This also reduces the availability of SW tools for programming them.

4. Real-Time Reconfigurable Units
Microprocessor systems are processing alternatives that provide full real-time programmability. Year after year, processors have shown remarkable performance increases. Whereas cycle clock increases have become more difficult due to technological barriers and also less desirably because of power consumption and power density concerns, the average number of instructions processed per clock cycle has increased by various means. This is often the result of having more features operating in parallel within the processor. This has advanced into multiple-core processors, e.g. multi-core GPPs, multi-core DSPs and Massively Parallel Processor Arrays. Notably, the trend towards parallel processing within microprocessor technology fits well with the characteristics of DPC SDR components.

While GPPs are designed for good average performance for a wide variety of program sources, DSPs have features specifically targeted for digital signal processing, e.g. combined multiply-accumulate operations, and including features to exploit parallelism [73]. There are DSPs that are optimized for performance, and others that are optimized for low power consumption, e.g. for battery-driven applications.

Most multi-core processors may be classified as Single Instruction Multiple Data (SIMD), Multiple Instruction Multiple Data (MIMD), or as a combination of the two.

SIMD units have a single instruction stream, i.e. they execute a single program and each processor is constrained to executing the same instruction. They operate on multiple data streams at a time, i.e. each processor may operate on a different set of data. They are very well suited for algorithms where there is high data parallelism, i.e. a number of identical operations that can or need to be performed at the same point in time, on multiple data sets.

An example of a unit that has several SIMD processing elements is the Cell processor [74]. The peak processing power is quoted at an impressive > 256 GFlops [75], however the corresponding power consumption is not stated, in a chart comparison with other processors [67] it is located at approximately 50W. Although lower-power versions have been released [76], [77], these presumably still have a somewhat high power consumption when considering battery powered applications.

Examples of units that have SIMD processing elements and that are targeted for low-power consuming units, are the NXP Embedded Vector Processor (EVP) [78], [79], the Sandbridge SB3011 [80] and the Icera Livanto [81]. An example university approach is the SODA architecture [67].

It is argued that a solution to the performance/power challenge of the fourth generation communication standards, is an increased number of cores, with each core including a very wide SIMD processor, hence exploiting the parallelism of the algorithms [82].

MIMD units have multiple instruction streams, and operate on multiple data streams at a time. This is hence a more general and flexible architecture than SIMD, allowing separate programs to be executed on each core. This allows problems that exhibit some parallelism, but where the parallelism does not have a regular structure in the form mentioned for SIMD above, to be speeded up through parallel execution.

Graphical Processing Units (GPUs) may also be used for accelerating or running signal processing algorithms [83]. Recent GPUs for the gaming market are powerful computing units with a number of parallel processors, e.g. the nVidia 8800 GTX has 128 floating point units running at 1.35GHz, and an observed 330GFlop/sec [83]. As the processors are targeted specifically at graphics related processing, they have a higher user threshold for general purpose or signal related processing than a GPP. However, due to the attractive prices and high computational capacity of GPUs, they may become attractive in low-budget-type PC-based SDR solutions, as accelerators for processing-intensive blocks. GPUs may also have importance for SDR as high-volume massively parallel computation technology that may be specifically tailored to SDR applications.

5. Processing Elements, Concluding Comments

With the wide variety of types of processing elements available, how do they compare and which ones should be preferred?

The answer depends on the individual weighting of a number of factors, and hence no easy answer is available. In addition to the processing performance the factors of reconfigurability time, power consumption,

size, weight, cost, suitability for the actual processing load and probably many more factors need to be taken into account to make proper choices.

Unfortunately there is no unified scale by which the processing capacity of the different types of processing elements can be judged. For the various processing elements there are different figures of merit that are promoted, examples being MIPS, MOPS, MMACS, MFLOPS and so on. An interesting approach is described in [84] where various elements (ASIC, FPGA, DSP, GPP) are evaluated and comparative charts provided. A specific FFT is used as a benchmark algorithm and Real-Time Bandwidth (RTBW) as a scale, defined as the maximum equivalent analogue bandwidth that the unit is able to process with the given algorithm, without losing any input information. Still these types of comparisons are only valid for the particular benchmark algorithm. Also they do not indicate the capacity of the unit for running other algorithms in parallel, e.g. in an FPGA case.

Guidance may also be obtained from commercially available benchmark analysis reports from Berkeley Design Technology, Inc. (BDTI) [85].

In order to give an indication of what may be achieved with the different types of elements, Table III provides some examples of implementation results from various published work [79], [80], [86], [87].

Table III Waveform Implementation Examples for Different Processing Elements (references, see text)

Proc. element class	Source and type	Implementation(s) and result(s)	Pub. year
FPGAs and CPUs	2x XiLinx XC2V4000, 6000, 8000 2x CPUs (unspecified) 430 MIPS	802.11a W-CDMA	2004
SIMD	Sandbridge Sandblaster SB3011	WiMAX, 2.9Mbps: 80% utilization 802.11b 11Mbps: 55% utilization W-CDMA 2.4Mbps: 87% utilization @0.5W	2007
SIMD	NXP EVP	Estimated approx 45% utilization for HSDPA @ few hundred milliwatts	2005
CCM	Stallion (VirginiaTech)	A benchmark suite of W-CDMA algorithms Computations/sec: 7118 @0.7W	2004
32-bit VLIW DSP	TI C6201	A benchmark suite of WCDMA algorithms Computations/second: 10293 @1.94W	2004

C. Requirements versus Capacity, the Way Ahead

With the continuous improvement of processing elements, will having adequate processing power in handheld SDR terminals soon be a non-issue? To make some projections, and inspired by [88], the data rate evolution of mobile cellular systems [89] has been plotted against the DSP performance evolution of Texas Instruments (TI) DSPs [70], [90], see Figure 4. Estimates of the single channel processing requirements of

2G/3G mobile systems [2], and for a particular 4G case (conditions in [91]) are also plotted.

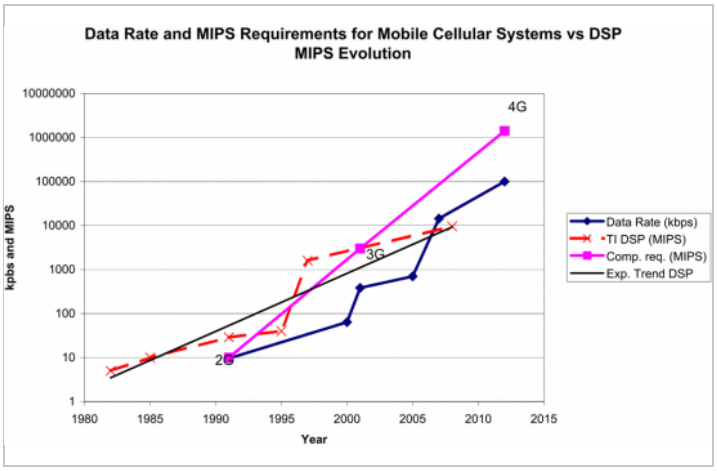

Fig. 4.
Data rate in kbps of the download channel of mobile cellular systems, and approximate MIPS requirements, plotted against the performance evolution of TI DSPs.

For this example, the throughput download data rate in mobile cellular terminals increases at a higher exponential rate than the exponential rate of the DSP processing capacity in MIPS. The required processing rate increases at an even higher pace, this being due to the algorithmic complexity increasing with the generations.

While the progress of processing element capacity continuously makes it easier to meet the capacity requirements of today's existing waveforms, the rapid system evolution particularly in the civilian mobile communications sector indicates that providing adequate processing power at target power consumption will remain a challenge in the years to come, and there will be an increasing need for data processing elements that further exploit parallelism.

SECTION V.
Security Related Challenges

The flexibility benefits of SDR at the same time causes challenges in the security area, both for developers and security certification organizations. In the following the most important of these security related challenges will be reviewed along with important research contributions in these areas, and a summary of the remaining difficulties.

A. Software Load and Protection against Unauthorized SW

A major security challenge is introduced through the possibility to load and install new SW on an SDR unit [92], possibly also over-the-air [23], [92]–[93][94] or via a fixed network [94] connection, and the consequent threat of having unauthorized and potentially malicious SW installed on the platform. This problem domain is very similar to that of maintaining SW installations on personal computers, and avoiding unintended or malicious functions to be installed. With SDRs, the consequences of unauthorized code can be even more far-reaching, from

compromising threats to the user's assets, e.g. his confidential items, via threats to the communication ability of the equipment, to threats to other users and networks, e.g. by the SDR jamming other radio activity [95]. In the USA, SDRs are required by the FCC to have the means to avoid unauthorized SW [96], the specifics of these means are however left to the manufacturer.

If the SW is downloaded over the air, this also exposes the system for someone illegally obtaining the SW (privacy violation) or altering the SW while in transport (integrity violation).

Several publications describe the preventing of unauthorized code by using Digital Signatures [97]–[98][99][100][101][102][103][104]. The manufacturer (or any other party authorizing the code) computes a one-way hash of the code module, then encrypts this hash code using their private key of a private-public asymmetric key pair. This encrypted hash is the digital signature which is added to the code module before it is sent to the SDR platform. A verification application on the SDR platform then verifies the signature by decrypting the signature using the manufacturer's public key, and checks that the decrypted signature equals the one-way hash of the code module. A Digital Certificate is a way of assuring that a public key is actually from the correct source. The Digital Certificate is digitally signed by a trusted third-party. The trusted third-party is verified through a chain of trust to a root certificate on the platform.

A critical issue with the above approach is that root certificates must be distributed to all terminals through a secure out-of-band channel. With a new platform this is easy as root certificates may be factory installed or installed through physically delivered SW, however the issue becomes important when a certificate has expired when the terminal is in the field. A further issue is that of revocation of certificates. A certificate can be revoked at any point in time, and in order for the terminal to know if this is the case or not, it needs to check against a revocation registry, for each and every download operation. It is well known from general computing that this pattern of action is not always obeyed.

Another published way of providing code authorization relies on the sharing of a secret between the SDR platform and the manufacturer. The manufacturer may then make a one-way hash of the SW code and the shared secret and send this hash to the SDR platform together with the SW code [102]. The SDR platform may then verify that the code is from the manufacturer by doing the same one-way hash and compare. A negative implication of this approach is that if the secret is a unique key for each SDR platform, there will be a high number of keys to administrate for the manufacturer. On the other hand, if it is a single key with a wide distribution, this makes it more susceptible to be compromised.

Trusted Computing (TC) functionality [95] is also an optional way to address threats against an SDR platform and against downloaded software. A trusted platform subsystem has a ‘trusted component’ integrated into the platform, which is immutable, i.e. the replacement or modification is under the control of the platform manufacturer. The trusted part may be used for integrity measurements of a program, and for creating certified asymmetric key pairs for the software downloading [95]. TC is further commented in section B.

A suggested further barrier against potentially malicious code is the pre-running of the new SDR component in a “sandbox” [99], [104], a

sheltered environment where it can be evaluated without posing threats to the actual system. Ordinary personal computer protection means as virus protection [92], [103] and memory surveillance [103] may be further barriers, as may also radio emission monitoring [99]. The efficiency of the sandbox pre-running may be debated, as there is no guarantee that the malicious code will expose its behaviour in this test.

The authorization schemes described above also provide integrity protection of the code while in transit. Privacy protection, i.e. protecting the code in transit from being disclosed to a third party, may be achieved through encrypting [98], [105], [106] the code and including the digital signature.

An SDR ideally should have exchangeable cryptographic algorithms too. A motivation for exchangeability of cryptographic components is that even if a current security evaluation does not reveal any weaknesses of some cryptographic approach, cryptanalysis techniques developed later may render it insecure [98], [103], [105]. The cryptographic components are in [98], [103], [105] viewed as a matrix with columns for hash algorithm, digital signature primitive, crypto cipher, secret key and public key, and with rows for the entries for alternative cryptographic components. It is assumed that there is a minimum of two alternatives for each of the crypto components, such that even if there is one that is compromised, there is one that is secure that can be used in the downloading process. Any weak cryptographic component, e.g. a crypto algorithm, is downloaded using trusted crypto components from the matrix, and in an automatic manner.

A solution utilizing Altera Stratix III FPGAs has been described [107]. The FPGA configuration bit stream is transferred to the FPGA in Advanced Encryption Standard (AES) encrypted form, with the FPGA containing a crypto key and a decryption module that allows it to decrypt the configuration bitstream when it is loaded into the circuit. Once inside the circuit, the configuration file cannot be read back [107].

In summary, many research contributions have been made in the area of software download, providing a menu of technological options that can be used. Still, there is increased potential for security threats to SDR systems versus non-reconfigurable ones. With the anticipated creativity of attackers, the specific solutions for the specific SDR systems, and ensuring these resist potential threats, will remain a challenging area for both developers and security organizations. Any allowed downloading of security components will be particularly challenging. Also, the question of who will authorize the SW remains. Should it be the hardware manufacturer, the SW company, a third-party certifier, a government institution, or all the above mentioned. This remains an issue that needs to be further matured.

B. Trusted and High-Assurance Systems

Many communication systems, in particular military ones, have high-assurance security requirements. Demonstrating such high-assurance security on a fully flexible and general computing platform is a very difficult task. This contrasts that the fully flexible platform, where all the functionality is defined in SW applications only, is the ideal computing platform for an SDR in terms of portability.

The high-assurance SDR system will have certain assets, like crypto keys, the user's plain text messages, his/her personal information and more, that need to be protected e.g. for confidentiality and integrity.

Hence practical strong security solutions typically employ combinations of hardware solutions and software solutions [103]. Examples of modules that have impact on the hardware structure are the protected storage for crypto keys, the protected storage for the crypto-algorithm, and the separation between black (encrypted) data and red (plain-text) data (Figure 5). Such architectures are typically custom and dedicated.

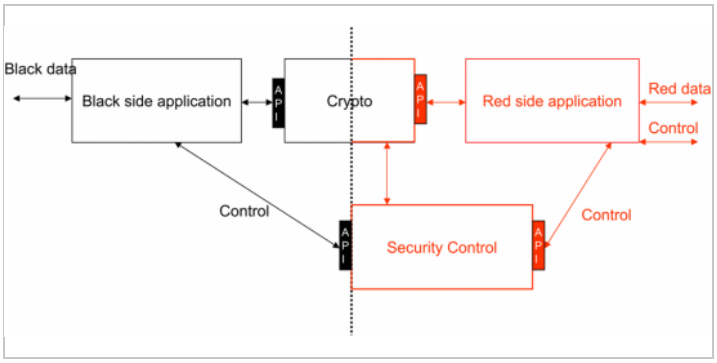

Fig. 5.
An illustration of a red (plain-text) to black separation barrier, and the data and control interfaces to the security related modules.

As mentioned, TC, standardized by the Trusted Computing Group [108], incorporates dedicated immutable hardware elements. The immutable elements are in this case the Root of Trust for Measurement (RTM), Root of Trust for Storage (RTS) and the Root of Trust for Reporting (RTR) [109]. These enable measurements of SW on the platform to be made so that changes can be detected, confidentiality-protection of data, and allow an outside challenger to assess whether the platform is trustworthy. TC also facilitates isolation between different SW components on the platform.

While being developed originally for general computing platforms and PCs, the benefits of TC on SDR platforms have been pointed to [95], and there is also a TCG Mobile Phone Work group in existence [108]. For SDR, TC can provide authenticated download, ensure access to SW only by the intended recipient, detection of malicious or accidental modification or removal, verification of the state that the platform boots into, and isolation of security-critical software [95]. TC functionality does not ensure the integrity of security-critical software while in storage, and does not prevent denialof-service attacks in the form of deletion of the downloaded SDR software [95]. While TC has many beneficial properties for use in SDR, it does not cover every relevant aspect of security for SDR, and hence in itself is not sufficient for a high-assurance SDR system.

Modern military information and communication systems often need to handle information at different security classification levels simultaneously. With conventional security architectures, this requires multiple sets of security HW and processors, which implies both high cost and high power consumption. Examples [110] have also shown that with the conventional security architectures it is demanding to certify the SW security core to the highest assurance levels, as this part of the SW and hence the certification task often grows too large to handle at the highest certification levels.

The Multiple Independent Levels of Security (MILS) architecture offers solutions to these issues. According to [111], the earliest references to MILS are NSA internal papers by Vanfleet and others from 1996. At the

basis of the architecture is a Separation Kernel (SK), as outlined by John Rushby in 1981 [112]. The SK allows several independent partitions, e.g. partitions handling different security levels (Figure 6), to run on the same microprocessor, through separating them in space and time [110]. The SK provides data separation, in ensuring that the different partitions operate on physically separated memory areas. It also provides authorized communication channels between the partitions. Further it provides for sanitization, i.e. the cleaning of shared resources (e.g. registers) before a new partition can use them. Finally it provides damage limitation, in that an error in one partition is not able to affect the processes in the other partitions. The SK schedules processing resources to each partition in such a way that the partition will always be able to process its tasks, independently of any behaviour in the other partitions. The SK is the only piece of SW in the architecture that is allowed to run in supervisor mode, all the partitions run in user mode, which means under no circumstance can they alter the SK.

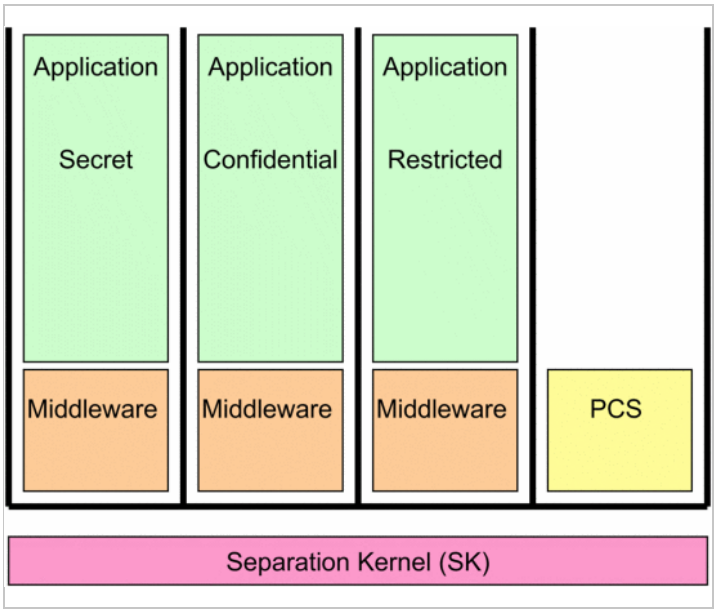

Fig. 6.
The basic building blocks of the MILS architecture.

Since the SK only includes the limited functionality as described above, it can be made fairly small, about 4K lines of code has been quoted [113]. This makes it manageable to certify the SK to the highest Evaluation Assurance Levels (EAL) levels in the Common Criteria [114], EAL 6 and EAL 7. Green Hills Inc recently announced the completion of the certification of their Integrity-178B SK-based OS as the worlds first certified at EAL6+ [115]. It should be noted also that real-time OSes using SK principles have been used for some years already, probably with some more functionality than the minimum-size SK, in the aviation industry, in other embedded applications and in SDR applications.

The SK requires a certain amount of support from specific HW [116], the most important function being the Memory Management Unit (MMU), needed for physical memory separation. A remarkable advantage with the architecture is that the specific HW support needed is already available in many commercial microprocessors [116].

The individual partitions include application code and middleware. Middleware in MILS has a wider interpretation than the conventional one, it includes both traditional communication oriented middleware

such as CORBA, and may also include more OS-related functions [110]. Some partitions may be Single-Level-Secure (handles data at one security level) while others may be Multi-Level Secure (for example a module that downgrades information from one security level to a lower one, through filtering or encryption) [116].

Application and middleware in a partition are to be security evaluated at the appropriate level for that partition, and without needing to take into account the other partitions in the system. This separates the problem of evaluation and certification, and assures that no part of the system needs to be certified at a higher level than what is needed for the security level of the information it is to handle.

Obviously the control of the information flow between the partitions is an important part of the security in a MILS system. The allowed information flow may be planned as a directed graph [110], specifying which partitions are allowed to exchange information in which direction. Within a single processor the information flow is moderated by the SK. In a distributed system with multiple processors involved, the information flow moderation is facilitated through the Partitioning Communication System [113] (PCS).

A vision for MILS, and one that potentially could significantly reduce the amount of time spent on evaluations, is that of a “compositional approach to assurance, evaluation and certification” [117], enabling security evaluation of a system to be based on previous evaluations of the components that it is composed of.

The disadvantages with the MILS architecture include the higher memory consumption due to the partition allocated memory areas, the less dynamic processing performance exploitation due to the need to guarantee processing resources to all partitions, and the higher cost of context switches due to the increased number of separate processes and due to the sanitization operations needed. With the advances in processors and memory devices, these disadvantages have become less important over time.

MILS is a very promising security architecture for SDR, offering a security-domain flexibility that goes hand-in-hand with the waveform application flexibility desired for SDR. MILS for SDR is still a work-in-progress, for example in terms of certifications of components, and it will be very interesting to follow the advances in this area.

In summary, it is challenging for an SDR system to obtain the best possible compromise between high-assurance security and having a computing platform that is as flexible and general as possible. The MILS architecture is very promising in this context, and additionally offers cost-effective handling of multiple security levels and a compositional approach to certification. The further availability of certified MILS components will strengthen its position.

C. Portability of Security Related Modules

As a way to achieve interoperability between secure radio systems, for example in military coalition operations, portability of security SW modules is a highly desired feature.

Code portability between platforms with conventional security architectures requires that the APIs to the security related devices and services are standardized. In many cases each user domain (e.g. a

nation) will be reluctant to disclose security features and APIs, due to the fear that the information may be useful for organizations that wish to develop threats against the type of SDR platform. The issue of whether security features should be disclosed or kept secret ("security by obscurity") is however a debated one. FCC stated in a final rule [118] that "manufacturers should not intentionally make the distinctive elements that implement that manufacturer's particular security measures in a software defined radio public...". SDR Forum in their response [119] pointed to that "History repeatedly has shown that "security through obscurity" often fails, typically because it precludes a broad and rigorous review that would uncover its flaws".

A possible way ahead is to design the security features and the security APIs in such a way that making the security APIs public does not increase the vulnerability of the platform. Another potential solution is having dual security APIs, an intra-domain API and another inter-domain API, where only the inter-domain API is disclosed outside of its own domain.

With the current (2.2.2) [25] version of the SCA, neither the security requirements nor the security APIs have been openly published, making portability between different development domains difficult. The ESSOR project aims at providing what it terms a 'common security basis to increase interoperability between European forces as well as with the United States' [16]. It remains to be seen though, if ESSOR will define the needed security parts for the whole of the European domain or which part, and whether this will contribute in any way to portability with US platforms.

MILS-type architectures are the most promising developments for providing drastic reductions in technical obstacles for portability of security code. Since the MILS-specific HW requirements are already present in many commercial microprocessors, this enables different platforms to provide compatible environments for MILS-type security code. It should be noted though that in the case of implementationspecific additional bindings to non-standard devices this would give similar concerns as discussed earlier.

In summary, the lack of interdomain security APIs and security feature documentation is presently a major challenge and obstacle for SDR application portability. Ongoing initiatives, e.g. ESSOR, are likely to improve this situation by providing complementing standards. MILS-type security architectures have the potential of greatly reducing technical obstacles for portability of security code, such that the dominant issue will be that of trust between organizations and the willingness to share crypto algorithms or having available coalition algorithms (such as the "Suite B" initiative [120]) and security related code. Thus MILS potentially forms an important part of the solution for exchanging secure operational waveforms between nations and thereby achieving multination interoperability in the battlefield.

SECTION VI.
Regulatory and Certification Issues

In the following, certification challenges with SDR equipment are reviewed. The remaining issues are pointed out.

A. SDR Certification

Traditionally, radio equipment has been approved with the specific frequencies, bandwidths, modulations and with specific, and fixed, versions of functionality. This certification regime is challenged when the future application waveforms of the equipment are not known at the time of shipment, and it must be expected that a specific radio platform is updated with new SW versions several times during its lifetime, or even, in future systems, reconfigured dynamically according to communications needs.

1. SDR Certification in the USA

In the USA, significant steps have already been taken in changing certification rules to accommodate SDR equipment. The FCC in the USA adopted rule changes on 13 September 2001 [96] that defined SDRs as a new class of equipment. With the previous rules any changes to output power, frequency or type of modulation implied that a new application form and a new approval would be needed and the equipment be re-labelled with a new identification number. With the changed rules, updates to the software that affected the output power, frequency or type of modulation could be handled in a more streamlined approval process referred to as a "Class III permissive change", provided the equipment originally had been approved as an SDR. An important requirement introduced was that "manufacturers must take steps to prevent unauthorized software changes". The concept of electronic labelling of equipment was also introduced.

The certification rules were further updated on 10 March 2005 [5]. Under these rules, radio equipment that has SW that affects the RF operating parameters, and where this SW is designed to or expected to be modified by a party other than the manufacturer, is *required* to be certified as an SDR. One of the reasons for this change was a fear that third-party SW modifiable radio equipment could otherwise be declared non-SDR, and hence would not be required to have protection against unauthorized software.

These updated rules also define SDR radios as where "the circumstances under which the transmitter operates in accordance with Commission rules, can be altered by making a change in software", which points to the conditional use of spectrum, modulation or output power, as given in FCC regulations.

Certification is required to be carried out at FCC labs, no self-certification or certification by Telecommunications Certification Bodies (TCBs) is allowed.

Further information on SDR certification may be found in [121]. A related standardization effort is IEEE 1900.3 [122].

2. SDR Certification in Europe

In Europe, steps have also been taken that allow faster certification of reconfigured radio equipment. Whereas previous processes demanded independent type approval processes in test houses, the Radio and Telecommunications Terminal Equipment Directive (R&TTE) that came into force in April 2000 in Europe allows self-certifications for telecommunications equipment, yielding faster update cycles when reconfiguration of equipment is needed [123]. Work on making specific adaptations to the R&TTE directive to accommodate SDR products has been carried out by the TCAM Group on SDR (TGS), where TCAM is the Telecommunications Conformity Assessment and Market Surveillance

Committee of the European Union. A final report was presented from TGS in 2004. Based on this and further discussion in TCAM, the European Commission has drawn current preliminary conclusions [124]. Further conclusions have been expected from TCAM, but as of November 2008, none have been drawn up. It is expected that this process will continue. It has been suggested that a specific harmonized standard for SDR in Europe is to be developed [124].

The TGS report identifies 'responsibility for the product' as a key issue, such as when third-party SW is installed on the equipment. The need for a more flexible marking, e.g. digital, is concluded. The need for safeguarding the equipment against unauthorized SW is a discussion point, but currently, unlike in the USA, the manufacturer is not responsible for unauthorized code installation [124].

Further perspectives on regulatory aspects in Europe may be found in [123]–[124][125].

3. Remaining Issues and Projected Evolution of SDR Regulatory Certification

It is clear from the above that the regulatory certification aspects of SDR need to be further matured, both in the USA and in Europe.

The FCC lab certification approach in the USA is likely to become an overwhelming task as the number of products increase and with the product complexity and the amount of functionality in an SDR. Even the Class III procedure for SW updates is likely to saturate with future highly reconfigurable equipment with a vast amount of possible SW combinations. Self-certification in the form of manufacturer Declaration of Conformity, combined with process and organizational certification of the manufacturers to ensure their capability of self-certification, will both allow the tasks to be manageable and give a shorter time to market.

In Europe, more final conclusions on SDR certification need to be drawn and standards updated. The concept of a single party being responsible for the equipment is likely to become increasingly difficult and will need reconsideration.

In both the USA and in Europe, since future dynamic reconfigurable equipment is likely to have a very large number of possible software application combinations, it is unlikely that each and every possible combination of software components can be tested on each and every platform type. An alternative way is to establish trust by testing the components themselves.

A way of creating a further barrier against non-conformity with applicable regulations is to include in each product a mandatory regulation policy enforcement SW module that defines the opportunity window of frequencies, modulations and output power. Additionally, a policy monitor component may be included, that monitors the spectrum from the SDR in a periodic manner and issues alarms or closes down transmission when a policy breach is detected (see Figure 7).

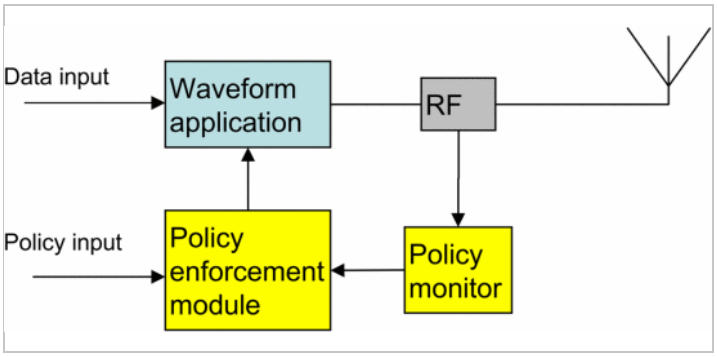

Fig. 7.
A way of creating a further barrier against non-conformance to regulations: A policy enforcement module communicates to the waveform application which regulatory policies apply at the present location and time, e.g. which frequency range and radiated power is allowed to use. A policy monitor periodically checks the actual generated waveform for nonconformance, in which case it will instruct the Policy enforcement module to shut down the transmitter.

B. SCA Compliance and Domain Certification

In market domains where the SCA specification is used, certification of compliance to the SCA specification is likely to be a market demand, and important for application portability. The same applies to other features that are particular to the domain, for example domain APIs. It is a challenge to establish such certification also outside the JTRS.

JPEO states that it is the Certification Authority for the SCA, and that it will assign one or more test organizations as the Test and Evaluation Authority [25]. An overview of the model for certification of JTRS products is provided in [126].

There is a need for architectural certification authorities also in other domains than the JTRS domain, and in other parts of the world than the USA. For example, since it is likely that there will be some differences between European standards and the US one, Europe will need its own certification facilities. Platform and waveform certification needs as defined in the WINTSEC project in Europe are discussed in [57]. Referring to the view of the Finnish Software Radio Programme, a 'European certification network must be operational about 2013–2014' [127]. How this will happen and which organizations will have the responsibility are open issues.

SECTION VII.
Opportunities Related to Business Models and Military and Commercial Markets

SDR provides new product and market opportunities, and has the potential of changing the business models in the radio communication industry. Here, these opportunities are reviewed along with the present status in pursuing these in the military and commercial domains, and with references to recent publications on this subject. Lastly, projections for the further development in this area are provided.

A. Opportunities in the Military Domain

1. SW Upgradeable and Reconfigurable Military Radio Communications Equipment

SDR provides opportunities for having military radio communications equipment which is SW upgradeable and reconfigurable, possibly even field reconfigurable and reconfigurable in space deployment [128]. This represents both benefits for users and a considerable market opportunity for manufacturers.

Since the military domain is characterized by long-lifetime acquisitions, while missions and technical requirements vary at a faster scale, SW upgradability and reconfigurability is very much in demand. Additionally, the possibility of contracting the SW updates from third-party providers may provide more competition and contribute to reduced lifetime-costs for the military.

The military SDR domain in the USA is dominated by the JTRS programme [7]–[8][9], [129]. JTRS has great focus on standardization and portability, with the JPEO managing the SCA. The first production contracts for JTRS 'interim' products were awarded in June 2007 [130], these are variants that meet some basic JTRS requirements [126] but not all requirements as laid out in [126]. According to [131], lowrate production start of 'Handheld, Manpack and Small Form Fit' and 'Ground Mobile' radios are scheduled for 2010 and 2011 respectively.

JTRS has evolved to also be a programme that delivers tactical wireless networking [9] for the US military and thus is an important part of the NCO transformation.

In recent reports [7], [132] the United States Government Accountability Office (GAO) raises its concern about JTRS, pointing to the risks due to the technology challenges [7], [132] and the cost of each unit being significantly higher (up to 10x) than the legacy units they replace [132].

In the military domain in Europe, as in the USA, there is support for and focus on the SCA. The military domain is dominated by several national [12], [13], [15] projects and demonstration platform developments and cooperative [16], [133], [134] projects. Sweden has received its first vehicular mount SCA-based GTRS units [14]. Apart from this it is expected that major development efforts for volume products will await the architectural outcomes of the ESSOR [134] project.

A discussion on SDR processing in satellites is provided in [128].

In summary, the opportunities provided by SDR in the military domain are starting to be exploited in the form of deployed SDR units. Some interim radio types meeting basic JTRS requirements have been contracted and production and deliveries have started. In Europe, several projects are ongoing. Still, taking into consideration the challenges discussed elsewhere in this paper and the concerns raised by GAO, the pace where fixed military radios are replaced by SCA-certified high-flexibility SDR ones is expected to be slow in the next few years.

2. Waveform Library

SDR also provides a possibility for building up libraries of waveform applications. In this way, SDR platforms may be loaded with the specific applications needed in the scenarios and operations they are to be deployed in. Libraries can be national ones or coalitional ones. Library waveforms represent market opportunities for SW companies, as well as units that will be tradable between organizations.

JTRS is building up a repository of waveform applications for porting onto the various platforms. In 2006 it was reported that JTRS code had

accumulated to 3.5 million lines with Government Purpose Rights [8].

In Europe the political and business issues of building up a waveform inventory are difficult, as manufacturers in some cases are the owners of the waveforms and as there are also many national interests. NATO's Industry Advisory Group (NIAG) has investigated “the dynamics and Business Models behind Industrial Contribution of Waveform Standards and how these may and could change with the advent of SDR technology” [133]. NIAG has issued its report but it is unfortunately not publicly available.

Availability of advanced communication waveforms for exchange of video and data in coalitions is a critical issue, which is hampered both by the above-mentioned political and business issues and due to the JTRS ones being classified as “US ONLY” [131]. A way of getting round such political and business issues for coalition waveforms is to develop new waveforms through collaborative contributions from nations. COALWNW is an example of such a multinational cooperative effort [131].

Due to the reasons presented and discussed in Section III, porting efforts are likely to be required for putting library software onto a specific platform.

It is projected that the trend towards building up libraries of waveform applications for national and coalitional use will remain an active one, and that this will be a growing market opportunity for third-party SW companies.

3. Military Cognitive Radio
SDR, as a base implementation technology, provides opportunities for providing the future military versions of Cognitive Radio (CR).

Cognitive radio in the military domain is a highly active research field, that generates considerable interest both as a means for reconfiguring waveforms according to sensed electromagnetic conditions, and as a means to provide increased spectrum utilization through dynamic use of spectrum [135]. It is also foreseen that as more and more radios in battlefield environments will have cognition, and as the cognitive abilities are likely to be used for both offensive and defensive purposes, cognitive abilities in military radio equipment will become mandatory. CR will thus be another major driver for the transition to SDR in the military domain.

The literature on CR and potential use of SDR for CR purposes is overwhelming, a few recommended sources of information are [135]–[136][137][138][139][140].

It is expected that the replacement of military radio systems with smarter CR ones will represent a continued SDR opportunity for many years forward.

B. Opportunities in the Commercial Domain

1. Multiprotocol Multiband Base Stations
SDR provides an opportunity to switch from conventionally designed cellular base stations to Software Defined Multi-Protocol Multi-Band (MPMB) base stations [58].

The reconfiguration possibilities provided by SDR MPMBs

accommodate future cellular base station needs, for example:

- the possibility to dynamically add services
- the rapid introduction of new communications standards [141]
- the trend that new communications standards are put into service in a less mature state than previous standards, implying an increased risk of post-deployment changes needed [141]
- context-related reconfigurability and the accommodation of the future cognitive terminals [125], [142]

SDR MPMBs also allow standardization of hardware platforms, which reduces the amount of capital tied up in hardware inventory. Since the total lifetime cost of the system is more important than the initial cost, the SDR solution may be preferred even if the initial cost of the SDR platform is higher. Also with base stations, increased power consumption over a conventional design can be tolerated.

At present, cellular base stations are dominated by the traditional non-SDR ones. The VANU [18] base stations, as well as some recent announcements from Huawei [143] and ZTE [144], are SDR examples.

Further background on SDR base station opportunities is provided in [58], [142].

The share of SDR base stations compared to the overall number is predicted to grow significantly from 2010, to become an approximate equal share of the overall number of base stations in 2016 and continue rising [145].

2. Mobile Multi-standard Terminals
Mobile Multi-standard Terminals (MMTs) represent another large market opportunity for SDR. As the number of standards needing to be served [141] by the MMT grows, SDR will at some point provide a cost advantage relative to a conventionally designed MMT. Further it provides opportunities for future mobile wireless users to change and personalize their units by installing additional pieces of waveform software, and upgrade their units as new standards emerge or as standards are updated. More importantly, with the future reconfigurable and cognitive radio networks it will be a necessity for the units to be able to add waveform applications or components dynamically.

MMTs are still almost exclusively using traditional non-SDR designs, utilizing waveform standard specific integrated HW [58] even if the terminals serve a high number of waveform standards (e.g. GSM, EDGE, W-CDMA, HSDPA, Bluetooth, WiFi). A multi-mode mobile phone with ‘softwaredefined modem’ processing up to 2.8 Mbps has been demoed [146]. This is possibly an important milestone in the SDR direction. Technical details about its SW flexibility and powerconsumption are however unknown.

MMTs presently are also characterized by a relatively small amount of dominant manufacturers having a high degree of vertical integration and proprietary solutions, e.g. being responsible both for the hardware platform and the waveform software, and which have an interest in maintaining this business model. There are, however, some signs of interfaces being opened up and value chain restructuring. An obvious observation is the trend of employing third-party operating systems (e.g.

the Symbian OS) allowing third-party user applications to be loaded. This has an effect in making end users accustomed to adding SW applications to their units.

So far, the user demand for field upgrading waveform application software on mobile handsets has been limited, simply due to the fact that the handsets are frequently replaced (the 'handset replacement' model [58], since the market is currently driven also by a lot of other factors than the waveform standards, e.g. improved platform devices such as cameras and displays. Several authors, however, predict a change to the 'handset service upgrade' [58] and 'personalization' [147] model where a 'naked handset' [58] is uploaded to suit the user's needs.

The mainstream MMT evolution into SDR-based design is dependent on several factors, the most important being power consumption and cost, with the cost trade-off being highly dependent on the number of waveform standards that the terminal is intended to serve. Due to these factors being more significant for MMTs than for base stations, the MMTs are predicted to come after the base station development in the transition to SDR design. However, since the MMT market has high price pressure, this implies that as soon as the SDR approach gives a cost advantage, and assuming acceptable power consumption, there will be a very significant drive in this direction.

3. Cognitive Radio
The projected evolution into CR capable MPMBs and MMTs represents a large future market opportunity and driver for SDR technology.

CRs may both provide context-aware services for the user [148] and improve spectrum utilization through dynamic spectrum access [135], [137], [149]. In order to continuously take advantage of spectrum opportunities and adapt to the specific context, CR requires platforms that have fast dynamic reconfiguration abilities. Recommended sources of information on CR and the application of SDR in CR systems are [136], [138].

While the first commercial-domain CR standard is already drafted [150], more advanced CRs are viewed as being further into the future.

4. Other Commercial Domain Opportunities
Commercial Satellite Communications has already been mentioned as a segment that will benefit from SDR, where SDR enables remote upgrades and possibly multiple uses during the lifetime of a satellite [17]. Equipment to be located in remote and poorly accessible locations on earth is another similar opportunity.

The Femtocell or Home Base Station has been put forward as another market segment with great opportunities for SDR [58], [151]. The reconfigurability and flexibility provided by SDR support the multiple bands, multiple standards and simultaneous 'sniffing' functionality needed in the Femtocells [151]. Other mentioned market opportunities include devices for laptops, automobiles, home entertainment and the medical and public safety segments [58].

SECTION VIII.
Conclusions

Although SDR technology has evolved more slowly than anticipated some years ago, there are now many positive signs, the clearest ones being in the form of SDR products entering the market. Several major initiatives, at national and cooperative levels between nations and the industry are paving the way for SDR.

The increasing availability of SCA SW tools and development platforms is contributing to reducing the learning threshold of the SCA and also increase the productivity of SDR development. Developments within Model Driven Design may further increase this productivity.

The SCA eases portability by providing a standard for deploying and managing applications. Even so the portability of SCA-based applications between different platforms is not straightforward. One major issue is the standardization of the APIs between the application and the system devices and services. Although a subset of the needed APIs have been published on the JTRS website, parts of the APIs will be difficult to standardize across domains, for example the security-related APIs. Another major issue is the instruction code compatibility between different processing elements, which at present requires porting efforts in terms of rewriting code to fit the processing elements of the target platform. It is expected in the long term that design in higher abstraction languages will reduce this type of porting effort.

Alternatives to the SCA include OMG's specification, NASA's STRS architecture and the GNU Radio architecture. MOM-based architectures have a potential of becoming alternatives, but the maturity and the acceptance of these specifications have to be demonstrated. For cognitive radio systems, additions to the SCA, such as middleware that supports adaptation, will be beneficial. Such middleware will increase productivity and standardize solutions when making adaptive and cognitive systems.

It is expected that the SCA will remain the dominating architecture in the military sector where waveform application portability and reuse are major priorities, especially through cooperative programmes. On the other hand, a significant portion of designs for the civilian commercial market, where hardware cost is a major factor, are likely to utilize dedicated and proprietary lighter-weight architectures. In a longer ($\sim$ 10 years) perspective, and as hardware cost progressively becomes a smaller part of the total system cost, standardized open architectures are likely to become more popular also on the civilian commercial market.

A fundamental challenge for SDR designs is that of providing sufficient computational performance for the signal processing tasks and within the relevant size weight and power requirements. This is particularly challenging for small handheld units, and for ubiquitous units. Parallel computation enhancements and the rapid evolvement of DSP and FPGA performance help to provide this computational performance. Processing units having multiple SIMD processing elements appear to be very promising for low-power SDR units. Also, as waveforms typically have many common functions, it may be sensible to make parameterized, optimal low-powerconsumption dedicated hardware blocks for these common functions, and run alternative source code on a more general processing element if they do not exist.

The reconfigurability of SDR systems has security challenges as a side effect. One such security challenge is that the system must be protected from loading unauthorized and/or malicious code. Also, the rigidity of conventional security architectures in many ways contrast the desired

flexibility and portability ideally required for SDR. The MILS architecture provides for larger flexibility and easier portability of security related modules, while offering multiple security-levels without the need for multiple sets of HW.

SDR has forced regulators to rethink the certification of radio equipment. While traditional equipment has a fixed number of functional modes and a more or less fixed design that may be fully characterized, SDRs may be SW loaded to function in a large variety of modes and hence may not be tested in every possible mode at the time of the initial certification. Changes in certification rules to deal with SDRs have taken place, but it is likely that as more SDR products approach the market, there will be a further evolvement of these rules.

The multitude of waveform standards and their rapid progress make it beneficial and economical to be able to easily update wireless network infrastructure equipment, such as cellular base stations. Also, base stations are less sensitive to the power consumption of the SDR processing platforms than the mobile devices. Thus SDR has promising potential in commercial wireless network infrastructure equipment.

SDR has the potential to increase the productivity of radio communication development and lower the lifecycle costs of radio communication. This will partly come through a change in the business models in the radio communication industry, allowing a separation into SDR platform providers and third-party SW providers. This again will provide volume benefits for the platforms and lower the threshold for companies entering the market as SW providers, and hence provide further competition in the SDR SW applications area.

SDR will have continued focus as a highly flexible platform to meet the demands from military organizations facing the requirements from network centric and coalitional operations. SDR will also have continued focus as a convenient platform for future cognitive radio networks, enabling more information capacity for a given amount of spectrum and have the ability to adapt on-demand to waveform standards.

ACKNOWLEDGMENT

The author would like to thank Christian Serra at Thales Communications, Marc Adrat at FGAN (the Research Establishment for Applied Science), Jon Olavsson Neset and Stewart Clark at NTNU (Norwegian University of Science and Technology), Audun Jøsang at UniK (University Graduate Center at Kjeller), Frank Eliassen at UiO (University of Oslo) and Torleiv Maseng, Tor Gjertsen, Asgeir Nysæter and Synnøve Eifring at FFI (the Norwegian Defence Research Establishment) for their valuable input and advice. The author would also like to thank the anonymous reviewers for their helpful comments.

Authors

Figures

References

Citations

Keywords

Metrics