Model-Based Distributed On-line Safety Monitoring

Amer Dheedan, Yiannis Papadopoulos
Department of Computer Science
University of Hull
Hull, United Kingdom,
{A.A.Aloqaili@2007. , Y.I.Papadopoulos@}hull.ac.uk

***Abstract*— On-line safety monitoring, i.e. the tasks of fault detection and diagnosis, alarm annunciation, and fault controlling, is an essential task in the operational phase of critical systems. Although current safety monitors deliver this task to some extent, the problem of effective and timely safety monitoring is still largely unresolved. In this paper, we propose a Distributed On-line Safety Monitor (DOSM) that can achieve a range of real-time safety monitoring tasks: fault detection and diagnosis, alarm annunciation and control of hazardous failures. The monitor consists of a Multi-agent Monitoring System (MaMS) operating on a Distributed Monitoring Model (DMM) that contains reference knowledge derived from off-line safety assessments, and a number of Distributed Data Structures (DDSs) that provide up-to-date sensory measurements. Guided by the knowledge contained in the DMM and real-time observations of the system provided by the DSSs, agents are hierarchically deployed and work collaboratively to integrate and deliver safety monitoring tasks, both locally at the sub-system levels and globally overseeing the overall behaviour of the system.**

Keywords-Fault Detection and Diagnosis; Optimal Alarm Annunciation; Fault Controlling; Multi-agent Monitoring System;

I. INTRODUCTION

Over the last 30 years, considerable work on model-based safety monitoring, has resulted in approaches that exploit knowledge about the normal operational behaviour and failure of a system. In the context of this work, models such as state-machines, goal trees, goal hierarchies and fault trees have been exploited and demonstrated their benefits as reference knowledge for system monitoring (for a comprehensive see [1]). Typically, these models incorporate deep knowledge of the target system and enable qualitative and quantitative (often probabilistic) reasoning about behavioural transitions, symptoms, causes and possible effects of faults [2, 3].

Recently, a centralised safety monitor [4] that exploits knowledge derived from the application of a semi-automated off-line safety assessment method and tool called Hierarchically Performed Hazard Origin and Propagation Studies (HiP-HOPS) [5] has been proposed. That knowledge is composed of two elements: (a) a hierarchy of state-machines describing the behaviour of the system, effectively capturing the normal and abnormal mode and state transitions of the system and its sub-systems; (b) a set of fault trees, which effectively represent diagnostic models that relate the symptoms of failure to ultimate root causes.

The motivation for that work has been the observation that, in the current industrial practice, vast amounts of knowledge derived in off-line safety assessments cease to be useful following the certification and deployment of a system. A key contribution of this work is that it brings this knowledge forward to the operational phase of a system and usefully exploits it for the purposes of on-line safety monitoring. The concept is potentially very useful. However, the monitor described in [4] is limited in its potential because it is monolithic and centralised, and therefore, has limited applicability in systems that have a distributed nature and incorporate large numbers of components that interact collaboratively in dynamic cooperative structures.

Recent work on Multi-agent Systems (MaS) shows that the distributed reasoning paradigm could cope with the nature of such systems. In [6], for example, a MaS has been exploited to increase the capacity of a diagnostic scheme of a large-scale system. MaS have also demonstrated prompt responses in detecting faults and diagnosing the underlying causes of failures in complex distributed chemical processes [7]. Despite these encouraging developments, serious operational hazards are still recorded in safety critical systems and disastrous failures do not seem out of the question. Accordingly, the problem of developing a robust on-line monitor is still debated mainly in terms of two aspects. One aspect concerns the type of knowledge that is required to inform the on-line reasoning of the monitor: should it be, for example, a set of rules defined by experts or should it be knowledge based on engineering models, and in the latter case, what kind of knowledge should such models contain [1, 8]? The second aspect of the problem arises from the increasingly distributed nature of modern systems and the inevitably complicated collaboration among their components. This aspect is concerned with overcoming the limitations of centralised and rigidly distributed monitors, and is, looking into employing intelligent monitoring agents as means for delivering flexible, timely, consistent and effective monitoring [1, 9].

In order to address the issues discussed above, this paper proposes a DOSM which combines the benefits of using knowledge derived in off-line safety assessments with the benefits of a collaborative distribution of MaS. The DOSM consists of a DDM derived from the HiP-HOPS safety assessment model, a MaMS incorporating a number of Belief-Desire-Intention (BDI) agents, and a set of DDSs. According to the architectural model of the target system, agents are hierarchically deployed as monitoring agents (MAGs) and each is provided with its portion of the DMM

 ISBN: 978-1-61208-174-8

and appropriate DSSs. By exploiting their portions of the DMM, MAGs reason on the operational parameters held by DDSs, to detect and assess the effects of deviations, diagnose the underlying causes of the detected deviations and automatically apply corresponding fault controlling measures. Moreover, in order to avoid alarm avalanches and latent alarms that may mislead the system operators [10, 11], MAGs are also able to optimise alarm annunciation by (a) suppressing unimportant and false alarms; (b) filtering spurious sensory measurements; (c) incorporating helpful alarm information, such as assessment of the operational conditions after the occurrence of the fault, guidance on controlling the occurred fault, and diagnostics of the underlying causes of failures.

Benefit of the proposed DOSM ranges from increasing the flexibility, composability and extensibility of on-line safety monitoring to ultimately developing an effective and cost-effective monitor for safety critical systems.

The rest of this paper is organised in the following sections: section two briefly describes the nature of modern critical systems and the requirements for representation of such systems for the purpose of safety monitoring. Section three presents the approach, and the role and architecture of the DOSM. To demonstrate the effectiveness of the delivered monitoring tasks, in section four, the DOSM is applied to an aircraft fuel system and some failure scenarios are discussed. Finally, section five draws a conclusion and proposes further work.

II. MODELLING SYSTEMS FOR MONITORING

Large scale and dynamic behaviour are two common aspects of modern critical systems, for example, modern transportation systems, manufacturing systems, chemical and power plants. While the large scale of these systems calls into question the ability of a monitor to deliver consistent monitoring over an architecture that may integrate thousands of components, dynamic behaviour mainly calls into question the ability of a monitor to distinguish between normal and abnormal operational conditions. More specifically, what is considered as normal in one mode or phase of operation of the system may simply be abnormal in another mode. A typical example of a "phased mission" system is an aircraft system which delivers a trip mission through a number of phases, which include pre-flight, taxiing, take-off, climbing, cruising, approaching, and landing. Thorough knowledge about the architectural components and the dynamic behaviour in each phase is essential to achieve effective safety monitoring.

In order to model the mutual relations among sub-systems and components in a system model, a hierarchical organisation is commonly used to arrange them in a number of hierarchical levels. Across those levels components appear as parents, children and siblings. As shown in Fig. 1, we classify those levels into three different types as follows: the lowest level (level*0*) is classified as the basic components (BC) level. The upper levels, which extend from level*1* to level*n-1*, are classified as sub-system (Ss) levels. Finally, the top level (level*n*) is classified as the system (S) level.

In order to model dynamic behaviour, one needs to understand the behaviour itself and how it is initiated. Typically, dynamic behaviour is an outcome of, normal operational conditions in which the system engages its components in different operational functions and structures, so that it can deliver different functionalities in different phases of operation. Given that sub-systems are abstractions that represent aggregations of BCs, signals upon which that structure of the system is altered are always initiated by BCs (even operators will initiate changes through components in a graphical user interface or hardware panel). Typically, upon a signal from a BC, a system controller may instruct other BCs to be engaged in a certain structure and deliver certain functions in collaboration. For example, during the cruising of an aircraft, the navigation sensors may convey signals to the navigator sub-system (NS) which in turn calculates and passes those signals to the flight control computer sub-system (FCCS). Assuming that it is time for launching the approach, the FCCS accordingly instructs the powerplant sub-systems to achieve the required thrust and the surface hydraulic controller sub-system to achieve the required body motions. Accordingly, we define the case in

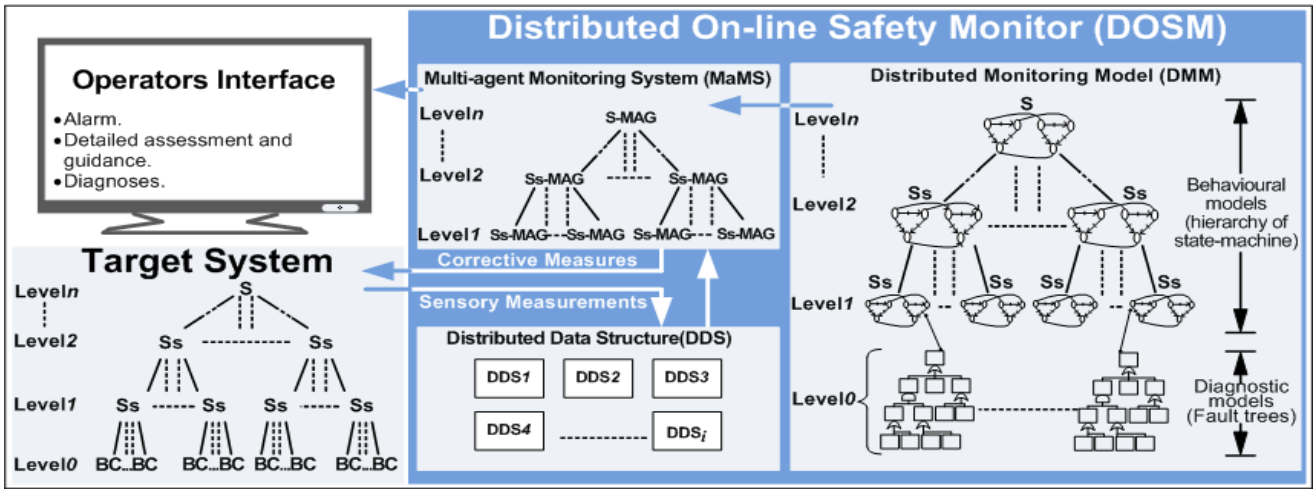

Figure 1. Target System and DOSM Position and Basic Constituents.

which the system uses a certain operational structure to deliver certain functionality as a *mode*.

Beyond being the result of normal changes in function and structure, dynamic behaviour also arises from the need to respond to and tolerate the faults of basic components. Fault tolerance is typically achieved using the following strategies: (a) recovery from a permanent fault usually achieved with functional or hardware redundancy, e.g. the fault of one engine of a two-engine aircraft can be compensated by the other engine; (b) the ability of tolerating the fault for a while until recovery of a healthy state can be achieved, e.g. temporary faults that are caused by ionisation, radiation, electromagnetic interference, or transient hardware failures, are often self-correcting, while certain types of tolerable software faults may be corrected with a controller restart.

It could, therefore, be said that during a mode, a system may appear in different health *states* which can be classified into two types. The first type is the *Error-Free State* (EFS) in which the system or a sub-system functions healthily. The second type is the *Error State* (ES), which in turn is classified into three different states: (a) Temporary Degraded or Failure State (TDFS) in which there is one or more functional failure, but corrective measures can be taken to resume a healthy state; (b) Degraded State (DS) in which a permanent fault has occurred, but part of the intended functionality is still delivered; (c) Failed State (FS) in which the component or system has lost its entire functionality.

In order to track dynamic behaviour, events that result in the normal and abnormal behavioural changes should be continuously monitored. The hierarchical level at which such events could be monitored most effectively is level*1*, i.e. one level above the level of BCs. This can easily be justified, because at that level low level events can be contextualised and could be identified as either normal or abnormal. For instance, the decreasing of velocity and altitude seem normal during the approaching mode of an aircraft, since the FCCS has already launched that mode. Excluding knowledge about the mode and focusing only on the measurements provided by the relevant sensors would certainly result in misinterpreting system behaviour. Specifically, decreasing velocity and altitude would appear as a malfunction and thus a misleading alarm would be released. This being the case, level*1* is preferable rather than any higher level, since it is the level at which a malfunction is detected while in its early stages. Finally, due to the number of the basic components, which is potentially huge, monitoring and reasoning about those events at level*0* is computationally expensive or even unworkable, whereas level*1* offers the required context and knowledge of local mode.

III. DISTRIBUTED ON-LINE SAFETY MONITOR (DOSM)

As shown in Fig. 1, the DOSM lies between the target system and the interface of the system operators. During normal operation, the role of the DOSM is confined to providing simple feedback about those conditions. The DOSM plays its role during abnormal operating conditions, which are triggered by and follow the occurrence of faults. In that role the DOSM achieves three real-time time safety tasks: fault detection and diagnosis, optimising alarm annunciation and automatic control of faults. In order to achieve those tasks, the DOSM employs three elements (see also Fig. 1): (a) a DMM which holds the reference monitoring knowledge, in other words, the DMM references the MAGs which in turn reason and achieve the three safety tasks; (b) Distributed Data Structures (DDS), which hold the necessary sensory measurements used by MAGs in order to monitor operational parameters and reason on the operational conditions of the monitored system; (c) A MaMS which is a set of BDI agents that are deployed over the components of the system to reason locally and collaborate globally towards achieving the three safety tasks.

A. Distributed Monitoring Model (DMM)

MAGs should be, in the first place, able to track the operational behaviour of the monitored components over different states, i.e. EFSs and ESs. Accordingly, both normal and abnormal behaviour should be modelled and recorded in the DMM. For that purpose, state-machines provide the means of recording behaviour at all levels of the architectural hierarchical decomposition of the system (see Fig. 1.). Accordingly, the spine of the DMM is a hierarchy of state-machines that describes dynamic behaviour. In those state-machines, every EFS or ES is represented as a state and every event whose occurrence results in a state transition is represented as a trigger event.

Practically, there are relationships among every sub-system and its parent and child components. For instance, the failure of a component within a sub-system may trigger a transition of the sub-system in a recovery state where another component changes function to compensate for the initial failure. Such relationships can be implemented in the state-machines in a similar way to the following example:

Let us assume that the flight control computer sub-system (FCCS) and power plant sub-system (PPS) are siblings and have the same parent, the aircraft control sub-system (ACS). During the cruising mode, an event may trigger a state transition to EFS of the approaching mode in the state-machine of the FCCS. That EFS appears as a trigger event whose occurrence triggers a state transition in the state-machine of the ACS, i.e. the parent, to the EFS of the approaching mode. The latter EFS appears, similarly, as a trigger event whose occurrence triggers a state transition in the state-machine of the PPS, i.e. a child, to the EFS of the approaching mode.

Similarly, ESs of the children could also trigger state transitions in the state-machines of the parents and vice versa. Consider, for example, when an engine of a two-engine aircraft fails; the FS of that engine triggers a state transition to the DS in the state-machine of the PPS. That DS, in turn, triggers a state transition to new EFS of the operative engine in which the lost functionality of the faulty engine is compensated.

In the state-machine of the sub-systems of level*1*, trigger events appear as (a) events that are originated by the BCs of level*0*, which might be failure, corrective or normal events; (b) events that are originated by the parent states, such as the EFS or ESs of the parent. In the state-machine of a sub-

system of the levels extending from level*2* to level*n-1*, trigger events appear as EFSs and ESs of the parent and the children. Finally, in the state-machine of the system, i.e. level*n*, trigger events appear as EFSs and ESs of the children.

Knowledge about the normal behaviour, i.e. EFS and normal events, of the system and its sub-systems can be obtained from design models, such as Data Flow Diagrams (DFD), Functional Flow Block Diagram (FFBD) and models in the Unified Model Language (UML) that model the system during the design life cycle. Knowledge about abnormal behaviour, i.e. ESs, abnormal events, assessment, guidance, and corrective measures, can be obtained by applying the Functional Failure Analysis (FFA) or HAZard and OPerability study (HAZOP) techniques on those models.

During the monitoring time, MAGs monitor only trigger events whose occurrence triggers transitions from the current state in the state-machine. As such, the computational load of the MAGs would be less and prompt responses to the occurrence of the events would be obtained.

In the state-machines of the sub-systems of level*1*, every failure event would be associated with (a) an alarm statement that would be quoted and provided to the operators upon the occurrence of the failure event; (b) corrective measures that can be applied to control the failure; (c) diagnosis, if the failure and the underlying cause are in a one-to-one relationship the cause would be associated, otherwise a diagnostic process should take place. Note, some corrective measures might be achievable only after diagnosing the underlying causes. In the state-machines of higher level sub-systems, normal and abnormal events are associated with a field of (a) assessment of the consequent operational conditions; (b) guidance on directing the hazards at that level. Knowledge of those fields can be obtained from the HAZOP.

A failure event and its underlying cause might not always be in a one-to-one relationship. Therefore, a diagnostic model that can relate failure events to their underlying cause is needed. The fault tree, a popular model used in safety assessments, can be used as a diagnostic model as it logically records the propagation paths and the associated symptoms of failure a long with underlying causes. In HiP-HOPS, fault trees are automatically constructed from the topology of a system and local failure logic specified at component level. This method can be applied to construct diagnostic fault trees for failure events that appear as trigger events in the state-machines of level*1* sub-systems. Corrective measures could also be incorporated in the failure mode nodes of the fault tree.

As shown above, knowledge encoded in the DMM is obtained from the design models and by applying classical manual safety analysis techniques (FFA, HAZOP, FMEA, Fault Tree analysis) [12] or more modern semi-automatic safety analysis techniques (HiP-HOPS) [5]. Hence, it could be said that a safety assessment model could be useful to derive a DMM after (a) associating the abnormal events with the alarm, controlling and diagnosis knowledge; (b) augmenting the states of the state-machines by assessment and guidance fields and the diagnostic model with the required corrective measures; (c) formalising the trigger events of the state-machine and the symptoms of the diagnostic model as monitoring expressions that could be evaluated computationally in real time. The deriving process would contribute essentially to providing the DOSM with thorough and consistent monitoring knowledge. Note that in this paper we adopt the HiP-HOPS as a safety assessment tool to produce the DMM.

B. Formal Monitoring Expressions and Distributed Data Structure (DDS)

Low level events that monitor the physical process and trigger state transition in the state-machines at Level*1*, should be formalised as monitoring expressions that reference parameters of the physical process. Through evaluating those expressions, the occurrence of the corresponding trigger events or symptoms could be verified. In the formalisation process, an event or a symptom is expressed as a constraint. In its simple form, a constraint consists of three main parts: (a) the status of operational conditions which is either a state of a child or the parent or a sensory measurement defined by the identifier of the relevant sensor; (b) a relational operator – equality or inequality; (c) a threshold whose violation results in evaluating that expression with a true truth value, i.e. the relevant event or symptom occurs. Thresholds might appear as a numerical or Boolean value.

Simple constraints may suffice for simple monitoring tasks. In general, though, events may require more complicated forms of constraints to be evaluated. In turn, such constraints might require (a) the status of a parameter to be calculated over a number of sensory measurements; (b) two operational operators, when the threshold is a range of values rather than a single value; (c) a threshold that represents a sensory measurement or a calculation of more than one measurement. Moreover, the status of parameters and the threshold might be calculated to find the average of the change of a quantity over an interval (Δt), i.e. differentiation, or the volumes from different sensory measurements at definite timings, i.e. integral calculus.

For the evaluation process of such monitoring expressions, we pre-declare a number of data structures that could hold satisfactory sensory measurements and the result of the calculation and the evaluation process. For every sub-system of level*1*, a DDS would be allocated to hold those structures; as shown in Fig. 1.For holding historical sensory measurements we use an updatable buffer of one-dimension array data structure that could hold two or more up-to-date sensory measurements. Such a structure is updated every Δt by (a) inserting the current measurement, which is collected at the current time (T) from the relevant sensor; (b) shifting out the earliest measurement, which is collected at $T-2\Delta t$ in the past. As such, that structure holds two (or more) measurements collected at current time T and $T-\Delta t$ in past.

Sensors may deliver spurious measurements because of (a) their own transient failures; (b) mode changes, which might be followed by an interval of unsteady behaviour in which the monitored parameter may temporarily fluctuate outside normal thresholds. One way of filtering out such spurious sensory measurements is by evaluating the

monitoring expressions successively over a filtering interval and based on a number of measurements. The final result of that evaluation is obtained by making accumulative conjunctions among the successive evaluations. If the final result is s true truth value, this means that the delivered measurements remain the same over the filtering interval, which is a confirmation that a parameter is persistently out of threshold and a sign of a persistent anomaly present - as opposed to a spurious measurement or a transient anomaly. The filtering interval of every expression is defined by examining both the conditions that may result in spurious measurements and the time intervals at which the involved sensors are requested by the monitor.

A three-value technique: 'True', 'False', and 'Unknown', is also employed to save evaluation time and produce earlier results in filtering spurious measurements and in the context of incomplete sensory data without violating the evaluation logic. Consider, for example, the following two expression forms:

Expression OR (Expression, ⊿t) (1)
Expression AND (Expression, ⊿t) (2)

Evaluating either of those expressions; (1) or (2), may require waiting time equal to Δt, i.e. until evaluating *(Expression, ⊿t)* part, regardless of the instant evaluation of the '*Expression'* part of either of the expressions. Knowing that the disjunction of 'True' with 'Unknown' is 'True' and the conjunction of 'False' with 'Unknown' is 'False', both expressions; (1) and (2), can be evaluated instantly. Therefore, in cases in which the '*Expression'* part of expression (1) is evaluated to 'True' and the '*Expression'* part of expression (2) is evaluated to False, both (1) and (2) could be evaluated instantly to 'True' and 'False', respectively.

C. Multi-agent Monitoring System (MaMS)

As shown in Fig. 1, MAGs are deployed over the sub-systems and the system, and appear as a number of subsystem MAGs (Ss-MAGs) and a system MAG (S-MAG), respectively. Fig. 2 shows a general illustration of the MAG. By perceiving the operational conditions and exchanging messages with other MAGs, a MAG obtains the up-to-date belief, deliberates among its desires to commit to an intention and achieves a means-ends process to select a course of action, i.e. plan. The selected plan is implemented by the MAG as actions towards achieving the monitoring tasks locally and as messages sent to other MAGs towards achieving those tasks globally. Upon having a new belief, MAG achieves a reasoning cycle; deliberation and means-ends processes.

Each Ss-MAG of level*1* would have its perception by perceiving (a) its own portion of the DMM which consists of a state-machine and a set of fault trees; (b) the corresponding DDS in which events and symptoms appear as expressions and are evaluated; (c) messages that are received from the parent to inform the Ss-MAG about the new states and the siblings, in which they either ask for or tell the given Ss-MAG about global sensory measurements, as they share their DDSs whenever needed. The main desires of a Ss-MAG of level*1* are to achieve local safety monitoring tasks and global collaboration and coordination. On the former desire, the intentions are to track the behaviour of the assigned sub-system and to provide the operators with alarms, assessment, guidance, and diagnostics and achieve automatic fault controlling. On the latter desire, the intention would be achieved by (a) informing the parent about the new states; (b) telling or asking the siblings about global sensory measurements.

Each Ss-MAG of the levels extending from level*2* to level*n-1*, would have its perception by (a) perceiving its own portion of the DMM which consists of a state-machine of the assigned sub-system, and (b) messages received from the parent and the children to tell about their new states. The main desires of the Ss-MAGs of those levels are to achieve local safety monitoring tasks and global collaboration. On the former desire, the intentions are to track the operational behaviour of the assigned sub-system and to provide the operators with assessment and guidance of their levels. On the latter desire, the intention would be achieved by telling, i.e. sending messages to, the parent and the children about the new states. The perceptions, desires and intentions of the S-MAG are similar to those of the Ss-MAGs of the levels extending from level*2* to level*n-1*. The only difference is that S-MAG has no parent to exchange messages with.

According to the Prometheus approach and notation for developing MaS [13], Fig. 3 shows the collaboration protocols among MAGs to track the operational behaviour of the monitored system. Fig. 4, similarly, shows the collaboration protocol among the Ss-MAGs of level*1* in which they share their sensory measurements globally.

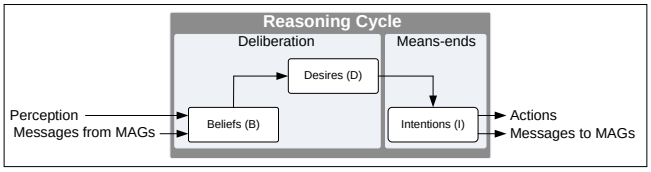

Figure 2. A general illustration of the MAG.

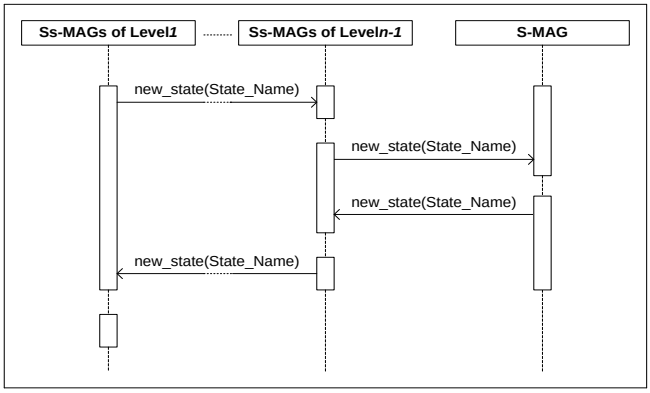

Figure 3. MAGs' collaboration protocol across the hierachical levels.

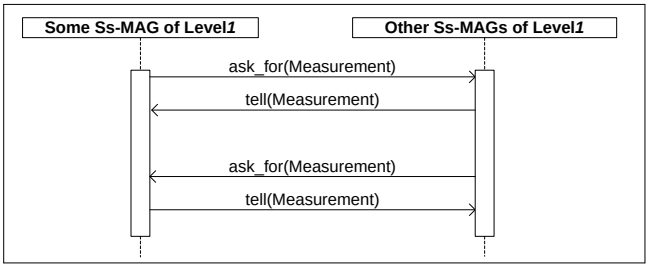

Figure 4. Collaboration protocol among Ss-MAGs of level*1*.

IV. CASE STUDY: AIRCRAFT FUEL SYSTEM (AFS)

Fig. 5 shows the physical illustration and the basic components of the AFS. The AFS functions to maintain safe storage and even distribution of fuel in two operational modes. The first is the consuming mode in which the AFS provides fuel to the port and starboard engines of a two-engine aircraft. The second is the refuel mode. During the consuming mode, and to maintain the central gravity and stability, a control scheme applies a feedback-control algorithm to ensure even fuel consumption across the tanks.

Another algorithm is applied similarly to control the even distribution of fuel injected from the refuelling point to the tanks during the refuel mode. The AFS is arranged in four sub-systems: a central deposit (CD), left and right wing (LW, RW) deposits and an engine feed (EF) deposit which connects fuel resources to the two engines.

In order to tolerate faults, an active fault-tolerant controller strategy is implemented. More specifically, in the presence of faults there are alternative flow paths, i.e. different configurations can potentially connect the two engines to the available fuel resources.

As shown in Fig. 6, five monitoring agents are deployed over the AFS as follows: four MAGs monitor the four sub-systems; EF-MAG, CD-MAG, LW-MAG, and RW-MAG. The fifth is AFS-MAG which monitors the entire FS. The DOSM is implemented by Jason interpreter; it is an extended version of AgentSpeak programming language [14].

In order to achieve fault detection, the four MAGs update their DDSs and evaluate the monitoring expressions and thus detect any parametric deviations at level*1*. Consider, for example, the deviation "no fuel flow to starboard engine"; this deviation could be detected locally by the EF-MAG, while the deviation of imbalance between the LW and RW deposits is detected through communicating sensory measurements globally between LW-MAG and RW-MAG.

Diagnosis of the underlying causes of a deviation is triggered when a deviation is detected. Through exploiting fault tree models and combining between depth-first and heuristic parse strategies, Ss-MAGs traverse and relate the top event in a tree to its bottom faulty components. Consider, for example, the deviation "no fuel flow to starboard engine". EF-MAG evaluates symptoms in the relevant fault tree to track the propagation path. The expected diagnosed causes could be one or more of the fault modes of EF basic components; likely, a pipe blockage, an inadvertent closure or a fault of a valve, a pump fault, or no fuel in the rear tank.

To achieve automatic fault controlling, corrective measures are provided across the DMM, some of which could be taken directly by a MAG and others may require global collaboration. Consider, for example, when the deviation "no fuel flow to starboard engine" is diagnosed with the cause of an inadvertent closure of valve VF5; the possible corrective measure that can be taken locally by EF-MAG is to instruct the controller to reopen that valve. However, if that fails to rectify the situation, then a global action should be taken through the following steps: (a) EF-MAG transits to a FS and tells the AFS-MAG about that state; (b) AFS-MAG, in turn, executes that state on its state-machine, achieves the corresponding transition and tells the four MAGs about the resulted state; (c) the four MAGs, in turn, execute the received state on their state-machines. According to their new states the four MAGs apply new flow rates for every sub-system. Thus, the FSA configuration will be changed and both engines will be fed from the front tank.

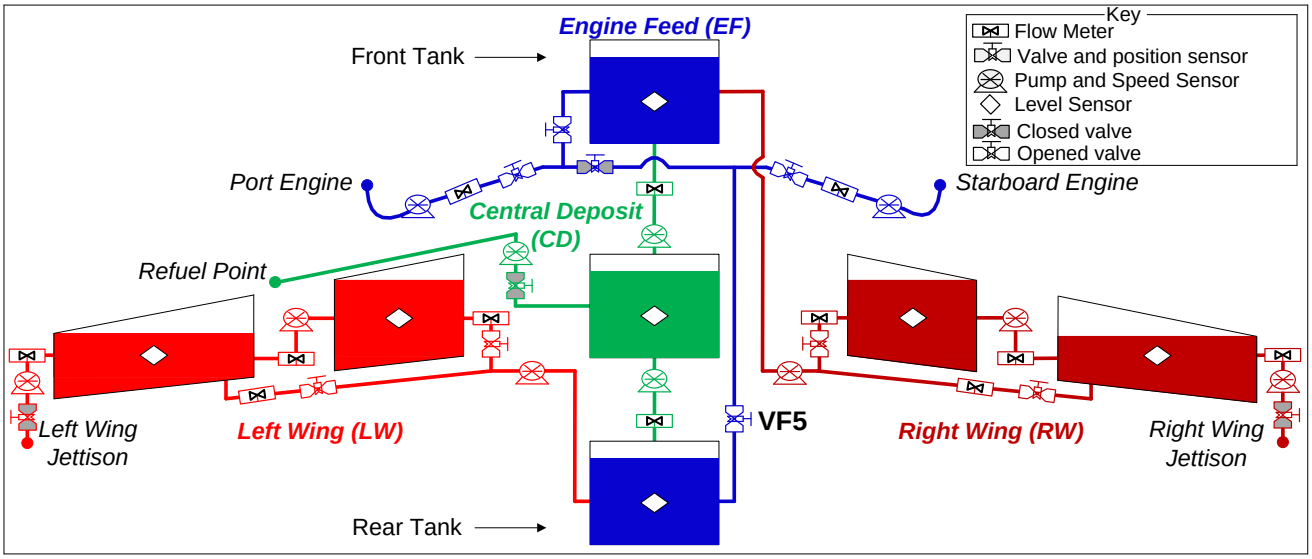

Figure 5. Physical illustration of aircraft fuel system.

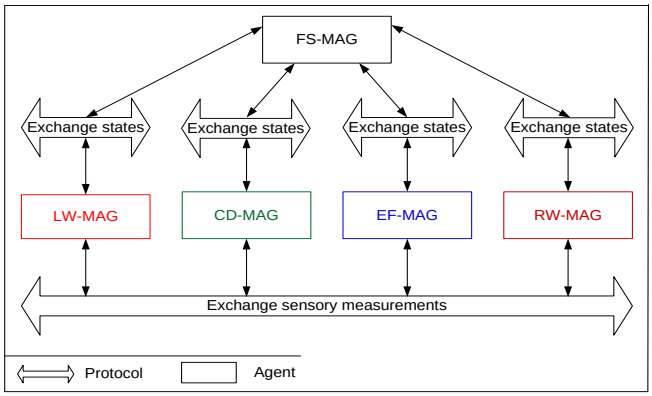

Figure 6. Architecture of the MaMS of the AFS.

When the occurrence of the failure event "no fuel flow to starboard engine" is verified, EF-MAG alarms and provides the pilots with the corresponding assessment and guidance. Moreover, as that failure triggers a state transition in the state-machine of the EF sub-system, which in turn triggers a transition in the state-machine of the AFS, assessment and guidance from the new state of the AFS would also be provided to the crew pilots, i.e. multi-level assessment and guidance.

V. CONCLUSION AND FUTURE WORK

This paper proposed a distributed on-line safety monitor (DOSM) based on a multi-agent system and knowledge derived from model-based safety assessment. Agents exploit that knowledge to deliver a range of real-time safety monitoring tasks which have been briefly discussed in the context of a study of an aircraft fuel system. The monitor can detect symptoms of failure on process parameters as violations of simple constraints, or deviations from more complex relationships among process parameters, and then diagnose the causes of such failures. With appropriate timed expressions, the monitor can filter normal transient behaviour and spurious measurements. By exploiting knowledge about dynamic behaviour, the monitor can also determine the functional effects of low-level failures and provide a simplified and easier to comprehend functional view of failure. Finally, by knowing the scope of a failure, the monitor can apply successive corrections at increasingly abstract levels in the hierarchy of a system.

Despite encouraging results certain research issues remain to be investigated. The first is that the quality of the monitoring tasks and the correctness of the inferences drawn by the monitor depend mainly on the integrity and consistency of the DMM. The validation of the DMM, therefore, is an area for further research. Secondly, more work is needed on uncertainty of the diagnostic model and the application of the three-value logic. For that purpose, the incorporation of Bayesian Networks will be investigated in the future.

VI. ACKNOWLEDGEMENT

This work was supported partly by the EU Project MAENAD (Grant 260057).

REFERENCES

[1] A. Dheedan and Y. Papadopoulos, "Multi-Agent Safety Monitoring System," Proc. of 10th IFAC Workshop on Intelligent Manufacturing Systems (IMS'10), Portugal, Lisbon, 1-2 July 2010, pp. 93-98.

[2] V. Venkatasubramanian, R. Rengaswamy, K. Yin and N. S. Kavuri, "A review of process fault detection and diagnosis: Part II: Qualitative Models and Search Strategies," Computers and Chemical Engineering 27(3), 2003, pp. 313-326.

[3] I. Monroy, R. Benitez, G. Escudero and M. Graells, "A semi-supervised approach to fault diagnosis for chemical processes," Computers and Chemical Engineering, vol. 34(5), 2010, pp. 631-642.

[4] Y. Papadopoulos, "Model-based system monitoring and diagnosis of failures using state-charts and fault trees," Reliability Engineering and System Safety, vol. 8(3), 2003, pp. 325-341.

[5] Y. Papadopoulos, J. McDermid, R. Sasse and G. Heiner, "Analysis and synthesis of the behaviour of complex programmable electronic systems in conditions of failure," Reliability Engineering & System Safety, 71 (3), 2001, pp. 229-247.

[6] M. Mendes, B. Santos, and J. Costa, "A matlab/Simulink multi-agent toolkit for distributed networked fault tolerant control systems," Proc. 7th IFAC symposium on Fault Detection, Supervision and Safety of Technical Processes, 30 June - 3 July 2009, in Barcelona, Spain, pp. 1073-1078.

[7] Y. S. Ng, and R. Srinivan, "Multi-agent based collaborative fault detection and identification in chemical processes," Engineering Applications of Artificial Intelligence, vol 23(6), 2010, pp 934-949.

[8] V. Venkatasubramanian, R. Rengaswamy, K. Yin and N. S. Kavuri, "A review of process fault detection and diagnosis: Part III: Process History based methods," Computers and Chemical Engineering, 27(3), 2003, pp. 327-346.

[9] C. J. Wallace, G. J. Jajn and S. D. J. Mcarthur, "Multi-Agent System for Nuclear Condition Monitoring' Proc. of 2nd International Workshop on Agent Technologies for Energy System (ATES'11), a workshop of the 10th International Conf. of Agent and Multi-agent System (AAMAS'11), 2nd of May 2011, in Taipei, Taiwan, pp. 17-23.

[10] E. J. Trimble, "Report on the Accident to Boeing 737-400 G-OBME near Kegworth, Leicestershire on 8 January 1989" Department of Transport Air Accidents Investigation Branch, Royal Aerospace Establishment. London, HMSO, 1990. Available from http://www.aaib.gov.uk/cms_resources.cfm?file=/4-1990%20G-OBME.pdf [Accessed 5th of March 2011].

[11] U.S.NRC, "Fact sheet: Backgrounder on the Three Mile Island Accident," United State nuclear regulatory commission (U.S.NRC), 2008. Available from: http://www.nrc.gov/reading-rm/doc-collections/fact-sheets/3mile-isle.pdf [Accessed 5th of March 2011].

[12] D. Pumfrey, The Principled Design of Computer System Safety Analyses, DPhil Thesis. University of York, 1999.

[13] L. Padgham, and M. Winikoff, Developing Intelligent Agent Systems: a Practical Guide. UK, Chichester:Wiley, 2004.

[14] R. Bordini, J. Hubner, and M. Woorldridge, Programming Multi-Agent Systems in AgentSpeak using Jason. UK, Chichester: Wiley, 2007.

 ISBN: 978-1-61208-174-8