Simulation Platform for Wired/Wireless Networks with Mixed Time and Reliability Requirements

Pablo Gutiérrez Peón*†, Francisco Pozo†, Guillermo Rodriguez-Navas†
*TTTech Computertechnik AG, Vienna, Austria
†School of Innovation, Design and Engineering, Mälardalen University, Västerås, Sweden
Email: pablo.gutierrez-peon@tttech.com, francisco.pozo@mdh.se

The use of computer simulations can help to evaluate the performance of real-time data networks and detect potential issues that compromise the ultimate goal of providing timely and reliable communications. A theoretical analysis over components of the communication system such as the physical layer, medium access, or routing, is too complex or only able to cover pessimistic cases. Further, simulations must include all aspects that are relevant for the performance evaluation. Wireless communication is subject to phenomena like fading, shadowing, etc. which are not always included in simulations. Still, advantages like mobility, flexibility in the deployment or reduced cost and weight might trigger the consideration of wireless communications. Instead of completely replacing wired links, wireless links are often seen as a complement to wired networks, enabling new application domains. Important to mention is that message scheduling is crucial in the design of a real-time communication systems, having the advantage that the behaviour of scheduled message transmissions are always known. Unfortunately, accessing the medium does not guarantee that the message will be delivered. Although absolute guarantees cannot be given, simulation can help to evaluate if sufficient reliability is provided for the transmissions and if the timing requirements can be kept. This work aims at providing a simulation platform to test hybrid wired/wireless networks where both time-triggered real-time and non-real-time traffic travel seemingly over different transmission media.

The user starts defining the networks and parameters that wants to simulate in an input file. The simulation tool can be executed using a *makefile* that will perform a different simulation for every set of parameters given and return if the network has sufficient reliability.

Internally, the simulation platform consists of four main software components: network and traffic generator, traffic scheduler, network simulator and results processing. In Figure 2 we can see how each software component interfaces with others using a file or set of files. In the next paragraphs we go more into detail into the function of each component.

The network and traffic generator creates a network with the topology given by the user and generates the network traffic following the traffic specified. Note that it is possible to specify more than one network and traffic, which will create multiple simulation results at the end of the simulation. The network is composed of four types of devices: wired end-systems, wireless end-systems, switches and access points. Wired end-systems and wireless end-systems are responsible to send and receive traffic and only differ between each other with the capabilities of receiving wireless communication for the wireless end-systems. Switches are responsible of relaying the messages through the network, creating multi-hop paths between end-systems, while access points, connected to a switch, transmit and receives messages in the wireless medium to and from wireless end-systems. The traffic is generated in form of messages, containing information of its size, period and deadline, that can be sent from one end system to any group of end systems of the network.

The traffic scheduler implements a scheduling algorithm capable of generating schedules for large and complex networks. It is developed using a combination of segmentation approaches and a SMT Solver. It takes as input the network and traffic generated before-hand and assigns the transmission times for all the messages in the network while satisfying its constraints.

The network simulator is developed in the OMNeT++ discrete event simulation framework. The basic components of an OMNeT++ simulation are the modules, the pieces where the functionality is implemented. Modules are connected to each other using gates and exchange information using message passing. The modules are partially described using the NED language from OMNeT++, that specifies the parameters, gates and nested modules, while the functionality is programmed in C++. The initialization file is used to give value to the module parameters. The values can be grouped with labels that define different configurations, each specifying a set of values for the module parameters. The four types of network devices are able to dispatch messages in a time-triggered manner (following the link schedule and frame paths files). The devices are based on IEEE 802.3 and IEEE 802.11 with a modified medium access control (MAC) layer. Inside the MAC, queues are FIFO, and traffic of different classes is segregated into different queues. To test the reliability of the wireless links, interference emulating CSMA and jamming at different levels of intensity, size of burst and frequency hopping is envisaged. The simulation can run with a graphical environment or using the command line. The results are provided in three types of files: scalar and vectorial result files and log file. Scalar and vectorial result files retrieve statistics and vectors of raw data, including, e.g., number of packets sent and received or the instants when those packets are sent. The log file

includes entries about relevant events and the time at which they happened.

The results processing tool takes the simulation log file as an input and evaluates whether the logic and timing of the events is according to the MAC protocol design.

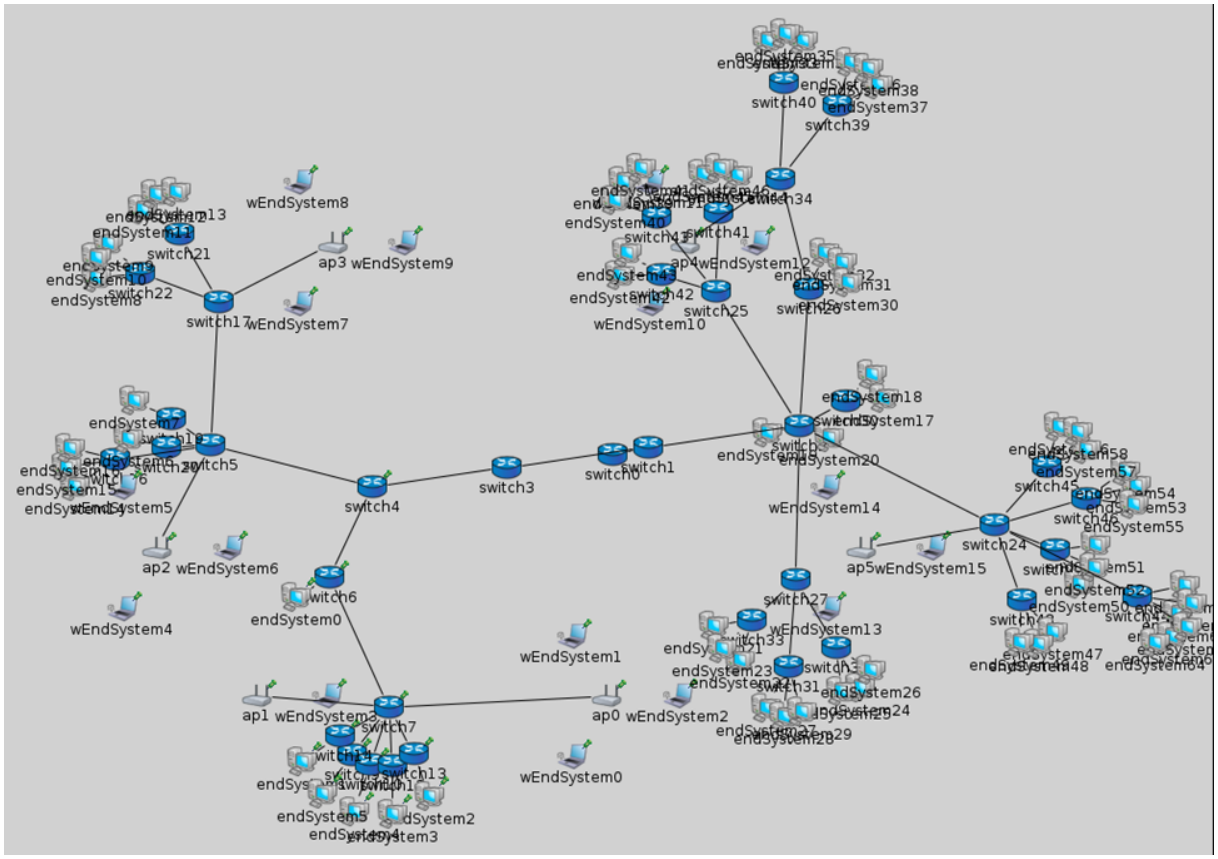

Fig. 1. Example of large network simulated containing 44 switches, 6 access points, 65 wired end systems, 16 wireless end systems, 212 wires dataflow links and 32 wireless dataflow links

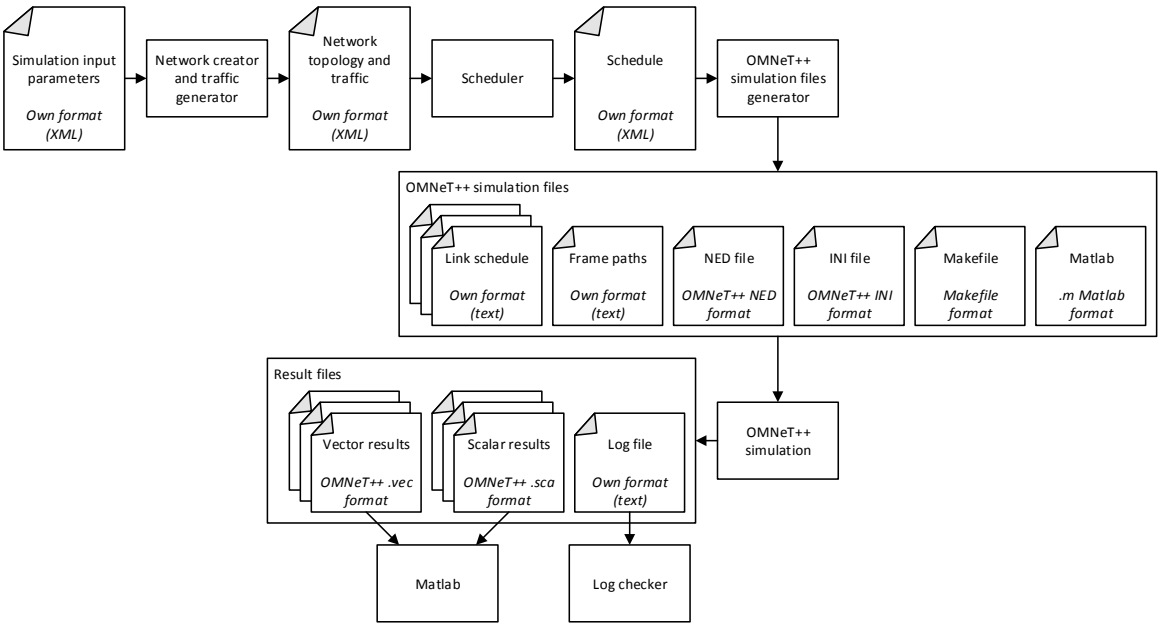

Fig. 2. Graphic representation of the tools flow implemented from the given simulation input up to getting the simulation results