A NOVEL FMI AND TLM-BASED DESKTOP SIMULATOR FOR DETAILED STUDIES OF THERMAL PILOT COMFORT

*Robert Hällqvist , **Jörg Schminder , *Magnus Eek , ***Robert Braun ,
**Roland Gårdhagen , ***Petter Krus
*Systems Simulation and Concept Design, Saab Aeronautics, Linköping, Sweden
**Applied Thermodynamics and Fluid Mechanics, Linköping University, Linköping, Sweden
***Fluid and Mechatronic Systems, Linköping University, Linköping, Sweden

Keywords: *OMSimulator, FMI, TLM, Pilot Thermal Comfort, Modelling and Simulation*

Abstract

Modelling and Simulation is key in aircraft system development. This paper presents a novel, multi-purpose, desktop simulator that can be used for detailed studies of the overall performance of coupled sub-systems, preliminary control design, and multidisciplinary optimization. Here, interoperability between industrially relevant tools for model development and simulation is established via the Functional Mock-up Interface (FMI) and System Structure and Parametrization (SSP) standards. Robust and distributed simulation is enabled via the Transmission Line element Method (TLM). The advantages of the presented simulator are demonstrated via an industrially relevant use-case where simulations of pilot thermal comfort are coupled to Environmental Control System (ECS) steady-state and transient performance.

1 Introduction

As aircraft sub-systems are becoming increasingly complex in modern aircraft, Modelling and Simulation (M&S) is essential for understanding both steady-state and transient behaviour of tightly coupled sub-systems. Rather than designing and analysing sub-systems separately, a holistic approach is needed. Development testing and Validation and Verification (V&V) activities associated with multiple integrated sub-systems can be shifted to earlier design phases if detailed simulations of large portions of a complete aircraft are feasible. However, in order to further increase the use of M&S, and expand the scope of analysis using models, simulation of coupled models developed in a wide variety of different tools need to be made available on the engineers' and researchers' desktop computers. Also, risks associated with tool-vendor lock-in and licensing costs need to be kept at a minimum if the benefits of M&S are to be further exploited.

The ITEA3 financed research project *Open Cyber Physical System Model-Driven Certified Development (OpenCPS)* [1] aims to address the challenges identified by academia and industry in terms of efficient model integration and simulation. A first step is to deploy available standards and techniques for robust and parallel simulation in current industrial M&S methods and processes. This paper aims to show how existing standards can be used to develop industrially relevant desktop simulators. One specific use of such a multi-purpose simulator is presented via an analysis of pilot comfort coupled to *Environmental Control System (ECS)* performance. Pilot thermal comfort is typically assessed without run-time connections to a detailed ECS model. Conversely, ECS analysis simulations are conducted excluding pilot thermal comfort. Consequently, design errors occurring as a result of sub-optimization are more likely if the system cross-coupling effects are omitted during design. The bi-directional dependence between pilot thermal comfort and ECS performance is demonstrated by means of a use-case where the pilot comfort measure *Fighter Index of Thermal Stress (FITS)* [2] is used to control the cockpit temperature during a simulated mission.

In Section 2, the enabling technologies are introduced along with the rational of their use in M&S. The implementation environment is briefly over-viewed in Section 3. The simulator implementations are described in Section 4.1. The simulator characteristics are introduced in Section 4.2. A pilot comfort study demonstrating one use of the simulator implementation is presented in Section 4.3. Finally, the conclusions are given in Section 5.

2 Techniques for standardized and distributed co-simulation

The Functional Mock-up Interface (FMI) standard [3], the *Systems Structure and Parametrization (SSP)* standard [4], and the *Transmission Line element Method (TLM)* [5] are three key technologies enabling improvement on the state-of-the art in aircraft systems M&S. These technologies are utilized in the OpenCPS project co-simulation framework the *OMSimulator*. A brief overview of a sub-set of the standardized methods for model export, and composite model description, is presented in Section 2.1. *The High Level Architecture (HLA)* standard [6] may be a feasible alternative to FMI for establishing tool interoperability and it is mentioned for context. However, the focus is placed on FMI and this delimitation is motivated by the available tool support along with the application domain. Some techniques for provision of a numerically stable co-simulation is provided in Section 2.2.

2.1 Available standards

The FMI standard specifies a generic format for export of model executables, referred to as *Functional Mock-up Units (FMUs)* [3]. Exported FMUs can be imported and simulated in any FMI supporting tool. Models can be exported as FMUs for *Model Exchange (ME)* or *Co-Simulation (CS)*. With FMI for ME, the responsibility of solving the model equations is passed to the integrating tool whereas each CS FMU contains its own independent solver. The integrating tool is merely responsible for orchestrating the communication between FMUs in the latter case. Various optional features enabling advanced master algorithms are specified in the standard. These options, for example the provision of directional derivatives for smoothing of sampled inputs, can be used to increase simulation performance and robustness.

The SSP standard is an emerging standard for description and exchange of composite simulation models. This standard is the outcome of a *Modelica Association Project* aimed to, along with FMI, provide a complete framework for standardized simulation of multiple connected subsystems models, from here on referred to as *composite models*. The FMI standard is crucial in the sense that it establishes a standardized format for model exchange between domain specific tools; however, it does not address the issue of exchanging parameterized composite models consisting of multiple connected model executables. The SSP standard aims to bridge this gap. In short, a standardized xml schema (denoted *<compositeModel>.ssd*) is used to carry information regarding composite model connections, properties, and graphics. The *<compositeModel>.ssd* is packaged along with its referenced resources in a*<compositeModel>.ssp* file. Examples of ssd referenced resources are other ssp/ssd files and FMUs. An overview of the SSP standard is provided by Köhler et al. in [4]. The SSP standard exists as a mature draft at the time of writing and as such the tool implementations are scarce.

2.2 Numerically stable co-simulation

The partitioning of co-simulation entities unavoidably results in sampling of inputs and outputs. When sampling a continuous system information is lost. The impact of this loss of information needs to be managed and minimized

TLM is a mature and well documented technique for numerically stable partitioning of simulation models [7] Naturally occurring time delays are utilized to derive an independence between connected components or models, specifying a clear and transparent time window for disconnecting and parallelizing coupled simulation models without introducing numerical errors as a result of sub-system de-coupling. *Aliasing effects*, see for example [8], are reduced as interpolated input values are available at the discretion of each individual co-simulation FMU solver. TLM is thus a way to pass high resolution interpolated data while maintaining the sub-system independence during the communication interval. A lossless, one dimensional, hydraulic transmission line can be described by

$$\begin{aligned} p_1(t) &= Z_c \cdot q_1(t) + c_1(t) \\ p_2(t) &= Z_c \cdot q_2(t) + c_2(t) \end{aligned} \tag{1}$$

where

$$\begin{aligned} c_1(t) &= p_2(t - \Delta t_{TLM}) + Z_c \cdot q_2(t - \Delta t_{TLM}) \\ c_2(t) &= p_1(t - \Delta t_{TLM}) + Z_c \cdot q_1(t - \Delta t_{TLM}). \end{aligned} \tag{2}$$

The pressure at side 1 of the transmission line is denote as p_1, and the volume flow as q_1. In the wave variable c_1, the information at side 2 at time $t - \Delta t_{TLM}$ is collected. The information propagation time through the transmission line is denoted as Δt_{TLM}, and the characteristic impedance as Z_c, in Equation 2. Equation 1 and Equation 2 describe hydraulic transmission lines; however, other engineering domains can be described similarly.

Traditional algorithms, see Section 4.2 of the FMI standard Specification [3], rely on oversampling or different filtering and step size error control methods to maintain and improve on simulation accuracy and stability [9]. Oversampling does not necessarily introduce numerical errors if the sampling frequency is at least twice the system bandwidth according to the sampling theorem; however, this method may have significant impact on the simulation performance. Filtering introduces numerical errors as high frequency dynamics are removed to allow for a lower sampling frequency without introducing aliasing. Both traditional co-simulation methods, and the inherently parallel approach of TLM, have benefits and drawbacks if stability is weighted against simulation performance. The traditional algorithms with an adaptive macro step size allow for long step sizes, without large numerical errors, for low frequency operating conditions. Schierz et al. present methods for communication step size control for co-simulation with FMI in [10]. In such operating conditions, TLM comes at the cost of simulation performance as the macro step size need to be limited by the physical delay even though limiting high frequencies are unexcited. Much like in the case of traditional co-simulation, a larger macro step size can be selected at the cost of altered simulation behaviour.

Co-simulation entities, utilizing the TLM technique, require access to interpolated input variables for all internal solver steps. The delay present across a transmission line means that future inputs are available, with a resolution only limited by the step size of the outputting FMU's internal solver, via interpolation in the integrating tool. However, the absence of callback functions in FMI 2.0 makes this information unavailable to the FMU. This would best be achieved by means of callback functions between slaves (FMUs) and the integrating tool; callback functions that allow a slave to ask the integrating tool for inputs valid at the local time they are needed, and for populating interpolation tables with data when made available by the slave. Such functionality is not available in FMI 2.0. Different methods for mitigation are investigated by Braun et al. in [11]. One such method, addressing the absence of the first type of listed callback functions, is referred to as *fine grained interpolation inside the FMU*. This particular method is utilized for the physical connections present in the aircraft systems simulator, see Section 4.1.

3 OMSimulator

The FMI and SSP standards are both utilized together with TLM in the simulation environment OMSimulator. The OMSimulator is an open source master simulation tool originally developed by the Swedish bearing manufacturer SKF, in collaboration with Linköping University, for connecting models of bearings with models from external tools using the TLM technique [12].

Many different FMI supporting master simulation tools already exist or are under development; examples are *Dymola* [13], *FIDE* [14], and *DACCOSIM* [15]. A complete list of the integrating tools supporting FMI is provided by the Modelica Association [16]. In contrast to these tools, the OMSimulator supports asynchronous simulation implementing both TLM and FMI. The open source effort *Hopsan* [17] enables simulation of FMUs using the TLM technique, though only using synchronous communication. Synchronous communication between FMUs may introduce parasitic inductance and capacitance as the communication between integrated subsystems is fixed for all physical connections in the composite model. The OMSimulator asynchronous interoperability between FMI and TLM is established via interpolation inside the FMU, see Section 2.2, enabling multiple different communication intervals in one composite model. A time stamped vector of wave variables is computed via Equation 1 and Equation 2 in the master. This vector is specified to have a resolution significantly higher than the communication step size and it is passed from the master to the FMU. The local solver can access input values for any local time by means of interpolating the wave variable and computing the input via Equation 1. The main drawbacks with such a solution are that the input vector resolution is fixed once the interface is set, models need to be wrapped with input interpolation functionality according to the TLM technique prior to FMU export, and passing interpolation tables to FMUs result in increased communication overhead.

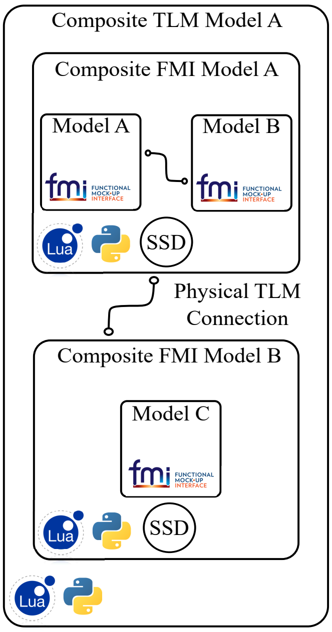


Fig. 1 : Schematic description example of composite model in OMSimulator. Models A, B and C are models exported according to the FMI standard

Composite models to be simulated in the OMSimulator can be implemented using Lua or Python scripting. At the time of this writing, graphical composite model editing support is available via the open-source software *OMEdit* [18] and *Papyrus* [19]. OMEdit is the graphical user interface shipped alongside the OMSimulator with the Modelica tool *OpenModelica* [20], and Papyrus is an open source tool primarily used for UML modelling. Composite models in the OMSimulator can be comprised of coupled FMUs for co-simulation or model exchange. Co-simulation using direct tool to tool coupling is also possible and pre-defined interfaces for the tools Dymola, Hopsan, ADAMS, Beast, and OpenModelica are available. A schematic representation of an OMSimulator composite model is presented in Figure 1. Composite models can be modeled as *Composite FMI models* or as *Composite TLM models*. Composite FMI models are executed sequentially utilizing the dependency information available via the FMI standard to iteratively resolve algebraic loops both during initialization and simulation. Currently, the OMSimulator supports export of *System Structure Description (SSD)* files for export of FMI Simulator composite models according to the SSP standad, see Section 2.2. Work is on-going regarding export of Composite TLM models according to the SSP standard. Composite TLM models consist of two or more composite FMI models connected using the Transmission Line element Method.

The composite model of Figure 1 is described using the OMSimulator Lua API in Listings 1.

Listing 1: Examples of Lua API commands for composite model descriptions in OMSimulator

```
oms2_newFMIModel("FMI Model A")
oms2_addFMU("FMI Model A",<Path>,"Model A")
oms2_addFMU("FMI Model A",<Path>,"Model B")
oms2_setCommunicationInterval("FMI Model A",
<interval>)
oms2_addConnection("FMI Model A",
"Model A.<output>","Model B.<input>")

oms2_newFMIModel("FMI Model B")
oms2_addFMU("FMI Model B",<Path>,"Model C")
oms2_setCommunicationInterval("FMI Model B",
<interval>)
oms2_setRealParameter(
"FMI Model B.Model C:<parameters>",<value>)

oms2_newTLMModel("TLM Model A")
oms2_addFMISubModel("TLM Model A","FMI Model A")
oms2_addFMISubModel("TLM Model A","FMI Model B")

oms2_addTLMInterface("TLM Model A","FMI Model A",
<interface name>, <dimension>, <causality>,
<interface type>, <FMU interface name>)
oms2_addTLMInterface("TLM Model A",
"FMI Model B",<interface name>, <dimension>,
<causality>,<interface type>,
<FMU interface name>)

oms2_addTLMConnection("TLM Model A",
"FMI Model A:<interface name>",
"FMI Model B:<interface name>",<delay>,
<impedance>)

oms2_setTLMSocketData("TLM Model A",<ip>,
<socket>)
oms2_setStartTime("TLM Model A", <start time>)
oms2_setStopTime("TLM Model A", <stop time>)
oms2_initialize("TLM Model A")
oms2_simulate("TLM Model A")
```

In the first block of Listings 1, the FMUs of *Model A* and *Model B* are added to the composite *FMI Model A*. Their intermediate connections are then specified as well as the FMI model communication interval. *Composite FMI Model B* is populated with an FMU of *Model C* in the second block. The FMI composite models are added to a top level Composite TLM model *TLM Model A* in the third block. The TLM interfaces and connections are specified in the fourth and fifth blocks respectively. Finally, the simulation settings are specified in the sixth block and the simulation is commenced.

The OMSimulator is specified to display the Composite FMI model structure of Composite FMI Model A via the first Lua command presented in Listings 2; information most useful for debugging of composite models expressed via the various supported scripting environments. The information generated by the second command is used to iteratively identify and resolve algebraic loops, if such exist, during initialization and simulation. The third Lua command exports the Composite FMI model structure as an SSD file according the SSP standard. At the time of writing the API of Listings 2 is only applicable for Composite FMI models in the OMSimulator, TLM Composite models do not contain algebraic loops and none of the underlying information is therefore explicitly necessary for initialization and simulation.

Listing 2: Lua API commands for composite model export and visualization

```
oms2_exportCompositeStructure("FMI Model A",
"FMI Model A.dot")

oms2_exportDependencyGraphs("FMI Model A",
"InitialUnknowns.dot", "Outputs.dot")

oms2_saveModel("Model.xml","Model")
```

4 Implementation and Results

Saab Aeronautics, Linköping University, and SKF, are collaboratively developing an aircraft systems simulator within the frame of the OpenCPS project. This desktop simulator, and the characteristics of its included sub-systems, are described in the following sub-sections. The focus lies on the simulator as a whole; more in-depth descriptions of the included models are presented by Hällqvist et al. in [21] and Schminder et al. in [22]. In addition, a mission is simulated in the OMSimulator implementation and the results are presented and analysed in Section 4.3.

4.1 Composite model implementation

The aircraft systems simulator is implemented in the OMSimulator using Lua scripting. Two different composite models of the same system are created: one as a single Composite FMI model and one as a Composite TLM model consisting of several connected Composite FMI models, cf. Figure 1. The Composite FMI model structure is presented in Figure 2 using the first API command of Listings 2. The connection circled in red represents feedback of the pilot comfort measure FITS to the implemented controlling software. The simulator behaviour is meant to principally mimic a fighter aircraft. However, the characteristics are not tuned to any specific aircraft.

4.2 Sub-systems

The presented models are modelled in the *Modelica* [23] language and *Simulink*. The Modelica models are developed using the *Modelica Standard Library* and the Saab developed library *Modelica Fluid Light (MFL)* [24]. The latter is a proprietary library; however, open source libraries tailored to modelling of aircraft cooling systems are available and one example for scalable ECS modelling is presented by Jordan et al. [25].

4.2.1 ECS

Figure 5 summarizes the steady-state relative cooling performance,

$$\dot{Q}_{rel} = \frac{\dot{Q}}{\dot{Q}_{req}}, \tag{3}$$

of the included ECS hardware model. In Equation 3, $\dot{Q}_{req}$ is a parameter specifying the requested cooling power and

$$\dot{Q} = C_p \cdot \dot{m} \cdot \Delta T \tag{4}$$

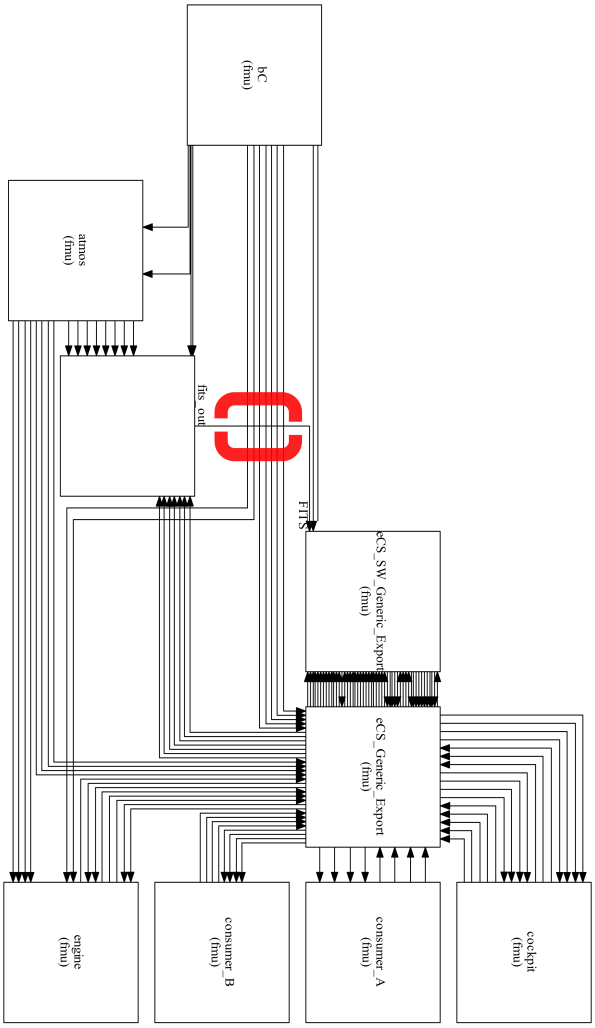


Fig. 2 : Composite model structure of aircraft system simulator implemented as a Composite FMI model in the OMSimulator. The connection circled in red represents the feedback of the FITS measure from the comfort model to the ECS controlling software

describes the cooling power resulting from the temperature increase ΔT across the on board avionics. In Equation 4, $\dot{m}$ denotes the avionics coolant mass flow and C_p the specific heat of the air.

The relative cooling performance governs the flow distribution to the different ECS consumers; flight critical equipment is prioritized above non flight critical equipment as well as the provision of pilot comfort air. A simple ECS controlling software was introduced in [21]. The software is refined and ECS output flow prioritization is included. The implemented

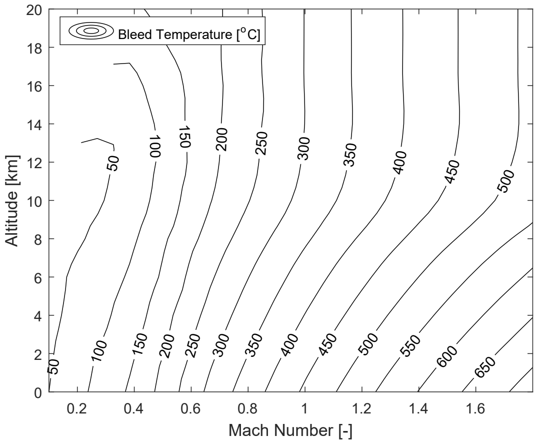


Fig. 3 : Bleed temperature as function of altitude and Mach number during cruise conditions.

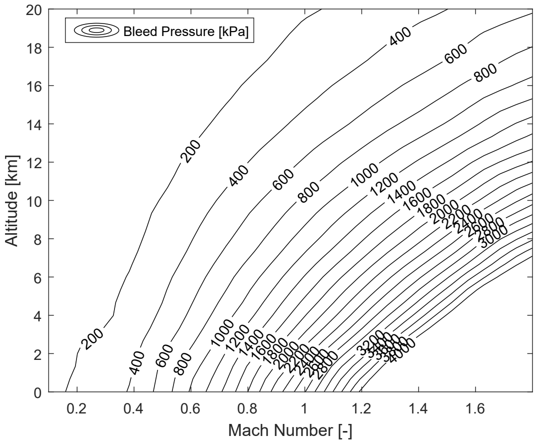


Fig. 4 : Bleed pressure as function of altitude and Mach number during cruise conditions. The pressure increases with increased aircraft speed and air density

prioritization provides a simple but realistic dependence between relative avionics cooling performance and cockpit comfort air flow. If the operating conditions result in a relative cooling performance below 80%, then the nominal comfort airflow is reduced to a minimal level enough to pressurize the cockpit.

4.2.2 *Engine*

An engine model designed to provide ECS input bleed temperature and pressure as function of altitude, Mach number, and ambient conditions is developed. The model outputs values at assumed steady-cruise conditions, meaning that the ECS bleed inputs do not depend on the throttle level rate of change. Cruise conditions are assumed to be conditions where the total aircraft lift is equal to the gravitational force

$$m \cdot a = q \cdot S_{ref} \cdot C_L \tag{5}$$

where m is the aircraft mass, a is the gravitational acceleration, and $q = \frac{1}{2}\rho v^2$ is the dynamic pressure. The aircraft wing area is denoted S_{ref} ,and C_L is the lift coefficient. The total aircraft drag coefficient C_D can be expressed as

$$C_D = C_{D0} + \frac{C_L^2}{AR \cdot e \cdot \pi} \tag{6}$$

where C_L is determined by Equation (5). The aspect ratio is denoted AR and e is an efficiency factor. The second term in Equation (6) is the induced drag coefficient. C_{D0} incorporates form and friction drag into the relation.

The drag force F_d is established by multiplying Equation (6) with the dynamic pressure q and the reference area S_{ref}. The required engine thrust F_T is equal to this drag force during cruise conditions.

The modelled ECS utilizes air bled from the engine post a suitable compression stage. A simple linear relationship $P_{bleed} = k * F_T$ between bleed pressure and the required engine thrust is assumed as the primary purpose of this engine model is to supply the ECS with pressurized air that depend on the aircraft operating conditions. The constant parameter k is tuned such that realistic values of bleed pressure are achieved.

The bleed temperature T_{bleed} is estimated by means of

$$T_{bleed} = T_{in}\left(\frac{P_{in}}{P_{bleed}}\right)^{\frac{\gamma-1}{\gamma}} \tag{7}$$

for adiabatic compression of an ideal gas. T_{in} and P_{in} are the stagnation temperature and stagnation pressure, respectively. The stagnation conditions are computed as altitude, Mach number, and environment dependent outputs of the included atmospheric model. The resulting model char-

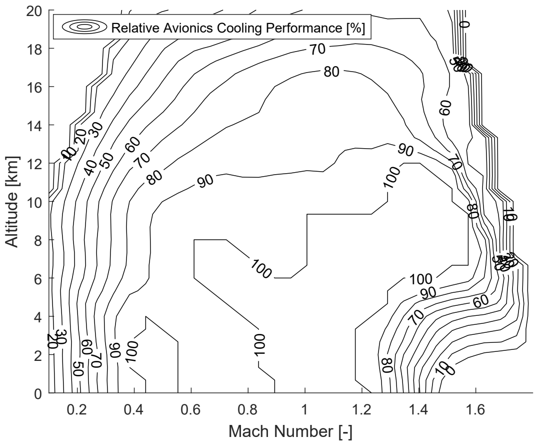


Fig. 5 : Relative main avionics cooling performance as function of altitude and Mach number. The cooling performance is provided by the ISO lines and it ranges from 0 to 100 %

acteristics are summarized in Figure 3 and 4. Numerous sources of information on the complex characteristics of gas turbine engines are available; Sayed provides a thorough compilation in [26].

4.2.3 Cockpit comfort

A detailed cockpit thermoregulatory model was presented by Schminder et al. in [27] and [22]. This cockpit model is included in the simulator where it accounts for the steady-state and transient cockpit temperature dependant on cockpit input airflow from the ECS. The cockpit input airflow includes air dedicated for pilot comfort as well as cockpit avionics coolant air. The included cockpit model accounts for conductive, convective, and radiative heat exchange with its interfacing environments, for example, the ambient conditions and the cockpit avionics. In this section, the specific parts relevant for demonstrating pilot comfort analyses during simulated missions are explained.

The cockpit model provides the necessary inputs to an included thermal comfort model estimating the level of comfort experienced by the pilot. The level of thermal comfort can be expressed through the *Fighter Index of Thermal Stress (FITS)*, presented by Nunnely et al. [2]. This measure is developed for aviation related applications supporting aircrew heat stress risks assessment. Other relevant measures quantifying thermal comfort are the *Wet Bulb Globe Temperature (WBGT)*, the *Predicted Mean Vote (PMV)*, the *Predicted Percentage of Dissatisfied (PPD)*, the *Draft Rate*, the *Oxford Index*, the *Discomfort Index*, and the *Modified Discomfort Index*. These measures are over-viewed by Schminder et al. in [27].

A model incorporating some of the listed measures is included in the composite model. Its use is demonstrated in the application example as the FITS measure is utilized by the controlling software to regulate the temperature and flow of cockpit comfort air. The control is implemented via a PI regulator in the software which outputs requested comfort air temperature provided a FITS temperature set-point of 20°C. The comfort model FITS output value is computed according to

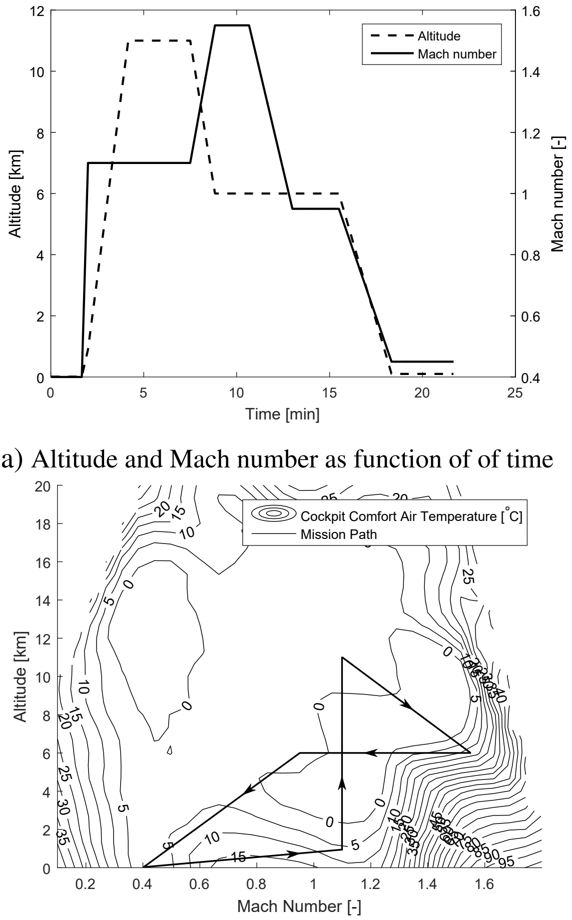
$$T_{FITS} = 0.8281T_{pwd} + 0.3549T_{db} + 5.08 \qquad (8)$$

where T_{pwd} is the wet bulb temperature. The wet bulb temperature corresponds to the minimum possible temperature that can be reached, during given ambient conditions, as a result of evaporation. The dry bulb temperature T_{db} is the ambient temperature excluding radiation and evaporation effects. Nunnley et al. specify upper T_{FITS} limits where, if exceeded, the pilot may suffer from degraded mental performance or physical implications such as dehydration. The control setpoint is selected to avoid such risks. The cockpit comfort air in fighter aircraft is traditionally controlled by the pilot; a measure of pilot thermal strain such as FITS is therefore particularly suited for use when a pilot is unavailable.

4.3 Flight mission simulation and analysis

In [11], the methods for establishing interoperability between FMI and TLM were evaluated using a number of small scale test-cases from different engineering domains. The mission simulation presented here serves as an industry grade verification test-case for the method of using fine grained interpolation, in the OM-Simulator, to ensure numerical stability during transient conditions. In addition, the purpose of simulating the flight profile presented in Figure 6 is to demonstrate the bi-directional dependence between pilot comfort and ECS overall performance. The feedback provided by a pilot in the loop is established via a cockpit comfort air temperature set point dependent on the pilot level of comfort. Such an analysis provides more realistic assessment of performance than the more traditional worst-case type investigations.

The aircraft is elevated to 12 km of altitude where it remains at a speed of Mach 1.1 for approximately 3 minutes. The steady-state relative ECS cooling performance, see Figure 5, is close to 90% in such operating conditions. The incorporated control system does not down-prioritize the flow of comfort air specified by the pilot, and the ECS is able to maintained the requested comfort air temperature. A 20 second dive to 6 km is then used to increase the aircraft speed from Mach 1.1 to 1.55. This is a challenging pose in terms of cooling performance, a conclusion highlighted by the resulting cockpit comfort temperature shown in Figure 6b. Even though the relative cooling performance is sufficient and nominal flow to the cockpit can be maintained, the ECS is unable to supply the specified cockpit comfort air temperature of 0°C as a result of the combination of high bleed air and ram intake temperatures. The aircraft is kept at these challenging conditions for approximately 2 minutes. The aircraft Mach number is then reduced to 0.95 where it is kept constant for approximately 2.5 minutes, a much less challenging situation where nominal flow and temperature levels are feasible, before a final descent sequence is commenced.

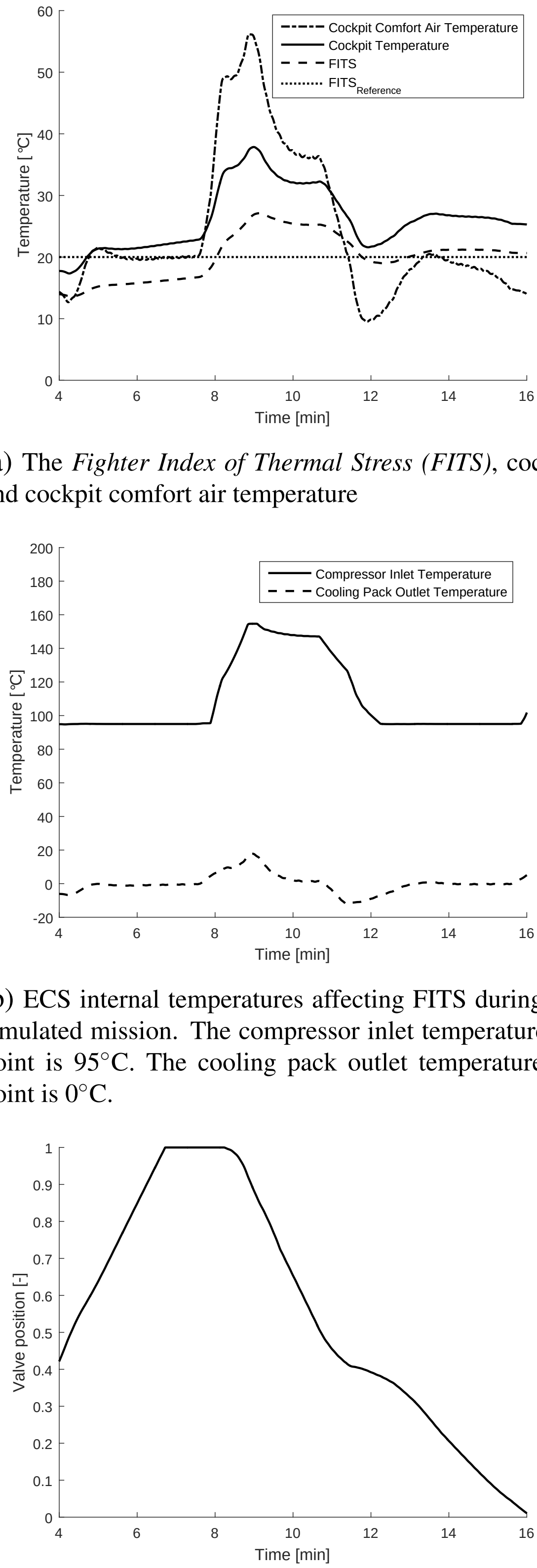


(a) Altitude and Mach number as function of of time

(b) Flight path on top of the cockpit comfort air temperature during steady state operation with a maximum cold cockpit comfort air temperature setting of 0°C

Fig. 6 : Flight profile designed to demonstrate the connection between comfort measures and ECS performance. The cockpit comfort air temperature is elevated at flight conditions where the ECS is unable to maintain nominal operation

A selected set of variables affecting the experienced pilot comfort are presented along with the FITS measure in Figure 7. The temperatures are extracted from a mission time segment where the results of the implemented feedback are clearly visualized. The ECS output cockpit comfort air temperature is shown as dash-dotted, the cockpit temperature as solid, and the FITS temperature T_{FITS} as dashed in Figure 7a. T_{FITS}, see Equation 8, is maintained in the close vicinity of the 20°C set-point throughout the mission via the cockpit comfor air temperature. During the first four minutes of the simulated flight, the FITS measure deviates from the set point by approximately 5°C. In this time span, the controlling software is attempting to increase the temperature by means of opening the corresponding control



(a) The *Fighter Index of Thermal Stress (FITS)*, cockpit and cockpit comfort air temperature

(b) ECS internal temperatures affecting FITS during the simulated mission. The compressor inlet temperature set point is 95°C. The cooling pack outlet temperature set point is 0°C.

(c) Position of cockpit comfort temperature control valve. 1 and 0 represents a fully opened and a fully closed valve respectively

Fig. 7 : Simulator temperatures for a selected time segment of the simulated mission

valve, see Figure 7c; however, as a result of the operating conditions, and the current system design, the control valve is saturated. The two minute temperature transient starting 10 minutes into the mission is a result of the rapid change in operating conditions. This change result in a bleed temperature increase of approximately 200°C, see Figure 3. Even though all air is guided through the primary heat exchanger, the increase in bleed temperature results in an elevated compressor inlet temperature, see Figure 7b. The pack temperature control valve manages to compensate for the majority of this increase; however, the cockpit comfort air temperature control valve is specified as slower, for control stability reasons, resulting in the temperature increase shown in Figure 7a. The impact on pilot comfort is clearly visualized by the elevated values of FITS.

Once the Altitude and Mach number are reduced to 6 km and 0.95 respectively, the bleed and compressor inlet temperatures are reduced to nominal levels. The cockpit comfort air temperature is gradually reduced by means of closing the cockpit comfort valve, after the slight initial undershoot, to compensate for the elevated values of FITS.

5 Discussion and Conclusions

Here, a multi-purpose desktop simulator is developed and presented. The simulator is deployed in the open-source simulation environment the OMSimulator providing an industrially relevant use-case for tool development, evaluation, and verification. Two different methods of connecting sub-system models, exported as Functional Mock-up Units, are successfully implemented in the OMSimulator, see Section 4.1. FMUs are coupled both using the TLM method in a Composite TLM model, and using the more traditional connections in a single Composite FMI model, see Section 3. The TLM Composite model has up to this point been the focal point and only preliminary benchmarking between the different implementations has been done.

Furthermore, the OMSimulator implementations are expressed using the Lua API of the tool and various OMSimulator functionality, such as the export of composite model structure for the Composite FMI model according to the SSP standard, has been verified.

A method to assess and control pilot comfort via the FITS comfort measure has been presented and used to demonstrate and verify the bi-directional dependence between ECS overall performance and pilot comfort. The Composite TLM model is simulated for this mission. The results serve as an initial industry grade verification of the solution of fine grained interpolation for FMI and TLM interoperability as no apparent macro level numerical stability issues can be identified. The composite model behaviour is as expected even though the implemented control is crude and the dynamics of the simulator are not tuned to any existing aircraft. Using a comfort measure for cockpit temperature control is one example of use enabled with coupled co-simulation, compared to the traditional sequential investigations. Other potential uses are optimization studies, maximizing pilot comfort while minimizing energy consumption, steady-state and transient analysis accounting for sub-system cross coupling effects, etc.

Incorporating FMI and TLM into M&S of aircraft sub-systems increases the flexibility in creating simulators comprised of detailed physics-based models. Even though each subsystem is modelled in the most suitable tool, overhead costs associated with model integration are kept at a minimum. From an end-user perspective, improvements in increased and simplified tool support for both model export and integration would aid in composite model development and simulation. For example, greater transparency in the methods for initialization, and data exchange between FMUs, would be of great benefit to the engineers using the techniques in industrial applications. FMI is a mature standard supported by many integrating tools; as such, expansions to the standard take time. Even so, more frequent releases, including expansions of basic functionality concerning for example numerical stability, would benefit the community at large. However, the ability to combine the more traditional execution of FMUs with the TLM approach provides great flexibility in using the most suitable technique for simulation. A load balanced simulator with physical properties resulting in non-negligible time delays and characteristic impedance, may gain significantly in robustness and performance if the TLM technique is implemented.

6 Acknowledgements

The presented research is conducted within the frame of the ITEA3 project OpenCPS and the authors would like to thank the associated funding bodies as well as all contributing project members. Special thanks to Dr. Lennart Ochel at the Programming Environment Laboratory (PELAB) of Linköping University for his continuous support and great efforts in OMSimulator development corresponding to our many of end-user needs.

References

[1] Open Cyber-Physical System Model-Driven Certified Development. https://www.opencps.eu/. Accessed: 2018-06-21.

[2] Nunneley S and Stibley R. Fighter Index of Thermal Stress: Development of Interim Guidance for Hot-Weather USAF Operations. *Journal of the American Society of Heating and Ventilating Engineers*, vol. 50, pp. 639–642, 1979.

[3] FMI development group. Functional Mock-up Interface for Model Exchange and Co-Simulation. Report 2.0, Modelica Association, 2014.

[4] Köler J, Heinkel H M, Mai P, Krasser J, Deppe M and Nagasawa M. Modelica-Association-Project "System Structure and Parameterization" - Early Insights. *Proceedings of the 1st Japanese Modelica Conference*. Tokyo, Japan, 2016.

[5] Krus P. Robust Modelling Using Bi-Lateral Delay Lines for Real Time and Faster Than Real Time System Simulation. *Computers and Information in Engineering Conference, Parts A and B*. San Diego, California, USA, 2009.

[6] Dahmann J and Morse K L. High Level Architecture for simulation: an update. *Proceedings of the 2nd International Workshop on Distributed Interactive Simulation and Real-Time Applications*. Montreal, Quebec, Canada, 1998.

[7] Braun R and Krus P. Multi-threaded distributed system simulations using the transmission line element method. *SIMULATION*, vol. 92, no. 10, pp. 921–930, 2016.

[8] Ljung L. *System Identification, Theory for the User*, vol. 2. Prentice Hall PTR, 2009.

[9] Benedict M and Drenth E. Relaxing Stiff System Integration by Smoothing Techniques for Non-Iterative Co-Simulation. *IUTAM Symposium on Solver Coupling and Co-Simulation*. Darmstadt, Germany, 2018.

[10] Schierz T, Arnold M and Clauß C. Co-simulation with communication step size control in an FMI compatible master algorithm. *Proceedings of the 9th International Modelica Conference*. Munich, Germany, 2012.

[11] Braun R, Hällqvist R and Fritzon D. TLM-based asynchronous co-simulation with the functional mockup interface. *IUTAM Symposium on Solver Coupling and Co-Simulation*. Darmstadt, Germany, 2018.

[12] Siemers A, Fritzson D and Nakhimovski I. General meta-model based co-simulations applied to mechanical systems. *Simulation Modelling Practice and Theory*, vol. 17, no. 4, pp. 612–624, 2009.

[13] Dymola Systems Engineering. https://www.3ds.com/products-services/catia/products/dymola/. Accessed: 2018-06-21.

[14] Cremona F, Lohstroh M, Tripakis S, Brooks C and Lee E A. FIDE - An FMI Integrated Development Environment. *Symposium on Applied Computing*. Pisa, Italy, 2016.

[15] Galtier V, Vialle S, Dad C, Tavella J P, Lam-Yee-Mui J P and Plessis G. FMI-based Distributed Multi-simulation with DACCOSIM. *Proceedings of the Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium*. San Diego, CA, USA, 2015.

[16] Modelica and the Modelica Association. https://www.modelica.org/. Accessed: 2018-06-21.

[17] Axin M, Braun R, Dell'Amico A and Eriksson B. Next Generation Simulation Software using Transmission Line Elements. *Fluid Power and Motion Control*. Bath, England, UK, 2010.

[18] Mengist A, Asghar A, Pop A, Fritzon P, Braun W, Siemers A and Fritzon D. An Open-Source Composite Modeling Editor and Simulation Tool Based on FMI and TLM Co-Simulation. *Proceedings of the 11th International Modelica Conference*. 2015.

[19] Bordeleau F. Model-Based Engineering: A New Era Based on Papyrus and Open Source Tooling. *OSS4MDE@MoDELS*. 2014.

[20] Fritzson P, Aronsson P, Lundvall H, Nyström K, Pop A, Saldamli L and Broman D. The OpenModelica Modeling, Simulation, and Software Development Environment. *Simulation News Europe*, vol. 44, pp. 8–16, 2005.

[21] Hällqvist R, Braun R and Krus P. Early Insights on FMI-based Co-Simulation of Aircraft Vehicle Systems. *Proceedings of the 15th Scandinavian International Conference on Fluid Power*. Linköping, Sweden, 2017.

[22] Schminder J, Gårdhagen R, Nilsson E, Storck K and Karlsson M. Development of a Cockpit-Pilot Model for Thermal Comfort Optimization During Long-Mission Flight. *Proceedings of the AIAA Modeling and Simulation Technologies Conference 2016*. 2016.

[23] Fritzson P. *Principles of Object-Oriented Modeling and Simulation with Modelica 3.3*. 2nd edition ed. IEEE Press, 2015.

[24] Eek M, Gavel H and Ölvander J. Definition and Implementation of a Method for Uncertainty Aggregation in Component-Based System Simulation Models. *Journal of Verification, Validation, and Uncertainty Quantification*, vol. 2, 2017.

[25] Jordan P and Schmitz G. A Modelica Library for Scalable Modelling of Aircraft Environmental Control Systems. *Proceedings of the 10th International Modelica Conference*. Lund, Sweden, 2014.

[26] El-Sayed A F. *Aircraft propulsion and gas turbine engines*. 2nd edition ed. Taylor and Francis Group, LLC, 2017.

[27] Schminder J and Gårdhagen R. A generic model to predict cockpit thermal conditions and resulting comfort. *Submitted to Building and Environmental*, 2018.

Copyright Statement