AVENS – A Novel Flying Ad Hoc Network Simulator with Automatic Code Generation for Unmanned Aircraft System

Emerson A. Marconato*, Mariana Rodrigues*, Rayner M. Pires*, Daniel F. Pigatto*, Luiz C.Q. Filho*, Alex S.R. Pinto† and Kalinka R.L.J.C. Branco*

*Institute of Mathematics and Computer Science, University of São Paulo, São Carlos-SP, Brazil
Emails: emerson@icmc.usp.br, rodrigues.mariana@usp.br, rayner@icmc.usp.br, pigatto@icmc.usp.br, querino@usp.br, kalinka@icmc.usp.br

†Department of Control Engineering and Automation, Federal University of Santa Catarina, Blumenau-SC, Brazil
a.r.pinto@ufsc.br

Abstract

The wireless communication has played a significant impact on our daily lives introducing simplicity and making life more comfortable. As a result of faster technological advances in electronics and communications, the development of different types of unmanned aerial vehicles (UAVs) has become possible. Recently, many efforts have been made to develop more efficient inter- and intra-vehicle communication protocols introducing new challenges, e. g. multiple-UAV communication and Flying Ad Hoc Networks (FANETs). However, most of the experiments using real prototypes or systems are not feasible due to the costs and risks involved. Thus, simulating network protocol behavior in FANET scenarios is increasingly required to evaluate the applicability of developed network protocols. Thereby, we have been developing AVENS, a hybrid aerial network simulation framework, which merges LARISSA Architectural Model, X-Plane Flight Simulator and OMNeT++ Discrete Event Simulator. In a proof-of-concept study, we highlighted its advantages. Using AVENS, we can advance in the state-of-the-art concerning performance evaluation of intelligent aerial vehicles and provide means to evaluate the development of protocols, codes and systems more accurately.

1. Introduction

Embedded systems are a highly-integrated hardware and software set that are part of a larger system, usually performing real-time monitoring and control tasks. Technological advances have supplied these systems with greater processing power, memory, and adaptation to different needs, including the capability of communicating with any other device, embedded or not. Embedded systems are considered safety-critical if eventual failure may result in loss of lives or high-value assets [1], [2], [3].

UAV is a typical application of a critical embedded system. Several papers have demonstrated the feasibility of using such vehicles as important tools for performing precision agriculture, reconnaissance, ground, sea and air surveillance, tracking, location and target analysis, traffic monitoring, transport logistics, environment monitoring, etc [4], [5], [6], [7], [8], [9], [10], [11], [12].

When it comes to reducing the cost and time of carrying out a mission, a frequently considered technique is the multiple UAV approach [13], [14], [15]. This approach, however, brings many communication-related issues. In order to address them and given that UAVs have a high mobility level, ad hoc networks are usually applied. For ground vehicles, ad hoc networks are known as VANETs (Vehicular Ad Hoc Networks), while for aerial vehicles the corresponding term is FANETs (Flying Ad Hoc Networks) [15].

Most protocols and wireless technologies are developed assuming 2D communications, low mobility (people) or high mobility (ground vehicles). In terms of connectivity, data delivery, latency, service, mobility, 3D nature, etc., FANETs are significantly different from traditional ad hoc networks (MANETs and VANETs). Communication in FANETs is focused in Airplane-to-Airplane communication (A2A) and Airplane-to-Infrastructure communication (A2I). Thus, the degree of nodes mobility is greater than in MANETs and in VANETs; the topology changes

URI: http://hdl.handle.net/10125/41924
ISBN: 978-0-9981331-0-2

frequently; it needs peer-to-peer communication; the communication range must be greater than other networks; and different strategies of data distribution are necessary.

Both VANETs and FANETs present challenges for performing practical experiments. Chung et. al tested autonomous launch, flight, and landing of 50 UAVs and showcased results in [16]. It is not feasible or easy, in scenarios like this, to vary so many parameters regarding hardware, network and environment, and the results are valid only for that specific set of variables. Therefore, the use of simulations can both speed up and improve the development process, once it makes possible to test a variety of parameters in a easy manner.

There are plenty of published research papers about the advantages of VANETs, e.g. the availability of precise and well-tested models of communication protocols, and the use of a network simulator such as NS-2 [17] or OMNeT++ [18], [19]. Based on recent advances, researchers and developers have been using a simulator that provides not only the network protocol simulation, but also the realistic pattern of mobility models from real vehicles called VEINS [20]. Despite the availability of other simulators for VANETs [21], VEINS is the most used and accepted in recent researches.

On the other hand, researches on FANETs still present results based only on general network simulations. However, these tools do not address the height and orientation imposed by an aerial network, nor their mobility models represent a real FANET [22], [23], [24]. To the best of our knowledge, there is no simulator capable to represent realistic mobility models for FANETs, once current protocols and mobility models assume directed antenna radiation characteristics such as isotropic or omnidirectional radiation. Thus, characteristics concerning the specific aerial scenario are not fully taken into account [25].

Considering the lack of simulators for FANETs and the need to explore UAV characteristics more assiduously, we introduce AVENS, the **A**erial **VE**hicle **N**etwork **S**imulator. It is expected that our implemented tool will provide great contribution to the state-of-the-art improving FANET simulations integrated to a flight simulator certified by the Federal Aviation Administration (FAA). Plus, AVENS is also capable of automatically generate code to OMNeT++ based on LARISSA architecture [26].

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators [27]. It was chosen due to its flexibility on making changes to its modules and specialized frameworks. In our simulator, we implemented changes to INET framework in order to achieve more realistic mobility models. It will be further discussed on Section 3.

The main contributions of this research are:

- A FANET simulator, including not only network aspects, but also aircraft information;
- An automatic OMNeT++ code generation using a model architecture (LARISSA) which guarantees the real modeling and standards information about the aircraft, once safe unmanned avionic operations require a detailed analysis of the communication aspects;
- Integration of a network simulation framework (OMNeT++) with a well-known and certified flight simulator (X-Plane [28]); and
- A platform to simulate different parameters for aircraft and for network aspects as well as to analyze network behavior without spending as much time in field experiments and not requiring physical assets as in [16].

The remainder of this paper is organized as follows: Section 2 presents the LARISSA architecture model describing all the steps for OMNeT++ automatic code generation; Section 3 presents AVENS and the integration between OMNeT++ and X-Plane; Section 4 shows the experimental results and analysis; and finally, conclusions are drawn and suggestions are made for future work in Section 5.

2. LARISSA – Layered Architecture Model for Interconnection of Systems as an Automatic Code Generator

One of the problems in UAS (Unmanned Aircraft System) R&D is the need to make exhaustive tests to guarantee its reliability and accuracy. Flying a UAV or a group of UAVs to test and verify new parameters every time they change is an impracticable and unfeasible situation. The use of simulations before performing field tests allows a cheaper and feasible development of these systems. Aiming to automatize the simulation processes, we are using LARISSA to provide an automatic code generation for OMNeT++.

LARISSA (Figure 1) is a layered architecture model that interconnects unmanned systems [26]. It divides the components of a UAS into aerial and ground segments.

The aerial segment is hierarchically composed by six layers: (i) physical, (ii) distributed RTOS (Real Time Operating System), (iii) system abstraction, (iv) monitoring and control, (v) navigation & services, and (vi) mission layer. The ground segment is divided into

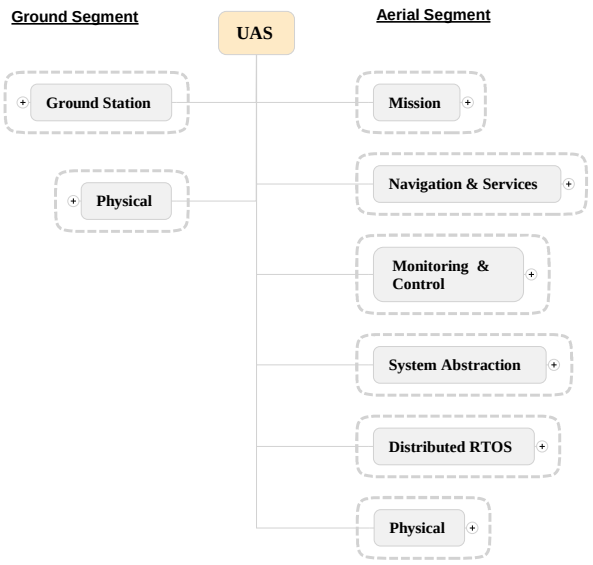

Figure 1. Overview of the LARISSA reference model [26].

the (i) physical layer and (ii) ground control station layer.

These layers can be represented by models that guide the UAS development, specifying how to interconnect the various components such as sensors, control circuits, GPS, payload, communication with the ground control station, and others.

In Information Technology, a layered architecture is used to define the specific responsibilities of each layer and the interconnection among them. Based on an architectural model, hardware manufacturers and/or software designers can develop their products being aware of which exact layer they will interact with in a UAS, what the input and output parameters are, and what type of connection should be used.

LARISSA was specified in UML (Unified Modeling Language), which is a general-purpose language based on object-oriented techniques and is intended for visualization, specification, construction and documentation of software system artifacts [29]. UML can model any system to be implemented with object-oriented languages (Java, C++, Smalltalk, etc.) and, with the help of specialized tools, it is possible to generate source codes in a selected language. The inverse process can also be done [29].

The UML meta-model has building blocks (abstractions, relationships and diagrams) that can be extended to create a profile. A profile is an extension of UML modeling that provides specific details of a domain. UML modeling language has some extension mechanisms to define specific sets of notational elements of a given field, tool or project: the stereotypes, the valued labels (tagged values) and the constraints. These extensions can be defined and grouped in a certain profile. Profiles are intended to provide specific notation elements in certain application areas in order to better represent their particularities and thereby improve the understanding of the specification.

2.1. LARISSA Profile

LARISSA profile was designed using the concepts of UML2 by OMG [30]. The chosen tool for creating this UML profile was the Papyrus component [31] from MDT (Model Development Tools) by Eclipse Foundation. Papyrus provides support for SysML (System Modeling Language), a general-purpose modeling language applied to engineering systems, supporting systems specification, analysis, design, verification and validation, including embedded systems and real-time systems.

LARISSA layers are defined as packets and artifacts at the lowest level, represented by blocks extended from SysML. The properties of the blocks were shown as required and some of the possible values to be assigned were modeled using the Enumeration artifacts representation. An example of this model is shown in Figure 2, in which the *AirToGround* sub-layer was modeled. It belongs to the *Communications* sub-layer, that belongs to the *AvionicsElectronics* sub-layer, which in turn belongs to the *Physical* layer under the Aerial segment.

The *TelecommandTelemetry* sub-layer implements the *RadioModem* stereotype, which is a generalization of the *WirelessDevice* stereotype.

In the following section, we describe a case study, in which two basic components of a UAV are modeled using the LARISSA profile.

2.2. Case Study – Automatic Code Generation to OMNeT++

In this case study, the autopilot and a wireless network interface were modeled in order to perform network communication simulations in OMNeT++. Simulation parameters are generated from the model using the *LARISSA Model2Simulation* code generator.

2.2.1. Modeling using LARISSA Profile

The UML UAV modeling was done on Eclipse using the Papyrus component. A simple autopilot board was modeled, composed by a processor, a main memory, an auxiliary memory, a General Purpose Input/Output (GPIO) interface, and a wireless network

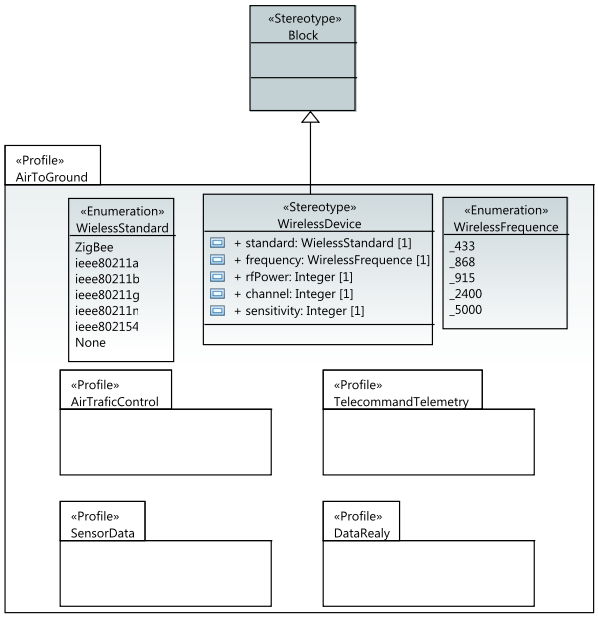

Figure 2. Sub-layer *AirToGround*, from sub-layer *Communications*.

device. It is important to point out that the network communication is the focus of this case study, thus other components of the autopilot are not being taken into account. All parameters were modeled based in a real autopilot named Paparazzi, an open source system of hardware and software [32].

In addition to the main package, two *ControllerBoard* packages were also created. These packages contain the class that forms the autopilot in which the profile `larissa::UAS::Aerial::Physical::AvionicsElectronics::Controller` is applied. For the *AutoPilot*, *Processor*, *MainMemory* and *AuxiliaryMemory* packages, the following stereotypes were applied respectively: *board*, *microcontroller*, *memory* and *memory* again. The SysML Block Definition Diagram regarding the model is shown in Figure 3.

The same method was applied to model the *CommunicationDevice* package, using `larissa::UAS::Aerial::Physical::AvionicsElectronics::Communications::AirToGround::TelecommandTelemetry` profile. In *RadioModem* block, the *radioModem* stereotype was applied. Figure 4 shows this block and some tagged values such as `frequency`, `rfPower` and `sensitivity`, which were used as parameters representing the transmission frequency (433 MHz, 900 MHz, etc.), the transmission power (1 mW, 10 mW, 50 mW, etc.) and the signal attenuation threshold in dBm (-95 dBm). All of them were used to configure the network physical layer in OMNeT++.

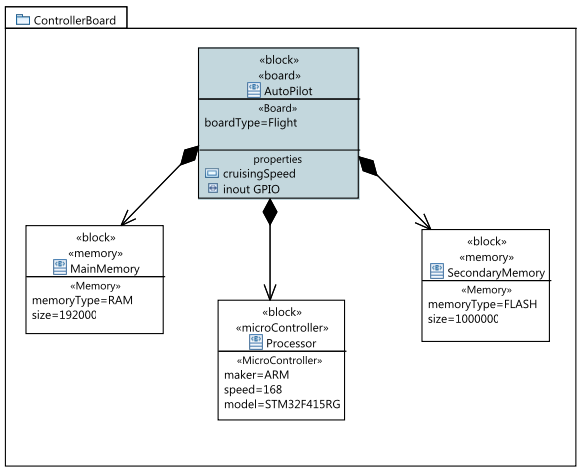

Figure 3. Package *ControllerBoard* with profile *Controller*.

The model is finished by using a SysML Internal Block Diagram, shown in Figure 5, in which we added one *AutoPilot* and one *RadioModem* blocks and linked them by their GPIO ports. GPIO was modeled instead of a serial port because: (i) the Paparazzi autopilot has a dedicated port to link the autopilot and the radio modem and (ii) LARISSA models not only the devices but also the link between them, aiming to standardize UAVs development.

There are several ways to build this type of model, from a simple graphical representation of the UAS' components to a complete model validation. This allows, for instance, to prevent a particular type of autopilot to obtain its model validated in case it does not own a GPS receiver. Another considered possibility is to use Tagged Values to generate parameters to be used in simulators.

Data communication among UAVs and the Ground Control Station can be simulated in OMNeT++. It is possible to change Tagged Values and rerun the

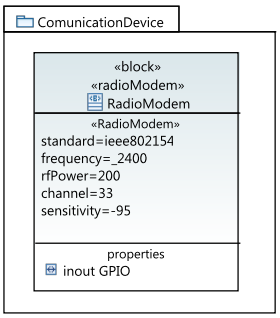

Figure 4. *RadioModem* device.

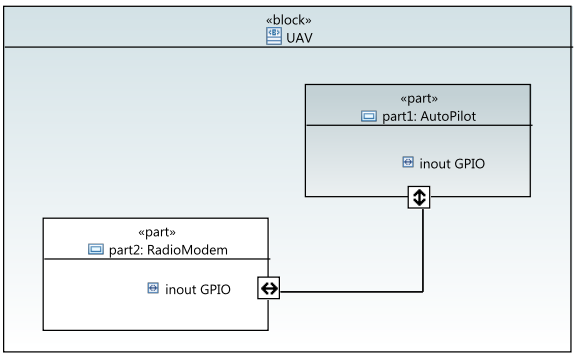

Figure 5. *UAV* – Internal blocks diagram.

simulation until ideal values are reached.

For this purpose, we created *LARISSA Model2Simulation*, a template built with model-to-text code generator Acceleo [33]. This concept allows starting simulation executions from a UML model that uses the LARISSA profile, creating parts of the OMNeT++ settings file, such as power and frequency of a wireless device, the sensitivity of the receptor used and constant or dynamic speed at which objects move in a specific area. Now we are able to transform abstract representation of a UAV into source code for OMNeT++ simulator in few steps.

3. AVENS – Aerial Vehicle Network Simulator

AVENS is part of a major research project concerning the provision of a test bed to simulate UAV flight and control, using different, controlled and changeable configurations. The main intention of AVENS is to provide a simulation test bed for virtual experiments of network coverage and connectivity among UAVs flying in cooperation or sharing the same airspace.

The purpose of AVENS is to offer a platform for mobile ad hoc networks analysis where UAVs are mobile nodes sharing the wireless medium for exchanging messages. The goal is to use a flight simulator for controlling the aerial vehicles and a network simulator for obtaining network measurements such as transmission rate, goodput, RSSI (Received Signal Strength Indication), throughput, package loss, number of retransmission etc.

The two base simulation platforms selected for integration are the X-Plane Flight Simulator and the OMNeT++ Network Simulator, which is integrated with LARISSA in AVENS. As above mentioned, LARISSA provides a reliable and easy way to generate automatic code for OMNeT++ from an abstract UAV modeling. Figure 6 illustrates the AVENS structure.

AVENS provides more accuracy and reliability for the simulation by retrieving mobility model information from X-Plane's navigation pattern and updating UAVs positions on OMNeT++. OMNeT++ is used to simulate the network conditions when the nodes are moving based on positions generated by X-Plane. At the current stage, this mobility model does not consider node connectivity when determining node's position. In future releases, a mobility model that governs the UAVs flight based on the network connectivity will be available for simulation. Such approaches allow the modelling and development of more realistic protocols and mobility models for FANETs.

The integration between simulators is implemented by a plug-in on X-Plane side and a module on OMNeT++ side, both of them responsible for exchanging information through an XML file.

Once configuration files are ready (provided by LARISSA), new OMNeT++ module *fileHandler* must be inserted in user simulation, with its parameters properly configured. Parameter `filename` (string) determines the file in which information will be exchanged; `hostVariableName` (string) determines which variable is used to represent the UAVs; and `numHostVariableName` determines how many UAVs are present in the simulation.

The INET Framework is an open-source OMNeT++ model suite for networks. We have extended this framework with a new mobility model called *Arbitrary Mobility*. This new module extends the *MovingMobilityBase* INET module and has three position parameters in its NED file (`dXPosition`, `dYPosition` and `dZPosition`) that are used to update the node's position, invoking the overloaded method `move()`. Figure 7 shows the inheritance diagram for *Arbitrary Mobility* module.

New *fileHandler* module creates the XML file containing the number of UAVs present in the simulation and all position data, which is initialized with zero. Afterwards, it implements a simple state machine that (i) waits for X-Plane to acknowledge it; (ii) reads the coordinate references for the simulation and (iii) repeatedly updates UAVs position by setting the mobility position parameters according to XML file data until the simulation is finished.

In X-Plane side, a plug-in was developed to gather all planes position information and write them to the XML file. Figure 8 shows the aircraft modelling implemented by the plug-in. All planes are modelled by *Aircraft* class, which stores all position information data. *Aircraft* class is specialized in two classes: *User Aircraft*, which models the user-controlled aircraft in X-Plane, and *MultiAircraft*, which models other aircraft

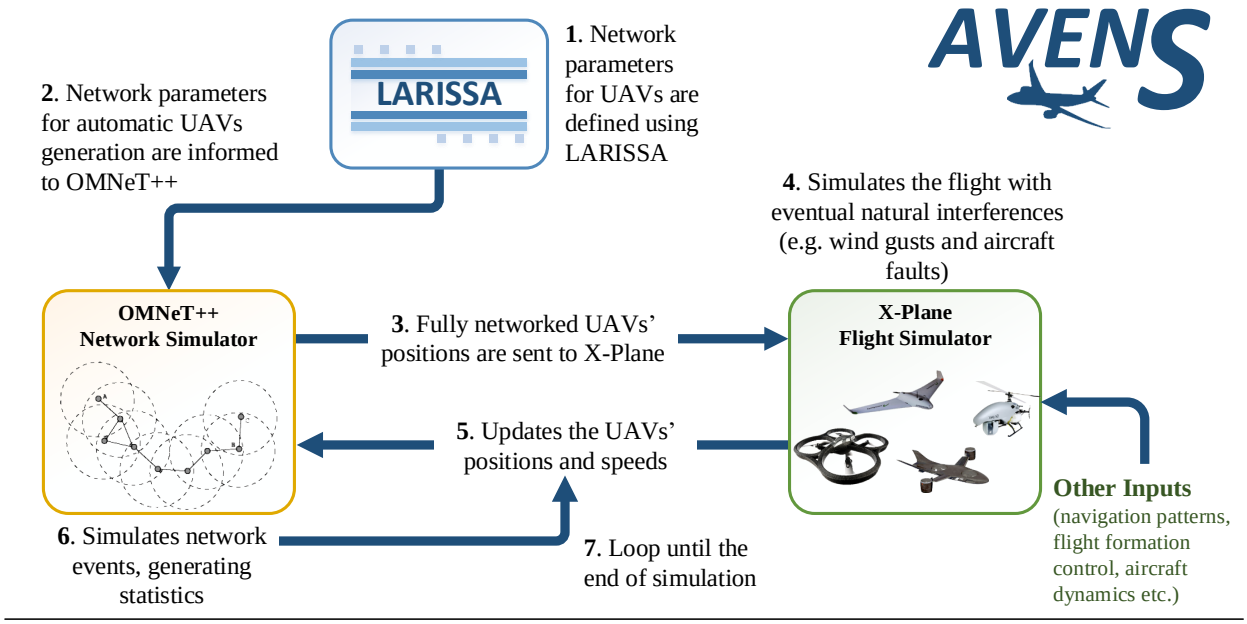

Figure 6. AVENS Structure – Simulators integration.

of the simulation. Since there is only one aircraft directly controlled by user, *UserAircraft* was modeled using Singleton design pattern, with its constructor kept in private and a public static method to retrieve its single instance.

When enabled, X-Plane plug-in initializes a callback that keeps pooling for the XML file existence. Once the file is found, it verifies the number of planes present in the simulation and instantiates the classes accordingly. A second callback is then registered to be executed at a given interval – in which the positioning data for all planes are read (by invoking overloaded method `UpdatePosition()`) and then written in XML file, making them available to OMNeT++.

The communication flow between the OMNeT++ module and the X-Plane plug-in can be seen in Figure 9. First, the `initialize()` method from OMNeT++ module *fileHandler* creates the data sharing XML file with the number of planes present in the simulation. After the initialization, the self-message `WAIT_XPLANE` is sent repeatedly by module until X-Plane initializes the values and acknowledge the file by setting a flag within it. As soon as it happens, a self-message `READ_REFERENCES` is sent once in order to read initial positions and finally self-message `UPDATE_POSITIONS` is sent repeatedly until the simulations ends, refreshing aircraft position in OMNeT++.

When X-Plane plug-in is loaded, it creates a call-

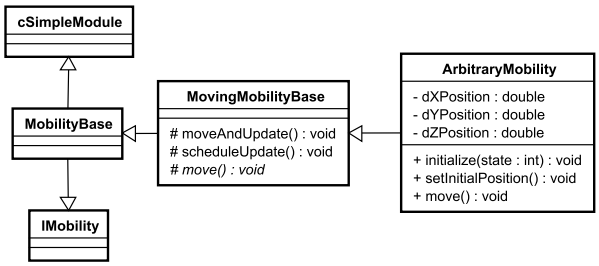

Figure 7. Arbitrary Mobility inheritance diagram.

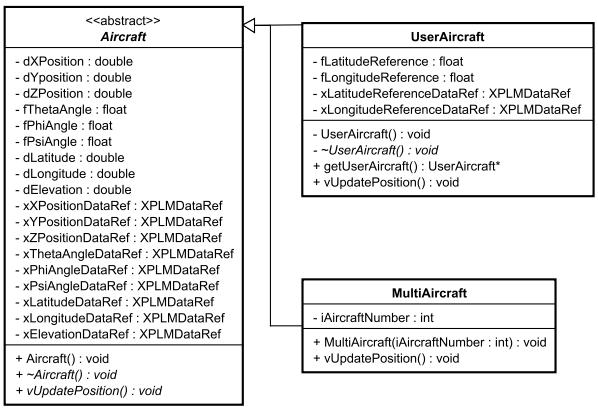

Figure 8. Aircraft modelling in X-Plane plug-in.

back function called every second that checks the file existence and structure. If the file is found and loaded correctly, a second callback is registered and consequently called every time the screen is redrawn, that updates the plane positions in the XML file.

Steps 3 and 4 in Figure 9 represent a loop that lasts as long as the simulation is running.

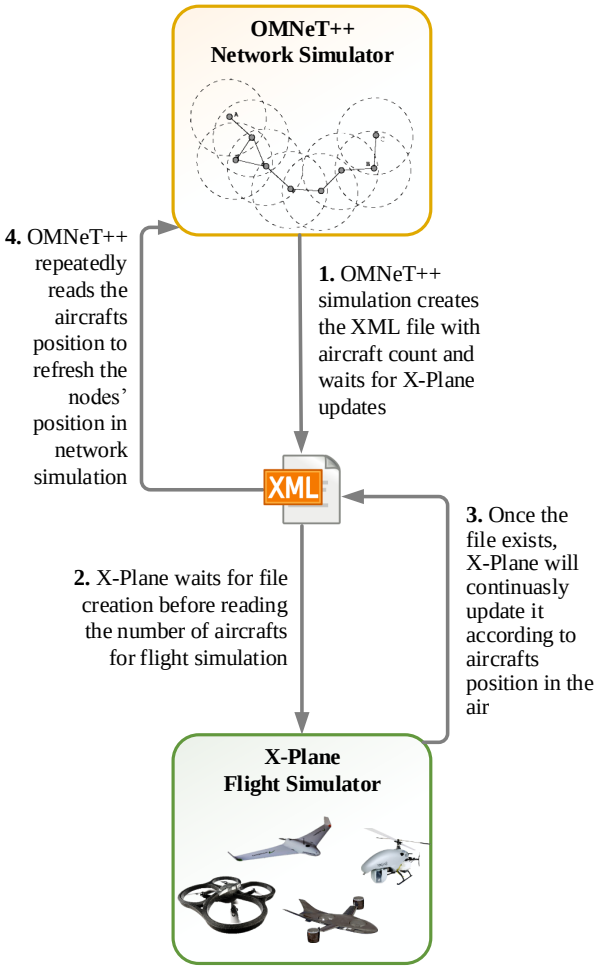

Figure 9. Communication diagram between the OMNeT++ simulation and X-Plane plug-in.

At the current stage, X-Plane is responsible for the mobility pattern, respecting aircraft constraints. OMNeT++ simulates the network states based on the nodes' relative position, which is calculated using the positions informed by X-Plane. The maximum number of UAVs is currently limited by X-Plane, which allows 1 main controlled aircraft plus 19 others. On OMNeT++ side there is no such limitation.

A single instance of X-Plane is run in each simulation, ensuring smaller memory and CPU consumptions. Each aircraft can be configured with different characteristics, allowing the definition of different types of aircraft. It is important to point out that the class model used in X-Plane plug-in and OMNeT++ module can be easily extended to include any new data the user needs exchanged.

4. RESULTS AND DISCUSSIONS

The availability of LARISSA profiles for automatic code generation to OMNeT++, and the forward of position data by X-Plane plug-in made it possible to integrate all needed features into AVENS, our proposed FANET simulator.

Eight simulation scenarios, which parameters are detailed in TABLE 1, were set in AVENS in order to validate it. We have varied the following parameters in AVENS simulations: average sending interval, transmission power, receiver sensitivity, transfer rate, playground dimensions, number of aircraft and aircraft speed. Here, *playground* refers to the simulation area inside the simulators.

LARISSA Model2Simulation uses the MiXiM framework [34], which provides support to wireless devices such as those that implement the IEEE 802.15.4 standard, used in these case studies. On the generation of simulation environment, one of the steps is to create the *omnetpp.ini* file, which contains a number of parameters and instructions that guide the simulation.

The procedure is to insert a line in this file with the expression `"include uavconf.ini"`, that refers to the configuration file generated by *LARISSA Model2Simulation*. Therefore, when the simulation is executed, the parameters that were specified in the modeling and were submitted to the transformation by *LARISSA Model2Simulation* are read and used from *uavconf.ini*.

The *Model2Simulation* obtains specific characteristics of the aircraft from the UML model that uses the LARISSA profile. It integrates such information for code generation, and then allows the model to become reliable for a set of features based on the data obtained directly from the simulator. The model can be improved by repeating the cycle, therefore obtaining an ideal model.

The plug-in inside X-Plane is responsible for converting OMNeT++ files into a set of specification details which will be further used by the flight simulator. The plug-in also guarantees parameters updates in both OMNeT++ and X-Plane during the whole simulation.

The UAVs trajectory can be simulated with AVENS. The latitude, longitude and altitude were retrieved from X-Plane and converted to the navigation frame represented as North-East-Up using MATLAB®. For instance, the resulting trajectory of a single UAV

Table 1. Parameter values of simulation scenarios implemented in AVENS.

Fixed Parameters	Values
Frequency	2.4 GHz
Mobility type	Linear Mobility
Routing algorithm	Wise Route
Standard protocol	802.15.4
Playground size (x)	1.6 km
Playground size (y)	1.6 km
Playground size (y)	1.6 km
Simulation time	3 minutes
Number of GCS	1
Transfer rate	250 Kbps
Sent packets (per node)	170

Variable parameters	Simulation scenarios	Number of UAVs	Transmission power	Receiver sensitivity
Values	1	3	100 mW	-104 dBm
	2			-150 dBm
	3		200 mW	-104 dBm
	4			-150 dBm
	5	5	100 mW	-104 dBm
	6			-150 dBm
	7		200 mW	-104 dBm
	8			-150 dBm

can be seen in Figure 10. Due to AVENS flexibility and robustness, it is possible to visualize the trajectory of a single UAV in a non-formation flight (the case in Figure 10) or even multiple UAVs sharing the same airspace.

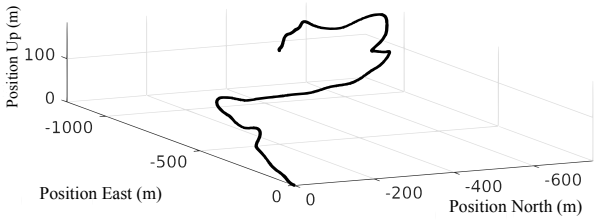

Figure 10. Trajectory represented in the navigation frame North-East-Up.

The charts in Figures 11 and 12 present eight simulation scenarios. The parameters of each simulation are listed in TABLE 1.

These simulations present UAVs flying in cooperation, thus only the y axis is changing throughout the simulation. However, AVENS is not limited by such configuration and might be dealing with different paths in every direction, including x and z axes, if needed. Every simulation was performed for 3 minutes, which is the time needed for a UAV to cover 1.6 km at 8 m/s. Dimensions of the simulation playground are the same for every simulation (1.6 km x 1.6 km x 1.6 km) as well as the transmitted packets per node (170 packets). The Transmission Power varied between 100 and 200 mW, and the Receiver Sensitivity varied between -104 and -150 dBm. Such values were chosen following the specific configuration of an IEEE 802.15.4 board, allowing to obtain an ideal environment.

Figure 11 shows the simulations 1 – 4, with 3 UAVs and 1 GCS (Ground Control Station). Due to the distance among UAVs cooperating to cover the entire playground, power transmission and sensitivity

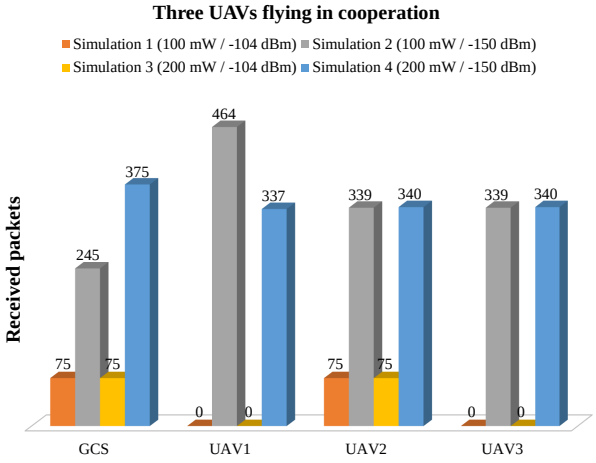

Figure 11. Results of simulation scenarios with 3 UAVs flying in cooperation.

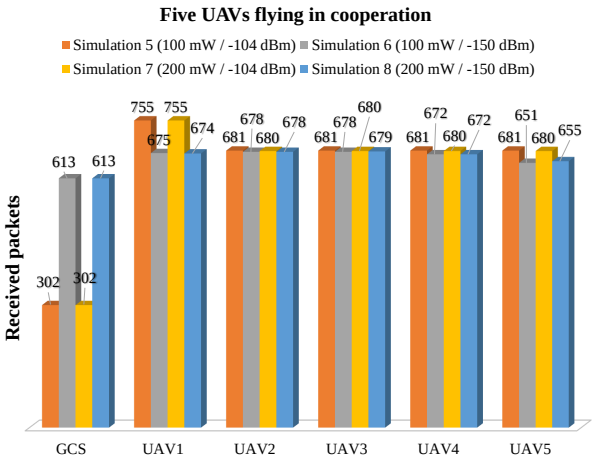

Figure 12. Results of simulation scenarios with 5 UAVs flying in cooperation.

parameters play an important role in results. Nevertheless, we may point out a less significant difference in results from simulations 5 – 8 operating with 5 UAVs in cooperation to cover the same playground size, as seen in Figure 12. Such behavior is expected since the distance among UAVs is smaller whether compared to simulations 1 – 4.

Figures 11 and 12 present results that reinforce the contributions of AVENS to researchers and developers of robotic UAV systems, since it helps preanalysing the behavior of unmanned aircraft systems networks, and specifically measures how transmission would affect the choice for more appropriate hardware appliances. It is important to point out that the main goal of the experimental results shown above is to evaluate AVENS as a simulator for robotic purposes, not network performance or behavior. However, it is intrinsic that one feature cannot be evaluated isolated from another. That is the reason why network parameters were slightly explored in this paper, despite the fact that such attributes are not the main focus. The network behavior can be further explored in new simulations that would eventually be focused on network improvements. Thereby, AVENS is fully capable of measuring advanced network-specific parameters, e. g. transmitted/received packets and packet losses.

AVENS fills the gap shown in [15], providing a multiple UAV simulation tool that can simulate various UAV platforms and network protocols (for FANET design). AVENS can model multiple UAV mobility structures, as well as different network protocols, sensors and others. Updates on AVENS are periodically released in http://www.lsec.icmc.usp.br/avens.

5. Conclusions

This paper introduced AVENS, a new FANET simulator. Using the described methodology, we provide means to evaluate not only developed protocols more accurately, but also the control system, the boards and all sensors implemented in UAVs. The simulator relies on state-of-the-art simulators from both domains, since it incorporates well-known models of aircraft using the X-Plane and a comprehensive selection of models of network protocols from OMNeT++.

The presented proof-of-concept study demonstrated not only the applicability but also the need for multiple UAV and network simulation. Furthermore, the integrated model is able not only to investigate the effects of the network and aircraft, but also to generate code, providing an accurate, flexible and reusable simulator. In the future, security and safety requirements will be integrated to the current AVENS version with the purpose of making simulations even more realistic. Security and safety are requirements of utmost importance in UAS and must be taken into account by an aerial network simulator framework.

Other algorithms are being integrated into X-Plane allowing autonomous flights, trajectories and different control systems to be tested in an integrated flying ad hoc network simulator.

6. References

[1] L. Lazić and D. Velašević, "Applying simulation and design of experiments to the embedded software testing process: Research articles," *Softw. Test. Verif. Reliab.*, vol. 14, no. 4, pp. 257–282, Dec. 2004.

[2] W. Dunn, "Designing safety-critical computer systems," *Computer*, vol. 36, no. 11, pp. 40–46, Nov 2003.

[3] S. P. Kumar, P. S. Ramaiah, and V. Khanaa, "Architectural patterns to design software safety based safety-critical systems," in *Proceedings of the 2011 International Conference on Communication, Computing & Security*, ser. ICCCS '11. New York, NY, USA: ACM, 2011, pp. 620–623.

[4] D. Jacques, "Modeling considerations for wide area search munition effectiveness analysis," in *Simulation Conference, 2002. Proceedings of the Winter*, vol. 1, dec. 2002, pp. 878 – 886 vol.1.

[5] K. D. Mullens, E. B. Pacis, S. B. Stancliff, A. B. Burmeister, T. A. D. Saic, M. H. Bruch, and H. R. Everett, "An automated UAV mission system," *AUVSI Unmanned Systems in International Security 2003 USIS 03*, 2003.

[6] L. Merino, F. Caballero, J. Martínez-de Dios, J. Ferruz, and A. Ollero, "A cooperative perception system for multiple uavs: Application to automatic detection of forest fires," *Journal of Field Robotics*, vol. 23, no. 3-4, pp. 165–184, 2006.

[7] J. L. Drury, L. Riek, and N. Rackliffe, "A decomposition of UAV-related situation awareness," in *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, ser. HRI '06. New York, NY, USA: ACM, 2006, pp. 88–94.

[8] J. Cooper and M. A. Goodrich, "Towards combining UAV and sensor operator roles in UAV-enabled visual search," in *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, ser. HRI '08. New York, NY, USA: ACM, 2008, pp. 351–358.

[9] R. Pitre, X. Li, and D. DelBalzo, "A new performance metric for search and track missions 2: Design and application to uav search," in *FUSION '09. 12th International Conference on Information Fusion, 2009.*, July 2009, pp. 1108–1114.

[10] I. Maza, K. Kondak, M. Bernard, and A. Ollero, "Multi-UAV cooperation and control for load transportation and deployment," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1-4, pp. 417–449, 2010.

[11] O. Trindade Júnior, K. R. L. J. C. Branco, L. d. O. Neris, and L. F. C. Chavier, "Robôs aéreos," in *Robótica Móvel*, R. A. F. Romero, F. S. Osório, D. F. Wolf, and E. P. e. Silva Jr, Eds. LTC, 2012, ch. 16, pp. 461–478.

[12] A. B. Colturato, A. B. Gomes, D. F. Pigatto, D. B. Colturato, A. S. R. Pinto, L. H. C. Branco, E. L. Furtado, and K. R. L. J. C. Branco, "Pattern recognition in thermal images of plants pine using artificial neural networks," in *Engineering Applications of Neural Networks*, ser. Communications in Computer and Information Science, L. Iliadis, H. Papadopoulos, and C. Jayne, Eds. Springer Berlin Heidelberg, 2013, vol. 383, pp. 406–413.

[13] W. Ren, R. Beard, and E. Atkins, "Information consensus in multivehicle cooperative control," *Control Systems, IEEE*, vol. 27, no. 2, pp. 71–82, April 2007.

[14] J. Wang and M. X. 0001, "Integrated optimal formation control of multiple unmanned aerial vehicles." *IEEE Trans. Contr. Sys. Techn.*, vol. 21, no. 5, pp. 1731–1744, 2013.

[15] İlker Bekmezci, O. K. Sahingoz, and Şamil Temel, "Flying ad-hoc networks (FANETs): A survey," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1254 – 1270, 2013.

[16] T. H. Chung, M. R. Clement, M. A. Day, K. D. Jones, D. Davis, and M. Jones, "Live-fly, large-scale field experimentation for large numbers of fixed-wing UAVs," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 1255–1262.

[17] "NS-2 network simulator – Wiki," Available: http://nsnam.isi.edu/nsnam/index.php/Main_Page, 2014, last access: 20th May 2015.

[18] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved ivc analysis," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 1, pp. 3–15, Jan. 2011.

[19] C. Sommer, O. K. Tonguz, and F. Dressler, "Traffic information systems: efficient message dissemination via adaptive beaconing," *Communications Magazine, IEEE*, vol. 49, no. 5, pp. 173–179, May 2011.

[20] "VEINS documentation," Available: http://veins.car2x.org/documentation/, 2014, last access: 20th May 2015.

[21] S. A. Khandelwal and V. S. Gulhane, "Survey the simulation tool of vehicular ad hoc networks," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, Jan. 2012.

[22] E. Yanmaz, R. Kuschnig, and C. Bettstetter, "Achieving air-ground communications in 802.11 networks with three-dimensional aerial mobility," in *INFOCOM, 2013 Proceedings IEEE*, April 2013, pp. 120–124.

[23] E. Yanmaz, S. Hayat, J. Scherer, and C. Bettstetter, "Experimental performance analysis of two-hop aerial 802.11 networks," in *Wireless Communications and Networking Conference (WCNC), 2014 IEEE*, April 2014, pp. 3118–3123.

[24] O. Bouachir, A. Abrassart, F. Garcia, and N. Larrieu, "A mobility model for UAV ad hoc network," in *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, May 2014, pp. 383–388.

[25] A. Boukerche, H. A. B. F. Oliveira, E. F. Nakamura, and A. A. F. Loureiro, "Vehicular ad hoc networks: A new challenge for localization-based systems," *Computer Communications*, vol. 31, no. 12, pp. 2838 – 2849, 2008, mobility Protocols for ITS/VANET.

[26] E. A. Marconato, D. F. Pigatto, K. R. L. J. C. Branco, and L. H. C. Branco, "LARISSA: Layered architecture model for interconnection of systems in UAS," in *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, May 2014, pp. 20–31.

[27] "OMNeT++ user manual," Available: http://www.omnetpp.org/doc/omnetpp/manual/usman.html, 2014, last access: 20th May 2015.

[28] "X-Plane Home Page," Available: http://www.x-plane.com, 2014, last access: 28th nov 2014.

[29] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*, 2nd ed. Addison-Wesley Professional, 2005.

[30] Object Management Group®, "Specification of unified modeling language™(UML®)," Available: http://www.omg.org/spec/UML/, 2011, last access: 20th May 2015.

[31] Eclipse Foundation, "Papyrus," Available: https://eclipse.org/papyrus/, 2015, last access: 20th May 2015.

[32] "Paparazzi: The free autopilot," Available: http://wiki.paparazziuav.org/wiki/Main_Page, 2016, last access: 14th Jun 2016.

[33] Eclipse Foundation, "Acceleo support," Available: https://eclipse.org/acceleo/support/, 2015, last access: 20th May 2015.

[34] M. Project, "MiXiM website," Available: http://mixim.sourceforge.net/, 2011, last access: 20th May 2015.