Wireless World Research Forum
Working Group 6 White Paper
Element management, flexible air interfaces, SDR

Element Management, Flexible Air Interfaces, SDR

Joint Editor Group:

Vera Stavroulaki, University of Piraeus (UPRC), Greece, veras@unipi.gr
Panagiotis Demestichas, University of Piraeus (UPRC), Greece, pdemest@unipi.gr
Lars Berlemann, RWTH Aachen University, Germany, ber@comnets.rwth-aachen.de
Terence Dodgson, Samsung Electronics, UK, terry.dodgson@samsung.com
Jörg Brakensiek, Nokia, Germany, jorg.brakensiek@nokia.com

Abstract:

In today's wireless world a large number of Radio Access Technology (RAT) standards is available. The recent trend of "wireless beyond the third generation" (B3G) assumes that cellular, Broadband Radio Access Networks (BRAN) / Wireless Local Area Networks (WLAN) and Digital Video Broadcasting (DVB) systems can be co-operating components of a Composite Radio (CR) infrastructure. Through such a CR system, that provides the possibility of co-operation among the various available RATs, users can be directed to the most appropriate one, according to the service area regions, time zones, profile and network performance criteria. In this context, the deployment of CR systems requires technologies that allow terminals and network elements to dynamically select and adapt to the most appropriate RAT (in a transparent manner). The Reconfigurability concept (which is an evolution of "software defined radio") provides such technologies posing essential issues with respect to element management. In this direction, one of the targets of this white paper is to present a concept for a Management and Control System that enables elements to operate in an end-to-end reconfigurability context. The main idea of this concept is a clear separation of the management and the control functions. Hardware abstraction is a research topic widely discussed in the reconfigurability community (e.g. specific RFI in the SDR Forum). The paper addresses the issue of Hardware abstraction in an en-to-end reconfigurable device and presents a possible approach. Some design and integration challenges for reconfigurable systems are also highlighted. The paper concludes with an overview of a verification tool for 4G/ Beyond 3G Systems.

Table of Contents

List of Figures

List of Tables

Abbreviations List

AAA	Authentication, Authorisation and Accounting
ACL	Access Control List
ACL	Asynchronous ConnectionLess
ADC	Analog to Digital Conversion
AEP	Application Environment Profile
AP	Access Point
API	Application Programming Interface
ARQ	Automatic Repeat Request
AS	Access Stratum
ATM	Asynchronous Transfer Mode
BB	Baseband
BF	Beamforming
BIOS	Basic Input Output System
BMM	Bandwidth Management Module
BS	Bearer Services
BS	Base Station
BT_AP	Bluetooth Access Point
BTS	Base Transceiver Station
CALLUM	Combined Analogue Locked Loop Universal Modulator
CAST	Configurable radio with Advanced SW Technology
CDMA	Code Division Multiple Access
CF	Core Framework
CM	Configuration Management
CM	Connection Management
CMM	Configuration Management Module
CN	Core Network
CODEC	CODer/DECoder
COM	Component Object Model
CORBA	Common Object Request Broker Architecture
CPU	Cental Processing Unit
CSI	Channel State Information

CSW	Computer Software Components
DAB	Digital Audio Broadcast
DAC	Digital to Analog Conversion
DGAPS	Design of Generic and Adaptive Protocol Software
DL	Downlink
DLL	Data Link Layer
DNS	Domain Name Service
DSP	Digital Signal Processor
DVB	Digital Video Broadcast
EIR	Equipment Identity Register
ETSI	European Telecommunications Standards Institute
EVM	Error Vector Magnitude
FDD	Frequency Division Duplex
FEC	Forward Error Correction
FLP	Flexible Linearity Profile
FPGA	Field Programmable Gate Array
FTP	File Transfer Protocol
-g	Generic
GGSN	Gateway GPRS Support Node
GIOP	General Inter-ORB Protocol
GLL	Generic Link Layer
GP	Generic Protocol
GPP	General Purpose Processor
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
H/2	HiperLAN 2
HiperLAN/2	High Performance LAN type 2
HLR	Home Location Register
HO	Handover
HRM	Home Reconfiguration Manager
HTTP	HyperText Transport (or Transfer) Protocol
HW	Hardware
IDL	Interface Definition Language

IETF	Internet Engineering Task Force
IF	Intermediate Frequency
IGMP	Internet Group Management Protocol
IIOP	Internet Inter-ORB Protocol
IMD	Intermodulation Distorsion
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
IMT-2000	International Mobile Telecommunications 2000
IP	Internet Protocol
ISA	Instruction Set Architecture
ISDN	Integrated Services Digital Network
ISV	Independent Software Vendor
LAN	Local Area Network
LINC	Linear Amplification with Non-linear Components
LLC	Logical Link Control
LNA	Low Noise Amplifier
LO	Local Oscillator
LR	Location Register
MAC	Medium Access Control
MCM	Modes Convergence Protocol
MDA	Model Driven Architecture
ME	Mobile Equipment
MEMS	Micro ElectroMechanical Systems
MExE	Mobile Execution Environment
MIB	Management Information Base
MIMM	Mode Identification & Monitoring Module
MIMO	Multiple Input - Multiple Output
MIP	Mobile IP
MIPS	Million Instructions Per Second
MISO	Multiple Input - Single Output
MM	Mobility Management
MMS	Multimedia Message Service
MN	Mobile Node.

MNSM	Mode Negotiation and Switching Module
MOBIVAS	downloadable MOBIle Value Added Services
MS	Mobile Station
MSC	Mobile Switching Centre
MT	Mobile Terminal
MU	Multiple User
MVCE	Mobile Virtual Centre of Excellence
NBS	Network Bearer Services
NE	Network Element
NO	Network Operator
OEM	Original Equipment Manufacturer
OMG	Object Management Group
ORB	Object Request Broker
OS	Operating System
OTA	Over the Air
PA	Power Amplifier
PAN	Personal Area Network
PBA	Parametrizable Basic Architecture
PDU	Protocol Data Unit
PHY	Physical Layer
PIM	Platform Independent Model
PLMN	Public Land Mobile Network
PRM	Proxy Reconfiguration Manager
PSM	Platform Specific Model
QoS	Quality of Service
RAN	Radio Access Network
RAT	Radio Access Technology
RCS	Radio Control Server
RF	Radio Frequency
RLC	Radio Link Control
RM	Reconfiguration Manager
RMA	Reconfiguration Management Architecture
RMM	Reconfiguration Management Module

RNC	Radio Network Controller
ROM	Read-Only Memory
RPC	Remote Procedure Call
RRC	Radio Resource Control
RRM	Radio Resource Management
RSMM	Resource System Management Module
RSSI	Receive Signal Strength Indicator
RX	Receive
-s	Specific
SAP	Service Access Point
SAW	Surface Acoustic Waves
SCOUT	Smart user-Centric cOmmUnication environmenT
SDL	System Description Language
SDM	Software Download Module
SDMA	Space Division Multiple Access
SDP	Service Discovery Protocol.
SDR	Software Defined Radio
SDRC	Software Download and Reconfiguration Controller
SDRF	SDR Forum
SDU	Service Data Unit
SEG	Security Gateway
SIM	Subscriber Identity Module
SIR	Signal to Interference Ratio
SOAP	Simple Object Access Protocol
SPRE	Software Download and Profile Repository
SRA	Software Radio Architecture
SRM	Serving Reconfiguration Manager
S-RNC	Serving RNC
SW	Software
TCP	Transport Control Protocol
TDD	Time Division Duplex
TOI	Third Order Intercept
TRSA	Terminal Reconfiguration Serving Area

TRUST	Transparently Reconfigurable UbiquitouS Terrminal
TX	Transmit.
UE	User Equipment
UL	Uplink
UML	Unified Modelling Language
UMTS	Universal Mobile Telecommunication System
UP	User Profile
UPS	User Plane Server
USIM	UMTS Subscriber Identity Module
UTRAN	UMTS Terrestrial Radio Access Network
VCO	Voltage Controlled Oscillator
VHE	Virtual Home Environment
VLR	Visitor Location Register
VM	Virtual Machine
WAN	Wide Area Network
WAP	Wireless Application Protocol
W-CDMA	Wideband Code Division Multiple Access
W-LAN	Wireless LAN
XML	eXtensible Markup Langage

1 Introduction

In today's wireless world a large number of Radio Access Technology (RAT) standards is available. The recent trend of "wireless beyond the third generation" (B3G) assumes that cellular, Broadband Radio Access Networks (BRAN) / Wireless Local Area Networks (WLAN) and Digital Video Broadcasting (DVB) systems can be co-operating components of a Composite Radio (CR) infrastructure. Through such a CR system, that provides the possibility of co-operation among the various available RATs, users can be directed to the most appropriate one, according to the service area regions, time zones, profile and network performance criteria. In this context, the deployment of CR systems requires technologies that allow terminals and network elements to dynamically select and adapt to the most appropriate RAT (in a transparent manner). The Reconfigurability concept (which is an evolution of "software defined radio") provides such technologies posing essential issues with respect to element management. In this direction, one of the targets of this white paper was to present a concept for a Management and Control System that enables elements to operate in an end-to-end reconfigurability context. The main idea of this concept is a clear separation of the management and the control functions.

The paper also introduces a reference model for a multi-mode protocol stack of a flexible, dynamic reconfigurable air-interface for future wireless networks. This future wireless network has the vision of a ubiquitous radio system concept providing wireless access from short-range to wide-area, with one single adaptive system for all envisaged radio environments. It will efficiently adapt to multiple scenarios by using different modes of a common technology basis. The generic protocol stack enables an efficient realization of reconfigurable protocol software as part of a completely reconfigurable wireless communication system.

In the sequel of the paper some issues related to SDR are addressed. Hardware abstraction, which is a research topic widely discussed in the reconfigurability community (e.g. specific RFI in the SDR Forum), is discussed. The paper presents a possible hardware abstraction approach in an en-to-end reconfigurable device. Some design and integration challenges for reconfigurable systems are also highlighted. The paper concludes with an overview of a verification tool for 4G/ Beyond 3G Systems, the PRAGA platform.

2 Element Management

The aim of this section is to present a concept for a Management and Control System that enables elements to operate in an end-to-end reconfigurability context. The main idea of this concept is a clear separation of the management and the control functions. This section mainly focuses on reconfigurable terminals.

2.1 *Initial Functional Description of the Equipment Management Architecture*

A high level view of the management and control of equipment in an end-to-end reconfigurability context is depicted in Figure 1.

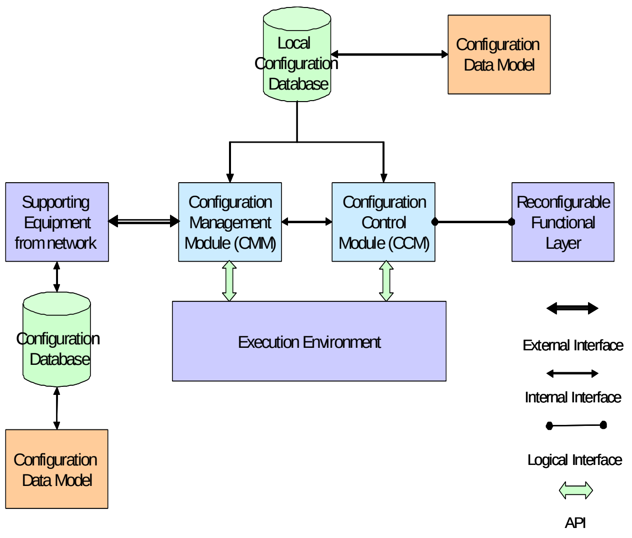

Local Configuration Database
Configuration Data Model
Supporting Equipment from network
Configuration Management Module (CMM)
Configuration Control Module (CCM)
Reconfigurable Functional Layer
Configuration Database
Execution Environment
Configuration Data Model
External Interface
Internal Interface
Logical Interface
API

Figure 1: High Level view of the management and control of equipment in an E2R system

The proposed framework consists of two main modules:

- The Configuration Management Module (CMM), which is a functional entity within the equipment (terminal, base station/access point or network), that manages the reconfiguration processes according to, specified semantic, protocols and configuration data model (which may be stored in distributed configuration data-base system). From the equipment perspective the various CMMs also interact among themselves as well as with supporting equipment entities within the network, through an external (transparent) interface.
- The Configuration Control Module (CCM), which is a supporting entity responsible for the control and supervision of the reconfiguration execution. This is done using specific commands/triggers and functions of a given layer or a given execution environment. Three main layers are considered here: application, protocol stack (L2 – L4) and modem (L1).

Other entities closely related to equipment management are the Execution Environment and the Reconfigurable Protocol Stack framework (or Reconfigurable Functional Layers).

The Execution environment is the means for providing the basic mechanisms required for dynamic reliable and secure change of equipment operation. The execution environment aims to offer a consistent interface to the equipment reconfiguration manager in order to apply the needed reconfiguration actions. For reconfigurable equipment, reconfigurable components need to be used. Such components are programmable processors, reconfigurable logic, parameterized ASICs (offering software control on their parameters). The Execution Environment sits on top on this hardware platform and offers basic mechanisms enabling the exploitation of the reconfigurable hardware components.

The Reconfigurable Protocol Stack Framework is an open protocol stack framework, which can be used to support several RATs with diversified protocols and protocol functions. This implies an architecture that supports dynamic insertion and configuration of different protocol modules in a common manner taking into account the resources and capabilities of the target devices.

The functional entities of the equipment management architecture and other internal and external entities will be described in detail in the following.

2.2 Equipment Management Reconfiguration modules/components

In the context of end-to-end reconfiguration the following scenarios have been identified:

- Device management
- Multi-mode/multi-standard
- Service adaptation
- Adaptation of device algorithms

The next sections describe the entities (modules) of the Equipment Management required for supporting the above-mentioned scenarios.

2.2.1 Reconfigurable Protocol Stack Framework (RPS_FW)

In general, it is known that the wireless terminals and network equipments that are currently available in the market are complex systems that implement communication protocols, for each layer of the OSI reference model, using different platforms (i.e. RTOS). Moreover, it has been recognized that the OSI model enforces a strict partitioning between layers that is not always easily applicable for instance presentation services are required by lower layers to communicate [2]. Furthermore, most of these OSI protocols consist of a set of mandatory (core) and multiple options, with possible inconsistencies. In addition, the most critical core protocol functions have been identified that can be used for Integrated Layer Processing architectures [3].

It is obvious from the above that in most cases the same functionality is used from several layers. For this reason, it is proposed to build the common functions of each layer inside the equipment infrastructure (built-in blocks) using the host operating system and middleware of the platform. These built-in blocks and their implementation comprise the main idea of the Reconfigurable Protocol Stack Framework (RPS_FW). However, in order to re-use and compose the built-in blocks for on the fly constructing and running of the stack, a software component-based architecture is recommended for developing the RPS_FW.

A software component (SWC) is a unit of composition with contractually specified interfaces and explicit context dependencies only. Stating the required interfaces and the acceptable execution environment specifies context dependencies. A software component can be deployed independently and is subject to composition by third parties. Therefore, in addition, this RPS_FW can be extended with deploying, installing and implementing the downloaded software components. These components (Resources), which are embedded inside the equipment, are used for extending protocol implementations and finally support the particular communication system or protocol. The main goals of this framework are first the ability to be extended or/and parameterized in order to support a particular functionality and second the ability to use shared resources.

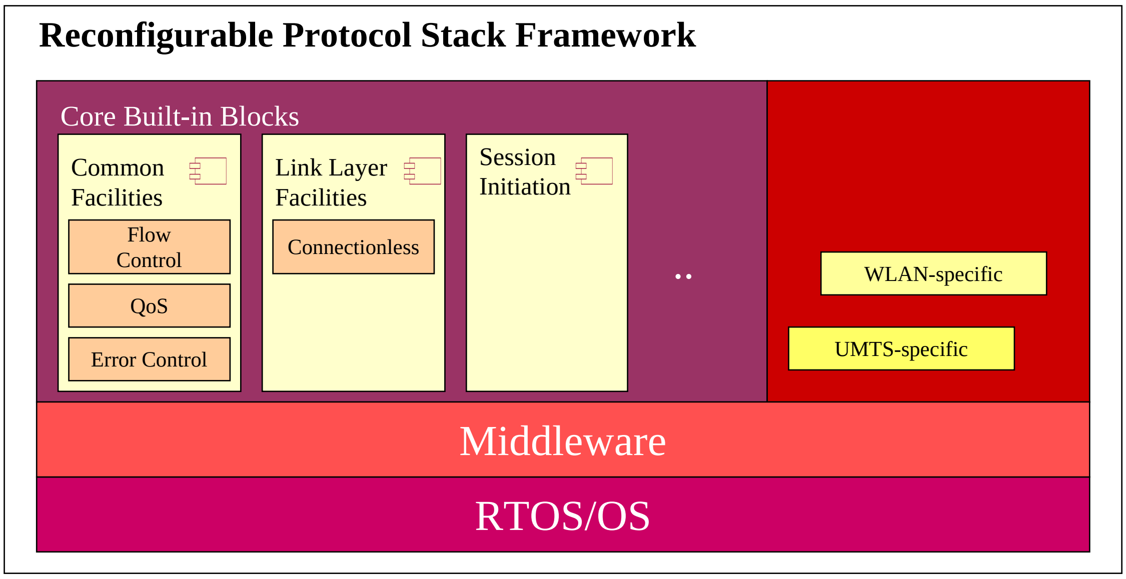


Figure 2: Reconfigurable Protocol Stack Framework (RPS_FW)

There is tight relation between the execution environment and both the protocol stack level and the application level. The application level could be assumed to be the upper layers and specifically the layer to the user space. The main requirements that the RPS_FW tries to satisfy are:

- The system firmware that permits application and protocol entities to use the hardware resources (OS, Virtual Machines, Middleware, low-level drivers etc.);
- The reconfigurable protocol stack framework which is software that offers basic support for the implementation of flexible protocol stacks and provides the needed reconfiguration mechanisms;
- The protocol stack consisting of the logic that implements the protocol stack functionality at the different layers;
- The components and modules must be uniquely defined for each protocol function at each layer. For the construction of the reconfiguration environment, a flexible execution environment must be provided which includes SW load modules, real-time schedules, communication maps, interfaces between SW and HW modules.
- The operation environment that supports the SW downloading procedure. Software is formed with a component-based fashion [4].

2.2.2 Configuration Management Module (CMM) Entities

The Configuration Management Module (CMM) is responsible for managing the distributed controllers, which will initiate, coordinate and perform the different reconfiguration functions such as monitoring and discovery, software download, mode selection and switching (multimode/multi standard), security. The CMM consists of the following functional entities:

- *"Interfaces with the Network Support Services" (CMM_IfNss)*. This functional entity is responsible for network-initiated re-configuration and other services while the terminal is in on-line idle mode. This module will receive messages from the network, and it may activate other modules to start configuration or other actions in the terminal. The network-supported services can issue reconfiguration commands to the CMM. Moreover, supporting information can be exchanged between the network support services and the CMM, so as to consolidate on the best reconfiguration decisions.

- *"Monitoring and discovery" (CMM_MD).* The role of this entity is to identify the available networks in a certain area and to monitor their status. It acquires information on the context in the environment of the device and takes into account data from multiple CCMs.
- *"Negotiation and Selection" (CMM_NS).* This functionality is targeted for the negotiation of offers with the various available networks. It selects the most appropriate available network taking into account information such as the user and terminal profile and the offers negotiated with the networks. The user profile specifies services, QoS levels, cost levels etc. The terminal profile specifies the capabilities, configurations present at memory, etc. The network offers specify the services offered, the QoS levels supported, cost information etc. The goal of the negotiation functionality is the refinement of parameters (e.g. related to network offers). For this purpose, standard negotiation protocols need to be adopted and customized.
- *"Configuration downloads" (CMM_Dwnld).* This functional entity provides the capability to perform downloads of the different components that may be required for the reconfiguration process. In other words, it undertakes the management of the downloading procedure. The downloaded information could be a whole protocol function in form of components or a parameter of a component. The downloading procedure encompasses stand-alone or distributed mechanisms that are required in order to communicate with the software provider.
- *"Profiles" (CMM_Prof).* This functional entity provides configuration profiles information on applications, user classes, equipment classes/capabilities and configuration data models. The CMM_Prof is able to compare profile of the current configuration and the proposed future configuration in order to recognize the absence of a particular function needed for activating the target communication system, protocol stack or application for each level respectively.
- *"Security" (CMM_Sec).* This module supports the security functions required during the reconfiguration process within the different layers.
- *"Decision Making and Policy enforcement" (CMM_DMP).* This entity communicates with the Reconfiguration Management Plane (RMP) entity. It interacts with the Reconfiguration Management Plane (RMP) to provide information and mechanisms for the decision of reconfiguration actions. It enables the provision of reconfiguration policies and actions throughout the network and locally in the network nodes and equipment. The interface between the RMP and the CMM_DMP mainly supports context and policy management procedures. Policy-based mechanisms and procedures are being implemented and performed by entities that are dedicated to mode selection and switching dovetail the mechanisms of RMP.
- *"Reconfiguration Installation"* (CMM_Instl). This function is to provide, by interacting with the CMM_Dwnld and CMM_DMP functional entities the means for configuration representation and configuration deployment, which involves configuration download, validation, installation and switching.
- "*Event handler" (CMM_Evnt).* This entity enables the coordination of the different reconfiguration triggers, which activates scheduling and implementation procedures through the corresponding CCM according to the target reconfiguration. Events may be received from within the terminal (e.g. discovering new network: CMM_MD, or completing configuration software download: CMM_Dwnld), or externally (e.g. receiving a message via CMM_IfNss).

2.2.3 Configuration Control Module (CCM) Entities

The Configuration Control Module (CCM) initiates, coordinates and performs the different reconfiguration functions. The CCM consists of the following functional entities:

- *Configuration Control Module – Application Layer (CCM_AP),* that provides the interface between the CMM and the application layer.

- *Configuration Control Module – Protocol Stack layer (CCM_PS)*, that provides the interface between the CMM and the protocol stack of a protocol suite like TCP/IP (e.g., layer 3/4, namely TCP/IP) or/and of a communication standard (e.g. layer 2/3 of UMTS/WLAN standard). This entity enables the addition of a complete new protocol function to the equipment and the parallel operation of this new protocol function with the existing functions.
- *Configuration Control Module– Reconfigurable Modem (CCM_RM)*, that provides the interface between the CMM and the physical layer resources.

2.2.3.1 CCM_AP functionality

This entity performs and activates the reconfiguration in the application layer. At the application layer communication partners are identified, quality of service is identified, user authentication and privacy are considered, and any constraints on data syntax are identified. Everything at this layer is application-specific.

Due to its nature, reconfiguration must be applied with software orientation. This orientation will enable the application layer to trigger reconfiguration to the above laying applications. Such an example could be the request from the top layer to an application in order to send data with lower bit rate due to network congestion. The aforementioned scenario can be developed only with the addition of extra capabilities to the application layer, which is considered as the main layer inside the application level.

2.2.3.2 CCM_PS functionality

This entity enables a complete new protocol function to be added to the equipment and operate with the existing protocol functions. For the incorporation procedure a deployment function is required while for the instantiation a scheduling and context switching function are implemented. Consequently, the configuration control module for the protocol stack level consists of the following functions:

2.2.3.2.1 Deployment Function

This is responsible of deploying the new components into the RPS_FW. The deployment function indicates the component"s composition graph that is required for system stack construction and implementation. The deployment procedure could be triggered by the completion of the downloading procedure through the event handler. The deployment function includes some of the following procedures: configuration, releasing, installation, updating, adapting of each component.

2.2.3.2.2 Scheduling Function

This function is related to the protocol stack since the actual and final outcome of this process must be the execution of the sequence task of a particular stack. The schedule consists of the ordering constraints and rules required for the perfect operation of the specific system. This procedure can be called a “communication bootstrap” for a particular protocol stack. The scheduling procedure could be triggered by the completion of the deployment procedure through the event handler.

2.2.3.2.3 Context switching

In the RPS_FW, *Context* encompasses the operations of all layers that comprise the protocol stack. Here, context switching is considered as a function only between several protocol stack instances and not between protocol functions. The switching process could be performed using threads in the same process.

Context switching to another thread in the same process is much cheaper than switching to a thread in another process. Threads can share resources easier because they live in the same address space.

2.2.3.3 CCM_RM functionality

The CCM_RM receives the physical layer related configuration issues from the CMM. The physical layer architecture has to provide the environment, which is needed to implement the functionality required by the CCM_RM.

2.2.3.3.1 Hardware resources

At this level, the hardware resources appear as elements with different type of intelligence and programmability/ configurability. Beside instruction programmable elements (GPP, DSP), configurable logic (FPGA), special parameterised accelerators (e.g. ASICs) also communication elements (switches, bus control logic, multi-port memory etc.) belong to this category. The hardware components of the RF-frontend (oscillators, converters, filters, mixers, etc.) are not directly under control of the CCM_RM even if they are adjustable by parameters by according interfaces (registers, I2C, SPI…). In place of that the RF-frontend behaves like one of the configurable elements listed above.

The single elements have to be converted into configuration execution modules (CEM) implementing a certain level of configuration functionality, which fits to the resource interface provided by the CCM_RM. By configuration, such elements are combined into functional modules implementing a specific functionality (e.g. Down-Conversion, Modulation, Decoding). This is also valid for the communication resources connecting the processing elements to guarantee the required functionality and maintain the required data throughput.

2.2.3.3.2 Operational Software

The operational software entity takes a substantial role in CCM_RM to manage different levels of abstraction, and the temporal scheduling of hardware and software resources that requires appropriate operational software support. As the CCM_RM is a logical element, it is the challenge of an operational software module (OSM) to manage all these processes, which are required to maintain the reconfigurability of the whole system and at the same time to guarantee the functionality of the underlying data processing system.

The execution platform of the OSM consists of above said hardware resources, which are connected by a variety of communication elements. The OSM may incorporate a real-time operating system (RTOS) with some control interface to real-time mechanisms of other programmable processing elements (ISAs) that requisites allocation and management of processing resources (e.g. execution time, priority and scheduling policy) for tasks to be scheduled on them. Similarly, methods are provided for managing the associated memory resources.

Supporting of run-time re-configuration is a critical function of the OSM. Loading and unloading of software modules on programmable processing elements (PPEs) also be possible. The OSM incorporates appropriate loaders for the target PPEs, while providing a common logical interface to the CCM_RM. Drivers and loaders to load and configure various re-configurable and parameterizable hardware resources are implemented in a hardware abstraction layer. Other services for support of development and execution such as non-volatile storage, file system, logging service, diagnostics etc. may additionally be provided.

2.2.4 Security Architecture

2.2.4.1 Description of entities

As shown in Figure 3 the security architecture consists of the following entities:

- *"Installation manager"* exists within the CMM and manages the sequencing of payload installation. It also records progress information as the installation proceeds to allow recovery or rollback after various installation errors.
- *"Recovery Manager"* is invoked after an installation error has occurred, and its functions include determining the appropriate action to take to recover from an installation error, restoring the context of the install process at the point of the error, and instigating the recovery or rollback actions.
- *"Security Manager"* communicates to CMM (Installation recovery manager) regarding the correct data delivery, error correction and recovery procedures.
- *"Security HW Configuration"* may exist as a specialized entity in CCM and handles all security related hardware configuration (part of WP4).
- *"Security SW Configuration"* is responsible for configuring algorithms, which exist in software.
- *"RTOS"* will accommodate secure procedures to ensure integrity and specialized bootstrap methods.

The "Security Manager" could also enforce flexible authorization strategies over resources located within the terminal. "Security Manager" takes the security tasks from the CMM. "Security Manager" connects to and is coordinated by the CMM. "Security Manager" interacts with, "Security SW Configuration" and "Secured Memory".

A "Security Manager" within the terminal provides a number of security related tasks to ensure only authorised access to the reconfiguration management system. This Security Manager is defined as an independent process; it requires its own processor and memory space.

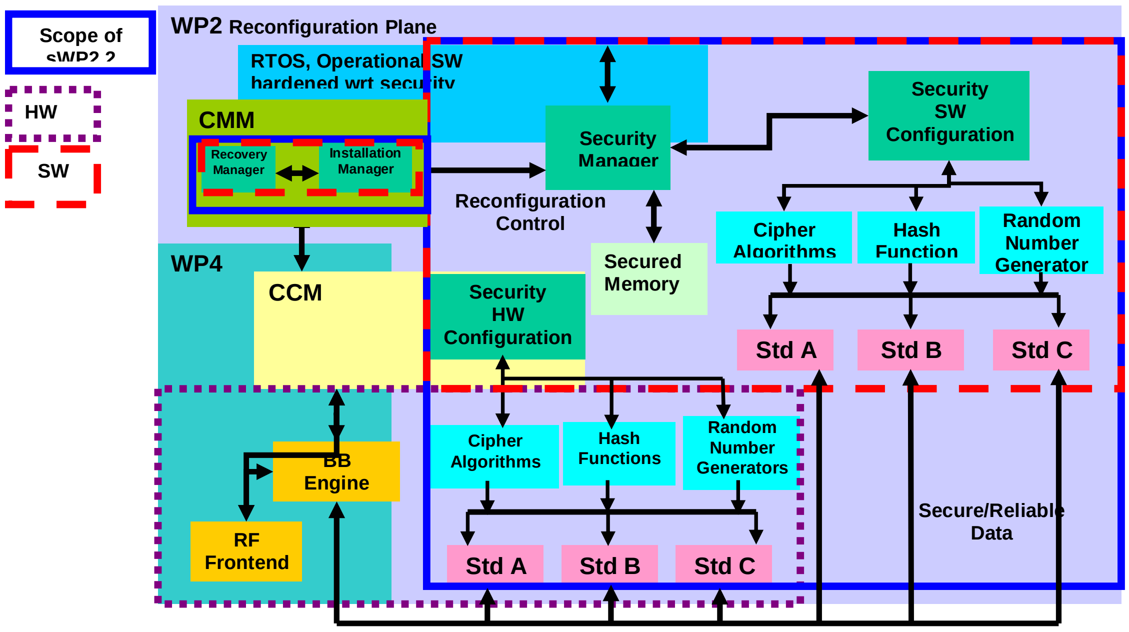


Figure 3: Security Architecture

The Security Manager is responsible for safeguarding the access to the functions of the reconfiguration management part within the terminal and also to prevent attempts of fraudulent access from the outside world. Additionally, it provides the required security related information to the network, stores the security information of the terminal (i.e. public keys and private keys) and the necessary external encryption keys. The Security Manager is responsible to establish secure connections between the terminal and the network.

The Security Managers" internal architecture, consists of an "Access Manager Entity" (responsible for the establishment of secure connections between terminal and network and also for the processing of messages between Configuration Manager, network and the other functional entities within the Security Manager). The Encryption & Decryption Factory (EDF) implements the security features presented by Sec_SW_CNF and encrypts both messages and reconfiguration software, before any transmission between terminal and network takes place. For messages and software transmissions originated within the network, the EDF decrypts the streams and passes the data to CMM. The Security Manager in general performs following functions:

- Establishment of secure connections with the network;
- Encryption and decryption of messages and data transfer;
- Routing of reconfiguration messages and software to CMM;
- Communication with Sec_SW_CNF for security algorithms reconfiguration;

All activities of the Security Manager are based on these basic tasks.

2.2.4.2 Security domains

It is desirable that reconfiguration operations respect the contract between the user of the reconfigurable equipment and its operator. Depending on this contract, only a subset of reconfiguration capabilities may be granted to the user. Security mechanisms are also meant to insure that only those reconfiguration operations which have been granted for a given user are authorized.

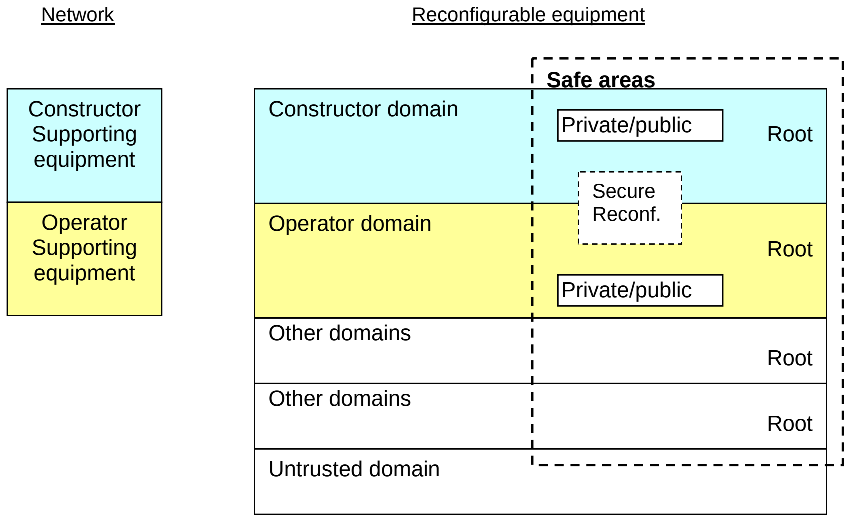


Figure 4: Security Domains

For this purpose, it is necessary that at least 2 security domains are involved in reconfiguration operations: the security domain of the constructor of the reconfigurable equipment, and the one of the operator which maintains the reconfigurable equipment (on behalf of the subscriber). Only the constructor is able to guarantee the sanity of most reconfiguration content. On the other hand, the operator should operate a fleet of devices in accordance to contracts with subscribers.

The security domains within the equipment are represented in Figure 4.

Each domain in the reconfigurable equipment contains a public key root which is used as the last authority of authentication chains to check signatures. Secure configuration mechanisms are assumed which prevent any use of reconfiguration content if it is not checked against both constructor domain and operator domain.

Each domain the reconfigurable equipment also owns a pair of private/public key on behalf of the device, which will be used for authentication of the device by the supporting equipments related to the corresponding domain.

2.2.5 Execution Environment Architecture

As was introduced earlier, the execution environment is the means for providing the basic mechanisms required for dynamic reliable and secure change of equipment operation. The execution environment aims to offer a consistent interface to the equipment reconfiguration manager in order to apply the needed reconfiguration actions. For reconfigurable equipment, reconfigurable components need to be used. Such components are programmable processors, reconfigurable logic, parameterized ASICs (offering software control on their parameters). The Execution Environment sits on top on this hardware platform and offers basic mechanisms enabling the exploitation of the reconfigurable hardware components.

The goal of the flexible protocol stacks is to provide an open protocol stack framework that is extensible and can be used for supporting different radio access network technologies with different protocols and protocol features. Therefore, the underlying execution environment (ExENV) is subject to several design constraints, described as follows:

- *Flexibility:* The ExENV needs to support multiple wireless standards, evolving standards and new applications.
- *High performance*: Is required by process intensive and latency sensitive protocol operations and multimedia processing. For example, some real-time audio and video tasks require the ExENV have sufficient performance to satisfy the demanding QoS constraints.
- *Power-efficient*: Means to utilise the limited amount of energy in the battery appropriately for the performance requirements. Dynamic power management comprises techniques that assign tasks to the most energy-efficient devices available and that force other unused system components into their power-down modes or even shut them off when appropriate.

A hybrid execution environment combines various hardware components (ExHW) for example general-purpose processors (GPPs), field programmable gate arrays (FPGAs) and application specific integrated circuits (ASICs) as well as various software environments (ExSW) for example Virtual Machines (VM), Common Language Run-time (CLR), Operating System (OS) which can be distributed across multiple ExHW or confined to individual ExHW components.

GPPs execute applications in an instruction set architecture, which provide high flexibility. FPGAs use SRAM cells to control the functionality of logic and I/O blocks as well as routing, and can be reprogrammed in-circuit arbitrarily often by downloading bitstream of configuration data to the device, which can exploit the parallelism in algorithms better than GPPs. ASICs provide optimised solution in power consumption, speed, and circuit area, but the development of ASICs is expensive in time, manpower and cost. As a result, it"s believed a heterogeneous organisation should produce an execution environment with the advantage of all the resources.

The execution environment hardware architecture is shown in Figure 5. In the architecture, every execution environment hardware module (ExHW) is connected to the Control Bus for exchanging control signals. The ExHW_LocalCtrl locates in each ExHW and schedules platform independent module interaction. The bus arbitrator is to organise the traffic on the bus. The bus bridge module enables communication between share memory and the external execution environment, e.g. data and commands can be passed via the bus bridge across distributed ExENVs. The ExESW can support interaction between software running on different ExHW by utilising the shared memory. For example this could support protocol software module interaction, control command passing between CMM and CCM components or loading of configuration data or software components from storage (non-volatile shared memory).

An additional Data Bus can be used for simultaneous inter-module communications. The shared memory can be accessed by all ExHWs via the data bus. Therefore, an ExEHW can transfer data via the Data Bus to the share memory and concurrently the bus bridge transfers data to external world.

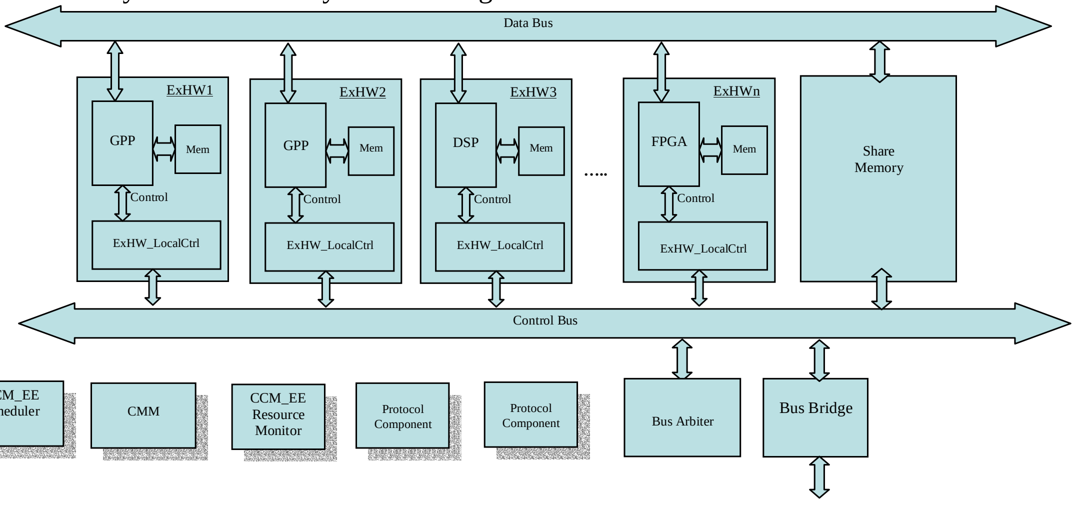


Figure 5: Execution Environment architecture

Within the ExENV (which is likely to consist of a heterogeneous combination of ExHW and ExSW components), the software modules (which could be core software modules such as CMM or CCMs or protocol stack modules) need to be mapped and run on ExSW and ExHWs and different instances of these modules can be initiated. A scheduler instance of CCM_EE schedules the reconfiguration procedure in the execution environment, e.g. installation, deletion, execution and suspension etc. The preparation of the ExEnv includes both the preparation of the ExHW and the configuration of the necessary ExSW (OS, virtual machine, common language run-time or could even include the component support frameworks). There could be a Virtual Operating System (VOS) to abstract away from the underlying specific OS (or multiple OS in a heterogeneous ExEnv consisting of multiple ExHW) which may be COTS based or even proprietary. The CCM_EE must also configure and control the mechanism for interaction between the ExHWs and ExSW, which may involve the use of specific Hardware Abstraction Layers (HAL) that are designed to permit ExEHW component from different vendors to be easily integrated into a heterogeneous ExEnv.

Another instance of CCM_EE is Resource Monitor, which collects the ExHWs statistics data periodically, e.g. memory, power, supply voltage, clock frequency, available FPGA area, Bus load, etc. Configuration Management Module (CMM) makes the reconfiguration decision based on the monitoring data. The reconfiguration commands can be triggered by the CMM internally or external triggers from the network interface.

2.3 Reconfiguration procedures/mechanisms

2.3.1 Internal Mechanisms

This subsection presents the need for terminal reconfiguration management, with reference to some internal procedures/mechanisms that correspond to certain functional entities of the Configuration Management Module.

2.3.1.1 Extension in protocol functionality or/and changes in internal protocol parameters

TCP adaptation

According to the nature of the handover (vertical or horizontal) and the new access network conditions, the TCP parameters to modify may be different. Potentially, all the possible TCP parameters should be modifiable:

- srtt: the smoothed round trip, indicating the mean time a segment lasts in the network,
- ato: the acknowledgement timeout,
- rto: the retransmission timeout,
- cwnd: the congestion window,
- ssthresh: the slow start threshold,
- snd.wnd: the advertised window size, i.e. the amount of data that can be sent,
- rcv.wnd: the received window size, i.e. the amount of data that can be received,
- snd.nxt: the next sequence number to be sent,
- snd.una: the first unacknowledged byte.

In the framework of end-to-end reconfigurability, the most common parameters to be dynamically adapted will probably be the rto, the cwnd and the ssthresh. However, depending on the TCP stack implementation, some of the TCP parameters will be modified for the running sessions, i.e. the running sessions parameters will be updated in real-time, and others will be modified only for the new sessions, i.e. only for the sessions launched after the handover completion.

Besides, in some cases, the CMM_NS (see Figure 20) may decide to give up the TCP adaptation, considering that the new TCP parameters values do not represent a significant change and will not impact the TCP sessions behaviour.

Note that this change in internal TCP parameters should be triggered by both technology and resource availability changes, thanks to a handover notification coming from a specific entity, that is called here a TMM (Terminal Mobility Manager, defined by WP3), in charge of managing the handovers.

2.3.1.2 Functional description and architecture of the reconfiguration procedure

This functional description contains the generic procedure of download, configuration and implementation of a reconfiguration process. It defines the mechanism of delivering the reconfiguration context, storing, managing & uploading the context to the HW platform. It also represents the implementation, alteration and executing of the reconfiguration procedures and module control communication [6].

If a trigger for reconfiguration is passed to the CMM, the CMM will discover whether the software and policies required for an intended reconfiguration are available in the CMM_Dwnld and CMM_Prof, respectively. If policies and software are available, a further request will be dispatched to the CMM_DMP, to initiate the generation of a new terminal profile. In response, the CMM_DMP will request the profiles and necessary software-policies from the CMM_Prof. Upon receipt of these policies, the CMM_DMP generates the new terminal profile. The next step for the CMM_DMP is to announce the generation of the new terminal profile to the CMM and to upload it to the network for a verification configuration procedure (VCP). The profile is sent to network via the secure connection. After receiving the confirmation from the network about completion of a successful VCP, the CMM_DMP forwards a notification to the CMM, that the reconfiguration procedure may proceed. The message issued by the CMM_DMP contains information about the required changes in the configuration. Using the information about required changes, the Configuration Manager generates the new CCM(s) and consequently the specified reconfigurable modules (i.e. installed and controlled by the CCM(s)). Upon completion of the Reconfigurable Module(s) installation, the CMM requests an update of the registration of the new terminal profile. Therefore, the CMM_DMP dispatches a registration request to the network. After completion of the (terminal-configuration) registration, the network notifies the CMM_DMP, which then forwards the response to the CMM. At the end of any reconfiguration procedure, the CMM requests the Sec_Mng to terminate the secure connection between terminal and network.

Similar to the previous case, if a request for reconfiguration is submitted by the CCM and the software module to be installed is known, the CMM will issue a request whether the source code is available within the CMM_Dwnld. There are two possible answers from the CMM_Dwnld either the software is available or it is not available. The previous sequence presented the case when the software is available, but in case the required software module is not available within the CMM_Dwnld storage, a download from the network resident software server will be required. Therefore, the CMM issues a request to the CMM_Dwnld to perform software download sequence. Furthermore, the CMM requests to the Sec_Mgr to establish a secure connection to the next available network server. After the connection is established and a confirmation is received by the CMM, the CMM passes the handle of the connection to CMM_Dwnld. The CMM_Dwnld then dispatches a software download request via secure connection, to the network. The request carries a number of parameters specifying the software type and location (i.e. URL, IP address, FTP site etc.). After completion of the download negotiation, the actual download path leads from the software store to the CMM_Dwnld via the Sec_Mgr. Once the last packet is received from the network, the CMM_Dwnld notifies the Configuration Manager about accomplishment of the software download. A similar sequence is followed for the download of the policies, in which case however the download will be initiated by the CMM_Prof. After completing the download of the policies, the CMM_Prof will also inform the CMM that all policies are available and the reconfiguration procedure may proceed. Once all policies and software modules required are available, the procedure follows the same sequence as the one described above.

This case describes one of the core parts of a reconfiguration procedure, with a particular focus on the mechanisms of the terminal profile handler and the CCM. The sequence depicted assumes that the preliminary signalling has been completed and the CMM now requires the actual generation of a new terminal profile and implementation of the subsequent steps of the reconfiguration procedure. Configuration and software policies are then requested from the CMM_Prof and interpreted by the CMM_DMP, which in turn generates and compiles the source code for the new profile. In case the compilation failed, this process needs to be repeated or the reconfiguration procedure abandoned. After successful completion, the new profile then becomes uploaded to the network where its validity and correctness will be evaluated during VCP. Assuming this evaluation proves the validity of the file, the network issues the permission (to the terminal) to implement the new configuration. If the validation failed, the terminal profile generation process must either be repeated or the reconfiguration sequence abandoned.

During the validation process, the CMM_DMP still retains the old terminal profile and awaits the acknowledgement about the validation from the network. After that, the CMM_DMP"s comparisons block extracts the differences between the old and the new tag-file and forwards it to the CMM together with a confirmation about the successfully performed (or, if necessary, failed) VCP. The next step in the reconfiguration procedure is then the implementation of the required CCMs (i.e. they are instantiated by the CCM).

The scenario, when configuration of the software radio platform takes place during system initialization (i.e. boot-time), offers a slightly different case for the definition of the reconfiguration terminal. A boot sequence provides distinctive differences to the afore-described reconfiguration procedures. In the beginning, after switching on the terminal, the initialisation of the different reconfiguration management modules (i.e. their processes) is taking place. Each of these processes initialises its main thread and passes a handle to the CMM. After this initialisation procedure is finished, the CMM triggers a configuration sequence by dispatching requests to the CMM_Dwnld and the CMM_Prof to check the availability of software and policies, respectively. The procedure is required to ensure that policies and the software are in the storage places (although in this case this is rather a formality, because when the terminal is in the process of turning off it stores its last state, including the Policies and Software in CMM_Prof and CMM_Dwnld, respectively), this however is not the case when the terminal is in a restricted (manufactures) state (i.e., the manufacture may need to include the initial policies for terminal be able to start at very first time). The configuration procedure continues with the activation of the profile, which, in this particular case, does not require a VCP. The rational for omitting the VCP is that the intended configuration has already been approved by the network in the course of the previous reconfiguration procedure. However, this is not the case when the terminal changes the RAT then the complete reconfiguration of the terminal is needed so as a follow up the VCP will be required. The Configuration Manager then creates the required number of CCMs and starts their initialisation. Once a required CCM is ready and functional, it confirms its availability to the Configuration Manager, which then announces that the terminal is readily configured.

In this particular scenario, when the terminal boots up to a previously known configuration, there are some parts of the reconfiguration sequence which can be omitted (i.e. due to the availability of policies and software, and the already validated profile).

This reconfiguration procedure, compared with the previously described scenarios, relies on the condition that the terminal has its main modules already installed but there is no trace of any previous configuration (i.e. no validated profile). It may be assumed that this case will only occur in the manufacturer stage, when the required minimum radio modules are to be installed. For this reason, the first active reconfiguration procedure of the terminal has a minimum or no radio stack available. A possible way to install modules would be through wired connections and direct installation of the software modules and policies (in particular the profile and software policies) in CMM_Dwnld and CMM_Prof, respectively. After this, the terminal needs to be reset and to reboot. Using the (then) already installed software and policies, the CMM will install the minimum possible terminal configuration, which suffices to provide a basic configuration of the reconfigurable part and enables future reconfigurations. The scenario starts the reconfiguration process with a mutual authentication procedure between the terminal Sec_Mng and the vendor"s software server (the PC, sim card etc.). The authentication procedure is performed in a similar manner as the sequence described in 3.2.2.6. After completion of the authentication, the CMM receives a request to install an initial set of configuration software. The CMM responds once the system is ready to pursue the procedure and to install the required software, the CMM also dispatches a notification to both CMM_Dwnld and CMM_Prof to inform them about the procedure. In addition to the notification, similar to the previously described download cases, the CMM starts to request the availability or download of software and policies from the CMM_Dwnld and CMM_Prof, respectively. These two procedures can be performed concurrently and may last until the required data is completely received and stored. The sequences may be repeated, like in the previous cases, until both entities have to respond to the CMM that software (policies) download is completed. The difference in the sequence is that the course of actions finishes as soon as the downloads are completed i.e. no reconfiguration procedure is required or followed. Then the reconfiguration can be performed following the boot procedure. Reconfigurable terminals require this initial installation of a radio configuration; otherwise the terminal will not be able to connect to an air interface.

2.3.1.3 Software Updates CMM Installation

After the downloading process the installation process takes place (Figure 6). The deployment is related to installation process since downloaded code must be mounted in the RPS_FW in order to the instantiation process that follows the installation being with the appropriate manner. The installation information must be preserved inside a file including information about where the components must be located and what the interfaces between components are to allow linking with the existing components. For this reason, the downloaded code must reside in local memory, at least temporarily.

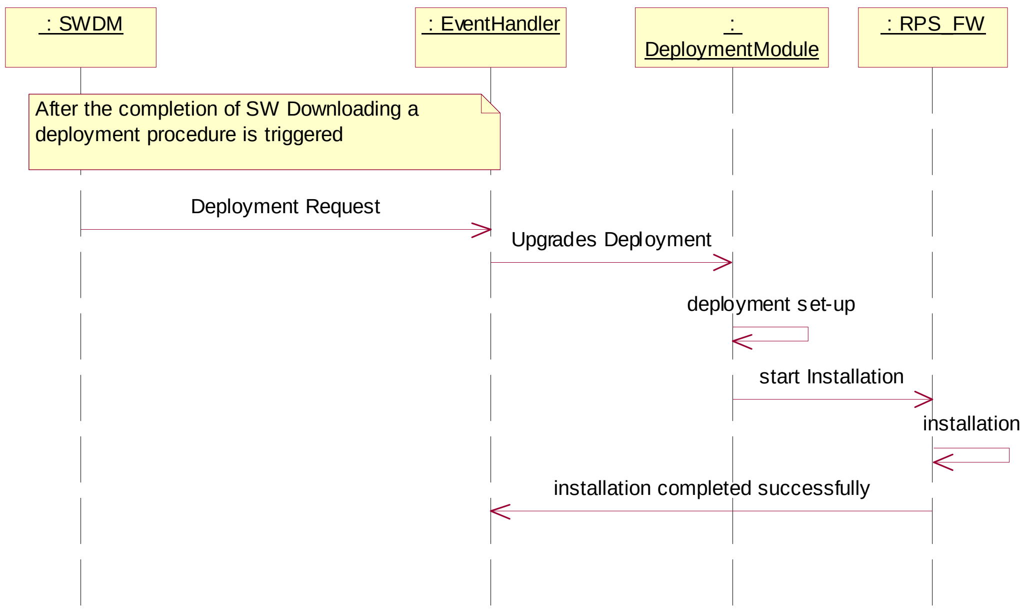


Figure 6: Software Updates Installation

2.3.1.4 Scheduling of protocol stack reconfiguration

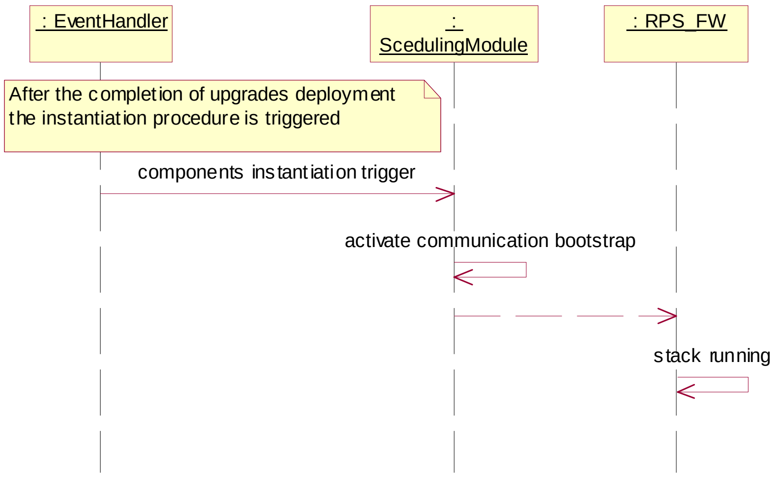

: EventHandler
: ScedulingModule
: RPS_FW
After the completion of upgrades deployment the instantiation procedure is triggered
components instantiation trigger
activate communication bootstrap
stack running

Figure 7: Running the schedule of protocol suite tasks

Using the built-in components residing inside the RPS_FW a communication bootstrap must be performed. The communication bootstrap is the procedure that uses the schedule of the protocol tasks for running the particular stack of the protocol suite. The stack is running since the components have been composed and instantiated.

2.3.2 External mechanisms

This subsection presents the need for terminal reconfiguration management, with reference to some external procedures/mechanisms that correspond to certain functional entities of the Configuration Management Module.

2.3.2.1 Monitoring and discovery

A terminal, in particular should be constantly monitoring the environment. This procedure is an essential entity foreseen for the CMM, as in the terminal centric scenario, the need for reconfiguration is imposed by the terminal, and thus by the CMM of the reconfigurable terminal. This need would become apparent, when the terminal “comes to realize” that its operation parameters could be better, if it would use an alternate RAT instead.

So, the purpose of the “Monitoring and Discovery” procedure is the identification (within the monitoring range of the terminal) of an alternate RAT (other than the one operating so far), with better offers, in terms of better circumstances of coverage, QoS, etc.

Therefore, the CMM should include a functional entity that implements this procedure. This entity (CMM_MD) should be capable of interacting with external to the whole management module entities, such as the network support functions, through the appropriate interfaces. Furthermore, it should also be capable of communicating and exchanging monitoring with the rest of the internal entities, through the Configuration Control Modules (CCMs).

2.3.2.2 Negotiation and Selection

After the discovery of alternative RAT choices, the terminal should be capable of interacting with these RATs, in order to negotiate in terms of quality and cost factors. The negotiation phase is of course followed by the selection phase, during which the terminal selects the best reconfiguration pattern.

The above necessitate the existence of a "Negotiation and selection" functional entity within the CMM, in order to realize this procedure. The CMM_NS functional entity should interact with the network support functions, in order to exchange the necessary negotiation information, as well as with the rest of the, internal to the equipment, entities.

2.3.2.3 Secure diagnostic

In this case, a diagnostic on the reconfigurable equipment is initiated by the operator supporting equipment, and driven by the constructor supporting equipment. Though this operation is secure, no preliminary establishment of a secure channel is assumed. On the contrary, each transaction includes its own elements (such as signatures) for insuring security requirements.

The sequence of message exchanges is described below.

"G":

The operator supporting equipment finds out that some diagnostic driven by the constructor of the reconfigurable equipment should be performed. A list of possible diagnostics is already established between this operator and the constructor of the reconfigurable equipment. The operator supporting equipment builds a message containing a diagnostic ID which is a pointer into this list.

It appends to this diagnostic ID the public key for this reconfigurable equipment which pertains to its constructor domain. It also appends a nonce value (as an anti replay measure), and it may encrypt the result by using the public key of the constructor equipment. Finally it signs the total by using its own private key.

"H":

Upon receiving message "G", the constructor supporting equipment checks the operator signature (against its own certificates and trusted entities). If necessary, it decrypts the message and retrieves its content. Then the constructor supporting equipment builds a detailed diagnostic request specifying to the recipient reconfigurable equipment which measurements and test it should perform. It appends the nonce as just received from the operator, and it signs the resulting message by using its own private key. The resulting message is forwarded to the operator equipment.

"I":

Upon receiving H, the operator equipment checks the constructor signature, and if OK, it appends an message counter and an equipment identifier, and then it signs the resulting message, in order to obtain message "I" which is forwarded to the reconfigurable equipment.

"J":

Upon receiving message "I", the reconfigurable equipment does the following:

- It checks the operator signature against its operator domain,
- (if OK) it checks that the received equipment identifier matches its own identifier,
- (if OK) it checks that the received message counter is greater than the value received on last similar message,

- (if OK) it checks the constructor signature according its own constructor domain,
- (if OK) it retrieves the detailed diagnostic request which was built by the constructor supporting equipment,
- It performs any measurements and test which are required, and it builds a corresponding report.
- Optionally, it encrypts this report by using its private key pertaining to its constructor domain,
- It appends the received nonce,
- It signs the resulting message by using its private key pertaining to the constructor domain, so that it obtains message J which is forwarded to the operator supporting equipment.

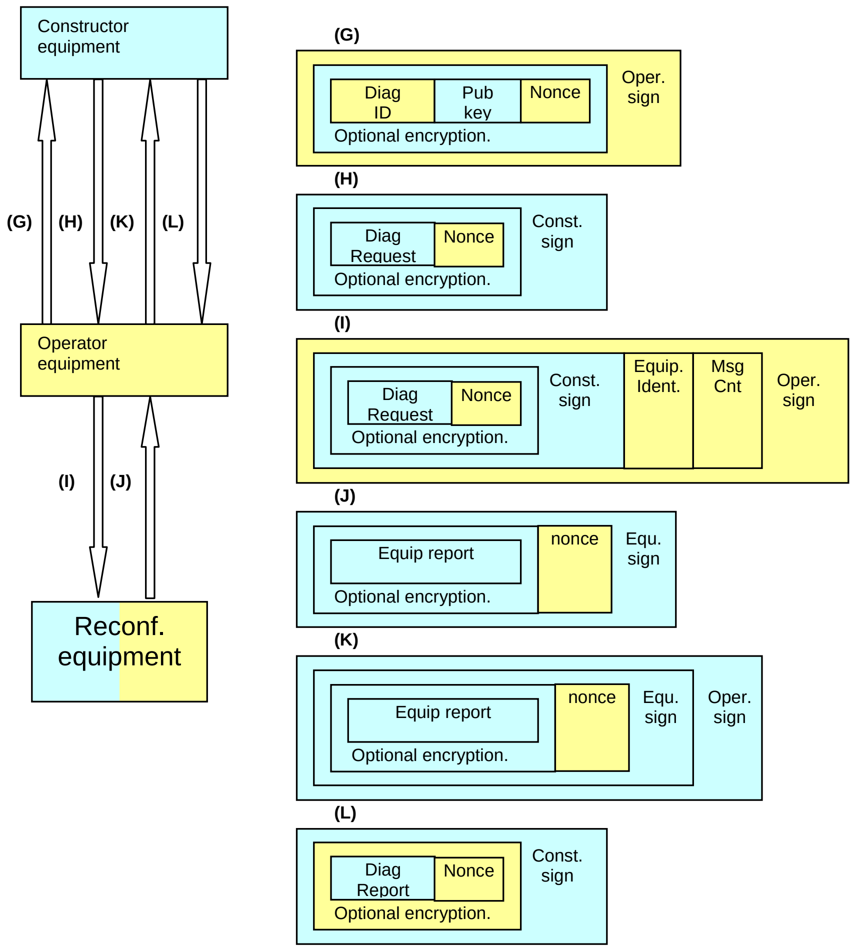


Figure 8: Secure diagnostic

“K”:

Upon receiving message “J”, the operator equipment checks that the nonce matches its initial nonce selection, and if OK, it appends its signature in order to obtain message K. This message is forwarded to the constructor supporting equipment.

“L”:

Once the constructor equipment receives message “K”, it performs the following operations:

- Check of the operator signature,
- Check of the reconfigurable equipment signature, with respect to public key received in initial message G.
- Check that the nonce value matches what was received in message "K"
- Elaborate a final diagnostic report intended to the operator supporting equipment.
- Add the nonce.
- Optionally encrypt the report plus nonce.
- Append its signature.
- Forward the message to the operator supporting equipment.

Eventually, the operator supporting equipment receiving message "L" will check constructor signature, nonce correspondence, and will read and exploit the final report.

2.3.2.4 Software Downloading

2.3.2.4.1 Terminal Initiated

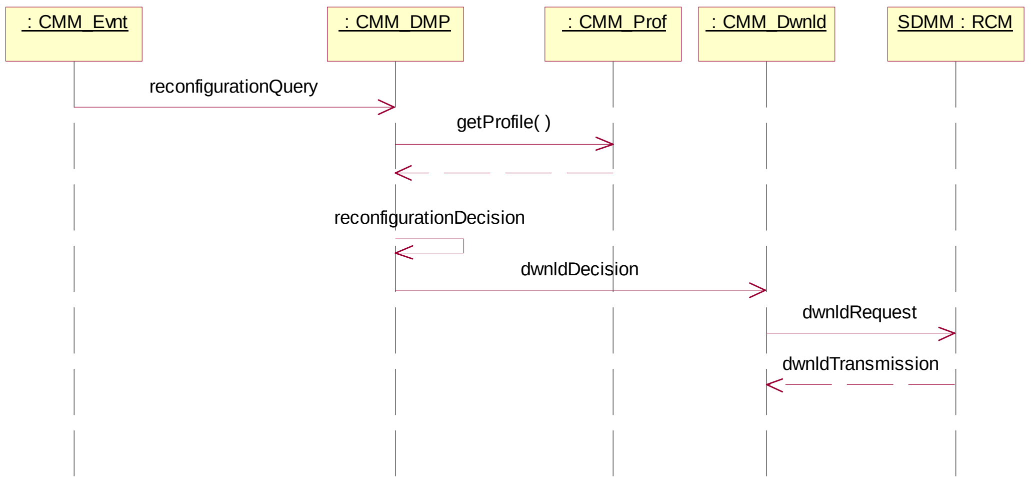


Figure 9: Terminal Initiated SW Download Management

- The Event Handler receives the download event from the corresponding CCM and then it sends a trigger for downloading to CMM_Dwnld.
- The CMM_DMP chooses the appropriate SW running using a set of decision algorithm and applies them on the information retrieved from the CMM_Prof and RCM.
- The Decision Component makes the decision on SW Version (i.e. module, component) based on list of software available and current SW Profile.

2.3.2.4.2 Network Initiated

The following diagram depicts the network-initiated trigger for downloading reconfiguration procedure.

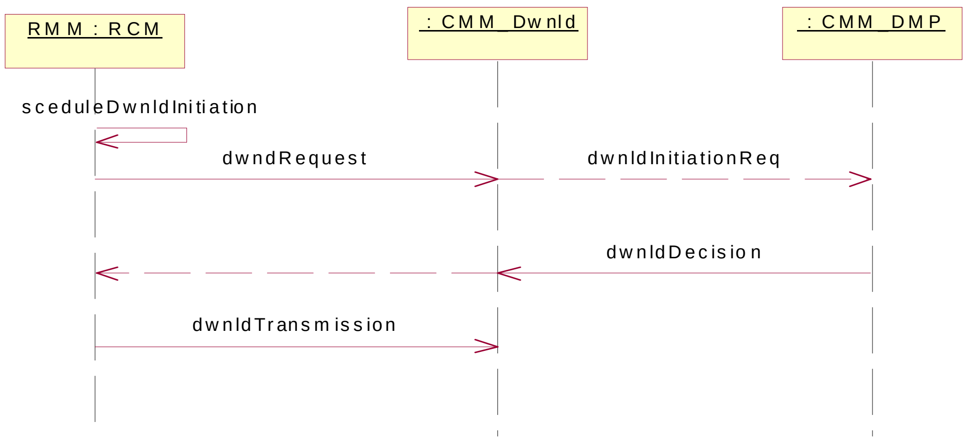


Figure 10: Network-initiated trigger for downloading reconfiguration procedure

2.3.3 Relationships Between Entities (Modules/Components) and Procedures

2.3.3.1 Relations between procedures and modules

The entities depicted in Figure 1 are intended to support mainly the following procedures:

1. Reconfiguration decision-selection
2. SW Downloading
3. SW Deployment
4. Tasks scheduling and instantiation

Those procedures have been described in the above subsection, however, at the following text, there is a more detailed description for the role of each module regarding the reconfiguration procedures.

The following sections describe the relationships between the various modules and procedures of the multi-mode/standard switching scenarios. The first sub section gives an overview on the internal relations and the subsequent sub sections provide sequence diagrams that illustrate the internal relationships as well as the external interactions in the respective scenarios.

2.3.3.1.1 Overview of module relationships

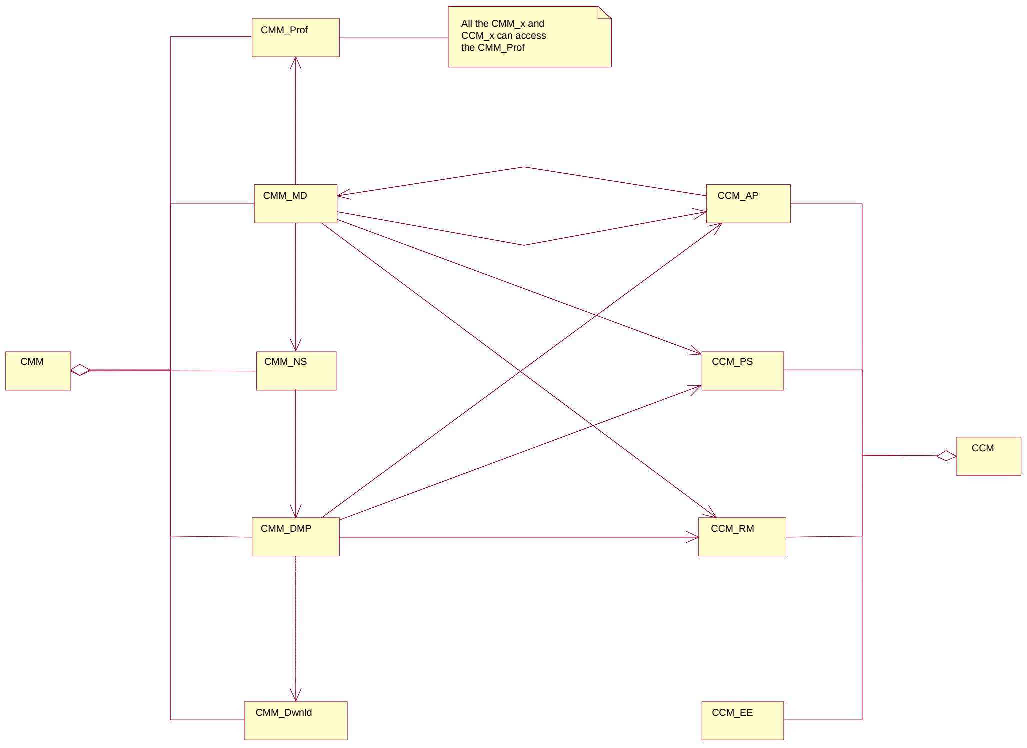


Figure 11: Overview on the internal relations between the modules

The multi-mode/standard switching distinguishes different entities inside the local CMM on the terminal:

- Monitoring and Discovery (CMM_MD)
- Negotiation and Selection (CMM_NS)
- Reconfiguration Implementation (CMM_DMP)
- Configuration Download (CMM_Dwnld)
- Profile information database (CMM_Prof)

Figure 11 depicts the relationships between the CMM modules and the different CCMs.

CMM_MD and CCM_AP

The application layer configuration controller (CCM_AP) informs the monitoring and discovery entity of the CMM (CMM_MD) about changes in the applications requirements and user preferences. Upon this information the CMM_MD can search for a technology that is suitable to the changed requirements. A switch to the new RAT is made.

The CMM_MD on the other hand provides the CCM_AP with information on the current technology capabilities, i.e. available RATs and their service level.

CMM_MD and CCM_PS

The CMM_MD interacts with the protocol layer configuration controller (CCM_PS) to query information on the current protocol stack configuration and to change the protocol stack configuration.

CMM_MD and CCM_RM

The CMM_MD requests the modem reconfiguration controller (CCM_RM) to scan for additional access technologies and to monitor the available access technologies. The CCM_RM provides information on the available access technologies to the CMM_MD.

CMM_MD and CMM_NS

When the CMM_MD has received information on the currently available access technologies, it instructs the negotiation and selection entity of the CMM (CMM_NS) to negotiate and decide which technology shall be used in the future.

CMM_NS and CMM_DMP

After the CMM_NS has selected the RAT to which the terminal shall switch, it issues the implementation of this reconfiguration to the reconfiguration implementation entity of the CMM (CMM_DMP).

CMM_DMP and CCM_x

During the implementation of a new configuration the CMM_DMP interacts with the different CCMs in order to reconfigure and install the software for the new configuration. Furthermore the CMM_DMP is responsible for initiating the handover from one radio chain to another in terminals with multiple radio chains.

CMM_DMP and CMM_Dwnld

In case the necessary software modules for the reconfiguration are not available in the terminal, the CMM_DMP requests the download of these software modules at the CMM_Dwnld.

CMM_x, CCM_x and CMM_Prof

All the CMM and CCM modules can requests profile information from the CMM_Prof as well as they can store / change information in the CMM_Prof.

2.3.3.1.2 Module Relations: Change of Profile Information

Changes of the user preferences, e.g. the required service level, the connectivity and the costs are passed from the CCM_AP to the CMM_MD. The CMM_MD stores the changes in the profile and decides whether these changes require a transfer of profile data to the network or not. Furthermore it decides whether a change of the RAT should be considered (Figure 12). This would mean starting monitoring and discovery actions as well as initiating the negotiation and selection process.

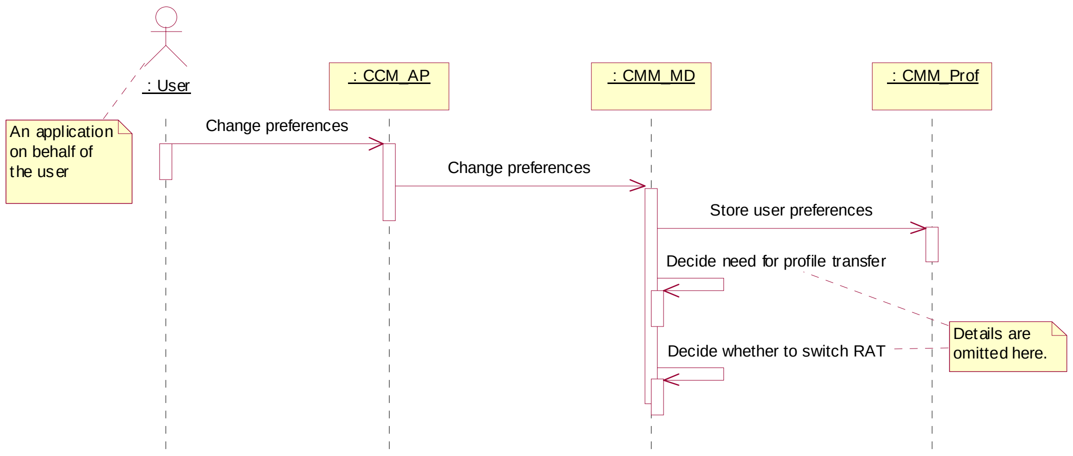

: User
: CCM_AP
: CMM_MD
: CMM_Prof
An application on behalf of the user
Change preferences
Change preferences
Store user preferences
Decide need for profile transfer
Decide whether to switch RAT
Details are omitted here.

Figure 12: Relationships for the change of profile information

2.3.3.1.3 *Module Relations: Profile Transfer*

A terminal and the network exchange profile information during initialisation and whenever a change in the shared parts of the profile occurs (Figure 13).In both directions the profile can be transferred on request or automatically when the sending side feels the transfer makes sense (Figure 14).

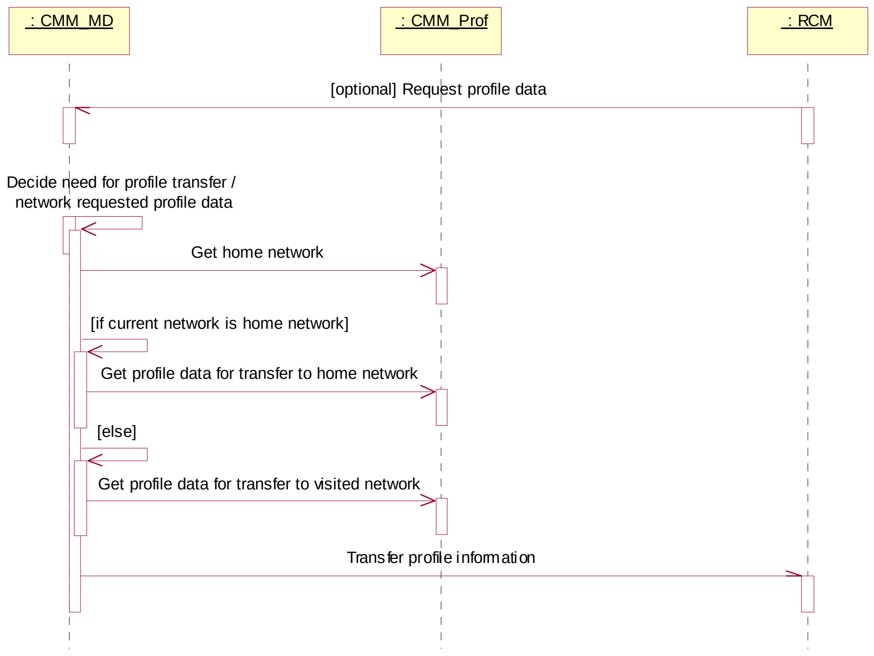

: CMM_MD
: CMM_Prof
: RCM
[optional] Request profile data
Decide need for profile transfer / network requested profile data
Get home network
[if current network is home network]
Get profile data for transfer to home network
[else]
Get profile data for transfer to visited network
Transfer profile information

Figure 13: Sequence diagram: Profile transfer to the network

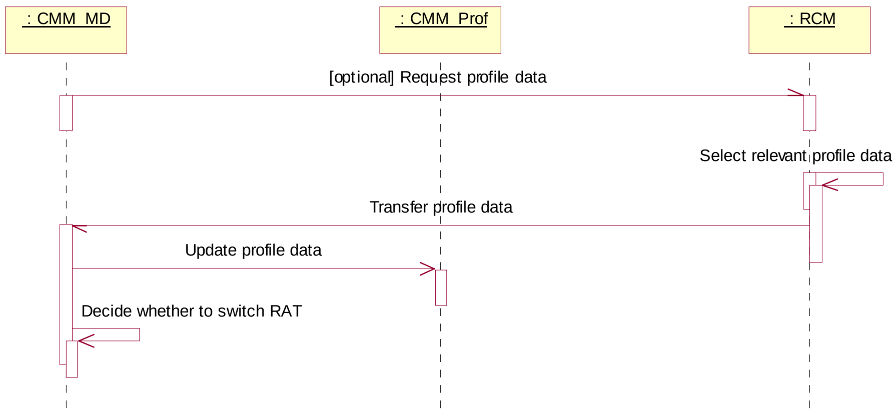

: CMM_MD
: CMM_Prof
: RCM
[optional] Request profile data
Select relevant profile data
Transfer profile data
Update profile data
Decide whether to switch RAT

Figure 14: Sequence diagram: Profile transfer to the terminal

2.3.3.1.4 Module Relations: Monitoring and Discovery

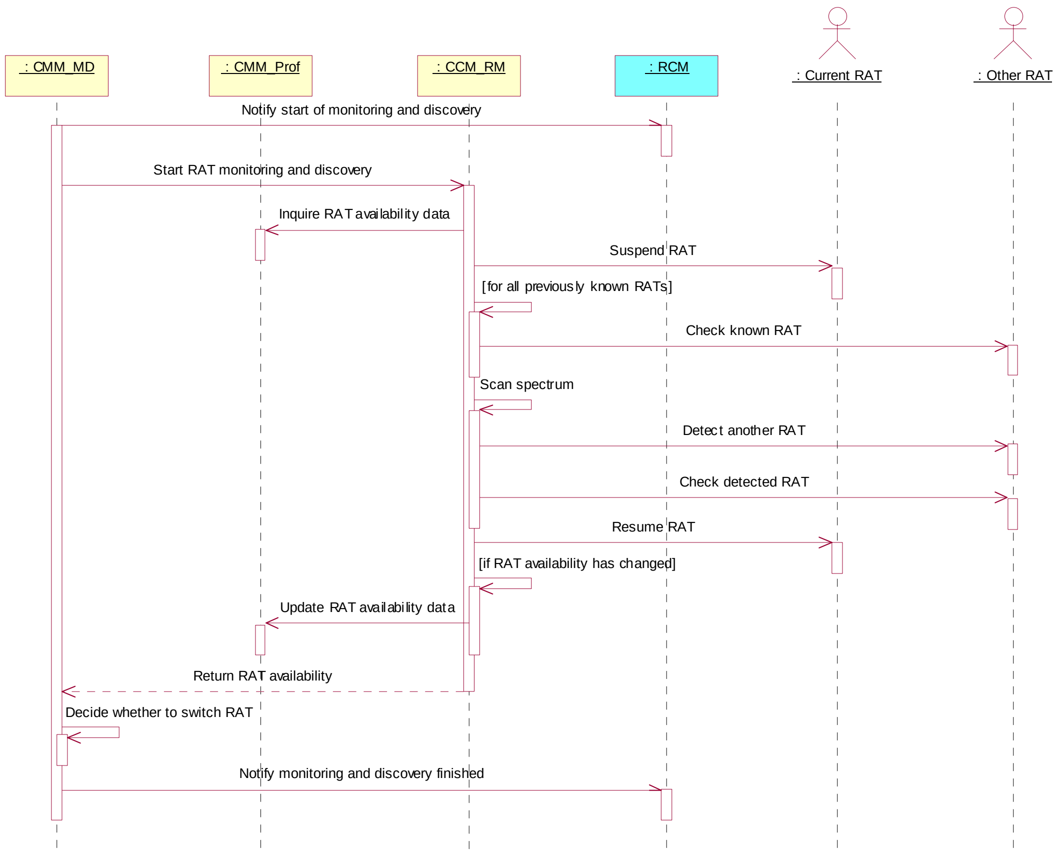

: CMM_MD
: CMM_Prof
: CCM_RM
: RCM
: Current RAT
: Other RAT
Notify start of monitoring and discovery
Start RAT monitoring and discovery
Inquire RAT availability data
Suspend RAT
[for all previously known RATs]
Check known RAT
Scan spectrum
Detect another RAT
Check detected RAT
Resume RAT
[if RAT availability has changed]
Update RAT availability data
Return RAT availability
Decide whether to switch RAT
Notify monitoring and discovery finished

Figure 15: Sequence diagram: Monitoring and discovery for a single radio chain terminal

During the monitoring and discovery process single radio chain terminals suspend their current RAT temporarily. They inform the RCM on the network side when they start the monitoring and discovery activity and as soon as they have finished it. These notifications allow the RCM to react appropriate, e.g. buffer data packets.

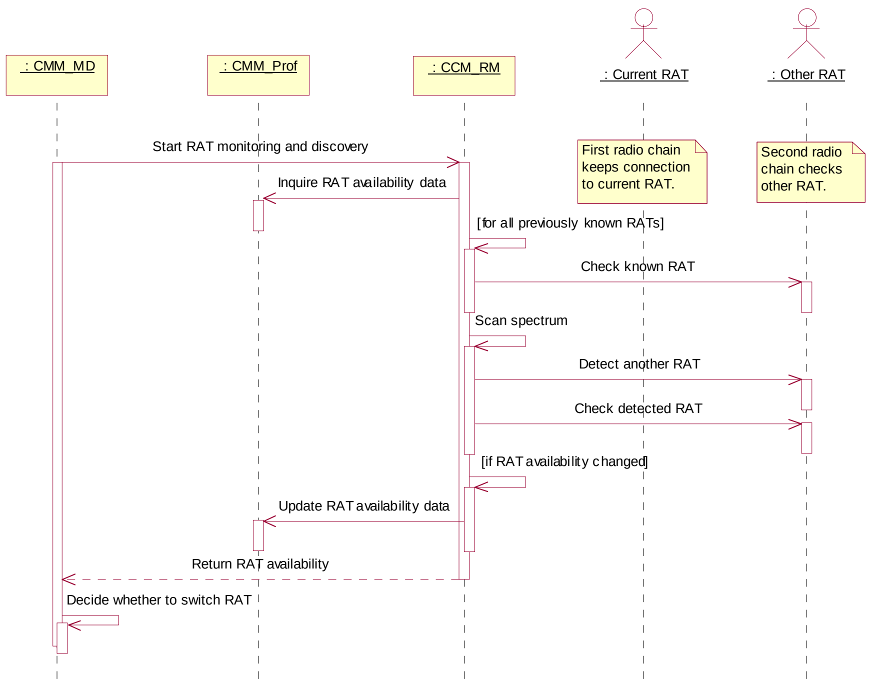

: CMM_MD
: CMM_Prof
: CCM_RM
: Current RAT
: Other RAT
Start RAT monitoring and discovery
Inquire RAT availability data
First radio chain keeps connection to current RAT.
Second radio chain checks other RAT.
[for all previously known RATs]
Check known RAT
Scan spectrum
Detect another RAT
Check detected RAT
[if RAT availability changed]
Update RAT availability data
Return RAT availability
Decide whether to switch RAT

Figure 16: Sequence diagram: Monitoring and discovery for a multi radio chain terminal

2.3.3.1.5 Module Relations: Negotiate and Select

When the CMM_MD considers that a change of the RAT might be useful, it triggers the CMM_NS to select an appropriate RAT. Therefore the CMM_NS uses the information stored in the profile database as well as it interacts with the RCM on the network side to include the current dynamic capabilities of the network side into the decision. Profile information may be exchanged between the network and the terminal to update the information stored on both sides before the decision is made, see Figure 17.

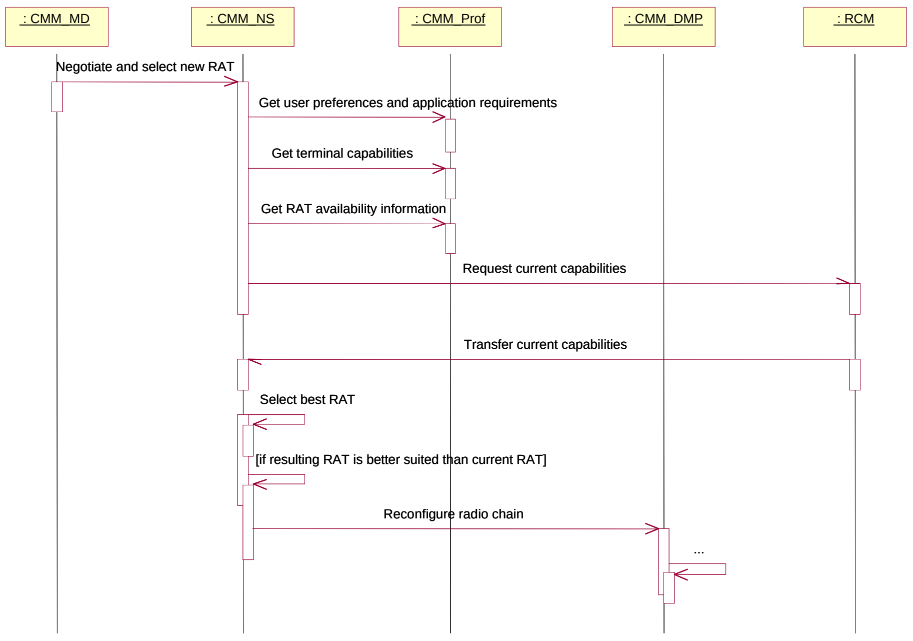

: CMM_MD
: CMM_NS
: CMM_Prof
: CMM_DMP
: RCM
Negotiate and select new RAT
Get user preferences and application requirements
Get terminal capabilities
Get RAT availability information
Request current capabilities
Transfer current capabilities
Select best RAT
[if resulting RAT is better suited than current RAT]
Reconfigure radio chain
...

Figure 17: Sequence diagram: Negotiation and selection of a suitable RAT

2.3.3.1.6 Module Relations: Reconfigure Radio Chain

The reconfiguration of a radio chain and the handover (seamless or non-seamless) from one RAT to another involves multiple interactions between the CMM_DMP and the various CCM modules.

If software modules required for the reconfiguration are not yet available on the system, the CMM_DMP interacts with the CMM_Dwnld to download the required modules from the network.

The terminal informs the RCM before starting the reconfiguration and after finishing it. These notifications allow the RCM to react on the RAT switch, e.g. routing all ongoing communication to the new connection and buffering data packets during an interruption of the connection.

Single radio chain terminals cannot handover seamlessly between two RATs, see Figure 18. Multi radio chain terminals stay connected to the old RAT until the configuration of the new RAT is completed see Figure 19. They seamlessly hand over from the old RAT to the new RAT.

Many steps in the configuration of a radio chain are optional depending on the actual configuration. Especially changes in the protocol stack layer and the application layer are only needed during mode switch (e.g. UMTS to WLAN), but not during a handover between two access points of the same technology (e.g. switch between two UMTS base stations).

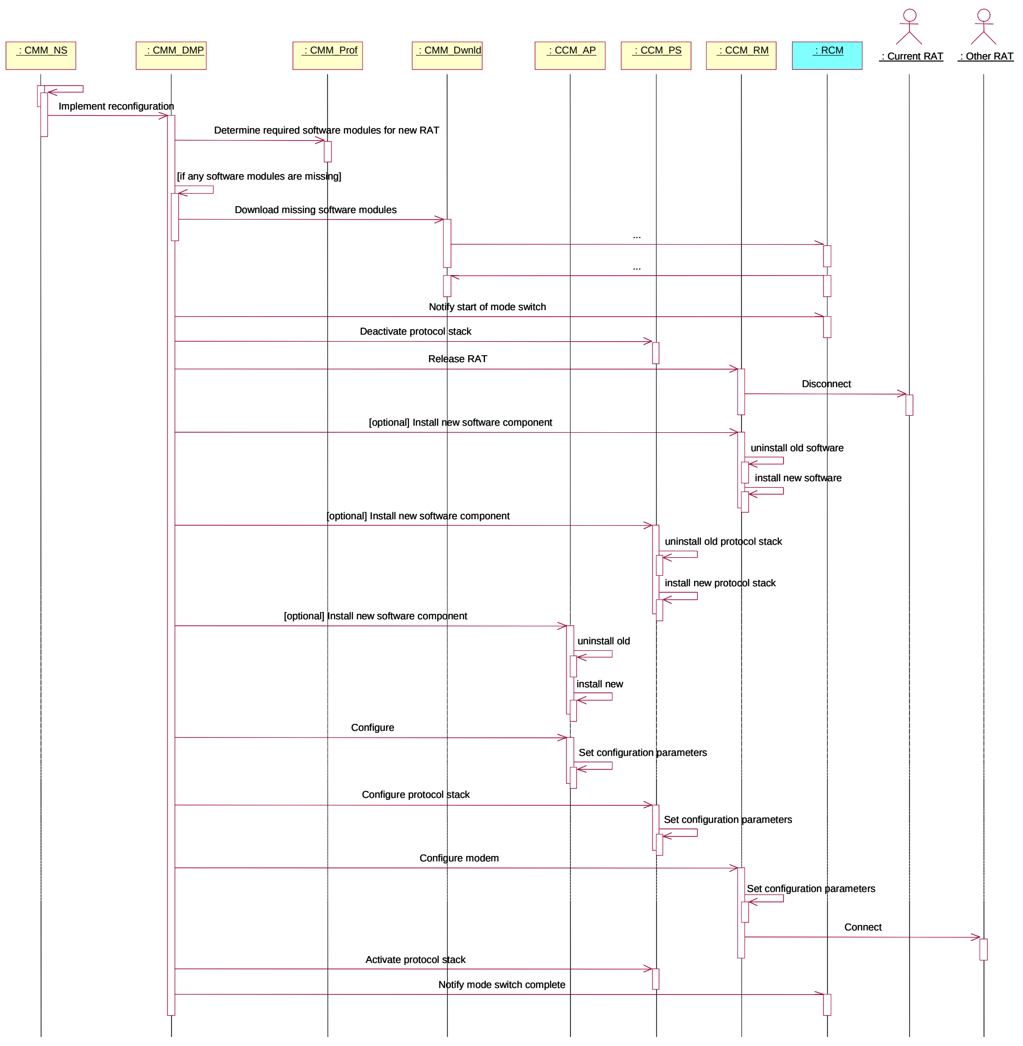

: CMM_NS
: CMM_DMP
: CMM_Prof
: CMM_Dwnld
: CCM_AP
: CCM_PS
: CCM_RM
: RCM
: Current RAT
: Other RAT
Implement reconfiguration
Determine required software modules for new RAT
[if any software modules are missing]
Download missing software modules
...
...
Notify start of mode switch
Deactivate protocol stack
Release RAT
Disconnect
[optional] Install new software component
uninstall old software
install new software
[optional] Install new software component
uninstall old protocol stack
install new protocol stack
[optional] Install new software component
uninstall old
install new
Configure
Set configuration parameters
Configure protocol stack
Set configuration parameters
Configure modem
Set configuration parameters
Connect
Activate protocol stack
Notify mode switch complete

Figure 18: Sequence diagram: Reconfigure radio chain of a single radio chain terminal

2.3.3.1.7 Module relations: reconfiguration of TCP parameters

Figure 20 shows an example of the TCP adaptation process after detection of a vertical or horizontal handover. First, it is supposed that the TMM informs the CMM about the nature of the HO, and the new access network conditions, via the CMM_IfNss. Then, the CMM_IfNss delivers the information enclosed in the handover notification (HO type, network conditions and access technology information) to the CMM_NS. This entity can then consult the CMM_Prof in order to associate new TCP parameters values to the new access network conditions and characteristics, and/or the user profile. Then, the CMM_NS decides if the TCP parameters change is necessary. Note that this decision will be taken according to configuration parameters set by the user or the mobile device administrator.

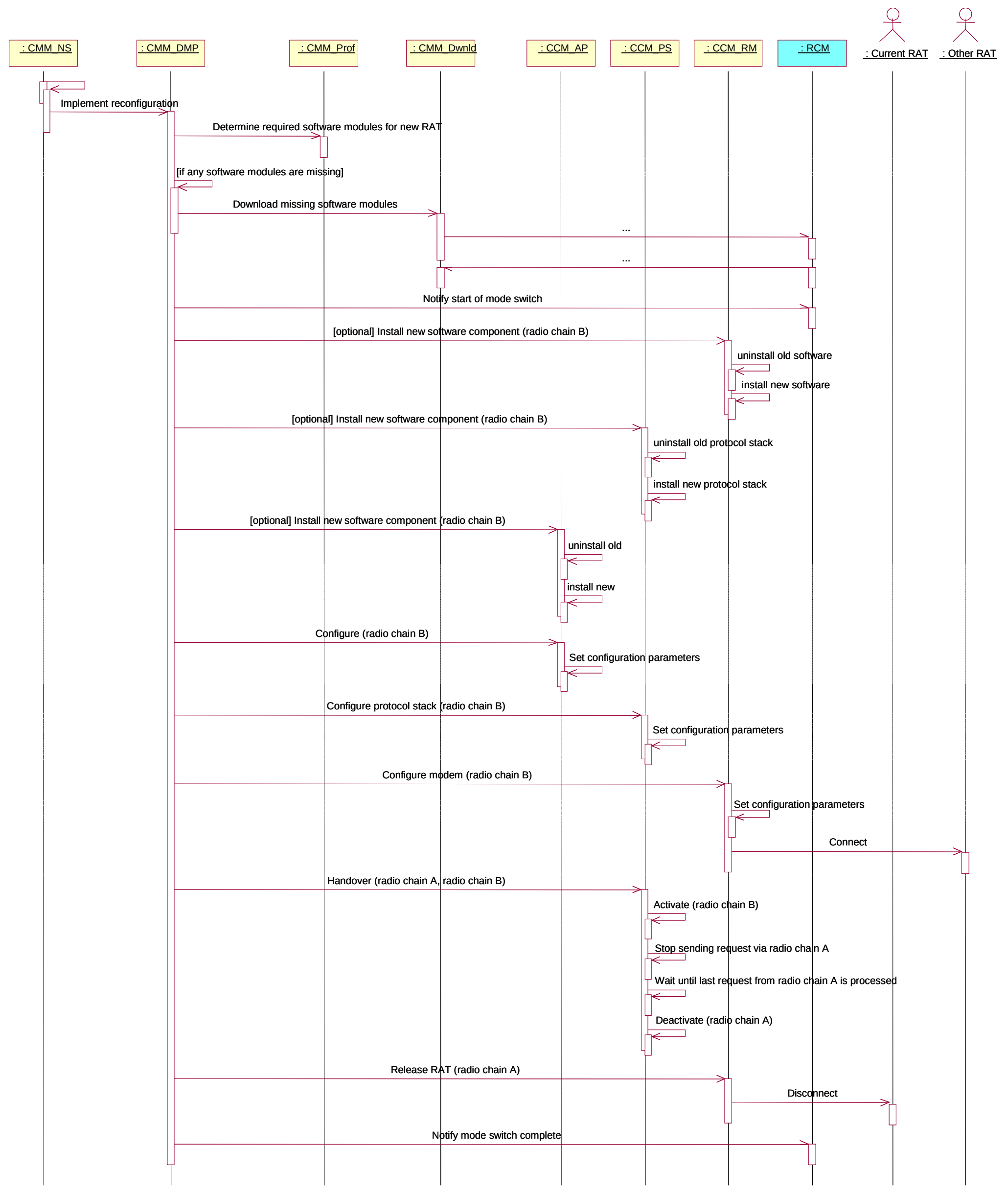

: CMM_NS
: CMM_DMP
: CMM_Prof
: CMM_Dwnld
: CCM_AP
: CCM_PS
: CCM_RM
: RCM
: Current RAT
: Other RAT
Implement reconfiguration
Determine required software modules for new RAT
[if any software modules are missing]
Download missing software modules
...
...
Notify start of mode switch
[optional] Install new software component (radio chain B)
uninstall old software
install new software
[optional] Install new software component (radio chain B)
uninstall old protocol stack
install new protocol stack
[optional] Install new software component (radio chain B)
uninstall old
install new
Configure (radio chain B)
Set configuration parameters
Configure protocol stack (radio chain B)
Set configuration parameters
Configure modem (radio chain B)
Set configuration parameters
Connect
Handover (radio chain A, radio chain B)
Activate (radio chain B)
Stop sending request via radio chain A
Wait until last request from radio chain A is processed
Deactivate (radio chain A)
Release RAT (radio chain A)
Disconnect
Notify mode switch complete

Figure 19: Sequence diagram: Handover for a multi radio chain terminal

If the TCP parameters modification is decided by the CMM_NS, then it generates a request to the CMM_DMP, i.e. the entity in charge of pushing the TCP parameters change. At last, the CMM_DMP forwards the new TCP parameters values to the CCM_PS in order to make this change operational.

After the TCP parameters modification completion, acknowledgement messages can be sent back to the CMM_DMP and the CMM_NS if required, but this kind of notification, in most cases, is not required.

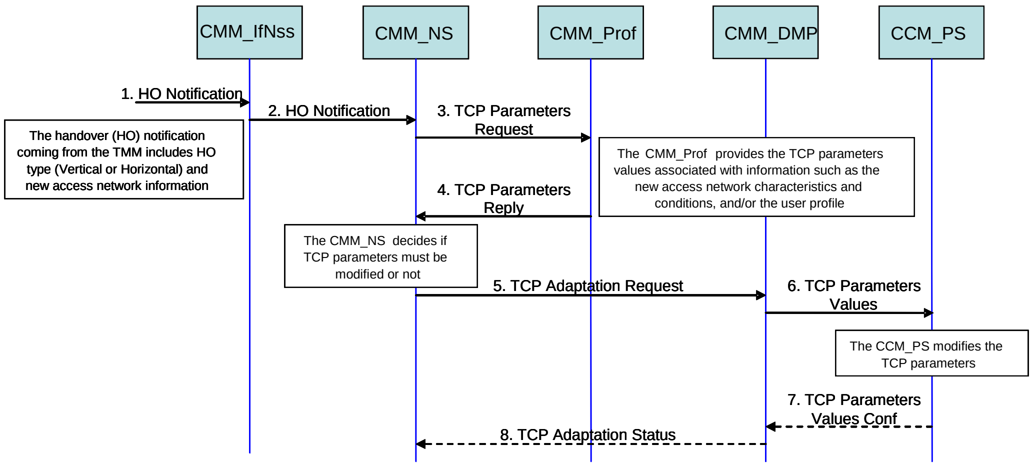

CMM_IfNss
CMM_NS
CMM_Prof
CMM_DMP
CCM_PS
1. HO Notification
2. HO Notification
3. TCP Parameters Request
The handover (HO) notification coming from the TMM includes HO type (Vertical or Horizontal) and new access network information
The CMM_Prof provides the TCP parameters values associated with information such as the new access network characteristics and conditions, and/or the user profile
4. TCP Parameters Reply
The CMM_NS decides if TCP parameters must be modified or not
5. TCP Adaptation Request
6. TCP Parameters Values
The CCM_PS modifies the TCP parameters
7. TCP Parameters Values Conf
8. TCP Adaptation Status

Figure 20: Example of TCP parameters adaptation after handover

2.3.3.2 Definition of relationship between reconfiguration module controller and CMM

The network node (terminal/base station) is configured to a standard (or agreed transmission scheme) “x” (e.g. WLAN) and needs to be re-configured to a completely different standard “y” (e.g. UMTS).

Assuming a scenario in which a reconfiguration procedure, requiring the complete terminal reconfiguration from an access standard x to a standard y, is triggered, and the required downloads of policy and software and the activation of the profile (including the VCP) have been accomplished, the network dispatches a message about the successful completion of the VCP. The CMM_DMP receives this notification and informs the CMM about the permitted reconfiguration and also forwards information about the changes to be implemented in the reconfiguration part. The CMM uses the information about the changes necessary to generate a set of new CCMs, which, in turn, install the reconfigurable modules. Every CCM then performs an initialisation of both state machines and its parameters, it then allocates memory and the I/O parameter types of the reconfigurable module. After this initialisation, the CCM issues a request to the CMM_Dwnld to provide handles to the required software modules. The CMM_Dwnld responds to the requesting CCM, by forwarding the required handles. After this exchange the CCM implements the modules and sends a notification to the CMM. Once the installation is complete, the result is passed to the CMM.

This sequence has to be repeated for every CCM, the responses and outcomes are collected by the CMM. If the installations of the reconfigurable modules are completed, the CMM sends requests to the modules (i.e. to their CCM) to create the connection points between the modules. Confirmations about the establishment of the ports between the modules are to be forwarded from the CCM before the CMM can send a test signal to the CCMs. The test signal (or sequence of test signals) ensures that every module individually and also the complete module structure can be tested. After receiving a positive response from every CCM confirming the functionality of every connection point and the new radio configuration as a whole, the CMM issues a connect message to all CCMs. The “old” reconfigurable modules continue to function until the “new” implementation is operable. During the transition period, the first (if there are more then one) module in the chain of the "old" chain implementation starts to buffer incoming information (i.e. to prevent possible loss of data), simultaneously the signal is passed through the "new" module chain. The transition period constitutes a second function test for the new radio implementation. After a series of performance checks of the new configuration implementation, the CMM issues a request to all CCMs to finally hand over from standard x to the new standard y. The hand over is implemented in a sequential way, whereby the buffered data in the first (old) module is sent to the first CCM of the "new" module chain, at the same time the old CCM continues buffering the incoming signal. This CCM then processes the buffered data from the first old module through the new modules and then forwards the (meanwhile) buffered data. This is repeated until the last "new" module is connected within the module chain. Once the new configuration is in place and every message delivered from the reconfigurable modules to CCMs is within the set parameter limits, notification messages from the CCMs are send to the CMM to finally confirm the reconfiguration. As a further step, the CMM requests the old CCMs to destroy their modules. If the destruction of all old reconfigurable modules are confirmed, the CMM destroys the old CCMs. Finally, the CMM requests, from the CMM_DMP, the update of the status of the new terminal profile and the registration of the new terminal configuration to the network. The registration of the new configuration and the acknowledgement sent by the network concludes this complete terminal reconfiguration sequence.

2.3.3.3 Definition of relationship between security manager and CMM

If the Security Manager receives a request, from the CMM, to establish a secure connection between terminal and network, it performs a mutual authentication procedure with the network part. The next step then is to respond to the CMM, the response carries an indication that authentication has taken place and a connection may then be established. In case the request was sent by the network, an identical process has to be performed (i.e. however in the opposite direction).

If any message is transmitted between CMM_Dwnld, CMM_Prof, CMM_DMP, CMM and network, it becomes processed and packed into a secure frame. After encryption, it becomes transmitted to the network. If a packet arrives from the network, the Access Manager authenticates and forwards it to the "E&D Factory" where the necessary decryption is performed, after this, the message is passed to the appropriate functional entity. The secure connection remains open until its termination by the Configuration Manager. The same sequence of events is repeated if the network requires establishing a secure connection with the terminal for reconfiguration request.

3 FLEXIBLE AIR INTERFACES

3.1 Introduction

This chapter introduces a reference model for a multi-mode protocol stack of a flexible, dynamic reconfigurable air-interface for future wireless networks. This future wireless network has the vision of a ubiquitous radio system concept providing wireless access from short-range to wide-area, with one single adaptive system for all envisaged radio environments. It will efficiently adapt to multiple scenarios by using different modes of a common technology basis. The generic protocol stack enables an efficient realization of reconfigurable protocol software as part of a completely reconfigurable wireless communication system. Following a bottom-up approach this chapter considers parameterizable modules of basic protocol functions corresponding to the Data Link Layer (DLL) of the ISO/OSI reference model. System specific aspects of the protocol software are realized through adequate parameterization of the modules. Further functionality and behavior can be added through the insertion of system specific modules or inheritance. The subsequent sections will elaborate on the way how such a generic protocol stack can be constructed in a general way, followed by the more specific example of a “generic link layer”. Additionally, the last section enligths physical layer related aspects of a flexible air interface in introducing a multi-antenna based approach for adaptive data transmission.

3.2 Protocol Reconfigurablity based on Generic Protocol Stack

3.2.1 The Idea of a Generic Protocol Stack

The rationale for approaching a generic protocol stack is that all communication protocols share much functional commonality, which can be exploited to build an efficient multi-mode capable wireless system. The term “generic” can be substituted in the following by “common” and “general”. The aim is to gather these common parts in a single generic stack and specialize this generic part following particular requirements of the targeted mode, also referred to as Radio Access Technology (RAT), as depicted in Figure 21. The targeted advantages of this concept are: runtime reconfigurability and maintainability, code/resource sharing and protocol development acceleration through reusability.

A key issue of the later introduced reference model for multi-mode protocols is the separation of a layer into specific and generic parts. The term generic is used by multiple authors with different knowledge backgrounds leading to dissimilar or even contradictory understandings of genericity. In taking the realization of a protocol stack out of generic and specific parts into account this becomes a software engineering problem of generic programming. Generic programming can be defined as:

“*programming with concepts, where a concept is defined as a family of abstractions that are all related by a common set of requirements. A large part of the activity of generic programming, particularly in the design of generic software components, consists of concept development - identifying sets of requirements that are general enough to be met by a large family of abstractions but still restrictive enough that programs can be written that work efficiently*” [121]

The balancing of the trade-off between general usability and implementation effort is crucial for the success of the separation of complex protocol software into generic and specific parts.

In general, generic protocol software may be realized through parameterizable modules and/or inheritance of system specific behaviour. Well known programming patterns from computer science provide thereby a suitability-proven fundamental approach to the efficient realization of reconfigurable multi-mode protocol software.

As depicted in Figure 21, generic parts can be identified on different levels in the context of communication protocols:

- Architecture and composition of a protocol stack, introduced in Section 3.3
- Functions fulfilled by a layer that imply a certain behaviour, outlined in Section 3.5
- Data structures, i.e., protocol data units, used for communication between peer-entities of a layer
- Protocol framework: Common rules for communication, as for instance the structure of a Medium Access Control (MAC)-frame (sequence and duration of broadcast, downlink and uplink phase)
- Management of a layer and protocol stack

The communalities form, together with mode specific parts, a system specific protocol stack. An efficient multi-mode capable stack is realized in adding cross stack management related functions as introduced in the previous section.

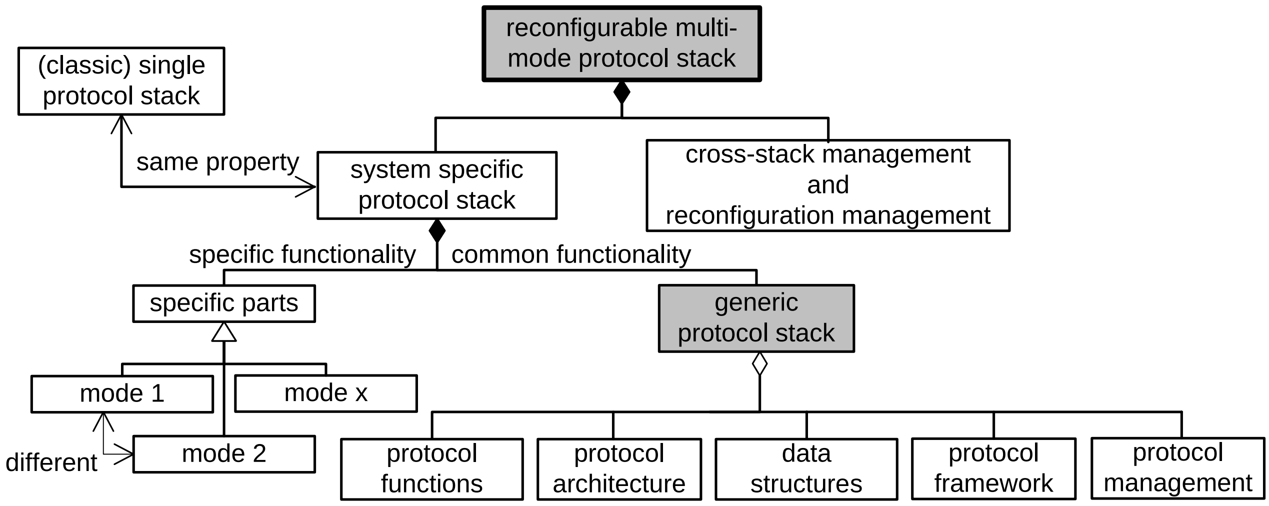


Figure 21: UML diagram of the generic protocol stack in the context of protocol reconfigurablility

From the software engineering perspective, there are in general two possibilities for approaching the generic protocol stack: (1.) Parameterizable functional modules and/or (2.) inheritance, depending on the abstraction level of the identified protocol commonalities. As introduced above the focus here is more on the modular approach while the inheritance-based approach is in considered in [103],[104] and [105]. The combination of both approaches is promising to fulfil all requirements of protocol reconfigurability. Additionally, [109] takes up the idea of a generic protocol stack in focusing on a generic link layer for the cooperation of different access networks at the level of the data link layer. However, not only the link layer protocols have to be considered in a multi-mode capable network but also higher layer functions as for instance the control and management of the radio resources as well as mobility.

3.2.2 Development of a Generic Protocol Stack

Using the classical ISO/OSI protocol stack reference model to compare the stacks of different RATs, a high degree of similarity can be found. Many of the features of the control software can be implemented as *shared resources*. Therefore a certain software development process should be applied, called Design of Generic and Adaptive Protocol Software (DGAPS). Applying DGAPS results in a *generic protocol stack* that provides a common basis for a number of different systems. Specialization by introducing standard-specific functions to the generic stack stepwise results in a specific realization towards a specific protocol stack.

In a first step (step1) different systems, say System I and System II, need to be analyzed layer by layer to identify their commonalities. A more detailed description of the analysis process together with a reference implementation is described in [99]. The number of different systems to be considered may be two or larger. The result will be a specification of a common subset of the access protocol stacks for the systems, [103]. Since this stack provides the common characteristics of the considered air-interface standards it is called a *generic protocol stack* or *protocol skeleton*.

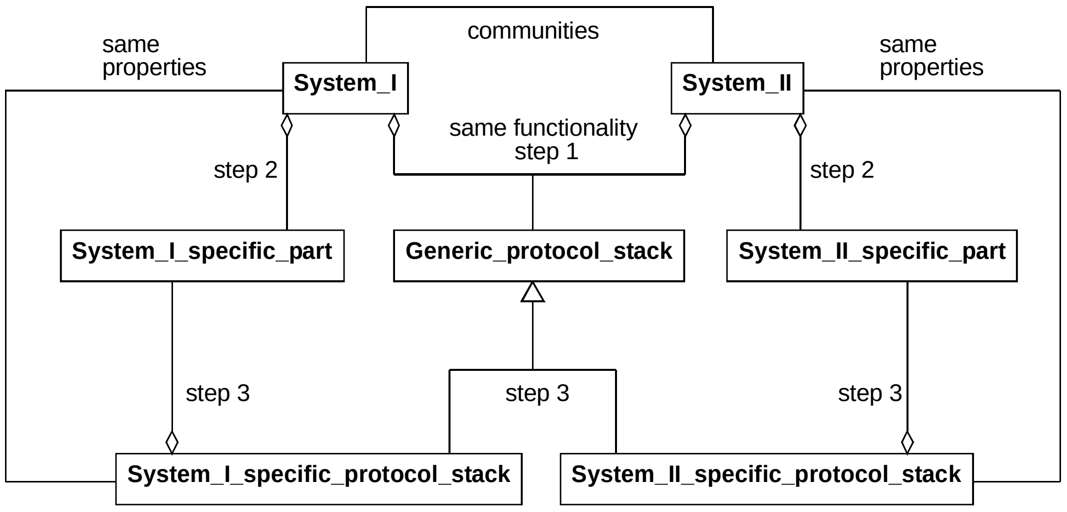


Figure 22: Interaction of Components for an SDR Protocol Stack

The next step (step2 in Figure 22) is to develop specifications dedicated to given air-interface standards, say for System I or System II. These include functions that are specific to respective standards and thus represent the individual behavior of a system. Different approaches can be taken to achieve that goal. In order to make use of the object-oriented properties together with inheritance, it is suggested to implement these parts as subclasses derived from base classes implemented within the generic stack. This is of special advantage, if more than two systems are considered; procedures that are common to most but not necessarily to all standards still will be implemented within the generic stack. The standard-specific supplements than will have to redefine/overload the respective procedures and the behavior required is achieved then.

To result in a dedicated air-interface standard, the generic protocol stack and the standard-specific supplement, have to be merged (step3). This can be done by means of inheritance. Figure 22 shows the correlations and dependencies of the aforementioned parts in the notation of UML. In order to distinguish between a specific protocol stack that is designed either conformant with the above presented approach or not, the notation *System_X* (non-conformant) and S*ystem_X_specific_protocol_stack* (conformant) is used.

3.2.3 Separation of the Protocol Stack into Generic and Specific Parts

The reference model presented in this chapter is based on the widespread perception that radio interface protocol functions can be divided into two sets of functionalities:

I. **Mode-/System-specific functions**: These are protocol functions that are unique to a certain kind of radio interface mode and can not be found in any other mode of the same or any other radio interface. Examples for such mode-specific functions are the allocation of a dedicated physical resource and parameters for dimensioning a mode related to the local communication environment.

II. **Generic (common) functions**: The view that is usually taken in standardization is that "generic" functions are not fully specified and have to be enriched by specifying missing parts. This view does not apply in this case. It has to be noted that in this context, the "generic" functions are assumed to be the identified set of common (mode independent) functions of a set of air interface modes. This means they are "generic" from the viewpoint of the modes, but **not from the viewpoint of functionality**. It is assumed that these functions can be adapted to the use in any of the targeted modes through proper parameterization. They are generic in the following sense: In most cases, they will have to rely on additional mode-specific functions to provide the full functionality of a certain protocol layer of a certain air interface mode. An ARQ protocol for instance is a generic protocol which can be specified as Go-Back-N, Selective Reject or Hybrid ARQ protocol providing an error-free data transfer. In regarding the generic protocol stack as toolbox of common protocol functions which may be used in a specific protocol implementation the multitude of protocol functions considered as being generic is increased: It suffices that they can be found in at least two protocol modes to be counted as belonging to the toolbox of generic protocol functions.

A general and trivial observation is that the degree of commonalities between a set of different air interface modes is decreasing with the number of modes being integrated, as depicted in Figure 23. The interesting observation that can be made in this context is that the design processes of new air interfaces can be expected to highly benefit from cooperation, resulting in the potential to increase commonalities between these new air interfaces and thus reduce development and equipment costs. Thus, the degree of similarities can be improved, when considering multi-mode capability based on genericity during the development and standardization of new protocols compared to the common parts of existing 2G and 3G protocols.

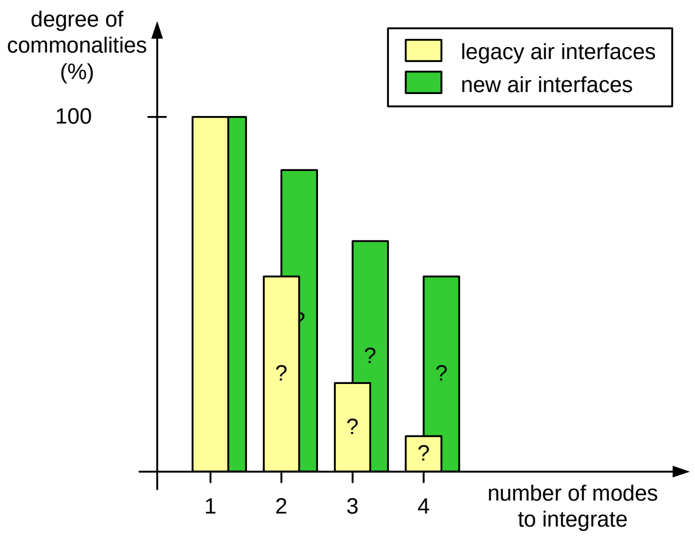


Figure 23: Estimated degree of commonalities in multi-mode protocol stacks as a function of the number of integrated modes. New air interfaces can benefit from cooperative design taking general usability of the protocol into account.

3.3 Multi-Mode Reference Model based on Modes Convergence

3.3.1 A Multi-Mode Protocol Architecture

Figure 24 illustrates the architecture of a multi-mode protocol stack for a flexible air interface [126]. The layer-by-layer separation into specific and generic parts enables a protocol stack for multiple modes in an efficient way: The separation is the result of a design process that is referred to as cross-stack optimisation, which means the identification and grouping of common (generic) functions. The generic parts, marked green in Figure 24, of a layer can be identified on different levels as introduced in Section 3.2.1. The generic parts are reused in the different modes of the protocol stack. All generic parts together can be regarded as generic protocol stack [103], [104] and [105]. The composition of a layer out of generic and specific parts is exemplarily depicted in Figure 24.

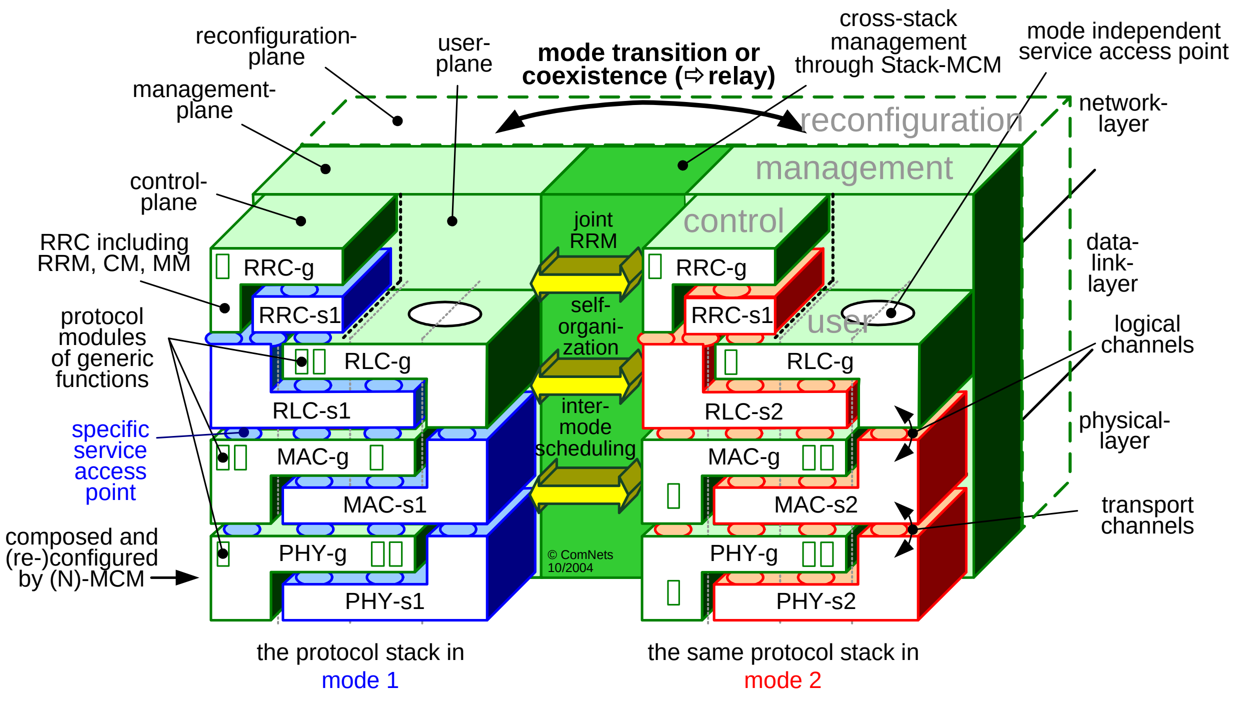

reconfiguration-plane
user-plane
mode transition or coexistence (⇨relay)
cross-stack management through Stack-MCM
mode independent service access point
management-plane
control-plane
reconfiguration
management
control
user
network-layer
data-link-layer
logical channels
physical-layer
transport channels
RRC including RRM, CM, MM
protocol modules of generic functions
specific service access point
composed and (re-)configured by (N)-MCM
joint RRM
self-organi-zation
inter-mode scheduling
RRC-g
RRC-s1
RLC-g
RLC-s1
MAC-g
MAC-s1
PHY-g
PHY-s1
RRC-s1
RLC-s2
MAC-s2
PHY-s2
© ComNets 10/2004
the protocol stack in mode 1
the same protocol stack in mode 2

Figure 24: The Multi-mode protocol architecture, facilitating transition between modes (inter-mode handover) and coexistence of modes (in relay stations connecting different modes) by way of the cross-stack management supported by the modes convergence manager of a layer or stack.

The composition and (re-)configuration of layer is performed by the (N)-Layer Modes Convergence Manager ((N)-MCM). The protocol modules of generic functions are exemplarily introduced: Some of them are reused in a layer and/or additional functions are taken from the toolbox of common protocol functions as part of the generic protocol stack. The RRC on the control-plane and the RLC on the user-plane are generic to the layers located above. A mode specific protocol stack has an individual management-plane. Radio Resource Management (RRM), the Connection Management (CM) and the Mobility Management (MM) are located in the Radio Resource Control (RRC) layer. The cross-stack management of different modes completes the reference model for multi-mode protocols in connecting the management-planes of the device's modes with the help of the Stack Mode Convergence Manager (Stack-

MCM). Stack-MCM and (N)-MCM exchange in a hierarchical order data between two modes. The transition between modes and the coexistence of several modes is performed by the Stack-MCM. The (N)-MCM enables composing a layer out of different parts as depicted in Figure 24 and introduced in the previous section. The split between user and control-plane is limited to the network layer, as known from H/2, and the DLL is used for both, signalling and user-data transfer.

The (N)-MCM is the intermediator between the generic and specific parts of the multi-mode protocol stack's layers: All SAPs, i.e., interfaces, touched during the mode transition of a layer are administrated by the (N)-MCM. In the classical view of protocols as state machines, the (N)-MCM transfers all state variables of the protocol layer between two modes with the help of the Stack-MCM. This could for instance imply the state transfer of being connected from one mode to the other together with a data transfer of received but unconfirmed data frames.

Concrete, the (N)-MCM manages a single layer and has the following tasks and responsibilities which are introduced later in this section:

- Layer composition and reconfiguration, considering all interfaces related to the transition between two modes
- Protocol convergence:
  - horizontally – between "generic" and "specific" parts
  - vertically – mapping of higher layer user data flows for RLC as known for instance from ATM
- Data preservation and context transfer

Furthermore, the convergence between mode specific protocol stacks is realized through the Stack-MCM implying implicitly the following functions:

- Joint Radio Resource Management (Radio resource coordination) between different modes
- Inter-mode scheduling
- Self-organization (frequency allocation of adjacent relays and APs, user data flow routing)

The reconfiguration-plane located behind the management-plane, see Figure 24, is not considered in this paper. The Stack-MCM realizes the reconfiguration of the protocol stack from one mode to the other in providing services to the reconfiguration-plane. The reconfiguration plane contains all functions related to reconfiguration management [8] as for instance the security aspects of reconfiguration and software download as well as the communication of the reconfiguration capabilities of a device.

3.3.2 Composition of a Layer from Specific and Generic Parts

Figure 25 shows the general structure of a protocol layer conforming to the reference model in [126]. It is assumed that the functionality inside the layer is always composed of a generic (common to all modes) part and mode-specific parts, which jointly provide the modes' services of the layer via Service Access Points (SAPs). Through this, layers can be configured for one mode at a time. Modes can also coexist temporarily or permanently. The specific SAPs of a layer are defined via the currently used mode or set of modes[1]. This does not preclude the possibility that SAPs of different modes can be accessed by higher layer entities in a common way as visualized by L(N)-SAP-g.

The composition and (re-)configuration of the layer is taken care of by a layer-internal instance, the (N)-Layer Modes Convergence Manager ((N)-MCM), which resides in the management plane. The (N)-MCM

[1] As a result of the actual configuration of a protocol layer (N) for a mode X, it makes available mode X's (N)-service number Y via a mode X specific (N)-SAP for the service Y (Notation used: (N)-SAP-sX.Y)

enables that a (N)-layer provides multiple modes and makes functionality of one mode or common to several modes available. An instance of the MCM serves as a reconfiguration handler in each layer of the air interface protocols.

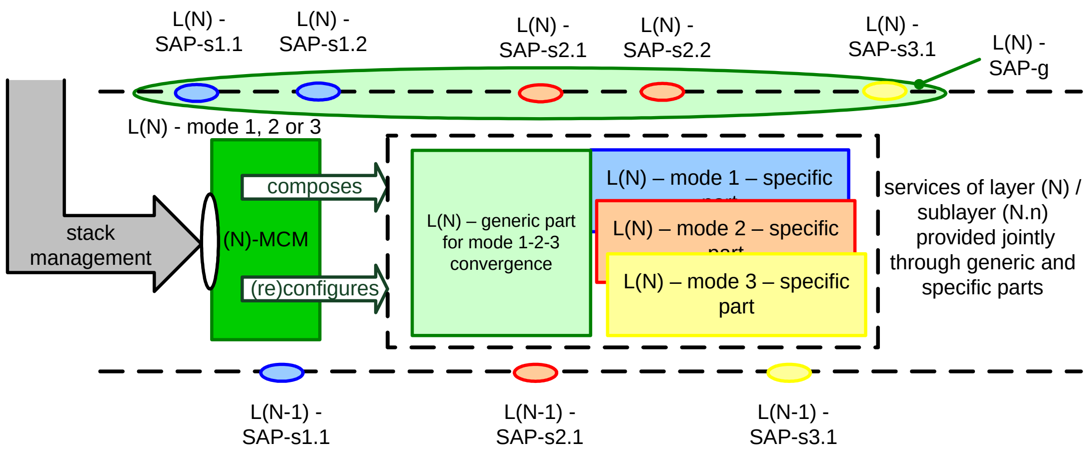
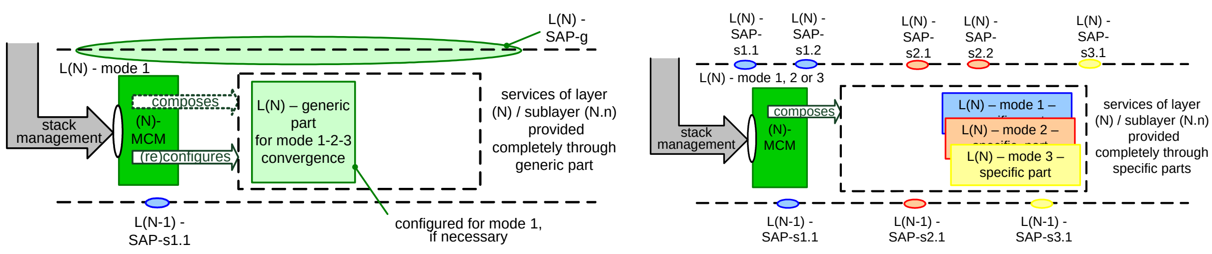

L(N) - SAP-s1.1
L(N) - SAP-s1.2
L(N) - SAP-s2.1
L(N) - SAP-s2.2
L(N) - SAP-s3.1
L(N) - SAP-g
L(N) - mode 1, 2 or 3
stack management
(N)-MCM
composes
(re)configures
L(N) – generic part for mode 1-2-3 convergence
L(N) – mode 1 – specific part
L(N) – mode 2 – specific part
L(N) – mode 3 – specific part
services of layer (N) / sublayer (N.n) provided jointly through generic and specific parts
L(N-1) - SAP-s1.1
L(N-1) - SAP-s2.1
L(N-1) - SAP-s3.1

Figure 25: Composition of a layer (N) from generic and specific functions. The composition and (re-)configuration is handled by the (N)-MCM. The (N)-MCM is controlled by a layer-external stack management entity, namely the Stack-MCM.

Figure 26 depicts two exemplary cases of a (N)-layer: In Figure 26(a) a completely generic layer configured (if necessary) for mode 1 by the (N)-MCM is shown. The common, generic part is thereby composed by the (N)-MCM. Contrary Figure 26(b) illustrates a layer, where the different integrated modes exhibit no commonalities. Here, the layer cannot provide generic services. The role of the MCM is restricted to support the Stack-MCM by choosing the demanded mode specific part.



(a) completely generic **(b) mode specific without common part**

Figure 26: Exemplary composition of a layer (N). Two extremes are depicted.

3.3.3 "Mode Transition" vs. "Mode Coexistence"

The Stack-MCM administers the protocol stacks of several modes under consideration of the environment, i.e., receivable modes at the location of a device and requirements of the user and its applications. The reconfiguration of the protocol stack from one mode to the other is referred to as mode transition. The mode transition may be done on several levels depending on architectural constraints of the protocol layers and the general treatment of generic parts: In case of a unique permanent existing DLL [109] of functions which are reconfigured (re-parameterized, restructured or extended) the mode transition is limited to the layers below – the MAC and PHY. The same stands for the case of a protocol stack of two modes used for relaying as introduced below: The termination of an end-to-end retransmission protocol above the relay limits the considered protocol layers during transition from one relaying-mode to another. Thus, mode transition may be done (i) on MAC level or (ii) on RRC level depending on termination of generic parts of the DLL.

Temporal or permanent in parallel existing modes of a protocol stack are referred to as mode coexistence. Mode coexistence can be reasoned through, (i) a relaying function in case of two simultaneous existing modes or (ii) the simultaneous connection to multiple modes for other reasons (a dedicated mode for broadcasts, applications of the user with different QoS requirements, cost preferences from user, etc.) or (iii) short-term coexistence for inter-mode handover and (iv) seamless mode handover.

3.3.4 Functions of the (N)-Layer Modes Convergence Manager ((N)-MCM)

3.3.4.1 Protocol Convergence

The convergence of multi-mode protocol stacks has two dimensions: First the convergence between two adjacent layers, in the following referred to as vertical convergence as it is known from the user-plane of H/2 protocol stack. Second the convergence between layers located in the different modes of the protocol stack which have the same functions: In the following referred to as horizontal convergence. The generic protocol stack, managed by the (N)-MCM as introduced above, enables both the horizontal as well as vertical protocol convergence.

From the perspective of higher layer protocols the multi-mode protocol stack is transparent on the user- as well as on the control-plane, i.e., generic parts terminate the stack to the layers above, as depicted in Figure 24. The vertical conversion of the (N)-MCM implies the adaptation of the on the multi-mode protocol stack working packet data protocols to the specific mode. This may be for instance the conversion of an IP datagram in compressing the IP-Header.

3.3.4.2 Layer Composition and Reconfiguration

The separated approach of generic and specific parts requires an administration when taking the transition between modes into account: The common generic parts of the old mode need to be adopted for being reused in the new mode of the protocol stack. It is assumed that the generic parts of a layer exist permanently and are to be reconfigured and/or recomposed by the (N)-MCM corresponding to the characteristics of the targeted new mode. This assumption may imply a module-based composition concept of the generic parts as introduced in [124] and [125]. The composition and configuration of the layer out of generic and specific parts, see Section 3.3.2, is done by the (N)-MCM.

3.3.4.3 Data Preservation and Context Transfer

The communication between two modes and the mapping between generic and specific parts of a mode is done by the (N)-MCM (inside layer) and the Stack-MCM (transfer between modes). The transition between two modes can be optimized in using the data from the old mode to the new mode. The ability of a user-plane protocol to reuse status information in the generic part and protocol data after transition to another mode requires an extension of the protocol into the control-plane though it performs only user-plane tasks. Depending on the status of the related protocol parts, the data transfer is referred to as “data preservation” or “context transfer” as illustrated in Figure 24: If the generic part is reconfigured and recomposed the data needs to be preserved, i.e. adopted, to the new mode. In the case of a deletion of the old specific/generic part the data transfer is named “context transfer” which implies preservation for the new mode.

3.3.5 Functions of the Stack Modes Convergence Protocol (Stack-MCM)

3.3.5.1 Joint Radio Resource Management

The functions of the user and control-plane are administrated by the RRM which may be coordinated centralized or decentralized and the RRM decisions are executed by the RRC of the corresponding modes. The RRM may assign multiple modes to one specific data flow. The RRC provides status information about the mode specific protocol stack in a generic structure to the RRM of the multi-mode protocol. In case of a (semi-)centralized coordination of the radio resource allocation, this generic information structure about the status of the different modes of the protocol stack can be transmitted to enable an adequate decision.

The RRM of a single multi-mode device may also support the coordination across neighbouring operating devices as for instance the coordination across base stations.

3.3.5.2 Inter-Mode Scheduling

The Stack-MCM as intermediator between modes performs scheduling among different modes, as illustrated in Figure 24. Contrary the scheduling inside a mode across logical/transport channels: It is done in MAC-g or RLC-g of the specific mode’s protocol stack. The inter-mode scheduling considers the dynamic scheduling of different user data flows over multiple modes. The scheduling strategy may for instance be based on the modes’ interference situation which requires a provision of necessary information directly from the PHY if the decision is done in the MAC independent from RRC/RRM. This information about the quality of the radio link is again provided in a generic information structure.

3.3.5.3 Self-Organization

The envisaged communication system is able to autonomously decide about its radio resource allocations in taking the environment into account. This implies the for instance the adequate selection of frequencies used for transmission or the routing of user data packets. The radio resource is selected under consideration of interference avoidance with other radio systems. Further, the optimized spectrum utilization coordinated with neighbouring radio systems of the same technology is taken into account, which may also be related to efficient multi-hop relaying depending on the selected deployment scenario. The self-organisation comprises scenarios of breaking down and installation of additional devices in an operating communication system. The Stack-MCM has to support the addressed functionalities in activating for instance different modes to provide information about the interference situation or the role of devices (if it is acting as relay or access point) in reception range.

3.4 Definition of a Generic and Reconfigurable Link Layer

In mobile communications the radio link is in general the bottleneck of the end-to-end path and it is costly or even impossibly to increase its capacity. Therefore it is required to utilise the available radio resources in the most efficient way. This role is performed by sophisticated radio physical layers and radio link layers, which are optimised to the radio access technology in use. In Figure 27 (left side) a simplified protocol stack is depicted, in which a correspondent node communicates with a mobile terminal. The end-to-end connection is established with e.g. the Internet Protocol (IP). The radio link layer (here called GLL) and the radio physical layer (PHY) enable data transmission over the radio link.

Assuming that the radio link is the bottleneck, most data which is currently in process of being transmitted along the end-to-end path is typically either queued or being processed within the radio link layer. This approach has proven to be an efficient design for communication in today”s mobile communication networks, where a specific mobile network with a specific radio access technology is used. However, there are certain limitations in the context of co-operating networks, in which seamless communication is desired via a multitude of mobile networks, which may deploy different radio access technologies.

This scenario is depicted in Figure 27, where a mobile terminal dynamically selects one of the available radio access networks (RAN A or RAN B) during a session. Each radio access network uses its" specific radio link layer and radio physical layer (left side of Figure 27). During an inter-system handover from RAN A to RAN B, the radio link in RAN A is torn down and a new radio link is being set-up in RAN B. Such a handover can only be lossless if a further layer of error recovery is applied, e.g. end-to-end on top of IP. But even then an inter-system handover is neither efficient nor without disruption of the service.

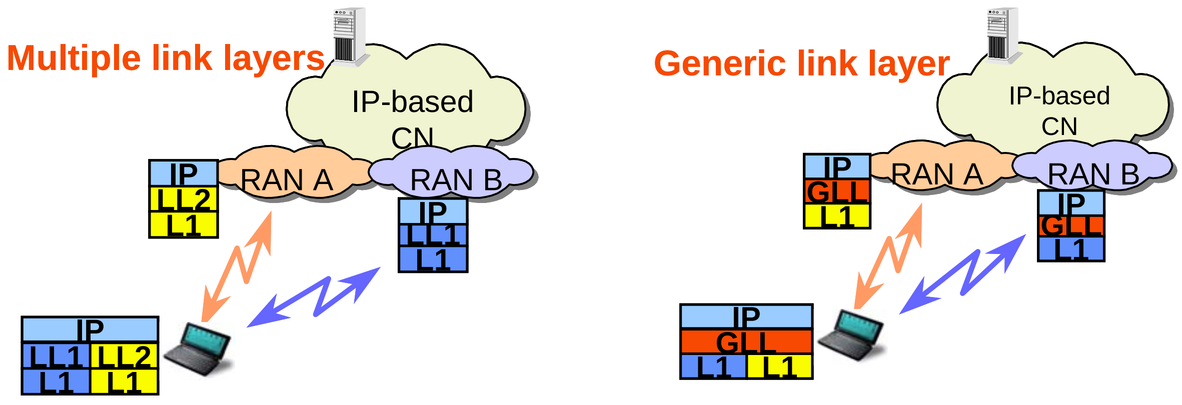


Figure 27: Multiple Link Layer Scenario compared to Generic Link Layer Scenario (GLL – generic link layer, LL – link layer, L1 – layer 1, RAN – radio access network, CN – core network)

Since different radio link layers have in general the same functionality for all radio access technologies, this problem can be solved if the radio link layers are made compatible. The old radio link layer state can then be handed over to the new radio link layer, which continues the transmission in a seamless way. This is achieved by defining a *generic link layer*, which can be used as radio link layer for all radio links (right side of Figure 27).

The *Generic Link Layer* (GLL) is a specified radio layer protocol, which provides the link layer functions required in every radio link layer [108]. It can be configured in a flexible manner to perform these link layer functions in an optimised way for different radio access technologies with different properties. The generic specification of radio link layer functions enables reconfiguration of the generic link layer in which the existing communication context at time of reconfiguration is transformed into a new context within the new configuration. As a result the communication session can "survive" the reconfiguration procedure lossless and without disruption. From a service perspective it is a seamless reconfiguration.

The GLL concept requires a reconfiguration of the GLL on both sides of the wireless link, in the mobile terminal as well as in the radio access node. This is required in order to seamlessly continue with the old context of the communication. To implement a GLL following interfaces and reference points have to be define (Figure 28):

- The higher layer interface: Via the interface to the higher protocol layer data is received for transmission and delivered after reception. This interface further allows to configure the QoS requirements for the transmission of higher layer datagrams,

- The physical layer interface: At the interface to the physical layer radio blocks are sent to the physical layer for transmission over the radio link,
- The control interface: Via control interface the generic link layer is configured and reconfigured,
- The internal interface to embed specific functions: Via this interface it shall be possible to include a specific function, e.g. a ciphering algorithm, into general functions.

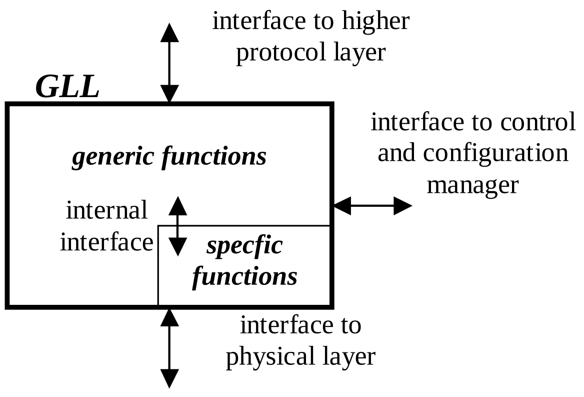

interface to higher protocol layer
GLL
generic functions
interface to control and configuration manager
internal interface
specfic functions
interface to physical layer

Figure 28: Generic Link Layer Functions and Interfaces

3.5 Generic Protocol Functions as Parameterizable Modules

3.5.1 Modular Approach – the Generic Protocol Stack as Toolbox of Protocol Functions

The generic protocol stack is the realization of the common parts as introduced above and implements its common functions based on modules. These common protocol functions get their system specific behavior based on parameterization. Once specified, these modules can be repeatedly used with a different set of parameters corresponding to the specific communication system. The modules of generic protocol functions form together with system specific modules a complete protocol layer, as depicted in Figure 29.

The communication inside said layer is performed by employing generic service primitives and generic PDUs, which are also considered as being a part of the generic stack, see again Figure 21. The functional modules form a toolbox of protocol functions as introduced in [124] and [125].

A unique manager as well as interfaces for the Service Access Points (SAP) to the adjacent layers complete the fully functional protocol layer as depicted in Figure 29. In detail, the mentioned components have the following tasks:

- **Functional module** (generic or RAT specific): Realizes a certain fundamental functionality as black-box. In case of a generic module, a list of parameters for characterizing the functionality is given and the underlying functionality is hidden. The comprehensiveness of the fulfilled function is limited to fit straightforward into a single module.
- **Manager, (N)-MCM:** Composes and administrates the layer during runtime. This implies the composition, rearrangement, parameterization and data questioning of the functional modules. Additionally, the manager administrates the layer internal communication, as for instance the

connection of the layer's modules through generic service primitives. It is the layer's counterpart of the Stack-MCM as introduced above and realizes the reconfigurability of the layer.

- **Interface:** Translates the generic service primitives with specific protocol information as payload to system specific ones and enables thus the vertical as well as horizontal integration of the system specific parts of the layer.
- **Service Access Point (SAP):** Here, services of the layer are performed for the adjacent layers. The layer may communicate via generic primitives without a translation interface to an adjacent layer if said layer has the same modular composition. The interface is needed if it is demanded that the layer appears as a classic layer fitting into an ordinary protocol stack.

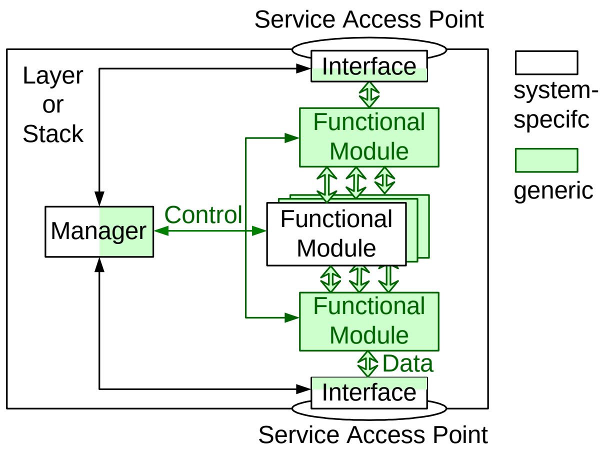


Figure 29: Composition of a protocol specific layer or sublayer on the basis of generic and system-specific functional modules

This approach enables the dynamic protocol reconfiguration on several levels: A single (sub-) layer as well as a complete protocol stack can be composed out of the introduced parameterizable modules.

3.5.2 Generic Protocol Functions of the Data Link Layer

As the architecture of modern communication protocols cannot be forced into the classical layered architecture of the ISO/OSI reference model, it is rather difficult to identify similarities and attribute these to specific layers. Therefore, this paper deepens the level of examination in the search for similarities and considers fundamental protocol functions, contrary to [104] and [122] where complete protocols are analyzed for genericity. Though these protocol functions mainly correspond to the DLL as specified in the ISO/OSI reference model, they can be found in multiple layers of today's protocol stacks as shown below. The following functions are considered for the generic protocol stack:

- Error handling with the help of Forward Error Correction (FEC) or Automatic Repeated reQuest (ARQ) protocols as for instance Send-and-Wait ARQ, Go-back-N ARQ or Selective-Reject ARQ
- Flow control
- Segmentation, concatenation and padding of Protocol Data Units (PDUs)
- Discarding of several times received segments*
- Reordering of PDUs
- Multiplexing/De-Multiplexing of the data flow, as for instance the mapping of different channels
- Dynamic scheduling
- Ciphering

- Header compression

The composition and performance evaluation of specific layers out of these modular protocol functions can be found in [124] and [125].

3.5.3 Parameterization of Functional Modules

In this context parameterization implies not only specific values, as for instance the datagram size of a segmentation module, but also a configuration of behavior and characteristics of a module, as for example the concretion of an ARQ module as a Go-back-N ARQ protocol with specified window sizes for transmission and reception. This implies as well a configuration of the modules' interface to the outside. The parameterization of functional modules may imply (i.) a specification of certain variables, (ii.) the switching on/off of certain functionality/behavior and (iii.) an extension of the module's interface to the outside.

At the example of the ARQ module, the parameterization may imply among other things:

- ARQ protocol characteristic, for instance Go-Back-N ARQ or Selective-Reject ARQ
- Transmitter or/and receiver role
- Receive and transmission window size
- Fixed, variable (TCP) window length or open/shut mechanism (LLC)
- Timer value, after a packet is assumed to be lost
- Connection Service: inexistent (UMTS RLC), separated for each direction (802.11 - CSMA/CA with RTS/CTS), 2-way handshake (GSM LLC) or 3-way handshake (TCP)
- Use of Negative ACKnowledgments (NACKs)

The introduced ARQ module is described and analyzed in detail in [124].

3.6 Multi-Antenna based Approach for Adaptive Data Transmission

The research in the field of wireless mobile communications leads to variety of different standards for networks with high data rates, high capacity, high QoS, security and flexibility for a highly mobile user. This imposes the need to integrate all of them to form a global heterogeneous network, characterized by ubiquity and flexibility of service. The infrastructure should provide users with variety of possible air interfaces which can be chosen according to current propagation and interference conditions.

There are a lot of parameters that should be considered in design: the asymmetric traffic in the downlink and the uplink, the nature of the transmitted data and type of service, battery life time for mobile users, system load, cost etc. Since the available frequency spectrum is limited, high spectral efficiency is the major issue to be addressed. For treating all of them, new reconfigurable integrated system concept needs to support a number of advanced technologies; for physical layer design, certainly smart antenna technologies (beamforming and Multiple Input - Multiple Output (MIMO)) [130], [131] and [132].

With the higher demands for per-user data rates, better coverage and QoS for highly mobile users, besides support from higher layers, more powerful technologies in physical layer are needed. Smart antenna technology will with no doubt play important role in future wireless systems. Using smart antennas has proved to give many benefits, including:

- interference rejection,
- higher capacity,
- increased cell throughput,
- decreased required power,
- channel robustness,

- decreased per bit cost, etc.

The drawbacks are more complex transceiver and radio resource management, and higher layers.

Channel characteristics which have to be taken into account for smart antenna techniques are [129]:
- Delay spread (describes the time dispersion of the channel, and tells how fast the channel decorrelates with frequency (correlation bandwidth)),
- Doppler spread (describes the frequency dispersion of the channel, is determined by mobile velocity, and tells how fast the channel decorrelates with time (correlation time)) and
- Angular spread (describes the angular dispersion of the channel and tells us how fast the channel decorrelates with distance).

These parameters vary for different environments. Algorithms and antenna array geometries give different performance depending on the environment, but also depending on the accuracy of Channel State Information (CSI) at both receiver and transmitter. Accurate CSI at transmitter can give great benefits when using smart antennas. The problem of fast transfer of CSI in TDD and FDD based systems with smart antennas is addressed in [133].

Active research in the field of smart antnenas has led to a range of different solutions for particular systems and scenarios. A common classification of smart antenna technologies differentiates two approaches in using antenna arrays in communication systems: beamforming, when signals received on antenna elements are highly correlated and spatial multiplexing for uncorrelated received signals. In these two classes, there are many different concepts developed and they are still under research.

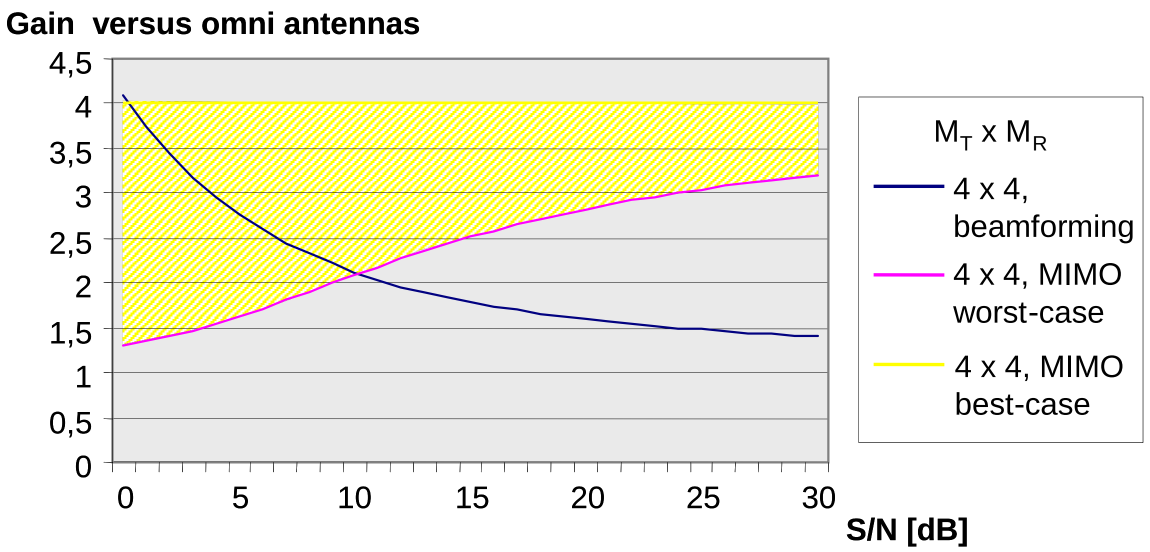


Figure 30: Estimation of MIMO & beamforming capacity: MT x MR = 4 x 4

Under high interference, beamforming is much superior to MIMO; whilst MIMO is much superior to beam forming under low interference, see Figure 30 [127]. Beamforming techniques reduce total transmit power by focusing the beam to the intended user and in that way reduce interfering other stations. On the other hand, MIMO gives support for high data rates taking advantage of multipath in wireless channels.

Propagation conditions for wide-area coverage will typically show relatively low angular spread at the Base Station (BS). On the other hand, the angular spread at the mobile terminal is often high (except for line-of-sight conditions) and thus low correlation can be achieved with relatively small element separation (and of course when using polarization and/or pattern diversity). The probability for (relatively) high velocities is high for wide-area coverage. Although it can be feasible to acquire and exploit short-term CSI at the transmitter for some low-velocity users in the coverage area, it is likely that for most terminals only long-term CSI and/or highly quantized short-term CSI can be made available to the transmitter. Therefore, the closed-loop techniques that require very accurate short-term CSI are less applicable. On the other hand, techniques that allow the restriction of the instantaneous knowledge to a reduced set of quantities (beamforming) can still be feasible up to a certain terminal velocity. Particularly attractive are also closed-loop techniques which only apply long-term channel knowledge and open-loop techniques that do not require any channel knowledge apart from maybe a low-rate feedback to switch between transmission modes.

Major differences between the appropriate concepts for short-range scenarios and the wide-area coverage case arise from the differences in the propagation conditions (with respect to space and frequency selectivity) and in general lower terminal velocities (which influences the time selectivity characterized by the Doppler spectrum). While the angular spread at the BS is typically very low for wide-area deployment (BS above rooftop level), the angular spread is often much higher for indoor/outdoor short-range communications, leading to reduced correlation at BS/AP (access point) for fixed antenna separation (or reduced size of arrays for similar correlation, where of course pattern/polarization diversity can be used to further decrease the array size). Use of spatial diversity and/or spatial multiplexing gain is simplified. On the other hand, due to the broader angular spread (and hence lower antenna correlation), beamforming is less effective here.

Maybe even more important than the changes in the spatial characteristics is the reduced time variability of the channel (assuming much lower velocities of typically below walking speed 5km/h). Schemes which require high-quality short-term channel knowledge may be feasible here. On the other hand, open-loop signal design or even non-coherent transmission is not appropriate here. Different MIMO schemes could be appropriate here: Multi-User (MU) linear/non-linear precoding for MISO/MIMO downlink, linear/non-linear closed-loop MIMO, open-loop vector/matrix modulation is of interest above pedestrian speed, etc.

The transmitter in the Figure 31 [128] includes a generalized beamformer. For the selection of the BF matrix different options are available. For example, if no explicit CSI is estimated at or fed back to the TX, but it is known that the transmitter is situated in a rich-scattering environment, the particular choice of having no beamforming and hence no scheduling, link adaptation, or space-time mapping in element space, may be the appropriate choice. Under the same conditions, but for a TX well above rooftop level, a preselected set of beams is likely to be superior. Similarly, there are propagation conditions and applications for which user-specific long-term beams or MIMO is more appropriate. In any case, scheduling and/or link adaptation has to be performed for the parallel inputs to the beamformer. Scheduling and Radio Resource Management (RRM) decides on the scheduled users, subsets of beams assigned to selected users and is combined with other link adaptation mechanisms. As intuitively clear, the functions implementing RRM, scheduling, link adaptation and selection of the beamforming matrix are strongly connected and the challenge is to find a good trade-off between performance, complexity, and signaling overhead for this joint optimization problem for the range of scenarios and the related channel conditions.

It is clear that there is a fundamental trade-off between the gains due to spatial multiplexing and Space Division Multiple Access (SDMA). With new technologies, physical layer should not stay isolated, but as a rich source of performance improvement potentials, tightly connected with MAC and higher layers. For best network performance, integrating physical layer with the higher layer protocols is necessary. Medium Access Control (MAC) layer has to be upgraded with respect to spatial dimension of channel. Support from higher layers is a necessary for an air interface to dynamically adapt to current propagation and interference conditions.

There is no single solution that is applicable to all types of environment, and therefore reconfigurable architecture is needed, including multiple air interface standards. Highly adaptive and reconfigurable radio interfaces are needed to adequately respond to changes in radio propagation and the interference environment.

After detecting changes in the environment, an algorithm that could give better performance (then the current one) under current fading and interference conditions should be chosen. First level is adapting to current conditions: measurement results processed in MAC (or higher) layer lead to this decision. This does not go further from changing some parameters while keeping the basic algorithm the same or switching to another known one.

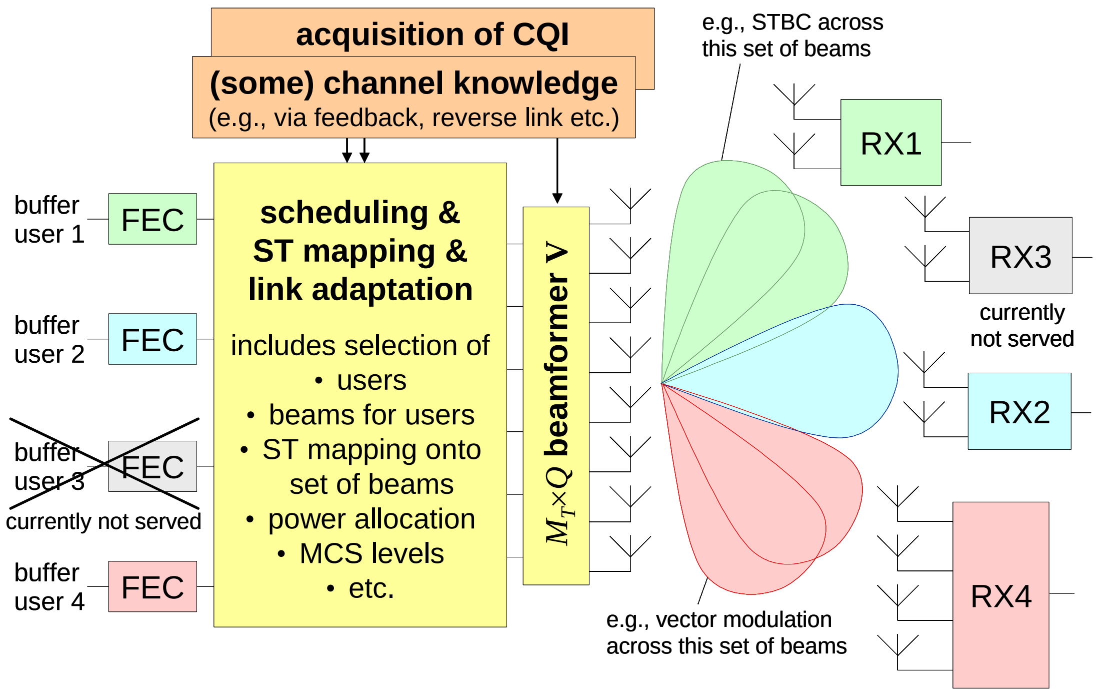


Figure 31: Block diagram for (flat-fading) multi-user MIMO downlink with generalized beamforming (BF matrix V) and space time mapping of user data onto subsets of beams. Scheduled users, BF matrix, subsets of beams, space-time mapping, power levels, and modulation and coding schemes are adapted according to the scheduling/link-adaptation strategy and the available CSI.

Second approach is the genuine reconfigurability; it is the ability to learn about new elements of physical layer (coding, modulation techniques) and adopt them by pure software update. Therefore it is necessary for the terminals to have signal processing modules which can be easily reprogrammed for different coding and modulation scheme. Most of the signal processing should be performed in software and re-configurable digital hardware. Stations which serve as access points should be simultaneous-multi-mode capable. For terminals such capability is also useful for simultaneous access to different services. Having multi-mode capability should not mean multiplication of radio and signal processing modules, but a single module for all air interfaces.

Context awareness should provide user's device with the ability to dynamically choose air interface to use, based on propagation conditions and scenario, user's location and needs, real time communication, location specific services, security and reliability demands. Merging software and hardware solutions will result in realizing reconfigurable network with multimode solutions for improved services with lower per-bit price. From system point of view, in order to further improve the network performance, flexible air interface resource allocation should enable the network to allocate dynamically resources according to the operator's strategy, network load, prioritization, costs, etc. Legacy, privacy and security are also the issues which should be considered in an integrated systematic approach.

4 SDR

4.1 Introduction

Future mobile communications systems have been studied for a long time, leading to the emergence of various cheap and efficient communications devices: GSM/GPRS, UMTS, 802.11, DVB-T, Bluetooth… Integration of the various, associated, protocols into a global framework, enabling a transparent seamless diagonal handover, is a key objective to End-to-end reconfigurability. During the last decade, reconfigurability has been evangelized by techno-addicts advocating that technology could achieve radio reconfigurability. Joe Mitola mentions (about Software Radio): “A software radio is a radio whose channel modulation waveforms are defined in software”. Such an approach can be described as techno-oriented because it refers to digital convertion and processing power enhancement.

In its beginning, Software Radio used to be compared to the PC concept: a standard platform composed of cheap boards provided by multiple companies. This paradigm has been observed with the emergence of the Personal Digital Assistant (PDA) and its ensuing success – ensuring a market corresponding to the need of the “easy to use” and “easy to transport” “computer”. The PDA concept has introduced a different way of using a computer without a keyboard and mouse, using instead a stylus. The success of the PDA rests on the use of a common hardware and software platform. As soon as a platform has been stable and made “open” to software developers, an increased number of software programs are created - enabling value added services. Despite these first concepts on what Software Defined Radio might be, the PC paradigm has yet to be fully exploited in application to the mobile phone market which is now facing a new phase of its development.

The convergence of mobile phones with PDA-type technology and the emergence of Smart-phones begin to pave the way for Software Defined Radios or terminals. Mobile phones have become, and are becoming increasingly a way for a user to display their own identity and many companies target various market segments in terms of the products themselves (for example producing a line of phones for the Business User, another for those who are Fashion conscious and perhaps phones aimed towards the youth end of the market, incorporating the latest computer games). Mobile phone platforms are becoming more flexible as they evolve to cater for the increased expectations of their users. The ability to change the mobile device’s cover exists, along with downloading ringtones, games, etc. However, although the emergence of an all-in-one concept is being witnessed, the underlying communications technology is currently not reconfigurable.

Within the past few years extensive research on reconfigurability has been conducted. A very strong heritage in reconfigurability was gained through former EU-IST FP5 projects such as DRIVE, OVERDRIVE, TRUST, SCOUT, CREDO and MOBIVAS, where the expertise in the functions offered to user terminals, applications and services, was capitalized. Each of these projects concentrated on a variety of different technical aspects such as terminals, value-added service provision, enabling technologies, applications, reconfigurable devices, network provisions, security, proof of concept of reconfigurability.

WWRF has defined SDR Reference Models and has analysed issues and problems with respect to SDR Architectures in a recent White Paper, titled “Reconfigurable SDR Equipment and Supporting Networks” ([120]), that has been produced within the WWRF WG3 SDR Group ([118],[119]). Several existing SDR system and supporting networks reference models and architectural approaches were reviewed. The paper also pointed out some of the main research areas on SDR for the next decade. The work presented in this section is complementary to the previous work mentioned above. In this direction it addresses an important relative research topic, hardware abstraction. This issue is currently widely discussed in the reconfigurability community (e.g. specific RFI in the SDR Forum) as physical implementation will be proprietary to manufacturer. In this direction, this paper discusses on Hardware Abstraction in an End-to-End Reconfigurable Device and presents a possible hardware abstraction approach (Section 4.2).

A further issue with respect to ongoing and future work on the exploration and development of SDR and of new radio interfaces in general is performance validation. This is understood not as the testing of new interface features by the company or group which develops them, but as third party verification. This is an issue of special importance for telecommunications operators, who additionally have to test interoperability between equipment and systems coming from different manufacturers. From this background, a need can be seen for the implementation of B3G verification tools. Section 4.4.1 describes a test bed targeting to the evaluation of new air interfaces and interoperability testing. This test bed, referred to as the PRAGA platform, has been developed at Telefonica and is based on flexible, modular and upgradable SDR transceivers.

4.2 Hardware Abstraction in an End-to-End Reconfigurable Device

4.2.1 Hardware Abstraction Reasoning

In the past the configuration control for the physical layer in a wireless terminal has been limited to changes requested by the Radio Resource Controller (RRC) or similar entity defined in the supported standard. These changes would be limited to a subset of the standard, as defined by the terminals class mark, and would include services such as switching between data services and speech calls or a change of speech codec. Each of these different configurations would be known at design time and would be created and tested in DSP/CPU software (sometimes in assembler code but more recently in C) and then implemented in an embedded ROM. A ROM solution was used rather than RAM or FLASH to save silicon area and allow operation at the high clock speeds required in a data processing application. Typically the only mechanism available for modifying the physical layer to implement functionality above and beyond that conceived at design time was a small amount of patch RAM in the program space of the DSP. This was combined with some form of patch vectoring and used to fix bugs in the ROM code.

More recently research projects have been looking at methods for reconfiguring the physical layer to implement functionality not conceived of at design time. Earlier projects such as TRUST, SCOUT and CAST have directly addressed the problem of Configuration Control. A common theme exists i.e. all projects have used object-modelling techniques to encapsulate functionality and all have three primary components, as shown in Table 1.

Table 1: Comparison of Configuration Components in FP5 projects with E2R

TRUST	SCOUT	CAST	E2R
BPC (Baseband Processing Cell)	Proxy	Java Class	[CEM] (Configurable Execution Module)
TMM (Terminal Management Module)	TMM	RSC (Reconfigurable Resource Controller)	[CMM] (Configuration Management Module)
RMM (Reconfigurable Baseband Management Module)	RMM	PLC (Physical Layer Controller)	CCM (Configuration Control Module)

Both SCOUT and CAST address heterogeneous architectures and recognise the requirement to support communication between the processing elements. Only SCOUT appears to have addressed the issue of timing deadlines. ADRIATIC addresses this problem indirectly by working on the re-configuration times of the processing elements. None of these projects directly address the problem of verifying that a new configuration will always operate correctly. MuMoR reduce this problem by constraining the configuration space to a number of RAT"s.

4.2.2 Hardware Abstraction Possible Approach

The physical layer architecture of a reconfigurable device consists of a set of reconfigurable functional elements (RF-frontend, communication, digital processing). They appear as (re)Configurable Execution Modules (CEMs) implementing a specific functionality (e.g. Down-conversion, Modulation, Decoding, Rake). A specific entity (called Configuration Control Module, CCM) reconfigures such modules and also manages the communication resources between to guarantee the required functionality and maintain the required data throughput.

Configuration Control Module (CCM)

The role of the CCM is to supply an abstract configuration interface to the signal processing system in a wireless terminal, base station or access point.
The CCM is located in the System Abstraction Layer (SAL) or high level Hardware Abstraction Layer (HAL). This layer implements a set of interfaces that allow high level entities to configure resources, i.e. create and link signal processing functions (e.g. modulation, demodulation, channel coding, source coding, RF transceiver, AFE, etc.). By supplying a platform independent interface, called Service API, a system can reconfigure either ends of the wireless link without a detailed understanding of the underlying implementation.
The configuration of the wireless terminal includes downloading of software (ISA objects, FPGA code, etc.) and parameters into resources to implement different functions as well as the configuration of the communication fabric to implement the required data flow and control flow structures.
Control and supervision of the re-configuration process takes place in co-operation with the Configuration Management Module (CMM) and/or by autonomous processes implemented by dedicated CCM functions and optional modules like the adaptation module.
The CCM will be implemented by means of software modules, build on top of the underlying operational software, i.e. a certain operating system (OS), using its extensions like device driver to gain access to the real hardware interfaces or additional OS extensions to implement diverse CCM functionalities.

Configurable Execution Modules (CEMs)

The Configurable Execution Modules may support different levels of reconfiguration (configuration capabilities):

- Dedicated signal processing entities, programmable IPs blocks
- Algorithm specific accelerators
- Programmable logic (e.g. FPGA, CPLD)
- Application tailored DSPs, AIPS
- General purpose processors, DSPs

A manufacturer will exercise the choice and selection of suitable CEMs, implementing certain physical layer functionality. This manufacturer dependency requires an implementation independent view of the physical layer hardware in order to support certain reconfigurations and use cases.

Hardware Abstraction Layer Concept

Hardware abstraction will be used on various levels to support the Configuration Control Module and the Reconfiguration Management plane on top. In particular, the exchange of reconfiguration capability parameters between certain entities (e.g. network entities, operator) has to be addressed.

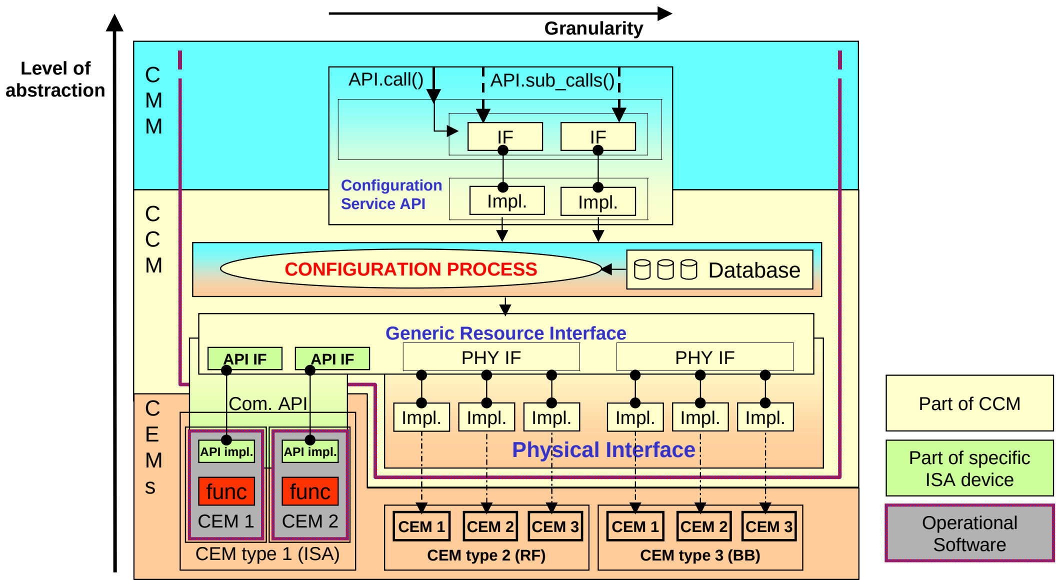


Figure 32: Abstract functional View

A hardware abstraction layer concept might be used, which introduces hardware abstraction on certain levels as shown in Figure 32:

- The Configuration Control Module can be seen as the high-level abstraction of the re-configuration capabilities of the physical layer hardware. Abstraction has to be done on function rather than implementation level. This level implements a kind of Service API.
- The Generic Resource Interface, abstracts the different classes of Configurable Execution Modules (CEMs) for the CCM.
- The Physical Interface abstracts the real implementation of a CEM
- Intelligent CEMs have at least a basic operating system running on top. Their physical interface will be replaced by a low-level communication API, which enables communication (i.e. exchange of configuration data, program code).

Configuration Service API

The *Configuration Service API* is the mentioned interface towards the upper layer between the Configuration Control Module (CCM) and the platform independent Configuration Management Module (CMM). The Configuration Service API is implemented in the CCM, and is used by CMM. This is a platform independent entity and provides services for the configuration. Different levels of configuration granularity can be envisaged:

- Coarse grained: "configure WLAN chain with param1 = x & param2 = y"
- Mid grained: "configure channel estimation = algo1"
- Fine grained: "configure FIR coefficient1 = 0.654"

Typically the upper level management entity (i.e. the CMM) is using the coarse-grained layer, but optimisation of the radio link or certain adaptation functions, may require the direct access to the mid grain or even the fine grain configuration layer, which could be done either by the CMM or additional adaption or radio link modules.

Inside the Configuration Service API, the coarse grain level will use functions of the mid grain level and the mid grain level those of the fine-grained. Thus, the architecture of underlying hardware is hidden, but assessable in an abstracted common way.

Generic Resource Interface

The intention for the definition of the Generic Resource Interface is to have a common single interface to the lower layer. Abstract HW models will encapsulate front-end, digital baseband resources respective functions and communication resources. The CCM has to interact with the resources available, which are executing the physical layer processing.

The physical implementation will be proprietary and might be organized in several different ways. Functional partitioning will lead into several possible allocations of functionality into multifarious CEMs. In the described approach for the way of reconfiguration, the CEMs can be grouped into two different interface classes:

- **ACTIVE**: ISA devices which have their own firmware respective Operating System (OS), and therefore an API for handling accesses. This is named the Low-level Communication API.
- **PASSIVE**: All other devices without their own firmware respective OS. They do not provide an API, and therefore need an active handling. This is referred to as low-level Physical Interface.

The Generic Resource Interface (i.e. the upper-level interface) is therefore subdivided in the different properties of generally different access-types of CEMs as shown in Figure 1.

Physical Interface

The Physical Interface is part of the low-level HAL for devices without own firmware or OS. They need an active handling of each configuration, which is demanded. Some examples are:

- Analogue circuits: e.g. AD-converter (configure required number of bits)
- Accelerator e.g. upload Microcode
- Programmable ASIC: e.g. Set certain register values & read status registers

The Physical Interface provides mapping of Generic Resource function calls to the proprietary configuration procedures of dedicated CEMs, i.e. the real physical addresses contained in the database are utilized for the re-configuration.

Low-level Communication API

The low-level Communication API provides common services for configuration of ISA devices running under certain OS. They are implemented on certain ISA device. ISA devices have their own firmware respective Operating System (OS), and therefore an API for handling the reconfiguration requests. The configuration through the Communication API includes downloading of software modules and parameters.

A UML (Universal Modelling Language) description of the described hardware abstraction is shown as a summary in Figure 33.

The previous sections have indicated an approach towards Hardware Abstraction in an End-to-End reconfigurable device. This approach will lead to a clear understanding of the requirements for each relevant abstraction layer within the reconfigurable physical layer and combine different classes of reconfigurable devices. This includes the exchange of capability parameters for the physical layer. The capability parameters are used by the user, service supplier and operator to decide on the most suitable terminal/basestation configuration for the given context.

The abstraction layer will translate an implementation specific configuration, proprietary to a certain manufacturer and specific to a certain implementation, into a functional description. These translations and descriptions have to be generic and flexible enough, in order to serve different requirements. At certain levels, harmonization between manufacturers is required.

Additional certain CEMs will be evaluated and modelled in an abstract way, in order to be integrated into the system architecture.

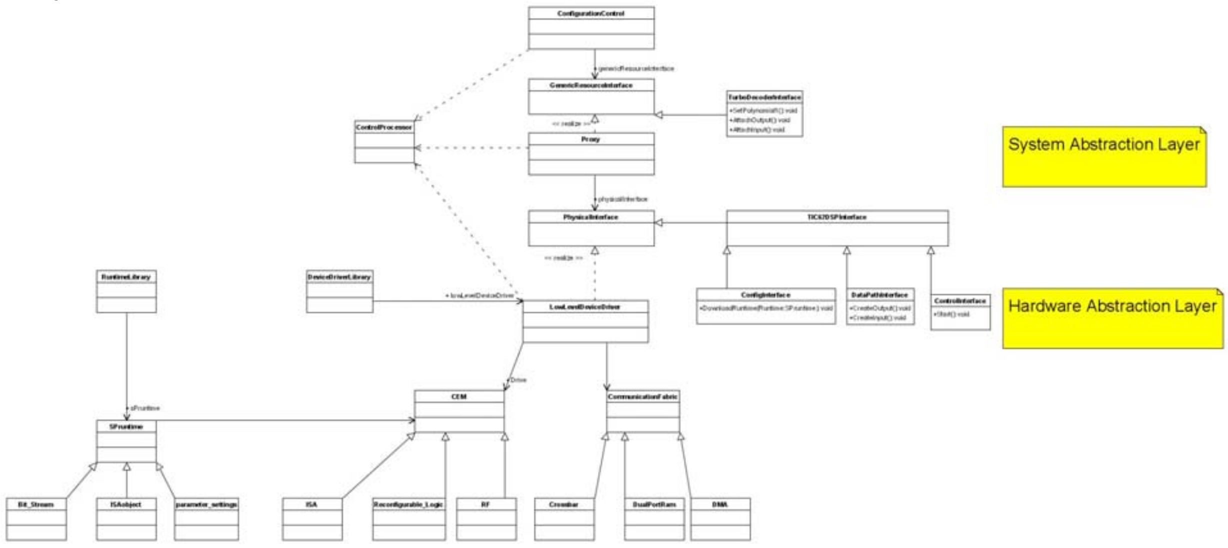


Figure 33 : UML Description of the Hardware Abstraction

4.3 Design Exploration and Integration Challenges for Reconfigurable Architectures

Most wireless physical layers are characterized by several signal processing blocks which are similar in their structure and functionality but differ in their implementations. The digital baseband processing complexity is dominated by channel estimation, detection and decoding in the presence of interference. Interconnect is a limited design resource needed for achieving parallelism in these architectures ([134]-[136]). In recent research, algorithms and architectures for these three major blocks ([137]-[145], [148], [149]) have been proposed for both base-stations and mobile handsets in cellular and indoor wireless LAN systems. The proposed fixed-function ASIC-like implementations simultaneously achieve high performance and area-power efficiency by exploiting special algorithmic and architectural structures, and it is important to preserve these features when developing reconfigurable systems.

With rapid development and adoption of advanced wireless systems, it is important to reduce the time to develop hardware research prototypes of these new algorithms ([150], [151]). It is well known that VLSI implementations consume minimal power but typically require long design cycles. On the other hand, DSPs are completely programmable but are in general unable to meet real-time deadlines and power budgets for high data rate mobile communications. A new class of application specific processors is an area of current research [152]. These architectures are programmable and may also be reconfigurable to allow for application specific instructions [153], [154]. FPGAs present an excellent platform for prototyping and evaluation of these architectures, particularly the class of heterogenous FPGAs that contain both programmable fabric and embedded processor cores. However, there are many new challenges in interconnect design to effectively integrate the embedded host and programmable fabric co-processor functions in these FPGAs, see Figure 34.

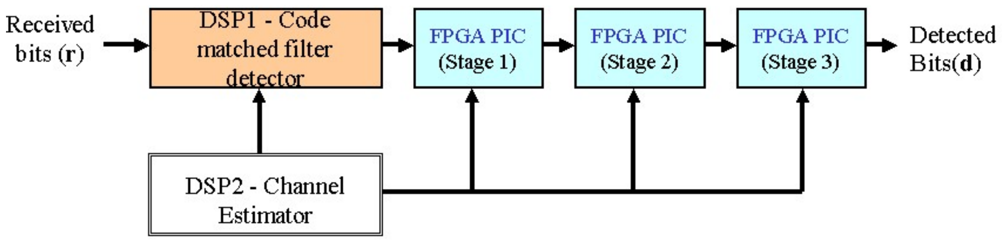


Figure 34: Parallel Interference Cancellation detector example on a hetergeneous DSP-FPGA system. Interconnect among modules may reduce system performance, after [5].

4.3.1 Design Exploration and Integration Challenges

Advanced signal processing algorithms contain a number of key numerically intensive kernels. Many of these kernels operate on streams of vectors or matrices and present challenges in interconnect and datapath design. As an example, wireless communication systems are composed of a pipeline of sub-algorithms with varying degrees of instruction and data parallelism. The key research innovations are in signal processing algorithm to parallel architecture mapping. As wireless systems both increase in data rate and require increased flexibility to adapt to changing environmental channel conditions, VLSI signal processing architectures are expanding from fixed-function ASICs and general programmable DSPs to application-specific instruction processors (ASIPs) and heterogeneous FPGAs with embedded processor cores and programmable fabrics ([155], [157]). The power and interconnect challenges of these systems through partitioning into clusters [158] and the design of host to co-processor interfaces [162] are being investigated. The Rice University CMC wireless testbed will contain several high-density Xilinx Virtex-II Pro FPGAs that will allow us to study the efficiency of these devices and to provide FPGA modules and design techniques for intra-chip and inter-chip communication.

As an example, our research in flexible Low Density Parity Check (LDPC) codes [159], has an initial prototype on standard FPGAs. LDPC is a highly parallel algorithm which requires large amounts of interconnect resources, and is therefore a research challenge in system integration. Our use of FPGA systems for rapid prototyping provides new core modules beyond those currently available from existing sources ([160], [161]). In the current research, it is planned to utilize the Rice testbed to verify performance and integration in heterogeneous FPGAs and to make these host to coprocessor interface modules available. Additionally, the simulation environment for SoC design exploration, which models the TI C64x DSP core with customizable programmable co-processors, will be descibed. The goals are to clearly and carefully model the software and hardware communication interfaces and bottlenecks in heterogeneous systems to improve the modularity and reusability of hardware and software for high data rate reconfigurable communication systems. It is important to model and simulate the data transfer between the hosts and co-processors so that the design of interface port and the communication link reduce the overall system cycle counts. A poorly design interface may actually reduce system performance.

4.3.2 Processor Challenges for 4G Systems

The ongoing definition of 4G wireless systems underscores several urgent challenges for system realization and design. Higher data rates and complex antenna and modulation schemes will require efficient algorithms and flexible chip architectures for baseband processing. Not only will area, time, and power consumption be the main design constraints, but also system reconfigurability and design reuse will be needed for fast creation of new systems. This flexibility will be key for MIMO antenna processing blocks that may be configured with a varying number of antennas.

As data rates have increased, a single DSP processor is not capable to keep up with the signal processing requirements of advanced algorithms. Multiple DSP processors do not have the power efficiency to exploit the instruction and data parallelism in the algorithms. A solution to this DSP limitation has focused on custom ASIC co-processor designs clustered around a programmable DSP host. However, the multiple co-processor solution requires extensive design and verification time and with the spiraling costs of ASIC fabrication, these systems are increasing in cost and have little ability for re-use or modification. The class of application specific instruction set processors has recently emerged to fill this middle ground between DSP and ASIC solutions. For example, stream processors, as shown in Figure 35, contain multiple SIMD-like clusters with specialized memory systems. Our recent research on chip equalizers for HSDPA and 1X-EV-DO systems and LDPC decoding has shown the flexibility of these ASIP solutions which will lead to System on Chip (Soc) processors. From design exploration of parallelism and flexibility requirements, these future SoC processors will be an efficient integration of DSP, ASIP, and ASIC structures.

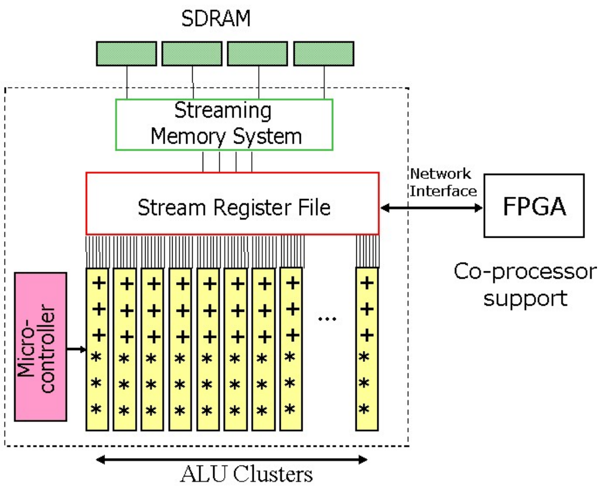


Figure 35: Programmable Multi-cluster Stream processor architecture with FPGA co-processor highlighting interconnect challenges between functional units and register files, after [1].

For high data rate 4G systems operating at greater than 100 Mbps, there are several research trends that are emerging that will require significant advances in VLSI architectures to provide performance with limited area, time and power constraints. For many MIMO OFDM systems, effective coding will be important. In [163], an iterative turbo-like system integrating LDPC decoding with the receiver is presented. However, for outdoor highly mobile environments, equalization will be needed even in OFDM systems ([164], [165]). Furthermore, parallel interference cancellation can also be used to enhance these MC-CDMA systems [166]. With the focus in 4G on adapting OFDM, there can also be benefits from applying techniques from WLAN applications, and also from low-power UWB OFDM architectures [167].

In addition to the flexibility needed for adaptation as 4G standard continue to evolve, there is growing interest in system reconfigurability and design modularization and reuse. However, system reconfigurability typically introduces overhead (lower data rate and higher power) that may not be acceptable in many applications. The commercial aspects of the military XG and JTRS hardware research are leading to cognitive radio and also are studied in the European E2R end to end reconfigurability project and the WWRF WG-6 on Reconfigurability. These current system initiatives highlight research challenges in increasing data rate and lowering power consumption. In order to develop low-cost, low-power systems, there is much research to be done in creating a low-power architecture with an efficient hardware abstraction layer (HAL). This HAL would allow for efficient hardware and software partitioning and for the use of appropriate modules to create an adaptable 4G system. Current wireless system architectural blocks do not have the design standardization and modularization to allow for efficient mix and match for reconfigurable (or cognitive) radio systems. Research in the use of design and architecture simulation tools, such as Xilinx System Generator, are leading to more efficient design reuse strategies.

Reconfigurability and system design and re-use will be important for 4G wireless systems. The mixture of DSP, ASIP, and ASIC structure will require efficient interconnection interfaces for high performance system design. The design of these ports, buffers, queues, and the resulting hardware abstraction layer will need to be optimized for overall system performance. Simulation environments will need to be enhanced to collect system statistics for design exploration of processor utilization and power efficiency.

4.4 4G/Beyond 3G Verification Tools

4.4.1 SDR Testbed for New Air Interface Evaluation and Interoperability Testing - General

Ongoing and future work on the exploration and development of new radio interfaces raises the issue of performance validation, understood not as the testing of new interface features by the company or group which develops them, but as third party verification. This is an issue of special importance for telecommunications operators, who additionally have to test interoperability between equipment and systems coming from different manufacturers. From this background, a need can be seen for the implementation of B3G verification tools.

Broadly speaking, new interfaces can not be in-depth tested with standard commercial equipment, because such equipment is usually adapted to the evaluation of those systems or elements for which there is a market, i.e., are backed by standards. Besides, in the fluid environment of new mobile communication systems research, it is becoming increasingly important to perform third party testing at intermediate development levels: quite often what has to be measured is not a complete element, like a transmitter or receiver, but a hardware, software or even middleware block, such as a resource demanding signal processing algorithm.

A possible answer to this evaluation request are test beds which simultaneously fulfil the following requirements. Their interfaces are open at all levels, to foster voluntary testing requests. They are as flexible as possible: modular in their construction and expansion capabilities, standard in equipment practice and with externally programmable signal processing sections. They can be configured and run remotely through Internet connections. Another interesting requirement is that of being simple to replicate, to enable low scale trials which require the interplay of several network elements. Such is the case, for example, of trials which involve testing MAC procedures or ad-hoc networking algorithms.

The next section describes a test bed or platform which conforms to these requirements, with the intention of having continuously upgrades for the evaluation of new radio interface features. The PRAGA platform developed at Telefonica is based on flexible, modular and upgradable SDR transceivers. It allows the establishment of radio communications between different transceivers, on which the radio interfaces can be partially or totally modified by means of SDR techniques. All internal and external interfaces, either HW or SW, are open to the WWRF community, to foster voluntary experimentation of new radio developments coming from different sources. Future developments will add new capabilities to the already developed radio interfaces. Service demonstration (video streaming) is already under testing on the platform, as part of a larger service prototyping activity on B3G radio interfaces.

4.4.2 The PRAGA platform

4.4.2.1 Design and Setup

The platform main elements are configurable transceivers, which are referred to as PRAGA (as in Figure 36). Figure 36 shows two PRAGAs in a typical platform set up configuration, intended to carry out radio interface response characterisation test.

Platform management and control procedures (namely, test runs start and stop, user interface, radio interface reprogramming and presentation of test results) is conducted through TCP/IP connections. In Figure 36 two different TCP/IP networks are presented for each transceiver, but both could be connected to the same network. In this case only one control terminal is needed, in which two different sessions of an Internet navigator application runs.

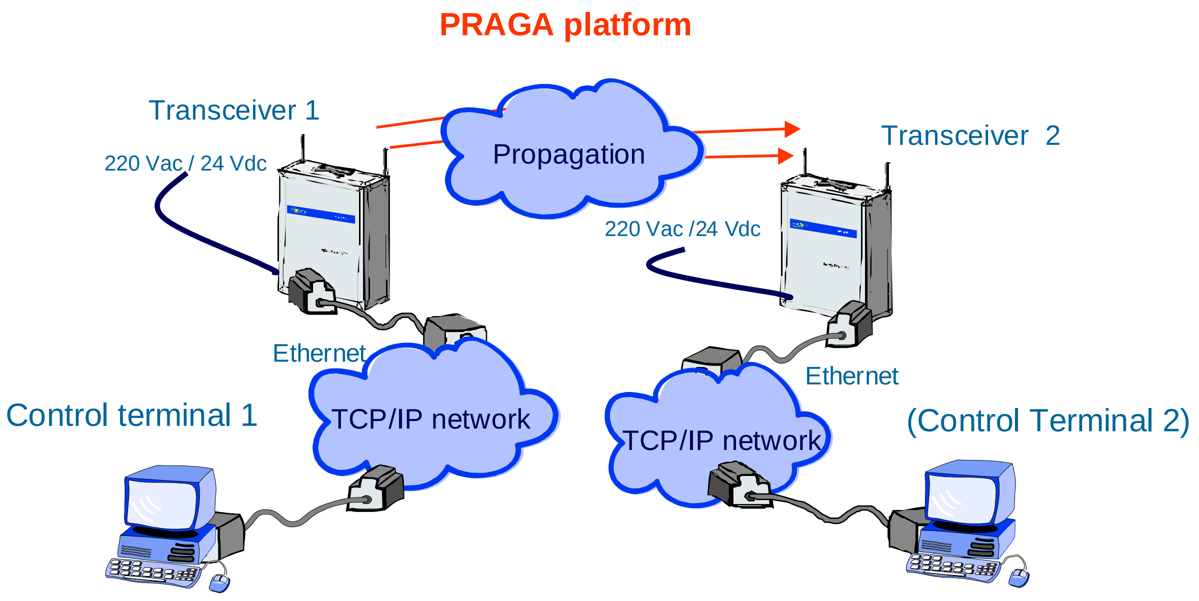


Figure 36: SDR platform configuration example

Moving into the transceiver details, Figure 37 shows the PRAGA block diagram, with its internal and external interfaces. In its present version, each transceiver has four different modules, housed in a 19" rack, with capability for expansion. The modules follow the SDR concept, i.e., radiofrequency (RF) module, termed CR, digital signal processing, or MPS module, and communications and control, or MCC. A fourth module, for power supply, MA, is also included.

The modules main characteristics are described in the following sections.

MCC: Communications Control Module

It is the transceiver interface with the outside. It is in charge of programming and executing control of all the devices in the transceiver. It design has been based on a PXA 255 microprocessor, in which a Linux operating system has been embedded.

The MCC basic functions are:

- To manage communications between the transceiver and outside networks, through an Ethernet 100 Base-T interface.
- To house a WEB page server application, which constitutes the demonstrator user interface for configuration and performance metering.
- To control, and channelize radio communications through-traffic from, up to four Signal Processing Modules (MPS).

MPS: Signal Processing Module

This module performs digital base band processing, and for some modulation schemes it is also the first IF block. It interfaces the data bit streams coming from, or being sent to, the MCC and the analog signals exchanged with the Radio Frequency Module (CR).

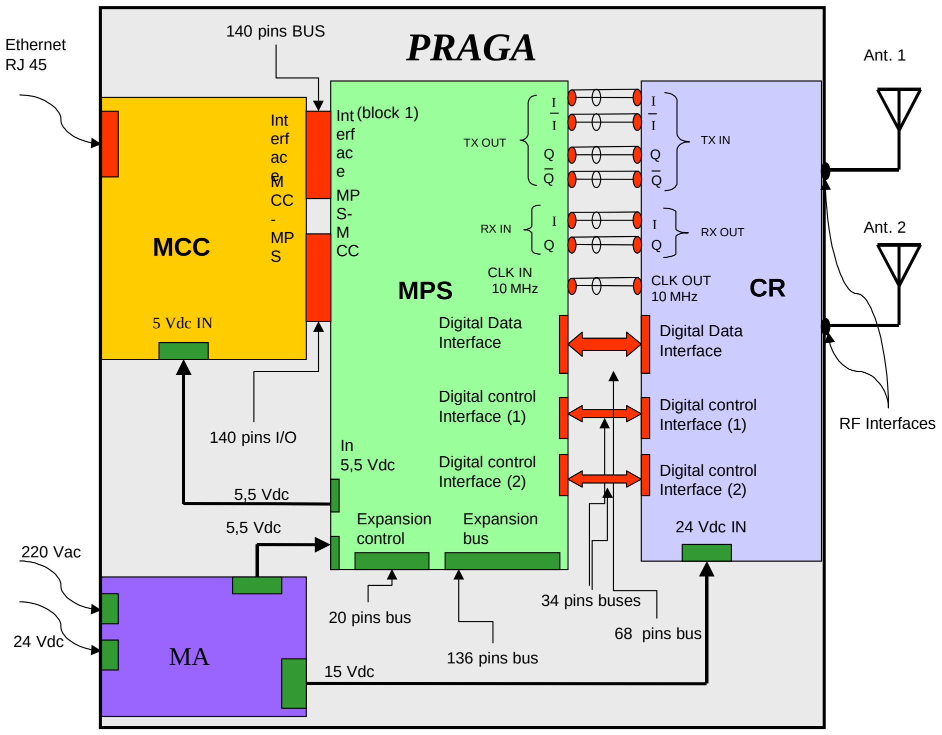


Figure 37: PRAGA transceiver block diagram

Its main function is to implement in base band all required signal processing algorithms associated with the radio interface in re-configurable devices, like FPGAs and DSPs (avoiding the use of ASICs). Thus the MPS can be reprogrammed for almost any modulation technique, within the limitations of algorithm speed for real-time transmissions and ADC/DAC bandwidth.

It interfaces with the MCC through two 140 pin connectors with control, address and data buses. The interfaces with the CR module are:

- Analog input: through it the MPS receives two signals in single ended mode. These signals can either be the In phase (I) and Quadrature (Q) base band components of a radio communications channel or two different IQ modulated signals in Intermediate Frequency (IF) from two different radio communications channels.

- Analog output: used to send two analog signals in differential mode from the MPS, which are the output of 300 Msps DACs. The signals can either be the Inphase (I) and Quadrature (Q) baseband components of a radio communications channel or two different IQ modulated signals in Intermediate Frequency (IF) for two different radio communications channels
- Input reference clock (10 MHz) generated at the RF module, which is used as the master clock for the generation of devices operating clocks.
- Digital Control Signal: used to control RF parameters in the Radio Frequency Modules (tx power, channel RF band, etc.).
- Digital Signal Interface: It is the digital equivalent of the analog input and output. It is an LVDS interface that provides very high speed data transmission rate (up to 600 Mbps) between the MPS and RF module. This interface is used when the Radio Frequency Module baseband or IF interface is already digital.

The MPS module also incorporates an expansion bus to which up to three additional MPS modules can be connected to increment its signal processing capabilities.

RF Module (CR)

This is the analog module. Its mission is to convert base band signals to selected RF band signals and vice versa. So far, two different RF modules have been developed, one designed for OFDM transmission and reception and the other for UWB transmission.

4.4.2.2 Platform Development Environment

One of the key points to achieve short development time, in the implementation of new algorithms, is the availability of a well established simulation engine which will corroborate the algorithm suitability in different simulated radio channels.

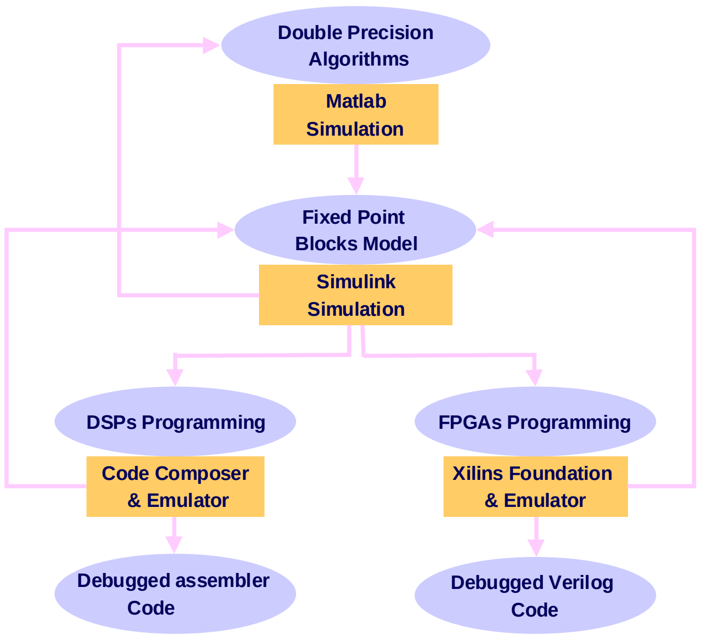
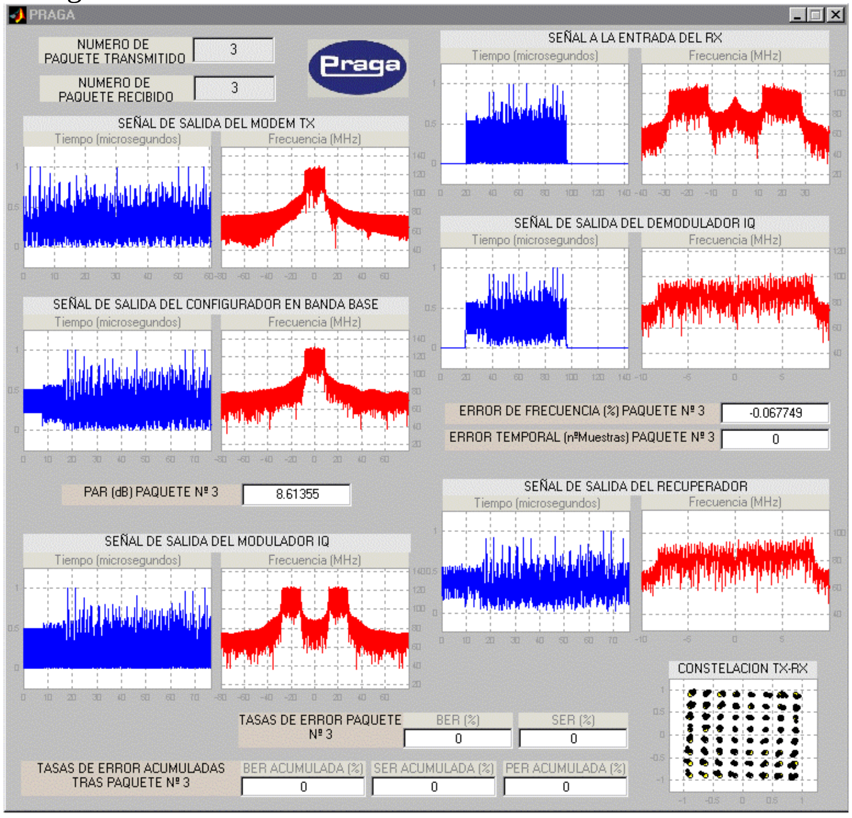


Figure 38: PRAGA Development workflow

These simulations take much less time to develop than actual algorithm implementation in the platform, first results being usually available in a few days time. The same simulation engine allows comparison between double precision and fixed point versions. It is also the source of bit trial vectors to check correct algorithm implementation on DSPs and FPGAs.

In Figure 38 the platform development workflow is shown, in which two loops can be observed. One is the already mentioned comparison between double precision and fixed point simulations, and the other corresponds to actual device constraints, mainly due to real time implementation. The simulation engine has been developed with Matlab and Simulink tools, to which a user friendly graphical interface has been added, as shown in Figure 39.


PRAGA
Praga
SEÑAL DE SALIDA DEL MODEM TX
SEÑAL DE SALIDA DEL CONFIGURADOR EN BANDA BASE
SEÑAL DE SALIDA DEL MODULADOR IQ
SEÑAL A LA ENTRADA DEL RX
SEÑAL DE SALIDA DEL DEMODULADOR IQ
SEÑAL DE SALIDA DEL RECUPERADOR
Tiempo (microsegundos)
Frecuencia (MHz)
CONSTELACION TX-RX

NUMERO DE PAQUETE TRANSMITIDO: 3
NUMERO DE PAQUETE RECIBIDO: 3
ERROR DE FRECUENCIA (%) PAQUETE Nº 3: -0.067749
ERROR TEMPORAL (nºMuestras) PAQUETE Nº 3: 0
PAR (dB) PAQUETE Nº 3: 8.61355

	BER (%)	SER (%)
TASAS DE ERROR PAQUETE Nº 3	0	0

	BER ACUMULADA (%)	SER ACUMULADA (%)	PER ACUMULADA (%)
TASAS DE ERROR ACUMULADAS TRAS PAQUETE Nº 3	0	0	0

Figure 39: PRAGA simulator graphic interface

4.4.2.3 User Interface

The platform has been designed with a user interface based on a WEB page server hosted in the MCC, easily accessible to the user with any standard Internet navigator. With this open interface, the addition of new radio interfaces only needs modifying the first page, named "MODULATION SCHEDULE SELECTION" in the Web map of Figure 40, and adding new pages corresponding to the new radio interface.

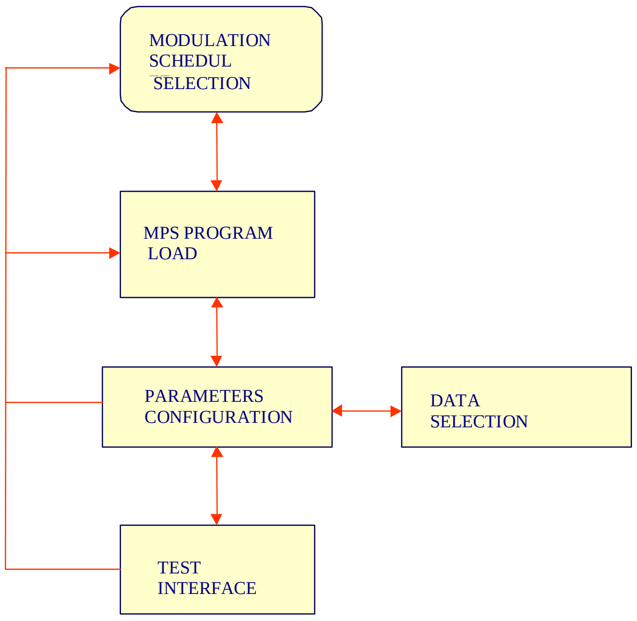


Figure 40: PRAGA WEB site map

The main functions of each WEB page can be described as follows:

1) "MODULATION SCHEDULE SELECTION": In this page the radio interface is selected, among the set of available ones.
2) "MPS PROGRAM LOAD": In this page the signal processing software load is performed on the MPS DSPs and FPGAs, as a function of selected radio interface, showing the loaded version identification.
3) "PARAMETER CONFIGURATION": Graphical user interface for inputting the configuration parameters of the chosen base band and RF signal.
4) "DATA SELECTION": The transmitted data source can be a file allocated at the user control terminal, at the transceiver MCC, or be a pseudorandom file (with selectable seed and length), the selection being made in this page
5) "TEST INTERFACE": Online real results are displayed, using Java Runtime Environment. In this page the user can select the graphics to be displayed (in the TX and RX control units) during the test, as well as the signal to be stored in ASCII format in files at the control units. This page also manages the test runs with START, STOP and PAUSE buttons.

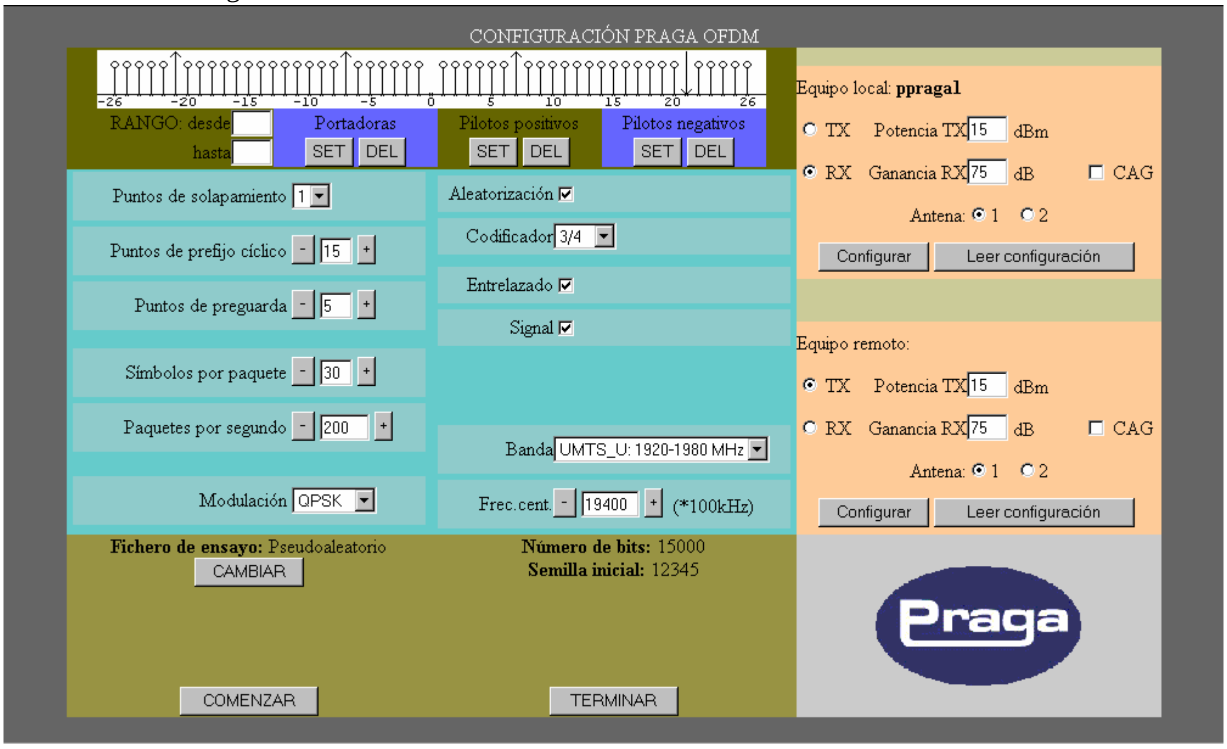
4.4.2.4 Developed Radio Interfaces

Two different radio frequency interfaces have already been developed, each with its own RF module, one designed for OFDM transmission and reception and another for UWB transmission. Currently, a video-streaming demonstrator is being implemented on the OFDM modulation scheme.

In Figure 41, the "CONFIGURATION PARAMETER" page corresponding to the OFDM link, in which the user can select the base band and RF parameters, is shown. The configurable parameters are:

- Every sub-carrier from -26 to 26 (for a standard 64 point IFFT in 3.2 μs) can be programmed as data sub-carrier, positive pilot, negative pilot, or not transmitted.
- Cyclic prefix length.
- Configurable windowing.
- Number of OFDM symbols transmitted in each burst.

- Coding rate.
- Preguard interval considered for reception FFT implementation.
- Data subcarrier modulation (BPSK QPSK 16 QAM and 64 QAM).
- Optional scrambling and puncturing codes
- Transmitted output power.
- RF channel usage.



CONFIGURACIÓN PRAGA OFDM

RANGO: desde
hasta

Portadoras SET DEL
Pilotos positivos SET DEL
Pilotos negativos SET DEL

Puntos de solapamiento 1
Puntos de prefijo cíclico 15
Puntos de preguarda 5
Símbolos por paquete 30
Paquetes por segundo 200
Modulación QPSK

Aleatorización ☑
Codificador 3/4
Entrelazado ☑
Signal ☑
Banda UMTS_U: 1920-1980 MHz
Frec.cent. 19400 (*100kHz)

Equipo local: **ppragal**
TX Potencia TX 15 dBm
[✓ RX] Ganancia RX 75 dB ☐ CAG
Antena: [✓ 1] 2
Configurar | Leer configuración

Equipo remoto:
[✓ TX] Potencia TX 15 dBm
RX Ganancia RX 75 dB ☐ CAG
Antena: [✓ 1] 2
Configurar | Leer configuración

Fichero de ensayo: Pseudoaleatorio
CAMBIAR
Número de bits: 15000
Semilla inicial: 12345

COMENZAR | TERMINAR

Figure 41: Example of user interface parameter configuration WEB page

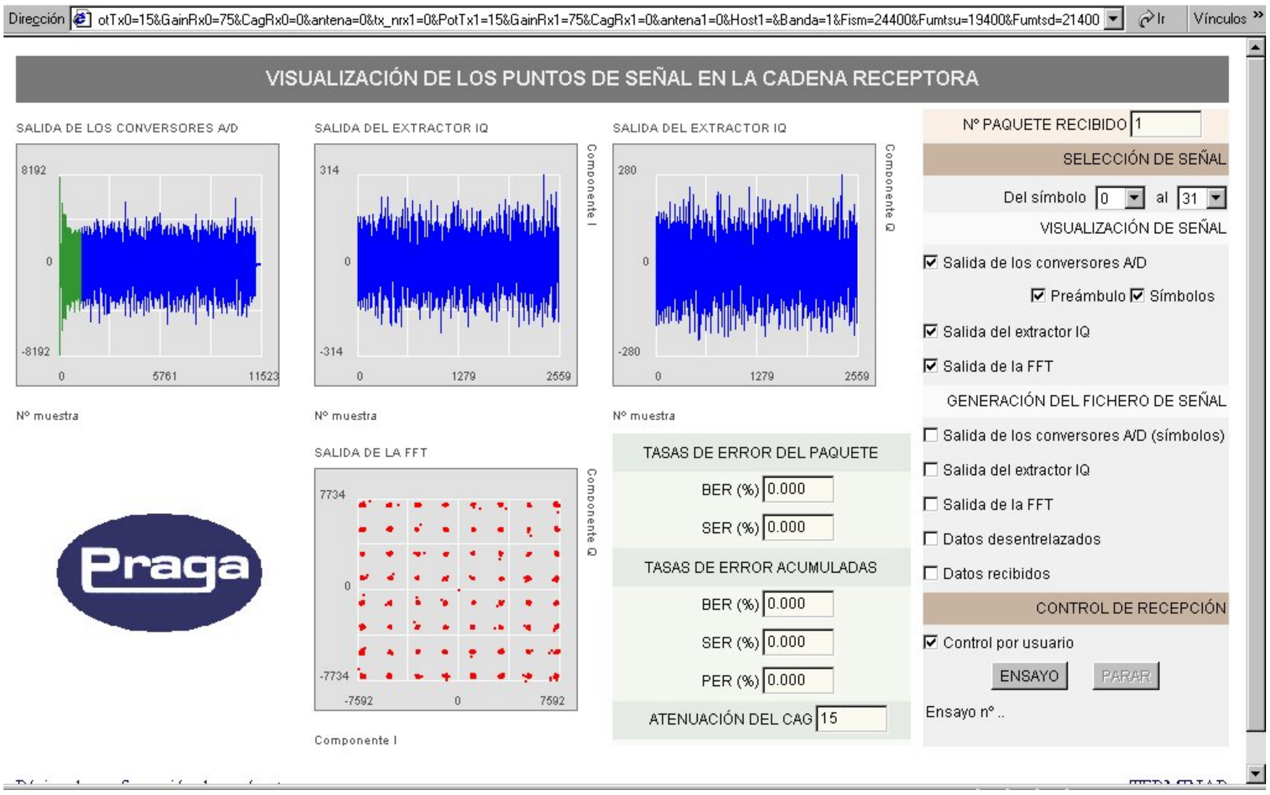
The UWB page offers the possibility of selecting the pulse width and the transmitted power. Once the chosen parameters have been programmed on both transceivers, and the data source has been selected, the "TEST INTERFACE" page will be served by the MCC.

On Figure 42 the receiver "TEST PAGE" interface is shown, in which several signal graphics can be selected to be displayed, and/or stored, the figure corresponding to a 64-QAM OFDM transmitted signal.

The PRAGA platform outlined is based on flexible, modular and upgradable SDR transceivers. It allows the establishment of radio communications between different transceivers, on which the radio interfaces can be partially or totally modified by means of SDR techniques. All internal and external interfaces, either HW or SW, are open to the WWRF community, to foster voluntary experimentation of new radio developments coming from different sources.

Future developments will add new capabilities to the already developed radio interfaces. Service demonstration (video streaming) is already under testing on the platform, as part of a larger service prototyping activity on B3G radio interfaces.

Research is needed to define what is an optimum hardware architecture for a given wireless standard set. A crucial issue of the resulting architecture is the resulting implementation complexity, i.e. power consumption, chip area of single modules and the overall resulting architecture. This may also include research on efficient HW-Structures of re-configurable logic modules. Design flow for implementation of SW-modules for constituent standards on the hardware architecture has also to be considered. In the recent and current research of re-configurability in wireless communications, such as EC funded projects TRUST and CAST, the problems of base band re-configuration were identified and addressed. With reference to TRUST, the top down design approach was adopted and it was identified that for the base band re-configuration purposes an additional entity is needed. This entity is termed Management Module.


Dirección otTx0=15&GainRx0=75&CagRx0=0&antena=0&tx_nrx1=0&PotTx1=15&GainRx1=75&CagRx1=0&antena1=0&Host1=&Banda=1&Fism=24400&Fumtsu=19400&Fumtsd=21400

VISUALIZACIÓN DE LOS PUNTOS DE SEÑAL EN LA CADENA RECEPTORA

Nº PAQUETE RECIBIDO: 1

SELECCIÓN DE SEÑAL

Del símbolo: 0 al: 31

VISUALIZACIÓN DE SEÑAL

☑ Salida de los conversores A/D
☑ Preámbulo ☑ Símbolos
☑ Salida del extractor IQ
☑ Salida de la FFT

GENERACIÓN DEL FICHERO DE SEÑAL

☐ Salida de los conversores A/D (símbolos)
☐ Salida del extractor IQ
☐ Salida de la FFT
☐ Datos desentrelazados
☐ Datos recibidos

TASAS DE ERROR DEL PAQUETE

BER (%): 0.000
SER (%): 0.000

TASAS DE ERROR ACUMULADAS

BER (%): 0.000
SER (%): 0.000
PER (%): 0.000

ATENUACIÓN DEL CAG: 15

CONTROL DE RECEPCIÓN

☑ Control por usuario

Ensayo nº ..

Figure 42: Example of RX user interface page

In CAST designers tried to address the problem of re-configurability using the top down and bottom up approaches. Therefore, two additional separate control components were designed for managing the process of the base band re-configuration. These are the Re-configurable Resource Controller and Physical Layer Controller.

These two entities can perform the following:

1. Cover the functionality of Management Module as in TRUST,
2. Allocate physical layer resources,
3. Optimise physical layer resources in terms of:
    a. Power consumption,
    b. Speed of processing, and
    c. Memory usage,
4. Implement re-configuration, and
5. Accommodate different vendors.

In these projects the designed base band software can be characterised as “passive”. This is because of a need for additional management entities to download, manipulate and control software modules in the process of re-configuration. In order to eliminate the need for these additional management units and reduce the requirements for downloading and re-configuration control signalling, further research is needed in producing new methods for designing the base band processing software. These methods should enable the base band software to be “active”. This means that base band software should have the ability to change functionality, behaviour and performance without the need for additional management units. These actions should be completed by base band software itself. Hence, there will be reduced requirements for downloading and re-configuration control signalling. This approach should allow the design to be based upon known methods with clearly defined data flow, control mechanisms and interfaces. Procedures for re-configuration should also be simple and well defined.

In addition, the research should be conducted in identifying a specification for enhancing existing Real Time Operating Systems into Software Defined Radio based Real Time Operating Systems (SDRbRTOS). The SDRbRTOS will be needed to support “active” base band processing software and perform optimisation of re-configurable physical layer hardware.

The overall research process therefore includes:

- Thorough analysis of existing wireless standards,
- Tracking of trends in upcoming wireless standards,
- Analysis and discussion of SDR-BB architecture proposals,
- Research and analysis of state-of the art and future component technology, including General Purpose-DSP, memory, re-configurable logic, semiconductor technology,
- Comparison of cost with conventional approaches.

5 Conclusions

In today's wireless world a large number of Radio Access Technology (RAT) standards is available. The recent trend of "wireless beyond the third generation" (B3G) assumes that cellular, Broadband Radio Access Networks (BRAN) / Wireless Local Area Networks (WLAN) and Digital Video Broadcasting (DVB) systems can be co-operating components of a Composite Radio (CR) infrastructure. Through such a CR system, that provides the possibility of co-operation among the various available RATs, users can be directed to the most appropriate one, according to the service area regions, time zones, profile and network performance criteria. In this context, the deployment of CR systems requires technologies that allow terminals and network elements to dynamically select and adapt to the most appropriate RAT (in a transparent manner). The Reconfigurability concept (which is an evolution of "software defined radio") provides such technologies posing essential issues with respect to element management.

In this direction, this white paper presented a concept for a Management and Control System that enables elements to operate in an end-to-end reconfigurability context. The main idea of this concept is a clear separation of the management and the control functions. The paper also introduced a reference model for a multi-mode protocol stack of a flexible, dynamic reconfigurable air-interface for future wireless networks. In the sequel of the paper some issues related to SDR were addressed. Hardware abstraction, which is a research topic widely discussed in the reconfigurability community (e.g. specific RFI in the SDR Forum), was discussed. The paper presented a possible hardware abstraction approach in an en-to-end reconfigurable device. Some design and integration challenges for reconfigurable systems are also highlighted. The paper concluded with an overview of a verification tool for 4G/ Beyond 3G Systems, the PRAGA platform.

6 References

[1] E2R, WP2, “D2.1: State-of-the-Art, Scenario Analysis and Requirements Definition for Equipment Reconfiguration Management”

[2] Software Defined Radio Forum, “Specification for PIM and PSM for SW Radio Components”, Draft Revised Submission, 2004-01-01.

[3] Clark, D. and D. Tennenhouse, “Architectural Considerations for a New Generation of Protocols”, Proc ACM SIGCOMM, Sept 1990.

[4] Clemens Szyperski, “Component Software - Beyond Object-Oriented Programming”, Addison-Wesley/ACM Press, 1998

[5] Gultchev S, Moessner K, Tafazolli R, “Management and Control of Reconfiguration Procedures in Software Radio Terminals”, WSR2002, Karlsruhe, Germany, 20-21 March 2002

[6] Gultchev S, Moessner K, Tafazolli R, “Reconfiguration Mechanisms and Processes in RMA controlled Soft-Radios Signalling”, SDR Technical Conference, Orlando, Florida, USA, 17-19 November 2003

[7] 3GPP, “TSG Terminals; UE capability requirements”, v3.5.0, 2002-09.

[8] Alonistioti N., Glentis A., Foukalas F., Kaloxylos A., “Reconfiguration Management Plane for the Support of Policy-based Network Reconfiguration”, Personal Indoor and Mobile Radio Communications (PIMRC) Symposium, Barcelona, Spain, September 5-8, 2004.

[9] E2R, WP3, D3.1, “State-of-the-Art and Outlooks for Reconfiguration and Download Procedures, Network Support Functions, Protocol Architectures, Flexible Service Provision, and Enabling Platforms”.

[10] Book of Visions for the Wireless World, issued by the Wireless Strategic Initiative (www.ist-wsi.org), December 2000.

[11] 2nd Book of Visions for the Wireless World, issued by the Wireless World Research Forum (www.wireless-world-research.org), December 2001.

[12] http://mobivas.cnl.di.uoa.gr/

[13] Alonistioti, N. Houssos, S. Panagiotakis, M. Koutsopoulou, V. Gazis, "Intelligent architectures enabling flexible service provision and adaptability", Wireless Design Conference (WDC 2002), London, UK, 15-17 May 2002.

[14] S. Panagiotakis, N. Houssos, A. Alonistioti, "Integrated generic architecture for flexible service provision to mobile users", IEEE 12th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2001), San Diego, California, USA, 30/9 - 3/10 2001, pp. 40-44.

[15] Alonistioti, N. Houssos, S. Panagiotakis, "A framework for reconfigurable provisioning of services in mobile networks", Sixth International Symposium on Communication Theory & Applications (ISCTA 2001), Ambleside, Lake District, UK, pp. 21-26.

[16] M. Koutsopoulou, N. Alonistioti, E. Gazis, A. Kaloxylos, "Adaptive Charging Accounting and Billing system for the support of advanced business models for VAS provision in 3G systems" Invited paper at the PIMRC 2001, September - October 2001, San Diego, USA.

[17] Nikos Houssos, Vangelis Gazis, Athanassia Alonistioti, "A flexible management architecture for the support of advanced business models in 3G mobile service provision", 1st International Conference on Mobile Business (M-Business 2002), Athens, Greece, 8-9 July 2002.

[18] N. Houssos, V. Gazis, S. Panagiotakis, S. Gessler, A. Schuelke, S. Quesnel, "Value Added Service Management in 3G Networks", 8th IEEE/IFIP Network Operations and Management Symposium (NOMS 2002), Florence, Italy, 15-19 April 2002, pp. 529-545.

[19] Alonistioti, S. Panagiotakis, N. Houssos, A. Kaloxylos, "Issues for the provision of Location-dependent services over 3G networks", 3rd generation infrastructure and services conference (3GIS) , Athens, Greece, July 2001.

[20] S. Panagiotakis, N. Houssos, N. Alonistioti, "Generic architecture and functionality to support downloadable service provision to mobile users", 3rd generation infrastructure and services conference (3GIS), Athens, Greece, July 2001.

[21] Nikos Houssos, Spyros Pantazis, Athanassia Alonistioti, "Towards adaptability in 3G service provision", IST Mobile Communication Summit 2002, Thessaloniki, Greece, 16-19 June 2002

[22] N. Alonistioti, E. Gazis, M. Koutsopoulou, Spyridon Panagiotakis, "An Application platform for downloadable VASs provision to mobile users", proceedings of the IST Mobile Communications Summit 2000, Galway, Ireland, September-October 2000.
[23] 3GPP TS 29.198: "Open Service Access (OSA); Application Programming Interface (API); Part 1-12 (version 4.3.0)".
[24] Open Services Gateway Initiative (OSGi) Service Platform, Release 2, October 2001 , available from URL http://www.osgi.org/resources/docs/spr2book.pdf
[25] Parlay Group, "Parlay API Spec. 2.1", July 2000, available from URL http://www.parlay.org/specs/index.asp
[26] J. Keijzer, D. Tait, R.Goedman, "JAIN: A new approach to services in communication networks", IEEE Communications Magazine, January 2000.
[27] LIF TS 101 "Mobile Location Protocol" v2.0.0.
[28] Report #9, UMTS Forum, http://www.umts-forum.org/.
[29] Spyridon Panagiotakis, Maria Koutsopoulou, Athanasia Alonistioti, Alexandros Kaloxylos, "Generic Framework for the Provision of Efficient Location-based Charging over Future Mobile Communication Networks", PIMRC, Lisbon, Portugal, September 2002.
[30] Spyridon Panagiotakis, Maria Koutsopoulou, Athanasia Alonistioti, "Advanced Location Information Management Scheme for Supporting Flexible Service Provisioning in Reconfigurable Mobile Networks", IST Mobile Communication Summit, Thessaloniki, Greece, June 2002.
[31] M. Beach, D. Bourse, R. Navarro, M. Dillinger, T. Farnham, T. Wiebke, "Reconfigurable Terminals Beyond 3G and Supporting Network System Aspects", Presentation in WG 3, Stockholm, Sweden, 17-18 September 2001.
[32] www.ist-trust.org, 2002.
[33] www.sdrforum.org, 2002.
[34] www.mobilevce.com, 2002.
[35] J. Mittola, "The Software radio Architecture," IEEE Communications Magazine, May 1995.
[36] Bourse D, "WWRF WG3 SDR Architectures", a document circulated in the SDR community within the Wireless World Research Forum, May 2002.
[37] http://www.sdrforum.org/tech_comm/mobile_wg.html
[38] Metha M., Wesseling M, "Adaptive Baseband Subsystem for TRUST", Proceedings of PIMRC 2000, pp. 29-33, London, UK, 18th-21st September 2000.
[39] Gultchev S, et al, "Management and Control of Reconfiguration Procedures in Software Radio Terminals", 2nd Workshop on Software Radio, Karlsruhe, Germany, 20/21st March 2002.
[40] N. Nakajima, R. Kohno, S. Kubota, Research and Developments of Software-Defined Radio Technologies in Japan, IEEE Communications Magazine , vol. 39, no. 8 , Aug. 2001, pp. 146 -155.
[41] J.D. Laster, Robust GMSK Demodulation Using Demodulator Diversity and BER Estimation, Ph.D. Thesis, Virginia Tech, 1997.
[42] SCOUT webpage: http://www.ist-scout.org
[43] H. Kaaranen et al, "UMTS Networks. Architectures, Mobility and Management", John Wiley & Sons, Ltd, 2001.
[44] X. Qiu et al, "Network Assisted Resource Management for Wireless Data Networks", *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 7, pp 1222-1234, July 2001.
[45] H. Yomo, S. Hara, "Impact of Access Schemes Selectability on Traffic Performance in Wireless Multimedia Communication Systems", *IEEE Transactions on Vehicular Technology*, vol. 50, n. 5, pp 1298-1307, Sept 2001.
[46] http://www.tinac.com/about/about.htm
[47] E. Buracchini, "The Software Radio Concept", IEEE Communications Magazine, September 2000.
[48] Shiao-Li Tsao, Chia-Ching Lin, Hung-Lin Chou, Chin-Lien Chiu, Min-Chiao Wang, "Design and Implementation of Software Framework for Software Defined Radio System", to appear in Proceedings of IEEE VTC Fall 2002.www.sdrforum.org, 2002.
[49] 3GPP TS 23.057: "Mobile Execution Environment (MExE) Functional Description", Version 5.0.0, March 2002.
[50] 3GPP TS 33.102: "3G Security – Security Architecture", Version 5.0.0, June 2002.

[51] Rainer Falk, Nicola Greco: “Securely Reconfigurable Terminals”, IST Mobile Communications Summit 2001, Barcelona, Spain, September 2001.
[52] Lachlan B. Michael, Miodrag J. Mihaljevic, Shinichiro Haruyama, Ryuji Kohno: A Framework for Secure Download for Software Defined Radio, IEEE Communications Magazine, pp. 88–96, July 2002.
[53] K. Moessner, S. Gultchev, R. Tafazolli, “Managing Reconfiguration in Software Defined Communication Systems”, ISCTA”01, Ambleside, Lake District, United Kingdom, 15-19 July 2001.
[54] http://www.iis.ee.ethz.ch/nwp/lemon/lemon.html
[55] Springer A., Maurer L., Weigel R., *RF System Concepts for Highly Integrated RFICs for W-CDMA Mobile Radio Terminals*, IEEE Transactions on Microwave Theory and Techniques, Vol. 50, No. 1, January 2002, pp254-267.
[56] Sagers R. C., *Intercept Point and Undesired Responses*, IEEE Transactions on Vehicular Technology, Vol. 32, February 1983, pp121-133.
[57] Crols J., Steyaert M. S. J., “Low IF Topologies for High Performance Analog Front Ends of Fully Integrated Receivers*”, IEEE Transactions on Circuits and Systems – II: Analog and Digital Signal Processing* Vol. 45, No. 3, March 1999, pp269-282.
[58] Banu M., Wang H., Seidel M., Tarsia M., Fischer W., Glas J., Dec A., Boccuzzi V., *A* “BiCMOS Double-Low-IF receiver for GSM”, *IEEE 1997 Custom Integrated Circuit Conference*, pp521-524.
[59] Abe M., Sasho N., Brankovic V., Krupezevic D., *Direct Conversion Receiver MMIC based on Six-Port Technology*, European Conference on Wireless Technology, 2000.
[60] Hyyryläinen J., Bogod L., Kangasmaa S., Scheck H.O., Ylämurto T., *Six-port Direct Conversion Receiver*, 27th European Microwave 97 Conference Proceedings, Vol. 1, Sept. 97, pp341-346.
[61] I. Mann, M. A. Beach, P. A. Warr and J. P McGeehan, “Increasing the talk-time of mobile radios with efficient linear transmitter architectures,” *IEE Electronics & Communication Engineering Journal*, Vol. 13, No 2, pp., 65-76, April 2001.
[62] P. A. Warr, M. A. Beach and J. P. McGeehan, “Gain-element transfer response control for octave-band feedforward amplifiers,” *IEE Electronics Letters*, Vol. 37, No. 3, pp., 146-147, 1st February 2001.
[63] T. Nesimoglu and M. A. Beach, “Linearised mixer using frequency retranslation”, UK Patent Application No. 0117801.1, 20th July 2001.
[64] T. Nesimoglu, M. A. Beach, P. A. Warr and J. R. MacLeod, “Linearised Mixer using Frequency Retranslation”, IEE Electronics Letters, Vol. 37, No. 25, pp., 1493-1494, December 2001.
[65] P. Katzin, V.Aparin, “Active, Self Adjusting, L-S Band, MMIC Filter”, *IEEE GaAs Symposium*, pp 41-43, 1994.
[66] P Katzin, Personal Correspondence, January 2001.
[67] G. Matthaei, L. Young, E.M.T. Jones, “*Microwave Filters, Impedance-Matching Networks, and Coupling Structures”*, Artech House 1980.
[68] R Levy S.B. Cohn “A History of Microwave Filter Research, Design, and Development”, *IEEE Transactions on Microwave Theory and Technique*, Vol. MTT-32, No.9, pp1055- 1067, September 1984.
[69] I.C. Hunter and J.D. Rhodes “Electronically Tuneable Microwave Bandpass Filters”, *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-30, No9, pp1354- 1360, September 1982.
[70] R. Y. Loo, G. Tangonan, D. Sivenpiper, J. Schaffner, T.Y. Hsu, H.P. Hsu, “Reconfigurable Antenna Elements using RF MEMS Switches”, *Proceedings of ISAP2000*, pp 887- 890, Fukuoka, Japan, 2000.
[71] M Sagawa, K Takahashi, M Makimoto. “Miniaturised Hairpin Resonator Filters And Their Application To Receiver Front End MIC”s”, *IEEE Transactions on Microwave Theory and Techniques*, Vol. 37, No. 1`2, pp1991-1997, December 1989.
[72] M Makimoto, S Yamishita, “Bandpass Filters Using Parallel Coupled Stripline Stepped Impedance Resonators”, *IEEE Transactions On Microwave Theory And Techniques*, Vol. MTT-28, No. 12, pp1413-1417, December 1980.
[73] O Rostbakken, G.S. Hilton, C.J. Railton, C.J., “Adaptive Feedback Frequency Tuning for Microstrip Patch Antennas”, 9th International Conference on Antennas and Propagation, IEE. Part 1, 1995, pp.166-70,vol.1. London, UK.
[74] O Rostbakken, G.S. Hilton, C.J. Railton, “An adaptive microstrip patch antenna for use in portable transceivers” IEEE 46th Vehicular Technology Conference. Mobile Technology for the Human Race. IEEE. Part vol.1, 1996, pp.339-43 vol.1. New York, NY, USA).

[75] P.R. Urwin-Wright, G.S. Hilton, I.J. Craddock & P.N. Fletcher, "A Tuneable Electrically-Small Antenna Operating in the "DC" mode", Abstract submitted for IEE Radio Spectrum Seminar "Getting the Most out of the Radio Spectrum," London, October 2002.
[76] R. C. Johnson (Ed), "Antenna Engineering Handbook", 3rd Edition, New York, McGraw-Hill, 1993.
[77] Ali, Hayes, Hwang and Sandler, "A Triple – band Integrated Antenna for Mobile Hand – Held Terminals, APS 2002, pp 32–35.
[78] Fusco, V., "Integrated Antennas for Wireless Applications", Applied Microwave and Wireless, June 2002, pp 22-31.
[79] S. A. Long, M.W. Mcallister and L.C.Shen, "The resonant cylindrical dielectric cavity antenna", IEEE Trans on Antennas and Propagation, vol AP-31, pp. 406-412, May 1983
[80] http://literature.agilent.com/litweb/pdf/5968-1613E.pdf
[81] R. J. Richards, H. J. De Los Santos, "MEMS for RF/Microwave Applications The Next Wave", *Microwave Journal*, Vol. 44, No. 3, pp 20 – 41, March 2001.
[82] R. J. Richards, H. J. De Los Santos, "MEMS for RF/Microwave Applications The Next Wave –Part II", *Microwave Journal*, Vol. 44, No. 7, pp 142 – 152, July 2001.
[83] P. M. Campbell et al., "*A Very Wide Bandwidth Digital VCO Using Quadrature Frequency Multiplication and Division Implemented in ALGaAs/GaAs HBT"s"*, IEEE Trans. Very Large Scale Integration System, pp. 52-55, March 1998.
[84] P. Warr et al., "*Quadrature Signal Splitting Technique Offering Octave-Band Performance"*, accepted to IST2000.
[85] Abidi, "*Direct-Conversion Radio Transceivers for Digital Communications"*, IEEE Journal Solid-State Circuits, pp. 1399-1410, Dec. 1995.
[86] T. Burger, Q. Q. Huang, "A 13.5mW 185-Msample/s Delta-Sigma Modulator for UMTS/GSM Dual-Standard IF Reception", IEEE Journal of Solid-State Circuits, vol. 36, No.12, pp.1868-1878, Dec. 2001.
[87] Teetzel A., "*Circuit Design: Design of a wideband I Q modulator"*, 1997 RF Design Seminar, Hewlett Packard Corporation.
[88] http://products.analog.com/products/info.asp?product=AD8347
[89] Razavi B., "*Design Considerations for Direct-Conversion Receivers"*, IEEE transactions on Circuits and Systems II: Analog and Digital Signal Processing, Vol. 44, No. 6, June 1997, pp428-435.
[90] Parssinen A., Jussila J., Ryynanen J., Sumanen L Halonen A.I., "*A 2-GHz Wide-Band Direct Conversion Receiver for WCDMA Applications"*, IEEE journal of Solid –State Circuits Vol. 34, No. 12, December 1999, pp1893-1903.
[91] M Metha et al., "Reconfigurable Terminals: An Overview of Architectural Solutions", IEEE Communications Magazine, August 2001.
[92] P. R. Chevillat et al., "Hardware Architecture of a Software-Defined Radio fo Mobile Commmuniction Systems beyond 3G", WWRF, Helsinki, 2001.
[93] P. Jung et al., "Future Microelectronoic Hardware Concepts for Wireless Communication Beyond 3G", WWRF, Helsinki, 2001.
[94] P . Ting et al., An adaptive Hardware Platform for SDR", WWRF, Stockholm, 2001.
[95] M. Beach et al., "Re-Configurable Terminals Beyond 3G: Re-Configurable Baseband", WWRF, Paris, 2001.
[96] S. Hsieh et al., "A SDR Transceiver Design with Re-Configurable IF and Basband Platform", WWRF, Paris 2001.
[97] K. Moessner et al., "Software Radio Integration and Reconfiguration Management", WWRF, Paris, 2001.
[98] A. Marath et al., "System Level Issues in the Implementation of a Mulimode Terminal (WCDMA+GSM+802.11a+802.11b WLAN)", WWRF, Tempe, 2002.
[99] T. Rautio et al., "SDR Architecture Design for Future Information Sense Terminals", WWRF, Tempe, 2002.
[100] L. Kristiansen: TINA-C Service Architecture. Version 5.0. TINA-C, 1997.
[101] OMG Telecommunications Specifications: Telecom Service Access & Subscription. http://www.omg.org/cgi-bin/doc?dtc/2000-10-03
[102] The Parlay Group: http://www.parlay.org, 2002.
[103] M. Siebert, "Design of a Generic Protocol Stack for an Adaptive Terminal", Proc. of the 1st Karlsruhe Workshop on Software Radios, pp. 31-34, Karlsruhe, Germany, March 2000.

[104] M. Siebert, B. Walke, “Design of Generic and Adaptive Protocol Software (DGAPS)”, Proc. of the Third Generation Wireless and Beyond (3Gwireless “01), San Francisco, US, June 2001.
[105] M. Siebert, M. Steppler, “Software Engineering in the face of 3/4G Mobile Communication Systems”, Proc.of the 10th Aachen Symposium on Signal Theory, ISBN 3-8007-2610-6, pp. 89-94, Aachen, Germany, September 2001.
[106] K. Moessner “Reconfigurable Mobile Communication Networks”, Doctoral Thesis, University of Surrey, UK, 2001.
[107] J. Mitola, “Software Radio Architecture Object-Oriented Approaches to Wireless Systems Engineering”, Wiley-Interscience, ISBN 0-471-38492-5, 2000.
[108] 3GPP TS 25.322, RLC protocol specification, ftp://ftp.3gpp.org/specs/latest/Rel-4/25_series/25322-430.zip.
[109] Joachim Sachs, Andreas Schieder, “Generic Link Layer”, Wireless World Research Forum, Tempe, USA, March 7-8 2002
[110] "Request For Proposal PIM and PSM for SWRADIO Components", Object Management Group Document swradio/02-06-02, http://cgi.omg.org/cgi-bin/doc?swradio/02-06-02 .
[111] Sun Microsystems: Mobile Information Device Profile (JSR-37), JCP Specification, Java2 Platform, Micro Edition, 1.0, September 2000.
[112] Sun Microsystems: Mobile Information Device Profile, v2.0 (JSR-118), Public Draft Specification, Java2 Platform, Micro Edition, 2002.
[113] R. Hirschfeld, W. Kellerer, C. Prehofer, H. Berndt: *An Integrated System Concept for Fourth Generation Mobile Communication*. Eurescom Summit 2002, to appear.
[114] An Architecture Supporting Adaptation and Evolution in Fourth Generation Mobile Communication Systems**,** Robert Hirschfeld, Wolfgang Kellerer, Christian Prehofer, K. Kawamura, Hendrik Berndt, submitted
[115] A. T. Campbell, M. E. Kounavis, and R. R.-F. Liao: *Programmable Mobile Networks*. Computer Networks, Vol. 31, No. 7, pg. 741-765, April 1999.
[116] H. Harada, Y. Kamio, M. Fujise, Multimode Software Radio System by Parameter Controlled and Telecommunication Component Block Embedded Digital Signal Processing Hardware, IEICE Trans. on Communications, vol.E83-B, no.6, pp.1217.
[117] J. Pereira, Re-Defining Software (Defined) Radio: Re-Configurable Radio Systems and Networks, IEICE Transactions on Communications, vol. E83-B, no. 6 pp. 1174, 2000.
[118] S. Mann, M. A. Beach, P. A. Warr and J. P McGeehan, “Increasing the talk-time of mobile radios with efficient linear transmitter architectures,” *IEE Electronics & Communication Engineering Journal*, Vol. 13, No 2, pp., 65-76, April 2001.
[119] P. A. Warr, M. A. Beach and J. P. McGeehan, “Gain-element transfer response control for octave-band feedforward amplifiers,” *IEE Electronics Letters*, Vol. 37, No. 3, pp., 146-147, 1st February 2001.
[120] "Reconfigurable SDR Equipment and Supporting Networks Reference Models and Architectures", WWRF Working Group 3 - White paper, 2002.
[121] D. R. Musser, “Generic Programming,” http://www.cs.rpi.edu/~musser/gp/, November 2004.
[122] L. Berlemann, M. Siebert, B. Walke, “Software Defined Protocols Based on Generic Protocol Functions for Wired and Wireless Networks,” in Proc. of *Software Defined Radio Technical Conference 2003*, Orlando USA, November 2003.
[123] B. Walke, R. Pabst, L. Berlemann, D. Schultz, “Architecture Proposal for the WINNER Radio Access Network and Protocol,” *WWRF11*, Oslo Norway, June 2004.
[124] L. Berlemann, A. Cassaigne, B. Walke, B. “Generic Protocol Functions for Design and Simulative Performance Evaluation of the Link-Layer for Re-configurable Wireless Systems,” *WPMC’04*, Abano Terme Italy, September 2004.
[125] L. Berlemann, A. Cassaigne, R. Pabst, B. Walke “Modular Link Layer Functions of a Generic Protocol Stack for Future Wireless Networks,” *Software Defined Radio Technical Conference 2004*, Phoenix USA, November 2004.
[126] L. Berlemann, R. Pabst, B. Walke “A Reconfigurable Multi-Mode Protocol Reference Model Facilitating Modes Convergence,” submitted to *EW’05*, Nicosia Cyprus, April 2005.
[127] "System Requirements", IST-WINNER Project - Public Deliverables, D7.1, July 2004
[128] "Identification of Advanced Beamforming and MIMO Technologies", IST-WINNER Project, IR2.2 v1.0, July 2004.

[129] Seshaiah Ponnekanti, "An Overview of Smart Antenna Technology for Heterogeneous Networks", IEEE Communications Surveys, Fourth Quarter 1999.
[130] A. N. Barreto, M. Mecking, and G. Fettweis, "A Flexible Air interface for Integrated Broadband Mobile Systems", Proceedings of IEEE Vehicular Tech. Conference 2000.
[131] Martin Haardt, Angeliki Alexiou, WWRF/WG4/Subgroup on Smart Antennas, "Smart Antennas and Related Technologies (SMART)", White Paper, March 2003.
[132] Stefan Kaiser, Bernard Hunt, WWRF/WG4/Subgroup on New Air Interfaces, "New Air Interface Technologies - Requirements and Solutions", White Paper, April 2004.
[133] T. Marzetta, B. Hochwald, "Fast Transfer of Channel State Information in Wireless Systems", IEEE Transactions on Signal Processing, June 2004.
[134] S. Rajagopal, J. R. Cavallaro, and S. Rixner, “Design Space Exploration for Real-Time Embedded Stream Processors,” IEEE Micro, vol. 24, July-August 2004.
[135] M. Sgroi, M. Sheets, A. Mihal, K. Keutzer, S. Malik, J. Rabaey, and A. Sangiovanni-Vincentelli, “Addressing the System-on-a-Chip Interconnect Woes through Communication-based Design,” in Proceedings of the Design Automation Conference, (Las Vegas, NV), pp. 667–672, June 2001.
[136] I. Verbauwhede, P. Schaumont, C. Piguet, and B. Kienhuis, “Architectures and Design Techniques for Energy Efficient Embedded DSP and Multimedia Processing,” in Proceedings of the Design Automation and Test in Europe Conference, vol. 2, pp. 988–993, February 2004.
[137] S. Das, S. Rajagopal, C. Sengupta, and J. R. Cavallaro, “Arithmetic acceleration techniques for wireless communication receivers,” in 33rd Asilomar Conference on Signals, Systems, and Computers, (Pacific Grove, CA), pp. 1469–1474, October 1999.
[138] S. Rajagopal, B. A. Jones, and J. R. Cavallaro, “Task partitioning base-station receiver algorithms on multiple DSPs and FPGAs,” in International Conference on Signal Processing, Applications and Technology, (Dallas, TX), August 2000.
[139] C. Sengupta, J. R. Cavallaro, and B. Aazhang, “On multipath channel estimation for CDMA using multiple sensors,” IEEE Transactions on Communications, vol. 49, pp. 543–553, March 2001.
[140] B. Aazhang and J. R. Cavallaro, “Multi-tier Wireless Communications,” Kluwer Wireless Personal Communications, Special Issue on Future Strategy for the New Millennium Wireless World, vol. 17, pp. 323–330, June 2001.
[141] S. Rajagopal and J. R. Cavallaro, “A bit-streaming pipelined multiuser detector for wireless communications,” in IEEE International Symposium on Circuits and Systems (ISCAS), vol. 4, (Sydney, Australia), pp. 128–131, May 2001.
[142] K. Chadha and J. R. Cavallaro, “A Dynamically Reconfigurable Viterbi Decoder Architecture,” in Proc. Asilomar Conference on Signals, Systems and Computers, (Pacific Grove, CA), pp. 66–71, November 2001.
[143] J. R. Cavallaro and M. Vaya, VITURBO: A Reconfigurable Architecture for Viterbi and Turbo Decoding, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 497-500, Volume II, Hong Kong, China, April 2003.
[144] G. Xu, S. Rajagopal, J. R. Cavallaro, and B. Aazhang, “VLSI Implementation of the Multistage Detector for Next GenerationWideband CDMA Receivers,” Journal of VLSI Signal Processing, vol. 30, pp. 21–33, March 2002.
[145] S. Rajagopal, S. Bhashyam, J. R. Cavallaro, and B. Aazhang, “Real-time algorithms and architectures for multiuser channel estimation and detection in wireless base-station receivers,” IEEE Transactions on Wireless Communications, vol. 1, pp. 468–479, July 2002.
[146] W. R. Davis, N. Zhang, K. Camera, D. Markovic, T. Smilkstein, M. J. Ammer, E. Yeo, S. Augsburger, B. Nikolic, and R. W. Brodersen, “A Design Environment for High-Throughput Low-Power Dedicated Signal Processing Systems,” IEEE Journal of Solid-State Circuits, vol. 37, pp. 420–431, March 2002.
[147] R.W. Brodersen, W. R. Davis, D. Yee, and N. Zhang, “Wireless Systems-on-a-chip Design,” in Proc. Intl. Symposium on VLSI Technology, Systems, and Applications, (Hsinchu, Taiwan), pp. 45–48, April 2001.
[148] B. A. Jones, “Rapid Prototyping of Wireless Communications Systems,” Master’s thesis, Department of Electrical and Computer Engineering, Rice University, Houston, TX, January 2002.
[149] S. Rajagopal, S. Bhashyam, J. R. Cavallaro, and B. Aazhang, “Efficient VLSI architectures for baseband signal processing in wireless base-station receivers,” in 12th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP), (Boston, MA), pp. 173–184, July 2000.

[150] A. Bria, F. Gessler, O. Queseth, R. Stridh, M. Unbehaun, J.Wu, and J. Zander, “4th-Generation Wireless Infrastructures: Scenarios and Research Challenges,” IEEE Personal Communications, pp. 25–31, December 2001.

[151] U. Varshney and R. Jain, “Issues in Emerging 4G Wireless Networks,” IEEE Computer, pp. 94–96, June 2001.

[152] H. Corporaal, “A different approach to high perfromance computing,” in High Performance Computing, 1997. Proceedings. Fourth International Conference on, (Bangalore, India), Dec. 1997.

[153] S. Srikanteswara, J. Neel, J. H. Reed, and P. Athanas, “Designing Soft Radios for High Data Rate Systems and Integrated Global Services,” in Proc. Asilomar Conference on Signals, Systems and Computers, (Pacific Grove, CA), November 2001.

[154] I. P. Seskar and N. B. Mandayam, “A Software Radio Architecture for Linear Multiuser Detection,” IEEE Journal on Selected Areas in Communications, vol. 17, pp. 814–823, May 1999.

[155] S. W. Ellinson and M. P. Fitz, “A Software Radio-based System for Experimentation in Wireless Communications,” in IEEE Vehicular Technology Conference (VTC), vol. 3, (Ottawa, Canada), pp. 2348–2352, May 1998.

[156] J. Ou, S. Choi, and V. K. Prasanna, “Performance Modeling of Reconfigurable SoC Architectures and Energy-efficient Mapping of a Class of Applications,” in IEEE Symposium on Field-Programmable Custom Computing Machines, pp. 241–250, April 2003.

[157] C. Dick and F. J. Harris, “Configurable Logic for Digital Communications: Some Signal Processing Perspectives,” IEEE Communications Magazine, pp. 107–111, August 1999.

[158] S. Rajagopal, Data-Parallel Digital Signal Processors: Algorithm mapping, Architecture Scaling, and Workload Adaptation. PhD thesis, Rice University, 2004.

[159] M. Karkooti, “VLSI architectures for real time LDPC decoding,” Master’s thesis, Rice University, 2004.

[160] OpenCores Library. http://opencores.org/.

[161] Xilinx Programmable Logic Core Library. http://xilinx.com/.

[162] P. Willmann, M. C. Brogioli, and V. Pai, “Spinach: A liberty–based simulator for programmable network interface architectures,” in Languages, Compilers, and Tools for Embedded Systems, 2004.G. O. Young, “Synthetic structure of industrial plastics (Book style with paper title and editor),” in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.

[163] B. Lu, G. Yue, and X. Wang, “Performance Analysis and Design Optimization of LDPC-Coded MIMO OFDM Systems,” IEEE Transactions on Signal Processing, Vol. 52, No. 2, February 2004, Pages 348-361.

[164] W. Bai, L. Tang, and Z. Bu, “An Equalization Method for OFDM in Time Varying Multipath Channels,” Vehicular Technology Conference, 2004. VTC 2004-Fall. 2004 IEEE 60th, Sept. 2004, Paper 0303_16.

[165] M. B. Breinholt, M. D. Zoltowski, and T. A. Thomas, “Space-Time Equalization and Interference Cancellation for MIMO OFDM,” Proc. of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers, 2002, Volume: 2 , 3-6 Nov. 2002 Pages:1688 - 1693 vol.2

[166] S. Iraji and J. Lilleberg, “Interference Cancellation for Space-Time Block-Coded MC-CDMA Systems over Multipath Fading Channels,” Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th,Volume: 2, 6-9 Oct. 2003, Pages:1104 - 1108 Vol.2.

[167] A. Batra, J. Balakrishnan, and A. Dabak, “Multi-band OFDM: A New Approach for UWB,” ISCAS '04. Proceedings of the 2004 International Symposium on Circuits and Systems, Volume: 5, 23-26 May 2004, Pages: 365 – 368.

Contributing Authors:

Ramon Agusti, Universitat Politechnica de Catalunya (UPC), Barcelona, Spain, ramon@tsc.upc.es
Nancy Alonistioti, University of Athens, Greece, nancy@di.uoa.gr
Miguel Alvarez, Telefonica, Madrid, Spain, macalvo@tid.es
Byron Alex Bakaimis, Samsung Electronics (UK) Ltd, United Kingdom, byron.bakaimis@samsung.com
Christophe Beaujean, Motorola Labs, Paris, France Christophe.beaujean@motorola.com
Lars Berlemann, RWTH Aachen University, Germany, ber@comnets.rwth-aachen.de
Alexis Bisiaux, Mitsubishi France, bisiaux@tcl.ite.mee.com
Didier Bourse, Motorola Labs, Paris, France Didier.Bourse@motorola.com
Alexandre de Baynast, Rice University, USA, debaynas@ece.rice.edu
Jorg Brakensiek, Nokia Germany, jorg.brakensiek@nokia.com
Michael C., Brogioli, Rice University, USA, brogioli@rice.edu
Soodesh Buljore, Motorola Labs, Paris, France, Soodesh.buljore@motorola.com
Joseph R. Cavallaro, Rice University, USA, cavallar@rice.edu
Antoine Delautre, Thales Land & Joint Systems, France, antoine.delautre@fr.thalesgroup.com
Panagiotis Demestichas, University of Piraeus (UPRC), Greece, pdemest@unipi.gr
Markus Dillinger, Siemens, Germany, markus.dillinger@siemens.com
George Dimitrakopoulos, University of Piraeus (UPRC), Greece, gdimitra@unipi.gr
Terence Dodgson, Samsung Electronics (UK) Ltd, United Kingdom, terry.dodgson@samsung.com
Craig Dolwin, Toshiba, Bristol, UK, craig.dolwin@toshiba-trel.com
Peter Dornbush, Technical University of Munich, Germany, dornbusc@informatik.tu-muenchen.de
Karim El-Khazen, Motorola Labs, Paris, France, karim@motorola.com
Michael Fahrmair, Technical University of Munich, Germany, fahrmair@informatik.tu-muenchen.de
Tim Farnham, TRL, Tim.Farnham@toshiba-trel.com
Fotis Foukalas, University of Athens, foukalas@di.uoa.gr
Raquel Garcia – Perez, Telefonica, Spain rqp@trid.es
JE. Goubard, Thales Land & Joint Systems, France, Jean-Etienne.GOUBARD@fr.thalesgroup.gr
Stoytcho Gultchev, University of Surrey, UK S.Gultchev@surrey.ac.uk
Mirsad Halimic, Panasonic, Uxbridge UK, Mirsad.Halimic@panasonic-pmdc.co.uk
Kostas Kafounis, University of Athens, Greece kafounis@di.uoa.gr
Alexandros Kaloxylos, University of Athens, Greece agk@di.uoa.gr
Apostolos Katidiotis, University of Piraeus (UPRC) katidiot@unipi.gr
Dominik Lenz, Nokia, Germany, dominik.lenz@nokia.com
Ulf Lucking, Nokia, Germany, ulf.lucking@nokia.com
Jijun Luo, Siemens AG, Germany, jesse.luo@siemens.com
Luis M. Campoy, Telefonica, Spain campoy@tid.es
Jelena Mirkovic, RWTH Aachen University, Germany, jem@comnets.rwth-aachen.de
Klaus Moessner, University of Surrey, UK, K.Moessner@surrey.ac.uk
Eiman Mohyeldin, Siemens, Germany, eiman.mohyeldin@siemens.com
Syed Naveen, I2R, Singapore, naveen@i2r.a-star.edu.sg
Shailen Patel, PMDL, shailen.patel@panasonic-pmdc.co.uk
Gerald Pfeiffer, PEL, pfeiffer@panasonic.de
Christian Prehofer, DOCOMO, prehofer@docomolab-euro.com
Jacques Pulou, FTRD, jacques.pulou@francetelecom.com
Predrag Radosavljevic, Rice University, USA, rpredrag@rice.edu
Pierre Roux, Motorola Labs, Paris, France, Pierre.roux@motorola.com
Oriol Sallent, Universitat Politechnica de Catalunya (UPC), Barcelona, Spain, sallent@tsc.upc.es
Egon Schulz, Siemens, Germany, egon.schulz@siemens.com
Makis Stamatelatos, University of Athens, Greece, g.stamatelatos@di.uoa.gr
Vera Stavroulaki, University of Piraeus (UPRC), veras@unipi.gr
Bernd Steinke, Nokia, Germany, bernd.steinke@nokia.com

Jose Emilio Vila, Telefonica, Spain, jvila@tid.es
Guillame Vivier, Motorola Labs, Paris, France, Guillaume.Vivier@crm.mot.com
Jorg Vogler, PEL, vogler@panasonic.de
W. Warzansky, Telefonica Spain, wwg@tid.es
Qing Wei, DOCOMO, wei@docomolab-euro.com
Shi Zhong, TRL, Shi.zhong@toshiba-trel.com