# A survey on the programmability of wireless MAC protocols

# A Survey on the Programmability of Wireless MAC Protocols

Pedro Heleno Isolani*, Maxim Claeys*, Carlos Donato*, Lisandro Zambenedetti Granville†, Steven Latré*

* University of Antwerp - imec,
IDLab - Department of Mathematics and Computer Science
Sint-Pietersvilet 7, 2000 Antwerp, Belgium
E-mail: {pedro.isolani, maxim.claeys, carlos.donato, steven.latre}@uantwerpen.be

† Federal University of Rio Grande do Sul - UFRGS,
Institute of Informatics - Computer Networks Group
Av. Bento Gonalves, 9500 - Bloco IV - Agronomia 91501-970 - Porto Alegre, RS - Brazil
E-mail: granville@inf.ufrgs.br

***Abstract*—Self-organizing networks able to adapt to changes in the environment have already been a longstanding research topic. Given the limited number of license-free Industrial, Scientific, and Medical (ISM) radio bands, wireless technologies end up competing with one another for the wireless spectrum. As such, the proper employment of Medium Access Control (MAC) protocols is essential to guarantee efficient and reliable wireless communication. At the data link level, there has been extensive research towards programmable and more future-proof MAC protocols (*e.g.,* Software-Defined Radios (SDRs), which enable to reconfigure the entire protocol and hence access/control fine-grained radio functionalities). However, actual deployments are so far limited because of performance issues and cost. With the increasing popularity of Software-Defined Networking (SDN), also in the wireless domain, and the increasing performance of SDRs, we are evolving into a fully programmable data link layer. In this survey, we deliver: a landscape of the state-of-the-art on programmable MAC protocols; a coherent terminology that represents scope and level of programmability supported; an in-depth study of their advantages and disadvantages; and a discussion about future research challenges on MAC programmability. Many surveys have investigated the use of specific MAC protocols for a wide range of optimization criteria and application demands. This survey is the first that investigates the *scope* and the *level* of programmability that MAC protocols support.**

## I. Introduction

The presence of wireless communication technologies is increasing significantly, especially because of the rise of the Internet of Things (IoT) [1]. Given many of these technologies compete with one another in the same shared environment and have limited number of license-free Industrial, Scientific, and Medical (ISM) radio bands, the proper employment of wireless Medium Access Control (MAC) protocols is essential to guarantee efficient and reliable wireless communication [2]. Over the last decades, wireless MAC protocols have been proposed as typical hardware-specific implementations, designed as a single building-block where the data-link layer is tightly coupled with the Physical Layer (PHY). Therefore, as these implementations treat MAC as a non-modifiable building-block, countless implementations have been proposed aiming at the different wireless technologies and scenarios [3]–[14].

### A. Motivation: Need for Wireless MAC Programmability

Despite the wide range of wireless MAC protocols and the many heterogeneous requirements, there is no one-size-fits-all solution. As application demands become more strict and wireless networks become larger and more heterogeneous, devices have to be able to effectively use and share the spectrum. In addition, because of changing topologies and application requirements, on-the-fly adaptations must be supported. As a consequence, the field is going towards a more flexible and programmable MAC layer [15]–[17]. In these proposals, the authors have designed solutions where a set of predefined MAC protocols can be selected based on performance degradation thresholds or predicted future traffic patterns. However, given the inability to interact with those predefined implementations in an easy and efficient way, these approaches are unable to cope with dynamic and unstable environments. For instance, management entities cannot improve wireless connectivity and energy efficiency by interacting with MAC-specific features (*e.g.*, the back-off mechanism of a *contention-based* protocol or the duty-cycle of a *schedule-based* protocol).

Looking for more control over the MAC layer, plenty of software-based approaches have been proposed as programmable MAC frameworks and software overlays [2] [40]–[47]. These approaches often take advantage of Software-Defined Radios (SDRs) to introduce solutions that can coordinate or even act as the traditional hardware-based implementations. Given the rise of SDRs and the Software-Defined Networking (SDN) paradigm, there have been many discussions about the trade-off between efficiency—offered by the hardware-specific implementations—and the flexibility provided by software-based approaches. The main argument in favor of adopting hardware-based approaches has long been the fact that software-based implementations fail to achieve timing requirements, resulting in poor performance [48]. However, as a consequence of the evolution of radio technologies, software-based implementations are achieving today satisfactory performance compared to hardware-specific ones. Therefore, the popularity and usage of programmable MAC

TABLE I
RELATED SURVEYS ARTICLES ON WIRELESS MAC PROTOCOLS AND THEIR MAIN TARGETS.

| **Authors** | **Year** | **Main Target** | **Focus** | **Technology** |
|---|---|---|---|---|
| H. Peyravi [18] | 1999 | Classify MAC protocols based on five classes: *fixed assignments*, *demand assignment*, *random access*, *hybrid* of *random access* and *reservation*, and *adaptive* protocols. | Mode-of-operation Performance Reconfigurability | Satellite Networks |
| A. C. V. Gummalla and J. O. Limb [19] | 2000 | Classify MAC protocols based on architecture design, mode-of-operation, performance, and application domain. | Mode-of-operation Architecture | Wireless Networks |
| S. Kumar, V. S. Raghavan, and J. Deng [20] | 2006 | Classify MAC protocols based on their brief description, mode-of-operation, and underlying features. | Mode-of-operation Underlying Features | Ad Hoc Networks |
| I. Demirkol, C. Ersoy, and F. Alagoz [21] | 2006 | Describe MAC protocols emphasizing energy consumption, strengths and weaknesses. | Mode-of-operation Energy efficiency | WSNs |
| T. V. Krishna and A. Das [22] | 2009 | Compare MAC protocols for centralized and decentralized Opportunistic Spectrum Access (OSA) networks. | Architecture Reconfigurability | CRNs |
| C. Cormio and K. R. Chowdhury [23] | 2009 | Compare MAC protocols according to its features and the different modes-of-operation. | Mode-of-operation Underlying features | CRNs |
| M. J. Booysen, S. Zeadally, and G. J. van Rooyen [24] | 2011 | Survey the different MAC protocols focusing on the benefits and limitations of their mode-of-operation on future deployments. | Mode-of-operation Performance | Vehicular Ad Hoc Networks |
| S.-L. Tsao and C.-H. Huang [25] | 2011 | Present a survey and an experimental study regarding energy consumption of different MAC protocols. | Mode-of-operation Energy efficiency | WLANs |
| Zhao *et al.* [26] | 2012 | Present unique features of Wireless Sensor Networks (WSN) and classify MAC protocols based on their mode-of-operation. | Mode-of-operation Performance | WSNs |
| Gallego *et al.* [27] | 2012 | Identify the key functions supported by MAC prototyping platforms and compare the existing ones. | Prototyping Platforms | Wireless Networks |
| P. Suriyachai, U. Roedig, and A. Scott [28] | 2012 | Survey MAC protocols with different modes-of-operation that can serve mission-critical applications. | Mode-of-operation Performance | WSNs |
| Rahim *et al.* [29] | 2012 | Compare MAC protocols with different modes-of-operation aiming for energy efficiency. | Mode-of-operation Energy efficiency | WBANs |
| P. Ju, W. Song, and D. Zhou [30] | 2013 | Classify MAC protocols according to how user cooperation issues are addressed in *contention-based* protocols. | User Cooperation | Wireless Networks |
| Chen *et al.* [31] | 2014 | Present a comparative study of MAC protocols with different modes-of-operation in underwater WSN. | Mode-of-operation Performance | WSNs |
| A. S. Althobaiti and M. Abdullah [32] | 2014 | Compare MAC protocols according to their modes-of-operation and their features to reduce energy consumption. | Mode-of-operation Energy efficiency | WSNs |
| F. Alfayez, M. Hammoudeh, and A. Abuarqoub [33] | 2015 | Classify low duty-cycle MAC protocols into synchronous and asynchronous based on their mode of operation. | Mode-of-operation Energy efficiency | WSNs |
| Liao *et al.* [34] | 2015 | Survey *random access-based* MAC protocols that support Multi-user Multiple-Input and Multiple-Output (MU-MIMO). | Performance | WLANs |
| A. Balobaid [35] | 2016 | Survey study regarding energy-efficient MAC protocols according to their mode-of-operation. | Mode-of-operation Energy Efficiency | WSNs |
| R. Sadeghi, J. P. Barraca, and R. L. Aguiar [36] | 2017 | Compare existing cooperative proposals based on mode-of-operation, architecture, and model of cooperation. | Mode-of-operation MAC Cooperation | WLANs |
| A. A. Khan, M. H. Rehmani, and M. Reisslein, [37] | 2017 | Evaluate the requirements and key design challenges for routing and MAC protocols. | Mode-of-operation Underlying features | CRNs |
| N. Z. b. Zubir, A. F. Ramli, and H. Basarudin [38] | 2017 | Evaluate applications of Machine Learning (ML) at the MAC layer to improve throughput, energy efficiency, and latency. | Mode-of-operation Performance | WSNs |
| Zareei *et al.* [39] | 2018 | Survey mobility-aware MAC protocols according to synchronization aspects and underlying features. | Mode-of-operation Mobility | WSNs |

protocols are expected only to increase where application requirements and network conditions often change, which deserves proper attention [49].

## B. Review of Related Survey Articles

There exist several survey articles that summarize the state-of-the-art on wireless MAC protocols. Given to their target technology and optimization criteria, wireless MAC protocols have been classified according to their mode-of-operation and resulted/expected behavior. The mode-of-operation of a MAC protocol consists of the logic and the mechanisms employed to access the shared medium. In 1999, Peyravi [18] investigated five different classes of MAC protocols with regards to their application on satellite communications. In the following year, Gummalla and Limb [19] also focused on analyzing protocols with different modes-of-operation but looking at their architectures and application domains in which they are best deployed. Since then, several others have been proposing classifications of MAC protocols [20] [21] [23]–[26] [28] [29] [31]–[36] [38].

Table I lists a set of surveys that address MAC protocol classification in wireless networks. Most of the surveys are focused on two metrics: *performance* and *energy efficiency*. For instance, in WSN, given the hazardous terrain in which sensor devices are usually deployed, batteries may not be easily replaceable or rechargeable [32]. In this case, MAC protocols were analyzed considering how efficient the use of medium access mechanisms is to save energy while keeping acceptable performance. In addition, surveys [18] [20] [23]–[25] [29] [36] analyzed similar aspects, but applied to other scenarios and technologies (*e.g.*, different MAC protocols in Ad Hoc Networks, Cognitive Radio Networks (CRNs), Wireless Body Area Networks (WBANs), and Wireless Local Area Networks (WLANs)). However, the principle of analyzing the mode-of-operation versus the resulted *performance* and *energy efficiency* of MAC protocols remains.

Some other surveys [22] [27] [30] focused on the identification of key functions and requirements that MAC protocols and MAC prototyping platforms should comprise so to support time-critical, delay-sensitive, and user cooperation features. Krishna and Das [22] presented a comparison of the essential features of the different MAC protocols for OSA networks. They have analyzed several MAC protocols and their impact depending on the network topology that they are employing. Gallego *et al.* [27] identified the key functions to be supported by a MAC prototyping platform. The authors argued that the steep learning curve required for protocol prototyping has forced researchers to employ off-the-shelf hardware as an inexpensive prototyping environment based on commercial Network Interface Cards (NICs), which only provides limited flexibility and partial control over the MAC layer. Ju *et al.* [30], on the other hand, presented a different classification where MAC protocols, from a single category of mode-of-operation, are classified based on the available features to deal with user cooperation.

Given the aforementioned overall increase in network density, recent surveys [36] [39] analyze aspects such as MAC protocols cooperation and mobility features. In addition, others [37] [50] have analyzed important attributes of existing MAC protocols for Cognitive Radio (CR)-based smart grid networks. The authors have presented the design requirements and the challenges that such protocols impose (*e.g.*, interoperability between licensed/unlicensed bands, channel access delay, and the trade-off between energy and spectrum efficiency). However, to the best of our knowledge, ours is the first survey that investigates the *scope* and the *level* of programmability that MAC protocols support. In addition, we highlight the challenges of the state-of-the-art on the programmability of wireless MAC protocols. Although programmability does offer the freedom to interact with the MAC layer implementation, it is also important to analyze how such programmability is addressed in different scenarios with different network requirements. Therefore, we provide our overview of the evolution from small/limited MAC parameter configurations to the design of a complete software-defined MAC layer.

## C. Summary of Contributions

In summary, we present in this survey the following contributions:

- We provide an overview of the state-of-the-art on programmable MAC protocols in wireless networks as well as the evolution of SDRs and the SDN paradigm, which have fostered the research in the field;
- We define a coherent terminology to describe the *scope* and the *level* of programmability supported by MAC protocols;
- We provide an in-depth study of the pros and cons of each of the most relevant and recent wireless MAC protocol available in the literature and we classify them according to our terminology;
- We provide a discussion about research challenges on the programmability of MAC protocols for wireless networks in general.

TABLE II
LIST OF ACRONYMS AND CORRESPONDING DEFINITIONS.

| Acronym | Definition |
|---|---|
| AI | Artificial Intelligence |
| ASIC | Application-Specific Integrated Circuit |
| API | Application Programming Interface |
| C2C | Contend-to-Coordinate |
| CQI | Channel Quality Indicator |
| CR | Cognitive Radio |
| CRNs | Cognitive Radio Networks |
| CRSNs | Cognitive Radio Sensor Networks |
| CSMA | Carrier-Sense Multiple Access |
| CSMA/CA | Carrier-Sense Multiple Access with Collision Avoidance |
| D2D | Device-to-Device |
| DCF | Distributed Coordination Function |
| DSA | Dynamic Spectrum Access |
| DSL | Digital Subscriber Line |
| FAPI | Femto Application Platform Interface |
| FD | Full-Duplex |
| FDD | Frequency-Division Duplex |
| FiWi | Fiber-Wireless |
| FPGA | Field-Programmable Gate Array |
| GCP | Global Control Plane |
| GPP | General-Purpose Processor |
| GPS | Global Positioning System |
| GUI | Graphical User Interface |
| HAL | Hardware Abstraction Layer |
| HD | Half-Duplex |
| HDL | Hardware Description Language |
| IC | Integrated Circuit |
| IoT | Internet of Things |
| ISM | Industrial, Scientific, and Medical |
| KPI | Key Performance Indicator |
| LBT | Listen Before Talk |
| LPL | Low Power Listening |
| LTE | Long-Term Evolution |
| LTE-U | Long-Term Evolution-Unlicensed |
| LTE-LAA | LTE Licensed-Assisted Access |
| MAC | Medium Access Control |
| MANET | Mobile Hoc Networks |
| MI | Management Interface |
| ML | Machine Learning |
| MLME | Medium Access Control Sublayer Management Entity |
| MCS | Modulation and Coding Scheme |
| MU-MIMO | Multi-user Multiple-Input and Multiple-Output |
| nAPI | Network API |
| NIC | Network Interface Card |
| OSA | Opportunistic Spectrum Access |
| OPEX | Operating Expense |
| PFSM | Programmable Finite State Machine |
| PHY | Physical Layer |
| QoS | Quality of Service |
| RBA | Role-Based Architecture |
| RF | Radio Frequency |
| RSSI | Received Signal Strength Indication |
| RTS/CTS | Request to Send/Clear to Send |
| SDMAC | Software-Defined MAC |
| SDN | Software-Defined Networking |
| SDR | Software-Defined Radio |
| SIMD | Single Instruction Multiple Data |
| SINR | Signal-to-Interference-plus-Noise Ratio |
| TDD | Time-Division Duplex |
| TDMA | Time-Division Multiple Access |
| USRP | Universal Software Radio Peripheral |
| VNF | Virtual Network Function |
| V2V | Vehicle-to-Vehicle |
| WBANs | Wireless Body Area Networks |
| WLANs | Wireless Local Area Networks |
| WSNs | Wireless Sensor Networks |
| XFSM | Extended Finite State Machine |

### D. Article Structure

In this article, Table II lists the symbols introduced in this section. The remainder of this survey is organized as follows. In Section II, we review and motivate the need for MAC programmability. In Section III, we present a coherent terminology for programmable MAC protocols. In Sections IV, V, and VI, we discuss the lessons learned from the most recent and relevant programmable wireless MAC protocols. In Sections VII and VIII, we discuss the general evolution of wireless MAC protocols and identify the major challenges of programmability of MAC protocols. Finally, we present conclusions and final remarks in Section IX.

## II. The Need for MAC Programmability

The literature in the area reveals an extensive list of MAC protocols with different architectures and modes-of-operation. Typically, MAC protocols are hardware-specific *monolithic* building blocks, designed to optimize wireless connectivity and energy efficiency in specific and well-known scenarios. However, as time went on, application demands became higher, and networks became bigger and more heterogeneous. Different wireless technologies started to have to coexist and share the same spectrum at the same time (*e.g.*, Wi-Fi, Long-Term Evolution (LTE), and Bluetooth). Thus, MAC layer adaptability became an essential aspect to be considered, especially in the presence of coexisting wireless technologies.

### A. MAC Layer Adaptability

Since application-layer requirements often change and given that *monolithic* designs cannot cope with fine-grained MAC adaptation, novel MAC layer designs became needed. To deal with the varying network demand and application requirements, ML and probabilistic algorithms [17] [51]–[54] are employed at the MAC layer. MAC-specific parameters are tuned, and so the MAC behavior can be adapted according to network conditions (*e.g.*, transmission power, Modulation and Coding Scheme (MCS), and Acknowledgments (ACKs)). For instance, in WSNs, ML algorithms are used to predict when devices are sending data, so parameters such as the duty cycle can be tuned accordingly and hence optimize energy efficiency without performance degradation [55].

Originally, a first modularized architecture for the data-link layer was introduced with the Click modular router in the 2000's [56]. By abstracting the functionalities from commodity routers into programmable modules, its design enables modules to be responsible for implementing each of them separately, independently from one another (*e.g.*, packet processing and interfacing with network devices). However, as the Click modular router was explicitly designed for router functionalities in wired networks, its architecture was not the most appropriate to handle the wireless MAC layer.

Some approaches specifically aimed for *modular* MAC layer designs in wireless networks, but they failed to achieve time-critical requirements [57]. Messerschmitt [58] and Nychis *et al.* [48] identified the minimum set of core MAC functions/modules that must be implemented close to the radio to achieve acceptable performance. However, Universal Software Radio Peripherals (USRPs) and GNU Radio[1] platforms are composed of software components that, at that time, ran entirely on general purpose Central Processing Units (CPUs). Hence, those platforms were not able to meet the performance requirements as the traditional hardware-based implementations, which normally run on Application-Specific Integrated Circuits (ASICs) or Field-Programmable Gate Arrays (FPGAs). Thanks to parallelization, quantization, and other features, ASICs and FPGAs can achieve low runtime latency, but slow design cycle and low versatility [49].

To provide support for programmability on wireless networks, there has been work on dedicated platforms such as SDRs. Typically, SDRs are equipped with FPGA circuits or ASICs, and its architecture distributes the processing of the signals across these processing units, *i.e.*, across FPGAs, ASICs, and CPUs, located both at the SDR device and at the CPU host [59] [60]. For instance, Wireless Open Access Research Platform (WARP) [59] and AirBlue [60] consist of software platforms in which both MAC and PHY are implemented on the FPGA board. Therefore, they provide flexibility and, since FPGAs are integrated circuits and provide high processing capacity, they can achieve the time-critical requirements of the lower MAC layer. On the other hand, since they are designed as hardware-specific platforms, their applicability is very limited and constrained. Besides, since the implementation is specified using Hardware Description Language (HDL), the easy customization of the high-performance blocks is still challenging.

### B. SDR Next Generation

In 1999, based on the approach of dynamic spectrum sharing in the licensed spectrum, a paradigm has emerged in the design of wireless networking systems for the unlicensed spectrum [61]. This concept is popularly referred in the researching community to as Dynamic Spectrum Access (DSA), OSA, or CRN and envisages that the overall spectrum utilization could be effectively improved by opportunistically sharing the underutilized licensed spectrum. In this subsection, we present the concepts of CRNs and the evolution of CRs to the nowadays SDRs.

*1) Cognitive Radio Networks:* In 2002, the *Federal Communications Commission* strengthened that the majority of the licensed spectrum bands were either underutilized or unoccupied [62]. At then, the CR was the key technology to enable the use of the spectrum dynamically [61]. The basic approach of CRNs came from the design of unlicensed networking systems that take advantage of the unused licensed spectrum. Users from both licensed and unlicensed networking systems are classified as *primary* and *secondary*, respectively. A *primary* user is the one who holds the rights (license) to the spectrum while the *secondary* user is the one who is authorized to use the licensed spectrum opportunistically, without causing significant interference to *primary* users [22]. Mitola and Maguire [63] envisaged that *secondary* user networks should sense the licensed spectrum for an idle channel, estimate the channel capacity, calculate the data rate used

[1]https://gnuradio.org/

in its transmission, and then transmit the information to an intended receiver. As CRN introduces those new aspects, new requirements and challenges on MAC protocol designs are expected on fairness, smarter, and better use of the unused and underutilized shared spectrum.

A CR is a radio able to change its transmitter parameters based on interaction with the environment. In summary, the main functions of a CR are:

- **Spectrum Sensing -** Detects unused spectrum and shares it without exorbitant interference from other users;
- **Spectrum Management -** Captures the best available spectrum that meets the requirements from the user;
- **Spectrum Mobility -** Maintains seamless communication during the transition to another spectrum; and
- **Spectrum Sharing -** Provides fair spectrum scheduling among coexisting users within the network.

Most of the existing CRNs employ Half-Duplex (HD) communication systems to exploit the spectrum usage. However, there are two major drawbacks [37]. First, devices cannot simultaneously sense and access the spectrum, which limits transmission capabilities. Second, devices utilize two separate/orthogonal channels for data transmission and reception, *i.e.*, requiring precious spectral resources and increasing latency because two channels have to be sensed. These constraints are addressed by Full-Duplex (FD), where it is possible to sense and access the spectrum simultaneously [64]. In addition, data transmission and reception can be done over the same idle channel and during the same period. Since new communication capabilities were enabled, new requirements arise at the MAC layer as well (*e.g.*, self-interference management and sensing overhead control) [65]. Therefore, enhanced MAC layer mechanisms had to be designed considering the specific requirements from each of the wireless technologies.

Nowadays, the basic control mechanisms of CRNs are not sufficient to deal with extreme and diverging communication needs (*e.g.*, ultra-low latency, ultra-high throughput and low/high data rate, time-critical/non-time critical requirements) [49]. SDRs arose as an alternative to innovate and provide future-proof solutions where its transceiver components, typically implemented on hardware, are instead implemented in software. SDR capabilities such as over-the-air remote reprogramming and the high-level of software re-usability across radio technologies have boosted, even more, the research on programmable, intelligent, and collaborative MAC layers. The following subsection presents the evolution of SDRs and how they relate to the SDN paradigm.

*2) Software-Defined Radios:* SDRs and the SDN paradigm were parallel evolutions carried out from different and isolated research communities [49]. While, at the networking level, SDN decouples the logic from forwarding devices to a controller entity, SDR has decoupled the functionality of its transceivers into means of software at the radio level. These transceiver components are typically hosted on a computer or in embedded systems equipped with programmable hardware such as ASICs or FPGAs. However, the very first radios were developed as non-adaptable *Hardware Radio* systems where their resources and channel access methods were allocated and defined beforehand and thus limiting applicability in dynamic environments. According to the Wireless Innovation Forum (WINNF) [66] (formerly called SDR Forum), the programmability of the SDR technology is characterized and defined by the following tiers:

- **Tier 0 -** The *Hardware Radios*, which consist of non-configurable hardware radios where modifications cannot be performed except through physical intervention;
- **Tier 1 -** The *Software Control Radios*, where limited functions are implemented by means of software (*e.g.*, power levels, and interconnections, but not frequency bands or modulation schemes);
- **Tier 2 -** The SDRs, where a significant part of the radio is software configurable (*e.g.*, software control of a variety of modulation schemes, wide-band or narrow-band operation, security functions, and waveform generation of evolving standards over a broad frequency range);
- **Tier 3 -** The *Ideal Software Radios*, where programmability extends to the entire system, including its front-end (*e.g.*, having analog conversion only at the antenna, speaker, and microphones); and
- **Tier 4 -** Defined for comparison purposes only, the *Ultimate Software Radios* consist of having full programmability and support to a broad range of functions and frequencies at the same time.

Nowadays, Tier 2 SDRs are the most popular ones given that Tier 3 and 4 are still not realistic. As presented by Moy and Palicot [67], the first SDR-related articles were published in 1996, when the SDR Forum was set up. In 2002, the number of SDR-labeled publications increased significantly, meaning that the field became concrete and promising. After 2008, as a consequence of the rise of SDN with the OpenFlow protocol, the number of SDR-related publications (based on Google Scholar) increased, even more, reaching up to 12,000 papers/articles at the end of 2015 [68]. Under the funding of the Defense Advanced Research Projects Agency (DARPA)[2] and from industrial developers such as Xerox, BBN Technologies, IBM, ATT, and Cisco, many wireless-networking researchers are continuously evolving the SDR technology to opportunistically and collaboratively use unused spectrum in underutilized and potentially licensed spectral bands, thus aiming to make it as ubiquitous as the wired Internet [69]. Figure 1 presents the radio technologies where programmable MAC protocols are usually implemented.

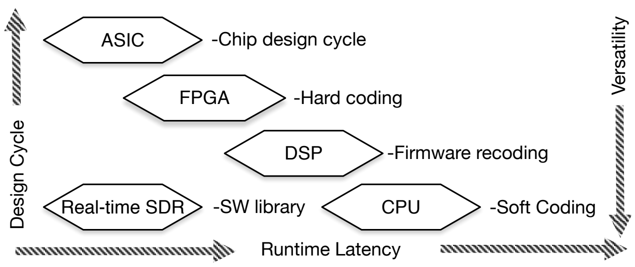


Fig. 1. Real-time SDR along with other radio technologies for implementing wireless MAC protocols, Moerman *et al.* [49].

[2]https://www.darpa.mil/

Along with the radio technologies, Figure 1 illustrates the versatility, design cycle, and the run-time latency that these technologies offer/impose to the MAC protocol development and running processes. ASICs are Integrated Circuits (ICs), normally customized for a particular use rather than intended for general-purpose. Therefore, they are more suitable for the programmability *scope* where changes in the MAC implementation rarely happen. FPGAs also present similar performance compared to the ASICs, but a new bitstream can be uploaded remotely, which gives more flexibility to change the MAC implementation. On the other hand, by implementing a MAC protocol as an application on general-purpose CPUs or real-time SDRs, it is possible to design it quickly and with high versatility. Therefore, general-purpose CPUs and real-time SDRs are more appropriate when application requirements and network conditions change.

## C. The SDN Paradigm

SDN is an emerging paradigm that initially refers to a network architecture where forwarding decisions are decoupled from the network control logic, *i.e.*, the network control logic is removed from forwarding devices—that become simple packet forwarding devices—to a centralized element called controller. The SDN architecture is designed to enable network innovation based on four fundamental principles: (*i*) network *control*, and *forwarding* planes are clearly decoupled; (*ii*) forwarding decisions are flow-based instead of destination-based; (*iii*) the network forwarding logic is abstracted from hardware to a programmable software layer; and (*iv*) an element, called SDN controller, is introduced to coordinate network-wide forwarding decisions [70].

Figure 2 depicts the SDN architecture with the Application Programming Interfaces (APIs) used to establish communication along its planes.

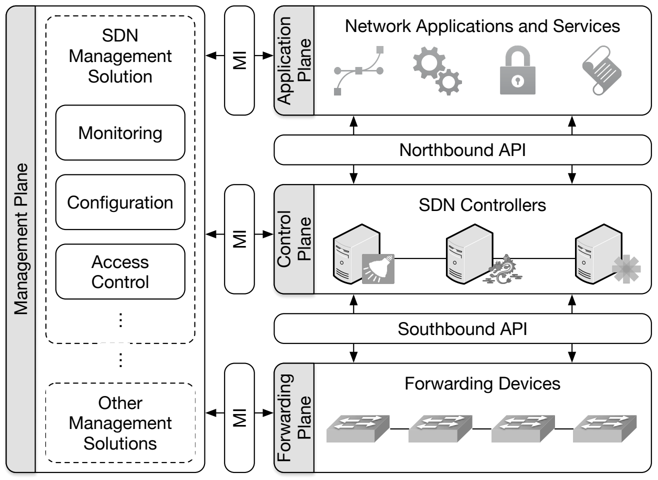


Fig. 2. Overview of SDN Architecture [71].

Overall, SDN introduces an architecture with four planes: *management*, *application*, *control*, and *forwarding* [71]. All of them communicate with each other through interfaces. The *management* plane communicates with the other planes through Management Interfaces (MIs). The *northbound* API establishes bidirectional communication between *application* and *control* planes while the *southbound* API does the same for *control* and *forwarding* planes. Ideally, all these interfaces should be standardized to allow easy replacement of technologies. In practice, however, the OpenFlow protocol is currently the only *de facto* standard *southbound* API. All other interfaces are undergoing discussion and development.

Conceptually, each of the four planes performs a set of specific functions:

- ***Management Plane -*** Contains SDN management solutions responsible for managing components on the other SDN planes (*e.g.*, monitoring device status, configuring and allocating resources, and enforcing access control policies). To perform such management, these solutions communicate with the other planes through MIs;
- ***Application Plane -*** Contains applications that serve several different purposes (*e.g.*, firewall, circuit establisher, and load balancer). One or more SDN controllers grant each of these applications access to a set of resources;
- ***Control Plane -*** Contains a set of SDN controllers (*e.g.*, Floodlight[3], Ryu[4], and OpenDaylight[5]) that comprise the logic to coordinate all forwarding devices. At least one SDN controller needs to execute the requests coming from the *application* plane. Commonly, these controllers already include internal logic to handle network events and make traffic forwarding decisions; and
- ***Forwarding Plane -*** Comprises a set of forwarding devices with transmission capacity and traffic processing resources. In SDN, the notion of keeping forwarding devices simple and leaving the high-level decisions at software layers is fundamental.

SDN is grabbing the attention of academia, standardization bodies (*e.g.*, ONF and IETF), and industry (*e.g.*, Google, Cisco, NEC, and Juniper). Since it allows the easy creation of new abstractions in networking, simplifying management and facilitating network innovation, the SDN paradigm started to be solidly applied also in the wireless environment [71]–[75]. In this context, it is understandable that most of the efforts concentrate on some of the wireless technologies where the firmware is open source and so well researched (*e.g.*, IEEE 802.11 standard with the Open FirmWare for Wi-Fi Networks (OpenFWWF)). Unfortunately, still, there are no concrete approaches for the other wireless technologies because: (*i*) the necessary equipment for experimenting is costly and (*ii*) most solutions are proprietary. For instance, because of the expensive equipment and the lack of open source firmware for LTE, there are only theoretical work or proposals based on simulations so far [76]. However, as SDR enables software reuse with high performance and versatility, it is expected that, in the near future, SDN principles will be employed for the control and management of those technologies as well.

[3]http://www.projectfloodlight.org/floodlight/
[4]https://osrg.github.io/ryu/
[5]https://www.opendaylight.org/

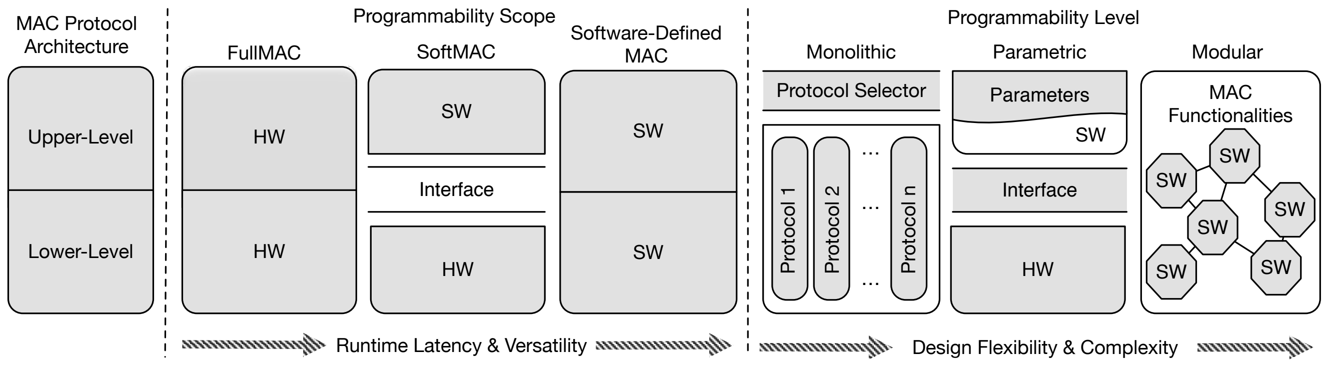


Fig. 3. *Scope* and *level* of programmability according to upper and lower MAC levels.

## III. Terminology Definition

As existing terminologies target the different modes-of-operation and are often used interchangeably, it is not possible to use them to classify MAC protocols according to programmability aspects. Therefore, we define a new terminology to represent the *scope* and the *level* of programmability supported by their architectures. The *programmability scope* dictates how the MAC layer is designed (*e.g.*, hardware-based, hardware/software codesign, or software-based). The *programmability level*, in turn, represents how much of a MAC protocol logic can be changed based on its architecture (*e.g.*, the entire protocol at once, *parametric-level* modifications, or *modular* schemes). In subsection III-A, we describe how MAC protocols are usually classified and, in subsections III-B and III-C, we present our proposed terminology to address programmability of wireless MAC protocols.

### A. Current MAC Protocols Classification

Recent studies [28] [32] have used five main groups to classify wireless MAC protocols: (*i*) *contention-based*, (*ii*) *schedule-based*, (*iii*) *contention-free*, (*iv*) *channel-polling-based*, and (*v*) *hybrid*. These groups represent the modes-of-operation of MAC protocols based on the degree of coordination among the nodes to avoid collisions on data transmissions. In addition, subcategories such as *random access*, *slotted access*, and *frame-based access* are also used, but instead to represent how nodes organize the access to the shared transmission channel [77]. Additionally, authors are analyzing MAC protocols according to performance aspects to achieve Quality of Service (QoS), for instance, by determining how much delay-aware and how reliable MAC protocols are and therefore classify them according to their resulted performance and energy efficiency. However, no categorization considers how programmable are those implementations.

In contrast with the above-described classification, we define a coherent terminology that represents *scope* and *level* of programmability supported by MAC protocol implementations. Figure 3 presents the *scope* and *level* of programmability according to the general MAC protocol architecture.

The *programmability scope* dictates how the MAC layer is designed (*e.g.*, hardware-based, hardware/software codesign, or software-based). In other words, it represents how much from the MAC layer remains within the hardware NIC and how much can be implemented by means of software. For instance, MAC protocols that run on ASICs are normally tightly connected to specific hardware. Since their configuration is generally specified using HDL, both upper and lower MAC layers remain within the NIC. Other implementations leave the lower MAC functionalities on hardware while the upper MAC layer is implemented on personal computers or embedded systems (*e.g.*, SoftMAC [57]) while implementations done on SDRs are entirely accomplished by means of software. On the other hand, the *programmability level* represents how much of a MAC protocol logic can be changed based on its architecture (*e.g.*, switch the entire protocol at once, tune MAC-specific parameters, or change entire modules, where functionalities can be developed from scratch, independently from one another).

### B. Programmability Scope

Regarding *programmability scope*, wireless MAC protocols can be classified into three categories: *fullMAC*, *softMAC*, and *Software-Defined MAC (SDMAC)*.

**FullMAC** – Also known as *one-of-a-kind*, *fullMAC* protocol implementations are designed by sacrificing the layering principle to increase performance and efficiency. In this case, the data-link layer remains tightly connected to PHY and thus becomes one single *monolithic* building block. As these implementations are vendor-specific, technology and environment aspects need to be considered in their design. Once the protocol is coded, compiled, and deployed on specific hardware, external changes on its logic are no longer possible. In this case, the Medium Access Control Sublayer Management Entity (MLME), that manages the PHY MAC state machines, is implemented at the *hardware* side instead of at the *kernel* side. Therefore, *fullMAC* protocol implementations are widely used for standard protocols in which implementations are stable, optimized for specific hardware (*e.g.*, standardized Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol implemented on ASICs), and designed for scenarios where requirements do not change that often (*e.g.*, habitat monitoring and structural health monitoring). On the other hand, to cope with network behavior changes, these implementations usually employ ML algorithms and heuristics to adapt the MAC according to network changes [17], [51]–[54]. In this manner, even without external intervention, a certain level of adaptability for the MAC layer can be provided.

**SoftMAC** – A *softMAC* protocol is a hardware/software codesign implementation where the time-sensitive functions of the lower MAC layer are implemented at the *hardware* side, while the upper MAC layer stays at the *kernel* side [57]. The term *hybrid* is also used to define these *softMAC* implementations, in addition to when its logic combines features from protocols with different modes-of-operation [31], for instance, in case of a single protocol comprising features from a *contention-based* protocol combined with others from a *schedule-based* [5], [78], [79]. Therefore, in this survey, for the sake of clarity, we use the term *softMAC* for protocol architectures that have a hardware/software codesign. In this context, *softMAC* protocols enable non-time-critical functionalities, *i.e.*, functionalities from the upper MAC layer to be implemented in software and hence they can be easily modified at runtime (*e.g.*, MAC protocols implemented on FPGAs, where a new bitstream can be uploaded remotely). In this manner, by allowing to perform remote and runtime parameters adaptation, *softMAC* implementations extend the *fullMAC* protocol design.

**SDMAC** – Given the fact that the SDR platform is implemented entirely by means of software, many researchers have been using it to propose *modular* schemes for prototyping wireless MAC protocols [48] [58]. Radio components that were typically implemented in hardware (*e.g.*, mixers, filters, amplifiers, modulators/demodulators, and detectors), are now implemented by means of software. Besides, these implementations can run on top of general-purpose hardware such as personal computers and embedded systems. For this survey, we employ the concept of *SDMAC* for those implementations where the lower and upper MAC layers are located at the *user space*, providing the maximum level of flexibility. We acknowledge that these implementations must run on an SDR platform with respect to the other categories. Since they enable rapid prototyping and a high-level of customization, *SDMAC* protocols have been the best option to experiment with new algorithms without vendor-specific knowledge. Recent advances have made it possible to control SDR in real-time, making them increasingly more suitable for the deployment of MAC protocols in production networks, where substantial flexibility is required [49]. Thus, the usage of the term *software-defined* for MAC protocol is straightforward, referring to implementations that are fully-programmable, *i.e.*, both upper and lower MAC layers are implemented by means of software.

Figure 4 illustrates where the different implementations are usually placed and the relationship among the hardware NIC, *user*, and *kernel* space. While *fullMAC* and *softMAC* designs have to rely on APIs and libraries to interact with the lower MAC implementation, *SDMAC* designs provide, in addition to the upper layer features, control over the medium access mechanisms at the lower MAC layer as well.

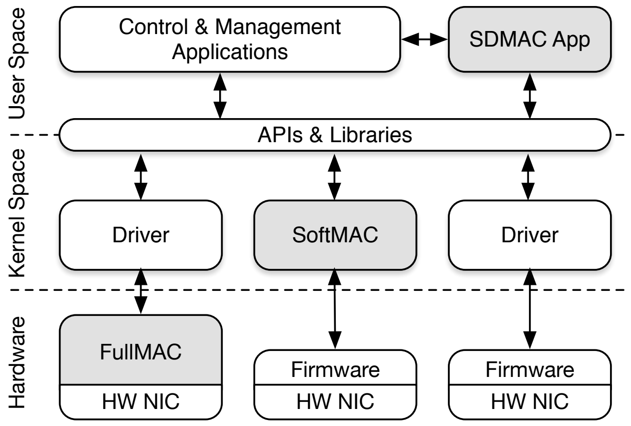


Fig. 4. *Programmability scope* along with a generic Linux kernel architecture.

## C. Programmability Level

According to their architectures, MAC protocols fit into three categories that define their *level* of programmability. We define them as *monolithic*, *parametric*, and *modular*.

**Monolithic** – We define that a MAC protocol implementation has a *monolithic-level* of programmability when its architecture only supports to switch the entire implementation at once [15]–[17]. In a nutshell, these implementations are just non-interactive *fullMAC* implementations that are deployed in advance and therefore can be further switched. The decision to switch among these set of predefined implementations can be triggered locally or through external entities, *i.e.*, at the forwarding device itself or through external controllers. Usually, these decisions are taken mostly to change between protocols with different modes-of-operation (*e.g.*, *scheduled-based* to a *contention-based* implementation and *vice-versa*) and they are triggered based on performance degradation thresholds and predicted traffic patterns mostly. Since these predefined MAC protocols are usually stable and well optimized hardware-specific implementations (implemented on ASICs or FPGAs), most of them have the *fullMAC programmability scope*.

**Parametric** – Usually, MAC protocol implementations have to suffer modifications according to network conditions and application demands. To cope with these dynamic conditions and demands, some architectures allow performing parameter tuning through interfaces, sometimes called *software overlays* or *softMAC* implementations [57] [80]–[83]. Most of them interact with components and modules that belong to the upper MAC layer, which is usually software-based, leaving time-critical implementations within the NIC. Therefore, we consider a *parametric-level* of programmability when external entities (*e.g.*, network administrators or management applications) can perform parameter tuning.

**Modular** – In some cases, tuning parameters at the MAC layer may not offer enough flexibility to cope with dynamic and unusual scenarios. MAC layer should be more programmable, and for this reason, MAC protocol implementations started to be abstracted into software modules where each of them is responsible for executing a specific MAC functionality independently from the others [58] [48]. These *modular* approaches often make use of SDRs to prototype and evaluate new algorithms at low cost, and usually on a standard general-purpose computer platform. Therefore, a MAC protocol is considered to have a *modular-level* of programmability when its architecture decomposes its logic into independent software modules, *i.e.*, modules where it is possible to be rearranged or even redesigned from scratch. Thus, as a consequence of the high flexibility offered by *modular* designs, the architecture presents the highest *level* of programmability for wireless MAC protocols.

As can be seen in Figure 3, *monolithic* architectures support changing *fullMAC* implementations so they concentrate their effort on how and when these implementations may switch. *Parametric* proposals already offer some access to upper MAC functionalities, leaving the medium access mechanisms of the lower MAC within the hardware NIC. Last, the *modular* approaches are the most sophisticated implementations where all functionalities are abstracted into software modules and, hence, can be easily modified.

Frequently, the chosen hardware indicates the *scope* and *level* of programmability addressed at the MAC layer. However, other aspects influence it. For instance, the programmability *scope* also depends on the software that accesses and performs the changes at the MAC layer (*e.g.*, firmware, drivers, and software overlays), and this software might not be available as open source or even not fully supported by the network hardware vendors. Notwithstanding the programmability *level*, architectures with a coarse-grained *level* of programmability are feasible to be implemented on hardware that supports more fine-grained *level* of programmability, such as SDRs. Although the majority MAC layer architectures that are developed on SDRs a fine-grained *level* of programmability (*e.g.*, *modular* designs), coarse-grained architectures can be addressed as well (*e.g.*, *monolithic* and *parametric-level* architectures implemented on SDRs). The granularity of the changes that specific architecture/framework performs at the MAC layer is what characterizes the *level* of programmability. Therefore, the hardware does not necessarily imply how programmability is addressed at the MAC layer.

## IV. Monolithic-level of Programmability

The first *level* of programmability that we address is the *monolithic level*, where architectures only enable to switch among *fullMAC* implementations at once. Given that neither *modular* nor *parameters* tuning are supported, *monolithic-level* architectures represent the lowest *programmability level* of our terminology. It is natural that the research on programmable MAC evolved more on wireless technologies where the environment is more dynamic and heterogeneous (*e.g.*, IEEE 802.11 and CRN). When resource capabilities are scarce and network behavior is more stable and predictable, a single adaptable MAC is enough to cope with the demand (*e.g.*, IEEE 802.15.4). In addition, the availability of open source efforts (*e.g.*, drivers and firmware) that enable hardware NIC interaction fostered the research on *monolithic-level* architectures on certain technologies more than on others. Therefore, in the following sections, it is expected that the presence of *monolithic*, *parametric*, and *modular* architectures are not balanced addressed among wireless technologies.

### A. Principle & Method

One of the most relevant efforts with the *monolithic-level* of programmability is the Meta-MAC protocol [15]. In the 2000's, Farago *et al.* have proposed an architecture and a method to automatically combine any set of existing MAC protocols as components into a single higher layer, just above the existing MAC, as illustrated in Figure 5. Within this layer, individual transmission decisions are aggregated into a final decision, requiring no centralized control nor message exchanging. Thus, based on this final decision, the Meta-MAC protocol can automatically choose the component that fits best under the actual network conditions.

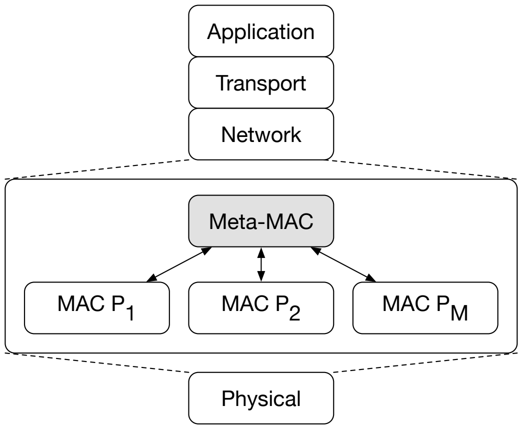


Fig. 5. The Meta-MAC protocol architecture in a simplified protocol stack, Farago *et al.* [15].

To ensure this, during every defined time slot the Meta-MAC algorithm generates local decisions for each of its components (*e.g.*, binary or decisions based on probabilities), specifying whether a certain protocol $P_i$ would transmit during the next time slot or not. After each time slot, such component decisions are combined into a final decision and a *feedback* score is calculated. Such *feedback* is used to determine whether the previous decision was correct or not. For instance, if the transmission was successful or the channel was occupied while it was decided to not transmit on that channel, then the decision was right. On the other hand, if there was a collision or if there was a packet in the queue and the channel remained idle during the time slot, then the decision was wrong because the time slot was wasted. Therefore, based on the *feedback*, weights that represent whether protocols should transmit at a certain time slot are updated, and the reconfiguration cycle ends without the need for external intervention.

Meta-MAC presented one of the pioneering ideas on MAC programmability, and implementations are still being proposed based on its concept. By changing from one protocol implementation to another, Meta-MAC enables the best protocol with the proper mode-of-operation to be chosen. However, Meta-MAC also has limitations. The most important one is the fact that its architecture relies only on local knowledge, so issues may arise when protocols with different modes-of-operation are triggered, for instance, when a node is using a *scheduled-based* protocol in the same range of another node that is using a *contention-based* protocol. In this case, given that *scheduled-based* protocols do not sense the medium before transmitting and *contention-based* are not aware of any node's schedule by default, collisions may occur and the overall performance may be compromised.

## B. IEEE 802.15.4 Networks

There is plenty of work on MAC adaptability in IEEE 802.15.4 networks [26] [32] [33] [50]. Although most of them comprise a single adaptable MAC layer implementation, there is some work where different MAC layer behaviors are implemented beforehand so that they can be further switched to accommodate real-time applications. In WSN-based smart grid applications, three types of MAC layer schemes are being addressed: *contention-based*, *reservation-based*, and *hybrid* [50]. Several proposals [84]–[87] are employing *hybrid* MAC protocol implementations to overcome the shortcomings of using a single MAC scheme. For instance, Hsieh *et al.* [87] proposed a cross-layer design where the two different MAC modes-of-operation can exchange performance information and decide which behavior is more appropriate according to the demand. In this case, to minimize both energy consumption and packet latency simultaneously, the authors proposed a MAC protocol capable of switching between a *contention-based* and a *schedule-based* behavior. First, a Time-Division Multiple Access (TDMA) [88] protocol is set as the initial MAC behavior. Then, when a node overhears routing request messages, a Carrier-Sense Multiple Access (CSMA) [89] mode protocol is activated to shorten the transmission latency of the subsequent packets.

Besides, other proposals such as T-MAC [4] and S-MAC [6] also adopted the combination of two different protocol implementations into their MAC adaptation mechanisms. Both of them focusing on switching between CSMA and TDMA protocol implementations. De Mil *et al.* have observed that real-life performance evaluations of WSN MAC protocols have shown that, depending on the traffic pattern and the packet interval, a given combination of MAC and routing is better under certain circumstances and can outperform other combinations. Therefore, the authors have proposed PluralisMAC, a generic multi-MAC framework able to switch among different MAC strategies [90]. To provide such flexibility, PluralisMAC is composed of two other frameworks within its architecture: Multi-MAC and Neighbour Management Framework (NMF). Multi-MAC is responsible for enabling access the medium access logic (*e.g.*, when to listen to the medium, how packets are sent, and the data transfer model) and NMF is in charge of performing monitoring and filtering (*e.g.*, gather neighbor statistics such average Received Signal Strength Indication (RSSI) and inter-packet delay). Thereupon, PluralisMAC combines the information gathered with the MAC switching mechanism to be able to take better decisions and cope with application requirements (*e.g.*, maximum hop-by-hop latency). In this manner, PluralisMAC better handles wireless communications in low-power systems with heterogeneous devices.

## C. IEEE 802.11 Networks

In 2005, MultiMAC [91] extended the Meta-MAC concept and introduced an actual implementation of it. MultiMAC built its framework upon the functionality of the SoftMAC Linux kernel implementation [57]. SoftMAC provides a driver that allows control over the MAC layer while still allowing the use of the waveforms defined by the underlying IEEE 802.11b/g/a physical layers. By acting as a mediating driver, between the physical device and the network stack in the *kernel space* (Figure 4), SoftMAC provides precise control over the content and timing of wireless transmission and reception (*i.e.*, *softMAC programmability scope*). When MultiMAC gets a packet from the Linux kernel, it must decide which MAC is best suited to transmit the particular packet. Then, the chosen MAC must encode the packet and specify the transmission timing constraints.

Although SoftMAC enables to perform changes at the MAC layer itself, its design only supports a single MAC to be used at a time. Therefore, MultiMAC extends SoftMAC to allow multiple MAC layers to coexist concurrently in the network stack, with minimal switching impact. Its functionality introduces mechanisms and policies to choose the MAC layer that suits best for particular network conditions, instead of changing a particular MAC implementation. In addition, to optimize a reliable transmission, MultiMAC selects protocols according to changes observed in the MAC layer and based on user-level requests. MultiMAC exports an interface to the user-level where the rules for deciding which MAC to use for transmitting a particular packet are viewable and editable, thus incorporating the user into the decision-making process. Another characteristic that distinguishes MultiMAC from Meta-MAC is the fact that, to determine whether a particular protocol is feasible, MultiMAC makes its decisions by estimating the channel occupancy instead of choosing a specific MAC on a packet-by-packet basis with no interactions between successive transmissions.

## D. CRN

Based on the *monolithic-level* architecture, Huang *et al.* [16] extended the Meta-MAC concept and proposed the Adaptive MAC (AMAC) protocol for CRN. Unlike Meta-MAC, which bases its MAC protocol changes on local decisions only, AMAC supports global knowledge where each node negotiates with one another through a *voting scheme*. Hence, nodes can agree on a common and suitable MAC protocol that performs better for the overall wireless communication. Since AMAC switches between MAC protocols with different modes-of-operation, the *voting scheme* is necessary to deal with the MAC protocol incompatibility. Besides, AMAC adopted a strategy based on performance degradation thresholds and predicted future traffic patterns, instead of relying only on the probability that a protocol would transmit during a specific time slot. To set up network adaptation functions (*e.g.*, bootstrap, discovery, and *voting scheme*) and PHY parameters (*e.g.*, operating channel, power, and modulation type), AMAC applies a Global Control Plane (GCP) cross-layer network management overlay [92]. GCP sits in the *kernel* space, so it enables a *softMAC programmability scope*. Although GCP enables *parametric-level* of programmability, AMAC only performs parameter-level adaptations at PHY level. MAC layer adaptations are made by switching the entire protocol at once. In the GCP-based architecture, nodes have a dedicated control interface (along with data interface) to set up these functions. Thus, offering separation between *control* and *data* planes.

Recently, Qiao *et al.* [17] proposed a MAC protocol selection scheme based on an ML algorithm. With their approach, network nodes can classify network parameters in real-time with a classifier (*e.g.*, data package size, transmitting interval, and transmitting rate). Then, based on the classification result, the suitable MAC protocol is selected. In this case, the authors have chosen between two *fullMAC* implementations with different modes-of-operation: a competitive Distributed Coordination Function (DCF) protocol and a non-competitive TDMA protocol. Despite the limited MAC protocol options available, that proposal can make intelligent and efficient MAC protocol selections.

### E. Lessons Learned

*Monolithic-level* architectures present an initial *level* of programmability for the MAC layer. By analyzing the aforementioned related work, we list important lessons learned regarding such architectures:

*1) Monolithic-level architectures provide simplicity and standard compliance:* Usually, *monolithic-level* approaches are ideal where the protocol's logic should only vary according to stable and well-adopted implementations. In general, *monolithic level* architectures provide the benefit of *simplicity and standard compliance*. Hence, they represent an initial *level* of MAC programmability. In some cases, according to network density, MAC protocols have to be switched to improve wireless connectivity. For instance, CSMA [89] is a MAC protocol where nodes verify the absence of other traffic before transmitting, while TDMA [88] is a protocol where several users share the same frequency channel by dividing the signal into different time slots. Both of these implementations are standardized and well-adopted MAC protocols. Thus, in addition to the fact that their expected behavior is more predictable and reliable, they have different medium access mechanisms that fit better depending on the network density. In addition, the decision-making process becomes relatively simple compared with parameters tuning or the design of an entire MAC protocol from scratch.

*2) Monolithic-level architectures offer a coarse-grained level of programmability:* It is natural that the research on programmable MAC evolved more on the wireless technologies where the network is more dynamic and heterogeneous (*e.g.*, IEEE 802.11 and CRN). Given the availability of open source efforts (*e.g.*, drivers and firmware), which enable hardware NIC interaction, and the need for more network programmability motivated and fostered the research on *monolithic-level* architectures. As these architectures usually comprise several predefined *fullMAC* implementations, there are significant limitations to be considered. The limited memory capacity to handle sets of MAC implementations within the NIC and the lack of standardized interfaces for monitoring and interacting with the MAC layer determine that *monolithic-level* architectures have a *coarse-grained level of programmability*. On the one hand, given its autonomously and simplicity, *monolithic-level* architectures can still be the best option [93]. On the other hand, the degree of flexibility provided may not be sufficient to cope with future dynamic and heterogeneous wireless environments.

## V. Parametric-level of Programmability

Given the limitations presented by *monolithic-level* approaches, improvements regarding programmability of wireless MAC protocols started being proposed. Inspired by the success of *overlay networks*, several proposals were introduced (as *enablers*) to allow interacting with the MAC layer. Instead of changing from sets of pre-defined MAC protocol implementations, which are usually allocated within the hardware NIC, some parameters were enabled to be tuned (*e.g.*, transmission power, channel frequency, and MCS). Therefore, different approaches [52] [81]–[83] have been introduced to tune a wide range of MAC-specific parameters and thus control the MAC layer according to network behavior. In this section, we discuss the proposals in which their architectures match the *parametric-level* of MAC programmability.

### A. 3rd Generation Partnership Project (3GPP) LTE

LTE is a wireless technology designed for high-speed communications in cellular networks specified by 3GPP in its Release 8. LTE is responsible for opening the 4th Generation of Broadband Cellular Networks (4G) [94]. Although LTE was initially designed for long-range deployments, the standard has been adapted to offer additional MAC operation modes such as Device-to-Device (D2D) for partially or no coverage scenarios and Long-Term Evolution-Unlicensed (LTE-U) in coexistence with other technologies as IEEE 802.11 in the unlicensed 5 GHz band. Due to the nature of the commercialization and licensing of this technology, so far most work is theoretical or based on simulations [76]. Given the rise of SDR platforms, it is now possible to implement the full LTE stack using software, allowing researchers to experiment with their own MAC proposals in a real environment with full functionality.

*1) Traditional LTE deployment:* The basic scheme of an LTE deployment consists of a Base Station (BS), denominated as evolved Node B (eNB), that is in charge of managing and provisioning radio resources for a set of User Equipment (UE) attached. The MAC layer of the eNB stack executes a process known as radio resource scheduler that is often a complex computational task. The research community has focused on proposing new MAC resource scheduling algorithms, modifications on the existing schedulers, and new schemes to address different parameters of the radio link (*e.g.*, aggregated throughput, Signal-to-Interference-plus-Noise Ratio (SINR), and inter-BS interference) [76] [95]–[97].

In this context, Zhou *et al.* [76] proposed a global scheduling algorithm where resource scheduler is computed in a central entity, extending the vision of the network with the global information provided by the base stations. Rost [96] proposed a collaborative scheduler among base stations to minimize interference by employing asymmetric resource allocation in Time-Division Duplex (TDD) schemes. Prasad *et al.* [97] presented mechanisms to maximize the load balance through the proportional fairness utility in HetNets topology deployments, while Ayhan *et al.* [95] proposed modifications in proportional fairness scheduler to enhance the throughput in the downlink (DL), replacing the scheduler utility function.

Since many efforts have been made on the MAC scheduler for LTE, in 2010, the FemtoForum[6] (today is known as Small Cell Forum) developed an LTE MAC scheduler API called Femto Application Platform Interface (FAPI) to specify a set of interfaces to interact with the eNB MAC scheduler. Eurocom has extended FAPI, and it is currently on version 2.0 to support new features such as Carrier Aggregation and improved Channel Quality Indicator (CQI) data structures [98]. In parallel, the Small Cell Forum has released a new version of its FAPI to include a Network API (nAPI) [99]. nAPI is oriented to support the LTE functional split in Virtual Network Functions (VNFs), *i.e.*, specifying where the new interfaces must be implemented and the information to be exchanged between VNFs.

*2) LTE in the unlicensed band - the challenge of co-existence:* 3GPP included, for the first time in Release 13, the work developed by the LTE-U Forum[7] to support LTE in unlicensed bands, in particular in 5 GHz shared with other technologies, such as IEEE 802.11. Several proposals [100]–[103] addressed the co-existence with other wireless technologies using different proposals and changes on the MAC protocol, including new frames or medium access mechanisms. Han *et al.* [100] and Zhang *et al.* [101] proposed a set of new mechanisms for LTE MAC layer following a Listen Before Talk (LBT) approach, where LTE equipment sense the medium before transmitting, to avoid collisions with IEEE 802.11 stations. Both works propose different parameters to allow the co-existence in the same band, adjusting the periods when LTE devices can transmit. However, Khairy *et al.* [102] proposed a similar methodology for LTE co-existence with IEEE 802.11 but including some new frames at the MAC layer that reserve the medium for a certain number of time slots. For 5G network deployments with heterogeneous access technologies, Salem *et al.* [103] presented a new MAC protocol for the distributed coordination of multiple LTE Licensed-Assisted Access (LTE-LAA) base stations. The authors added new MAC frames to synchronize technologies when (*i*) a UE attaches the network, providing the channel and grouping of users, and when (*ii*) developing a new mechanism, called Contend-to-Coordinate (C2C). In this manner, their protocol selects which MAC mechanism to use for inter-LTE-LAA domain coordination.

[6]http://www.eurecom.fr/ kaltenbe/fapi-2.0/

[7]https://www.lteuforum.org/

## B. IEEE 802.15.4 Networks

Self-adapting [13] [45] [104] [105], cooperative [106] and even single MAC protocols [14] receive much attention when focused on specific networks and scenarios. Cognitive MAC protocols have emerged as a promising solution to address this issue by opportunistically allowing *secondary* users to utilize unused licensed bands and, hence, enable spectrum co-existence of multiple users. This is only possible given to the OSA capability of the CR technology that is enabled on sensor nodes, the so-called Cognitive Radio Sensor Networks (CRSNs) [107] [108]. Such solutions are designed to meet specific requirements with the ability of fast and automated reconfigurations at the MAC layer, *i.e.*, requiring no human involvement. Based on the history of previous successful communications and cooperation, these solutions can adapt parameters and therefore change the MAC behavior.

However, as a result of the dynamic nature of the spectrum and the constant evolution of radio capabilities, Akan [107] *et al.* have identified that, in the literature, various of these MAC layer solutions are impractical for CRSNs. For instance, the requirement of multiple transceivers, network-wide synchronization dependency, and the poor performance under bursty traffic in densely deployed networks have to be addressed. Therefore, new approaches for CRSNs are needed. The authors have identified that specific MAC features have to be developed for such networks. First, novel MAC layer techniques have to be developed to make full use of the multiple alternative channels available. Second, devices must operate with minimum control overhead and with no additional hardware requirements (*e.g.*, extra transceiver, Global Positioning System (GPS) for synchronization). Last, duty cycle mechanisms have to be improved to consider neighbor discovery, spectrum sensing, and allocation jointly. Therefore, given these open issues and the constant need for the development of new MAC layer features, it is expected that MAC programmability receives more and more attention.

## C. IEEE 802.11 Networks

One of the first attempts to provide more control of MAC layer functionalities was the MadWifi WLAN driver [109], proposed by the no longer active MadWifi project. That driver was designed by overwriting the stock driver of the Atheros wireless chipsets and relied on the Hardware Abstraction Layer (HAL) kernel module to access the MAC layer. HAL is a piece of software that has direct access to Atheros hardware; hence, it has full access to the system's internals. However, since HAL is proprietary, closed-source, and distributed in binary form only, many MAC features remained unavailable (*i.e.*, hindering *softMAC* programmability *scope*). Legal frequencies and transmission power that the radio can use were not available to be modified. According to a given region (*e.g.*, Japan, UK, and the USA), HAL internally changes its behavior to cope with the regulatory domain and country code. Thus, because of this limitation and other issues regarding HAL source code, the MadWifi driver was never included into the Linux kernel [109].

Because of the ever-increasing demand posed by wireless network applications, unfairness and excessive collisions arose in heterogeneous and multi-hop environments. Besides, the widespread availability of IEEE 802.11-based hardware and the lack of control over IEEE 802.11 MAC encouraged the research community to search for more programmability over the MAC layer.

Figure 6 presents the most relevant *parametric-level* IEEE 802.11 approaches and how they interact with each other based on the generic Linux kernel architecture.

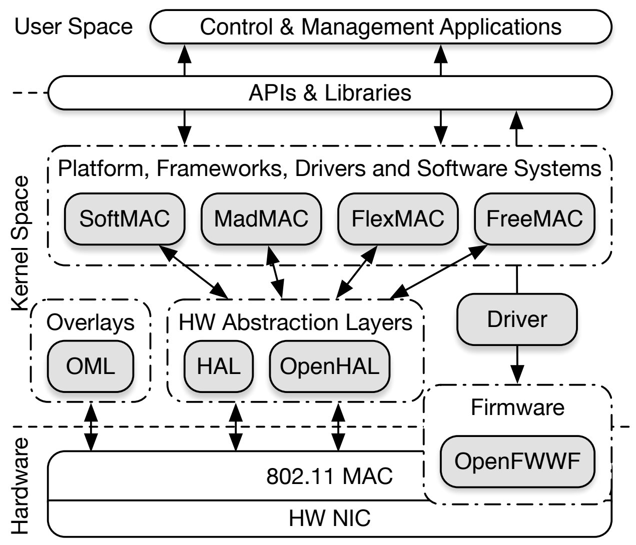


Fig. 6. *Parametric-level* IEEE 802.11 approaches along with the generic Linux kernel architecture.

In summary, these proposals consist of *softMAC* implementations in which some of the MAC features are implemented within the *kernel-space*, while others remain tightly connected to its hardware NIC. Therefore, they are classified as having a *softMAC* programmability *scope*. To present the *parametric-level* approaches more didactically, we describe them based on two main groups. First, we explain the *enabling firmware and software overlay layers*, in which it is allowed interaction within the hardware NIC. Then, we present the *frameworks, platforms, drivers, and software systems* that comprise the actual logic to perform *parametric-level* changes at the MAC layer.

*1) Enabling firmware and software overlay layers:* Rao and Soica [80] have proposed an *overlay layer* to provide more control over the hardware, without replacing the existing IEEE 802.11 MAC layer. The proposed Overlay MAC Layer (OML) sits on top of the IEEE 802.11 MAC, *i.e.*, at the *kernel* space, and protocols can be designed as an overlay on the existing stock. OML enabled control of packet scheduling, better integration with routing and application requirements, and enabled research on new protocols using existing IEEE 802.11 testbeds. Besides the fact that every software overlay has its implementation limited by the underlying MAC layer, there are two major drawbacks need to be considered before using the OML. First, OML does not work properly in the presence of interference from native IEEE 802.11 clients. Second, OML introduces additional control overhead to handle mobility.

Some research has also led the MadWifi team to provide open source access to HAL. Even with the limitations and implementation issues presented by HAL for the Atheros chipsets, researchers still applied reverse engineering on that implementation to design an open source version of it, called Open Hardware Abstraction Layer (OpenHAL) [110]. Alongside is the OpenFWWF [111], proposed by the UniBS telecommunication networks group[8]. OpenFWWF provides an open source effort as well as OpenHAL but as firmware for Broadcom/AirForce chipset based devices. However, some major limitations remain. First, some MAC features such as the Request to Send/Clear to Send (RTS/CTS) and hardware cryptography acceleration are not implemented. Second, to apply any modification, it requires knowledge of the Broadcom/AirForce platforms (*e.g.*, hardware registers and the operating mechanisms that rule data frames along the transmission and reception paths). Third, the *firmware* only supports a limited set of wireless devices, *i.e.*, compatible with the Broadcom/AirForce chipset based devices.

*2) Frameworks, platforms, drivers, and software systems:* There have been several proposals based on commodity hardware, similar to the MadWifi driver. The first to mention is SoftMAC [57]. SoftMAC uses the Atheros chipset along with a modified version of the MadWifi driver to bring more flexibility to MAC development. With their version of the driver, it is possible to implement different protocols and change some MAC-specific parameters (*i.e.*, *parametric-level* and *softMAC programmability scope*). For instance, enabling to modify the format of the packet, configure contention windows, slot time, and transmission power. However, it does not support precise time scheduling, as this functionality is implemented inside the NIC. Therefore, the authors did not focus on analyzing performance parameters such as throughput and latency.

SoftMAC was used as an inspiration to provide more programmability to the MAC layer. The MadMAC framework [81] was proposed as a kernel-mode driver to provide MAC reconfigurability on IEEE 802.11 commodity hardware. Built on top of MadWifi, MadMAC enables to send packets at a controllable time and with specific frame formats. Thus, it improves the precise timing scheduling presented by SoftMAC. However, since MadMAC uses relies on HAL to take control of the radio hardware, its implementation does not offer full control over the MAC layer.

Sharma and Belding have proposed FreeMAC [82], a multichannel MAC development framework on top of commodity IEEE 802.11 hardware. Making use of methodologies from SoftMAC and MadMAC, FreeMAC supports control over some of the TDMA-like radio parameters such as flexible frame formats, disable per-packet ACKs and virtual carrier sense, and channel switching. However, unlike MadMAC, FreeMAC makes use of the MadWifi driver with OpenHAL for the Atheros chipset-based commodity IEEE 802.11 devices.

[8]http://netweb.ing.unibs.it/ ntw/

TABLE III
PARAMETRIC-LEVEL PROPOSALS AND THEIR SUPPORTED FEATURES.

| **Name** | **Year** | **Type** | **Supported Standards** | **Hardware** | **Supported Fine-tunning Parameters** |
|---|---|---|---|---|---|
| HAL [109] | 2005 | Abstraction Layer | IEEE 802.11 | Atheros chipsets | Full access to the system internals |
| SoftMAC [57] | 2005 | Software Overlay | IEEE 802.11/a/b/g | Atheros chipsets | Format of transmitted packets<br>Control over content and timing<br>of transmission and reception |
| OML [80] | 2005 | Software Overlay | IEEE 802.11a/b/g | Netgear WAG511 tri-mode<br>PCMCIA wireless Ethernet adapter | Access control<br>Scheduling |
| OpenHAL [110] | 2006 | Abstraction Layer | IEEE 802.11 | Atheros chipsets | Full access to the system internals |
| MadMAC [81] | 2006 | Platform | IEEE 802.11 | Atheros chipsets | Control over slot structure<br>control over content and timing<br>of transmission and reception |
| FreeMAC [82] | 2008 | Framework | IEEE 802.11 | Atheros chipsets | Flexible frame format<br>Disable/enable per-packet ACK<br>Disable/enable virtual carrier-sense<br>Disable/control random back-off intervals<br>Channel Switching<br>Power and Rate<br>Acknowledgements<br>Virtual Carrier-Sense<br>Queue size |
| FlexMAC [83] | 2008 | Framework | IEEE 802.11 | Atheros Chipsets | Back-off mechanism<br>Retransmissions<br>Packet timing |
| OpenFWWF [111] | 2009 | Firmware | IEEE 802.11 | Broadcom/AirForce chipsets | Full access to the system internals |

FlexMAC [83], otherwise, addresses CSMA-based protocols for IEEE 802.11 commodity hardware instead of TDMA-based protocols. The approach is similar to the one presented with FreeMAC, *i.e.*, having more control over the MAC layer using Atheros chipsets and the MadWifi driver, but with FlexMAC the wireless card is put in promiscuous mode so that all packets are mirrored to the host. In this manner, the MAC functionality is implemented in the *kernel space*, and some of the functions are enabled for customization such as back-off, retransmission schemes, and timing. Notwithstanding, FreeMAC and FlexMAC are not generic enough to compose protocols whose hardware is not manufactured by the Atheros Corporation and, hence, reducing their applicability.

Table III presents details of the proposals mentioned above with their supported *parametric-level* of programmability. As time went on, more and more control over MAC-specific parameters is being enabled as well as the emerging open source software efforts. As can be seen in the table, many of them enable interaction over the IEEE 802.11 hardware NIC, especially over Atheros chipsets.

## D. Lessons Learned

*Parametric-level* architectures enable more control over the MAC layer by exposing/interacting with some MAC-specific parameters. As these *parametric-level* approaches normally address only a subset of parameters from the upper MAC layer and impose some overhead as well, we identified some valuable lessons learned regarding such architectures:

*1) Parametric-level architectures do not offer control over medium access mechanisms:* The main goal of *parametric-level* architectures is to support MAC-layer changes without significant overhead. For instance, leaving the time-critical functionalities within the NIC while the high-level control is written in software, *i.e.*, time-critical functionalities within the *hardware* NIC while the high-level control is done in the *user* and *kernel* spaces. In the literature, there are studies [112] [113] where a two leveled approach is addressed. Components regarding the underlying hardware should provide interfaces to enable interaction with the specific hardware while the high-level components remain independent from the PHY layer. Besides the overhead imposed by an extra layer, MAC *overlay* implementations also have the limitation that they cannot access some of the MAC functionalities. These are time-sensitive functionalities related to medium access mechanisms, which are implemented within the NIC (*e.g.*, frequency bands and modulation schemes). Therefore, even though *parametric-level* architectures support some parameters tuning, more *control over medium access mechanisms* should be provided.

*2) Parametric-level architectures impose extra layers and hence imply extra overhead:* Much has been discussed about the applicability of these *softMAC* protocol implementations. Because of the widespread availability of IEEE 802.11-based hardware, the presence of enabling open source *drivers* and *firmware*, and the increasing networking demand and its dynamic nature, most research on MAC programmability were done centered on IEEE 802.11 networks. The fact that such programmability *typically adds an extra layer and, hence, imposes extra overhead*, and because that some resource-constrained devices were not designed to support it (*e.g.*, sensor devices), it is expected that *parametric-level* architectures are not equally addressed among the wireless technologies. However, since SDRs are allowing fully-programmable MAC layers with high performance, many efforts are now addressing *SDMAC* protocols instead. By shifting all MAC functionalities from the *hardware* to the *user space*, it is possible to enable full control over the medium access mechanism and, thus, to provide more freedom for innovation.

## VI. Modular-level of Programmability

Expressive advances regarding MAC programmability already started with the Click modular router [56] in the 2000's. The Click modular router presented an architecture that supports the design of flexible and configurable routers in a *modular* fashion. These modules (also called as *elements*) represent routing functionalities (*e.g.*, packet classification, queuing, and scheduling) and these can be bonded together to compose different routing configurations. However, as its architecture is designed for routing functionalities in wired networks, specific tools for the design and reconfiguration of the wireless MAC layer are needed [114]. Nevertheless, Click has one of the first architectures to introduce decomposition of such complex system into functional modules, fostering the research towards *modular* designs for wireless MAC protocols.

### A. Modular MAC Designs and Architectures

One of the first *modular* designs for the MAC layer is presented by Braden *et al.* [115] with a Role-Based Architecture (RBA). RBA is an alternative to the layered approach where *roles* are defined as modules, and these are sent within packets to describe specific MAC functions relevant to forwarding and processing packets. The authors argue that it is difficult to evolve network protocols because of two significant limitations imposed by the *software overlays*. First, introducing an extra layer in the network stack implies extra overhead. Second, the relatively coarse granularity of protocol functionality restricts its programmability. In this case, there is no need for a layered approach that interacts with the MAC layer. On the other hand, RBA requires a more general data structuring on packet headers, imposing a cost on implementation, packet size, and hence performance.

Bianchi *et al.* [116] have discussed the advantages of adaptive MAC protocols. The authors present the trade-off between the flexibility, offered by implementing PHY/MAC functionalities in pure software, and the lack of performance to meet time-critical requirements. Since platforms such as the GNU Radio and the USRP comprise software components that run in a general purpose CPUs, they were not able to achieve equivalent performance as the traditional hardware-based platforms [57] [117]. Nychis *et al.* [48] have addressed this issue by identifying the minimum set of core MAC functions that must be implemented close to the radio to enable high performance and efficient MAC implementations in a high-latency SDR architecture. As a conclusion, the authors identified that time-sensitive functions (*e.g.*, carrier sense, backoff, scheduling, and packet recognition) should be implemented in hardware while the *user* and *kernel* spaces can handle the *upper MAC layer*. Besides, several other proposals [53] [58] [113] [118] investigated how *modular* designs and architectures could be addressed to enable MAC programmability and satisfactory performance.

### B. IEEE 802.15.4 Networks

Although *monolithic* and *parametric* architectures bring certain flexibility with high performance, keeping many *fullMAC* implementations without bloating the memory footprint is challenging. Sha *et al.* [45] addressed this challenge with Self-Adapting MAC Layer (SAML). SAML presented an approach where MAC protocols are stored within the NIC in a *modular* and shared design. To do so, SAML comprises a Reconfigurable MAC Architecture (RMA) that enables to switch among different MAC protocols at runtime. In addition, the RMA comprises a *MAC selection engine* that selects the most suitable MAC protocol according to the current network conditions and requirements. Figure 7 illustrates the overview of SAML system architecture.

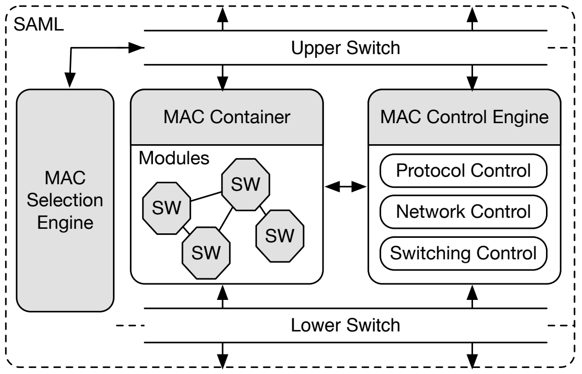


Fig. 7. Overview of the SAML system architecture, Sha *et al.* [45]

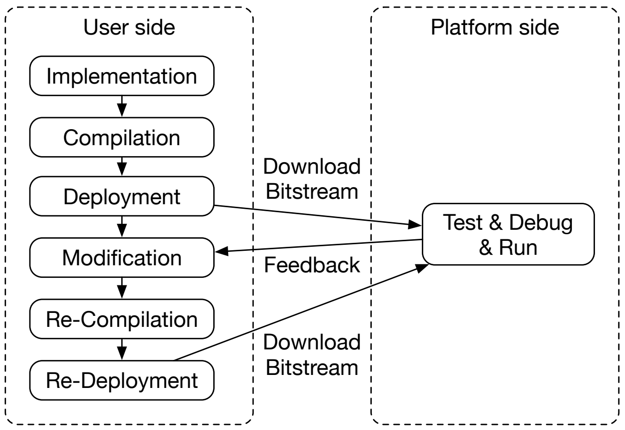
.

In response to changes in ambient conditions and application requirements, RMA enables that modules can be re-wired and so compose a wide range of MAC protocol implementations. Also, based on an ML algorithm, the MAC selection engine can optimally select the best modules to compose a MAC protocol implementation. Therefore, SAML provides an efficient *fullMAC* and *modular* MAC architecture in which improves: the memory usage (by storing modules in a shared design) and the decision-making process (by learning from the training data and building a decision tree).

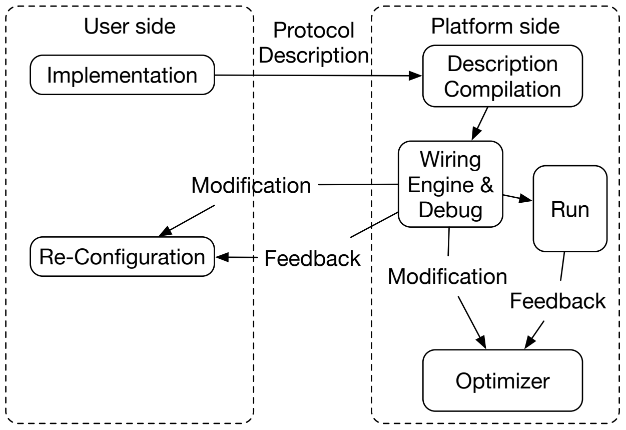
Riliskis *et al.* [118] presented a component-oriented approach where MAC protocols are considered as network controllers. Made up of interconnected components, each of these controllers has their own set of modules and parameters to be configured. Therefore, by abstracting performance requirements from the applications, different MAC protocol implementations can be composed.

Steiner *et al.* [119] presented a similar proposal where micro-components are used to design a large number of application-specific MAC protocols. Besides, these components support parameters adjustments such as synchronization, contention, and error detection. Even though the authors outlined the basic idea of the modular designs for WSNs, there are still parameters to be identified and control actions to be implemented. Nevertheless, some of the common critical WSN infrastructures require some predefined knowledge (*e.g.*, topology, traffic pattern, performance constraints); hence, *fullMAC* implementations usually cope with the flexibility needed.

De Mil *et al.* [120] proposed SnapMAC to provide more programmability to off-the-shelf IEEE 802.15.4 embedded



(a) Classical MAC development process on an SDR board.



(b) TRUMP MAC development process.

Fig. 8. Comparison between the classical protocol development process and the solution proposed by Zhang *et al.* [42]

.

radio platforms. In contrast with the aforementioned, by designing the MAC protocol as a chain of time-annotated commands, SnapMAC overcomes the timing constraint issue. However, the development has to add the timing information to each chain manually.

Vermeulen *et al.* [121] have proposed an alternative approach to provide fast runtime programmability for the MAC layer is implemented in Cross-Layer Adaptable Wireless System (CLAWS). Since CLAWS implements MAC real-time functionalities on FPGAs, it, therefore, achieves better performance than the other software-based approaches. However, CLAWS has to use a second CPU core, dedicated solely to run the real-time parts of the MAC layer.

Notwithstanding is WiSCoP [122] that takes advantage of the hybrid FPGA technology to provide more MAC programmability with high performance. Besides all those proposals provide MAC programmability, there are hardware and implementation restrictions to be considered in which strongly limits portability and re-usability of those proposals.

## C. IEEE 802.15.4

Parker *et al.* proposed a framework to design MAC protocols, called $\lambda$MAC [123]. $\lambda$MAC focuses on providing interfaces which can be used to implement common MAC functionalities separately, *i.e.*, as modules, and on the core MAC role of timing. Szczodrak *et al.* proposed a framework called Fennec Fox that enables to specify, at design time, MAC protocols for each application as well as events and policies that trigger its reconfigurations (*e.g.*, swap from Low Power Listening (LPL) MAC to CSMA) [124]. Black burst Integrated Protocol Stack (BiPS) however, besides comprising various MAC protocols and abstract interfaces to integrate new ones, it focuses on real-time-capable protocols [125]. Recently, Aoudia *et al.* [126] proposed a generic framework for modeling and compare MAC protocol schemes, which focuses on energy consumption, latency, and reliability. Moreover, there are other proposals [108] proposing frameworks for development and testing of MAC protocols for WSN that possess CRs capabilities (*i.e.*, CRSNs) [107]. Therefore, these new capabilities impose that, new proposals for MAC adaptability, programmability, and management are needed.

## D. CRN

In CRN, Ng *et al.* [60] have proposed a similar system where PHY and MAC layers are designed in a *modular* fashion, called Airblue. The authors have also implemented in FPGAs to support low latency cross-layer communication (*i.e.*, as a *fullMAC* implementation). Although Airblue achieved fast PHY/MAC intercommunication, the modules were not generic enough to handle complex protocols realization. Ansari *et al.* [40] [114] [127] and Zhang *et al.* [42] [128] otherwise have proposed a more sophisticated approach. The authors have proposed a component-based scheme and a tool-chain, called Toolchain for RUn-tiMe Protocol realization (TRUMP), for enabling fast prototyping of MAC protocols. Figure 8 illustrates the classical MAC development process on an SDR board in contrast with TRUMP MAC development process.

TRUMP consists of five parts that compose the MAC protocol development and realization cycle. First, a wiring engine that binds the MAC components together to compose a MAC protocol implementation. Second, a meta-language to describe the MAC design. Third, a compiler on the host which converts the meta-language for a particular platform. Fourth, a Graphical User Interface (GUI) for designing the MAC using the fundamental components. Last, an optimizer that enables efficient reconfiguration based on user preferences (*e.g.*, protocol memory size, execution speed, and power consumption).

To provide runtime reconfigurability, the authors shifted the protocol implementation from the *platform* to the *user side* (as illustrated in Figure 8). To correctly bind the components and execute the composed MAC protocol efficiently, TRUMP's wiring engine defines a unified API for all functions and logical connections. Also, TRUMP makes use of a linked list structure to maintain the MAC protocol components in an instruction set and then, a runtime execution manager takes care of the execution flow of the entire protocol. In this manner, TRUMP's MAC protocol development and realization cycle can compose reconfigurable MAC protocols in order of microseconds.

## *E. IEEE 802.11 Networks*

Gringoli *et al.* [53] proposed a *modular* architecture for programmable MAC with high performance in IEEE 802.11 networks, called MAC-Engine. MAC-Engine has an architecture where the logic of the MAC is shifted from the *driver* to the *firmware*, *i.e.*, from the host CPU to the NIC CPU. In this manner, by implementing a Programmable Finite State Machine (PFSM) at the *firmware* level, the card itself can support bytecode representation, and the logic can be injected into it at runtime.

Tinnirello *et al.* [129] proposed a similar approach where a programmable Extended Finite State Machine (XFSM) is introduced to execute MAC protocols at the *firmware-level*, called Wireless MAC Processor (WMP). Because that *lower* MAC features such as transmission, reception, and protocol control remain hard-coded in the NIC, more substantial MAC changes require access to the *firmware* code. Therefore, the authors re-flashed the *firmware* of the commercial Broadcom AirForce54G off-the-shelf chipset replacing its IEEE 802.11 WLAN MAC protocol to implement functionalities such as frame transmission, timer settings, frame classifiers, meta state machines, and control messages. Thus, WMP presented an approach with the *fullMAC scope* and *modular-level* of MAC programmability.

On top of the existing WMP architecture, Bianchi *et al.* [130] introduced a control framework to support MAC protocol code mobility, called MAClets. MAClets enables to transfer of MAC programs using regular packets among nodes within the wireless network (akin to the traditional active networks). In this manner, despite the unreliability of a fast response to network behavior changes, different MAC implementation can be distributed in the network. Also, MAC protocol actions can be triggered based on application requirements and network topology. Therefore, those proposals above (MAC-Engine, WMP, and MAClets) sit on the *kernel* and *hardware* sides to enable more access to the lower MAC functionalities and to provide automated and fast response according to network changes.

## *F. SDR and Cross-Platform*

Automated solutions can be efficient when decisions have to be taken extremely fast, and scenarios are quite predictable. For instance, the Chameleon-MAC [52] and LA-MAC [93] can adapt the MAC behavior according to the mobility characteristics of the environment. In this case, on Mobile Hoc Networks (MANET). Nevertheless, including the network administrator in the management loop may assist in troubleshooting. For instance, if the network administrator has access to information such as network density and application's demand, actions at the MAC layer could be triggered to prevent performance degradation (*e.g.*, bootstrap devices with a *schedule-based* protocol). Therefore, besides the presented TRUMP for CRN, a few other frameworks [41] [47] [2] were proposed to provide more interaction with the MAC layer.

As in software radio architectures (*e.g.*, GNU Radio, The Open Source Software Communication Architecture Implementation::Embedded (OSSIE) [132], and Sora [133]), Sutton *et al.* [41] proposed an architecture able to perform runtime reconfiguration for all layers of the network stack, called Iris. Besides exposing many parameters for dynamic MAC adjustment, Iris also allows reconfiguring PHY layer blocks, such as modulation schemes and filters. However, its architecture was not designed to provide accurate control over the *lower* MAC layer, so it does not guarantee that packets are transmitted at exact specific time slots [2].

Addressing real-time MAC programmability, Demirors *et al.* [47] have proposed Real-time Reconfigurable Radio (RcUBe), a radio framework based on abstractions in which offers real-time reconfigurability and optimization capabilities at the PHY, MAC, and network layers of the protocol stack. RcUBe divides the architecture of a network node into four planes: *decision*, *control*, *data*, and *register* planes. The *decision plane* consists of user-defined decision algorithms. The *control plane* controls the logic for routing, MAC protocol execution, and *data plane* management. *Data plane* is responsible for data processing and the *register plane* stores and manages access to system parameters and environmental information. Through this separation, RcUBe allows defining algorithms in runtime besides a clean separation between decision-making mechanisms and the execution of the protocol itself.

Furthermore, by extending the same principle of time-annotated commands and state machines presented in SnapMAC and WMP, Jooris *et al.* [2] proposed Time Annotated Instruction Set Computer (TAISC)[9], a hardware independent MAC protocol development and management framework. Similarly to TRUMP, TAISC also shifts the MAC development from the *platform* to the *user* side. To develop and execute a MAC protocol, TAISC has a four-steps work-flow. First, the MAC protocol designer describes the MAC logic using predefined commands in a C-like language or through a drag-and-drop interface. Then, a human-readable sequence is compiled into efficient, device-specific binary bytecode that can be executed by the TAISC execution engine running on the radio platform. Thereby, a bytecode is wirelessly disseminated and then added to the MAC application repository within the local TAISC execution engine. Last, the TAISC kernel executes the bytecode that contains the developed MAC protocol. In this manner, MAC protocols can be described in a platform independent language and optimized for specific radio chips.

Table IV presents a list of flexible radio development frameworks and MAC layer architectures. Columns *functionality* and *order of time* are there to represent capability versus feasibility to perform specific MAC reconfiguration analyzed by the authors (*e.g.*, send software-based ACKs in order of milliseconds or microseconds).

[9]TAISC cross-platform: http://www.wishful-project.eu/taisc

TABLE IV
COMPARISON OF FLEXIBLE RADIO DEVELOPMENT FRAMEWORKS AND MAC LAYER ARCHITECTURES.

| **Name** | **Year** | **Main Target** | **Target Platform** | **Reconfigurability** | | **Functionality** | **Order of Time** | **Self-adaptable** |
|---|---|---|---|---|---|---|---|---|
| | | | | **PHY** | **MAC** | | | |
| GNU Radio [131] & Click [56] | 2001 & 2000 | SDR creation & Modular Router | SDR | Yes | Off-line | Packet Forwarding | Few ms | No |
| C-MAC [119] | 2010 | MAC reconfigurability | Off-the-shelf IEEE 802.15.4 | Yes | At runtime | Apply CSMA/CA, ACK, and beacons | Few sec (RTT-based) | No |
| Iris [41] | 2010 | Runtime reconfiguration of all layers of the network stack | SDR | Yes | At runtime | MAC composition | N/S | No |
| λMAC [123] | 2010 | MAC programmability and code reuse | Off-the-shelf IEEE 802.15.4 | No | At runtime | MAC design optimization | N/A | N/A |
| MAC-Engine [53] | 2011 | MAC reconfigurability without performance loss | Off-the-shelf IEEE 802.11 | No | At runtime | MAC composition | N/S | Yes |
| TRUMP [42] | 2011 | Toolchain for reconfiguration of PHY/MAC layers | CR | Yes | At runtime | One variable assignment | Few µs | Yes |
| MAClets [130] | 2012 | MAC reconfigurability using active networking principles | Off-the-shelf IEEE 802.11 | No | At runtime (not reliable) | MAC code transfer | RTT-based | Yes |
| WMP [129] | 2012 | MAC reconfigurability without performance loss | Off-the-shelf IEEE 802.11 | No | Partially (manual) | N/A | N/A | Yes |
| Correia *et al.* [108] | 2012 | MAC protocols simulation | CR | Yes | At runtime | MAC development and testing | N/A | Yes |
| SAML [45] | 2013 | MAC composition by re-wiring MAC components | Off-the-shelf IEEE 802.15.4 | No | At runtime | MAC composition | Few ms | Yes |
| Fennec Fox [124] | 2013 | MAC reconfigurability | Off-the-shelf IEEE 802.15.4 | Yes | At runtime | MAC swapping (LPL MAC to CSMA) | Few ms | Yes |
| SnapMAC [120] | 2014 | Generic MAC/PHY development architecture | Off-the-shelf IEEE 802.15.4 | No | At runtime | Frame transmission | Few µs (preamble) Few µs (per byte) | No |
| RcUBe [47] | 2015 | Real-rime reconfigurable radio based on abstractions | SDR | Yes | At runtime | MAC swapping (DCF to TDMA) | Few sec | Yes |
| CLAWS [121] | 2015 | Platform able to access to all layers of the network stack | Off-the-shelf IEEE 802.15.4 | Yes | At runtime | Send ACK | Few µs (FPGAs) | No |
| TAISC [2] | 2016 | Hardware independent MAC development and management | Cross-platform | No | At runtime | Software-based ACKs generation | Few ms | No |
| BiPS [125] | 2016 | MAC composition for real-time systems | Off-the-shelf IEEE 802.15.4 | No | At runtime | Instructions delay | Few ms | No |
| Aoudia *et al.* [126] | 2017 | Analytical evaluation of different MAC schemes | Off-the-shelf IEEE 802.15.4 | No | At runtime | N/A | N/A | Yes |
| WiSCoP [122] | 2017 | Development of integrated cross-layer network designs | Off-the-shelf IEEE 802.15.4 | Yes | At runtime | N/S | N/S | No |

## G. Lessons Learned

According to our terminology, *modular-level* architectures present the highest *level* of programmability. Besides the advantages of complete flexibility, we identified some valuable lessons learned regarding such architectures:

*1) Most modular frameworks still depend on specific hardware and technology:* Because MAC prototyping frameworks *depend on specific hardware and technology*, they cannot be compared. RcUBe takes approximately 4 seconds to change from DCF to TDMA completely. This time starts when the node triggers a MAC protocol change until the moment where the change is performed and monitored. TRUMP was evaluated on WARP boards/sensor nodes and TAISC on top of MSP430F5437 micro-controller and the CC2520 IEEE802.15.4 radio integrated on the RM090 embedded platform. Therefore, the execution time of these frameworks have different granularities and generally aligned with different and specific hardware. However, it is possible to state that TRUMP can achieve performance in order of few microseconds while TAISC and RcUBe with a bit more overhead, but TAISC is cross-platform and RcUBe runs on SDRs. Therefore, besides SDRs cost may still be higher than other radios, there are some benefits such as software reuse, that reduced development costs dramatically. Moreover, since real-time SDRs are becoming a reality, the best options for a programmable MAC framework are the ones aiming for SDRs.

*2) Modular designs offer more room for innovation:* Comparing the performance from the different frameworks and platforms may lead to different conclusions, depending on the analyzed perspective. For instance, while CLAWS takes only 8 µs to send an ACK, TAISC takes 612 µs. In a first moment, CLAWS has a faster mechanism for transmitting ACKs then TAISC. On the other hand, by creating software-based ACKs using TAISC, protocol designers can introduce novel optimization algorithms and gather network information within the ACKs packets (*e.g.*, piggyback status and link information and aggregate ACKs within a single packet). Hence, there is more *room for innovation* for the software-based MAC development frameworks.

*3) Modular designs provide a more future-proof MAC layer:* Due to hardware obsolescence (*i.e.*, radio chips typically have a life cycle of just a few years), MAC protocols need to be re-designed to adapt to the newer versions or their products. Nowadays, MAC protocols cannot be entirely reused on different radio chips and technologies. Several proposals are aiming for more and more MAC programmability in most generic and abstracted way possible (*e.g.*, SnapMAC, MAClets, Iris, WMP, TRUMP, RcUBe, and TAISC). For instance, TAISC enables a more general design of MAC protocols where developers can design the protocol once, and then compile and reuse them on different radio chips. Moreover, TAISC supports to change radio instructions in order of microseconds, from the moment the radio execute the command plus the time for the changes take effect. Therefore, such proposals represent the state-of-the-art and the more *future-proof programmable MAC layers* for wireless networks.

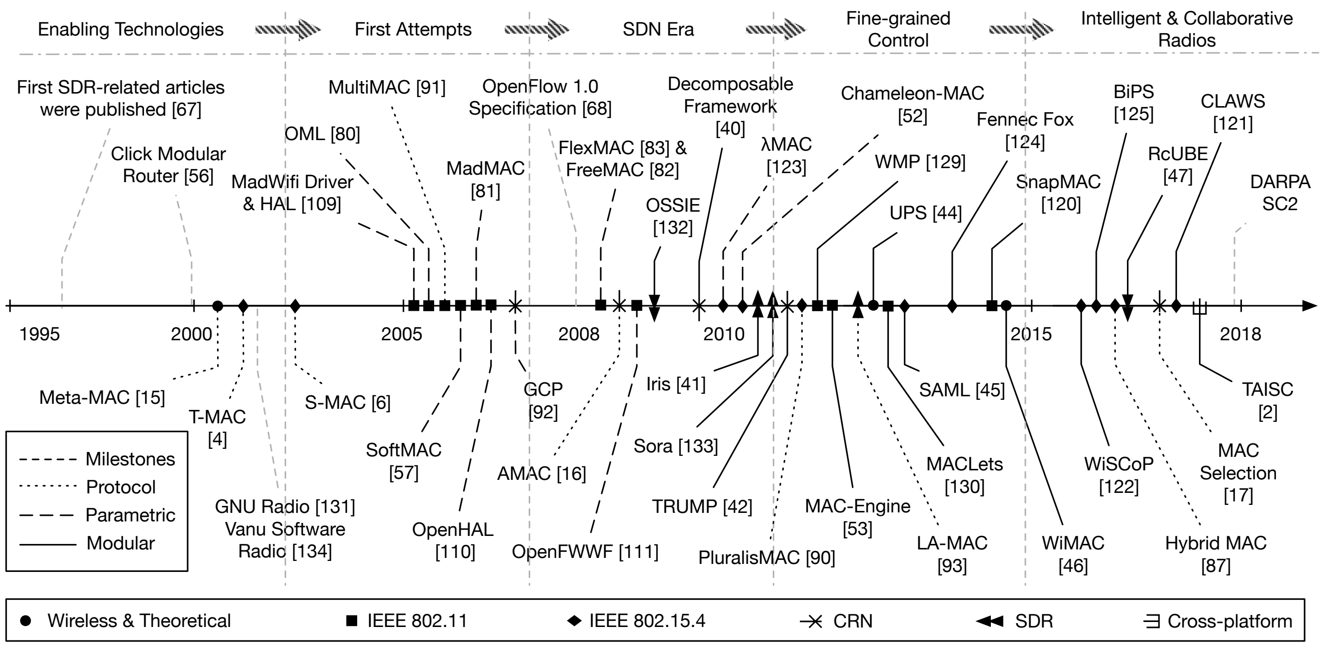


Fig. 9. Evolution timeline on programmable wireless MAC protocols according to the supported *level* of programmability and wireless technology.

## VII. Discussion

In this section, we present a discussion regarding programmable MAC protocols along with an evolution timeline. Then, we depict some questions that should be answered to determine the MAC programmability needed. Last, we summarize the most important simulators, prototyping platforms, and testbeds for programmable wireless MAC protocols.

### A. The Evolution of MAC Programmability

The notions of connectivity at any place and any time fostered radio technologies to support to fine-grained control over wireless networks (*e.g.*, GNU Radio [131] and Vanu Software Radio [134]) [67]. The development of the next generation of cellular networks 5G is fostering more and more in this research area due to the Key Performance Indicators (KPIs)[10], proposed by the 5GPP consortium. These KPIs foster the fast deployment of novel applications with reduction of the network management Operating Expense (OPEX) and achieve several aspects of IoT devices communication (*e.g.*, 90% reduction in energy usage, and the connection density of 1 million devices/km$^2$).

Since MAC protocols significantly impact wireless performance metrics (*e.g.*, throughput, energy consumption, and reliability), different approaches for MAC programmability been studied over the years to improve wireless connectivity [2]. Figure 9 shows an evolution timeline representing the recent effort on programmable MAC protocols for wireless networks. The timeline depicts efforts from the moment where the enabling technologies, such as SDRs, enabled fast prototyping until nowadays where more intelligent and collaborative designs become the new goal (*e.g.*, DARPA's Spectrum Collaboration Challenge (SC2)[11]). The *programmability level* is illustrated along with the corresponding wireless technology and the *milestones* that influenced the development of programmable MAC implementations.

*1) Enabling technologies:* In the 2000's, Meta-MAC [15] introduced a method that dynamically selects the best protocol, without knowing in advance which one matches the potentially changing and unpredictable network conditions. However, despite having several *fullMAC* implementations, there was no one-size-fits-all solution. Because application requirements often change and given the dynamic nature of such networks, the need to interact with MAC protocol logic was evident. In 2005, the MadWifi project [109] arose as one of the first successful attempts that enabled MAC protocol interaction. By overwriting the stock driver of the Atheros wireless chipsets into a WLAN Linux driver, it was possible to have more control over specific functionalities of the radio (*e.g.*, seamless roaming, 4-address header support, and background scanning). However, the MadWifi driver was dependent on a HAL in which provides access to the *firmware* within the NIC. Unfortunately, HAL was a closed-source implementation, and it was distributed in binary form only so researchers cannot easily modify it. Although some features remained inaccessible from the *kernel* and *user* sides (*e.g.*, frequency, channel, and transmission power), these efforts are considered one of the first steps towards *SDMAC* protocols.

*2) First attempts:* Inspired by the success of *overlay networks*, a few contributions [80]–[83] considered having an *overlay layer* to enable MAC protocol interaction. Placed on top of the existing MAC layer, these *overlay layers* enabled some control over the MAC. Having part of the MAC into means of software allows better integration with routing and

[10]https://5g-ppp.eu/kpis/

[11]https://spectrumcollaborationchallenge.com/

application requirements. For instance, with OML, it is possible to perform access control and scheduling over the IEEE 802.11 MAC layer, and it does not require any changes on the MAC hardware or at the standard itself. On the other hand, as these introduce an extra layer, they suffer some additional overhead compared to implementing the same changes without an extra layer. In addition, changes at the MAC layer are always limited by the NIC that exposes the tunable parameters to the other layers. Therefore, proposals [57] [81] started overriding the MAC layer or even adding an extra driver to enable more control over the MAC. However, due to their hardware-specific design and the fact that is not possible to have full access to radio functionalities, they still share the same applicability issues.

*3) SDN era:* To support high performance and efficient MAC implementations, Nychis *et al.* [48] identified the minimum set of core MAC functions that must be implemented close to the radio in a high-latency SDR architecture. Besides, by exploiting parallel programming techniques, radio technologies started to enable that *software-defined* MAC protocols to achieve satisfactory performance regarding high-throughput and low latency [133]. For instance, with Sora [133], to accelerate PHY processing with data-level parallelism, Single Instruction Multiple Data (SIMD) extensions were used in modern General-Purpose Processors (GPP). Although these extensions are designed for multimedia and graphics applications, in wireless signal processing, many PHY algorithms have fixed computation structures so can be mapped to large vector operations.

*4) Fine-grained control:* Besides new *monolithic* and *parametric level* approaches (*e.g.*, AMAC [16], FreeMAC [82], and FlexMAC [135]), fundamental MAC functionalities have been abstracted into modules and, by rewiring and reconfiguring them, it is possible to realize a wide range of implementations Chameleon-MAC [52], WMP [129], MAC-Engine [53], and some other self-adapting solutions [13] [45] [104] [105] often make use of PFSM at the *firmware level* to quickly change modules and so the MAC according to the environment. Several frameworks [2] [41] [42] [46] [47] [120]–[122] [130] proposed different methodologies for protocol development and realization cycle. TRUMP and TAISC stood out from the others because of their MAC development processes. Both proposals shifted the protocol implementation from the *hardware platform side* to *the user side* and, hence, offered a clean separation between MAC protocol design and the execution of the protocol itself.

*5) Intelligent & collaborative radios:* After the release of open source version for the HAL [109], the rise of SDN with the OpenFlow protocol [68] in 2008, and with *Tier 2* SDRs becoming a reality, programmability of wireless MAC protocols grabbed even more attention [67]. Efforts to enhance cross-layering interaction and standardization were proposed (*e.g.*, UPS [44]) and cross-platform frameworks such as TAISC become the new trend. Many recent studies [73] [136]–[139] have also been stating that by decoupling the *data* and *control* planes, regardless of wireless technology. Network devices are supposed to act just as simple packet forwarding boxes while the logic is implemented within network controllers. Because real-time SDRs recently enabled to design the protocol quickly with high versatility and the need for cooperation and coexistence among wireless networks, a lot is being investigated on *SDMAC* protocols on wireless networks.

## B. Programmable MAC Protocols: Things to Keep in Mind

Programmable MAC protocols are required to control the wireless communication and thus cope with application demands. There are several options to address programmability on MAC protocols. Each of them has their advantages and disadvantages. Hardware-based implementations can perfectly fit on controlled environments, where application requirements do not change that often (*e.g.*, habitat monitoring and health). However, current wireless networks are more dynamic and heterogeneous (*e.g.*, cellular and vehicular networks). Hence, programmable MAC layers should be addressed. To determine which degree of MAC programmability to use, there are a few questions that should be answered:

1) *What is the degree of flexibility needed? Monolithic* approaches offer some flexibility, but they are limited. The *monolithic-level* architectures support switching among predefined MAC implementations and thus provide a certain degree of flexibility (*e.g.*, changing from a *schedule-based* to a *contention-based* MAC protocol). Therefore, it is more appropriate when the network does not suffer excessive interference, application demands are quite predictable, protocol implementations are stable, widely adopted, and the aim is to switch between a handful of standardized MAC implementations. However, when external information is required to adapt the MAC (*e.g.*, data regarding monitoring, upper layers requirements, or high-level policies defined by the network administrator), *parametric* and *modular* architectures are more appropriate than *monolithic*. With some fine-tuning at the *upper* MAC layer, it is possible to adapt and therefore improve network connectivity not only based on local decisions but also based on information from other applications and higher layers. In this case, aspects such as the packet format, back-off mechanism, or the duty cycle can be easily modified. Therefore, for a more *future-proof* MAC layer, the *modular* architecture recommended as it offers the higher degree of programmability and so more *room for innovation* at the MAC level.
2) *Is there a need for adapting to other networks?* When multiple wireless networks have to compete and share the spectrum, collaborative information has to be exchanged and processed to understand and control network behavior. Therefore, it is expected that a MAC protocol incorporates this knowledge within its logic. Most *monolithic* approaches implement the MAC layer within the NIC and, hence, it is difficult to interact with them. Information regarding collaboration is still not widely addressed by the research community and therefore does not have standardized APIs for defining what and how messages shall be exchanged. Thus, MAC implementations may suffer modifications to incorporate

collaborative information. However, the chosen MAC layer architecture depends on the degree of flexibility needed as well, *i.e.*, if control access mechanism needs to be customized or not defines how much from the MAC layer has to be adapted). Over time, as wireless networks have to coexist next to one another and given the unforeseen and dynamic network requirements, *parametric* or *modular* architectures (that usually support *softMAC* or *SDMAC programmability levels*) are the most advised architectures to support be used.

3) *Is it required to have accurate time control at the MAC layer?* In most proposals, medium access control mechanisms remain within the NIC, *i.e.*, in the *hardware* space, and *parametric-level* architectures do not have control over them. In this case, the timing for transmitting and receiving packets cannot be precisely controlled from the *kernel* or *user* spaces. *Monolithic* architectures have direct access to those mechanisms, but it is all implemented in the *hardware* space. If the MAC layer has to provide a high *level* of programmability as well, for instance, *upper* and *lower* MAC layers, then *modular* architectures are the most appropriated. Some of these *modular* architectures are only flexible because they run on SDRs, but only the real-time SDRs are capable of achieving equivalent performance as ASICs and FPGAs in terms of runtime latency. Besides, they are quite more expensive than the normal SDRs. Therefore, architectures may vary depending on the available hardware and desired flexibility. If the flexibility desired is low and the hardware does not offer *SDMAC programmability scope*, then *monolithic* architectures shall be picked. Otherwise, *parametric* can deal with *upper* MAC layer and *modular* architectures can be chosen to customize the MAC layer completely, depending on the desired *level* of programmability.

4) *What is the target hardware?* An important aspect that has to be taken into account when designing wireless MAC protocols is the available hardware for implementing them. Real-time SDRs are still expensive, so if the requirements do not include fast processing and complete flexibility offered by the *modular* architectures, feasible options are also implementing *monolithic* and *parametric* architectures (*e.g.*, on ASICs or FPGAs). However, despite its cost, the design cycle is also a metric that has to be considered. Because that MAC layer has to be designed using HDL, the easy customization of the high-performance blocks is still challenging. Therefore, tasks such as testing and deployment MAC protocols on ASICs and FPGAs take relatively more time than using an SDR or on the real-time SDRs (*e.g.*, *SDMAC* on real-time SDRs). Therefore, the chosen hardware to use depends on cost, designing cycle, and on the desired *level* of programmability that the MAC layer has to support.

5) *Is there a need to deploy unforeseen MAC functionality in the future?* There are cases where the MAC layer has to be completely reprogrammed, and new features may have to be added and validated before releasing a new implementation. For instance, when application requirements and network conditions change, the MAC layer has to adapt accordingly. In some cases, it is necessary to build a completely different protocol, so *SDMAC programmability level* is recommended. *Parametric* and *monolithic* architectures normally have *softMAC* of *full-MAC levels* of programmability so are not the best options. On the other hand, *modular* architectures allow that features can be changed independently. Thus, combined with real-time SDRs, runtime fine-grained MAC-layer modifications can be performed and thus maintaining it up to date. In general, characteristics such as mobility, application requirements, demand, and given the shared nature of the wireless environment require *modular* designs to support a high degree of MAC programmability.

6) *Is there a need for local or global MAC control/management?* Control and management activities have been a longstanding research challenge, also in wired networks. There are many different ways of performing them, and there are advantages and disadvantages of choosing one or the other. Although the centralized approach gives the overall view of the network, it also implies in some issues (*e.g.*, single point of failure and the possibility of creating bottlenecks towards the control/management entity). Decentralizing the control and management among network nodes can considerably release the responsibility from one single entity to several. However, it also implies that synchronization mechanisms have to be implemented to keep all of these nodes aware of the state of the network, *i.e.*, the current MAC protocol implementation. Since modular architectures enable to split different control and management tasks into different software modules, *modular-level* architectures are advised to be addressed. In this manner, modules can be individually enabled/disabled according to the administrator needs.

## C. Getting Started: Simulators, Platforms, & Testbeds

In the literature, there is plenty of work where evaluations are conducted through analytic modeling combined with simulations. Simulators such as ns-3 [140], OMNeT++ [141], QualNet [142], Avrora [143], TOSSIM [144], and MATLAB [145] models are used to test and compare MAC protocol implementations. In general, these simulators provide a software platform where researchers can develop and test their own MAC protocols implementations and test them in diverse scenarios. However, several assumptions and simplifications are considered and, hence, may mislead to inaccurate representations of the wireless spectrum. Therefore, it is advised to combine different evaluation techniques to ensure accurate and reliable results [146]. Once the questions above (Subsection VII-B) are answered, several prototyping frameworks and testbeds can assist researchers in implementing their MAC protocols in more realistic environments.

Tables III and IV present the most relevant frameworks and platforms for programming wireless MAC protocols. In

Table III, the available firmware, drivers, and software overlays are concentrated for Atheros chipsets. In addition, they offer limited access to the MAC layer (*i.e.*, *parametric-level* of programmability). On the other hand, Table IV presents the frameworks where cognitive capabilities are enabled in addition to the complete software-coded design. For instance, in IEEE 802.15.4 networks, WMP[12] offers an open source framework that is suitable for the AirForce54G family of Broadcom wireless NICs. TAISC[13] otherwise, present a more sophisticated cross-platform where researchers can develop their MAC from scratch. Besides, TAISC can guarantee the exact time of transmission of packets, while minimizing the radio-on time.

Nowadays, there are several testbeds in which researchers can experiment their wireless solutions in more realistic environments. DARPA has recently released the world largest wireless testbed of all time with 256 SDR units, called Colosseum [147]. There, researchers can participate in a collaborative ML competition to overcome scarcity in the wireless spectrum, called SC2. Moreover, there other testbeds (*e.g.*, Federation for Fire (Fed4Fire) [148], Global Environment for Network Innovations (GENI) [149], and OFELIA [150]) that also offer support for experimenting with wireless networks, SDN, and OpenFlow. Usually, once the solution is developed, researchers from both academia and industry reserve resources on these testbeds to perform large-scale experimentation, which requires sophisticated infrastructure and a large number of wireless devices.

## VIII. Research Challenges

Alongside with the rise of the IoT [1], the SDN paradigm, and the SDRs evolution, the need for intelligent control and management of wireless devices is more and more evident. Although SDN enables high-level network programmability and a centralized view of the network resources, the paradigm itself already introduces its challenges [151]. Besides the shared and competitive wireless environment, the limited processing capacity of the IoT devices and the lack of standardized MIs make management even more challenging. Therefore, in this section, we identify and discuss a non-exhaustive list of research challenges regarding the programmability of wireless MAC protocols based on the lessons learned. We highlight the main gaps we identified between the state-of-the-art regarding design, prototype, deployment, control, and management of wireless MAC protocols.

### A. Intelligent MAC Layer Adaptation

Given the wireless dynamic nature, it is unfeasible to design heuristics to cope with the network behavior beforehand. Because of that, some proposals [17] [152]–[158] are making use of ML algorithms to improve their MAC selection/configuration mechanisms. *FullMAC* implementations have been switched based on classifications learned from the environment. Zubir *et al.* [38] presented a survey where protocols that use ML are compared based on their resulting performance, as well as the previous surveys where MAC algorithms were based in heuristics and probabilities. Besides those approaches provide a limited *level* of programmability, they have enabled intelligent selection and run-time adaptation of the MAC layer. However, there is still *the need for correlation and integration of Artificial Intelligence (AI) algorithms with technology-specific resource constraints and QoS requirements from higher layers*. In this manner, more intelligent MAC adaptation mechanisms could be developed to guarantee QoS in wireless networks.

### B. High-Level & Platform-Independent MAC Frameworks

The MAC layer is usually hardware-specific and is usually optimized for a specific standard and technology. MAC protocol's logic has to be aligned with the capabilities supported by their firmware and drivers (*e.g.*, MadWifi [109] and OpenFWWF [111]). *The need for high-level frameworks and platform-independent MAC designs* comes up when real-time SDRs allowed different standards and technologies to run on the same radio. Most effort on programmable wireless MAC protocols concentrates on IEEE 802.11, CRNs, and IEEE 802.15.4 standards. Therefore, frameworks should allow to describe the protocol logic in high-level and then generate a technology-specific implementation for them. For instance, TAISC presented a framework in which protocols can be described independently of the platform [2]. Each command represents one general behavior of the MAC protocol, and it is mostly independent of hardware. Customized parameters, time elapsed, pre-requirements, and the actual code is disseminated among network nodes and then executed locally. However, many of the wireless technologies are still not supported by TAISC, so there are still gaps to be fulfilled.

### C. MAC Prototyping & Testing

Traditionally, wireless MAC protocols have been evaluated based on theoretical analysis and computer-based simulations [27]. These over-simplified assumptions are not able to provide accurate PHY information and realistic channel conditions [159]. By making use of testbeds such as GENI [160] and FED4FIRE [161], it is possible to experiment and gather reliable and scalable results. Moreover, a recent powerful emulator of radio-signal traffic was created for the DARPA SC2. Therefore, there is *the need for more sophisticated and high-scale prototyping and testing procedures for MAC evaluations*. Furthermore, since new communication capabilities were enabled (*e.g.*, FD in CRN), new requirements arise at the MAC layer as well (*e.g.*, self-interference management and sensing overhead control).

Several articles addressed novel MAC-layer aspects that arise with advanced wireless technologies [64] [65]. For instance, packet fragmentation, collision probability, and sensing performance have been addressed in FD-CRNs [162]–[166]. However, many others are yet to be researched [64]. The hidden terminal problem, the potential high bit error rates, and the packet loss ratios have yet to be addressed in FD-CRNs. For instance, packet error rates can be reduced by

[12]http://wmp.tti.unipa.it/index.php

[13]https://github.com/bjooris/taisc

provisioning larger buffers. Therefore, extensive research and tests on appropriately sizing the buffers and minimizing the bit error rates have to be addressed. Therefore, considering the specific requirements from each of the wireless technologies, enhanced MAC layer mechanisms have to be designed. In this manner, evaluations should be able to represent how MAC protocols implementation behave in more realistic and dynamic environments (*e.g.*, coexisting with cooperative and non-cooperative wireless technologies).

### *D. MAC Configuration Management & Troubleshooting*

The impact that MAC-layer customizations may cause in the network must be intelligently managed. For example, software-based ACKs can be designed to be aggregated and sent based on destination MAC addresses. In this manner, fewer packets are generated and transmitted across the network, *i.e.*, fewer packet headers are created, transmitted, and checked at every hop. Therefore, this single ACK feature can significantly improve the overall network efficiency. However, there is *the need for management and correlation among enabled functionalities at the MAC layer*. In this case, if there is another functionality enabled, for instance, a packet retransmission feature together with the aggregated ACKs can lead to problems if both are not properly configured and aware of one another. When aggregated ACKs are enabled, ACK packets remain more time at the destination, due to the aggregation process, so the timeout for retransmissions have to be higher than the time to send the packet with aggregated ACKs to its destination. Otherwise, unnecessary retransmissions are triggered when packets are received at the destination, and the source still does not know about it, *i.e.*, causing extra overhead. Therefore, the correlation among MAC-enabled features should be present. Hence, dependencies and requirements shall be respected at the MAC level.

### *E. Cross-Technology MAC Cooperation*

MAC protocols have been adapted mainly based on performance degradation thresholds or/and predicted future traffic patterns [16]. These metrics can be analyzed in many different ways (*e.g.*, considering nodes performance thresholds, voting schemes, QoS requirements, and resource limitations). However, when there is interference from other networks, to predict future traffic patterns is challenging. Some wireless technologies have to share and compete for spectrum bands (*e.g.*, IEEE 802.11 and LTE-U). DARPA recently released another SC2 where the goal is to overcome the scarcity in the Radio Frequency (RF) spectrum. To overcome interference and improve wireless connectivity, ML and AI are indicated to be used together with collaborative information about other networks and therefore improve the overall network performance (*e.g.*, overall throughput, delay, and latency).

Moreover, an interesting research direction on cross-technology MAC cooperation is to examine and optimize the interactions between the MAC layer and the advanced features from the novel wireless technologies. The interactions between FD-CRNs and the different access and metropolitan area networks that support backhaul from wireless FD-CRNs is yet to be investigated [64]. For instance, in Fiber-Wireless (FiWi) networks, cooperation between wireless networks and fiber networks is considered [167] [168]. Commonly, optical networks that support wireless networks have specific bandwidth allocation and MAC protocols [169]–[173], which can jointly be investigated with FD-CRNs. Moreover, end-to-end connections involving FD-CRNs may traverse other common access networks (*e.g.*, coax cable and Digital Subscriber Line (DSL) or specific metropolitan area network structures) in which MAC-layer features might be exploited as well.

In this manner, wireless networks can cooperate and make fair use of the shared spectrum. However, the idea of collaboration is still not widely addressed at the MAC layer. Therefore, there is *the need for MAC programming interfaces to allow cooperation among wireless networks*. In this manner, future programmable wireless MAC protocols can cooperate and, hence, make fair use of the wireless spectrum.

### *F. Cross-Technology Monitoring & MAC Correlation*

Monitoring is one of the fundamental management activities to understand and control the network behavior [174]. Nowadays, different applications and services often have to share the same wireless infrastructure and the spectral bands, making it very challenging to meet diverging QoS requirements. In the current heterogeneous wireless environment, there are plenty of wireless devices besides other electronics appliances which use the same frequency to communicate, hence, generating interference. Because of this, there is *the need for monitoring and recognition of cooperative and non-cooperative networks at the MAC layer*. In this manner, MAC protocols could perform cooperation and interference mitigation. Networking metrics should be monitored and analyzed regardless of wireless technology, and standardized and high-level interfaces should be provided to gather network status information without exposing system-level details (*e.g.*, polling network elements, aggregating statistics, and identifying network changing events). Current proposals do not address MAC layer management based on monitoring information about other networks. Therefore, *there is the need for defining how monitoring information should be carried out and correlated with the MAC layer features* to trigger proper MAC-layer modifications.

### *G. SDN Controllers & MAC Programmability*

Much has been discussed about SDN and network management by itself, mostly from the perspective of where SDN is taken as a management tool [175]. Several approaches use SDN to deal with management activities because it simplifies or even solves some traditional management activities. For example, because the fact of forwarding devices need to be registered or discovered by the network, in SDN, all forwarding devices establish a communication path with the controller (*i.e.*, between *forwarding* and *control* planes) and the network discovery management activity—a traditional management activity—is intrinsically solved. However, SDN also creates new management challenges that are not yet discovered or widely addressed in the literature (*e.g.*, controller placement and resilience) [151]. Besides, wireless networks introduce many other peculiarities and problems that are not present in wired networks (*e.g.*, interference and multipath fading) and current SDN controllers are not designed to cope with them. Because that MAC protocols significantly impact wireless performance metrics, there is *the need for SDN controllers able to control and manage the wireless MAC layer*. In this manner, SDN management solutions have to be able to manage specific wireless features such as the MAC layer and therefore fulfill application-layer requirements, especially on the presence of coexisting wireless technologies.

### H. Standard Northbound and Southbound APIs

Recently, there is some effort that use SDN concepts to enable control over wireless networks [72]–[75]. The evolution of SDRs and the SDN paradigm enabled on-the-fly network reconfigurations with high performance and high versatility on the entire MAC layer. Thus, fostering the research towards programmable wireless MAC protocols [49]. However, the current SDN architecture does not offer a fully SDN-enabled control and management over wireless MAC protocols [71]. The absence of standardized MIs for all SDN planes, especially between *management* and *control* planes, makes it difficult to gather information for management purposes, even in wired networks (*e.g.*, control channel statistics) [174]. We argue that the standardization process of all these MIs may foster the development of SDN management solutions to manage any plane regardless of application, controller, or forwarding devices. Particularly, MAC layer configurations cannot be controlled by the current *de facto* standard southbound API, *i.e.*, the OpenFlow protocol [68]. In addition, there are no standard northbound API to enable communication with the higher layers. Therefore, there is *the need for standardized southbound and northbound APIs to access the MAC layer*, regardless of wireless technology.

### I. Security & Isolation

In SDN, controllers have to offer isolation among the different network slices. Devices resources may have to be shared among different network slices, and therefore network resources have to be managed appropriately. FlowVisor [176] is one example of an SDN controller that enables network virtualization by dividing a physical network into multiple logical networks. In wireless, it is not different. Devices may have to accomplish different requirements and, to do so, may have to employ different MAC layer implementations. In this case, multiple MAC layer implementations should run independently and only authorized controllers may have access to their logic. Therefore, there is *the need for security mechanisms and network traffic isolation at the MAC layer.* In this manner, constraints and requirements can be met, and multiple MAC layers can run simultaneously in the same wireless NIC, without conflicting with each other.

### J. Over-the-air MAC Layer Update

To have a dedicated wired connection to perform control and management of each wireless device may be not feasible or practical. Therefore, over-the-air MAC layer updates are performed. As the actual trend is to remove the logic from forwarding devices to a centralized entity (*e.g.*, SDN controller responsible for the network logic while network devices are only simple packet forwarding boxes). MAClets [130] enabled to transfer of MAC programs using regular packets among nodes within the wireless network (akin to the traditional active networks). This feature brings the flexibility to perform over-the-air MAC updates but brings the challenge of *the need for MAC dissemination mechanisms* that ensure synchronization and resilience when, for instance, an SDN controller triggers a MAC layer modification, and not all devices are reachable or available. Some resource-constrained devices apply duty cycles and therefore, during this period, cannot apply a new MAC configuration during that period. In this manner, reliable mechanisms should be provided for over-the-air MAC layer updates in wireless networks.

## IX. Conclusion

Together with the popularity of SDN, IoT, and the increasing performance of SDRs, fully programmable data link layers start to achieve the desired level of flexibility without performance degradation. Plenty of software-based protocols have been proposed together with platforms, frameworks, software overlays, and modular designs for the MAC layer. However, over the last decades, researchers have been summarizing protocols based on their mode-of-operation and resulted/expected behavior. In this survey, we took a different perspective by analyzing the *scope* and the *level* of programmability these efforts support. By classifying proposals as such, insights are provided regarding which category of MAC programmability suits best according to environmental conditions, application demands, and resource availability. Therefore, we initially defined a coherent terminology to classify wireless MAC protocols based on their *scope* and *level* of programmability. As current terminologies do not address these aspects we aimed, and they are often used interchangeably, it was hard to categorize them based on previous criteria. With those three categories: *fullMAC*, *softMAC*, and *SDM*, we classified, discuss, and compare the current relevant efforts accordingly.

Aiming to encourage future proposals to address programmability at the MAC layer, we discussed the advantages and disadvantages of each of the programmability level we identified at the MAC layer. The more programmability is provided, more complexity and room for misconfiguration/conflict with neighboring networks. On the other hand, more intelligent and adaptable proposals can be designed and redesigned quickly with such flexibility. The less programmability provided, more stability and therefore best suited for stable environments and with predictable demands. Furthermore, a non-exhaustive list of research challenges is discussed. We highlighted the main gaps between the state-of-the-art on programmable wireless MAC protocols regarding the design, deployment, control, and management. As far as we analyzed, many solutions focus on the design and deployment of wireless MAC protocols for specific standards and technologies, but there is none that provides enough abstractions and APIs for managing such protocols.

## Acknowledgment

The research leading to these results has received funding from the European Horizon 2020 Programme under grant agreement n$^o$732174 (ORCA project) and the VLAIO SBO project SAMURAI.

## References

[1] I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, and M. Guizani, "Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 10–16, 2017.

[2] B. Jooris, J. Bauwens, P. Ruckebusch, P. D. Valck, C. V. Praet, I. Moerman, and E. D. Poorter, "Taisc: A cross-platform {MAC} protocol compiler and execution engine," *Computer Networks*, vol. 107, Part 2, pp. 315 – 326, 2016, mobile Wireless Networks. [Online]. Available: //www.sciencedirect.com/science/article/pii/S1389128616300974

[3] G. Holland, N. Vaidya, and P. Bahl, "A rate-adaptive mac protocol for multi-hop wireless networks," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '01. New York, NY, USA: ACM, 2001, pp. 236–251. [Online]. Available: http://doi.acm.org/10.1145/381677.381700

[4] T. van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, ser. SenSys '03. New York, NY, USA: ACM, 2003, pp. 171–180. [Online]. Available: http://doi.acm.org/10.1145/958491.958512

[5] M. Maier, M. Reisslein, and A. Wolisz, "A hybrid mac protocol for a metro wdm network using multiple free spectral ranges of an arrayed-waveguide grating," *Comput. Netw.*, vol. 41, no. 4, pp. 407–433, Mar. 2003. [Online]. Available: http://dx.doi.org/10.1016/S1389-1286(02)00399-7

[6] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, pp. 493–506, June 2004.

[7] A. El-Hoiydi and J.-D. Decotignie, "Wisemac: An ultra low power mac protocol for the downlink of infrastructure wireless sensor networks," in *Proceedings of the Ninth International Symposium on Computers and Communications 2004 Volume 2 (ISCC"04) - Volume 02*, ser. ISCC '04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 244–251. [Online]. Available: http://dl.acm.org/citation.cfm?id=1126253.1129805

[8] M. Krishnam, M. Reisslein, and F. H. P. Fitzek, "Analytical framework for simultaneous mac packet transmission (smpt) in a multicode cdma wireless system," *IEEE Transactions on Vehicular Technology*, vol. 53, pp. 223–242, 2004.

[9] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-mac: A short preamble mac protocol for duty-cycled wireless sensor networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, ser. SenSys '06. New York, NY, USA: ACM, 2006, pp. 307–320. [Online]. Available: http://doi.acm.org/10.1145/1182807.1182838

[10] N. F. Timmons and W. G. Scanlon, "An adaptive energy efficient mac protocol for the medical body area network," in *2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace Electronic Systems Technology*, May 2009, pp. 587–593.

[11] N. Ghazisaidi, M. Maier, and M. Reisslein, "Vmp: A mac protocol for epon-based video-dominated fiwi access networks," *IEEE Transactions on Broadcasting*, vol. 58, no. 3, pp. 440–453, Sept 2012.

[12] "Ad-atma: An efficient mac protocol for wireless sensor and ad hoc networks," *Procedia Computer Science*, vol. 52, pp. 484 – 491, June 2015.

[13] P. Wang, J. Ansari, M. Petrova, and P. Mhnen, "Cogmac+: A decentralized {MAC} protocol for opportunistic spectrum access in cognitive wireless networks," *Computer Communications*, vol. 79, pp. 22 – 36, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366415003576

[14] Z. Yang, Z. Shi, and C. Jin, "Sacrb-mac: A high-capacity mac protocol for cognitive radio sensor networks in smart grid," *Sensors*, vol. 16, no. 4, 2016. [Online]. Available: http://www.mdpi.com/1424-8220/16/4/464

[15] A. Farago, A. D. Myers, V. R. Syrotiuk, and G. V. Zaruba, "Meta-mac protocols: automatic combination of mac protocols to optimize performance for unknown conditions," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 9, pp. 1670–1681, Sept 2000.

[16] K. C. Huang, X. Jing, and D. Raychaudhuri, "Mac protocol adaptation in cognitive radio networks: An experimental study," in *2009 Proceedings of the 18th International Conference on Computer Communications and Networks*, Aug 2009, pp. 1–6.

[17] M. Qiao, H. Zhao, S. Wang, and J. Wei, "Mac protocol selection based on machine learning in cognitive radio networks," in *2016 19th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, Nov 2016, pp. 453–458.

[18] H. Peyravi, "Medium access control protocols performance in satellite communications," *IEEE Communications Magazine*, vol. 37, no. 3, pp. 62–71, Mar 1999.

[19] A. C. V. Gummalla and J. O. Limb, "Wireless medium access control protocols," *IEEE Communications Surveys Tutorials*, vol. 3, no. 2, pp. 2–15, Second 2000.

[20] S. Kumar, V. S. Raghavan, and J. Deng, "Medium access control protocols for ad hoc wireless networks: A survey," *Ad Hoc Networks*, vol. 4, no. 3, pp. 326 – 358, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870504000873

[21] I. Demirkol, C. Ersoy, and F. Alagoz, "Mac protocols for wireless sensor networks: a survey," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 115–121, April 2006.

[22] T. V. Krishna and A. Das, "A survey on mac protocols in osa networks," *Computer Networks*, vol. 53, no. 9, pp. 1377 – 1394, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128609000097

[23] C. Cormio and K. R. Chowdhury, "A survey on mac protocols for cognitive radio networks," *Ad Hoc Networks*, vol. 7, no. 7, pp. 1315 – 1329, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870509000043

[24] M. J. Booysen, S. Zeadally, and G. J. van Rooyen, "Survey of media access control protocols for vehicular ad hoc networks," *IET Communications*, vol. 5, no. 11, pp. 1619–1631, July 2011.

[25] S.-L. Tsao and C.-H. Huang, "A survey of energy efficient mac protocols for ieee 802.11 wlan," *Computer Communications*, vol. 34, no. 1, pp. 54 – 67, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S014036641000424X

[26] Y. Z. Zhao, C. Miao, M. Ma, J. B. Zhang, and C. Leung, "A survey and projection on medium access control protocols for wireless sensor networks," *ACM Computing Surveys (CSUR)*, vol. 45, no. 1, pp. 7:1–7:37, Dec. 2012. [Online]. Available: http://doi.acm.org/10.1145/2379776.2379783

[27] F. V. Gallego, J. Alonso-Zarate, C. Verikoukis, and L. Alonso, "A survey on prototyping platforms for the development and experimental evaluation of medium access control protocols," *IEEE Wireless Communications*, vol. 19, no. 1, pp. 74–81, February 2012.

[28] P. Suriyachai, U. Roedig, and A. Scott, "A survey of mac protocols for mission-critical applications in wireless sensor networks," *IEEE Communications Surveys Tutorials*, vol. 14, no. 2, pp. 240–264, Second 2012.

[29] A. Rahim, N. Javaid, M. Aslam, Z. Rahman, U. Qasim, and Z. A. Khan, "A comprehensive survey of mac protocols for wireless body area networks," in *2012 Seventh International Conference on Broadband, Wireless Computing, Communication and Applications*, Nov 2012, pp. 434–439.

[30] P. Ju, W. Song, and D. Zhou, "Survey on cooperative medium access control protocols," *IET Communications*, vol. 7, no. 9, pp. 893–902, June 2013.

[31] K. Chen, M. Ma, E. Cheng, F. Yuan, and W. Su, "A survey on mac protocols for underwater wireless sensor networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1433–1447, Third 2014.

[32] "Medium access control protocols for wireless sensor networks classifications and cross-layering," *Procedia Computer Science*, vol. 65, October.

[33] "A survey on mac protocols for duty-cycled wireless sensor networks," *Procedia Computer Science*, vol. 73, pp. 482 – 489, December 2015.

[34] R. Liao, B. Bellalta, M. Oliver, and Z. Niu, "Mu-mimo mac protocols for wireless local area networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 162–183, Firstquarter 2016.

[35] A. Balobaid, "A survey and comparative study on different energy efficient mac-protocols for wireless sensor networks," in *2016 International Conference on Internet of Things and Applications (IOTA)*, Jan 2016, pp. 321–326.

[36] R. Sadeghi, J. P. Barraca, and R. L. Aguiar, "A survey on cooperative mac protocols in ieee 802.11 wireless networks," *Wireless Personal Communications*, vol. 95, no. 2, pp. 1469–1493, Jul 2017. [Online]. Available: https://doi.org/10.1007/s11277-016-3861-0

[37] A. A. Khan, M. H. Rehmani, and M. Reisslein, "Requirements, design challenges, and review of routing and mac protocols for cr-based smart grid systems," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 206–215, May 2017.

[38] N. Z. binti Zubir, A. F. Ramli, and H. Basarudin, "Optimization of wireless sensor networks mac protocols using machine learning; a survey," in *2017 International Conference on Engineering Technology and Technopreneurship (ICE2T)*, Sept 2017, pp. 1–5.

[39] M. Zareei, A. M. Islam, C. Vargas-Rosales, N. Mansoor, S. Goudarzi, and M. H. Rehmani, "Mobility-aware medium access control protocols for wireless sensor networks: A survey," *Journal of Network and Computer Applications*, vol. 104, pp. 21 – 37, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804517304216

[40] J. Ansari, X. Zhang, A. Achtzehn, M. Petrova, and P. Mahonen, "Decomposable mac framework for highly flexible and adaptable mac realizations," in *2010 IEEE Symposium on New Frontiers in Dynamic Spectrum (DySPAN)*, April 2010, pp. 1–2.

[41] P. D. Sutton, J. Lotze, H. Lahlou, S. A. Fahmy, K. E. Nolan, B. Ozgul, T. W. Rondeau, J. Noguera, and L. E. Doyle, "Iris: an architecture for cognitive radio networking testbeds," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 114–122, Sept 2010.

[42] X. Zhang, J. Ansari, G. Yang, and P. Mhnen, "Trump: Supporting efficient realization of protocols for cognitive radio networks," in *2011 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, May 2011, pp. 476–487.

[43] X. Zhang, J. Ansari, L. M. A. Martinez, N. A. Linio, and P. Mhnen, "Enabling rapid prototyping of reconfigurable mac protocols for wireless sensor networks," in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, April 2013, pp. 47–52.

[44] C.-H. Feng, I. Demirkol, and W. B. Heinzelman, "Ups: Universal protocol stack for emerging wireless networks," *Ad Hoc Networks*, vol. 11, no. 2, pp. 687–700, Mar. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2011.07.013

[45] M. Sha, R. Dor, G. Hackmann, C. Lu, T. S. Kim, and T. Park, "Self-adapting mac layer for wireless sensor networks," in *2013 IEEE 34th Real-Time Systems Symposium*, Dec 2013, pp. 192–201.

[46] S. Yau, L. Ge, P.-C. Hsieh, I.-H. Hou, S. Cui, P. Kumar, A. Ekbal, and N. Kundargi, "Wimac: Rapid implementation platform for user definable mac protocols through separation," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 109–110, Aug. 2015. [Online]. Available: http://doi.acm.org/10.1145/2829988.2790031

[47] E. Demirors, G. Sklivanitis, T. Melodia, and S. N. Batalama, "Rcube: Real-time reconfigurable radio framework with self-optimization capabilities," in *2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, June 2015, pp. 28–36.

[48] G. Nychis, T. Hottelier, Z. Yang, S. Seshan, and P. Steenkiste, "Enabling mac protocol implementations on software-defined radios," in *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, ser. NSDI'09. Berkeley, CA, USA: USENIX Association, 2009, pp. 91–105. [Online]. Available: http://dl.acm.org/citation.cfm?id=1558977.1558984

[49] I. Moerman, S. Giannoulis, E. De Poorter, and X. Jiao, "Softwarization of radio and wireless network," *Newsletter IEEE Software Defined Networks*, pp. 1–5, 2017.

[50] E. Fadel, V. Gungor, L. Nassef, N. Akkari, M. A. Maik, S. Almasri, and I. F. Akyildiz, "A survey on wireless sensor networks for smart grid," *Comput. Commun.*, vol. 71, no. C, pp. 22–33, Nov. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2015.09.006

[51] P. Serrano, A. Banchs, P. Patras, and A. Azcorra, "Optimal configuration of 802.11e edca for real-time and data traffic," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 5, pp. 2511–2528, Jun 2010.

[52] P. Leone, M. Papatriantafilou, E. M. Schiller, and G. Zhu, "Chameleon-mac: Adaptive and self- algorithms for media access control in mobile ad hoc networks," in *Symposium on Self-Stabilizing Systems*. Springer, 2010, pp. 468–488.

[53] F. Gringoli, D. Garlisi, P. Gallo, F. Giuliano, S. Mangione, and I. Tinnirello, "Mac-engine: A new architecture for executing mac algorithms on commodity wifi hardware," in *Proceedings of the 6th ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization*, ser. WiNTECH '11. New York, NY, USA: ACM, 2011, pp. 99–100. [Online]. Available: http://doi.acm.org/10.1145/2030718.2030742

[54] M. Bkassiny, Y. Li, and S. K. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 1136–1159, Third 2013.

[55] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, "The evolution of mac protocols in wireless sensor networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 15, no. 1, pp. 101–120, First 2013.

[56] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Transactions on Computer Systems (TOCS)*, vol. 18, no. 3, pp. 263–297, Aug. 2000. [Online]. Available: http://doi.acm.org/10.1145/354871.354874

[57] M. Neufeld, J. Fifield, C. Doerr, A. Sheth, and D. Grunwald, "Softmac - flexible wireless research platform," in *Fourth Workshop on Hot Topics in Networks (HotNets)*, 2005.

[58] D. G. Messerschmitt, "Rethinking components: From hardware and software to systems," *Proceedings of the IEEE*, vol. 95, no. 7, pp. 1473–1496, July 2007.

[59] A. Khattab, J. Camp, C. Hunter, P. Murphy, A. Sabharwal, and E. W. Knightly, "Warp: A flexible platform for clean-slate wireless medium access protocol design," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 12, no. 1, pp. 56–58, Jan. 2008. [Online]. Available: http://doi.acm.org/10.1145/1374512.1374532

[60] M. C. Ng, K. E. Fleming, M. Vutukuru, S. Gross, Arvind, and H. Balakrishnan, "Airblue: A system for cross-layer wireless protocol development," in *2010 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, Oct 2010, pp. 1–11.

[61] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey," *Computer Networks*, vol. 50, no. 13, pp. 2127 – 2159, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128606001009

[62] Federal Communications Commission, "Et docket no. 03-322," Notice of Proposed Rule Making and Order, Tech. Rep., 2002.

[63] J. Mitola and G. Q. Maguire, "Cognitive radio: making software radios more personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, Aug 1999.

[64] M. Amjad, F. Akhtar, M. H. Rehmani, M. Reisslein, and T. Umer, "Full-duplex communication in cognitive radio networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2158–2191, Fourthquarter 2017.

[65] M. P. Chang and P. R. Prucnal, "A photonic integrated circuit for full duplex spectrum monitoring in cognitive radio," in *Summer Topicals Meeting Series (SUM), 2015*. IEEE, 2015, pp. 105–105.

[66] SDR Forum. (2006) Software defined radio technology for public safety. [Online]. Available: http://www.wirelessinnovation.org/

[67] C. Moy and J. Palicot, "Software radio: a catalyst for wireless innovation," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 24–30, September 2015.

[68] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM Computer Communincation*, vol. v.38, no. 2, pp. 69–74, mar. 2008.

[69] B. A. Fette, *Cognitive radio technology*. Academic Press, 2009.

[70] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. v.103, no. 1, pp. 14–76, jan. 2015.

[71] O. N. Foundation, "SDN Architecture 1.0 Overview," *Project Architecture & Framework*, vol. v.1, no. 1, pp. 1–68, june. 2014, available at: <https://www.opennetworking.org/software-defined-standards/archives/>. Accessed: sept. 27. 2017.

[72] S. Monin, A. Shalimov, and R. Smeliansky, "Chandelle: Smooth and fast wifi roaming with sdn/openflow," in *Proceedings of the 2014 Open Networking Summit Research Track, USENIX, March 3-5*. Santa Clara, USA, 2014.

[73] R. Riggio, M. K. Marina, J. Schulz-Zander, S. Kuklinski, and T. Rasheed, "Programming abstractions for software-defined wireless networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 146–162, June 2015.

[74] J. Schulz-Zander, N. Sarrar, and S. Schmid, "Aeroflux: A near-sighted controller architecture for software-defined wireless networks," in *Presented as part of the Open Networking Summit 2014 (ONS 2014)*. Santa Clara, CA: USENIX, 2014. [Online]. Available: https://www.usenix.org/conference/ons2014/technical-sessions/presentation/shulz-zander

[75] O. Stiti, O. Braham, and G. Pujolle, "Virtual openflow-based sdn wi-fi access point," in *2015 Global Information Infrastructure and Networking Symposium (GIIS)*, Oct 2015, pp. 1–3.

[76] H. Zhou, P. Fan, and J. Li, "Global proportional fair scheduling for networks with multiple base stations," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 4, pp. 1867–1879, May 2011.

[77] K. Langendoen, "Medium access control in wireless sensor networks," *Medium access control in wireless networks*, vol. 2, pp. 535–560, 2008.

[78] T. Shono, Y. Shirato, H. Shiba, K. Uehara, K. Araki, and M. Umehira, "Ieee 802.11 wireless lan implemented on software defined radio with hybrid programmable architecture," *IEEE Transactions on Wireless Communications*, vol. 4, no. 5, pp. 2299–2308, Sept 2005.

[79] S. Hu, Y. D. Yao, and Z. Yang, "Mac protocol identification approach for implement smart cognitive radio," in *2012 IEEE International Conference on Communications (ICC)*, June 2012, pp. 5608–5612.

[80] A. Rao and I. Stoica, "An overlay mac layer for 802.11 networks," in *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '05. New York, NY, USA: ACM, 2005, pp. 135–148. [Online]. Available: http://doi.acm.org/10.1145/1067170.1067185

[81] A. Sharma, M. Tiwari, and H. Zheng, "Madmac: Building a reconfiguration radio testbed using commodity 802.11 hardware," in *2006 1st IEEE Workshop on Networking Technologies for Software Defined Radio Networks*, Sept 2006, pp. 78–83.

[82] A. Sharma and E. M. Belding, "Freemac: Framework for multi-channel mac development on 802.11 hardware," in *Proceedings of the ACM Workshop on Programmable Routers for Extensible Services of Tomorrow*, ser. PRESTO '08. New York, NY, USA: ACM, 2008, pp. 69–74. [Online]. Available: http://doi.acm.org/10.1145/1397718.1397734

[83] M.-H. Lu, P. Steenkiste, and T. Chen, "Flexmac: A wireless protocol development and evaluation platform based on commodity hardware," in *Proceedings of the Third ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization*, ser. WiNTECH '08. New York, NY, USA: ACM, 2008, pp. 105–106. [Online]. Available: http://doi.acm.org/10.1145/1410077.1410102

[84] S. Ullo, A. Vaccaro, and G. Velotto, "Performance analysis of ieee 802.15. 4 based sensor networks for smart grids communications," *Journal of Electrical Engineering: Theory and Application*, vol. 1, no. 3, pp. 129–134, 2010.

[85] B. Yahya and J. Ben-Othman, "Energy efficient and qos aware medium access control for wireless sensor networks," *Concurr. Comput. : Pract. Exper.*, vol. 22, no. 10, pp. 1252–1266, Jul. 2010. [Online]. Available: http://dx.doi.org/10.1002/cpe.v22:10

[86] I. Rhee, A. Warrier, M. Aia, J. Min, and M. Sichitiu, "Z-mac: A hybrid mac for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 3, pp. 511–524, 2008, cited By 403. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-45749132181&doi=10.1109%2fTNET.2007.900704&partnerID=40&md5=36e53f9780e47005cf9968a658e52312

[87] T. Hsieh, K. Lin, and P. Wang, "A hybrid mac protocol for wireless sensor networks," in *2015 IEEE 12th International Conference on Networking, Sensing and Control*, April 2015, pp. 93–98.

[88] G. Miao, J. Zander, K. W. Sung, and S. B. Slimane, *Fundamentals of Mobile Data Networks*. Cambridge University Press, 2016.

[89] L. Kleinrock and F. Tobagi, "Packet switching in radio channels: Part i - carrier sense multiple-access modes and their throughput-delay characteristics," *IEEE Transactions on Communications*, vol. 23, no. 12, pp. 1400–1416, December 1975.

[90] P. De Mil, P. Ruckebusch, J. Hoebeke, I. Moerman, and P. Demeester, "Pluralismac: a generic multi-mac framework for heterogeneous, multiservice wireless networks, applied to smart containers," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, p. 166, May 2012. [Online]. Available: https://doi.org/10.1186/1687-1499-2012-166

[91] C. Doerr, M. Neufeld, J. Fifield, T. Weingart, D. C. Sicker, and D. Grunwald, "Multimac - an adaptive mac framework for dynamic radio networking," in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.*, Nov 2005, pp. 548–555.

[92] D. Raychaudhuri, N. B. Mandayam, J. B. Evans, B. J. Ewy, S. Seshan, and P. Steenkiste, "Cognet: an architectural foundation for experimental cognitive radio networks within the future internet," in *Proceedings of first ACM/IEEE international workshop on Mobility in the evolving internet architecture*. ACM, 2006, pp. 11–16.

[93] W. Hu, X. Li, and H. Yousefi'zadeh, "La-mac: A load adaptive mac protocol for manets," in *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, Nov 2009, pp. 1–6.

[94] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE-advanced for mobile broadband*. Academic press, 2013.

[95] M. Ayhan, Y. Zhao, and H. A. Choi, "Utilizing geometric mean in proportional fair scheduling: Enhanced throughput and fairness in lte dl," in *2016 IEEE Global Communications Conference (GLOBECOM)*, Dec 2016, pp. 1–6.

[96] P. Rost, "Robust and efficient multi-cell cooperation under imperfect csi and limited backhaul," *IEEE Transactions on Wireless Communications*, vol. 12, no. 4, pp. 1910–1922, April 2013.

[97] N. Prasad, M. Arslan, and S. Rangarajan, "Exploiting cell dormancy and load balancing in lte hetnets: Optimizing the proportional fairness utility," in *2014 IEEE International Conference on Communications (ICC)*, June 2014, pp. 1916–1921.

[98] SOLDER. (2018) Fapi 2.0.0 - small cell forum originated lte mac scheduler api. [Online]. Available: http://www.eurecom.fr/~kaltenbe/fapi-2.0/index.html

[99] Small Cell Forum. (2017) Small cell forum - release 10.0. [Online]. Available: https://scf.io/en/documents/082_-_nFAPI_and_FAPI_specifications.php

[100] S. Han, Y. C. Liang, Q. Chen, and B. H. Soong, "Licensed-assisted access for lte in unlicensed spectrum: A mac protocol design," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 10, pp. 2550–2561, Oct 2016.

[101] R. Zhang, M. Wang, L. X. Cai, X. Shen, L. L. Xie, and Y. Cheng, "Modeling and analysis of mac protocol for lte-u co-existing with wi-fi," in *2015 IEEE Global Communications Conference (GLOBECOM)*, Dec 2015, pp. 1–6.

[102] S. Khairy, L. X. Cai, Y. Cheng, Z. Han, and H. Shan, "A hybrid-lbt mac with adaptive sleep for lte laa coexisting with wi-fi over unlicensed band," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Dec 2017, pp. 1–6.

[103] M. Salem and A. Maaref, "A mac solution for distributed coordination of 5g laa operator networks and fair coexistence with wlan in unlicensed spectrum," in *2016 IEEE Wireless Communications and Networking Conference*, April 2016, pp. 1–7.

[104] A. T. Aby, A. Guitton, P. Lafourcade, and M. Misson, "Slack-mac: Adaptive mac protocol for low duty-cycle wireless sensor networks," in *International Conference on Ad Hoc Networks*. Springer, 2015, pp. 69–81.

[105] J. Zhen and V. Rodoplu, "Automated mac protocol generation under dynamic traffic conditions," in *2013 IEEE Global Communications Conference (GLOBECOM)*, Dec 2013, pp. 152–157.

[106] M. S. Gokturk, O. Gurbuz, and M. Erman, "A practical cross layer cooperative mac framework for wsns," *Comput. Netw.*, vol. 98, no. C, pp. 57–71, Apr. 2016. [Online]. Available: https://doi.org/10.1016/j.comnet.2016.01.013

[107] O. B. Akan, O. B. Karli, and O. Ergul, "Cognitive radio sensor networks," *IEEE Network*, vol. 23, no. 4, pp. 34–40, July 2009.

[108] L. H. A. Correia, E. E. Oliveira, D. F. Macedo, P. M. Moura, A. A. F. Loureiro, and J. S. Silva, "A framework for cognitive radio wireless sensor networks," in *2012 IEEE Symposium on Computers and Communications (ISCC)*, July 2012, pp. 000 611–000 616.

[109] Edgewall Software. (2005) The madwifi driver. [Online]. Available: http://madwifi-project.org/

[110] T. M. Project. (2006) Openhal. [Online]. Available: http://madwifi-project.org/wiki/About/OpenHAL?redirectedfrom=OpenHAL

[111] U. N. group. (2009) Openfwwf. [Online]. Available: http://netweb.ing.unibs.it/~openfwwf/

[112] C. Hunter, J. Camp, P. Murphy, A. Sabharwal, and C. Dick, "A flexible framework for wireless medium access protocols," in *2006 Fortieth Asilomar Conference on Signals, Systems and Computers*, Oct 2006, pp. 2046–2050.

[113] K. Klues, G. Hackmann, O. Chipara, and C. Lu, "A component-based architecture for power-efficient media access control in wireless sensor networks," in *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, ser. SenSys '07. New York, NY, USA: ACM, 2007, pp. 59–72. [Online]. Available: http://doi.acm.org/10.1145/1322263.1322270

[114] J. Ansari, X. Zhang, and P. Mhnen, "A compiler assisted approach for component based reconfigurable mac design," in *2011 The 10th IFIP Annual Mediterranean Ad Hoc Networking Workshop*, June 2011, pp. 135–141.

[115] R. Braden, T. Faber, and M. Handley, "From protocol stack to protocol heap: Role-based architecture," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 17–22, Jan. 2003. [Online]. Available: http://doi.acm.org/10.1145/774763.774765

[116] G. Bianchi and A. T. Campbell, “A programmable mac framework for utility-based adaptive quality of service support,” *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 2, pp. 244–255, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/49.824807

[117] H. S. Lichte, S. Valentin, and H. Karl, “Automated development of cooperative mac protocols,” *Mobile Networks and Applications*, vol. 15, no. 6, pp. 769–785, Dec. 2010. [Online]. Available: http://dx.doi.org/10.1007/s11036-009-0210-5

[118] L. Riliskis, E. Osipov, and W. Birk, “A component-based approach to design and analysis of dependable mac protocols for wireless sensor networks,” *Technical report from the Lulea University of Technology, Department of Computer Science and Electrical Engineering*, June 2009.

[119] R. V. Steiner, T. R. Mck, and A. A. Frhlich, “C-mac: A configurable medium access control protocol for sensor networks,” in *2010 IEEE Sensors*, Nov 2010, pp. 845–848.

[120] P. De Mil, B. Jooris, L. Tytgat, J. Hoebeke, I. Moerman, and P. Demeester, “snapmac: A generic mac/phy architecture enabling flexible mac design,” *Ad Hoc Networks*, vol. 17, pp. 37–59, Jun. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2014.01.004

[121] T. Vermeulen, B. Van den Bergh, and S. Pollin, “Demo: A software defined radio platform for rapid cross-layer prototyping,” in *Proceedings of the 2015 Workshop on Software Radio Implementation Forum*, ser. SRIF ’15. New York, NY, USA: ACM, 2015, pp. 1–4. [Online]. Available: http://doi.acm.org/10.1145/2801676.2801683

[122] T. Kazaz, X. Jiao, M. Kulin, and I. Moerman, “Demo: Wiscop - wireless sensor communication prototyping platform,” in *Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks*, ser. EWSN &#8217;17. USA: Junction Publishing, 2017, pp. 246–247. [Online]. Available: http://dl.acm.org/citation.cfm?id=3108009.3108063

[123] T. Parker, G. Halkes, M. Bezemer, and K. Langendoen, “The $\lambda$mac framework: redefining mac protocols for wireless sensor networks,” *Wireless Networks*, vol. 16, no. 7, pp. 2013–2029, Oct 2010. [Online]. Available: https://doi.org/10.1007/s11276-010-0241-7

[124] M. Szczodrak, O. Gnawali, and L. P. Carloni, “Dynamic reconfiguration of wireless sensor networks to support heterogeneous applications,” in *2013 IEEE International Conference on Distributed Computing in Sensor Systems*, May 2013, pp. 52–61.

[125] D. Christmann, T. Braun, M. Engel, and R. Gotzhein, “Bips - a real-time-capable protocol framework for wireless sensor networks,” in *Proceedings of the 6th International Joint Conference on Pervasive and Embedded Computing and Communication Systems*, ser. PECCS 2016. Portugal: SCITEPRESS - Science and Technology Publications, Lda, 2016, pp. 17–27. [Online]. Available: https://doi.org/10.5220/0005938300170027

[126] F. A. Aoudia, M. Gautier, M. Magno, O. Berder, and L. Benini, “A generic framework for modeling mac protocols in wireless sensor networks,” *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1489–1500, June 2017.

[127] J. Ansari, X. Zhang, O. Salikeen, and P. Mhnen, “Enabling flexible medium access design for wireless sensor networks,” in *2011 Eighth International Conference on Wireless On-Demand Network Systems and Services*, Jan 2011, pp. 158–163.

[128] X. Zhang, J. Ansari, and P. Mähönen, “Demo: Runtime mac reconfiguration using a meta-compiler assisted toolchain,” in *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM ’12. New York, NY, USA: ACM, 2012, pp. 277–278. [Online]. Available: http://doi.acm.org/10.1145/2342356.2342408

[129] I. Tinnirello, G. Bianchi, P. Gallo, D. Garlisi, F. Giuliano, and F. Gringoli, “Wireless mac processors: Programming mac protocols on commodity hardware,” in *2012 Proceedings IEEE INFOCOM*, March 2012, pp. 1269–1277.

[130] G. Bianchi, P. Gallo, D. Garlisi, F. Giuliano, F. Gringoli, and I. Tinnirello, “Maclets: Active mac protocols over hard-coded devices,” in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT ’12. New York, NY, USA: ACM, 2012, pp. 229–240. [Online]. Available: http://doi.acm.org/10.1145/2413176.2413203

[131] Free Software Foundation, Inc. (2017) Gnu radio: The free and open software radio ecosystem. [Online]. Available: https://gnuradio.org/

[132] C. R. A. Gonzalez, C. B. Dietrich, S. Sayed, H. I. Volos, J. D. Gaeddert, P. M. Robert, J. H. Reed, and F. E. Kragh, “Open-source sca-based core framework and rapid development tools enable software-defined radio education and research,” *IEEE Communications Magazine*, vol. 47, no. 10, pp. 48–55, October 2009.

[133] K. Tan, H. Liu, J. Zhang, Y. Zhang, J. Fang, and G. M. Voelker, “Sora: high-performance software radio using general-purpose multi-core processors,” *Communications of the ACM*, vol. 54, no. 1, pp. 99–107, 2011.

[134] Vanu Technical Innovation. (2017) Vanu: Proven innovation for wireless coverage challenges. [Online]. Available: http://vanu.com/

[135] J. Ansari, X. Zhang, A. Achtzehn, M. Petrova, and P. Mhnen, “A flexible mac development framework for cognitive radio systems,” in *2011 IEEE Wireless Communications and Networking Conference*, March 2011, pp. 156–161.

[136] P. Gallo, K. Kosek-Szott, S. Szott, and I. Tinnirello, “Sdn@home: A method for controlling future wireless home networks,” *IEEE Communications Magazine*, vol. 54, no. 5, pp. 123–131, May 2016.

[137] M. Jacobsson and C. Orfanidis, “Using software-defined networking principles for wireless sensor networks,” in *Proceedings of the 11th Swedish National Computer Networking Workshop*, 2015. [Online]. Available: http://www.sncnw.se/

[138] H. Moura, G. V. C. Bessa, M. A. M. Vieira, and D. F. Macedo, “Ethanol: Software defined networking for 802.11 wireless networks,” in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, May 2015, pp. 388–396.

[139] I. T. Haque and N. Abu-Ghazaleh, “Wireless software defined networking: A survey and taxonomy,” *IEEE Communications Surveys Tutorials*, vol. 18, no. 4, pp. 2713–2737, Fourthquarter 2016.

[140] NS-3 Consortium. (2018) ns-3. [Online]. Available: https://www.nsnam.org/

[141] OMNet++. (2018) Omnet++ - discrete event simulator. [Online]. Available: https://www.omnetpp.org/

[142] Scalable Network Technologies. (2018) Qualnet network simulator software. [Online]. Available: https://web.scalable-networks.com/qualnet-network-simulator-software

[143] B. L. Titzer, D. K. Lee, and J. Palsberg, “Avrora: scalable sensor network simulation with precise timing,” in *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, April 2005, pp. 477–482.

[144] P. Levis, N. Lee, M. Welsh, and D. Culler, “Tossim: Accurate and scalable simulation of entire tinyos applications,” in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, ser. SenSys ’03. New York, NY, USA: ACM, 2003, pp. 126–137. [Online]. Available: http://doi.acm.org/10.1145/958491.958506

[145] MathWorks. (2018) Matlab for wireless communications: Wireless design starts with matlab. [Online]. Available: https://mathworks.com/solutions/wireless-communications.html

[146] R. Jain, *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. John Wiley & Sons, 1990.

[147] Defense Advanced Research Projects Agency. (2018) Spectrum collaboration challenge (sc2). [Online]. Available: https://spectrumcollaborationchallenge.com/wp-content/uploads/System-Specification-Document.pdf

[148] Fed4Fire. (2018) Fed4Fire: Federation for Fire Plus. [Online]. Available: https://www.fed4fire.eu/

[149] GENI. (2018) GENI: Exploring Networks of the Future. [Online]. Available: http://www.geni.net/

[150] Fibre. (2018) OFELIA Testbed. [Online]. Available: http://www.fibre-ict.eu/index.php/testbeds/ofelia

[151] J. A. Wickboldt, W. P. D. Jesus, P. H. Isolani, C. B. Both, J. Rochol, and L. Z. Granville, “Software-defined networking: management requirements and challenges,” *IEEE Communications Magazine*, vol. 53, no. 1, pp. 278–285, January 2015.

[152] Y. Yi, G. de Veciana, and S. Shakkottai, “Mac scheduling with low overheads by learning neighborhood contention patterns,” *IEEE/ACM Transactions on Networking*, vol. 18, pp. 1637–1650, 2010.

[153] Z. Lan, H. Jiang, and X. Wu, “Decentralized cognitive mac protocol design based on pomdp and q-learning,” in *7th International Conference on Communications and Networking in China*, Aug 2012, pp. 548–551.

[154] S. Amuru, Y. Xiao, M. van der Schaar, and R. M. Buehrer, “To send or not to send - learning mac contention,” in *2015 IEEE Global Communications Conference (GLOBECOM)*, Dec 2015, pp. 1–6.

[155] A. D. Shoaei, M. Derakhshani, S. Parsaeefard, and T. Le-Ngoc, “Learning-based hybrid tdma-csma mac protocol for virtualized 802.11 wlans,” in *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Aug 2015, pp. 1861–1866.

[156] I. Kakalou, G. I. Papadimitriou, P. Nicopolitidis, P. G. Sarigiannidis, and M. S. Obaidat, “A reinforcement learning-based cognitive mac protocol,” in *2015 IEEE International Conference on Communications (ICC)*, June 2015, pp. 5608–5613.

[157] A. Pressas, Z. Sheng, F. Ali, D. Tian, and M. Nekovee, “Contention-based learning mac protocol for broadcast vehicle-to-vehicle communication,” in *2017 IEEE Vehicular Networking Conference (VNC)*, Nov 2017, pp. 263–270.

[158] P. Tiwari, D. K. Meena, and L. S. Pillutla, “Adaptive learning based directional mac protocol for millimeter wave (mmwave) wireless networks,” in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Oct 2017, pp. 1–5.

[159] D. Kotz, C. Newport, R. S. Gray, J. Liu, Y. Yuan, and C. Elliott, “Experimental evaluation of wireless simulation assumptions,” in *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWiM ’04. New York, NY, USA: ACM, 2004, pp. 78–82. [Online]. Available: http://doi.acm.org/10.1145/1023663.1023679

[160] National Science Foundation. (2017) Geni: Exploring networks of the future. [Online]. Available: http://www.geni.net/

[161] Horizon 2020 Research and Innovation Programme. (2017) Fed4fire: Federation for fire plus. [Online]. Available: https://www.fed4fire.eu/

[162] Y. Liao, T. Wang, L. Song, and B. Jiao, “Cooperative spectrum sensing for full-duplex cognitive radio networks,” in *2014 IEEE International Conference on Communication Systems*, Nov 2014, pp. 56–60.

[163] E. Askari and S. Assa, “Full-duplex cognitive radio with packet fragmentation,” in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, April 2014, pp. 1502–1507.

[164] Y. Liao, T. Wang, K. Bian, L. Song, and Z. Han, “Decentralized dynamic spectrum access in full-duplex cognitive radio networks,” in *2015 IEEE International Conference on Communications (ICC)*, June 2015, pp. 7552–7557.

[165] L. T. Tan and L. B. Le, “Distributed mac protocol design for full-duplex cognitive radio networks,” in *2015 IEEE Global Communications Conference (GLOBECOM)*, Dec 2015, pp. 1–6.

[166] S. ElAzzouni, O. Ercetin, A. El-Keyi, T. ElBatt, and M. Nafie, “Full-duplex cooperative cognitive radio networks,” in *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, May 2015, pp. 475–482.

[167] F. Aurzada, M. Lvesque, M. Maier, and M. Reisslein, “Fiwi access networks based on next-generation pon and gigabit-class wlan technologies: A capacity and delay analysis,” *IEEE/ACM Transactions on Networking*, vol. 22, no. 4, pp. 1176–1189, Aug 2014.

[168] A. G. Sarigiannidis, M. Iloridou, P. Nicopolitidis, G. Papadimitriou, F. Pavlidou, P. G. Sarigiannidis, M. D. Louta, and V. Vitsas, “Architectures and bandwidth allocation schemes for hybrid wireless-optical networks,” *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 427–468, Firstquarter 2015.

[169] B. Kantarci and H. T. Mouftah, “Bandwidth distribution solutions for performance enhancement in long-reach passive optical networks,” *IEEE Communications Surveys Tutorials*, vol. 14, no. 3, pp. 714–733, Third 2012.

[170] A. Mercian, M. P. McGarry, and M. Reisslein, “Offline and online multi-thread polling in long-reach pons: A critical evaluation,” *Journal of Lightwave Technology*, vol. 31, no. 12, pp. 2018–2028, June 2013.

[171] J. A. Arokkiam, K. N. Brown, and C. J. Sreenan, “Optimised qos-aware dba mechanisms in xg-pon for upstream traffic in lte backhaul,” in *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, Aug 2016, pp. 361–368.

[172] M. Xu, X. Liu, N. Chand, F. Effenberger, and G. Chang, “Flex-frame timing-critical passive optical networks for delay sensitive mobile and fixed access services,” in *2017 Optical Fiber Communications Conference and Exhibition (OFC)*, March 2017, pp. 1–3.

[173] S. Zhou, X. Liu, F. Effenberger, and J. Chao, “Mobile-pon: A high-efficiency low-latency mobile fronthaul based on functional split and tdm-pon with a unified scheduler,” in *2017 Optical Fiber Communications Conference and Exhibition (OFC)*, March 2017, pp. 1–3.

[174] P. H. Isolani, J. A. Wickboldt, C. B. Both, J. Rochol, and L. Z. Granville, “Interactive monitoring, visualization, and configuration of openflow-based sdn,” in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, May 2015, pp. 207–215.

[175] H. Kim and N. Feamster, “Improving Network Management with Software Defined Networking,” *IEEE Communications Magazine*, vol. v.51, no. 2, pp. 114–119, feb. 2013.

[176] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, “Flowvisor: A network virtualization layer,” *Deutsche Telekom Inc. R&D Lab, Stanford, Nicira Networks, Tech. Rep*, vol. 1, p. 132, 2009.