AIAA Guidance, Navigation, and Control Conference
13 - 16 August 2012, Minneapolis, Minnesota

AIAA 2012-4679

HMS-Control-Interaction Architecture for Rocket Engines

Andrés Marcos*, Luis F. Peñín
Deimos Space S.L.U., Madrid, 28760, Spain

Serge Le Gonidec, Alban Lemaitre
SNECMA, Vernon, 27208, France

In this article, an architecture addressing the interaction between the health monitoring and control modules for launcher's rocket engines is proposed. The first module is responsible for the monitoring tasks (diagnosis, prognosis and decision), while the second for the control tasks (management, reconfiguration and sequencing). For launchers, these two modules are typically designed and implemented in an independent manner since,most often times, the rockets' controllers follow a set of limited and clear instructions (e.g. change operating set-point or perform engine shut-down) not requiring detailed monitoring information. The new generation of launchers is envisioned to utilize advanced controllers and monitoring modules that will allow optimizing the performance of the mission while improving the dependability of the system. In order to carry these modules on-board it is necessary to examine their interaction from a functional and architectural point of view. Such an interacting architecture is proposed and exemplified in this article as part of the technological investigations performed within the European Space Agency Future Launcher Preparatory Programme (FLPP).

I. Introduction

Currently, engineering systems such as launchers are very complex due to the increasing need for performance and safety. These needs must not reduce the dependability of the system, which can be defined as the confidence the user has that the system will deliver its stated service. As described in references[1,2] the dependability of an engineering system depends on three key factors:

- Availability: The ability of an item to be in a state to perform a required function under given conditions at a given instant of time or over a given time interval, assuming that the required external resources are provided.

- Reliability: The ability of an item to perform a required function under given conditions for a given time interval. All parts of the system (hardware, firmware and software) contribute to the reliability.

- Criticality/safety: The level of the damaging consequences to the system and to its environments in case of failure: people, mission, ground facilities.

Health Management systems (HMaS†) are being developed with the goal to improve the dependability of complex systems. In essence, an HMaS should contain a module to monitor the availability of the system and a module to ensure the reliability (i.e. hardware redundancy or advanced controllers capable of reconfiguring in the face of failures). Thus, in order to obtain a dependable HMaS system, the availability of the HMaS must be close to 100%, the reliability close to 100% and the criticality must be null[1].

Although the previous system attributes can be considered in isolation and in increasing, from top-to-bottom, importance, in fact they are interdependent[2]. Thus, integration efforts must be performed in all phases of the HMaS development in order to achieve the goal of dependability. This is even more important with the use of advanced monitoring and control modules which require information from, and provoke changes in, each other. Indeed, from a general perspective, the perceived problems for on-board deployment of HMaS systems are[1,2,4,5,6]: (i) the

* Email: andres.marcos@deimos-space.com. AIAA senior member.

† Sometimes the term HMaS is used to distinguish the referred system from those purely dedicated to monitoring, i.e. HMS (Health Monitoring System). We will use HMaS as a general acronym, accounting also for the interacting relation/interface between the components.

development and validation costs, (ii) the reliability of the HMaS system itself and (iii) the reliability of its components. For example, in off-nominal cases not arising from any type of failure (e.g. due to miscalibration or improper implementation), an HMaS could misconstrue the off-nominal behaviour as a failure due to misdiagnosis and recommend a corrective action that could lead the system towards performance deterioration or even failure.

On the other hand, besides the improved system dependability, the drive in developing HMaS systems arises from: (i) reduced life-cycle costs, (ii) reduced design safety margin, and thus improved performance, (iii) reduced human operation burden due to increased autonomy, and (iv) optimized maintenance and/or pre-operational checks.

Not much is found about the interaction between the control and monitoring modules from an architecture perspective when we look at the "pure" (i.e. algorithmic) academic literature. It is necessary to look at the intelligent control and formal methods communities to find any methodological assessment on the interactions and functionalities of HMaS-type architectures. Indeed, as it is claimed in[7] the main ideas and concepts for HMaS systems are well established from the intelligent control community (good reviews of the concepts and references are found in[4,5]). The proposed HMS-Control-Interaction (HCI) architecture is based on these sources but adapted to the general on-board architecture used by SNECMA for the development of the future Staged Combustion Rocket Engine - Demonstrator (SCORE-D). This HCI architecture has been developed as part of the technological investigations being carried out within the scope of the Future Launcher Preparatory Programme (FLPP) of the European Space Agency (ESA) and is an important component to take the decision (against technological readiness level, needs and cost) to further mature it towards demonstration of advanced rocket controllers in SCORE-D.

The layout is as follows: Section II summarizes the FLPP program and its objectives; Section III generally discusses the functional principles and architecture of HMaS systems; Section IV presents the proposed HCI architecture: context, architecture and functional descriptions, including a discussion on the hierarchy of fault tolerant strategies that can be applied. Section V provides an exemplification of the use of the proposed HCI architecture using conceptual and from the literature study cases while Section VI presents the conclusions.

II. European Space Agency's Future Launcher Preparatory Programme (FLPP)

The Future Launcher Preparatory Program (FLPP) of the European Space Agency (ESA) is preparing the next generation of launchers to be developed in Europe (see reference[8,9,10]). The FLPP is a system-driven programme that targets long term applications as well as possible short/mid-term spin-offs of technologies and components related to the next generation of launchers.

The general objective of FLPP is to prepare the technical and programmatic elements to decide on the best European launch system required to respond to the future institutional needs while maintaining European competitiveness on the commercial market. The preparation of these technical and programmatic elements is mainly based on the maturation of enabling technologies which will mitigate the risk during the development of a future launcher[8,10]. In addition, integrated demonstrators are considered in order to increase the technology readiness level while addressing the necessary improvement and mastering of system-level competences. The objective of the technology maturation is to reach a technological readiness level (TRL) of 6 which is considered as an adequate level to mitigate risks before entering firm development.

The FLPP was accepted by the 2001 ESA ministerial Council as an optional programme within the Agency framework to prepare for possible future evolutions of Ariane 5 and Vega and for the Next Generation Launcher (NGL). The 2003 ESA ministerial Council finally adopted the 1st Period of FLPP, covering the years 2004-2006, in which activities were focused on system studies and technology developments for Reusable Launch Vehicles (RLV). Step-1 of the 2nd Period of FLPP was adopted by the ESA Council meeting at ministerial level in Berlin in 2005, with the objectives to continue the preparation for the Next Generation Launcher (already started in FLPP Period 1) for the long term, and to contribute to the preparation of short/medium term decisions through activities on expendable launch systems. The Phase-2 of the 2nd Period was adopted at the Council on Ministerial Level in 2008 in Den Haag. Overall, more than 60 companies, institutes and universities are involved. The programme takes benefit from this large industrial organization, where sub-contractors can be involved early in system studies for trade-offs and actors currently not involved in ESA-developed launchers can bring their know-how for the preparation of the future NGL.

Within the FLPP, the Main Stage Propulsion program started in 2005 and has pursued the main goal of preparing the necessary tools and technologies that will be used to enable the design and development of the post-Ariane 5 first stages propulsion system, which is slated to enter into service around 2020-25. The activity is driven by the Joint Propulsion Team (JPT), a consortium led by SNECMA (France), EADS-Astrium ST GmbH (Germany) and AVIO SpA (Italy), and it is divided into two main tasks: the design and trade off of (i) a propulsion system with

staged combustion engine, and (ii) the accompanying technologies in the different fields (turbo-pumps, thrust chamber…).

As part of the first task, the development of the Staged Combustion Rocket Engine - Demonstrator (SCORE-D) targets to demonstrate the integration of staged combustion technology for the European high-thrust engines and to validate the associated design and analysis tools as well as the critical manufacturing processes[11]. Additionally, the SCORE-D design, analysis and manufacturing will enable developing required competences in Europe and will provide the technical and programmatic basis to make informed decisions on future launchers and propulsion systems. The demonstration includes, as much as possible, new technologies developed as part of FLPP and those developed in national programmes.

The features of SCORE-D, see Figure II-1, have been chosen so as to maximize the ratio between representative future operational engines and cost. The engine is based on versatility and modularity, allowing the testing of different variations of the same function by the easy replacement of specific components. In addition, and depending on the costs and risks, it may serve thanks to its modularity, as a preliminary study for testing propulsion technologies beyond the next generation high-thrust engines.

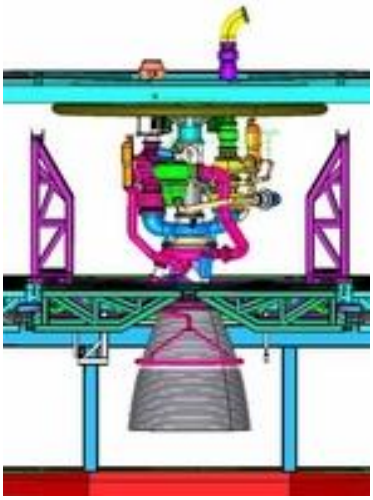

Figure II-1 Staged Combustion Rocket Engine - Demonstrator (courtesy of ESA)

III. General HMaS Functional Principles and Architecture

A. Functional principles

Any HMaS system should aim for mission success while satisfying safety requirements[1,2], see Figure III-1. In order to accomplish these objectives, the safety and availability (equal to probability of correct operation at a given moment) of the system must be improved especially in terms of the handling of failures. Safety is affected when a failure occurs that is not detected (no-detection) while availability is affected when a stop due to an unjustified alarm is declared (false alarm).

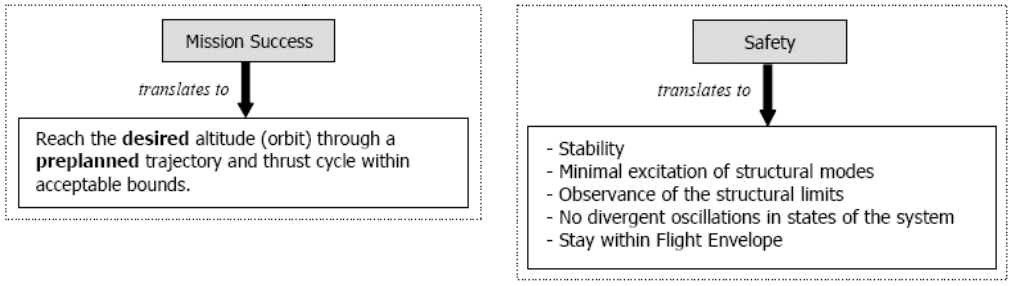

Figure III-1 HMaS Architectures Objectives and General Control Actions[2]

In reference[1] a conceptual HMaS architecture is broken down between control and diagnostic components in terms of functional pyramids, see Figure III-2. Presently, both pyramids have a decreasing TRL according to the bottom-up direction.

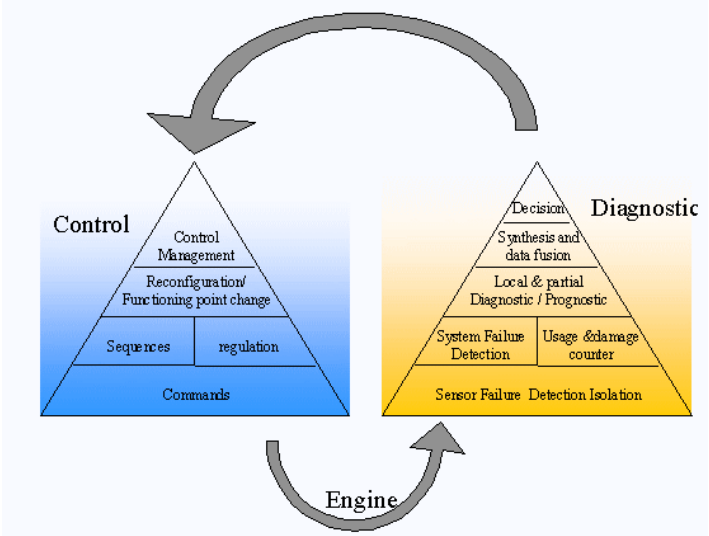

Figure III-2 Conceptual Pyramid Health and Control Management[1]

The list of the main HMaS functional modes are:

- *Detect*: it consists in deciding whether the system is in normal working state. The result of the detection procedure is an alarm meaning that the real operation of the system does not agree any more with the model of healthy operation.
- *Isolate*: or locate, which comes down to assign the defect to the defective module of the system: sensors, actuators, process or control unit.
- *Diagnose*: it consists in carrying out classification defects according to certain parameters which characterize them, e.g. moment of appearance and amplitude. This stage also consists in envisaging the evolution of the defects and quantifying their degree of severity.
- *Prognose*: consists in carrying out computations of the previous defects. In this process the remaining lifetime of the elements is computed and used in support of the estimation of a maintenance objective.
- *Correct*: this functionality can be viewed as the optimization of available resources given the constraints of mission objectives and safety.

  All HMaS corrective actions can be divided into two general modes: "reconfiguration" and "fail-safe" see Figure III-3. If sufficient information is available, the HMaS system can try first to reconfigure either by changing the mission objectives or by reconfiguring the control module. The latter can be performed by using analytical/hardware redundancy (i.e. selecting different actuators, sensors or estimating/reconstructing measurements), by applying or activating new control orders/modules (accommodation or switching), or by calculating and updating control parameters (adaptation). If in some situation neither of the "reconfiguration" actions are feasible, the more conservative but safe approach is to switch to a Fail-Safe/Fail-Operational mode.

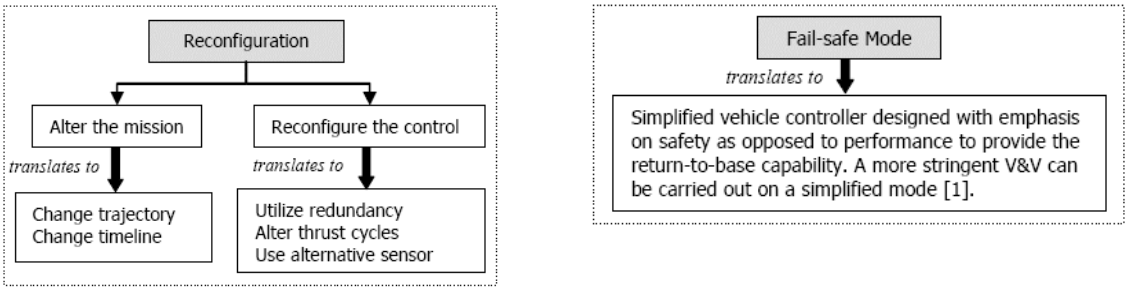

Figure III-3 HMaS Architectures General Corrective Actions[2]

B. General architecture

Reference[5] describes a general functional architecture for autonomous control systems that contains the main applicable and fundamental concepts for any HMaS architecture (including the interaction between the different modules). Figure III-4 presents this hierarchical architecture decomposed in levels whose main functionalities are:

- *Management and organization level*. Responsible for: (i) interfacing with the mission/pilot/user, (ii) assessment of capabilities, (iii) performing high-level planning and decision-making, (iv) monitoring of all activities, and (v) satisfaction of directives at the lower levels.
- *Coordination level*. Its main responsibility is the "practical" decision-making (based on the directives from the above level) of the actions for the lower level subsystems. Some of the specific functions are: (i) performing the sequence of control actions dictated by the upper level, (ii) identifying which specific controller/actuator/sensor algorithm at the lowest level performs a defined action, (iii) identifying and sending the updated parameters/gains required for the lower level.
- E*xecution level*. Its main responsibilities are: (i) to generate low-level control actions via the pre-defined or the adaptive algorithms and (ii) to apply these actions to the vehicle.

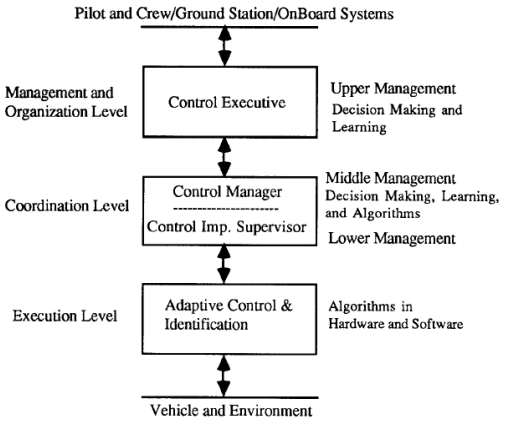

Figure III-4 A General Functional Architecture for an Autonomous Controller[5]

It is noted that the above general autonomous architecture condenses well the functionalities established in the SNECMA-HMaS development architecture, Figure III-5, albeit with some differences due to the split of the latter into a control (left-side box) and diagnosis (right-side box) component. Nevertheless, if these two branches are merged the resulting three-level architecture almost exactly maps the standard one in Figure III-4. In exactitude, the general management and organization level will be composed of the "control management" and "decision making" blocks; the coordination level by the "control reconfiguration", "prognosis", "data-fusion" blocks; and the execution level by the "control laws" and "diagnosis FDI".

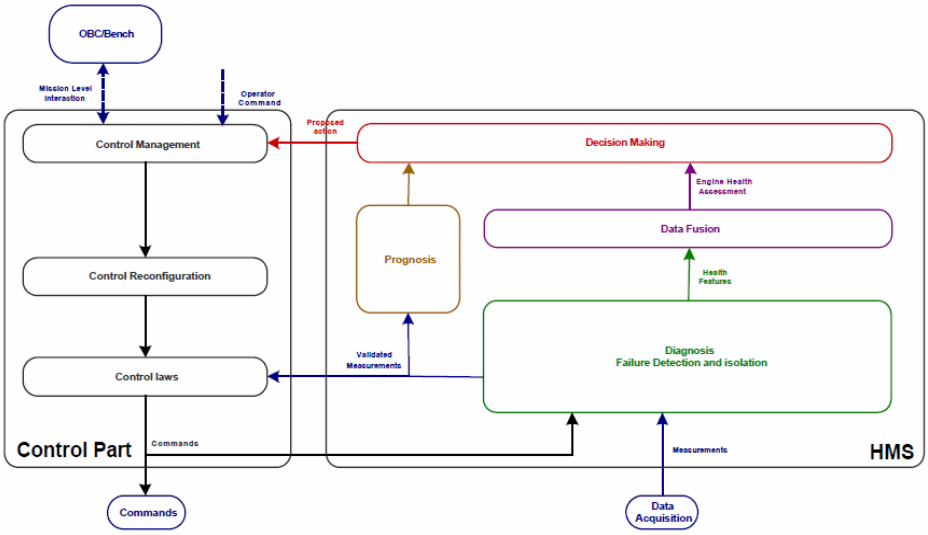

Figure III-5 SNECMA Selected Architecture for HMaS Development[1]

A standard functional architecture has been described briefly above and compared to the HMaS architecture proposed within the FLPP activities. This comparison is performed to set common structures and concepts that help to define the individual and specific functions for the HCI architecture. It is also beneficial to compare functionalities at a finer granularity level, i.e. closer to the algorithmic design level. For this, it is necessary to look at the work performed by the fault tolerant (FTC) and fault detection & isolation (FDI) communities reflected in the so-called "reconfigurable", "integrated" or "fault tolerant" control. In reference[12] a good overview of the methods and issues for diagnosis and reconfigurable control interaction is given. The review is performed from an algorithmic perspective and thus the terminology reflects this mentality. In what follows the nomenclature used in that reference is maintained except when the mapping of the notation with that from the above HMaS is not clear.

Typically, integrated control/diagnosis architectures are composed of four main parts, see Figure III-6:

(i) A reconfigurable Controller (RC),

(ii) A fault detection and diagnostic scheme (FDD),

(iii) A controller reconfiguration mechanism

(iv) A command/reference governor.

Reference[12] terms this architecture as "active fault tolerant control system" (AFTCS) and is characterized by the inclusion of the FDD and RC components (a "passive" scheme is one that does not require an FDD module since the controller is designed to be robust to faults and uncertainties). Note that in comparison to the standard architecture from Figure III-4, the AFTCS only lacks the supervisory/management layer and the interfaces from/to the FDD and RC components to said layer.

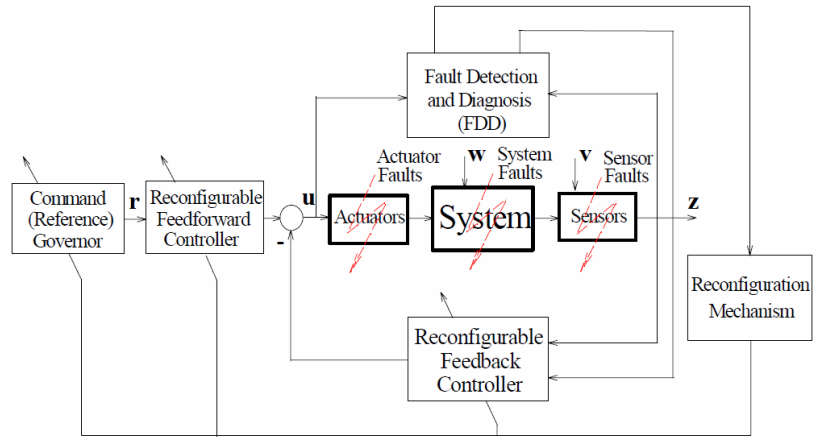

Figure III-6 General architecture of AFTCS[12]

IV. HMS/Control Interaction (HCI) Architecture

This chapter presents the proposed Control-HMS Interaction architecture. It is conceptually very generic, following established guidelines for similar type of "intelligent" architectures[13], but reconciled with SNECMA's HMaS architecture from Figure III-5. The layout of the chapter is as follows: Section 7.1 presents the architecture operational context, Section 7.2 the proposed HCI architecture and its functional components, and Section 7.3 presents a hierarchy of recovery strategies that can be followed using the architecture.

C. Interaction context diagram

Figure IV-1 presents the operational context for the HCI while Figure IV-2 presents the design context diagram. The operational context diagram represents the interaction with the external world of the HCI during its operation. On the other hand, the design context diagram represents all the different elements that should be considered during the design process, along with the major relevant issues that will affect the HCI performance.

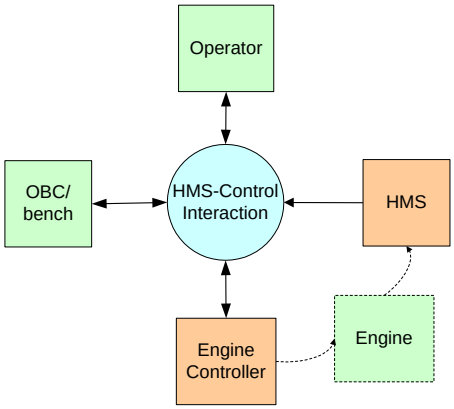

Figure IV-1 HCI Operational Context

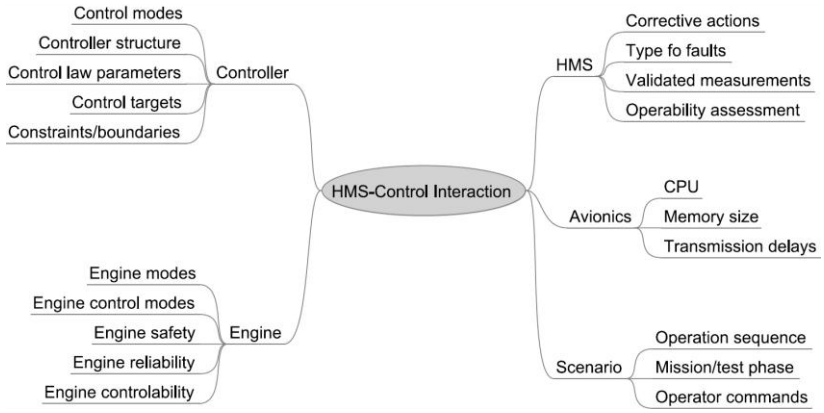

Figure IV-2 HCI Design Context

D. Proposed HCI architecture

The proposed control-HMS interaction (HCI) architecture is built around the control component architecture developed by SNECMA for SCORE-D[1]. The specific task in developing the HCI is to further specify and mature the first two blocks within SNECMA's architecture of Figure III-5, which are the basis of the control-HMS interactions.

Several modifications are suggested in order to reconcile the architecture with the general "autonomous and intelligent" control architectures reviewed in Section III, and to reflect the development of the assigned blocks in terms of required input/outputs. Figure IV-3 shows, in red, the proposed modifications (based on the left block from Figure III-5):

- The name of the F6 block is changed to "Control Coordinator". This reflects better the aim of this block as well as better reconcile it with the general concepts from the intelligent community[13] and similar engine control-HMS architectures[14,15,16].
- The inputs to the F5 and F6 block are now augmented. Note that the input term "operability assessment" includes also the definition of the "operability reference".

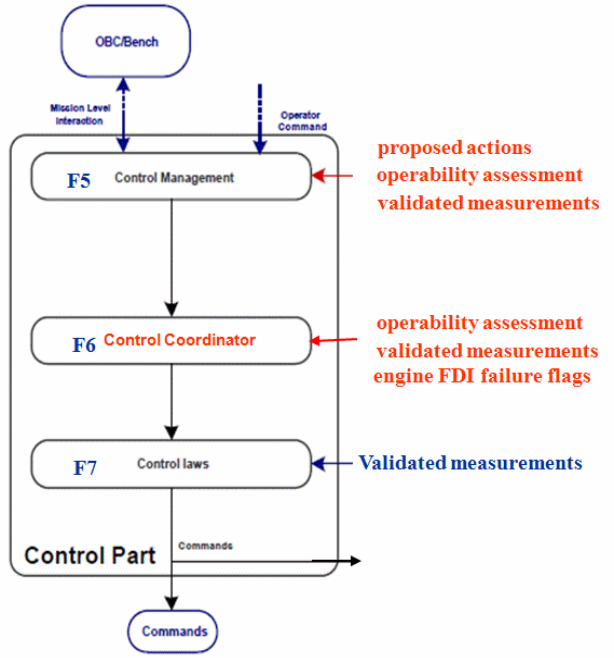

Figure IV-3 Proposed modifications to the general SNECMA HMaS control-component

The two developed control-HMS blocks, i.e. the "Control Management" (F5 block) and the "Control Coordinator" (F6 block), are presented in Figure IV-4. The figure also shows their immediately lower-level functional decomposition, together with their interconnectivity and that with external components. A functional description for each of block is detailed in the next subsection. As the names indicate, the F5 block responsibility is to manage the implementation of the (propulsion/engine) control actions demanded by the external as well as the health monitoring (HMS) components. The F6 block's responsibility is then to coordinate the interaction between the higher-level operations (performed by the "Control Management" block) and the low-level execution actions (performed by the "Control Law" block).

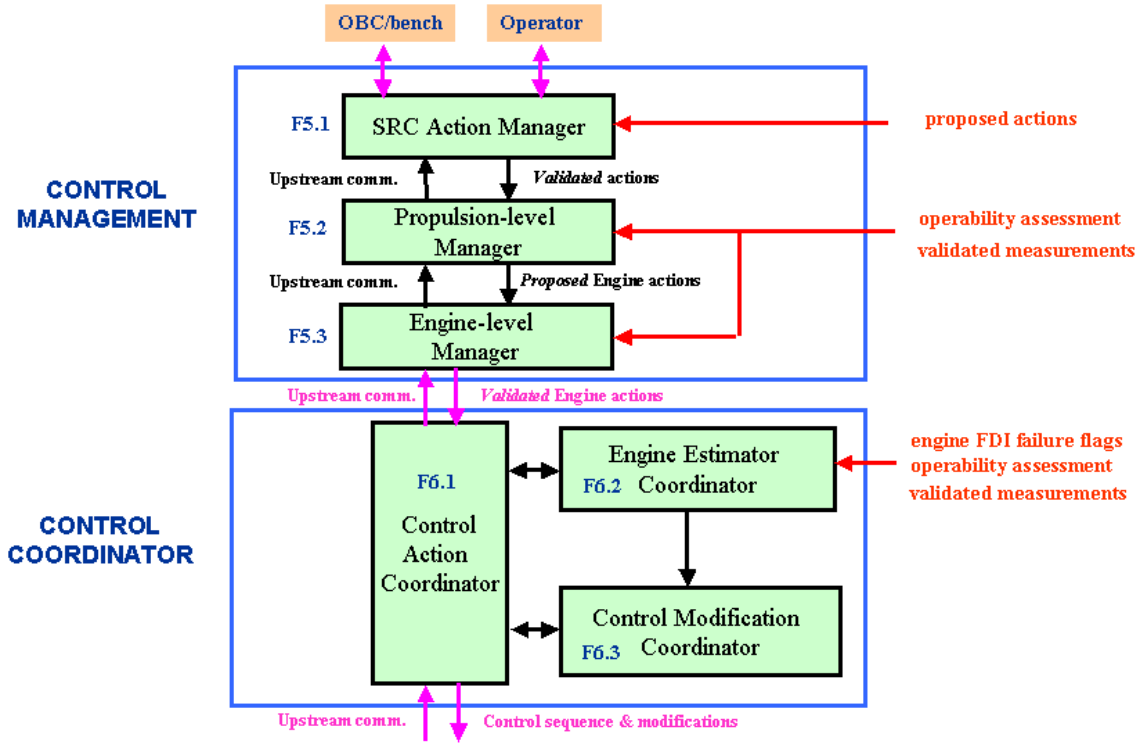

Figure IV-4 Proposed HCI architecture

E. Functional Description

For the "Control Management", the main functions are –corresponding to each of the functional blocks F5.1, F5.2 and F5.3 shown in Figure IV-4:

- To evaluate and resolve any conflict on the actions requested by the different external entities (operator, on-board computer (OBC) and HMS). This conflict resolution evaluation is performed by examining the criticality of the requests in terms of their impact on the system's *safety, reliability and controllability* (SRC).
- To propose, from a propulsion-level point of view, the most acceptable engine control action based on the validated action from the previous block. For example, in the case of a single engine, the block may propose a new operability reference or, if an advanced engine control exists, a modification of the thrust (e.g. directly or through chamber pressure changes).
- To assess, at the individual engine-level, the validated engine-control action based on the proposed engine action defined in the previous block.

For the "Control Coordinator" block, the main tasks are –corresponding respectively to the F6.1 to F6.3 blocks:

- To coordinate the accomplishment of the validated engine-control actions by: (i) defining and sequencing the most optimal control sequence, (ii) demanding and coordinating the modification of the control laws and (iii) defining the estimation requirements on the engine and engine control parameters.

- To coordinate the estimation of all the current and expected engine and engine control parameters required by the previous and subsequent functional blocks.
- To coordinate the control law modifications demanded by the "control action coordinator".

A more detailed functional description for each of the HCI F#.# blocks is provided next. Each description includes a graphical representation of the block showing the input/output sets, and a general description of the function's goals, specific tasks and inputs/outputs.

F5.1 SRC Action Manager

The objectives of this block, see Figure IV-5, are to manage any conflict among the requested control actions by the different external entities (operator, OBC and HMS component) and to centralize all communications from/to the external entities. The inputs to the F5.1 block are: the mission phase (from the OBC), operator commands (from operator), *proposed* actions (from the HMS module) and any upstream communications (from the F5.2, F5.3 and F6 blocks). The outputs are *validated* actions to the F5.2 block and communication messages to the OBC and/or operator.

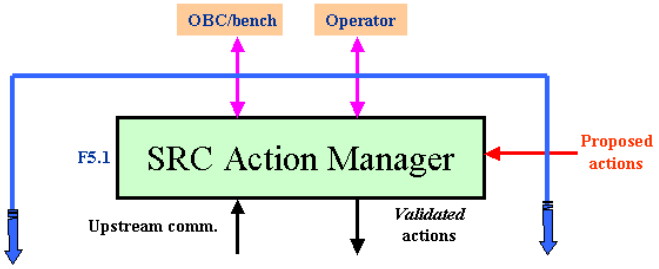

Figure IV-5 F5.1 "SRC Action Manager" block

In performing the conflict resolution, the "SRC action manager" looks at the criticality of the requests from their impact on the safety, reliability and controllability of the system. The SRC block uses a decision logic defined off-line and prior to the system operation. The following SRC hierarchy is proposed, including examples of the commands:

- Safety. This is the most critical action, for example commanding an engine shutdown.
- Reliability. This is a less demanding modification, i.e. mitigation (either target modification or operability adjustment for the system)
- Controllability. Divided in three types of actions: accommodation (i.e. modification of control law parameters), reconfiguration (i.e. modification of control law input/outputs), and adaptation (i.e. intelligent modification control law parameters).

F5.2 Propulsion-level Manager

The tasks of this block are to assess the propulsion system capability to handle the requested validated action, and to manage the definition and transmutation of the *validated* (propulsion-level) actions into *proposed* engine action (e.g. definition of thrust/fuel-utilization vectors from overall launcher thrust target). In essence, the aim of this block is to assess if the *validated* actions from F5.1 can be mitigated from a propulsion system level perspective. By mitigation it is understood the taking of actions that do not require modification of the control law (neither parameters nor input/output).

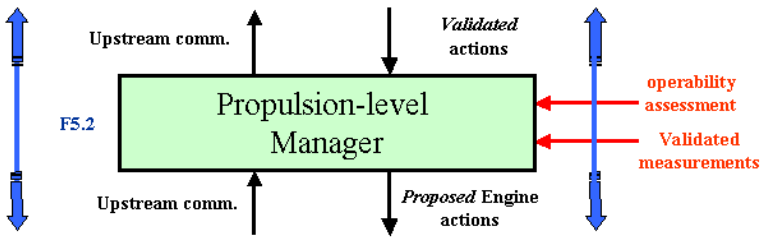

Figure IV-6 F5.2 "Propulsion-level Manager" block

The implementation of a *validated* action at propulsion-level results in a *proposed* engine action, for which there are two types: (i) target modification, e.g. different thrust level by engine, or (ii) operability modification, e.g. new commanded gimbal deflection to each engine. Thus, the inputs come from the HMS-component (operability assessment and validated measurements) and from the F5.1 blocks (*validated* actions). The outputs are the *proposed* engine actions and communication messages to upstream (i.e. to the F5.1 block)

F5.3 Engine-level Manager

The aim of this block is to examine the *proposed* engine actions from F5.2 and assess if they can be executed by each of the engines. The specific tasks are then: to assess the engines' capability to handle the *proposed* engine action based on their current and assessed operational status, and to manage the acceptance or modification of the *proposed* engine actions into *validated* engine actions. The *validated* engine actions, which are also calculated from a mitigation perspective, include target modification (e.g. expected achievable thrust levels) or operability modifications (e.g. expected achievable gimbal deflections).

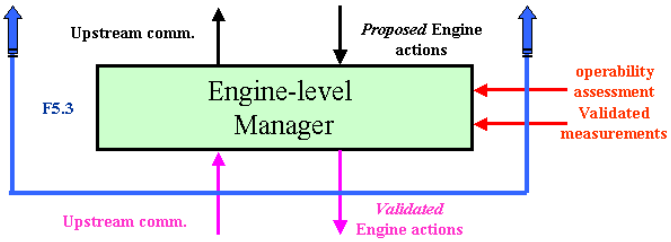

Figure IV-7 F5.3 "Engine-level Manager" block

F6.1 Control Action Coordinator

This block is in charge of coordinating the accomplishment of the *validated* engine actions commanded by the F5 block as well as of the information flow and communication within the F6 blocks. The main tasks are:

- Coordinate the definition and optimal calculation of the most appropriate engine sequence, evaluating along the process its effect on the engine performance.
- Calculate the optimization parameters used in the definition of the optimal control sequence solution.
- Determine the need for engine control modifications by the block F6.2 "Control Modification Coordinator" and coordinate the smooth implementation and distribution of said modifications.
- Coordinate communication from/to F6

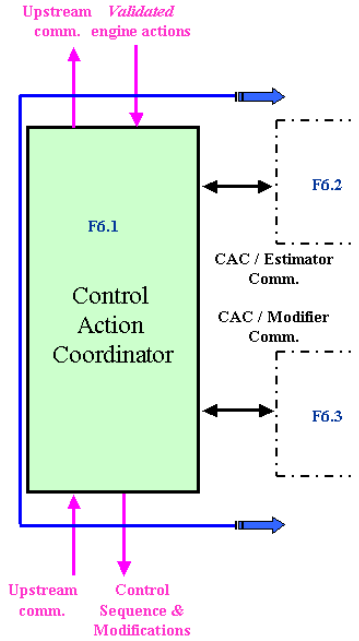

Figure IV-8 F6.1 "Control Action Coordinator" block

The inputs and outputs for the F6.1 block are given in the top row of Table 1.

Table 1 F6.1 and F6.2 input and output interfaces

F6.# block	Inputs	Outputs
F6.1	• *Validated* engine actions (from F5) • Upstream information (from F7) • Validated measurements (from HMS) • Estimated engine parameters (from F6.2) • Engine control law modifications (from F6.3)	• Control sequence & modification (to F7) • Upstream communication (to F5) • To F6.2 block ➢ Proposed control sequence ➢ Smoothed control modification • Control modification request (to F6.3)
F6.2	• From F6.1 block: ➢ Proposed control sequence ➢ Smoothed control modifications • From HMS block: ➢ Engine FDI failure flags ➢ Engine operability assessment ➢ Validated measurements	• To F6.1 block: ➢ Estimated engine operability parameters ➢ Estimated engine model parameters • To F6.2 block: ➢ Estimated engine control parameters ➢ Validated measurements
F6.3	• From F6.1 block: ➢ Proposed control sequence ➢ Smoothed control modifications • From F6.2 block: ➢ Estimated engine control parameters ➢ Smoothed control modifications	• To F6.1 block: ➢ Set of engine control law modifications

F6.2 Engine Estimation Coordinator

The aim of this block is to estimate all current and expected engine parameters required by the F6.1and F6.3 blocks. The estimation models and tables can be modified based on the smoothed control modifications transmitted by the F6.1 block from the F6.3 block. The set of tasks that the block can perform are: (i) estimate the engine operability (e.g. thrust, O/F range…), (ii) estimate the system parameters (e.g. fuel consumption rate…); and (iii) estimate the expected and achievable engine parameters for the application of the proposed control sequence by the F6.1 block. The list of inputs and outputs of this block are given in the midle row of Table 1.

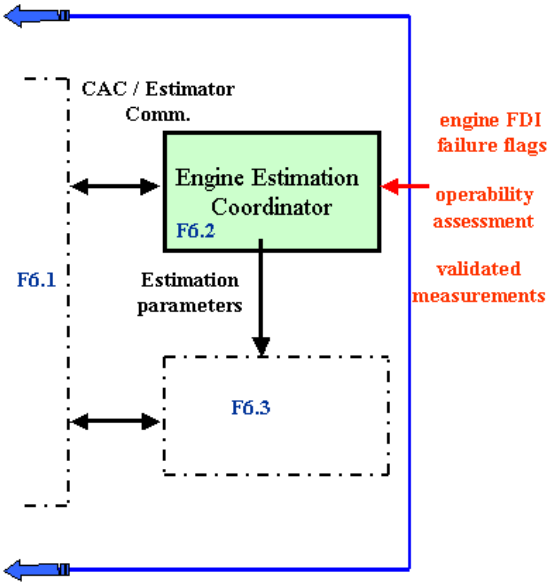

Figure IV-9 F6.2 "Engine Estimation Coordinator" block

F6.3 Control Modification Coordinator

The aim of this block is to coordinate the advanced engine control law modifications requested by the F6.1 block to achieve the F5 block's *validated* engine actions. The inputs / outputs are given in Table 1. The specific tasks are:

- To determine, based on satisfaction of specific conditions and authorization of F6.1 block, the level of engine control law modification. These modification levels can be arranged in increasing order of technological difficulty as, see also next subsection: accommodation (selection of pre-defined control law parameters), reconfiguration (selection of pre-defined control law architectures, i.e. input/output changes and possibly parameters) or adaptation (intelligent calculation of control law parameters based on unknown conditions, i.e. adaptive control systems).
- To determine the specific control law modification that best satisfies the control modification request from the F6.1 block subject to the estimated and validated engine parameters from F6.2.

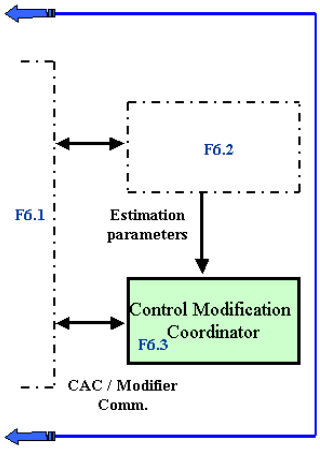

Figure IV-10 F6.3 "Control Modification Coordinator" block

F. A hierarchy of recovery strategies

An important part of the functionality of the proposed HCI architecture relates to the possible control recovery actions that can be performed by the F6 "Control Coordinator". This section presents the most established control modifications and provides the required background to subsequently develop 2nd and 3rd level functionality blocks (see the examples in Section V).

Fault tolerant control (FTC) is intrinsically related to HMS through the fault detection and isolation (FDI) module if an 'active' perspective is desired, i.e. if the control action is to be driven by the monitoring information. This is so because the FTC module depends as a minimum on a diagnostic signal from the FDI module –and for risk minimization purposes, might also depend on a decision signal from an HMS component. Of course, well-designed robust controllers provide always a natural level of tolerance to many upset situations but this comes at the cost of reduced performance. Thus, in the case of more severe upsets, special strategies are required to satisfy the stringent requirements on probability of failure. Further, any functional analysis of the interaction between HMS and control must take into account the possible solutions available for FTC.

From a broad perspective, the available recovery actions can be grouped into:

- *Mitigation*: By mitigation it is understood the taking of actions which do not require modification of the control law (neither parameters nor input/output). An example applicable to engine systems is the modification of the target mixture ratio. Approaches that are naturally robust or that have been designed directly against expected faults are also included in this solution.
- *Accommodation*: In general refers to the selection of pre-defined control law parameters. This solution includes those approaches known as (gain) scheduling and switching.

  It is important to note that other approaches that modify the control law parameters based on measurable and bounded time-varying parameters such as linear parameter varying (LPV) systems[17,18,19] are also considered within this type of solution.

- *Reconfiguration*: Modification of the structure (input/output) of the control law, possibly including modification of its parameters. An example is the blockage of an engine control valve, which results in a reduction in the control law output dimension and can force a change in the controller gain to compensate for the reduced actuation.
- *Adaptation*: Intelligent calculation of control law parameters based on unknown conditions (i.e. adaptive control systems[6] as opposed to LPV systems). Adaptive approaches have attracted considerable attention in the past 10 years but are well known to be extremely difficult to certify, if at all, due to their non-causal and unknown nature.

Note that the above classification is ordered in terms of complexity (in design, implementation and/or certification) and cost (in hardware, control action) from the lowest to the highest. But also note that the level of fault tolerance and achievable performance is better as the more complex controller is used.

V. Exemplification of Fault Cases Handling by the HIC Architecture

In this section, an exemplification of the main functional components of the proposed HCI architecture is presented. Each of the following subsections exemplifies their functioning and evolution by showing respectively: the envisioned conflict-resolution logic implementation of the "Control Management" block, and a potential 3rd-layer decomposition of the "Control Coordinator" block. The given examples are based on the NASA studies performed within the Space Shuttle Main Engine Intelligent Control Framework[14,15,16].

G. "Control Management" functioning example

Figure V-1 shows the flow diagram for a snippet of a possible management logic. The example shows the logic followed to determine the conflict between actions from two external entities but it also shows how to apply a hierarchical decision-making logic corrected by the effect of the action on the system.

Specifically, the example shows the case of a conflict between the "operator command" and an "HMS proposed action". The logic assumes, as it should be, that the operator commands are ranked hierarchically higher than those from the HMS block but that the latter block might be more reactive (in time) to conditions on the system. Thus, whenever the operator has not been informed of the HMS block action and there is disagreement between the actions, the logic decides on validating the proposed HMS action from the perspective of the system's safety (stability), reliability and controllability. It is important to note the different types of actions validated at each step: i.e. from "action" to "operator command" but also from "mitigation" to "controllability".

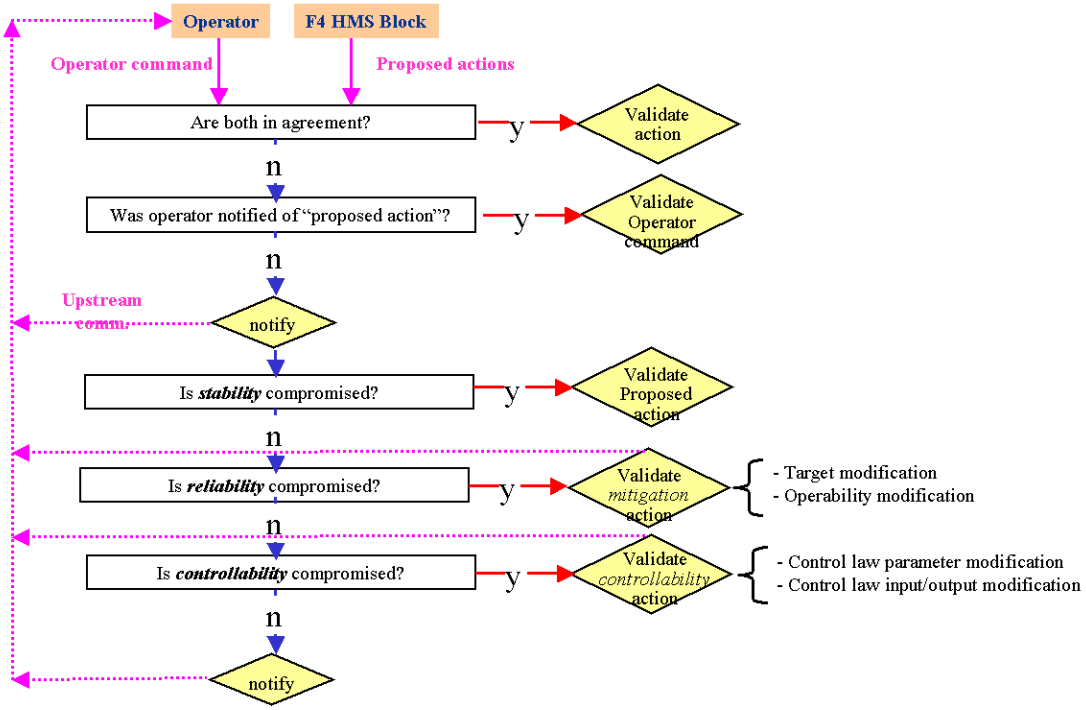

Figure V-1 Example of conflict-resolution logic for the F5 "Control Management" block

H. "Control Coordinator" evolution example

Figure V-2 shows a potential evolution of the "Control Coordinator" blocks using as a baseline the Reusable Rocket Engine Intelligent Control System (RREICS) advanced functional framework of reference[6].

It exemplifies, respectively per block, the use of functions within each of the F6 blocks for:

- Optimally calculate engine control parameters (e.g. thrust), determine possible and achievable control sequences, and smooth the effect of control law modifications;
- Estimate the different engine operation and model parameters required to perform the above calculations as well as those required for the control law modifications;
- Perform the implementation of a hierarchy of control law modifications based on established control strategies of increasing complexity. The blue-dashed line indicates blocks in Figure V-2 whose parameters can be modified.

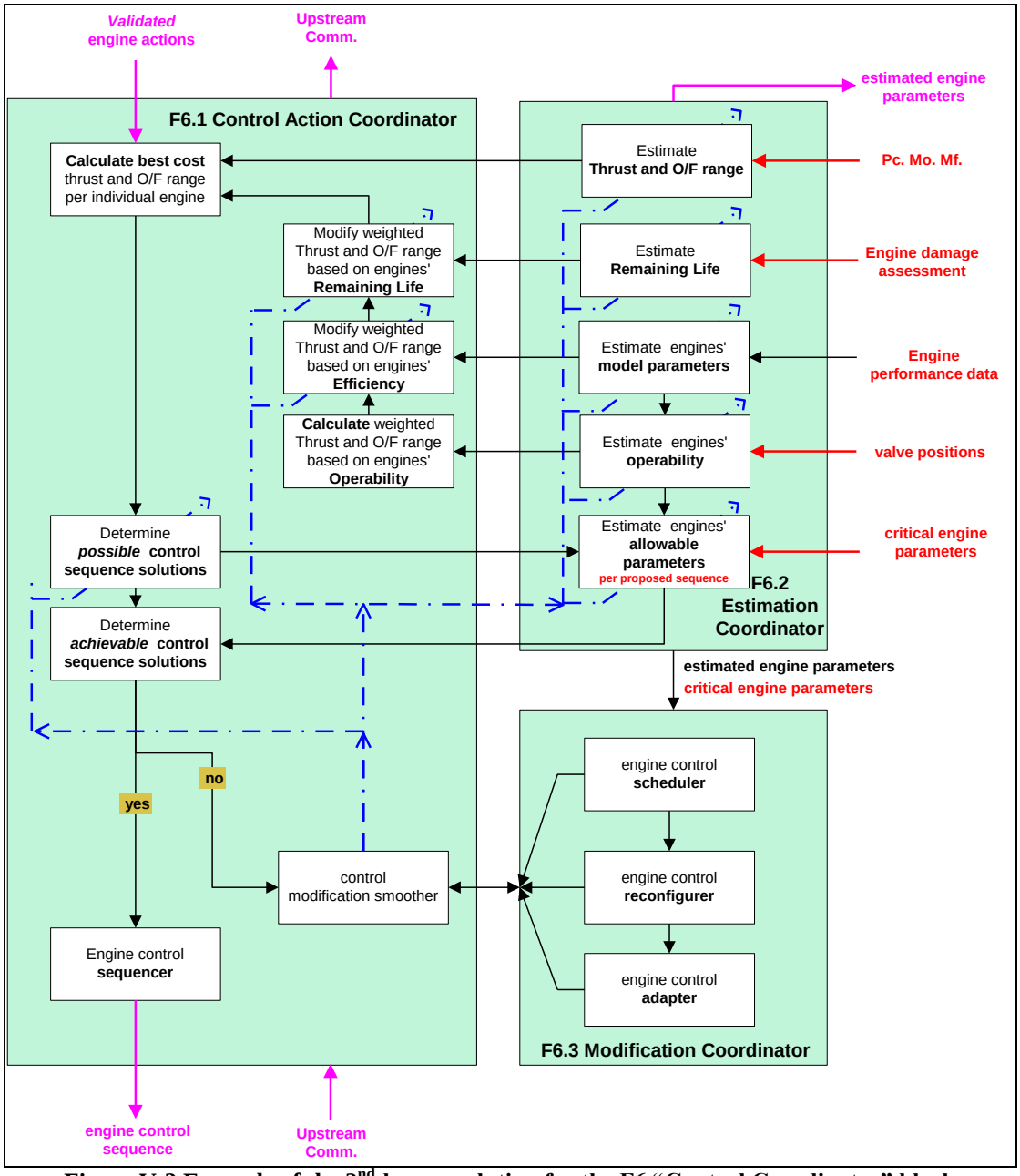

Figure V-2 Example of the 2nd-layer evolution for the F6 "Control Coordinator" block

VI. Conclusion

This article condenses the work performed to examine the HMS/control interaction issue under the auspices of the ESA's FLPP program. The proposed HCI architecture is conceptually very generic, follows established guidelines for similar type of "intelligent" architectures and its development is reconciled with the Health Management System architecture from SNECMA. The preliminary exemplification of the proposed HCI architecture has served to validate its adequacy while providing visibility and insight into the possible future needs concerning the development of associated technologies (control algorithms, SRC logic implementation, etc).

In conclusion, it has been shown that the HCI component is a critical element, at the same level as the HMS component, for the development of a reliable, robust and fault tolerant HMaS. This is so since a true HMaS architecture must effectively use the information from the HMS component (e.g. prognosis and FDI) in the control loop. This can be done in a safe manner only by using an HCI module in charge of managing and coordinating the HMS information and the engine status so that fault and/or abnormal engine conditions are recovered by the control laws and if not possible allow for graceful performance degradation –as opposed to direct engine shutdown and mission loss. This entails the use of modern control techniques which can actively accommodate, reconfigure and/or adapt to the fault/abnormal engine condition. The use of these control techniques is deemed necessary due to the continuous drive to provide higher performance and robust systems capable of multi-mission/orbit/payload.

Acknowledgments

This document has been produced within the context of SNECMA contract FLFRA N° 801/014 aimed at the study and development of the Next Generation Launcher (NGL) Preparatory activities in the framework of FLPP Period 2 – Step 1: "Main Stage Propulsion technology – 2nd set activities Health Monitoring System Activities".

References

[1] Le Gonidec, S., Musta, V., "Health Management System: Propulsion S/S Description," SNECMA EF-NT-1800E00-0008-SEPV-Ed2-v01.

[2] Saxena, A., Orchard, M.E., Zhang, B., Vachtsevanos, G., Tang, L., Lee, Y., Wardi, Y., "Automated Contingency Management for Propulsion Systems," European Control Conference 2007, Greece, July 2007

[3] Benoit, S., Bornert, P., Le Gonidec, S., Supié, P., "A diagnostic demonstrator: a platform for the evaluation of real time diagnostic data dedicated to space engines," 2009 Conference of the Society for Machinery Failure Prevention Technology.

[4] Meystel, A., Messina, E., "The Challenge of Intelligent Systems," 15th IEEE International Symposium on Intelligent Control (ISIC 2000), Greece, 2000

[5] An Introduction to Intelligent and Autonomous Control, Eds. Antsaklis, P.A, and Passion, K.M., Kluwer Academic Publishers, 1993

[6] Panossian, H., "Integrated Health Management and Adaptive Control Systems," The Boeing Company. Rocketdyne Propulsion & Power

[7] Litt, J.S., Simon, D.L., Garg, S., Guo, T.H., Mercer, C., Millar, R., Behbahani, A., Bajwa, A., Jensen, D.T., "A Survey of Intelligent Control and Health Management Technologies for Aircraft Propulsion Systems," NASA TM-213622, May 2005.

[8] Pilchen, G., Breteu, J., Caruana, J.N., Kauffmann, J., Ramusat, G., Sirbi, A., Tumino, G., "Future Launchers Preparatory Programme (FLPP) –Preparing for the Future through Technology Maturation and Integrated Demonstrators Status and Perspectives," 59th International Astronautical Congress, Scotland, 2008.

[9] Letourneur, Y., Leleu, F., Pinard, D., Krueger, J., Balduccini, M., "Status of Next Generation Expendable Launcher Concepts Within the FLPP Programme," 59th International Astronautical Congress, Scotland, 2008.

[10] Ramusat, G., Boggiatto, D., Francescani, D., "FLPP Technologies for a Future European Earth-to-Orbit Transportation System," 59th International Astronautical Congress, Scotland, 2008.

[11] Web page: ESA Propulsion activities. http://www.esa.int/SPECIALS/Launchers_Home/SEM1YAUTLKG_0.html

[12] Zhang Y., Jiang J., "Issues on Integration of Fault Diagnosis and Reconfigurable Control in Active Fault-Tolerant Control Systems," Safeprocess 2006.

[13] An Introduction to Intelligent and Autonomous Control, Eds. Antsaklis, P.A, and Passion, K.M., Kluwer Academic Publishers, 1993

[14] Nemeth, E., Anderson, R., Ols, J., Olsasky, M., "Reusable Rocket Engine Intelligent Control System Framework Design," NASA CR 187213, 1991

[15] Lorenzo, C.F., Musgrave, J.L. "Overview of Rocket Engine Control," NASA TM 105318, 1992

[16] Musgrave, J.L., Paxson, D.E., Litt, J.S., Merril, W.C., "A Demonstration of an Intelligent Control System for a Reusable Rocket Engine," NASA TM 105794, 1992

[17] Becker, G., Packard, A., Philbrick, D., and Balas, G. (1993). Control of parametrically-dependent linear systems: A single quadratic Lyapunov approach. Proc. of the American Control Conference, pages 2795–2799.

[18] Balas, G.J., "Linear, parameter-varying control and its application to a turbofan engine," International Journal of Robust Nonlinear Control 2002; 12:763–796

[19] Marcos, A., Balas, G.J., "Development of Linear Parameter Varying Models for Aircraft", Journal of Guidance, Control and Dynamics, Vol. 27, No. 2, pp. 218-228, 2004