12th International Conference on MOdeling, Optimization and SIMlation - MOSIM18 - June 27-29 2018
Toulouse - France "The rise of connected systems in industry and services"

Towards the design of a distributed aircraft flight control system connected to simulation components

Jean-Baptiste Chaudron

Complex systems engineering department
ISAE-SUPAERO, University of Toulouse
10 Avenue Edouard Belin, Toulouse, France
jean-baptiste.chaudron@isae-supaero.fr

David Saussié

Department of Electrical Engineering
École Polytechnique de Montréal
C.P. 6079, Montréal (QC), H3C 3A7, Canada
d.saussie@polymtl.ca

ABSTRACT: *The design and the implementation of Flight Control Systems (FCS) still remains a key element of modern avionic systems. During the development process and before flight tests, aeronautical standards require performing piloted simulations. Based on our background in distributed real-time aircraft simulation, we developed a distributed flight control system connected to a simulation environment. This complex distributed architecture is composed of several avionic entities (e.g., primary flight control systems, autopilot) interconnected to simulated components (e.g., aircraft flight dynamics, primary and secondary control surfaces, sensors). Based on a detailed bibliography, we present in this paper the building bricks of this special architecture and the design characteristics of its implementation. In particular, we are introducing architectural fault tolerance aspects and present results to assess the global behaviour of the system.*

KEYWORDS: *Aircraft simulation, Hardware-in-the-loop Simulation, Flight Control Systems, Distributed Systems*

1 INTRODUCTION

Simulation consists in imitating the operation of a real-world system over time, and requires a mathematical model of the system behaviour (J. Banks et al., 2009). It is a well-known design technique, widely used and accepted among research institutes and aerospace industry. Compared to the real flight environment, simulation offers tight control and easy tuning of the different parameters, and one can swiftly investigate many different scenarios and operating conditions. Over the years, the use of simulation has contributed to the improvement of aircraft, such as aerodynamic efficiency or advancements in avionic systems. In particular, to assess the maturity of certain types of equipment or hardware components, the use of a dedicated type of simulation called Hardware-In-the-Loop (HIL) simulation is essential. HIL simulation combines hardware components (to be implanted on the real system) interacting with a simulation environment, reproducing as close as possible the behaviour of the system to be controlled, as depicted in Fig. 1.

The first digital Flight Control Systems (FCS) were studied in the early 70s (J.P. Sutherland, 1968). As an example, in 1972, the F-8C military aircraft became the first aircraft equipped with digital FCS to operate without a mechanical backup system. Since

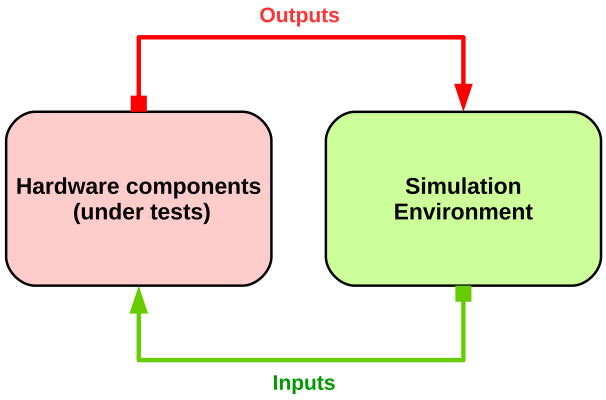

Figure 1 – HIL Simulation illustration

then, the industry and research institutes have been pursuing their effort in this area. Over the years, aircraft FCS have then become more and more complex to ensure multiple functionalities and are currently composed of several redundant components as on-board computers or fly-by-wire units (I.Moir, 2006). The Airbus A320 aircraft is a well-known example and was the first commercial aircraft to contain a full digital complex FCS architecture with autopilot and fly-by-wire functionalities (D. Briere and P. Traverse, 1993). The development of an aircraft FCS typically involves the use of a HIL simulation environment where the flight control system components

have to be tested in certain scenarios prior to the first flight (E.L. Duke, 1989). To allow this, the FCS architecture under study has to be plugged to a relevant and accurate simulation environment modelling the realistic behaviour of the aircraft with mathematical models of flight dynamics, actuators, engines and sensors. The whole system composed by the FCS connected to a dedicated simulation environment is depicted as in Fig. 2.

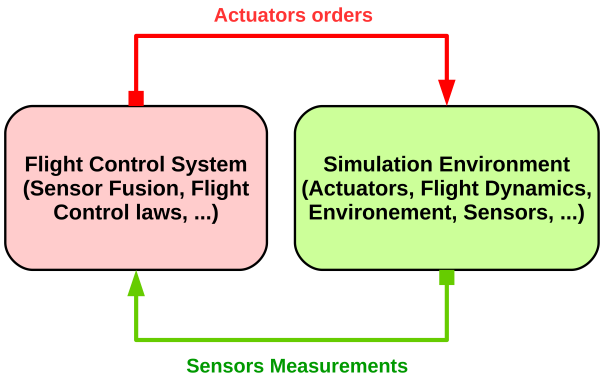

Figure 2 – HIL Simulation for Flight Control System

In the context of the PRISE project[1] (Research Platform for Embedded Systems Engineering), we implemented and tested an aircraft distributed simulation compliant with the High Level Architecture (HLA) standard and composed of several distributed simulators (J-B. Chaudron et al., 2014). Based on this background, we decided to further investigate avionic system architectures and started a new project called Simulation Modules for Avionics Real-TIme Embedded Systems (SMARTIES). Based on literature and assumptions, we designed and implemented from scratch our own aircraft flight control system which is connected to the simulation architecture.

The remainder of this paper introduces the main building blocks of the global system and is structured as follows:

- Section 2 provides a general overview of the whole architecture;
- Section 3 describes the different implementation details and design choices;
- Section 5 outlines some fault-tolerance aspects of our system;
- Section 6 concludes and presents the perspectives for future works.

[1]French acronym for *Plate-forme pour la Recherche en Ingénierie des Systèmes Embarqués*.

2 ARCHITECTURE DESCRIPTION

By analogy with the illustration given in Fig. 2, our HIL simulation architecture, depicted in Fig. 3, is also composed of two main blocks: the simulation environment which is described in Section 2.1 and the distributed flight control system architecture detailed in Section 2.2.

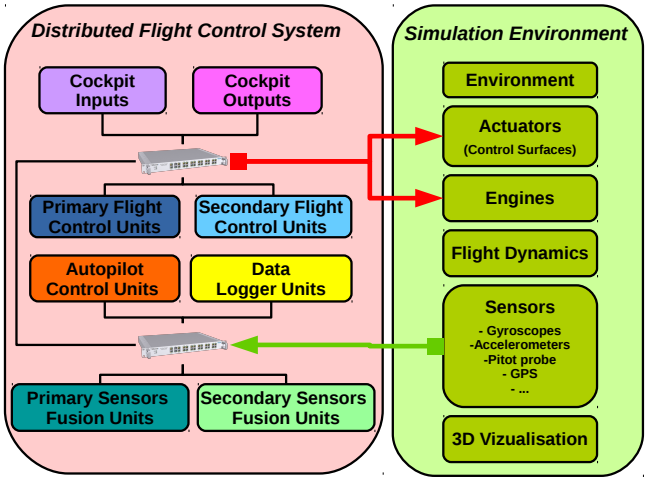

Figure 3 – Our HIL architecture overview

2.1 Simulation Environment

We decided to implement the model of a Boeing 747-100 (C.R. Hanke, 1970) (C.R. Hanke and D.R. Nordwall, 1970). The information provided in these two references are largely sufficient to reproduce faithfully the aircraft behaviour. The documents have been released publicly, and the accuracy and the relevance of the model have been demonstrated by NASA with careful comparisons to temporal simulations provided by Boeing; this motivated our choice to implement this model[2]. To be more explicit, the flight simulation environment is subdivided into several components:

- The ***Environment*** component reproduces realistic weather conditions. It models the US Standard Atmosphere 1976 (NOAA and NASA, 1976) and, depending on the altitude, it calculates the corresponding atmospheric variables such as temperature, pressure or air density. It also integrates different types of wind models, such as wind shears and gusts, as well as turbulence models like Dryden (H.L. Dryden and I.H. Abbott, 1949) and Von Karman ones (T. Von Karman, 1948).
- The ***Actuators*** component gathers all the control surfaces whose deflections change the aerodynamic forces, and thus influence the aircraft

[2]Note that the B747-100 aircraft was not equipped with a current generation FCS system, we are using the flight dynamics model because of its relevance and its accuracy.

motion. We consider here left and right ailerons, left and right inboard/outboard elevators, stabilizer, upper/lower rudder, slats, flaps and gears. Each control surface is modeled by a second-order system with position and rate saturations. Actuators defaults such as delay, bias, and hysteresis phenomena are also considered in our model. As described in Section 4, aircraft actuator systems are redundant (D. R. Ryder, 1973.) and four groups of actuators denoted *Actuators Group (AG)* are considered.

- The ***Engines*** component simulates four high bypass ratio Pratt & Whitney JT9D-3 jet engines whose behaviour changes with the atmospheric conditions (i.e., temperature) and the aircraft Mach number. An accurate model of these jet engines is also described in the B747-100 reference documents (C.R. Hanke, 1970) (C.R. Hanke and D.R. Nordwall, 1970) .

- The ***Flight Dynamics*** component (or Flight Dynamics Model, FDM) is the heart of the simulation model as it computes the equations of motion described, in their simplest form, by a set of twelve first-order ordinary differential equations. Under the action of aerodynamic, gravity, and propulsion forces, the aircraft state evolves accordingly. This part of the simulator is usually the most demanding one depending on the sought fidelity. Fortunately, the reference documents (C.R. Hanke, 1970) (C.R. Hanke and D.R. Nordwall, 1970) provide a sufficiently accurate description of the aerodynamic coefficients.

- The ***Sensors*** component simulates different sensors available in an aircraft. The navigation system is connected to numerous sensors and warning systems and they record data and transmit them to the flight control system. The various sensors included in our simulation are Inertial Navigation System (INS), Global Positioning System (GPS), Pitot Static System, Total Air Temperature (TAT) probe, Angle of Attack (AOA) and Side Slip Angle (SSA). We also consider sensors measuring the engines and control surfaces states. As each type of sensor has its own imperfection, we added phenomena such as delay, bias, drift, and noise. The role of these components is essential as they feed the FCS with the inputs from the simulation environment, the details on the hardware and software implementation will be discussed in Section 3. As described in Section 4, aircraft sensors systems are redundant and we are considering four groups of Sensors noted Sensors Group (SG).

- The ***3D Visualization*** part is the display of actual aircraft position, attitude, control surfaces states and so on, in existing virtual 3D environments such as FlightGear (`http://www.flightgear.org/`) or X-plane (`http://www.x-plane.com/`). We can instantiate many instances of these flight simulators to be able to reproduce an accurate graphical environment to the user. Currently, we are using several instances to get a full cockpit windows display with multiples screens and, also, we are using some other screens to visualize the aircraft from an external point of view.

2.2 Flight Control System

As mentioned in Section 1, our distributed flight control system tends to reproduce the behaviour of a real avionic network. Aircraft FCS are very complex systems composed by several entities with different purposes such as the autopilot functions or sensors fusion. For the time being, the different entities considered in our distributed FCS architecture are as follows:

- The ***Cockpit Inputs*** components are in charge of relaying the pilot/co-pilot inputs coming from the cockpit elements such as ailerons/elevators-stick, throttle-stick, flaps-stick, slats-stick and gears/stick devices in order to send commands to the aircraft actuators and engines (generally through the implemented fly-by-wire system). We also have a Flight Control Unit (FCU) interface which can be used by the pilot/co-pilot to set the parameters for the autopilot units. For example, a reference heading or a reference altitude can be selected via this interface and, if the autopilot in enabled, the FCS system will move the aircraft in this predefined reference state.

- The ***Cockpit Outputs elements*** are mainly pilot/co-pilot graphical interfaces such as in a real aircraft cockpit. As of now, our architecture Primary Flight Display (PFD), Navigation Display (ND) and Electronic Centralized Aircraft Monitor (ECAM). These interfaces provide visual and sound cues to the pilot/co-pilot from the information sent by the different units of the distributed system.

- The ***Primary Flight Control Units (PFCU)*** units implement the different algorithms to control the aircraft and to maintain it in a safe flight envelope (following a set of configurable rules) depending on the inputs from the pilot on the side-stick or throttle stick or the autopilot (via APCU). There are four PFCU units integrated in the system to ensure some fault-tolerant properties as described in Section 4. These flight control laws are composed of stability and control augmentation system (CSAS) loops that allow the pilot to control the normal acceleration in the longitudinal motion, and the roll rate in the

lateral motion to make the aircraft turn. Moreover, one usually finds protection system of the angle of attack and speed to avoid stall.

- The ***Secondary Flight Control Units (SFCU)*** also implement some algorithms to control the aircraft but with degraded features. There are two SFCU units which are only used in case of failure of the PFCU units, the reconfiguration of the architecture from PFCU to SFCU is not discussed in this paper.

- The ***Autopilot Units Control Units (APCU)*** units are in charge of the autopilot calculation based on the inputs from the pilot or the co-pilot on dedicated interfaces to select for example a reference altitude or a reference heading. There are two APCU units: one unit computes the autopilot functions based on the inputs from the pilot and the other one computes it based on the inputs from the co-pilot.

- The ***Data Logger Units (DLU)*** units are in charge of gathering and storing all the data coming from the whole FCS System. There is only one unit currently in our architecture.

- The ***Primary Sensor Fusion Units (PSFU)*** implement some algorithms for sensor fusion based on the inputs from the simulated sensors. These PFSU units, similar to Air-Data and Inertial Reference Units (ADIRU) (M.L. Sheffels, 1992), must provide the most possible accurate values to the control units for each measured parameter. Currently, the parameters that are estimated by our algorithms are attitude (Euler angles and quaternions), position (latitude, longitude and altitude), airspeed parameter (such as true airspeed), angle of attack, and so on. An Extended Kalman Filter (EKF), which is an estimation algorithm for non-linear systems, derived from the original Kalman filter (R.E. Kalman, 1960), has been implemented as a baseline for the algorithm of these units. There are four PSFU units, as described in Section 4, and we are illustrating the correctness of our algorithm by presenting some results in Section 4.3.

- The ***Secondary Sensor Fusion Units (SSFU)*** also implement Kalman filter-based algorithms to fuse the data from sensors but with degraded features. As for SFCU units, there are two SSFU units which are only used in case of failure of the PSFU units; however the reconfiguration of the architecture from PSFU to SSFU is not discussed in this paper.

3 DESIGN AND IMPLEMENTATION

3.1 Simulation Interfaces and Execution

In an HIL environment, the way to interface the hardware under test and the simulation is paramount because it must use the same interfaces as the ones found in the real system. In our case, we are considering a FCS system based on real-time Ethernet network to connect all the FCS computing devices (see Section 3.2). To allow the connection between the FCS system and the simulation environment, we used some specific devices called *Rackmount*, which is very similar to a computer with multiple Ethernet interfaces. As illustrated in Fig. 4, the Rackmount device is running Engines, Actuators and Sensors components, which are sending and receiving data from the real-time Ethernet network (via Real-Time Ethernet Switches i.e. RTES).

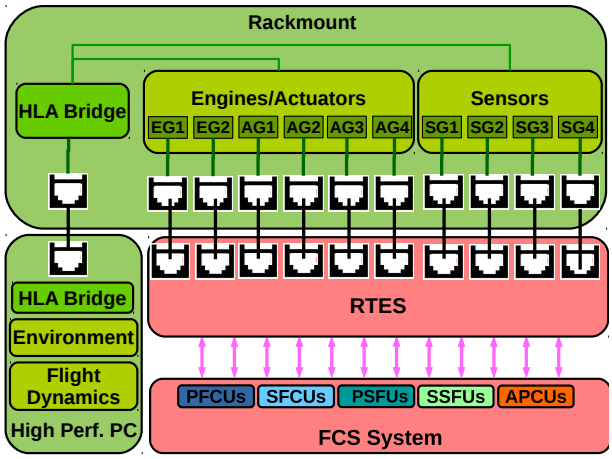

Figure 4 – Simulation/FCS interfaces illustration

The simulation part must respect the real-time constraints inherent to the components under test (such as the operating frequencies) and, therefore, our simulation environment has to be executed fairly quickly to behave according to real-time requirements imposed by the FCS system. To ensure a good execution of the simulation and allow enough computing power to each simulation component, we distributed the flight dynamics and the environment simulation components to another high performance machine. As depicted in Fig. 4, this distribution has been done using IEEE HLA standard and especially the open-source middleware CERTI (E. Noulard et al, 2009). In particular, HLA and CERTI mechanisms provide time management algorithms, guaranteeing a consistent global logical time throughout the whole simulation, which are one of the main benefits of this simulation standard. Our simulation models are based on Ordinary Differential Equations (ODE) and the use of HLA time management ensures the proper schedule of the computations and the communications between the distributed ODE based simulators (J-B.

Chaudron et al., 2016). Thus, the global simulation provides results that will always be fair and relevant according to simulation model semantics.

From the software point of view, the whole simulation environment has been home made and written in C/C++. The whole simulation environment is configurable through a set of eXtensible Markup Language (XML) files. As an example, the user can select the numerical integration method and the integration step to solve the corresponding ODE.

3.2 FCS Interfaces and Execution

As described in Fig. 5, the current version of our FCS system is composed of 2 standard devices (Cockpit Inputs/Ouputs and Data Logger) and 14 embedded devices (2 APCUs, 4 PFCUs, 2 SFCUs, 4 PSFUs and 2 SSFUs). The whole system use Commercial Off-The-Shelf (COTS) technologies and in particular an Ethernet based network. Thus, we use Real-Time Ethernet Switches (RTES), which are compliant with standard Ethernet protocols, as well as the ARINC 664 part 7 standard (A664p.7) (ARINC Industry Activities, 2006) and Time-Triggered Ethernet (TTEthernet) standard (SAE International, 2011).

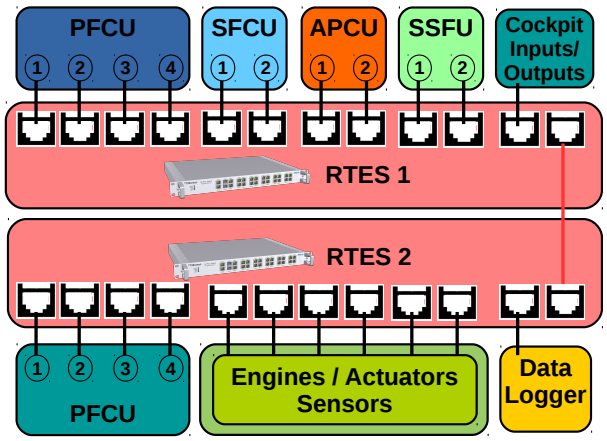

Figure 5 – FCS network illustration

At the processing unit level, the Cockpit and the Data Logger units are running on high performance machine with a Linux Fedora 24 (64 bits) installed. For the embedded units, we decided to use Beagleboards X15 cards (Gerald Coley, 2016) on which are installed Xenomai 3 real-time OS (Jan Kiszka, 2016) (one card per unit). The current version of the communication stack works with standard UDP Ethernet protocol and an home-made A664p.7 software-based compliant stack (light version).

As for the simulation, the whole software design and implementation for each has been home made and written in C/C++. The APCU, PFCU and PSFU units have been designed to work at 50Hz or 100Hz and the SFCU and SSFU units have been designed to work at 20Hz or 50Hz for the moment. The real-time scheduling analysis for the different algorithms implemented in the FCS devices is out of the scope of this paper.

3.3 Discussion

We have described, in this section, the implementation details of our global architecture (the simulation part and the FCS part). We have made some choices based on our current knowledge and experience. As an example, we decided to use the HLA standard, instead of other existing standard as the Functional Mockup Interface (FMI), because we have a complete knowledge of HLA CERTI software which we are using for years now. Also, the HLA synchronization mechanisms using time management services (which are not available in other simulation standards) are extremely useful to ensure proper schedule and consistency of the simulation (J-B. Chaudron et al., 2016). The whole software (for both simulation and FCS components) has been done per us and, therefore, we have a complete control on the implementation and the tuning of every part.

As a recall, the purpose of this position paper is to introduce our new project SMARTIES, we have described what we have done and how it is related with the different research areas. The overall architecture is complex and has required the combination of a lot of concepts, algorithms and details and it is not possible to describe every aspects within one paper. Thus, we have decided to illustrate, in the next Section 4, the fault-tolerance and redundancy features implanted and tested on our platform.

4 FAULT TOLERANCE AND REDUNDANCY

4.1 Quadruplex design

The occurrence of a fault in an aircraft flight control can lead to catastrophic events with significant costs, both economically and in terms of human life. Therefore, such systems are said to be safety critical systems and have to be designed to ensure fault tolerant properties to continue operating properly even if a fault occurs on one of its components (computers systems, actuators, sensors, etc.). The fault tolerance analysis is a wide research domain (D.K. PradhanK. 1986) which goes way beyond the scope of this paper. We introduce in this section the impact on the design of our architecture.

In avionic systems, the fault tolerance and reconfiguration properties can be considered in the implemented algorithms themselves (adaptive control/command laws) as well as in the global system architecture design (computers, backbone network,

etc.). From the architectural point of view, fault-tolerance and reconfiguration techniques are usually built upon per the usage of redundant, specific and heterogeneous computer based components. In aircraft flight control systems, fault tolerance is usually achieved through redundancy, i.e., the computing units are duplicated, triplicated or quadruplicated (R.P.G. Collinson, 2003). The redundancy ensures that the overall cumulative failure probability required for the whole global system (example, 10^{-9} failures/hour for civilian transport aircraft) is smaller than the failure probability of one of its standalone sub-system (example, 10^{-7} failures/hour for one processing unit). Therefore, if one of the redundant components fails, the architecture can ignore the faulty component, or switch to a spare one depending on the implanted fault detection and recovery mechanism.

Currently, the primary level of our distributed FCS is based on a quadruplex architecture (D.B. Mulcare et al., 1988) with multi-cast communication flows as illustrated in Fig. 6.

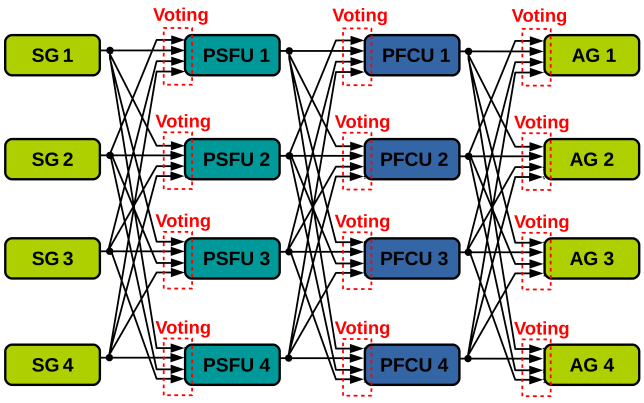

Figure 6 – Communication flows illustration

4.2 Voting algorithms

In addition, due to redundancy and communication flows (see Fig. 6), a voting technique must be integrated in order to elect the *best*[3] value to be processed. There are many varieties of voting techniques, which mainly rely on and depend on the architecture of the system. As detailed in Section 3.2, the current version of our demonstrator only implements asynchronous communication protocols (standard UDP and light version of A664 p7) and, therefore, we have implemented a voting algorithm that is compliant with this specificity (G.J. Davis and Ames Research Center, 1987).

Currently, the implemented algorithm is a well known fault-tolerant median algorithm which is composed per 4 successive phases and works as follows.

[3]With respect to the logic implanted in the voting algorithm

1. ***Collection phase:*** Collect the incoming data from each channel (collection duration is a configurable amount of time).
2. ***Consistency phase:*** Compare the incoming values two by two and ensure that they remain within a pre-defined interval.
3. ***Selection phase:*** Sort the incoming data and remove the smallest and the biggest inputs (only if 3 or 4 inputs are available).
4. ***Fusion phase:*** The remaining values are averaged (in case of 4 inputs) or the median value is kept as it is (in case of 3 inputs).

4.3 Simulation Results

We will illustrate in this sub-section the behaviour of the algorithm implanted in the redundant PSFU units. We have considered a scenario with 100Hz IMU sensors with 5% of white noise (for gyroscope and accelerometers) and 10Hz GPS sensors with +/-10 meters error in position (worst case) and +/-0.006 meters/second error in speed. Note that the error model considered here is worst than the navigation grade requirements for civilian transport aircraft sensors. In the followings graphics (see Figures 7, 8, 9 and 10), the purple line (the one from the Flight Dynamics Model i.e. FDM) represents the *real* value given per the simulation environment. The others lines show the output of the different PSFU units (the value estimated per our EKF algorithm implanted in PSFU units) as well as the voted value (i.e. the median value). For the x-axis, a computation step represents 20 milliseconds (i.e. 50Hz frequency) and, for the y-axis, Euler angles values are expressed in radians.

Fig. 7 and Fig. 8 show the simulation results obtained from redundant computations of the roll angle ϕ by the PSFU units.

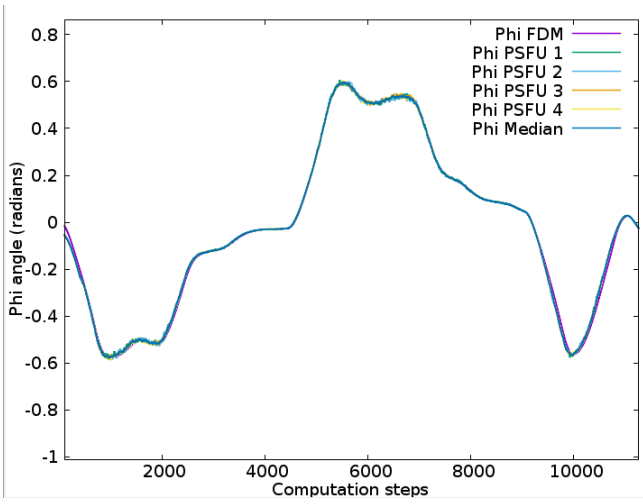

Figure 7 – Roll angle ϕ redundant calc. (Global)

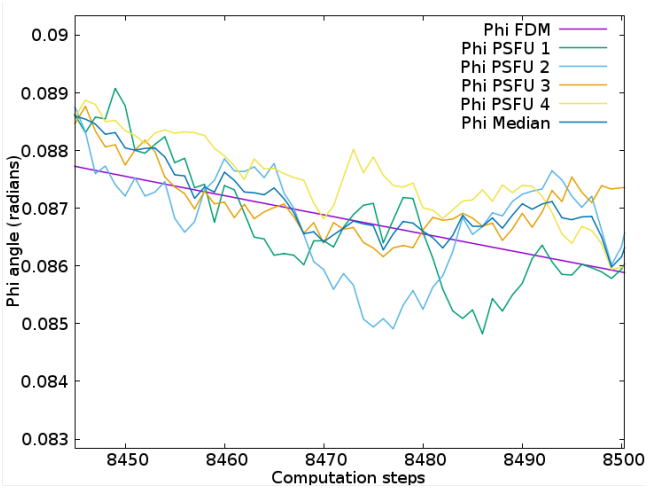

Figure 8 – Roll angle ϕ redundant calc. (Zoom)

Fig. 9 and Fig. 10 show the simulation results obtained from redundant computations of the pitch angle θ by the PSFU units.

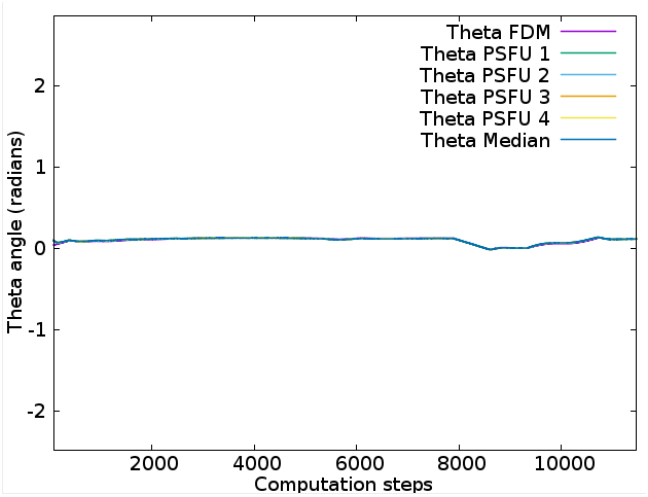

Figure 9 – Pitch angle θ redundant calc. (Global)

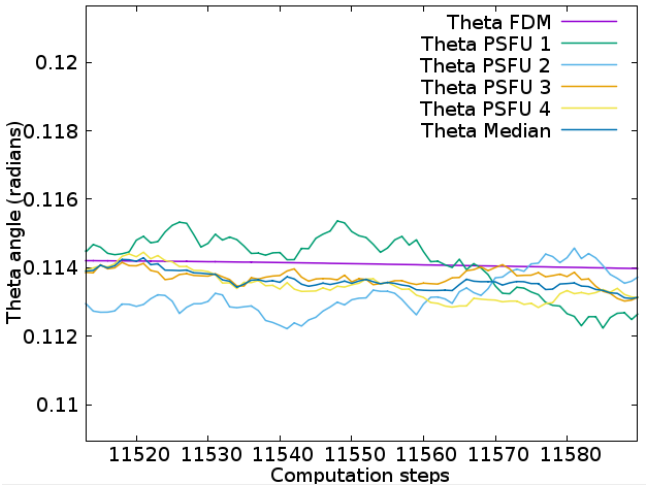

Figure 10 – Pitch angle θ redundant calc. (Zoom)

5 CONCLUSIONS AND PERSPECTIVES

In this position paper, we have presented an overview of our new SMARTIES architecture. To our knowledge, this is the first academic COTS based aircraft FCS architecture which tends to capture every aspects of real FCS systems, in particular it includes complex sensors fusion algorithms. This complex project has required the mastering of many aspects: from the realistic design of close-to-real avionics system unit running on dedicated target to the implementation of an efficient simulation environment compliant with real-time constraints. We have described here the current status of our architecture, and we have introduced fault-tolerant aspects and presented some first results from redundant computation and voting algorithms from sensor fusion units.

This experimental and research platform provides some realistic study cases to investigate different research areas such as fault-tolerance and reconfiguration, modelling and simulation, real-time scheduling and synchronization or future concepts and technologies for avionics systems. The project is ongoing and many parts still have to be investigated, tested and optimised. On the APCU unit level, we are currently working on some extension to integrate high level control loops to mimic the behaviour of aircraft Flight Management System (FMS) which allows pilot and co-pilot to define flight plan according to weather, fuel consumption and so on. We are also currently planning the integration of new units that aim to reproduce simple Full Authority Digital Engines Control Systems (FADEC) (L. Paddon, 1988) to control the engines. On the system level, we are investigating the feasibility of the migration to software based time triggered architecture.

ACKNOWLEDGMENTS

The authors would like to thank TTTech Computertechnik for their courtesy using theirs switches hardware components. The authors would also like to thank Mr. Vashan Srinath Kemthoor for his great work on the sensor fusion algorithms during his master internship at ISAE-SUPAERO.

REFERENCES

J. Banks, J. Carson, B. Nelson. D. Nicol, 2009. Discrete-Event System Simulation (5th edition). Prentice Hall.

J.P. Sutherland, 1968. Fly-By-Wire Flight Control Systems. *Joint Meeting of Flight Mechanics and Guidance and Control Panels of AGARD*, Oslo, Norway, page 1.

D. Briere and P. Traverse, 1993. AIRBUS A320/A330/A340 electrical flight controls - A

family of fault-tolerant systems. *FTCS-23 The Twenty-Third International Symposium on Fault-Tolerant Computing*, Toulouse, France.

I. Moir, 2006. Civil Avionics Systems (Chapter 9: Flight Control Systems). *Aerospace Series (PEP), John Wiley & Sons Ltd.*

E.L. Duke, 1989. V&V of flight and mission-critical software, *IEEE Software*, vol. 6, no. 3, pp. 39-45.

J-B. Chaudron, D. Saussié, P. Siron, M. Adelantado, 2014. Real-time distributed simulations in an HLA framework: Application to aircraft simulation, *Simulation Journal*, Vol.90, Issue 6, p. 627-643

C.R. Hanke, 1970. The simulation of a jumbo jet transport aircraft - Volume I: Mathematical Model. Prepared by Boeing Company Wishita Division, Kansas for National Aeronautics and Space Administration, Ames Research Center, Moffet Flield California.

C.R. Hanke and D.R. Nordwall, 1970. The simulation of a jumbo jet transport aircraft - Volume II: Modeling Data. Prepared by Boeing Company Wishita Division, Kansas for National Aeronautics and Space Administration, Ames Research Center, Moffet Flield California.

D. R. Ryder, 1973. Redundant Actuator Development Study. Report N74-21655. Prepared by Boeing Commercial Airplane Company Seatle, Washington for National Aeronautics and Space Administration, Ames Research Center, Moffet Flield California.

NOAA and NASA, 1976. Standard US atmosphere. *Washington DC: Government US Printing Office*, N77-16482.

H.L. Dryden and I.H. Abbott, 1949. The Design of Low-Turbulence Wind Tunnels. Technical Report 940, National Advisory Committee For Aeronautics, Washington, D.C, USA.

T. Von Karman, 1948. Progress in the Statistical Theory of Turbulence. *National Academy of Science Journal*, Vol. 34.

L. Paddon, 1988. Active-control Engines, *Flight International.*

M.L. Sheffels, 1992. A fault-tolerant air data/inertial reference unit. *Proceedings of 11th Digital Avionics Systems Conference*, Seattle, MA, USA.

R.E. Kalman, 1960. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, Vol. 82.

E. Noulard, J-Y. Rousselot, and P. Siron, 2009. CERTI, an Open Source RTI, why and how. *Proceedings of the 2009 SISO Spring Simulation Interoperability Workshop*, San Diego-Mission Valley, United States, p. 23-27.

J-B. Chaudron, David Saussié, Pierre Siron, Martin Adelantado, 2016. How to solve ODEs in real-time HLA distributed simulation. *Proceedings of the SISO Spring Simulation Interoperability Workshop*, Orlando, United States.

Gerald Coley, 2016. Beagleboard X15 System Reference Manual (Revision B1). SRM_X15, Released on July 22, 2016.

Jan Kiszka, 2016. Xenomai 3 - An Overview of the Real-Time Framework for Linux. *Embedded Linux Conference*, San Diego, California, USA,

ARINC Industry Activities, 2006. Aircraft Data Network Part 7 Avionics Full Duplex Switched Ethernet (AFDX) Network. *ARINC Standard Document.*

SAE International, 2011. Time-Triggered. *SAE Standard Document*, AS6802.

R.P.G. Collinson, 2003. Introduction. In: Introduction to Avionics Systems. Springer, Boston, MA, USA.

D.K. PradhanK. 1986. Fault-tolerant computing: theory and techniques. *Prentice-Hall, Inc.*, Vol. 1.

D.B. Mulcare, L.E. Downing and M.K. Smith, 1988. Quadruplex Digital Flight Control System Assessment, *National Aeronautics and Space Administration, Ames Research Center*, Moffett Field, California.

G.J. Davis and Ames Research Center, 1987. An analysis of redundancy management algorithms for asynchronous fault tolerant control systems, *National Aeronautics and Space Administration, Ames Research Center*, Moffett Field, California.