

# OMNeT++ based cross-layer simulator for content transmission over wireless ad hoc networks

R. Massin, C. Lamy-Bergot, C. J. Le Martret and R. Fracchia

**Abstract**—Flexibility and deployment simplicity are among the numerous advantages of wireless links when compared to standard wired communications. However, challenges do remain high for wireless communications, in particular due to the wireless medium inherent unreliability, and to the desired flexibility, which entails complex protocol procedures. In that context simulation is an important tool to understand and design the protocols that manage the wireless networks. This paper introduces a new simulation framework based on the OMNeT++ simulator [1] whose goal is to enable the study of data and multimedia content transmission over hybrid wired/wireless ad hoc networks, as well as the design of innovative radio access schemes. To achieve this goal, the complete protocol stack from the application to the physical layer is simulated, and the real bits and bytes of the messages transferred on the radio channel are exchanged. To ensure that this framework is reusable and extensible in future studies and projects, a modular software and protocol architecture has been defined. Although still in progress, our work has already provided some valuable results concerning cross layer HARQ/MAC protocol performance and video transmission over the wireless channel, as illustrated by results examples.

**Keywords**—Simulation, wireless, ad hoc, radio access, video, cross-layer.

## I. INTRODUCTION

The recent years have seen the explosion of new wireless networking solutions design and corresponding first deployments in real life. Those systems, taking advantage of the mobile devices and computers ever increasing capabilities, are becoming more and more complex, as can be seen by comparing the recently standardized WiMAX [2] with its WiFi ancestor [3]. One of the reasons for the aforementioned complexity increase is the apparition of cross-layer and cooperative design instead of the previously strictly separated Open System Interconnection (OSI) reference model layers definition. It follows that the use of monolithic C code simulation is no longer well suited to the evaluation of new waveform designs encompassing several research domains and layers. Cross-layer simulation in particular, either considered for intelligent Data Link and PHY co-design [4] [5] or for a more general complete cross-layer design [6], naturally entails the usage of complex simulation systems, which offer the capability to jointly optimize several modules of the complex transmission scheme.

Different works have shown recently, *e.g.* [7], the number and variety of system simulators, as well as their evolution and growing usage. The purpose of our work is thus not

to define or develop a new simulator that would eventually be better attuned to our specific goals, but to develop a generic framework over an existing simulation tool. Indeed, the development of a new simulator would require a complete system design and would raise the difficult question of system maintenance. The viability of such tools, as for instance YANS [8], is dubious if the users community is not strong enough to maintain and let them coherently evolving with the research state-of-the-art. Along the lines of the Mobility Framework [4], we have developed a generic framework built on the OMNeT++ simulation tool only using its most basic and generic features (*e.g.* discrete event scheduling) and simple and easily re-usable C/C++ code implementation. We have made this choice to ensure that this framework completely fits our purpose, *i.e.* the establishment of a generic architecture to simulate transmission of data and multimedia content over hybrid wired/wireless ad hoc networks and the design of innovative radio access schemes. Thanks to this approach, that was used in parallel for the two independent projects DITEMOI [9] and RISC [10] of the French National Research Agency (ANR), the integration of a complete radio access layer with the peer-to-peer oriented video data transmission solution could be merged and jointly exploited.

This paper is organized as follows. Section II presents the design principles established for the simulation chain realization, including the overall protocol architecture and examples of interfaces. Section III details specific realizations done to ensure the feasibility of high fidelity simulations when dealing with cross-layering solutions for wireless ad hoc networks. Section IV presents some examples of the experimental results that can be obtained with this framework, while explaining their interest and possible usage for real systems definition. Finally, conclusions are drawn in Section V.

## II. SIMULATION CHAIN PRINCIPLE AND DESIGN

### A. High-fidelity simulations with OMNeT++

As said before, we consider in this paper the event-driven discrete time simulation tool OMNeT++ as our reference framework. Nevertheless, the approach proposed could be easily extended to other comparable tools such as OPNET [11] or even NS-2 [12].

OMNeT++ has two main characteristics that allow to design the models used to validate network communication protocols in an efficient and cost reasonable way. The first one is its capability to allow an easy definition, through text files, of protocol architecture and information exchange between protocol layers. The second and most important aspect is

that it handles each event in sequence and maintains its own simulated time clock. This clock is only updated at the end of all the treatments associated to the events to be handled at the current time. This property is of great interest in complex systems simulation, as it allows to remove all problems related to real time and synchronization constraints.

Nevertheless, the classical approach of OSI layers separate design, reinforced by the specialization of most researchers on a part of the protocol stack, has led to define frameworks for OMNeT++ that enter in deep details for given layers, while making strong assumptions for the other ones. This is especially true for higher (*e.g.* application) and lower (*e.g.* physical) layers. Particularly, it can be observed that, even if this is evolving [5] [13], standard existing frameworks over OMNeT++ [4] [14] rely on a quite simple abstraction of the physical layer. It is usual to estimate packet error rate after channel decoding by simply drawing a random variable. We believe that building up an efficient cross-layer design enabled data link layer over such a simplified physical layer model leads in practice to questionable results. Indeed, due to the high number of variable parameters such as received power, number of interfering signals, multipath, etc. such a simulator is not adapted to perform detailed and reliable simulations.

Simulations allowing to obtain such fine detail level are conventionally referred to as *High-Fidelity Simulation (HFS)* [7]. The HFS approach is necessary to assess the performance of communication systems designed in a cross-layer way that may encompass the whole protocol stack from the application layer to the physical layer. As a matter of fact, when simulating end-to-end systems that may include wireless relay nodes such as in ad hoc networks, precise and realistic simulation of the numerous mechanisms derived to enhance the link reliability must be performed, in particular to determine how their effects can be combined and what is their joint gain. Indeed, mechanisms such as Hybrid Automatic Repeat Request (HARQ) [15] at the data link layer or TCP at the transport layer share the same goal of combating losses or errors occurring in the network. They both use similar techniques of retransmission, and consequently do not satisfy the independence conditions that would allow to separately add their gains. Furthermore, when considering the transmission of multimedia data [16][17], in particular over unreliable protocols such as UDP or UDP-Lite, the resilience of advanced decoders can be used to overcome remaining errors or losses thanks to concealment. For such applications, where codecs are operating on real data bit strings and can tolerate some errors or small packet losses, modeling the system at high level is limitative. Typically, this approach will lead to obtain only capacity evaluations but no actual quality measurements, as in [18]. The unequal relative importance of different portions of the multimedia bitstream also justifies an HFS approach, to ensure that the measured perceived quality of service (PQoS) at the application level is representative. This is even more critical when mechanisms at higher layers behave according to the information coming from lower layers, such as the packet error rate at transport level, the channel state information [6], the perceived effect of the interference for link adaptation, or spectrum aware routing as in the Cognitive Radio paradigm [19].

In the following, we explain how we have implemented an HFS simulation with OMNeT++, modeling each layer in detail by working at bit level, as described in Section III-A.

### B. Protocol stack organisation

Fig. 1 depicts the overall protocol architecture that is considered in this paper. Our objective being to define a generic ad hoc architecture with multiple nodes that would be used in a global simulation, we have defined two levels of components:

- global components, which allow to drive the simulation and have global knowledge about the whole network. The first one is the connectivity manager which determines, for each node of the network, the nodes in its range. The other one is the radio channel manager in charge of determining channel effects (see Section III-B.2);
- local components, which are the protocol entities within the network node. Each such node may be either a base station, a mobile station or even a data server.

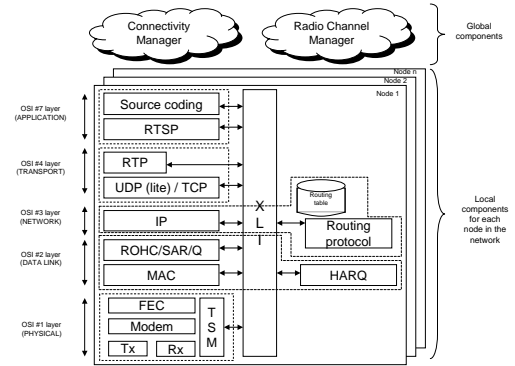


Fig. 1. Overall protocol architecture.

To accurately simulate the transmission of data and multimedia content, the node model covers five of the seven layers of the OSI reference model, having all the same generic format. However, the nodes can be specified separately (*i.e.* given specific protocols capabilities) in particular via the usage of OMNeT++ specific initialization parameters. Typically, multimedia source and receiver nodes will be able to use RTSP requests for RTP encapsulated video data transmission over UDP(-Lite)/IP sockets while data source and receiver nodes may use TCP/IP sockets. Similarly, Robust Header Compression (RoHC) or specific Segmentation and Reassembly (SAR) layers can be selected when needed. Finally, as in [20] a transverse module, denoted XLI for cross-layer interface, has been introduced in each node of the system to allow the joint optimization of several layers.

This approach follows the recent trend showing that the traditional separate decoding of source and channel codes can be efficiently replaced by overall end-to-end optimization [21].

### C. Interfaces

Generic interfaces to exchange information across the protocol stack have been defined.

1) *Data path*: From application to physical layer, each protocol entity receive data messages from their upper interface and forward them to their lower interface. In a way similar to what is done in the Mobility Framework [4] a software API (Application Programming Interface) has been defined to send and receive information on the data path: *sendDown()* and *sendUp()* are used to send data to the lower and upper layers, and *handleLowerMsg()* and *handleUpperMsg()* are used to receive data from the lower and upper layers.

What is particular in the proposed framework is that on the data path, the bits and bytes of the messages are really transmitted, as detailed in section III-A.

2) *Control information exchange*: Cross-layer optimizations are made possible through exchange of signaling information along the protocol stack. In our framework this is done through the XLI, which can be seen as a message switch enabling communication between all layers on the same network host: **ControlMessage** messages are sent to the XLI from one source layer and forwarded by the XLI to the destination layer. All possible destinations are identified by a unique number to allow XLI operation. This scheme allows any protocol layer to use a single *sendControl()* method with a **ControlMessage** as parameter, to transmit signaling information across the local protocol stack. Of course, object inheritance is used and the transferred message is in fact derived from **ControlMessage**, containing the proper information. An example of such derived message is the **QueueCreateNewNeighbourMessage** defined as follows (using OMNeT++ .msg format) :

```
message QueueCreateNewNeighbourMessage
    extends ControlMessage {
    fields:
        int idNeighbour;
        int nbPriorities;
};
```

This message is used to create *nbPriorities* new queues when a new one-hop neighbor (whose address is *idNeighbour*) has been detected. A similar message exists to destroy these queues when the node vanishes from the one-hop neighborhood.

### III. SPECIFIC REALIZATIONS

This section first presents the mechanism and API used to transfer bits between protocol layers and between network nodes. Then the flexible and modular approach followed in our framework is discussed. Finally, two examples of sequence diagrams are reported to illustrate specific realizations.

#### A. Working at bit level

Modeling communications at bit level allows to finely take into account the effect of the wireless channel at all protocol layers. Moreover, this level of detail is required to simulate some communication schemes. For example in the ANR RISC project LDPC error correcting codes are used to improve the quality of wireless communications over a CDMA UWB channel [22]. Since the interference noise perceived on the UWB channel depends on the number of interferers and their

signal level which constantly change during the simulations, it is extremely difficult to assess the performance of such codes without really running the LDPC codec within the simulation. To do this, bit level modeling is needed at the PHY layer. Another example concerns the ANR DITEMOI project. There, video codec resilient to residual errors are studied, implying the need of bit level modeling at the application layer.

This work is not the first one proposing bit level modeling. For example, in MiXiM [5] bit level modeling is possible even if not supported natively. The novelty of our work with respect to previous solutions is rather to formalize bit level modeling all along the path from the top to the bottom layer of the protocol stack and to associate messages generated at the highest level of the protocol stack with their bit content. This is not usually done using OMNeT++: only objects derived from class **cMessage** are exchanged between modules, and especially between the modules modeling the radio channel. Bit level modeling is introduced by associating a memory area to each message allocated at the top of the protocol stack, at the application or user level. This memory area is used to store the bits of the application message, and is big enough to include the headers added by the lower layers as the user message goes down the protocol stack. Also, differently from the usual OMNeT++ paradigm, the same **BytesMsg** message object is transferred through the different protocol layers. The **BytesMsg** sub-class of **cMessage** has three specific members: (1) *memoryArea*, a pointer to the memory area where are stored the bytes of the message; (2) *memoryAreaBytes* that stores the size of the previous area; and (3) *pduBytes* that stores the actual number of bytes of the message. The pointer to the first byte of the message is, as illustrated in Fig. 2,  $memoryArea + (memoryAreaBytes - pduBytes)$ .

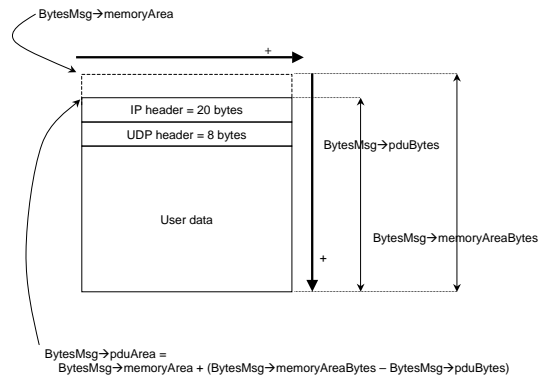


Fig. 2. Example of Application Programming Interface: transmitting real bits.

Upon reception from the upper layer, a protocol entity adds its signaling information in front of the first byte of the received SDU, increases the *pduBytes* member by an amount equal to the size of the added signaling header, and transfers it to the lower layer. Upon reception from the lower layer, a protocol entity reads the header inserted by its homologous entity on the source, decreases the *pduBytes* member by an amount equal to the size of this signaling header, and transfers it to the upper layer. In this scheme, there is no longer one

specific class derived of **cMessage** for each protocol layer, but only one generic **BytesMsg** class. The information usually contained in the data members of the classes derived from **cMessage** are contained in the properly encoded protocol headers.

At the physical layer where modulation and coding are applied, the **BytesMsg** is transformed in a **ComplexSignal** to allow the addition of the radio channel effects on the signal transmitted over the air.

A salient effect of this scheme is to dramatically simplify the duplication of messages sent on the radio channel. In fact, before the transmission over the radio channel, instead of duplicating a long chain of encapsulated messages, a simple **BytesMsg** is duplicated.

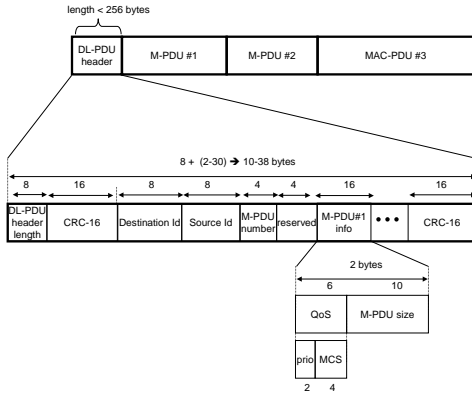


Fig. 3. Bit oriented implementation: example of UDP/IP framing.

Fig. 3 illustrates what is done for the data link PDU (DL-PDU) header: the different fields of this header are clearly defined, which allows for example to analyze the resilience of the signaling protocol when subject to radio channel effects. Such a header is added to the bytes of DL-PDU payload, similarly to the UDP and IP headers as presented in Fig. 2. This DL-PDU payload includes the bits of several MAC-PDU to be sent in one transmission over the radio channel.

### B. A flexible and modular approach to simulation

A major goal of the simulation framework is to enable the design of detailed radio access protocols, radio access encompassing both data link and physical layers. As illustrated in Fig. 1 the data link layer must offer many features such as robust (IP) header compression, segmentation and reassembly, queuing, medium access control, packing/unpacking of PHY-PDU. At the physical layer, services like forward error correction, modulation/demodulation, and amplification before transmission over the air must be implemented. Additionally, radio channel modeling is also needed.

Since the simulation framework is supposed to be used in successive projects, this goal must be attained in a flexible way. Modifications of models source code must be easy and must not touch the main part of the source code. Solving this difficulty involves two winning assets: the definition of modular protocol architecture and the clever use of the object oriented [23] [24] software techniques in order to design a modular software architecture.

1) *Modular simulation architecture*: This section illustrates the modular simulation architecture of the framework for the physical layer. Among the several modules composing the physical layer (Fig. 1), two main entities whose operation is scheduled by one key manager module can be found. The protocol entities implement Forward Error Correcting (FEC) and Modulation and demodulation (Modem). They must be capable of providing the following services: different kinds of error correcting codes for the former, and modulations of different orders for the latter. The selection of the service associated to each message to be sent over the air is made by the Transmission Scheme Manager (TSM) entity. The TSM is like a switch that forwards messages through the physical interface. This architecture is modular in the way that some entities may be skipped and others may be added. For example if no error correcting code capability is necessary then the TSM directly forwards the message received from the data link layer to the Modem. This example corresponds to the introduction of an hybrid ARQ strategy at the data link layer. Instead, when bit encoding is not needed (e.g., when only higher layer issues are investigated) both FEC and Modem layers are removed. A final example would be cooperative relaying [25] which needs an additional module, the Differential Space Time Coding (DSTC) entity that could be inserted between the Modem and the amplifier (Tx) entities.

2) *Modular software architecture*: To ensure good extensibility, a significant effort has been invested in object oriented software modeling. This section illustrates our approach by first presenting the design of the resource allocation function. In this work, this function is run by privileged nodes who manage resource allocation on behalf of all nodes in their one hop neighborhood. These nodes receive radio resource requests from their neighbors, determine which requests will be satisfied, and then send back a response to their neighbors.

The Fig. 4 presents as example the UML class diagram of the **SlotsAllocator** class. Filled in white are object class that compose the core software on which the resource allocation source code is based. Filled in grey are object class derived from class of the core software, that are related to a specific radio resource allocation scheme. In the Time Division Multiple Access (TDMA) scheme, radio resources are time slots (**MacSlot** objects) that follow each other on the time axis, organized in a MAC frame (**MacFrame** object). Input information to a slot allocator are radio resource requests. Objects derived from **SlotsCommand** are associated to each such request, and a slots allocator determines among these requests which ones will be and not be satisfied.

The MAC layer manages a list of allocators, associating each allocator to each resource request depending on the type of the command. For TDMA access, **TdmaSlotsCommand** are associated to a **TdmaSlotsAllocator** allocator. The benefit of this approach is to allow an easy extension of what currently exists: to add Orthogonal Frequency Division Multiple Access (OFDMA) [26] radio access, a new OFDMA allocator would have to be defined, associated with a new OFDMA command.

Fig. 5 presents the UML class diagram of the wireless channel model. A single **RadioChannelManager** object shared between all network nodes has pointers to objects that cal-

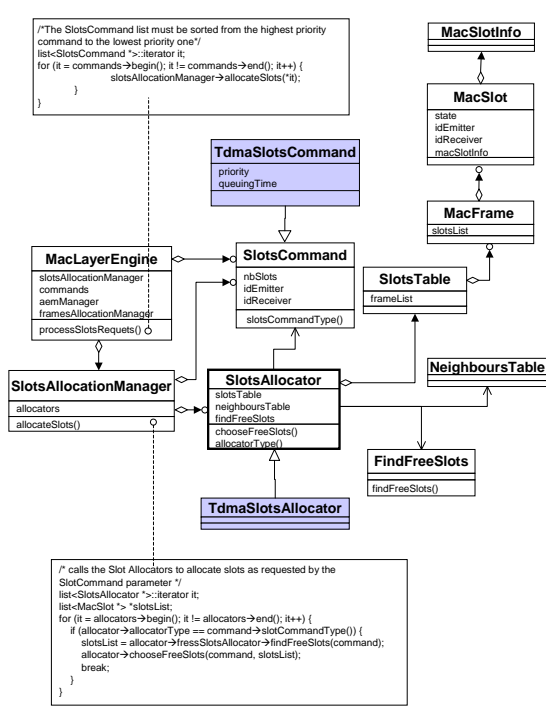


Fig. 4. Resource allocation class diagram.

culate the contribution of the four main parts of the radio channel: fast and slow fading, path loss and additive noise. In the RISC project, specific code was written to model the noise from multi-user interference on a CDMA UWB channel [22] (**UwbInterfererNoise** class derived from the generic **ReceiverNoise** class) as well as ground based shadowing (**GroundBasedShadowing** class derived from the generic **SlowFading** class). To make use of these two models the only source code modification is to create the appropriate objects when initializing the **RadioChannelManager**. The choice of different channel effects is made through the selection of the appropriate models, as in a toolbox.

### C. Message transmission in the radio access

Beyond protocol and software architecture described in the previous sections, we describe in Fig. 6 the sequence diagram of the transmission at the lower part of the radio access layer, from MAC to the radio channel. In phase 1, the MAC sub-layer sends the different MAC-PDU to its lower Packing/Unpacking Manager layer (PUM). Then, from phase 2 to phase 5 the MAC layer transmits a clock signal to the physical layer, triggering a request for data to the PUM entity and the transmission to the physical layer of a DL-PDU using the format illustrated in Fig. 3. The FEC then adds error correcting bits (phase 6), the Modem modulates bits into complex symbols (phase 7) that are forwarded over the radio channel through the Ampli (phases 8 and 9) using a **RadioMsgBB** message sent to all nodes that might be concerned. Phase 2 covers more than one clock signal. Indeed, in some cases it is necessary to transmit information which is not supposed to be corrupted by the radio channel. This is

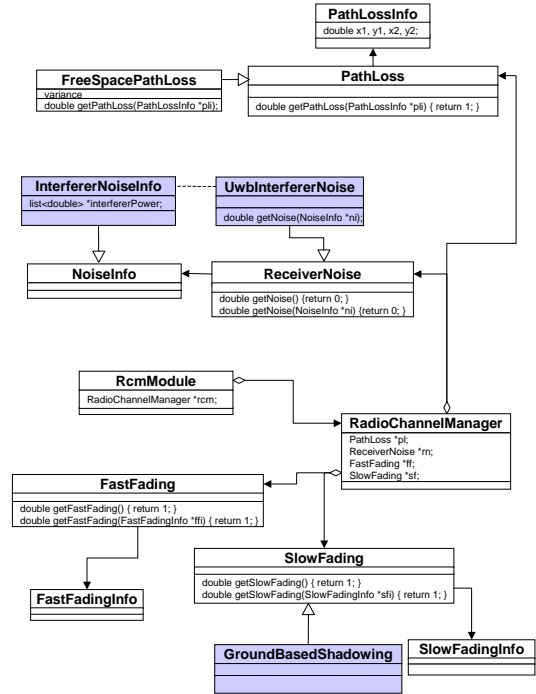


Fig. 5. Radio channel manager class diagram.

still possible in our simulation architecture: this information is forwarded as a usual C++ object with the **RadioMsgBB** message.

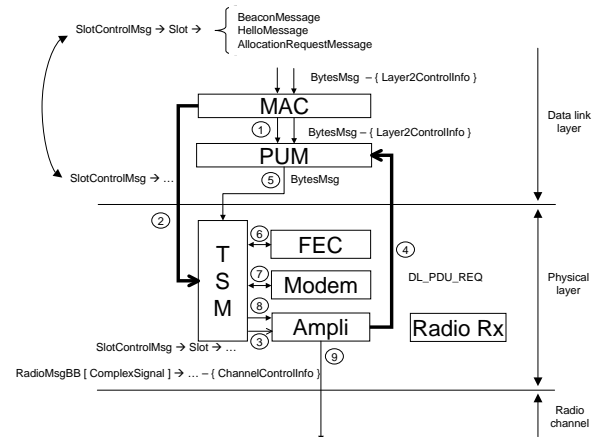


Fig. 6. Radio access sequence diagram at transmission side.

### D. Introducing a real application case: video live transmission

The emulation of higher layers operation is illustrated in this section. Typically, considering the case of video transmission, as can be done in a client/server architecture, real systems use the Real-Time Streaming Protocol (RTSP) [27] which controls the delivery of data with real-time properties. In particular this protocol allows the client and server to negotiate the data request, the transmission conditions and to choose the delivery channel (e.g. UDP, TCP, multicast or unicast, ...). As shown

in Fig. 7, in our system, we are launching the session with the client making a request for a given file (identified by its key) to the server, which then answers favorably if it has the content at its disposal. The session itself can then begin, with the start request containing the setup elements, and the converse reply and acknowledgment messages. Once the session is established, data can be transmitted. When the transmission conditions are too degraded and no data is received, a new start request can then be sent, which could be routed along a new (better) path to resume the transmission.

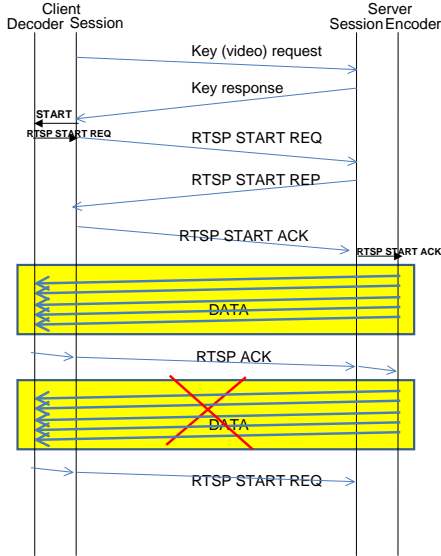


Fig. 7. Video session establishment sequence diagram.

#### IV. SIMULATION EXAMPLES

In this section we present several results that have been obtained with the proposed simulation framework and for which both HFS and bit level modeling were necessary.

1) *Data link HARQ - cross-layer scheme*: Usually, hybrid ARQ is integrated in the physical layer (e.g. WiMAX) for practical implementation reasons. Moving it to the data link layer allows to investigate cross-layer schemes such as the one introduced in [28] for ARQ when the IP packets are fragmented into  $N$  fragments to fit the MAC PDU size. In this cross-layer strategy, the retransmission mechanism at the data link layer exploits information from both the PHY layer and the IP layer. When HARQ is used with soft information in combination with such a cross-layer scheme [29] [30], the HFS is needed. The cross-layer approach considers a global persistence  $C$  for the set of fragments (MAC PDU) coming from the same IP packet, whereas the conventional one considers a per fragment persistence  $P$ , ignoring the fact that it comes from a fragmented IP packet. The cross-layer scheme will be referred to as SDU-Based Strategy (SBS) and the conventional one to as PDU-Based Strategy (PBS).

We have implemented both PBS and SBS in the simulation framework. This enables to assess the performance of the

different approaches using UDP traffic at different layers. Moreover, due to framework structure, once it is implemented for one node, it is easy to simulate multi-hop networks. Fig. 8 illustrates the simulation results of a UDP flow transmission using a TDMA access for one-hop and two-hop networks. The simulation parameters are:  $N = 3$  fragments per IP packet,  $P = 8$  and  $C = 24$ , which ensure a fair comparison since for both strategies the same maximum of retransmission credit per IP packet is allowed. The simulation shows that:

- the SBS outperforms the PBS and confirms the work in [29],
- the PER is larger at the IP layer than at data link layer, which is due to the IP packet fragmentation effect,
- the one-hop transmission performs better than the two-hop one as expected.

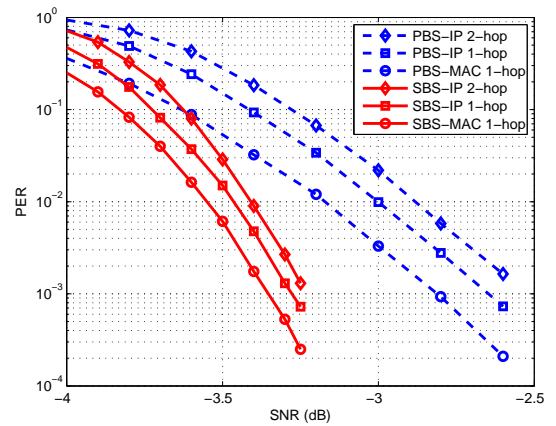


Fig. 8. PBS and SBS performance comparison with IR-HARQ, for  $N = 3$ ,  $P = 8$  and  $C = 24$ .

2) *Data link HARQ - TCP interactions*: A tight integration with the resource allocation scheme is necessary to provide the reverse way needed for the acknowledgment transmission. This leads to non-negligible delays between the data transmission and the reception of the corresponding acknowledgment. To cope with this delay we have introduced a sliding window. Fig. 9 represents the variations of both the HARQ sender window and the TCP congestion window during a 1Mbytes file transfer (with no loss on the wireless channel). The former opens and closes according to the radio resources allocated to the TCP flow. Note that between time 1.3s and 5.3s TCP does not allow the transmission of any data, implying a minimum HARQ sender window. After time 5.3s, the permanent state is reached and alternating congestion avoidance, fast retransmit and fast recovery TCP phases happen periodically, along with wide fluctuations of HARQ sender window.

Fig. 10 details what happens at the HARQ sender window level. Wide variations are visible, due to interactions between the TCP congestion control, the HARQ sliding window and the resource allocation mechanisms.

Note in Fig. 10 that the HARQ window periodically attains its maximum value, 31. In those cases, no more data is transmitted over the wireless channel until acknowledgments have been received, closing the window. When the maximum

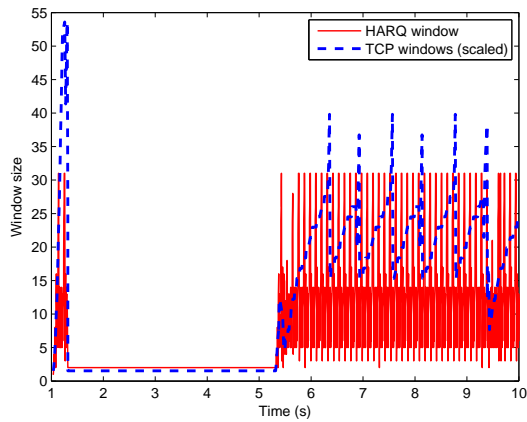


Fig. 9. Joint evolution of the HARQ sender window and TCP congestion window during ad hoc transmission.

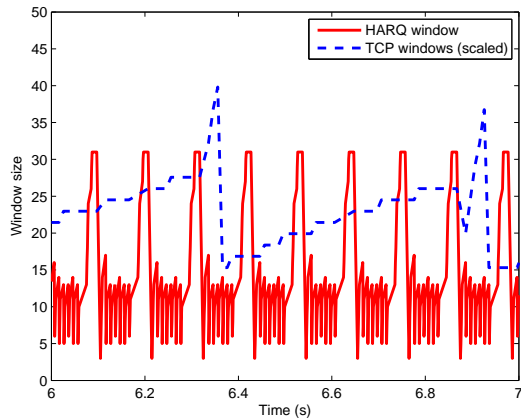


Fig. 10. Zoom of Fig.9.

value is not large enough, this entails a reduction in the TCP throughput, as shown in Table I.

TABLE I  
HARQ SLIDING WINDOW vs. TCP THROUGHPUT.

HARQ maximum window (#PDU)	31	16	8
Normalized TCP throughput	0,942	0,913	0,482

3) *Video transmission*: The usage of HFS from and up to the application level is justified in several cases by the interest and necessity of representing true bit reality. One of these cases corresponds to the introduction of forward error correction codes in the higher layers of the protocol stack, as for instance with RTP-FEC or more generally IETF FecFrame approaches, which aim at overcoming remaining losses or errors at transport or application layers, without requiring a full TCP integrity mechanism. Another case corresponds to the transmission of multimedia data, whose codecs are often resilient to small errors or losses, and for which errors or losses positions are critical to evaluate their real impact on the end-user and measure the PQoS. This is the case considered by the French ANR DITEMOI project, in which error and loss

resilient H.264/AVC decoders were introduced [31], and new strategies for limiting retransmission in video sessions in a multiple users context are being studied.

Fig. 11 and Fig. 12 illustrate the type of results that can be obtained for a video data transmission in the context of a peer-to-peer communication with two interested users. Since the simulator transmits the real bits of an input video, the video can be reconstructed at the receiver side image after image. Comparing the original video with the received one, the Picture Signal to Noise Ratio (PSNR), which is a classical objective measure of the video quality, can be computed. Fig. 11 reports for one user the variation of the PSNR as a function of the frame number of the video sequence in two cases: the first one corresponding to a reference case (solid line) and the second one corresponding to an optimized design (dashed line) where proxies are introduced and allow fine treatments of imperfect packets. Moreover, for a given frame number, the received images in the reference and optimized cases are placed side by side in Fig. 12, showing that the simulator allows not only an objective measurement but also a subjective evaluation of the video quality. The frame, number 145, associated to the pictures in Fig. 12 is identified by a vertical dotted line in Fig. 11.

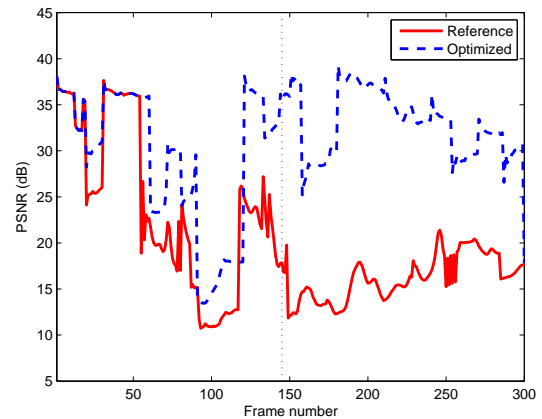


Fig. 11. PSNR comparison vs. frame number between reference and optimized processing.

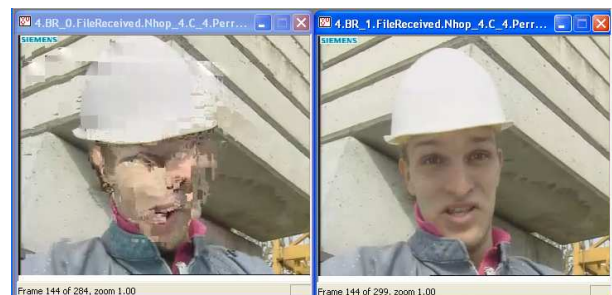


Fig. 12. Comparison of two video sequences quality at frame number 145, left: reference, right: optimized.



## V. CONCLUSIONS

In this paper we presented the main features of a new simulation framework using OMNeT++ for the study of transmission of data and multimedia content over hybrid wired/wireless ad hoc networks and the design of innovative radio access schemes. Details of the API allowing high-fidelity simulations by transmitting bits and bytes over the radio channel have been provided along with simulation results examples in the context of video and TCP/HARQ schemes transmissions. This framework structure is developed in a flexible manner and can encompass various other schemes such as multiple access, OFDMA or cooperative relaying communication techniques. Moreover, this flexibility brought by the modular conception also allows to capitalize on the previous developments by incremental update of the simulation framework and makes it sustainable in time. As future work we plan to compare results obtained through HFS as described in this paper with results coming from less detailed simulations that do not take into account the real bits of the traffic flows transferred on the network.

## VI. ACKNOWLEDGMENTS

This work was partially supported by the French Agence Nationale de la Recherche through projects DITEMOI (ANR-06-TCOM-003) and RISC (ANR-06-TCOM-015).

## REFERENCES

- [1] A. Varga. *OMNeT++ Discrete Event Simulation System.*, <http://www.omnetpp.org/>.
- [2] *IEEE Std 802.16e-2005 and IEEE Std 802.16-2004/Cor 1-2005 IEEE Standard for Local and metropolitan area networks: Part 16: Air Interface for Fixed Broadband Access Systems; Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1*, Feb. 2005.
- [3] J. Schiller, *Mobile Communications*, A. Wesley, Ed., 2003.
- [4] *Mobility framework (MF) for simulating wireless and mobile networks using OMNeT++.*, <http://mobility-fw.sourceforge.net/>.
- [5] A. Kopke *et al.*, “Simulating wireless and mobile networks in omnet++ the mixim vision,” *SIMUTools*, Mar. 2008.
- [6] J. Huusko, J. Vehkaperä, P. Amon, C. Lamy-Bergot, G. Panza, J. Peltola, and M. Martini, “Cross-layer architecture for scalable video transmission in wireless networks,” *Signal Processing: Image Communication*, vol. 22, no. 3, pp. 317–330, Mar. 2007.
- [7] W. Kasch, J. Ward, and J. Andrusenko, “Wireless network modeling and simulation tools for designers and developers,” *IEEE Communications Magazine*, vol. 47, no. 3, Mar. 2009.
- [8] *Yet Another Network Simulator*, Institut National de Recherche en informatique et automatique (INRIA), <http://hal.inria.fr/inria-00078318/fr/>, Oct. 2006.
- [9] *DITEMOI project web site*, <https://www.research-projects.org/projects/DITEMOI>.
- [10] *RISC project web site*, <http://risc.univ-reims.fr/>.
- [11] *OPNET Technologies, Inc.*, <http://www.opnet.com/>.
- [12] *The Network Simulator - ns-2.*, Information Sciences Institute. The University of Southern California, <http://www.isi.edu/nsnam/ns>, July 2006.
- [13] A. Kuntz *et al.*, “Introducing probabilistic radio propagation models in omnet++ mobility framework and cross validation check with ns-2,” *SIMUTools*, Mar. 2008.
- [14] *INET framework (MF) for communication networks simulation for the OMNeT++ environment.*, <http://inet.omnetpp.org/>.
- [15] S. Lin, D. Costello, and M. Miller, “Automatic-repeat-request error-control schemes,” *IEEE Communications Magazine*, vol. 22, no. 12, pp. 5–17, Dec. 1984.
- [16] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.
- [17] H. Schwarz, D. Marpe, and T. Wiegand, “Overview of the scalable video coding extension of the H.264/AVC standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, Sept. 2007.
- [18] Y. Wenpeng, Z. Guangxi, and L. Gan, “Cross-layer schemes for optimization of voip over 802.11e wlan,” in *Proceedings of IEEE GLOBECOM’07*, Nov. 2007, pp. 4883–4887.
- [19] J. Mitola and G. Maquire, “Cognitive radio: making software radios more personal,” *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, Aug. 1999.
- [20] N. Baldo *et al.*, “ns2-miracle: a modular framework for multi-technology and cross-layer support in network simulator 2,” in *NSTools, International Workshop on Network Simulation Tools*, Nantes (France), Oct. 2007.
- [21] P. Duhamel and M. Kieffer, *Joint source-channel decoding - A cross-layer perspective with applications in video broadcasting*. Academic Press Inc., 2009, no. ISBN: 978-0-12-374449-4.
- [22] F. Kharrat-Kammoun, P. Ciblat, and C. L. Martret, “Error probability approximation and codes selection in the presence of multi-user interference for IR-UWB,” in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Cannes (France), Sept. 2008.
- [23] G. Booch, *Object-oriented analysis and design with applications*, 1994.
- [24] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns - Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.
- [25] A. Sendonaris, E. Erkip, and B. Aazhang, “User cooperation diversity part I: Implementation aspects and performance analysis,” *IEEE Transactions on Communications*, vol. 51, no. 11, pp. 1927–1938, Nov. 2003.
- [26] G. Parsaee and A. Yarali, “OFDMA for the 4th generation cellular networks,” *Canadian Conference on Electrical and Computer Engineering*, 2004.
- [27] *Real Time Streaming Protocol (RTSP)*, IETF RFC 2326, <http://www.ietf.org/rfc/rfc2326.txt>, Apr. 1998.
- [28] Y. Choi, S. Choi, and S. Yoon, “MSDU-based ARQ scheme for IP-level performance maximization,” in *IEEE Global Telecommun. Conf.*, St. Louis, Missouri, USA, Nov. 2005, pp. 2495–2499.
- [29] A. L. Duc, C. J. L. Martret, and P. Ciblat, “Packet error rate and efficiency closed-form expressions for cross-layer hybrid arq schemes,” in *Proceedings of IEEE SPAWC’09*, Perugia, Italy, June 2009, pp. 379–383.
- [30] —, “Delay and jitter closed-form expressions for cross-layer hybrid arq schemes,” in *Proceedings of IEEE VTC Fall 2009*, Anchorage, USA, Sept. 2009.
- [31] C. Lamy-Bergot, B. Candillon, B. Pesquet-Popescu, and B. Gadat, “A simple multiple description coding scheme for improved peer-to-peer video distribution over mobile links,” in *Proceedings of the IEEE Picture Coding Symposium (PCS’09)*, Chicago, US, Apr. 2009.