On-line Safety Monitor Based on a Safety Assessment Model and Hierarchical Deployment of a Multi-agent System

Amer A Dheedan
Department of Computer Science
Delmon University
Manama, Bahrain
amer@delmon.bh

Abstract – **The operational safety of critical systems, such as nuclear power plants, aircraft and chemical processes, is typically maintained by the delivery of three real-time safety tasks: fault detection and diagnosis, alarm annunciation and fault controlling. Although current on-line safety monitors play this role to some extent, the problem of consistent and timely task performance is largely unresolved. An aspect of the problem is attributed to the type of monitoring knowledge that informs the real-time reasoning; should it be derived, for example, from off-line design models or the operational context of the monitored system? Another aspect is attributed to whether the monolithic or distributed monitor is able to scale up and cope with the complicated and distributed nature of modern critical systems. To address the problem, this paper develops a distributed on-line safety monitor from monitoring knowledge derived from a safety assessment model of the monitored system and a multi-agent system. Agents are deployed hierarchically according to the architecture of the monitored system and they are provided with portions of the knowledge to reason locally over the conditions of the monitored components and collaborate globally to reason over the overseen behaviour of the entire system. The paper also tests the monitor via an application to an aircraft fuel system and evaluates the approach and results by contrasting them with those of earlier work.**

Keywords-fault detection and diagnosis; alarm annunciation; fault controlling; prognosis; sensory measurements filtration and validation

I. INTRODUCTION

This article presents an extension of the work that has already been presented in [1].

Dating back to the early 1980s, research effort has focused on the development of advanced computer-based monitors. Since then, computerised on-line safety monitors started to appear as computer systems that are installed in the control rooms of plants and flight decks of aircraft [2], [3].

Computerised monitors have been approached differently in terms of (a) their capacity to deliver three safety tasks: fault detection and diagnosis, alarm annunciation and fault controlling; (b) their architectural nature, monitors could be developed from multi-agent (distributed) or monolithic (centralised) reasoning.

A. Fault Detection and Diagnosis

Fault detection and diagnosis techniques are typically developed as model-based and data-based techniques [4], [5]. The distinction between these techniques lies in the way of deriving the knowledge that informs the real-time reasoning. Specifically, knowledge of model-based techniques is derived from off-line design models, such as Data Flow Diagrams (DFD), Functional Flow Block Diagrams (FFBD), or more recently from models defined in the Unified Modelling Language (UML). Knowledge about the normal behaviour of the monitored system can be obtained directly from these models. To obtain knowledge about abnormal behaviour, analysis techniques such as HAZard and OPerability study (HAZOP), Functional Failure Analysis (FFA), and Failure Mode and Effect Analysis (FMEA) are used to analyse the design models [6].

Knowledge of data-based techniques, on the other hand, is derived from the on-line context of the monitored system. Knowledge about the normal behaviour is obtained by empirical experiment of fault-free operation of the monitored system. To derive knowledge about abnormal behaviour, possible faults of the basic components are identified (by applying the FMEA to the basic components) and injected experimentally in the operational context. The resulting symptoms and ultimate effects on the functionality of the system are then modelled [7].

In both model-based and data-based techniques, monitoring knowledge is applied to real-time reasoning in executable format as monitoring models. To deliver fault detection and diagnosis, a monitoring algorithm executes the monitoring model by instantiating, evaluating and verifying modelled conditions with real-time sensory data.

Model-based techniques have exploited a wide range of monitoring models, such as Goal Tree Success Tree (GTST) [8], [9], [10], fault trees [11], [12], [13], signed direct graph [14], [15], diagnostic observers [16], [17] and parity equations [18], [19], [20], [21]. Similar variety can be seen with the data-based techniques. Consider, for example, rule-based expert systems [22], [23], [24], [25], qualitative trends analysis [26], [27], [28], artificial neural networks [29], principal component analysis [30], [31] and partial least squares [32].

B. Alarm Annunciation

Alarm is the key means to bring the occurrence of faults to the attention of the operators [33]. Developing an alarm technique involves the consideration of alarm definition, alarm processing, and alarm prioritisation and availability [34], [35].

Alarm definition concerns the definition of mode dependency, which is required to establish a distinction between events that occur due to normal operation and others that occur due to faults, so confusing alarms can be eliminated. State-machines [12], operational sequence diagrams [36] and system control charts [33] are among the models that have been exploited to address this issue. Alarm definition also concerns the definition of an effective threshold, the violation of which would result in verifying the occurrence of an event. Thresholds should not be too sensitive and result in false verification, and at the same time, not too relaxed, which would result in late verification and depriving the operators of knowledge about the actual conditions [35].

In alarm processing, distinction among genuine, consequent and false alarms should be achieved. While genuine alarms should be released, consequent and false alarms should be filtered out to avoid confusing alarm avalanches. Cause-consequent analysis of the design models can establish the distinction between causal alarms that concern the maintenance operators and consequent alarms that concern the pilot operators [37], [38]. Sensory measurement validation can eliminate the potential for false alarms. Recent techniques achieve validation through analytical redundancy among sensors, e.g., see [39], [40], [41], [42]. On the other hand, earlier techniques depended on hardware redundancy [43], i.e., redundant sensors. Although redundancy techniques offer adequate robustness, their applicability is limited since they demand increase in cost, weight and volume.

Alarm prioritisation and availability is the process in which alarms are given priorities according to their importance, so they are selected and announced accordingly [35]. The highest priority is always given to safety consequences [44]. Dynamic and group-presentation are two strategies to prioritise alarms. In dynamic prioritisation alarms might be prioritised by (a) different colours (red, amber, magenta) [35]; (b) different severities, such as catastrophic, critical, marginal and insignificant [45]; (c) presenting the highest priority alarms and hiding and facilitating optional access to the less important ones [35]. Group-presentation takes advantage of the screen display (LCD) to present alarm information in windows according to the hierarchical architecture of the monitored process and the importance of the relevant functionality [46]. Windows may allow operator interaction through facilitating silencing of alarms' sound or suppressing illuminated alarms' lights [47].

C. Fault Controlling

Practically, fault controlling is considered in parallel with the controlling process. Fault controlling is implemented in two different approaches. The first is by manual interference of the system's operators, in which further to the need of an advanced alarm technique, the operators should also be trained and provided with guidance on controlling faults [48], [49], [50].

The other approach is achieved automatically by a computerised controller, which is commonly called a Fault-Tolerant Control System (FTCS) [51], [52]. FTCSs, in turn, are classified into Active Fault-Tolerant Controlling (AFTC) and Passive Fault-Tolerant Controlling (PFTC) [48].

Research on the AFTC has been motivated by the aircraft flight control system [52]. Faults are controlled by selecting and applying the corresponding corrective procedure. An engine fault of a two-engine aircraft, for example, requires a procedure of: (a) cutting-off fuel flow to the faulty engine; (b) the achievement of cross feed from the tanks that were feeding the faulty engine; (c) applying the corresponding command movements to control the surface and compensational instructions to the operative engine [53].

PFTC relies mainly on redundant components, such as multiple control computers and backup sensors and actuators [54], [55]. Typically, provision of redundant components is implemented by hot or cold standby redundancy. In hot standby redundancy, the system is provided with parallel redundant components, which operate simultaneously (powered up) and each component monitors the output of the other(s). Should any of them fail, the others take over. In cold standby redundancy, only one component is on-line (powered up) and other copies are on standby (powered down). Should the on-line component fail, it is powered down and one of the standby components is powered up by a controller [56].

D. Monolithic and Multi-agent On-line Safety Monitors

Monolithic and multi-agent are two common classes of computerised monitors. The monolithic monitor in [12] has been developed from a monitoring model derived from the application of the Hierarchically Performed Hazard Origin and Propagation Studies (HiP-HOPS) safety assessment technique [57]. The model consists of a hierarchy of state-machines (as a behavioural model) that records the behaviour of the monitored system and its sub-systems and a number of fault trees as diagnostic models that relate detected faults to their underlying causes. The concept was motivated by observation of the fact that immense off-line knowledge ceases its benefit and is rendered useless after certifying the safe deployment of critical systems. The exploitation of that knowledge in the context of on-line monitoring results accordingly in an effective and cost-effective monitoring model.

A quite similar monolithic monitor is developed in [13]. The only difference is that the hierarchy of the state-machine is replaced with the control chart of the monitored system and fault trees are maintained as the diagnostic models.

The main limitation of these monitors is that they are based on a monolithic concept in which all monitoring of a plant is delegated to a single object or device. This does not align well with the distributed nature of most modern systems. Systems are typically implemented as a set of sub-systems, which exist in a complex cooperative structure and coordinate to accomplish system functions. Systems are also typically large and complex and show dynamic behaviour that includes complex mode and state transitions.

As a result, such systems need a distributed mechanism for safety monitoring; first it is essential to minimise the time of on-line failure detection, diagnosis and hazard control;

second, a distributed monitoring scheme can help focus and rationalise the monitoring process and cope with complexity.

In [58] a number of agents are deployed on two levels, lower level and higher level. Each agent is provided with a corresponding portion of the monitoring model; agents of the lower level are provided with functional models, and the higher-level agent has a Markov model. Agents are able to exchange messages to integrate their models and observations and deliver safety monitoring tasks. In a similar concept [59], [60] agents are provided with monitoring models (functional models) and deployed to monitor the deliverable functionality of systems. Agents are also able to collaborate with each other to integrate their models and observations and deliver consistent monitoring tasks.

Multi-agent systems have also been exploited in a different monitoring concept. In [61], for example, a number of agents are deployed to monitor the whole functionality of the monitored system and each agent is provided with a different reasoning algorithm and monitoring model, such as self-organisation maps, principal component analysis, neural network or non-parametric approaches. Agents are also able to collaborate with each other to decide consistently on whether the monitored conditions are normal or abnormal. In [62], a number of agents are also deployed to monitor the entire functionality of the monitored system, but every agent monitors the functionality of the system from different sensory data sources and the same monitoring model and reasoning algorithm, which couples Bayesian network and the method of majority voting.

Despite the monitoring success of multi-agent systems, two limitations have also been highlighted: (a) the typical lack of collaboration protocols that can support effective integration among the deployed agents [63]; (b) the logical omniscience problem in which some monitored conditions may fall beyond the knowledge of the agents [64], [65].

E. Motivation

Despite the above discussed efforts and wide variety of monitoring concepts, still there have been numerous instances of accidents that could have been averted with better monitors. The explosion and fire at the Texaco Milford Haven refinery in 1994, for instance, was attributed to late fault detection, poor alarm presentation and inadequate operator training for dealing with a stressful and sustained plant upset [66]. The Kegworth Air disaster occurred in 1989 because of (a) delay in alerting the pilot of the occurrence of the fault and its underlying causes; (b) ineffective alarm annunciation; (c) the lack of automated fault controlling [67]. Recently, monitoring problems contributed to a fatal accident to Air France flight AF447, in which an Airbus A330 crashed in the Atlantic on 1st of June 2009 and all 228 people on board were killed. The technical investigation partly attributed the accident to late fault detection, misleading alarm annunciation and the absence of clear guidance on emergency conditions, which fell beyond the skills and training of the pilot and co-pilot [68].

Motivated by addressing the monitoring problems of such accidents, this paper develops a distributed safety monitor by synthesising the benefits of two strands. The first is the exploitation of knowledge obtained from the application of a model-based safety assessment technique (i.e., HiP-HOPS). The second is the distributed reasoning of multi-agent systems. Specifically, the paper looks at:

- The development of an effective formalisation and distribution approach to bring the off-line safety assessment model of HiP-HOPS forward to serve in on-line safety as a distributed monitoring model.
- Addressing limitations that have faced the development of multi-agent monitors. Issues of interest are selecting a suitable reasoning paradigm for the multi-agent system and the development of an effective deployment approach, collaboration protocols and monitoring algorithms.

The ultimate aim is the achievement of a spectrum of monitoring merits ranging from the delivery of effective safety monitoring tasks to the development of a scalable and cost-effective monitor.

The rest of the paper is organised as follows: Section two briefly describes the nature of the monitored system, i.e., modern critical systems. Section three presents the position, role, and constituents of the monitor. Section four tests the monitor through the application to an aircraft fuel system. Section five contrasts the developed monitor and obtained results against earlier work. Section six, finally, draws a conclusion and proposes further work.

II. THE MONITORED SYSTEM

Large scale and dynamic behaviour are two common aspects of modern critical systems, i.e., phased-mission systems. While the former aspect calls into question the ability of the monitor to deliver consistent monitoring tasks over a huge number of components, the latter calls into question the ability of the monitor to distinguish between normal and abnormal conditions. A typical example of such systems is an aircraft, which delivers a trip mission upon the achievement of a number of phases; pre-flight, taxiing, take-off, climbing, cruising, approaching, and landing. Thorough knowledge about the architectural components and the dynamic behaviour is essential to achieve effective monitoring.

To model the mutual relations among the components, a hierarchical organisation is commonly used to arrange them in a number of levels. Across the levels, components appear as parents, children and siblings. Fig. 1 shows a classification of those levels. Levels are classified into three types: the lowest level (level*0*) is classified as the basic components (BC) level. The intermediate levels extending from level*1* to level*n-1* are classified as sub-system (Ss) levels. The top level (level*n*) is classified as the system (S) level.

To model the behaviour of the monitored system, it might be required to understand the way in which behavioural transitions are initiated. Typically, transitions are outcomes of, firstly, normal conditions in which the system engages its components in different structures, so it delivers different functionalities. Signals upon which that structure is altered are always initiated by the basic components. For

example, during the cruising of an aircraft, navigation sensors may convey signals to the navigator sub-system (NS), which in turn calculates those signals and notifies the flight control computer (FCC). Assuming that it is time for launching the approaching phase, FCC accordingly instructs the power plant system (PPS) to achieve the required thrust and the surface hydraulic controller (SHC) to achieve the required body motions. The case in which the system uses a certain structure to deliver certain functionality is called a *mode*.

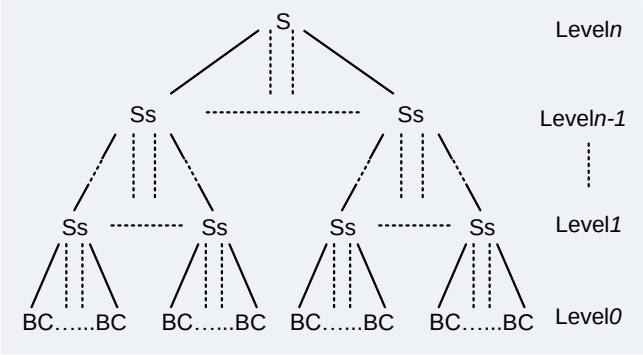

Figure 1. Hierarchical view of the monitored system.

Secondly, dynamic behaviour could be an outcome of the fault or fault tolerating of the basic components. Fault tolerance is typically implemented by two strategies: active fault-tolerant controlling (AFTC) and passive fault-tolerant controlling (PFTC). In the former strategy faults cannot be corrected totally but the consequent effects can be controlled as the system adapts to faults of its components, e.g., the fault of one engine of a two-engine aircraft can be compensated by the other engine. In the latter strategy the system has the ability to tolerate the fault for a while, e.g., faults that are caused by software error, ionisation radiation, electromagnetic interference, or hardware failure can be corrected within a short interval by restarting the relevant component or by isolating the faulty component and starting up a redundant one.

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 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:

- *Temporary Degraded or Failure State (TDFS)* in which there is one or more functional failure, but corrective measures can be taken to transit to another state;
- *Permanent Degraded State (PDS)* in which an uncontrollable fault occurs, but the safe part of the functionality can be delivered;
- *Failure State (FS)* in which the intended function is totally undelivered.

Events that are initiated by the basic components play a key role in making the behaviour of a system dynamic. To track the behaviour, such events should be continuously monitored. Thus, the best hierarchical level to monitor these events should be identified. Fig. 2 illustrates the relationships among the architectural levels and three factors based on which that level can be decided; early fault detection and diagnosis, computational cost and behavioural understanding. Achieving trade-off among these factors could help effectively in identifying the targeted level.

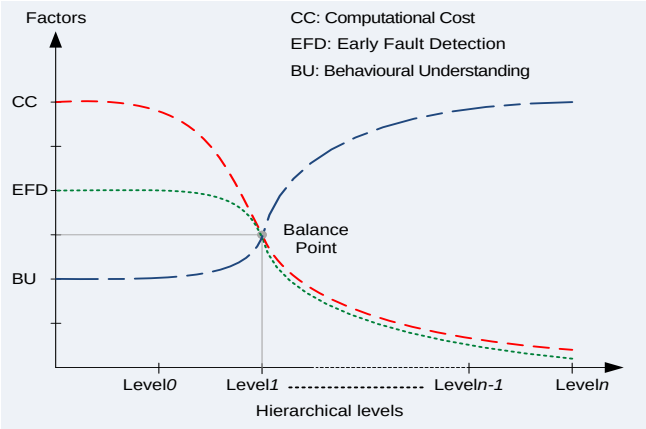

Figure 2. Three monitoring factors and architectural levels.

At level*1* the occurrence of events could be identified as either normal or abnormal, e.g., the decreasing of velocity and altitude seems normal when the flight control computer has already launched the approaching of the aircraft. Excluding knowledge about the modes and focusing only on the measurements provided by the relevant sensors would certainly result in misinterpreting system behaviour, i.e., decreasing velocity and altitude would appear as a malfunction and a misleading alarm would accordingly be released. Having that fact, level*1* would also be the best level – rather than any higher level – since at that level a malfunction is detected while in its early stages. Finally, due to the potentially huge number of the basic components, monitoring events at level*0* is computationally expensive or even unworkable, whereas level*1* offers the required rationality. Without loss of generality, it is assumed that primary detection of the symptoms of failure occurs at level*1*.

III. Distributed On-line Safety Monitor

The monitor takes a position between the system and the operators' interface. During normal conditions, the monitor provides simple feedback about those conditions. The monitor plays its role during abnormal conditions, which are triggered by and follow the occurrence of faults. It delivers three safety tasks; prompt fault detection and diagnosis, alarm annunciation and fault controlling.

Prompt fault detection and diagnosis refers to the timeliness of detecting faults while in their early stages and before they develop into real hazards, in parallel with diagnosing the underlying causes. This is supported by

selecting an appropriate hierarchical level at which efficient monitoring of the operational parameters can be achieved in addition to setting and monitoring those parameters against well-defined thresholds.

The task of effective alarm annunciation involves defining thresholds whose violation represents actual deviations of the monitored parameters. It also involves suppressing unimportant and false alarms whose release would overwhelm and confuse the operators. This is achieved by the following:

- Tracking the behaviour of the monitored system and distinguishing among the occurrence of normal, corrective and failure events.
- Releasing an alarm only on the occurrence of genuine failure events and not on other events, such as consequent, precursor or causal events.
- Developing techniques to filter out and validate the sensory measurements. Prioritising alarm presentation is also important to deliver effective alarm annunciation. This can be achieved by distinguishing the important alarms by using different colours, vibration or alerting sounds, and hiding the presentation of the less important alarms, e.g., optional access to the diagnostics list on the operators' interface.
- Annunciation of effective alarm information that could help the operators to direct the system effectively in the presence of faults and control abnormal conditions. Information is presented as (a) assessment of the operational conditions following the occurrence of the fault; (b) guidance on the corrective actions that should be taken manually by the operators; (c) timely prognosis of the future effects of the occurred fault. In order to avoid overwhelming the operators, prognoses would be presented in a timely manner and in the context of behavioural transitions of the monitored system.

The monitor can achieve both active and passive fault-tolerant controlling and also support manual fault controlling by assessment, guidance and prognoses to control abnormal conditions that may fall beyond the trained skills of the operators.

The monitor consists of two main elements. The first is a distributed monitoring model that is derived from an off-line HiP-HOPS safety assessment model, which consists of a behavioural model as a hierarchy of state-machines and fault propagation models as a number of fault trees (Fig. 3). To bring the assessment model forward to serve the on-line monitoring, the achievement of two processes is needed. The first is formalising events that trigger transitions in the behavioural model and symptoms that associate the error propagation paths of faults as monitoring expressions. Hence, the occurrence of events and symptoms can be verified computationally by instantiating and evaluating monitoring expressions based on real-time conditions. Verification of events supports tracking the behaviour of the monitored system and verification of symptoms supports tracking the error propagation path from the detected faults at level*1* towards the underlying causes at level*0*.

The second process is distributing the model into a number of models without violating the integrity and consistency of the encoded knowledge; for each sub-system there will be a monitoring model and a model for the entire system appears at level*n*.

The second element is a multi-agent system, which is a set of Belief-Desire-Intention (BDI) agents. Agents are deployed and provided with their portions of the monitoring models to reason locally at the sub-systems level and also provided with collaboration protocol to integrate globally at the system level and deliver the three safety tasks.

A. Monitoring Expressions

In its simple form, a monitoring expression appears as a constraint that consists of three main parts: (a) an observation, which is either a state of a child or the parent or 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.

The formalisation of events in the state-machine of level*1*'s sub-systems and the symptoms of the diagnostic model might require more complicated forms of constraint that incorporate (a) observations that should 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, observations 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. Consider, for example, an expression to monitor a structural leak of a tank of the aircraft fuel system, a case study presented in this paper (Fig. 13). Assuming that the leak is in the inner tank of the left-wing (LW) sub-system, the monitoring expression can be formalised as follows:

$$LL1(T-5) - LL1(T) > \int_{T-5}^{T} \big(FL1(t) + FL2(t)\big)\, dt + 0.06 \quad (1)$$

where

$\big(\mathrm{LL1(T-5) - LL1(T)}\big)$: is the reduction of fuel level in the inner tank over an interval extending from T-5 in the past to current time T.

$\int_{\mathrm{T-5}}^{\mathrm{T}} \big(\mathrm{FL1(t) + FL2(t)}\big)\,\mathrm{dt}$: is the total amount of fuel that has been (a) drawn from the inner tank by pump PL1 over an interval extending from T-5 in the past to current time T; (b) drawn or added by pump PL2 over the same interval. The interval is defined as

5 seconds as the shortest time to detect the structural leak.

0.06: is the maximum allowable discrepancy between the two above observations in normal conditions.

The calculation and evaluation of such an expression necessitate holding sensory measurements over time, i.e., historical measurements. Fig. 3 shows multi-measurement buffer along with its systematic updating.

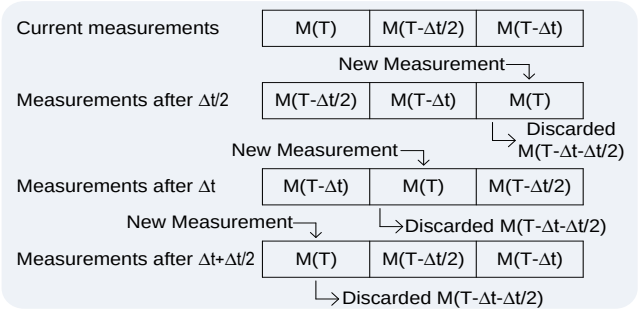

Figure 3. Systematic update of a multi-measurement buffer.

In Fig. 3, the updating process is applied continuously over time and after every elapsing of Δt/2. It can be seen how the updating maintains a systematic interval of Δt/2 among the measurements and replaces measurements that fall out of Δt. This structure could hold sensory measurements that suffice for the calculation and evaluation of expressions like expression (1).

Sensors may deliver temporary spurious measurements because of (a) additive white Gaussian noise, such as electromagnetic interference, ionisation radiation and thermal noise; (b) mode changes, which would typically be followed by an interval of unsteady behaviour. The best way to filter out such measurements is perhaps by forming a *timed expression*. Such an expression is evaluated successively over a filtering interval and based on a number of measurements. The final evaluation result is obtained by making accumulative conjunctions among the successive evaluations. If the final result is true, that means the delivered measurements remain the same over the filtering interval. Hence, the occurrence of that event should be verified. The filtering interval of every expression is defined by examining, firstly, the conditions that may result in spurious measurements; secondly, the time intervals at which the involved sensors are demanded – by the monitor – to deliver sensory measurements. For example, in the fuel system case study, monitoring the fuel flow to the port engine requires formalising a timed expression as follows:

$$| FF1 | < 0.03 \text{ for } 4 \text{ sec} \qquad (2)$$

In practice, sensors may fail permanently and deliver spurious measurements that persist over or even go beyond the filtering interval. In addition to misleading the monitor; such measurements could also affect the controller of the monitored system and result in hazardous failures. Sensory measurements should, therefore, be validated and faulty sensors should be detected, diagnosed and controlled.

To achieve that, a technique of formalising special monitoring expressions is developed. The technique is based mainly on the sub-grouping approach of [39] and the Sensory Failure Diagnosis Tree (SFDT) approach of [41], [42]. Drawing from the sub-grouping approach, sensors that can detect each other's faults are identified and based on the idea of SFDT the proper expression is formed.

For example, a sensory failure of the flow meter FC1 of the fuel system can be detected and diagnosed by the following expression:

$$\begin{gathered}(FC1 > R/7+0.03 \text{ for } 6 \text{ sec AND } FC1 > FC2+0.03 \text{ for } 6 \text{ sec}) \\ \text{OR} \\ (FC1 < R/7-0.03 \text{ for } 6 \text{ sec AND } FC1 < FC2-0.03 \text{ for } 6 \text{ sec})\end{gathered} \qquad (3)$$

To control sensory failures, the technique suggests isolating the faulty sensor by ignoring its measurements and measuring the same trend from an alternative sensor or from a number of sensors whose measurements can be calculated to correspond as an alternative to the isolated measurement. In the case of isolating the flow meter FC1, the alternative sensor can be the other flow meter FC2.

Extended-Backus Naur Form (E-BNF) notation is exploited to define a general grammar to formalise different monitoring expressions according to the nature of the monitored conditions. In that grammar a set of primitives has been introduced to allow expressions to reference historical values and calculate different monitoring trends. Primitives include a historical operator *S_ID(Δt)*, which returns historical sensory measurement collected in the past at current time T minus Δt, i.e., T-Δt. Primitives also define more complicated operators, such as the differentiation D(expression,Δt), integration I(expression,Δt), variation V(expression,Δt) and timed expression *T(expression, Δt)*.

By these primitives, monitoring expressions can be presented in standard computation forms. Consider, for example, expression (1); it can be presented as:

$$V(LL1\_L, 5) > I(FL1\_F + FL2\_F, 5) + 0.06$$

Expression (2) can be presented as:

$$T(|FF1\_F| < 0.03, 4 \text{ sec})$$

Expression (3) can be presented as:

$$\begin{gathered}(T(FC1 > R/7 + 0.03, 6) \text{ AND } T(FC1 > FC2 + 0.03, 6)) \\ \text{OR} \\ (T(FC1 < R/7 - 0.03, 6) \text{ AND } T(FC1 < FC2 - 0.03, 6))\end{gathered}$$

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

Expression OR T(Expression, Δt) (4)

Expression AND T(Expression, Δt) (5)

Evaluating expressions (4) or (5) may require waiting time equal to Δt, i.e., until evaluating *T(Expression, Δt)*. However, the *Expression* part of either (4) or (5) can be evaluated instantly. Hence, knowing that the disjunction of *True* with *Unknown* is *True* and the conjunction of *False* with *Unknown* is *False*, both (4) and (5) can be evaluated instantly. Therefore, in cases in which *Expression* of (4) is evaluated with *True* and *Expression* of (5) is evaluated with *False*, both (4) and (5) could be evaluated instantly with values *True* and *False*, respectively.

B. Distributed Monitoring Model

In the light of the intended three monitoring tasks, agents should be able to track the behaviour of the monitored components over different states, i.e., error-free states (EFSs) and error states (ESs). This is important to distinguish between normal and abnormal conditions and provide the operators with information that confirms whether the conditions are normal or not. In abnormal conditions, agents should provide alarm, assessment, guidance, diagnostics and prognoses. Agents should also have a reference to apply corresponding corrective measures for every fault.

Fig. 4 shows an illustrative view of the HiP-HOPs model. The model is a composite of a behavioural model and fault trees. The behavioural model is a hierarchy of state-machines that captures the behaviour of the system and its sub-systems. Each fault tree records the possible symptoms, propagation paths and underlying causes of a failure event.

Relationships among the components are implemented in the state-machine hierarchy as parent and children components. In the state-machine of the sub-systems of level*1*, events are originated by (a) the BCs of level*0*, which might be failure, corrective or normal events; (b) parent states, such as the error-free state of a new mode of the parent or error states.

In the state-machine of a sub-system of the levels extending from level*2* to level*n-1* events appear as error-free and error states of the parent and children. Finally, in the state-machine of the system (level*n*) events appear as error-free states and error states of the children.

Similarly, error states of the children could also trigger transitions in the state-machines of the parents and vice versa. For example, the failure state of an engine of a two-engine aircraft triggers a transition to the permanent degraded state in the state-machine of the power plant system. The degraded state, in turn, triggers a transition to a new error-free state of the operative engine in which the lost functionality of the faulty engine is compensated.

To distinguish between normal, fault and corrective events, the principle is applied that an alarm should be released on the occurrence of failure events only. Thus, corresponding alarm clauses should be associated with the failure events of level*1*, the level at which events are monitored. Computationally, if an occurred event is associated with a "none" then it is either a normal or a corrective event; on the contrary, any other clause means that it is a failure event and the associated clause should be quoted and released as an alarm. While assessment is a description of the given conditions and guidance is about the best actions to be applied in those conditions by the operators, their clauses should thus be enclosed by the states.

To find the appropriate place for incorporating corrective measures, further consideration of the nature of those measures is needed. Typically, there are two different types of corrective measures. The first should be taken after diagnosing the underlying causes. This is appropriate when the verified failure event can be caused by multiple faults of the basic components. Measures to correct any of those causes vary from one cause to another. Measures should, therefore, be incorporated in the diagnostic model (e.g., fault tree), precisely in association with the potential causes.

The second type of corrective measures should be taken at level*1*, when level*1*'s sub-systems supported by higher level components (sub-systems or system) apply measures to respond to deviations that have a clear cause. At level*1*, corrective measures are mostly applied with directions coming from higher levels. For example, in modern aircraft switching to the backup computer sub-system at level*1* is instructed directly by the flight control system (FCS) at level*2*, whenever the primary computer sub-system at level*1* fails. The instructions are implemented at level*1* by switching the primary computer off and backup computer on. Measures should also be taken at level*1*, when level*1*'s sub-systems supported by level*0*'s basic components apply measures to respond to deviations that have a clear cause. Expression (3), for example, relates a failure event of the condensing sub-system directly to a fault of the flow meter (FC1). In this case, measures are taken to isolate the FC1 and depend alternatively on the measurements obtained from another flow meter FC2.

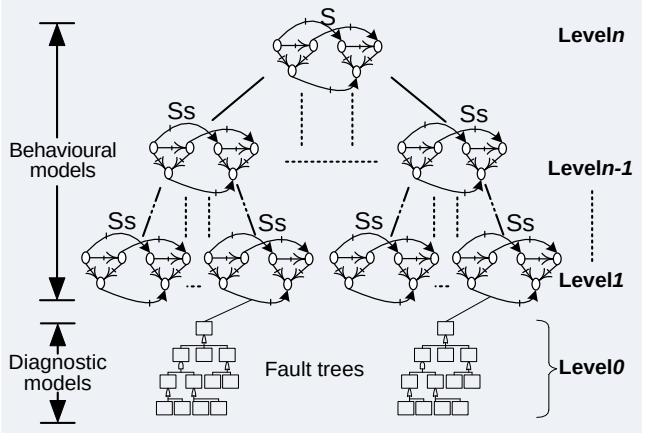

Figure 4. An illustrative view of the HiP-HOPS safety assessment model.

To present the graphical state-machines in an executable format, state-transition tables represent a classic choice. A state-transition table is usually defined as an alternative and formal form to present graphical state-machines and it typically offers the required capacity and flexibility to incorporate knowledge about the operational conditions [69].

Fig. 5 shows an excerpt of the state-machine of level*n* of the fuel system case study. Table I shows the state-transition table of the fuel system. It can be seen how the trigger events of the state-machine (Fig. 5) are formalised as monitoring expressions in Table I. For example, event CM_FS of EF, which is the failure state of the engine feed (EF) sub-system during the consumption model (CM) of the fuel system, is formalised as EF_CM_FS == true.

Fig. 6 shows an excerpt of the state-machine of the EF sub-system. It can be seen how the states of the fuel system and its sub-system appear mutually as trigger events in each other's state-machines. Table II shows the formal behavioural model of the engine feed sub-system.

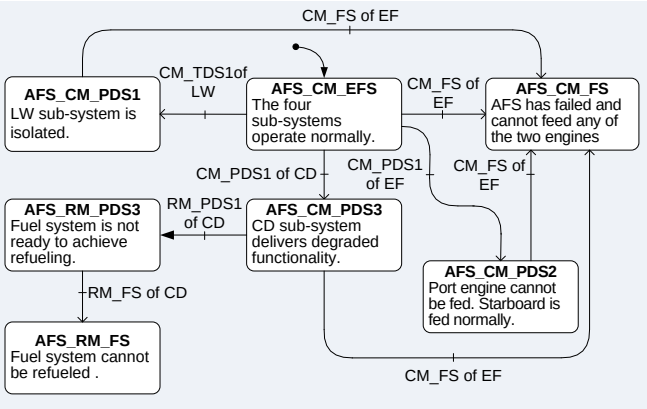

Figure 5. An excerpt of the state-machine of the aircraft fuel system.

TABLE I. STATE-TRANSITION TABLE OF THE AIRCRAFT FUEL SYSTEM.

CURRENT STATE	CONDITIONS	EVENT	NEW STATE
AFS_CM_EFS	**Assessment**: the four sub-systems operate normally. **Guidance**: none.	EF_CM_FS == true	AFS_CM_FS
		LW_CM_TDS1 == true	AFS_CM_PDS1
		EF_CM_PDS1 == true	AFS_CM_PDS2
		CD_CM_PDS1 == true	AFS_CM_PDS3
AFS_CM_PDS1	**Assessment**: LW sub-system is isolated. **Guidance**: none.	EF_CM_FS == true	AFS_CM_FS
AFS_CM_PDS2	**Assessment**: port engine cannot be fed, whereas starboard engine is feeding normally **Guidance**: none.	EF_CM_FS == true	AFS_CM_FS
AFS_CM_PDS3	**Assessment**: CD sub-system delivers degraded functionality. **Guidance**: none.	CD_RM_PDS1 == true	AFS_RM_PDS3
AFS_RM_PDS3	**Assessment**: fuel system is not ready to achieve refuelling. **Guidance**: none.	CD_RM_FS == true	AFS_RM_FS
AFS_CM_FS	**Assessment**: AFS has failed and cannot feed any of the two engines. **Guidance**: none	none	none

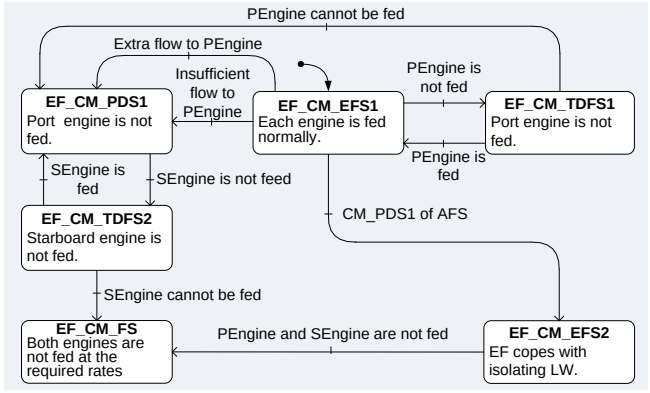

Figure 6. An excerpt of the state-machine of the engine feed sub-system.

Table II differs from Table I, as it incorporates three extra columns: alarm, controlling and diagnosis. The alarm column holds clauses that can be announced to alert the operators. The controlling column may hold corrective measures or "after diagnosis" based on the nature of those measures. The diagnosis column holds the status of whether the occurred failure event has a clear cause or a diagnostic process is needed.

TABLE II. STATE-TRANSITION TABLE OF THE ENGINE FEED SUB-SYSTEM.

CURRENT STATE	CONDITIONS	EVENT	ALARM	CONTROLLING	DIAGNOSIS	NEW STATE
EF_CM_EFS1	**Assessment:** each engines is fed normally. **Guidance:** none	**T**(|FF1_F| < 0.03, 4);	port engine is not fed	after_ diagnosis	needed	EF_CM_TDFS1
		T(FF1_F > R + 0.03, 6);	port engine fed with extra rate	- PF1 = 0; - VF1 = 0;	not_ needed	EF_CM_PDS1
		T(FF1_F < R – 0.03, 6);	port engine is fed with insufficient rate.	- PF1 = 0; - VF1 = 0;	not_ needed	EF_CM_PDS1
		AFS_CM_PDS1 == true;	none	none	not_ needed	EF_CM_EFS2
EF_CM_TDFS1	**Assessment:** port engine is not fed and recovery is in progress. **Guidance:** watch for further feedback.	**T**(|FF1_F| < 0.03, 4);	feeding port engine cannot be recovered.	- PF1 = 0; - VF1 = 0; - VF2 = 0;	not_ needed	EF_CM_PDS1
		T(|FF1_F - R| < 0.03, 4);	none	none	not_ needed	EF_CM_EFS1
EF_CM_PDS1	**Assessment:** port engine is not fed. **Guidance:** none.	**T**(|FF2_F| < 0.03, 4);	starboard engine is not fed	after_ diagnosis	needed	EF_CM_TDFS2
EF_CM_EFS2	**Assessment:** EF sub-system copes with isolating LW sub-system. **Guidance:** none.	**T**(|FF1_F| < 0.03 **AND** |FF2_F| < 0.03 , 4);	Both engines are not fed	impossible	needed	EF_CM_FS
EF_CM_TDFS2	**Assessment:** starboard engine is not fed and recovery is in progress. **Guidance:** watch for further feedback.	**T**(|FF2_F| < 0.03, 4);	feeding starboard engine cannot be recovered.	impssible	not_ needed	EF_CM_FS
		T(|FF2_F - R| < 0.03, 4);	none	none	not_ needed	EF_CM_PDS1
EF_CM_FS	**Assessment:** both engines cannot be fed. **Guidance:** none.	none	none	none	not_ needed	none

During the monitoring time, agents cyclically monitor events whose occurrence triggers transitions from the current state; every cycle is called a monitoring cycle. As such, the computational load of the agents would be less and prompt responses to the occurrence of the events would be established.

A diagnostic process is needed when a failure event and its underlying cause are in a one-to-many relationship. Therefore, a diagnostic model that could relate such events to their underlying cause is needed. As shown in Fig. 4, the HiP-HOPS model incorporates fault trees that can relate functional failures to their underlying causes. More specifically, for every functional failure, which may have multiple causes, there is a fault tree.

On the contrary, when the failure and its cause are in a one-to-one relationship, the name of the cause is stated in the state-transition table of level*1*'s.

Functional failures are related to their fault trees as every failure appears enclosed by the top node of the relevant fault tree. For example, the underlying cause of the failure event "PEngine is not fed" (shown in Fig. 6 and Table II) can be diagnosed by traversing the relevant fault tree, which is shown in Fig. 7. Fig. 8 shows the formal form of the diagnostic model that can be derived from the fault tree of Fig. 7.

Agents initiate the monitoring process by traversing, interpreting and uploading the state-transition tables and diagnostic models into interrelated data structures. Structure type and arrays are declared for this purpose. Arrays support direct addressing of the structures that hold the knowledge, so fast access during the monitoring time is established.

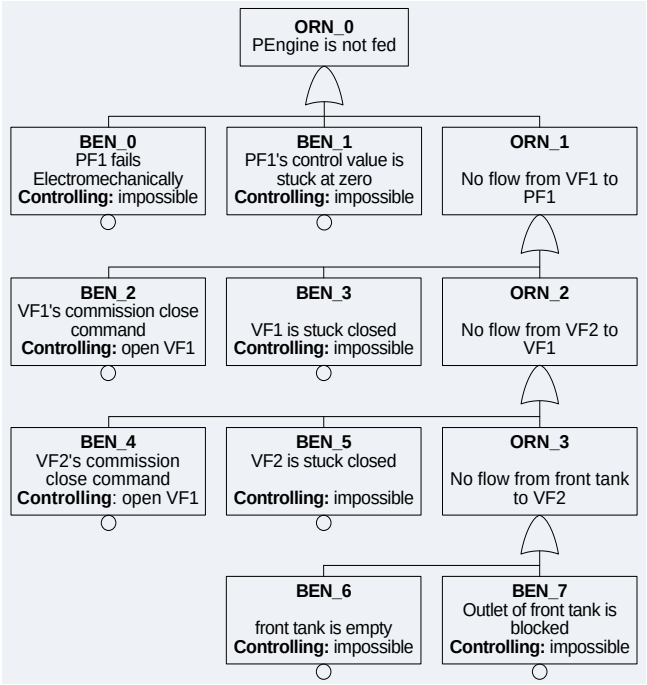

Figure 7. Fault tree of event "PEngine is not fed".

NodeName: ORN_0.
Symptom: T(|FF1**|** < 0.03, 4 sec).
ChildName: BEN_0.

NodeName: BEN_0.
Symptom: |PF1**|** <= 20.
Fault: PF1 fails electromechanically.
Controlling: none.
Sibling: BEN_1.

NodeName: BEN_1.
Symptom: |PF1**| <=** 20.
Fault: PF1's control value is stuck at zero.
Controlling: none.
Sibling: ORN_1.

NodeName: ORN_1.
Symptom: none.
Child: BEN_2.
Sibling: none.

NodeName: BEN_2.
Symptom: VF1 **==** 0.
Fault: VF1's commission close command.
Controlling: VF1_C = 1.
Sibling: BEN_3.

NodeName: BEN_3.
Symptom: VF1 **==** 0.
Fault: VF1 is stuck closed.
Controlling: none.
Sibling: ORN_2.

NodeName: ORN_2.
Symptom: none.
Child: BEN_4.
Sibling: none.

NodeName: BEN_4.
Symptom:VF2 **==** 0.
Fault: VF2's commission close command.
Controlling: VF2 = 1.
Sibling: BEN_5.

NodeName: BEN_5.
Symptom: VF2 == 0.
Fault: VF2 is stuck closed.
Controlling: none.
Sibling: ORN_3.

NodeName: ORN_3.
Symptom: none.
Child: BEN_6.
Sibling: none.

NodeName: BEN_6.
Symptom:(VF2 ==1) **AND** (VF1==1) **AND** (PF1 > 20).
Fault: front tank outlet is blocked.
Controlling: none.
Sibling: BEN_7.

NodeName: BEN_7.
Symptom: LF1 **<** 0.1.
Fault: front tank is empty.
Controlling: none.
Sibling: none.

Figure 8. Formal diagnostic model of the fault tree of Fig. 7.

C. Multi-agent System

In addition to the common ability of intelligent agents to achieve integrated reasoning among distributed processes [70], two more reasons underpin the particular adoption of BDI agents as monitoring agents. Firstly, as the reasoning

model of these agents is based on human reasoning, effective automation of the crucial responsibilities of system operators can be facilitated. Secondly, the informative communication as well as the semi-independent reasoning of the BDI agents can support effective collaboration and integration of two different deployment approaches. The first is spatial deployment in which agents are installed on a number of distributed computational machines. Such deployment is needed when the sub-systems of the monitored system are distributed over a geographical area, e.g., a chemical plant. The second approach is semantic deployment in which monitoring agents are installed on one computational machine. Such deployment is appropriate when the sub-systems of the monitored system, although distributed, are close to each other, e.g., an aircraft system.

Fig. 9 shows a general illustration of the monitoring agent. By perceiving the operational conditions and exchanging messages with each other, each agent obtains the up-to-date belief, deliberates among its desires to commit to an intention and achieves a means-ends process to select a plan, which is a course of actions. The selected plan is implemented, as actions towards achieving the monitoring tasks locally and as messages sent to other agents towards achieving global integration. Upon having a new belief, an agent achieves a reasoning cycle; deliberation and means-ends process.

Agents are deployed over the sub-systems and the system, and appear as a number of sub-system monitoring agents (Ss_MAGs) and a system monitoring agent S_MAG, as shown in Fig. 10. Each Ss_MAG of level*1* updates its belief base by perceiving (a) its portion of the monitoring model, which consists of a state-machine and a set of fault trees; (b) sensory measurements that are taken to instantiate and evaluate monitoring expressions; (c) messages that are received from the parent to inform the Ss_MAG about the new states and messages from siblings, in which they either ask for or tell the given Ss_MAG about global measurements; agents might need to share measurements globally. The main desires of an Ss_MAG of level*1* are to monitor the local conditions of the assigned sub-system and to collaborate globally with its parent and siblings. On the achievement of the local desire, the intentions are to track the behaviour of the sub-system and to provide the operators with alarms, assessment, guidance, diagnostics, prognoses and control faults. On the achievement of the global desire, the intentions are to exchange messages to inform the parent about the new states and to tell or ask the siblings about global measurements.

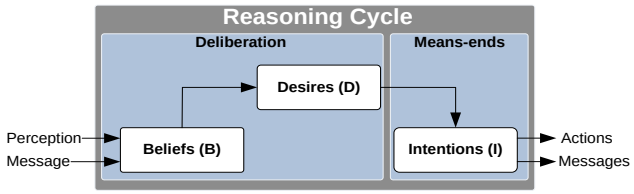

Figure 9. Reasoning cycle of the BDI agent.

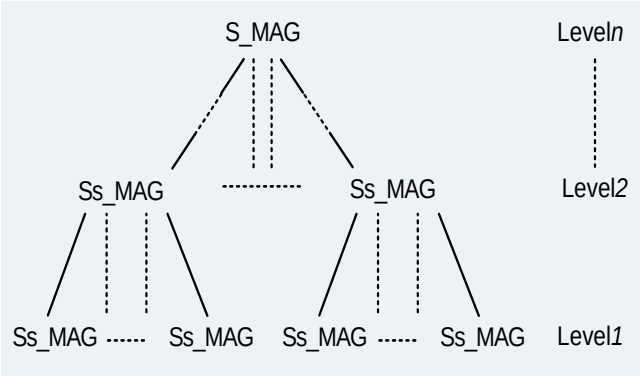

Figure 10. The hierarchical deployment of the monitoring agents.

Each Ss_MAG of the intermediate levels (levels extending from level*2* to level*n-1*) updates its belief by (a) perceiving its own portion of the monitoring, which consists of a state-machine of the assigned sub-system; (b) messages received from the parent and the children to inform it about their new states. The main desires of each of these Ss_MAGs are to monitor the local conditions of the assigned sub-system and to collaborate globally with its parent and child agents. On the local desire, the intentions are to track the behaviour of the sub-system and to provide the operators with assessment, guidance and prognoses of their levels. On the global desire, the intention is to exchange messages with the parent and child agents to inform each other about their new states.

The perceptions, desires and intentions of the S_MAG are similar to those of the Ss_MAGs of the intermediate levels. The only difference is that S_MAG has no parent to exchange messages with.

According to the Prometheus approach and notation for developing multi-agent systems [71], Fig. 11 shows the collaboration protocols among agents to track the behaviour of the monitored system. Fig. 12 shows the collaboration protocol among the Ss_MAGs of level*1* in which they share their sensory measurements.

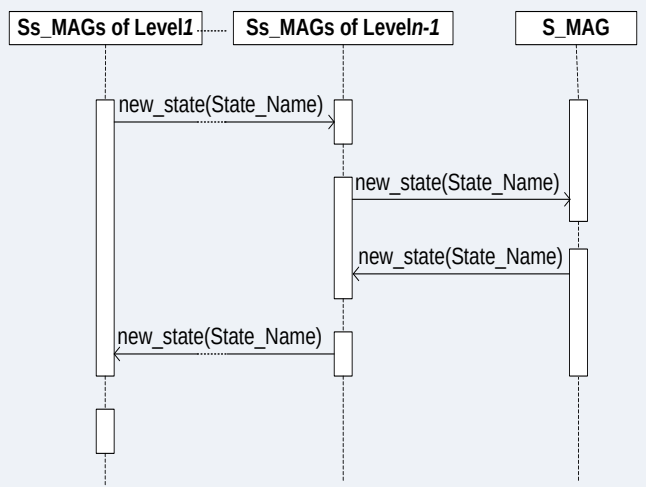

Figure 11. Collaboration protocol among agents across hierarchical levels.

According to the collaboration protocol of Fig. 11, every new state that results from a state transition at level*1* is communicated by the agent to its parent agent, which in turn communicates its own new state higher up to its parent, and so on successively to the S_MAG at the top level (level*n*). The S_MAG, in turn, communicates its own new state to the children at level*n-1*. Every child agent communicates its own new state similarly to its children. This scenario is repeated successively between every agent and its children until the agents of level*1* are reached.

According to Fig. 12, Ss_MAGs of level*1* share their sensory measurements (global measurements). Any Ss_MAG may ask for a measurement by sending an ask message to the intended Ss_MAG. The receiving Ss_MAG (asked Ss_MAG) should answer accordingly by sending a tell message.

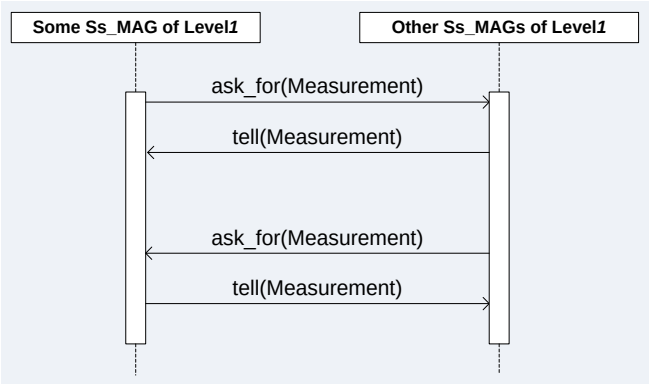

Figure 12. The collaboration protocol among MAGs of level*1*.

Every agent deployed at level*1* is provided with a portion of the monitoring model, which incorporates a state-transition table and a number of diagnostic models. An agent at this level is also provided with a monitoring algorithm to track the behaviour of the monitored sub-system and a diagnostic algorithm to relate the verified failure events to their underlying causes. Every agent deployed at levels extending from level*2* to level*n* is provided with a monitoring model, which is a state-transition of the assigned sub-system or system.

IV. Case Study: Aircraft Fuel System

Fig. 13 shows a graphical illustration and components of the fuel system. The system functions to maintain safe storage and even distribution of fuel in two modes. The first is the consumption mode in which the system provides fuel to the port and starboard engines of a two-engine aircraft. The second is the refuel mode.

During the consumption mode and to maintain the central gravity and stability, a control unit applies a feedback-control algorithm to ensure even fuel consumption across the tanks; flow rates and directions are as shown in Fig. 13. 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 system 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. An active fault-tolerant control strategy is implemented; specifically, in the presence of faults there are alternative flow paths to connect the two engines to the available fuel resources.

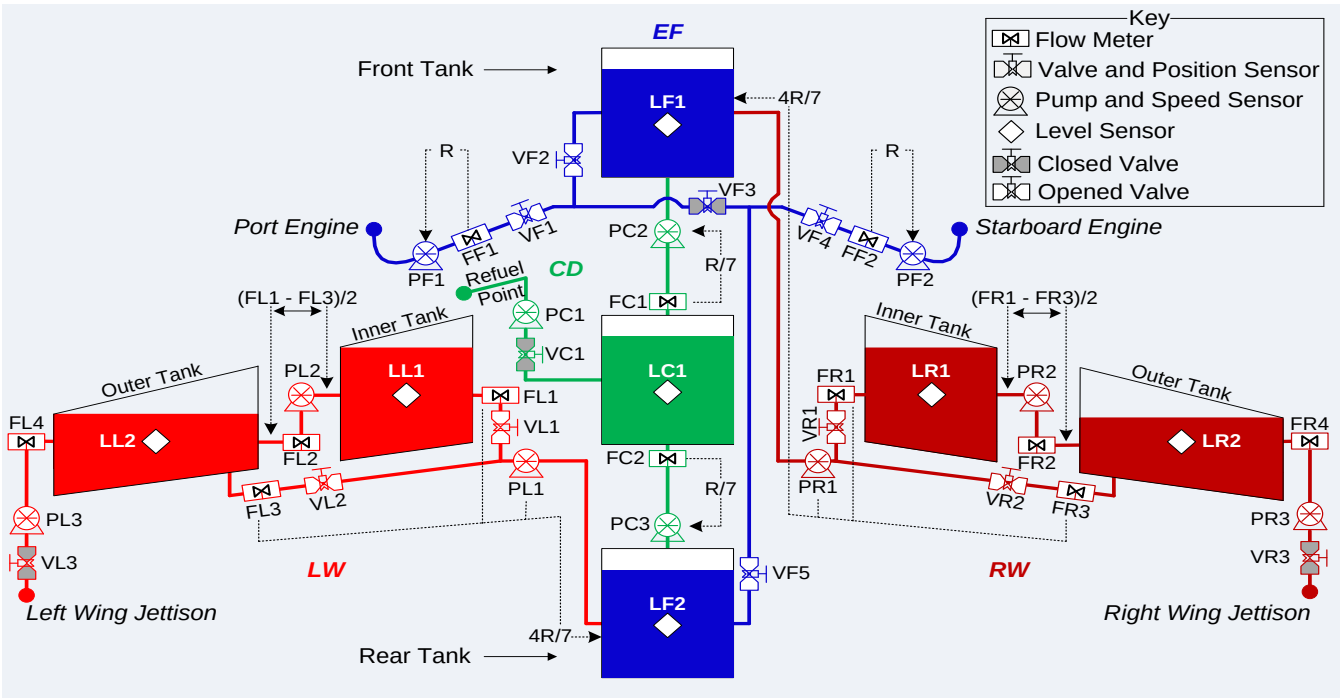

Figure 13. Graphical illustration of an aircraft fuel system.

As shown in Fig. 14, five monitoring agents are deployed to monitor the fuel system. Four of those agents are to monitor the four sub-systems; they appear as EF_MAG, CD_MAG, LW_MAG, and RW_MAG. The fifth agent is AFS_MAG, which monitors the entire fuel system. The monitor is implemented using Jason interpreter, which is an extended version of AgentSpeak programming language [72].

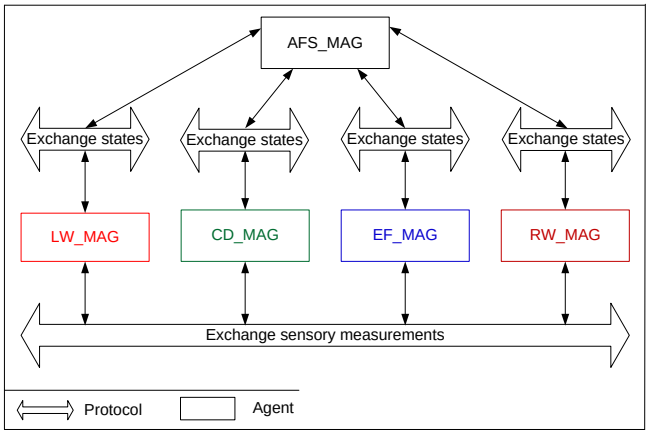

Figure 14. Deployment of agents to monitor the aircraft fuel system

To achieve the monitoring experiment, excerpts of the state-transition tables of the CD, LW and RW sub-systems are as shown by Table III, Table IV and Table V.

TABLE III. STATE-TRANSITION TABLE OF THE CD SUB-SYSTEM.

CURRENT STATE	CONDITIONS	EVENT	ALARM	CONTROLLING	DIAGNOSIS	NEW STATE
CD_CM_EFS1	**Assessment:** CD sub-system operates normally. **Guidance:** none	(**T**(FC1 > R/7 + 0.03, 6) **AND** **T**(FC1 > FC2 + 0.03, 6)) **OR** (**T**(FC1 < R/7 - 0.03, 6) **AND** **T**(FC1 < FC2 - 0.03, 6));	CD sub-system has a sensory failure.	- FC1= FC2;	Sensor FC1 has failed.	CD_CM_PDS1
		AFS_CM_PDS1 == true;	none	- FC1 = -R/5; - FC2 = 3R/5;	not_ needed	CD_CM_EFS2
		AFS_CM_PDS2 == true;	none	- FC1 = -3R/7; - FC2 = 4R/7;	not_ needed	CD_CM_EFS3
CD_CM_PDS1	**Assessment:** CD sub-system operates degradedly. **Guidance:** none.	VC1 == 1 **AND** VF1 == 0 **AND** VF4 == 0;	none	not_needed	not_ needed	CD_RM_PDS1
CD_RM_PDS1	**Assessment:** CD sub-system operates degradedly. **Guidance:** flow meter FC1 must be replaced.	**T**(|FC2| < 0.03, 4);	rear tank is not refueling.	- PC1 = 0; - VF1 = 0;	needed	CD_RM_FS
CD_CM_EFS3	**Assessment:** CD sub-system copes with a degraded state of the AFS **Guidance:** none.	**T**(|FC1| < 0.03 **OR** |FC2| < 0.03, 4);	abnormal flow from the central tank.	- PC2_S = 0; - PC3_S = 0;	needed	CD_CM_FS

TABLE IV. STATE-TRANSITION TABLE OF THE LW SUB-SYSTEM.

CURRENT STATE	CONDITIONS	EVENT	ALARM	CONTROLLING	DIAGNOSIS	NEW STATE
LW_CM_EFS1	**Assessment:** LW sub-system operates normally. **Guidance:** none	AFS_CM_PDS2 == true;	none	- FL3 = R/7; - FL1 = R/7;	not_ needed	LW_CM_EFS2
		V(LL1, 5) > I(FL1 + FL2, 5) + 0.06;	inner tank of LW sub-system is leaky	- PL1 = 0; - VL1 = 0; - VL2 = 0; - VL3 = 1; - FL2= - 0.285; - FL4 = 0.571;	leak in the inner tank of LW.	LW_CM_TDS1
LW_CM_EFS2	**Assessment:** LW sub-system copes with a degraded state of the AFS. **Guidance:** none	V(LL1_L, 5) > I(FL1 + FL2, 5) + 0.06;	inner tank of LW sub-system is leaky	- PL1 = 0; - VL1 = 0; - VL2 = 0; - VL3 = 1; - FL2 = -0.285; - FL4= 0.571;	leak in the inner tank of LW.	LW_CM_TDS1

TABLE V. STATE-TRANSITION TABLE OF THE RW SUB-SYSTEM.

CURRENT STATE	CONDITIONS	EVENT	ALARM	CONTROLLING	DIAGNOSIS	NEW STATE
RW_CM_EFS1	**Assessment:** RW sub-system operates normally. **Guidance:** none	AFS_CM_PDS1 == true;	none	- FR1 = 2R/5; - FR3 = 2R/5;	not_ needed	RW_CM_EFS2
		AFS_CM_PDS2 == true;	none	- FR1 = R/7; - FR3 = R/7;	not_ needed	RW_CM_EFS3
RW_CM_EFS2	**Assessment:** RW sub-system copes with isolating LW sub-system. **Guidance:** none	V(LR1, 5) > I(FR1 + FR2, 5) + 0.06;	inner tank of RW sub-system is leaky	- PR1 = 0; - VR1 = 0; - VR2 = 0; - VR3 = 1; - FR2 =- 0.285; - FR4 = 0.571;	not_ needed	RW_CM_TDS1
RW_CM_EFS3	**Assessment:** RW sub-system copes with a degraded state of the AFS. **Guidance:** none	V(LR1, 5) > I(FR1 + FR2, 5) + 0.06;	inner tank of RW sub-system is leaky	- PR1 = 0; - VR1 = 0; - VR2 = 0; - VR3 = 1; - FR2 =- 0.285; - FR4 = 0.571;	not_ needed	RW_CM_TDS1

Among the faults that have been injected to test the monitor is that the port engine is not fed and a fault of flow meter sensor FC1 of the central deposit (CD) sub-system.

A. First Simulated Failure scenario: "PEngine is not fed"

Once the monitoring agent EF_MAG evaluates expression (2) with true, it perceives the state-transition table (Table II) and achieves the following procedure:

- From the relevant ALARM attribute, agent FE_MAG quotes the statement "port engine is not fed" and alarms the pilot.
- From the relevant CONTROLLING attribute, agent FE_MAG checks the possibility of controlling that event. As the controlling depends on the underlying cause, that attribute accordingly tells the EF_MAG to achieve a diagnostic process by traversing the relevant fault tree ("after_diagnosis").
- From the relevant DIAGNOSIS attribute, agent FE_MAG verifies the need for a diagnostic process and updates the symptoms of the diagnostic model.
- From the relevant NEW STATE attribute, agent FE_MAG transits to the new state, which is the temporary degraded or failure state of the consumption mode EF_CM_TDFS1. From this state the pilot is provided with the assessment, "port engine is not fed and recovery is in progress" and guidance, "watch for further feedback".

- Agent EF_MAG also communicates the current state to the parent agent (AFS_MAG). The state does not trigger a state transition in the state-transition table of the AFS_MAG. At this point, the pilots are alarmed and informed on the operational condition as shown in Fig. 15.

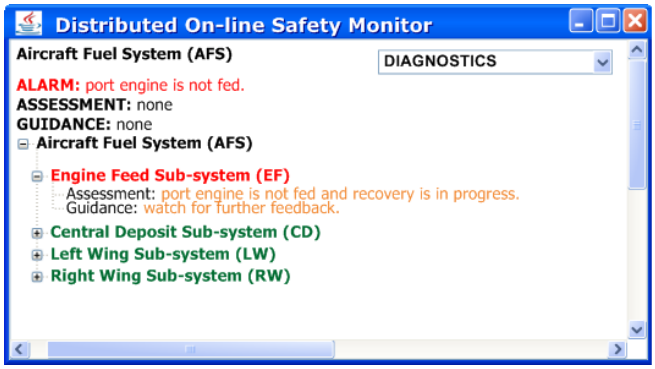

Figure 15. Operator interface after detecting and verifying the failure.

Since a diagnostic process is needed, agent EF_MAG retrieves the position of the top node of the relevant fault tree and launches a diagnostic process before launching a monitoring cycle for the new state EF_CM_TDFS1. By traversing the relevant diagnostic model (Fig. 8) the underlying cause is diagnosed and the required corrective measures are taken. Assuming that the cause is "VF1's commission close command", controlling is not possible and thus agent EF_MAG perceives Table II and achieves the following procedure:

- Launches a monitoring cycle to the active events of state EF_CM_TDFS1.
- During this cycle the occurrence of event T(|FF1_F| < 0.03, 4) is verified consequently.
- From the relevant ALARM attribute, agent EF_MAG quotes and announces an alarm of "feeding port engine cannot be recovered".
- From the relevant CONTROLLING attribute, agent EF_MAG takes the following actions: switching pump PF1 off and closing valves VF1 and VF2.
- As the diagnostic process appears not to be needed with this event, agent EF_MAG moves accordingly to the NEW STATE attribute, identifies and transits to a new state, which is EF_CM_PDS1. From this state the pilot is provided with assessment as "port engine is not fed" and guidance, "none".
- Agent EF_MAG also communicates EF_CM_PDS1 to the parent agent (AFS_MAG).

Feeding only one engine (starboard engine) requires changing the operational structure of the entire fuel system to maintain an even level across the seven tanks. Accordingly, the above procedure is not enough to control the fault; controlling these conditions requires global collaboration among the remaining three sub-systems: LW, RW, and CD.

Once agent AFS_MAG receives a message conveying state EF_CM_PDS1, it perceives the state-transition table (Table I) and achieves the following procedure:

- While the current state is AFS_CM_EFS, the received state results in verifying the occurrence of EF_CM_PDS1 == true.
- From the relevant NEW STATE attribute, agent AFS_MAG transits to the new state, which is the permanent degraded state AFS_CM_PDS2. From this state the pilot is provided with assessment as "port engine cannot be fed, whereas starboard engine is feeding normally" and guidance, "none".
- Agent AFS_MAG also communicates state AFS_CM_PDS2 to the child agents: CD_MAG, LW_MAG and LW_MAG.

Upon receiving messages conveying that state, each child agent achieves a certain fault controlling procedure to draw the corresponding flow rates and also transits to a new state. State transition and controlling procedures are as follows:

Agent CD_MAG perceives the state-transition table (Table III) and achieves the following procedure:

- While the current state is CD_CM_EFS1, the received state results in verifying the occurrence of AFC_CM_PDS2 == true.
- As the relevant ALARM attribute holds "none", no alarm is thus announced.
- From the relevant CONTROLLING attribute, agent CD_MAG applies the following flow rates: FC1 = -3R/7 and FC2 = 4R/7.
- As the relevant DIAGNOSIS attribute holds "not_needed", a diagnostic process is not launched.
- From the NEW STATE attribute, agent CD_MAG transits to the new state, which is another error-free state CD_CM_EFS3. From this state the pilot is provided with assessment, "CD sub-system copes with a degraded state of the AFS" and guidance, "none".

LW_MAG perceives the state-transition table (Table IV) and achieves the following procedure:

- While the current state is LW_CM_EFS1, the received state results in verifying the occurrence of AFC_CM_PDS2 == true.
- As the relevant ALARM attribute holds "none", no alarm is thus announced.
- From the relevant CONTROLLING attribute, agent LW_MAG applies the following flow rates; FL3 = R/7 and FL1 = R/7.
- As the relevant DIAGNOSIS attribute holds "not_needed", a diagnostic process is not launched.
- From the relevant NEW STATE attribute, agent LW_MAG transits to the new state, which is another error-free state LW_CM_EFS2. From this state the pilot is provided with assessment, "LW sub-system copes with a degraded state of the AFS" and guidance "none".

RW_MAG perceives the state-transition table (Table V) and achieves the following procedure:

- While the current state is RW_CM_EFS1, the received state results in verifying the occurrence of AFC_CM_PDS2 == true.
- As the relevant ALARM attribute holds "none", no alarm is thus announced.
- From the relevant CONTROLLING attribute, agent RW_MAG achieves the following flow rates: FR1 = R/7 and FR3 = R/7.
- As the relevant DIAGNOSIS attribute holds "not_needed", a diagnostic process is not launched.
- From the NEW STATE attribute, agent RW_MAG transits to the new state, which is another error-free state RW_CM_EFS3. From this state the pilot is provided with assessment, "RW sub-system copes with a degraded state of the AFS", and guidance, "none".

After achieving all the above procedures the operational structure of the fuel system appears different as fuel to feed the starboard engine only is drawn evenly from the seven tanks.

B. Second Simulated Failure scenario: Sensory Failure

Once the monitoring agent CD_MAG evaluates expression (3) with true, it perceives the state-transition table (Table III) and achieves the following procedure:

- From the relevant ALARM attribute, agent CD_MAG quotes and announces the alarm, "CD sub-system has a sensory failure".
- From the relevant CONTROLLING attribute, agent CD_MAG instructs the fuel system control unit to ignore measurements delivered by flow meter FC1 and depend alternatively on those delivered by flow meter FC2.
- From the relevant DIAGNOSIS attribute, agent CD_MAG quotes "Sensor FC1 has failed" and announces it as the diagnosed underlying cause.
- From the relevant NEW STATE attribute, agent CD_MAG transits to the new state, which is the permanent degraded state CD_CM_PDS1. From this state the pilot is provided with assessment, "CD sub-system operates degradedly" and guidance, "none".
- Agent CD_MAG also communicates the current state CD_CM_PDS1 to the parent (AFS_MAG).

When the agent AFS_MAG receives a message that conveys state CD_CM_PDS1, it perceives the state-transition table (Table I) and achieves the following procedure:

- While the current state is AFS_CM_EFS, the received state results in verifying the occurrence of CD_CM_PDS1 == true.
- From the relevant NEW STATE attribute, agent AFS_MAG transits to the new state, which is the permanent degraded state AFS_CM_PDS3. From this state the pilot is provided with assessment, "CD sub-system delivers degraded functionality" and guidance, "none".
- Agent AFS_MAG communicates state AFS_CM_PDS3 to the child agents: EF_MAG, LW_MAG and RW_MAG. As this state does not instantiate any active events of the children, no state transition is triggered and they do not take any action.

To demonstrate the ability of the monitor to deliver timely prognosis, let us assume that after controlling the fault, the aircraft has landed and during the pre-flying phase the refuelling mode is launched. This mode is triggered when the following expression is verified true:

VC1 == 1 AND VF1 == 0 AND VF4 == 0;

Then agent CD_MAG perceives the state-transition table (Table III) and achieves the following procedure:

- Executes the event on the table.
- As the ALARM attribute holds "none", no alarm is thus announced.
- As the relevant CONTROLLING attribute holds "none", no action is taken.
- As the relevant DIAGNOSIS attribute holds "not_needed", then diagnosis is not launched.
- From the relevant NEW STATE attribute, agent CD_MAG transits to the permanent degraded state of the refuelling mode CD_RM_PDS1. From this state the pilot is provided with prognosis of assessment, "CD sub-system has a sensory failure" and guidance, "Flow meter FC1 must be replaced".
- Agent CD_MAG also communicates the current state CD_RM_PDS1 to the parent agent (AFS_MAG).

When agent AFS_MAG receives a message conveying state CD_RM_PDS1, it perceives the state-transition table (Table I) and achieves the following procedure:

- While the current state is AFS_CM_PDS3, the received state results in verifying the occurrence of CD_RM_PDS1 == true.
- From the relevant NEW STATE attribute, agent AFS_MAG transits to the new state, which is the permanent degraded state of the refuelling mode AFS_RM_PDS3. From this state the pilot is provided with prognosis of assessment, "fuel system is not ready to achieve refuelling" and guidance, "none".
- Agent AFS_MAG communicates state AFS_RM_PDS3 to the children: EF_MAG, LW_MAG and RW_MAG. As this state does not instantiate any active events of the children, no state transition will be triggered and they do not take any action.

This prognosis would appear on the operator interface as shown by Fig. 16. It can be seen how the monitor avoids overwhelming the pilot with extra alarm information and

provides timely prognosis according to the evolutionary behaviour of the fuel system.

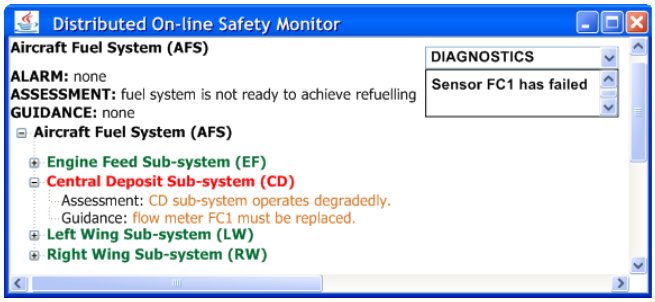

Figure 16. Operator interface provides the pilot with prognosis.

V. EVALUATION

The key aim of this work is to explore the concept of a novel architecture for a distributed safety monitor operating on a safety assessment model that hopefully could address some problems of earlier monitors and deliver effectively a broad range of safety monitoring tasks. Thus, it appears reasonable to compare the monitor developed herein with the earlier monitors and weigh their monitoring merits and drawbacks against each other.

In [12] and [13] two model-based monolithic monitors are developed to monitor critical applications and deliver fault detection and diagnosis, alarm annunciation and fault controlling. These monitors resemble the monitor developed herein in both the model-based approach and the deliverable monitoring tasks, but differ in their monolithic nature.

The monitoring model developed in [12] is quite similar in many ways to the monitoring model of the monitor developed herein; it too can be derived from the HiP-HOPS assessment model. The author of [12] has indentified the following limitations:

- Vulnerability to sensor failures.
- The centralised nature of the architecture has limited the applicability of the monitor and made it unable to scale up to monitor large-scale and distributed systems (e.g., nuclear power plants or chemical processes).

The monitor developed herein addresses to some extent these limitations via the following:

- The exploitation of techniques to validate sensor measurements, to a certain extent. With careful use of functional and hardware redundancy, single sensor failures can be captured and masked.
- As it is based on a distributed concept in which monitoring agents are deployed according to the hierarchical architecture of the monitored system, the monitor has an extendable architecture that makes it able to scale up and monitor large scale systems.

The monolithic monitor developed in [13] is also:

- Vulnerable to sensor failures.
- Unable to scale up to monitor large-scale and distributed systems.
- Unable to provide the operators with prognosis

As mentioned above, the monitor developed herein is provided with the required techniques and provisions that address these limitations.

The data-based monolithic monitors of [29] and [73] are developed to detect and diagnose faults of chemical processes. These monitors have a narrower scope than that of the developed monitor and differ in their monolithic nature and data-based monitoring knowledge. In the case of [29] the monitor has been tested on a large number of components, but it has no provision to cope consistently with dynamic behaviour and does not deal with sensor failures, unlike the herein-developed monitor, which is able to track and follow states and modes and has provision for sensory failures.

The model-based multi-agent monitors developed in [74], [75] and [76] are intended to be applied to large-scale and distributed processes. They match the herein-developed monitor in the delivery of this task and the exploitation of the model-based approach and multi-agent system. These monitors differ from the developed monitor in scope as they only focus on fault detection and diagnosis and they do not deliver the alarm organisation and fault controlling tasks.

In [74] and [75] the authors suggest the following limitations in their work:

- The monitor does not incorporate local diagnostic models. It depends, rather, on global diagnostic decision taken among the monitoring agents. This does not work well when more than one agent has faulty monitored conditions and in such a case the delivery of erroneous diagnostics is quite possible.
- The monitor is vulnerable to sensor failures.
- The monitor lacks a protocol for effective collaboration among its monitoring agents. In the currently implemented protocol there is no direct communication among the agents and messages may take a long time to be transmitted from one agent to another until they reach the intended agents. This delay could result in ineffective monitoring.

The monitor developed herein addresses to some extent these limitations with the following provisions:

- Providing every monitoring agent of level*1* with a diagnostic algorithm and a number of diagnostic models, so they can achieve local diagnosis and deliver accurate diagnostics.
- Applying techniques to filter and validate sensor measurements and detect, diagnose and control single sensor failure.
- Developing collaboration protocols by which messages can be exchanged among the agents directly and with no delay.

Two limitations have been observed in the monitor developed in [76]. Both concern the diagnostic process and can be listed as follows:

- As the diagnosis is achieved globally and depends mainly on exchange of messages among the high level agent and lower level agents, this may place a

heavy communication load on the higher level agent and consequently result in its late response.

- As the diagnostic decision is processed globally and based on identifying the anomalies among the consistent conditions, the appearance of a number of anomalous conditions could potentially mislead the diagnostic process.

As a precaution against such limitations, the monitor developed herein has been provided with the following strategies:

- The communication load is reduced, as the monitoring agents of level*1* are provided with diagnostic models and algorithms so they achieve a local and independent diagnostic process.
- The diagnostic process is achieved based on local observations of every sub-system and it is not affected by anomalous conditions of other sub-systems.

The data-based multi-agent monitors in [61] and [62] are developed to detect and diagnose faults of dynamic chemical processes. They match the herein-developed monitor in the delivery of this task and exploitation of the multi-agent system and they differ in their data-based monitoring knowledge. The monitor of [62] can detect and diagnose both single and multiple faults. Practically, this is an outcome of exploiting sensor fusion methods and also global fusion collaboration among the agents. Similarly, the herein-developed monitor is able to detect, diagnose and moreover control single and multiple faults (but not multiple dependent sensor failures). This has been materialised by providing agents of level*1* with effectively formalised monitoring expressions and models to achieve local detection and diagnosis. Moreover, across the hierarchical levels, agents collaborate to achieve global reasoning over the entire monitored process.

VI. Conclusion and Future Work

This paper proposed a distributed on-line safety monitor based on a multi-agent system and knowledge derived from model-based safety assessment. Agents exploit that knowledge to deliver a range of safety monitoring tasks extending from fault detection and diagnosis to alarm annunciation and fault controlling. The delivery of these tasks has been discussed and demonstrated in the context of a study of an aircraft fuel system.

The monitor can detect symptoms of failure 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. Furthermore, the monitor is able to validate sensory measurements, detect, diagnose and control faulty sensors.

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 monitoring model. The validation of the monitoring model is, therefore, 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.

Acknowledgment

The author would like to thank Professor Yiannis Papadopoulos (University of Hull).

References

[1] A. Dheedan and Y. Papadopoulos. “Model-Based Distributed On-line Safety Monitoring,” Proc. The Third International Conference on Emerging Network Intelligence (EMERGING 2011). Lisbon, Portugal, 20-25 November 2011.

[2] C. Billings, “Human-centred Aircraft Automation: A Concept and Guidelines,” Field CA United States: NASA Technical Memorandum TM-103885, NASA Ames Research Centre, Moffett, 1991. Available: http://www.archive.org/details/nasa_techdoc_19910022821 [Accessed 15th June 2012].

[3] I. Kim, “Computer-based Diagnostic Monitoring to Enhance the Human-machine Interface of Complex Processes,” Proc. Power Plant Dynamics, Control and Testing Symposium. Knoxville, TN, United States 27-29 May 1992.

[4] V. Venkatasubramanian, R., Rengaswamy, K. Yin and S. Kavuri, “A Review of Process Fault Detection and Diagnosis: Part I: Quantitative Model-based Methods,” Computers and Chemical Engineering, 27(3), 2003, pp 293-311.

[5] Y. Zhang and J. Jiang, “Bibliographical Review on Reconfigurable Fault-tolerant Control Systems,” Annual Reviews in Control, 32(2), 2008, pp 229-252.

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

[7] J. Ma and J. Jiang, “Applications of Fault Detection and Diagnosis Methods in Nuclear Power Plants: A Review,” Progress in Nuclear Energy, 53(3), 2011, pp 255-266.

[8] I. Kim and M. Modarres, “Application of Goal Tree Success Tree Model as the Knowledge-base of Operator Advisory Systems,” Nuclear Engineering and Design, 104 (1), 1987, pp 67-81.

[9] M. Modarres and S. Cheon, “Function-centred Modelling of Engineering Systems Using the Goal Tree Success Tree Technique and Functional Primitives,” Reliability Engineering & System Safety, 64(2), 1999, pp 181-200.

[10] D. Chung, M. Modarres and R. Hunt, “GOTRES: an Expert System for Fault Detection and Analysis,” Reliability Engineering & System Safety, 24(2), 1989, pp 113-137.

[11] L. Felkel, R. Grumbach and E. Saedtler, “Treatment, Analysis and Presentation of Information about Component Faults and Plant Disturbances,” Proc. symp Nuclear Power

Plant Control Instrument, IAEA-SM-266/40, 1978, pp 340–347.

[12] Y. Papadopoulos, "Model-Based System Monitoring and Diagnosis of Failures Using State-Charts and Fault Trees," Reliability Engineering and System Safety, 8(3), 2003, pp 325-341.

[13] H. Peng, W. Shang, H. Shi and W. Peng, "On-Line Monitoring and Diagnosis of Failures Using Control Charts and Fault Tree Analysis (FTA) Based on Digital Production Model." Proc. 2nd International Conference on Knowledge Science, Engineering and Management (KSEM'07). Lecture Notes in Computer Science (4798/2007). Berlin, Heidelberg: Springer, 2007, pp 544-549.

[14] M. Maurya, R. Rengaswamy and V. Venkatasubramanian, "A Signed Directed Graph-based Systematic Framework for Steady-state Malfunction Diagnosis inside Control Loops," Chemical Engineering Science, 61(6), 2006, pp 1790–1810.

[15] G. Dong, W. Chongguang, Z. Beike and M. Xin, "Signed Directed Graph and Qualitative Trend Analysis Based Fault Diagnosis in Chemical Industry," Chinese Journal of Chemical Engineering, 18(2), 2010, pp 265-276.

[16] S. Narasimhan, P. Vachhani and R. Rengaswamy, "New Nonlinear Residual Feedback Observer for Fault Diagnosis in Nonlinear Systems," Automatica, 44(9), 2008, pp 2222-2229.

[17] A. Zolghadri, D. Henry and M. Monsion, "Design of Nonlinear Observers for Fault Diagnosis: a Case Study," Control Engineering Practice, 4(11), 1999, pp 1535-1544.

[18] T. Chen, and R. You, "A Novel Fault-Tolerant Sensor System for Sensor Drift Compensation," Sensors and Actuators A: Physical, 147(2), 2008, pp 623-632.

[19] T. El-Mezyani, D. Dustegor, S. Srivastava and D. Cartes, "Parity Space Approach for Enhanced Fault Detection and Intelligent Sensor Network Design in Power Systems," Proc. IEEE'2010 Conference on Power and Energy Society General Meeting. Minneapolis, MN, 25-29 July 2010, pp 1-8.

[20] M. Borner, H. Straky, T. Weispfenning and R. Isermann, "Model Based Fault Detection of Vehicle Suspension and Hydraulic Brake Systems," Mechatronics, 12(8), 2002, pp 999-1010.

[21] M. Abdelghani and M. Friswell, "A Parity Space Approach to Sensor Validation," proc. of the International Society for Optical Engineering (SPIE'2001). USA, Bellingham: Society of Photo-Optical Instrumentation Engineers, ISSN: 0277-786X CODEN, 4359 (1), 2001, pp 405-411.

[22] W. Nelson, "REACTOR: an Expert System for Diagnosis and Treatment of Nuclear Reactor Accidents," proc AAAI 82, August, 1982, pp 296-301.

[23] T. Ramesh, S. Shum and J. Davis, "A Structured Framework for Efficient Problem-Solving in Diagnostic Expert Systems," Computers and Chemical Engineering, 12(9-10), 1988, pp 891-902.

[24] T. Ramesh, J. Davis and G. Schwenzer, "Catcracker: an Expert System for Process and Malfunction Diagnosis in Fluid Catalytic Cracking Units," proc. Annual Meeting of the American Institute of Chemical Engineering (AIChE), November 1989, San Francisco, CA.

[25] S. Rich, V. Venkatasubramanian, M. Nasrallah and C. Matteo, "Development of a Diagnostic Expert System for a Whipped Toppings Process," Journal of Loss Prevention in the Process Industries 2 (3), 1989, pp 145-154.

[26] M. Maurya, R. Rengaswamy and V. Venkatasubramanian, "Fault Diagnosis by Qualitative Trend Analysis of the Principal Components," Chemical Engineering Research and Design, 83 (9), 2005, pp 1122-1132.

[27] M. Maurya, R. Rengaswamy and V. Venkatasubramanian, "A Signed Directed Graph and Qualitative Trend Analysis-Based Framework for Incipient Fault Diagnosis," Chemical Engineering Research and Design, 85(10), 2007, pp 1407-1422.

[28] M. R. Maurya, P. K. Paritosh, R. Rengaswamy and V. Venkatasubramanian, "A Framework for On-line Trend Extraction and Fault Diagnosis," Engineering Applications of Artificial Intelligence, 23(6), 2010, pp 950-960.

[29] L. A. Rusinov, I. V. Rudakova, O. A. Remizova and V. Kurkina, "Fault Diagnosis in Chemical Processes with Application of Hierarchical Neural Networks," Chemometrics and Intelligent Laboratory Systems, 97 (1-15), 2009, pp 98-103.

[30] N. Kaistha and B. Upadhyaya, "Incipient Fault Detection and Isolation of Field Devices in Nuclear Power Systems Using Principal Component Analysis," Nuclear Technology, 136, 2001, pp 221-230.

[31] J. Miller, "Statistical Signatures Used with Principal Component Analysis for Fault Detection and Isolation in a Continuous Reactor," Journal of Chemometrics, 20(1-2), 2006, pp 34-42.

[32] S. Wold, A. Ruhe, H. Wold and W. Dunn, "The Collinearity Problem in Linear Regression, the Partial Least Squares (PLS) Approach to Generalized Inverses," SIAM Journal of Science Statistical Computer, 5(1984), 1984, pp 735-743.

[33] S. Hwang, J. Lin, G. Liang, Y. Yau, T. Yenn, and C. Hsu, "Application Control Chart Concepts of Designing a Pre-alarm System in the Nuclear Power Plant Control Room," Nuclear Engineering and Design, 238(12), 2008, pp 3522-3527.

[34] G. Jang, D. Seong, J. Keum, H. Park, and Y. Kim, "The Design Characteristics of an Advanced Alarm System for SMART," Annals of Nuclear Energy, 35(6), 2008, pp 1006-1015.

[35] W. Brown, J. O'Hara and J. Higgins, "Advanced Alarm Systems: Revision of Guidance and its Technical Basis," US Nuclear Regulatory Commission, Washington, DC (NUREG-6684), 2000. Available: http://www.bnl.gov/humanfactors/files/pdf/NUREG_CR-6684.pdf [Accessed 22nd February 2012].

[36] B. Oulton, "Structured Programming Based on IEC SC 65 A, Using Alternative Programming Methodologies and Languages with Programmable Controllers," proc IEEE Conference on Electrical Engineering Problems in the Rubber and Plastic Industries, IEEE no: 92CH3111-2, IEEE service centre, 31-14 April, Akron, OH , USA, 1992, pp 18-20.

[37] A. Ghariani, A. Toguyeni and E. Craye, "A Functional Graph Approach for Alarm Filtering and Fault Recovery for Automated Production Systems," proc 6th International Workshop on Discrete Event Systems (WODES'02). 02-04 October 2002, Zaragoza, Spain, pp 289-294.

[38] J. Lee, J. Kim, J. Park, I. Hwang and S. Lyu, "Computer-Based Alarm Processing and Presentation Methods in Nuclear Power Plants," proc World Academy of Science, Engineering and Technology. Library of Congress, Electronic Journals Library, 65(2010), 2010, pp 594-598.

[39] C. Yu and B. Su, "Eliminating False Alarms Caused by Fault Propagation in Signal Validation by Sub-grouping," Progress in Nuclear Energy, 48(4), 2006, pp 371-379.

[40] P. Baraldi, A. Cammi, F. Mangili and E. Zio, "An Ensemble Approach to Sensor Fault Detection and Signal Reconstruction for Nuclear System Control," Annals of Nuclear Energy, 37(6), 2010, pp 778-790.

[41] I. Kim, M. Modarres and R. Hunt, "A Model-based Approach to On-line Disturbance Management: The

Models," Reliability Engineering and System Safety, 28(3), 1990, pp 265-305.

[42] I. Kim, M. Modarres and R. Hunt, "A Model-based Approach to On-line Process Disturbance Management: the Models," USA: System Research Centre: University of Maryland, SRC TR 88-111, 1988. Available from: http://drum.lib.umd.edu/bitstream/1903/4837/1/TR_88-111.pdf [Accessed 4th May 2012].

[43] R. Clark, "Instrument Fault Detection," IEEE Transactions on Aerospace and Electronic Systems, 14(3), 1978, pp 456-465.

[44] J. O'Hara, W. Brown, P. Lewis and J. Persensky, "Human-System Interface Design Review Guidelines," Energy Sciences & Technology Department and Brookhaven National Laboratory, Upton, NY 11973-5000, 2002. Available: http://www.nrc.gov/reading-rm/doc-collections/nuregs/staff/sr0700/nureg700.pdf [Accessed 22nd February 2012].

[45] J. O'Hara, W. Brown, B. Halbert, G. Skraaning, J. Wachtel and J. Persensky, "The Use of Simulation in the Development of Human Factors Guidelines for Alarm Systems," proc 1997 IEEE 6th Conference on Human Factors and Power Plants, Global Perspectives of Human Factors in Power Generation. 08 - 13 Jun 1997, Orlando, FL, USA, pp 1807-1813.

[46] E. Roth, and J. O'Hara, "Integrating Digital and Conventional Human-System Interfaces: Lessons Learned from a Control Room Modernization Program," Division of Systems Analysis and Regulatory Effectiveness Office of Nuclear Regulatory Research U.S. Nuclear Regulatory Commission, Washington, DC 20555-0001 NRC Job Code W6546, 2002. Available:http://www.nrc.gov/reading-rm/doc-collections/nuregs/contract/cr6749/6749-021104.pdf [Accessed 23rd February 2012].

[47] J. Anderson, "Alarm Handler User's Guide," The University of Chicago, as Operator of Argonne National Laboratory, Deutches Elektronen-Synchrotron in Der Helmholtz-Gemelnschaft (DESY) and Berliner Speicherring-Gesellschaft fuer Synchrotron-Strahlung mbH (BESSY), 2007. Available: http://www.slac.stanford.edu/comp/unix/package/epics/extensions/alh/alhUserGuide.pdf [Accessed 23rd May 2012].

[48] J. Jiang, "Fault-tolerant Control Systems an Introductory Overview," Automatica SINCA, 31(1), 2005, pp 161–174.

[49] R. Patton, "Fault-tolerant Control: the 1997 Situation," proc 3rd IFAC symp on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS'97). August 1997, Hull, United Kingdom, pp 1033-1055.

[50] C. Seo and B. Kim, "Robust and Reliable H∞ Control for Linear Systems with Parameter Uncertainty and Actuator Failure," Automatica, 32(3), 1996, pp 465-467.

[51] R. Srichander and B. Walker, "Stochastic Stability Analysis for Continuous-Time Fault Tolerant Control Systems," International Journal of Control, 57(2), 1993, pp 433–452.

[52] I. Lopez and N. Sarigul-Klijn, "A Review of Uncertainty in Flight Vehicle Structural Damage Monitoring, Diagnosis and Control: Challenges and Opportunities," Progress in Aerospace Sciences, 46 (7), 2010, pp 247-273.

[53] Delta Virtual Airlines, "Boeing 767-200/300ER/400ER Operating Manual," 3rd ed., Delta Virtual Airlines, 2003. Available: http://www.deltava.org/library/B767%20Manual.pdf [Accessed 5th March 2012].

[54] Q. Zhao and J. Jiang, "Reliable State Feedback Control System Design against Actuator Failures," Automatica, 34(10), 1998, pp 1267-1272.

[55] W. Heimerdinger and C. Weinstock, "A Conceptual Framework for System Fault Tolerance," Technical Report CMU/SEI-92-TR-033 ESC-TR-92-033, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, 1992.

[56] M. Shooman, "Reliability of Computer Systems and Networks: Fault Tolerance, Analysis and Design," USA, New York: John Wiley & Sons, Inc, 2002.

[57] 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.

[58] X. Ren, H. Thompson and P. Fleming, "An Agent-based System for Distributed Fault Diagnosis," International Journal of Knowledge-Based and Intelligent Engineering Systems, 10(2006), 2006, pp 319-335.

[59] S. Eo, T. Chang, B. Lee, D. Shin and E. Yoon, "Function-behaviour Modelling and Multi-agent Approach for Fault Diagnosis of Chemical Processes," Computer Aided Chemical Engineering, 9 (2001), 2001, pp 645-650.

[60] S. Eo, T. Chang, B. Lee, D. Shin and E. Yoon, "Cooperative Problem Solving in Diagnostic Agents for Chemical Processes," Computers & Chemical Engineering, 24 (2-7), 2000, pp 729-734.

[61] Y. Ng and R. Srinivasan, "Multi-agent Based Collaborative Fault Detection and Identification in Chemical Processes," Engineering Applications of Artificial Intelligence, 23(6), 2010, pp 934-949.

[62] G. Niu, T. Han, B. Yang and A. Tan, "Multi-agent Decision Fusion for Motor Fault diagnosis," Mechanical Systems and Signal Processing, 21(3), 2007, pp 1285-1299.

[63] C. Wallace, G. Jajn and S. McArthur, "Multi-agent System for Nuclear Condition Monitoring," proc of the 2nd International Workshop on Agent Technologies for Energy System (ATES'11), a workshop of the 10th International Conference of Agent and Multi-agent System (AAMAS'11), 2nd of May 2011, in Taipei, Taiwan.

[64] A. Sayda, "Multi-agent Systems for Industrial Applications: Design, Development, and Challenges," In: Alkhateeb F., Al Maghayreh E., Abu Doush L., ed. Multi-Agent Systems - Modeling, Control, Programming, Simulations and Applications. Rijeka, Croatia: InTech, 2011, pp 469-494.

[65] E. Mangina, "Intelligent Agent-based Monitoring Platform for Applications in Engineering," International Journal of Computer Science & Applications, 2(1), 2005, pp 38-48.

[66] HSE, "Better Alarm Handling, HSE Information Sheet," UK: Health and Safety Executive, chemical sheet No 6, 2000. Available from: http://www.hse.gov.uk/pubns/chis6.pdf, [Accessed 13th June 2012].

[67] 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 April 2012]

[68] BEA, "Safety Investigation into the Accident on 1 June 2009 to the Airbus A330-203, flight AF447," France: Bureau of Investigations and Analysis for the safety of civil aviation (BEA), 2011. Available: http://www.bea.aero/fr/enquetes/vol.af.447/note29juillet2011.en.pdf [Accessed 2nd March 2012].

[69] M. Breen, "Experience of Using a Lightweight Formal Specification Method for a Commercial Embedded System Product Line," Requirements Engineering Journal, 10(2), 2005, pp 161-172.

[70] S. McArthur, E. Davidson, J. Hossack, and J. McDonald, "Automating Power System Fault Diagnosis through Multi-agent System Technology," proc 37th Annual Hawaii International Conference on System Sciences (HICSS'04), 5-8 Jan 2004, Big Island, Hawaii, pp 1-4.

[71] L. Padgham and M. Winikoff, Developing Intelligent Agent Systems: A Practical Guide. Chichester: John Wily & Sons LTD, 2004.

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

[73] X. Doan and R. Srinivasan, "Online monitoring of multi-phase batch processes using phase-based multivariate statistical process control," Computers & Chemical Engineering, 32(1-2), 2008, pp 230-243.

[74] S. Y. Eo, T. S. Chang, B. Lee, D. Shin and E. S. Yoon, "Cooperative problem solving in diagnostic agents for chemical processes," Computers & Chemical Engineering, 24 (2-7), 2000, pp729-734.

[75] S. Y. Eo, T. S. Chang, B. Lee, D. Shin and E. S. Yoon, "Function-behavior modeling and multi-agent approach for fault diagnosis of chemical processes," Computer Aided Chemical Engineering, 9 (2001), 2001, pp 645-650.

[76] X. Ren, H. A. Thompson, and P. J. Fleming, "An agent-based system for distributed fault diagnosis," International Journal of Knowledge-Based and Intelligent Engineering Systems, 10(2006), 2006, pp 319-335.