Future Network and MobileSummit 2012 Conference Proceedings
Paul Cunningham and Miriam Cunningham (Eds)
IIMC International Information Management Corporation, 2012
ISBN: 978-1-905824-29-8

On designing next generation MAC for cellular networks using the FLAVIA paradigm

Vincenzo MANCUSO[1], Erez BITON[2], Andreas MAEDER[3], Peter ROST[3],
Nelly ANDRUSIER[4], Yaniv WEIZMAN[4], and Omer GUREWITZ[2]
[1]*Institute IMDEA Networks, Leganes (Madrid), 28918, Spain*
Email: vincenzo.mancuso@imdea.org
[2]*CSE dept., Ben Gurion University of the Negev, Beer Sheva, 84105, Israel*
Email: berez@bgu.ac.il, gurewitz@bgu.ac.il
[3]*NEC Laboratories Europe, Heidelberg, 69115, Germany*
Email: andreas.maeder@neclab.eu, peter.rost@neclab.eu
[4]*Alvarion Ltd., HaBarzel street, Tel Aviv, 69710, Israel*
Email :yaniv.weizman@alvarion.com, nelly.andrusier@alvarion.com

Abstract: Implementing a flexible and modular architecture for scheduled systems will speed-up the deployment of novel and adjustable MACs for cellular systems. Such an architecture will be of great benefit for researchers, vendors and operators by enabling off-the-shelf devices to be used for testing modified MACs, by allowing fast upgrade of existing devices, and by drastically reducing the time-to-market of MAC products, which ultimately turns into enhanced services for the users. In this paper, we show how the architecture proposed in the FLAVIA project can enable the paradigm shift towards open, modular and flexible scheduled MAC architectures, thus allowing cellular operators to use up-to-date research results to augment and promptly update the capabilities of the cellular network by creating and instantiating MAC *services*.

Keywords: Scheduled systems, modularity, flexibility, wireless services.

1. Introduction

Innovative MAC solutions require long design, test and validation processes including actual implementations on dedicated platforms. Current solutions imply a rather long time-to-market for innovations, make it very expensive to introduce protocol modifications, and prevent thorough field validation of possible modifications proposed by researchers and developers. In particular, scheduled systems are currently developed and implemented on a limited number of proprietary platforms, which completely prevent independent researchers to challenge and validate their models and proposals. Furthermore, these proprietary platforms only allow for validation of enhancements which are supported by the platform's original design.

Recently, a few projects have been initiated which envision the possibility to develop an open platform for wireless systems that would allow for a much closer cooperation of academia and industry. For example, the Femto Application Platform Interface (FAPI) project [1] suggests a common and detailed MAC API definition. The defined APIs enable a high flexibility level and the possibility to interconnect out-of-the-shelf hardware and software. However, FAPI does not specify an architecture for developing or optimizing new services and functionalities. Optimization support in scheduled systems has been the focus of the OpenAirInterface project [2], in terms of spectral, algorithmic and protocol efficiency based on open-source design tools. The architecture proposed by OpenAirInterface is subdivided into functional sub-layers. Optimization support is also one of the targets of the ACROPOLIS network of excellence [3], focusing on

Research supported by the ICT FLAVIA Project, funded by the EU Seventh Framework Programme.

cooperative and cognitive wireless systems. ACROPOLIS is a holistic and dynamic approach, where cooperation, flexibility and certain learning capabilities (cognition) can yield significant benefits for future wireless systems. However, both OpenAirInterface and ACROPOLIS do not provide modularity at MAC service level.

The FLAVIA project [4] defines a flexible, modular and virtualizable architecture for wireless access networks. In particular, a main focus of the project is to design a novel architecture for scheduled systems. Specifically, in addition to covering requirements of existing standards such as 3GPP LTE and IEEE 802.16, the FLAVIA architecture also applies to generic schedule based systems, i.e., the architecture is technology agnostic.

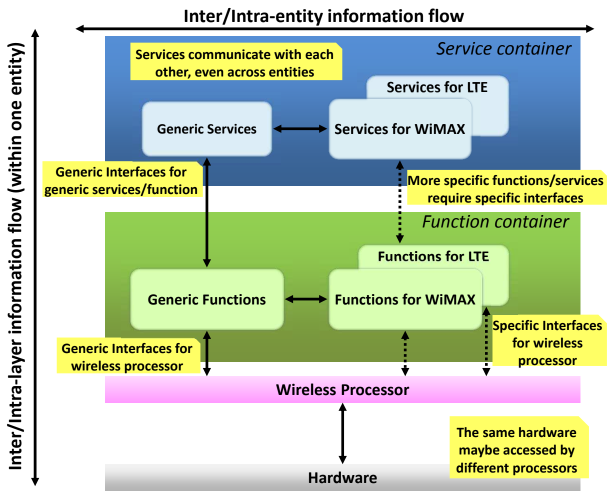
In this paper we follow the paradigm proposed in FLAVIA and show that it simplifies the implementation of wireless MAC for scheduled systems by making it flexible and modular. This paper presents the conceptual design of a scheduled MAC architecture, in which a comprehensive set of MAC functionalities—termed services—are well-defined and can be reused, replaced, reconfigured and updated with none or only minimal modifications to other services. In FLAVIA, we define a *function* as a *stateless MAC operation*, and a service as a *stateful composition of functions that can be accessed and controlled via programmable interfaces*. Services can communicate and coordinate with each other by passing data over well-defined interfaces. Accordingly, we define the following elements: (i) services and functions for scheduled system, (ii) a control system which orchestrates services and functions, and (iii) interfaces that allow the interaction of control, services, and functions.

The implementation of such architecture and its elements is out of the scope of this paper. However, we remark that FLAVIA proposes a conceptual architectural framework that could be implemented with different levels of complexity, e.g., by modularly re-writing firmwares/drivers of existing hardware, allowing for new code hooks, or by developing novel, hardware-independent wireless processors supported by novel general-purpose hardware interfaces. Similarly, the flexibility of the MAC architecture can be seen as the possibility to change the MAC operation by composing existing building blocks (functions and elementary hardware functions), which does not necessarily require the ability to change the MAC on-the-fly. For instance, a proof-of-concept of FLAVIA-style MAC design has recently appeared in [5, 6], where the 802.11 MAC has been statically modified to schedule conditional data/voice piggybacking within the MAC ACK frame.

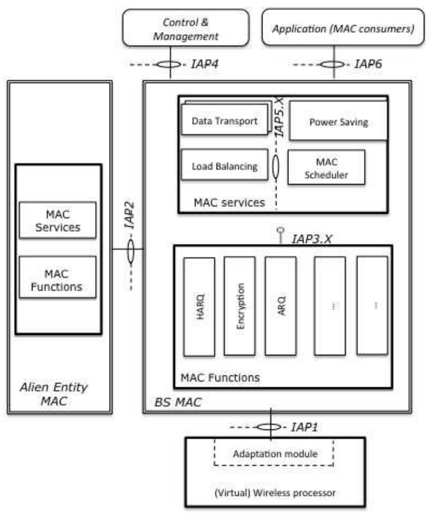
FLAVIA services for scheduled systems have been identified in [4, 7]. In this paper we first show the overall MAC service architecture, focusing on scheduling functionalities, then we illustrate the modularity and flexibility of the FLAVIA paradigm by means of a few service definition examples. Specifically, we select three services: the very fundamental *Data Transport* service (DATR), the *Link Adaptation* service (LADA), which is a typical service for scheduled MAC, and an innovative service for next generation systems, namely *Application Optimization Support* (AOPS). For each service, we give a description of functionalities and interfaces. Finally, we delineate how the interaction of modules can be used to optimize the system.

2. Architecture overview

FLAVIA's high level MAC architecture for scheduled systems is composed of modules, each representing a service that a scheduled access technology is expected to provide. Each service is a composition of MAC functions which are atomic building


Inter/Intra-entity information flow
Inter/Intra-layer information flow (within one entity)
Service container
Services communicate with each other, even across entities
Services for LTE
Generic Services
Services for WiMAX
Generic Interfaces for generic services/function
More specific functions/services require specific interfaces
Function container
Functions for LTE
Generic Functions
Functions for WiMAX
Specific Interfaces for wireless processor
Generic Interfaces for wireless processor
Wireless Processor
The same hardware maybe accessed by different processors
Hardware

(a) Scheduled MAC modular architecture and information flow description.



(b) Interface architecture for scheduling systems using the FLAVIA paradigm.

Figure 1: Scheduled MAC modular architecture and interfaces.

blocks/operations and have limited access to services (e.g., to notify an event to a service) or to the hardware modem (e.g., to the *wireless processor*, using FLAVIA's terminology). A function can be reused by multiple services, i.e., services can have common functions. The high-level FLAVIA architecture for scheduled access systems (e.g., 802.16 and LTE) is depicted in Figure 1(a).

As illustrated in Figure 1(a), services can be classified into generic services and technology-specific services. Generic services are common for all scheduled systems, while technology-specific services are only needed in a subset of possible MAC implementations. Nonetheless, generic services have to be tailored to the technology where they are applied, i.e., generic services can behave differently when applied to LTE or 802.16. Accordingly, MAC functions can be classified into generic functions and technology-specific functions [4].

MAC services offer complex operations with rich interfaces towards the control subsystem, other services located on the same machine or on a remote device, applications, and MAC functions. Interfaces are defined by *primitives* of four different types: *request* (REQ), meant to receive a command; *reply* (REP), used to reply to a REQ with the output of the invoked command; *subscribe* (SUB), meant to register a module for event notification on a different module; *indication* (IND), used to notify an event to the list of subscribers for that event.

Interfaces are grouped in *Interface Access Points* (IAP), according to the entities they connect (see Figure 1(b)). Furthermore, interfaces can be partitioned into two groups: (*i*) External interfaces which are used to connect the MAC to the wireless processor, other MACs, the control subsystem, and higher layers (IAP1, 2, 4 and 6 in Figure 1(b)); (*ii*) Internal interfaces which are used to connect different MAC components, such as services to services or services to functions (IAP3 and 5 in Figure 1(b)).

Note that IAP3, between services and functions, is required to contain some standard interfaces. These interfaces allow for the deployment of generic services and technology-specific interfaces. Furthermore, technology-specific interfaces allow for the deployment of technology-specific services through generic and technology-specific functions. In particular, IAP3 is used by MAC services in order to access the MAC functions needed to compose the service. The availability of IAP5, providing a set of inter-service interfaces, allows for the development of modular, programmable, and flexible MAC services.

Note also that services can communicate with other services running on the same device (IAP5) or other MAC entities (IAP2). On the contrary, MAC functions can only communicate with services on the same MAC entity (limited access on IAP3), and with the wireless processor running on top of the available hardware (IAP1).

3. MAC service architecture

A service oriented architecture is designed of small pluggable software units or services rather than large centralized software [8]. The service design is optimized for modularity by a loose coupling and interoperability of elements. FLAVIA services and their basic interaction are depicted in Figure 2. Each arrow in the figure corresponds to an interface between services.

The scheduler is the main building block of any scheduled system. Here it is composed of three services. First, *QoS Strategy* handles the support for different QoS classes and can assign different scheduling strategies to subsets of connections. Second, *Scheduling Strategy* defines different scheduling schemes incorporating link adaptation inputs. Both QoS Strategy and Scheduling Strategy services can be designed to be technology-agnostic. Third, *MAC Scheduler* manages the physical frame building process by getting a candidate-list from the QoS Strategy and applying relevant scheduling strategies. This service is technology-specific.

Another important service which is central to the radio resource and interference management is the *Link Adaptation* service. This service aims at optimizing the link performance by adapting transmission parameters in a varying channel environment. *Data Transport* is a fundamental service for receiving and sending packets. Among others, it is responsible for packet manipulation such as multiplexing, encryption, header encapsulation, and segmentation. Finally, the *Application Optimization* service aims at optimizing specific application protocols by dedicated MAC operations. In addition, FLAVIA defines services for power saving, load balancing, admission control, inter-cell coordination and cooperation, measurement and other functionalities.

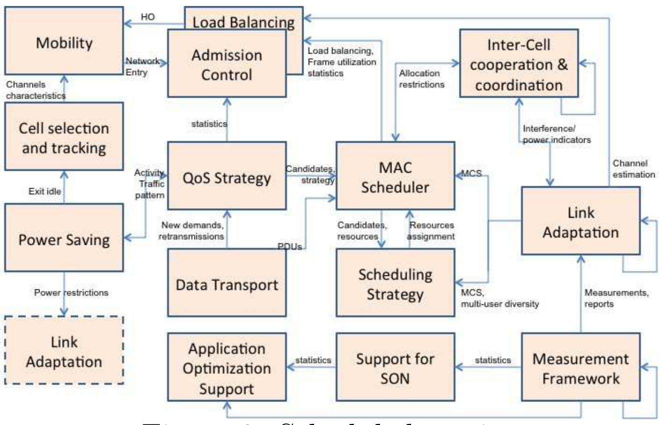


Figure 2: Scheduled services.

Due to space limitations, we omit here the description of many services' operation, and use Figure 2 to illustrate the data flow between a number of specific services. In particular, in the following we describe a transmission cycle. When the Data Transport receives new data to transmit, it informs the QoS Strategy service of new demands. The QoS Strategy prepares candidates for scheduling and decides on the specific scheduling strategy to be used. These pieces of information are passed to the MAC Scheduler, that decides about physical resources per group of candidates. The Scheduling Strategy service finally evaluates how to schedule the given candidates on the given resources. Modulation and coding schemes are provided by the Link Adaptation, so that, with the resource assignment at hand, the Data Transport service builds MAC PDUs, which are transmitted on the assigned physical resources.

In the following subsections, we move the focus to a more detailed design and interface definition for three representative services, namely Data Transport, Link Adaptation, and Application Optimization Support.

3.1 Data transport

The Data Transport service (DATR) provides forwarding and processing of user or control data before passing data to the physical layer. Data can be transported with protocols to increase transport reliability, e.g., encryption, ARQ or HARQ. DATR provides an interface towards higher layers which expects SDUs from a service flow. Several service flows can be multiplexed. The actual functionality of this service is illustrated in Figure 3.

The responsibilities of this service comprise the following tasks: (*i*) Receive and deliver MAC SDUs from and to the MAC consumer; (*ii*) Transport of MAC messages created inside the MAC, such as MAC management messages; (*iii*) Encryption of user and control plane data; (*iv*) Fragmentation and packing of MAC SDUs; (*v*) Management of ARQ and HARQ of MAC SDUs priorities; (*vi*) Manipulating and forwarding of uplink received packets; and (*vii*) Multiplexing and demultiplexing of packet flows.

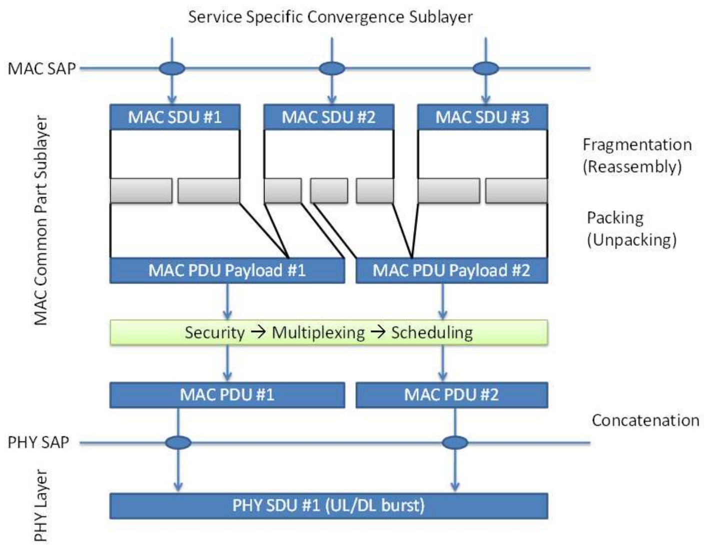


Figure 3: Data transport service.

Some service interface primitives of the DATR service are presented in Table 1. Note that due to lack of space, we only show a selected subset of primitives.

Table 1: Examples of service interface primitives of DATR

IAP	Primitive name	Description
IAP6	DATR_IAP6_add_sdu_REQ/REP	Adds SDU (to be sent) to the data buffer of a user's connection.
IAP5	DATR_IAP5_add_mac_msg_REQ/REP	Adds MAC message (to be sent) to the data buffer of a user's connection.
IAP5	DATR_IAP5_statistics_REQ/REP	Returns the current buffer statistics (fill level, delays, etc.) of a user's connection (downlink specific).
IAP5	DATR_IAP5_fill_resource_assignments_REQ/REP	Fills the given resource assignments with data from users' connections. It also invokes internal procedures to fragment SDUs, pack payload, apply CRC/ARQ, encrypt, and multiplex PDUs. Returns the final list of PDUs.
IAP5	DATR_IAP5_register_class_REQ	Request to update the class-filtering rules (from AOPS).

3.2 Link adaptation

The Link Adaptation (LADA) service aims to optimize the link performance by adapting transmission parameters. LADA builds upon the support of adaptive modulation and coding to improve link robustness. The support of multiple modulation schemes as well as MIMO diversity modes allows for the transmission to change on a burst-by-burst basis per link and depending on channel conditions. Using the channel quality feedback indicator, the mobile can provide the base station with feedback on the downlink. Specifically, LADA is responsible for the following operations: (i) *Collecting and analyzing the link status statistics*—collects statistics on the link quality in terms of SINR, packet error rate, HARQ retransmissions, transmission failures; (ii) *Setting the transmission parameters*—computes and applies the transmission parameters, including: modulation and coding scheme, transmission diversity scheme, multi-user scheme, and transmission power; (iii) *Interference management*—collects and distributes power and interference reports from and to neighboring cells, and limits the transmission power accordingly. Some relevant service interface primitives used for LADA operations are presented in Table 2.

Table 2: Examples of service interface primitives used with LADA

IAP	Primitive name	Description
IAP2	MEAS_IAP2_meas_REQ	Request link meas. from other MAC entities (e.g., CQI).
IAP4	LADA_IAP4_configure_REQ	Allows the FLAVIA control to configure the service.
IAP5	LADA_IAP5_PHY_profile_REQ	Output the transmission parameter (i.e., modulation and coding scheme, transmit diversity, transmission power, etc.) based on the link state, allocation size and power budget.
IAP5	DATA_IAP5_harq_feedback_IND	Notify about HARQ ack/nack.
IAP5	MEAS_IAP2_meas_REQ	Request link meas. (e.g., Noise Interference, RSSI, SNR)

3.3 Application optimization support

The Application Optimization Support (AOPS) service is a novel modular component meant to enforce automatic optimization of MAC operation as a response to application needs. Moreover, AOPS might be used to provide applications with timely updated MAC and PHY operational parameters.

As an example, AOPS could use MAC and channel statistics computed within the MAC to run optimization algorithms and suggest to update the MAC configuration accordingly. Thereby, AOPS is required to be able to fetch statistics, detect running applications, and propose MAC and PHY parameters that better suit application needs. In particular, AOPS is responsible for the following operations: (i) provide support for handling traffic classes, e.g., define and apply filters to be used to detect running applications; (ii) suggest how to tune the queuing and scheduling system on a per-class basis; (iii) configure MAC and PHY parameters by running local algorithms based on the statistics available for, e.g., bandwidth utilization, channel state information, and currently active applications; (iv) provide an interface to upper layers to present MAC and PHY statistics and configuration settings. A few examples of service interface primitives offered by AOPS are presented in Table 3, showing how to manage configuration changes at system control level (over IAP4), MAC consumer level (over IAP5), and inter-service level (over IAP6).

Table 3: Examples of service interface primitives offered by AOPS

IAP	Primitive name	Description
IAP4	AOPS_IAP4_configure_REQ	Allow the control subsystem to tune the AOPS service.
IAP4	AOPS_IAP4_update_config_IND	Suggest a new configuration to the control subsystem.
IAP5	AOPS_IAP5_new_config_IND	Notify changes to other services.
IAP6	AOPS_IAP6_config_application_REQ	Allow the upper layers (i.e., the MAC consumer) to request new per-application requirements and rules.
IAP6	AOPS_IAP6_notify_status_IND	Deliver status info to the MAC consumer.
IAP6	AOPS_IAP6_net_info_IND	Present to application layer information relevant for the optimization of running applications.

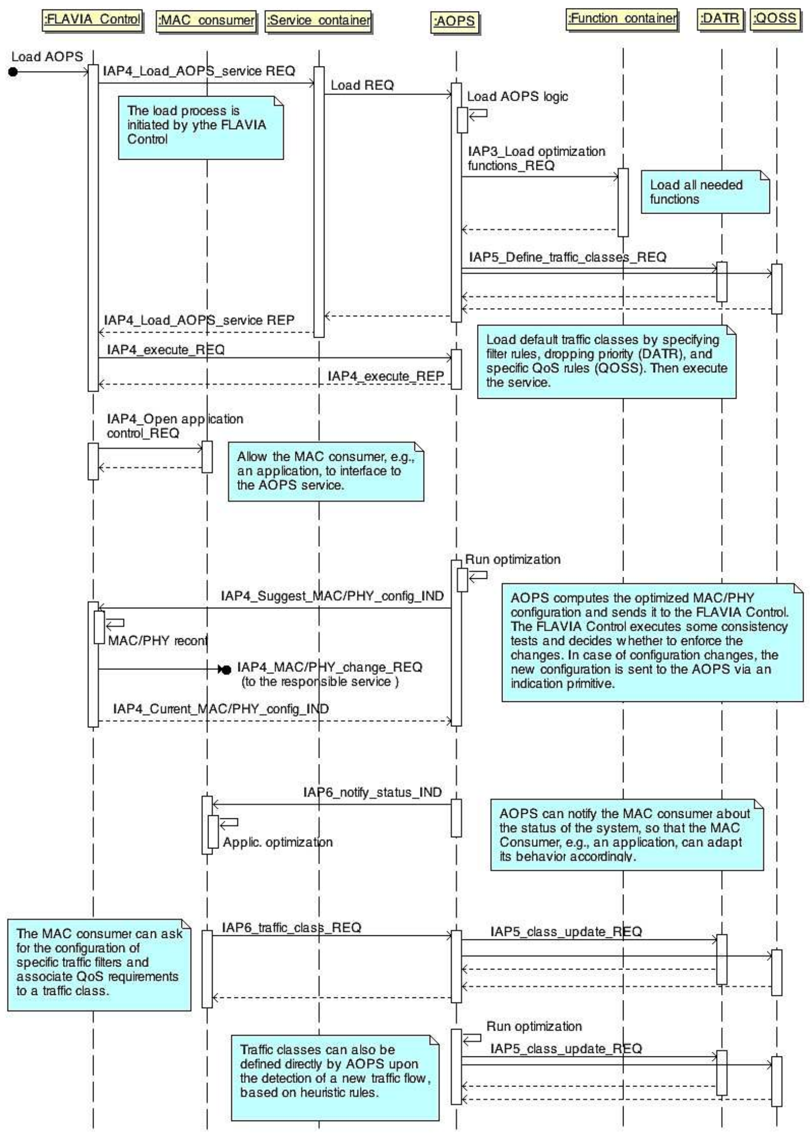

:FLAVIA_Control
:MAC_consumer
:Service_container
:AOPS
:Function_container
:DATR
:QOSS
Load AOPS
IAP4_Load_AOPS_service REQ
Load REQ
Load AOPS logic
The load process is initiated by ythe FLAVIA Control
IAP3_Load optimization functions_REQ
Load all needed functions
IAP5_Define_traffic_classes_REQ
IAP4_Load_AOPS_service REP
Load default traffic classes by specifying filter rules, dropping priority (DATR), and specific QoS rules (QOSS). Then execute the service.
IAP4_execute_REQ
IAP4_execute_REP
IAP4_Open application control_REQ
Allow the MAC consumer, e.g., an application, to interface to the AOPS service.
Run optimization
IAP4_Suggest_MAC/PHY_config_IND
AOPS computes the optimized MAC/PHY configuration and sends it to the FLAVIA Control. The FLAVIA Control executes some consistency tests and decides whether to enforce the changes. In case of configuration changes, the new configuration is sent to the AOPS via an indication primitive.
MAC/PHY reconf
IAP4_MAC/PHY_change_REQ (to the responsible service)
IAP4_Current_MAC/PHY_config_IND
IAP6_notify_status_IND
AOPS can notify the MAC consumer about the status of the system, so that the MAC Consumer, e.g., an application, can adapt its behavior accordingly.
Applic. optimization
The MAC consumer can ask for the configuration of specific traffic filters and associate QoS requirements to a traffic class.
IAP6_traffic_class_REQ
IAP5_class_update_REQ
Run optimization
Traffic classes can also be defined directly by AOPS upon the detection of a new traffic flow, based on heuristic rules.
IAP5_class_update_REQ

Figure 4: Basic operation of the AOPS service.

4. Modular and flexible MAC optimization Using FLAVIA

In this section we elucidate how the FLAVIA architecture for scheduled systems can be used to design a modular and flexibly scheduled MAC with support for operator-defined optimization of MAC and applications. In particular, we describe how the AOPS service can be loaded and configured, and how it can interact with other architectural components in order to support MAC and application optimization.

As shown in the topmost part of Figure 4, the AOPS service is loaded and launched by the control-subsystem. In Figure 4, for the sake of readability, we omit naming most of the exchanged _REP primitives, which travel in the diagram over dashed arrows. The service loading procedure involves the control-subsystem and repositories that contain the code to be executed to run the AOPS service (Service Container) and to load the specific service functions (Function Container). Additionally, other active services might have to be reconfigured upon AOPS activation. Specifically, Data Transport (DATR) and QoS scheduling (QOSS) have to be configured through inter-service interfaces (over IAP5) in order to enable traffic differentiation with the parameters suggested by AOPS (e.g., dropping policies, QoS rules, modulation schemes allowed).

During the loading procedure, only default traffic classes and QoS rules are loaded. Therefore, after having started the AOPS execution, the control subsystem tells the

MAC consumers that AOPS is running and accepting traffic classification requests.

AOPS periodically computes optimal MAC and PHY parameters and detects running applications according to the active rules. Therefore, AOPS periodically request updates of MAC and PHY configurations, while the control subsystem is responsible for actually enforcing the update and notifying the changes to AOPS.

The diagram in Figure 4 also shows the interaction between MAC consumer and AOPS. On the one hand, when AOPS detects, determines, or is notified of configuration changes and system status updates, it can in turn notify the MAC consumer of the new network status. As a consequence, the MAC consumer can adapt its traffic behavior accordingly. On the other hand, if the MAC consumer decides to introduce a new traffic class, it sends a traffic class request over IAP6. Finally, the bottom of the diagram in Figure 4 depicts the case in which the optimization routine detects the presence of traffic flows that can be dealt with by means of a new traffic class. As a consequence, AOPS will automatically determine the parameters to be used for the new traffic class, and the interfaces to DATR and QOSS services in order to deploy the new traffic class (alternatively, AOPS can ask the FLAVIA Control to deploy the traffic class).

5. Conclusions

We have shown how the paradigm proposed by the FLAVIA project can allow for the design of a flexible, modular and programmable MAC for scheduled systems. The proposed MAC architecture has been exemplified by a number of *services* and *interfaces*. We have provided examples of basic and advanced scheduled MAC services, and we have shown how these services can be orchestrated to implement the desired and operator-defined MAC features. In particular, we have detailed the operation of an advanced service meant to support the optimization of MAC and PHY configuration as well as application behavior through cross-layer interaction of MAC users and MAC services.

In conclusion, we have illustrated how the adoption of the FLAVIA paradigm would pave the way for the deployment of novel and adjustable MACs for cellular systems. Indeed, such a flexible and programmable MAC would be of great benefit to researchers, vendors and operators, because it would allow for fast and easy tests, validation, implementation and deployment of innovative MAC features.

References

[1] The Femto Forum, http://femtoforum.org/femto/technical.php

[2] C. Bonnet, D. Camara, R. Ghaddab, A. Hayar, L. Iacobelli, F. Kaltenberger, R. Knopp, B. Mercier, N. Nikaein, D. Nussbaum, E. Yilmaz, and B. Zayen, "Openairinterface and agile spectrum access," in *New Frontiers in Dynamic Spectrum Access Networks (DySPAN), 2011 IEEE Symposium on*, pp. 662 –663, may 2011.

[3] ICT ACROPOLIS, European Network of Excellence, http://www.ict-acropolis.eu/

[4] ICT FLAVIA project, http://www.ict-flavia.eu/

[5] P. Salvador, F. Gringoli, V. Mancuso, P. Serrano, A. Mannocci, and A. Banchs, "VoIPiggy: Implementation and evaluation of a mechanism to boost voice capacity in 802.11 WLANs," in *Proceedings of INFOCOM mini-conference*, (Orlando, Florida, USA), March 2012.

[6] I. Tinnirello, G. Bianchi, P. Gallo, D. Garlisi, F. Giuliano, and F. Gringoli, "Wireless mac processors: Programming mac protocols on commodity hardware," in *Proceedings of INFOCOM*, (Orlando, Florida, USA), March 2012.

[7] A. Maeder, V. Mancuso, Y. Weizman, E. Biton, P. Rost, X. Perez-Costa, and O. Gurewitz, "FLAVIA: Towards a Generic MAC for 4G Mobile Cellular Networks," in *Future Network & Mobile Summit 2011*, (Warsaw, Poland), June 2011.

[8] M. Bell, *Service-Oriented Modeling: Service Analysis, Design, and Architecture*. Wiley & Sons, 2008.