World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:6, No:11, 2012

Multi-agent On-line Monitor for the Safety of Critical Systems

Amer A. Dheedan

Abstract**—**Operational safety of critical systems, such as nuclear power plants, industrial chemical processes and means of transportation, is a major concern for system engineers and operators. A means to assure that is on-line safety monitors that deliver three safety tasks; fault detection and diagnosis, alarm annunciation and fault controlling. While current monitors deliver these tasks, benefits and limitations in their approaches have at the same time been highlighted. Drawing from those benefits, this paper develops a distributed monitor based on semi-independent agents, i.e. a multi-agent system, and monitoring knowledge derived from a safety assessment model of the monitored system. Agents are deployed hierarchically and provided with knowledge portions and collaboration protocols to reason and integrate over the operational conditions of the components of the monitored system. The monitor aims to address limitations arising from the large-scale, complicated behaviour and distributed nature of monitored systems and deliver the aforementioned three monitoring tasks effectively.

Keywords**—**Alarm annunciation, fault controlling, fault detection and diagnosis.

I. Introduction

SINCE their emergence, multi-agent systems have played a key role in addressing the rigidity of monolithic on-line safety monitors. Indeed, they have divided, distributing and rationalising the monitoring concepts [1].

Multi-agent monitors are typically developed from monitoring models that hold reference knowledge and a number of intelligent agents deployed to monitor the conditions of the monitored system. Agents work as engines that execute monitored conditions on the monitoring model, so distinction between normal and abnormal conditions can be made and monitoring tasks delivered. To achieve the required integration among their monitoring models and monitored conditions and deliver consistent safety tasks at the system level, agents are typically provided with corresponding protocols to allow the required collaboration.

In [2], a fault detection and diagnosis technique is developed from the deployment of a number of agents on two levels. A top level agent is provided with a Markov model to reason over conditions notified by the low level agents. Low level agents reason over the parameters of the monitored system and each is provided with a monitoring model; a functional model augmented with operators' expertise. In [3] another fault detection and diagnosis concept is developed from a number of agents, each of which is deployed to monitor the functionality of the entire monitored system using different reasoning methods: self-organisation maps, principal component analysis, neural network and non-parametric reasoning. To avoid conflicting local monitoring results, agents are also provided with decision fusion methods, namely, voting-based fusion and Bayesian probability fusion. Global consensus on detection and diagnosis decisions are agreed through collaboration among the agents.

Multi-agent monitors have also contributed to address problems of alarm annunciation. In [4] a number of agents, whose reasoning is based on the Dempster Shafer evidence theory, are deployed over the components of the monitored system. During a disturbance, agents prioritise alarms locally and collaborate on a composite global alarm to be announced on the operators' interface. In [5] a number of agents are provided with determination algorithms and deployed over the components of the monitored plant to process alarms in control rooms, distinguish between causal and consequent alarms and announce them in a well-organised presentation.

A key role of multi-agents systems is also demonstrated in addressing problems of fault controlling of large-scale processes. In [6], for example, a number of agents are deployed over the monitored process and each is provided with a corresponding functional model. Agents work collaboratively to achieve antifault-propagation supervisory control between the components of the monitored process. In [7] a quite similar technique is also developed; agents are also deployed over the monitored process, but provided with corresponding knowledge as a directed graph (digraph) to control faults.

In practice, critical systems are implemented as a set of sub-systems which exist in a complex cooperative structure and coordinate to accomplish system functions. Systems are also large and complex and show dynamic behaviour that includes complex mode and state transitions. As a result, a too heavy reasoning load may rest on the shoulders of system operators, especially during emergency conditions. Operators need to analyse alarms, understand the underlying conditions and take prompt correct control procedures to control faults. Such a load results, on many occasions, in late or even incorrect responses by operators, which has resulted in a number of fatal accidents. Consider, for example, the accident of 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 [8] and the explosion and fire at the Texaco Milford Haven refinery in 1994, in which 26 workers were injured associated with great loss of assets [9].

Rigorous integration and consistency in the delivery of prompt fault detection and diagnosis, effective alarm annunciation and automated fault controlling becomes thus a

Amer A. Dheedan is with Delmon University, Manama, Kingdom of Bahrain (e-mail: amer@delmon.bh).

precondition to support the operators, avoid their incorrect actions and ultimately boost the operational safety of critical systems.

However, as shown in the aforementioned discussion, the role of the current on-line safety monitors is implemented largely to deliver one or other of the three tasks, separately. This deficiency can be attributed to two main reasons. The first is the fact that the development of monitoring knowledge that can inform the on-line reasoning and support the delivery of the three tasks needs the involvement of different analysis processes and the expertise of system engineers and operators. Producing such knowledge is factually very expensive and its consistency is hard to be guaranteed [10]. The second reason is that integration among the distributed monitoring models and monitored conditions of each deployed agent requires informative communication supported by an effective collaboration protocol to deliver integrated safety tasks [11][12][13].

This paper addresses the problem by developing a multi-agent on-line safety monitor that delivers three integrated safety tasks: prompt fault detection and diagnosis, effective alarm annunciation and fault controlling. The monitor is developed from a number of belief-desire-intention (BDI) agents and a distributed monitoring model derived from the design models and safety assessment model of the monitored system. Agents are deployed in a hierarchical approach to monitor the conditions of the sub-systems of the monitored system and collaborate to integrate and deliver the three tasks at the system level.

Design models and safety assessment model are developed during the development (off-line phase) of the system and they hold thorough and consistent knowledge about the normal and abnormal behaviour of that system. The utility of the models ceases after certifying the safe deployment of the system. Their exploitation to extract monitoring knowledge is, thus, cost-effective [14]. In this paper, a derivation and formalisation technique is also developed to bring knowledge of those models forward to serve in the context of on-line monitoring. The targeted assessment model is the one produced via application of Architecture Analysis Design Language (AADL), which is a state-of-the-art analysis technique [15] [16].

The remainder of this paper is organised as follows: section two describes the nature of the monitored system, i.e. modern critical systems. Section three discusses the position, role, and constituents of the monitor. Section four tests the monitor through the application to an aircraft brake system. Section five draws a conclusion and proposes further work.

II. The Monitored System

Large scale and dynamic behaviour are two common aspects of modern critical systems, e.g. 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 over achieving 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 monitor such systems and deliver correct monitoring tasks.

To model the mutual relations among the components of a system, a hierarchical organisation is commonly used to arrange them in a number of levels. Across the levels components appear as parents, children and siblings. To facilitate an architectural view of systems, we introduce a components classification as shown in Fig. 1. Hierarchical levels are classified into three types: the lowest level (level*0*) is classified as the basic components (BC) level. 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.

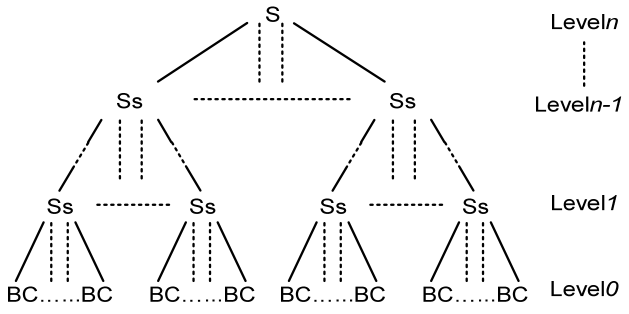


Fig. 1 Architectural view of the monitored system

To model the behaviour of the monitored system, it might be necessary 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 sub-systems (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*.

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 faults for a while until they are controlled totally, 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 activating 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: (a) a *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; (b) a *Permanent Degraded State (PDS)*, in which an uncontrollable fault occurs, but the safe part of the functionality can be delivered; (c) the *Failure State (FS)* in which the intended function is totally undeliverable.

Thus, it can be said that events that are initiated by the basic components trigger and make the normal and abnormal behaviour of systems. According to the behaviour they trigger, events are classified into three types. The first type is normal events whose occurrences result in transitions from EFS to any other EFS of a different mode, e.g. transition from the EFS of cruising mode to the EFS of approaching mode of an aircraft. The second type is failure events whose occurrences result in transition from an EFS to any ES, due to a fault. The last type is corrective events whose occurrences result in transition from a TDES to a PDS or EFS; due to applying corrective measures.

To track the behaviour of the monitored system, such events should be continuously monitored. To achieve that, the best hierarchical level at which to monitor these events should be identified. Three monitoring factors can practically identify that level; early fault detection, computational cost and behavioural understanding. Achieving trade-off among these factors could help in identifying the targeted level. Fig. 2 illustrates the relationships among the architectural levels and those factors.

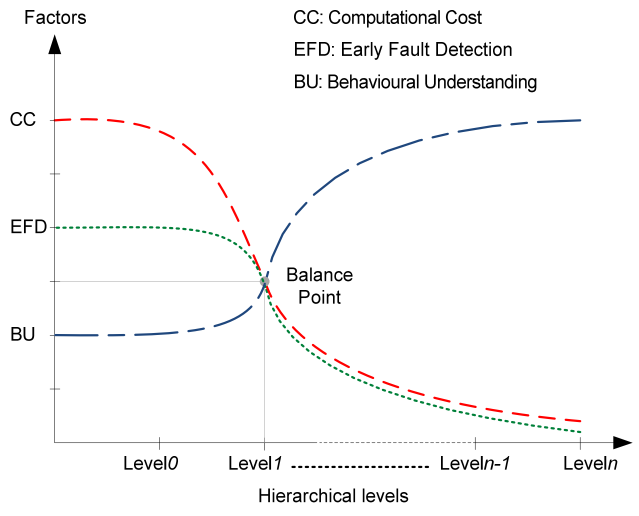


Fig. 2 Balance point between three monitoring factors and architectural levels

At level*1* the occurrence of events could be identified as either normal or abnormal, e.g. the decreasing of aircraft velocity and altitude seem normal when the flight control computer has already launched the approaching phase 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 a malfunction is detected while in its early stages. Moreover, 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. Multi-Agent On-line Safety Monitor

The monitor takes a position between the monitored system and the operators' interface. During normal conditions, the monitor provides simple feedback to confirm faultless operation. Its actual role is during abnormal conditions, which are triggered by and follow the occurrence of faults. The monitor delivers three safety tasks: prompt fault detection and diagnosis, effective alarm annunciation and fault controlling.

The term prompt, associated fault detection and diagnosis, refers to the timeliness of detecting with 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 (level*1*) to monitor the operational parameters and also by setting and monitoring those parameters against well-defined thresholds.

Effective alarm annunciation involves setting well-defined 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:

1. Track the behaviour of the monitored system and distinguish among the occurrence of normal, corrective and failure events.
2. Release alarm only on the occurrence of genuine symptoms of faults and not on other events, such as consequent, precursor or causal events.
3. Incorporate alarm information that could help the operators to control the abnormal conditions. Information is presented as assessment of the operational conditions following the occurrence of the fault and guidance on the corrective actions that should be taken manually by the operators.
4. Prioritise alarm presentation. This can be achieved by highlighting the important alarms through different colours, vibration or alerting sounds, and hiding the less important alarm information, e.g. optional access to the diagnostics list on the operators interface.

Fault controlling is achieved by both active fault-tolerant controlling and passive fault-tolerant controlling and also by announcing assessment and guidance to support the manual fault controlling of abnormal conditions that may fall beyond the trained skills of the operators.

Fig. 3 shows an illustrative view of the AADL assessment model from which the *distributed monitoring model*, a constituent of the monitor, is derived. It consists of a behavioural model as a hierarchy of state-machines and fault propagation models as a number of state-machines. 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*. 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, such that the occurrence of the events and symptoms can be verified computationally by instantiating and evaluating monitoring expressions with real–time conditions. Events verification supports tracking the behaviour of the monitored system and similarly symptoms verification 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 formalised model into a number of models without violating the consistency of the encoded knowledge.

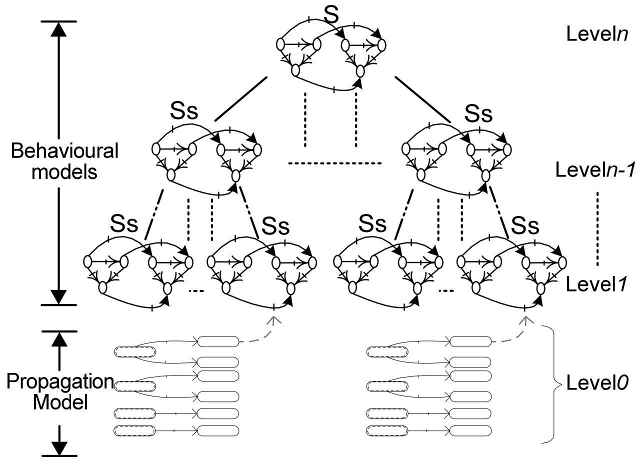


Fig. 3 Illustrative view of the AADL safety assessment model

Fig. 4 illustrates the hierarchical deployment of the multi-agent system, the second constituent of the monitor, over the monitored system. According to their deployment, monitoring agents appear as follows: a number of agents deployed to monitor sub-systems, which appear as Ss_MAG and an agent that monitors the system, which appears as S_MAG.

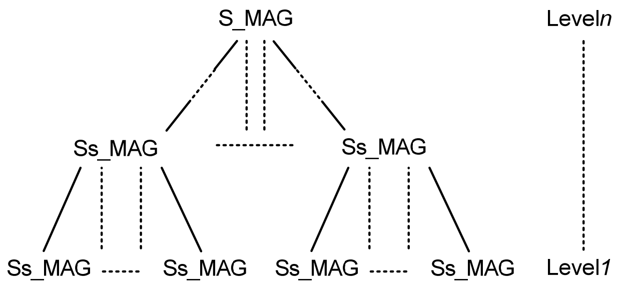


Fig. 4 Illustrative view of the hierarchical deployment of agents

A. Distributed Monitoring Model

In the light of the three intended monitoring tasks, agents should (a) track the behaviour of the monitored components over different states, i.e. error-free states (EFSs) and error states (ESs); (b) distinguish between normal and abnormal conditions; (c) provide the operators with information that confirm whether the conditions are normal or not; in abnormal conditions, agents should provide alarm, assessment, guidance and diagnostics; (d) be able to apply corresponding corrective measures to control faults.

Tracking the behaviour of the monitored system and its components requires informing the reasoning of the agent with behavioural knowledge. This knowledge can be derived from the hierarchy of the behavioural state-machines of the AADL model (Fig. 3). In the state-machine of the sub-systems of level*1*, trigger events are originated by (a) the BCs of level*0*, which might be failure, corrective or normal events; (b) parent states (EFSs and ESs). In the state-machine of a sub-system of the levels extending from level*2* to level*n-1*, trigger events appear as EFSs and ESs of the parent and children. Finally, in the state-machine of the system, i.e. level*n*, events appear as EFSs and ESs of the children. Such a communication among the hierarchical state-machines can be illustrated by the following example: the failure state (FS) of an engine of a two-engine aircraft triggers a transition to the permanent degraded state (PDS) in the state-machine of the power plant sub-system. The PDS, in turn, triggers a transition to another error-free state EFS of the operative engine in which the lost functionality of the faulty engine is compensated.

To distinguish between normal, fault and corrective events, the applied principle is that an alarm should be released on the occurrence of failure events only. Thus, corresponding alarm clauses should be associated with the failure events that can be verified at 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, the otherwise clause means that it is a failure event and the associated clause should be quoted and released as an alarm. 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 associated with the states.

To find the appropriate place of incorporating corrective measures, further consideration for 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 and its underlying causes are in a one-to-many relationship. In practice, measures to correct any of those causes may vary from one cause to another, so they should be incorporated in the diagnostic model (e.g. fault propagation state-machines), 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 the flight control system (FCS) of modern aircraft, switching to the backup computer sub-system at level*1* is instructed directly by the FCS at level*2*, whenever the primary computer sub-system (at level*1*) fails. Corrective 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, i.e. the detected fault and its underlying cause are in a one-to-one relationship.

State-transition tables are suggested to hold the behavioural knowledge derived from the AADL model. A state-transition table is usually defined as an alternative, executable and formal form to present a state-machine and it also offers the required capacity and flexibility to incorporate knowledge about the operational conditions [17].

According to the aforementioned monitoring needs, state-transition tables of levels extending from level*2* to level*n-1* record the following: (a) state transitions (current state, trigger event and new state); (b) assessment and guidance clauses. State-transition tables of level*1* would record the following: (a) state transitions (current state, trigger event and new state); (b) alarm, assessment and guidance clauses; (c) corrective measures; (d) diagnosis status which confirms whether a diagnostic process is needed depending on the relationships between the failure events and the underlying causes.

Table I shows the state-transition table of the aircraft brake system, which is the case study presented in this paper. The system has two sub-systems; Left-side wheel Brake (LWB) sub-system and Right-side Wheel Brake (RWB) sub-system. It can be seen how the states of those sub-systems trigger transitions of the brake system. Consider, for example, the first monitoring expression (event) in the table:

LWB_NM_FS == true **OR** RWB_NM_FS == true

The expression can be interpreted as: if the LWB or RWB is in a failure state (FS) of the normal mode (NM) then the aircraft brake system (ABS) should transit from the error-free state (EFS) of the normal mode (NM), i.e. ABS_NM_EFS, to the temporary degraded or failure state (TDFS) of the same mode, i.e. ABS_NM_TDFS. The table incorporates also assessment and guidance on the conditions at the system level.

Table II shows the state-transition table of the right-side wheel brake (RWB) sub-system. It can be seen that the table of level*1*'s sub-system incorporates additionally alarm, controlling and diagnosis attributes.

In real time, monitoring agents evaluate only *active events*, which represent exits from the current states. Agents cyclically update expressions of those events with up-to-date sensory measurements and evaluate them, thereby achieving a *monitoring cycle.* After achieving every cycle, a new cycle is launched in which up-to-date measurements are collected and every expression is evaluated again. This state focus effectively reduces the work load of the agents and rationalises the monitoring process.

Monitoring knowledge held by the state-transition tables of level*1*'s sub-systems and the higher levels could satisfy (a) tracking the behaviour of the system and its sub-systems; (b) announcement of alarm and multi-level assessment and guidance; (c) fault controlling at those levels.

TABLE I
STATE-TRANSITION TABLE OF THE AIRCRAFT BRAKE SYSTEM

Current state	Conditions	Event	New state
ABS_NM_EFS	**Assessment**: normal line is operative and brake could be applied automatically or manually. **Guidance**: switching between manual and auto-brake is possible.	LWB_NM_FS == true **OR** RWB_NM_FS == true	ABS_NM_TDFS
ABS_NM_TDFS	**Assessment**: brake system is in a temporary failure. **Guidance**: fault controlling is in progress.	LWB_AM_EFS == true **AND** RWB_AM_EFS== true	ABS_AM_EFS
ABS_AM_EFS	**Assessment**: LWB and RWB are pressured by the alternative line. **Guidance**: only manual brake is applicable.	LWB_AM_FS == true **OR** RWB_AM_FS == true	ABS_AM_TDFS
ABS_AM_TDFS	**Assessment**: brake system is in a temporary failure. **Guidance**: fault controlling is in progress.	LWB_ACM_EFS== true **AND** RWB_ACM_EFS== true	ABS_ACM_EFS
ABS_ACM_EFS	**Assessment**: LWB and RWB are pressured by the accumulative line. **Guidance**: apply manual brake but anti-skid is unavailable.	LWB_ACM_FS == true **OR** RWB_ACM_FS== true	ABS_ACM_FS
ABS_ACM_FS	**Assessment**: brake system has failed permanently. **Guidance**: emergency conditions.	none	none

TABLE II
STATE-TRANSITION TABLE OF THE RWB SUB-SYSTEM OF THE AIRCRAFT BRAKE SYSTEM

Current state	Conditions	Event	Alarm	Controlling	Diagnosis	New state
RWB_NM_EFS	**Assessment:** RWB operates normally. **Guidance:** none	RN_ASV_OP > RN_ASV_C + 50 OR RN_ASV_OP < RN_ASV_C - 50;	normal brake failed	- RA_CV_C = 1; - RN_CV_C = 0;	needed	RWB_NM_FS
		ABS_NM_TDFS == true	none	- RA_CV_C = 1; - RN_CV_C = 0;	not_needed	RWB_AM_EFS
		N_PS_P – 1300 < 50	low pressure on normal line	- RA_CV_C = 1; - RN_CV_C = 0;	Pressure at the normal line is low.	RWB_NM_FS
RWB_NM_FS	**Assessment:** normal line of RWB has failed. **Guidance:** fault controlling is in progress.	ABS_NM_TDFS == true	none	not_needed	not_needed	RWB_AM_EFS
RWB_AM_EFS	**Assessment:** RWB is pressured by the alternative line. **Guidance:** apply manual brake.	RA_ASV_OP > RA_ASV_C + 50 OR RA_ASV_OP < RA_ASV_C - 50;	alternative brake failed	- RAC_CV_C=1; - RA_CV_C = 0;	needed	RWB_AM_FS
		ABS_AM_TDFS == true	none	- RAC_CV_C=1; - RA_CV_C = 0;	not_needed	RWB_ACM_EFS
		A_PS_P – 1300 < 50	low pressure at alternative line	- RAC_CV_C=1; - RA_CV_C = 0;	Pressure at the alternative line is low.	RWB_AM_FS
RWB_AM_FS	**Assessment:** alternative line of RWB has failed. **Guidance:** fault controlling is in progress.	ABS_AM_TDFS == true	none	not_needed	not_needed	RWB_ACM_EFS
RWB_ACM_EFS	**Assessment:** RWB is pressured by the accumulative line. **Guidance:** apply manual brake.	RAC_MV_OP > RAC_MV_C + 50 OR RAC_MV_OP < RAC_MV_C - 50;	accumul-ative brake failed	impossible	needed	RWB_ACM_FS
		AC_PS_P – 1300 < 50	low pressure on accumul-ative line	impossible	Pressure on the accumulative line is low.	RWB_ACM_FS
RWB_ACM_FS	**Assessment:** RWB fails there is no brake on the right-side landing gear **Guidance:** no brake is available	none	none	not_needed	not_needed	none

To support diagnosing the underlying causes of the faults and also controlling the faults of the basic components (level*0*), a diagnostic model is needed. This can be derived from the fault propagation state-machines of the AADL model. Fig. 5 show the derived diagnostic model of the failure event "normal brake failed"; it can be seen as the first event in Table II.

FailureEvent: RN_ASV_OP > RN_ASV_C + 50 **OR** RN_ASV_OP < RN_ASV_C - 50.
Propagator: RN_ASV **OR** RN_ABV **OR** RN_CV.

EStateName: RN_ASV.
Symptom: RN_CV_PO == 1 **AND** RN_ABV_IP – RN_ABV_OP < 100.
Fault: anti-skid valve RN_ASV is faulty.
Controlling: none.

EStateName: RN_ABV.
Symptom: RN_ABV_IP – RN_ABV_OP > 100.
Fault: auto-brake valve RN_ABV is faulty.
Controlling: none.

EStateName: RN_CV.
Symptom: RN_CV_PO == 0.
Fault: control valve RN_CV is faulty.
Controlling: none.

Fig. 5 Diagnostic model derived from fault propagation model of ADDL assessment model

For every failure events that is in a one-to-many relationship with its underlying causes there is a diagnostic model. Links between failure events and their relevant diagnostic models are established through the appearance of the failure event at the top of the corresponding diagnostic model (see the first event in Table II and the "FailurEvent" field in the diagnostic model Fig.5).

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

B. *Multi-agent System*

In addition to the common ability of intelligent agents to achieve integrated reasoning among distributed processes [18], 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. 6 shows the reasoning model of the BDI 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 course of action, i.e. plan. 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

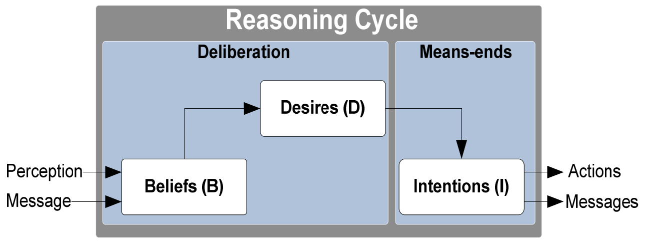


Fig. 6 BDI agent reasoning model

As agents are deployed hierarchically, each Ss_MAG of level*1* updates its belief base by perceiving (a) its own portion of the monitoring model which consists of a state-transition table and a number of diagnostic model; (b) sensory measurements that are taken to instantiate and evaluate monitoring expressions; (c) messages that are received from the parent to inform the given Ss_MAG about the new states and from siblings, in which they either ask for or tell the given Ss_MAG about global measurements, as there is potential 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 and guidance and control faults. On the achievement of the global desire, the intentions are to exchange messages to (a) inform the parent about the new states; and (b) tell or ask the siblings about global measurements.

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 model, which consists of a state-transition table of the assigned sub-system; (b) messages received from the parent and children to inform 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 and guidance. On the global desire, the intention is to exchange messages with the parent and child agents to inform each other about the new states. The perceptions, desires and intentions of the S_MAG of level*n* 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 [19], Fig. 7 shows the collaboration protocols among agents to track the behaviour of the monitored system. Fig. 8 shows the collaboration protocol among the Ss_MAGs of level*1* in which they share their sensory measurements.

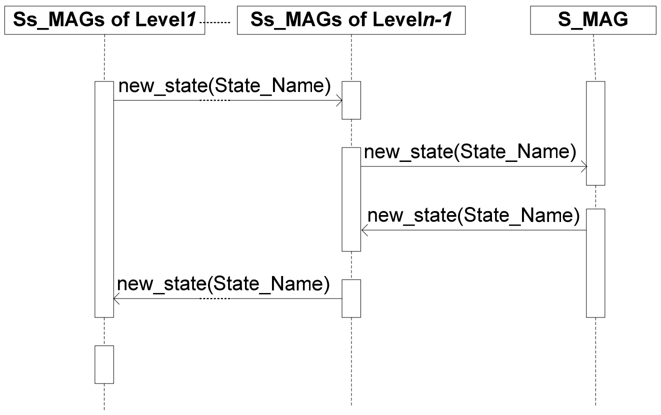


Fig. 7 Collaboration protocol across the hierarchical levels

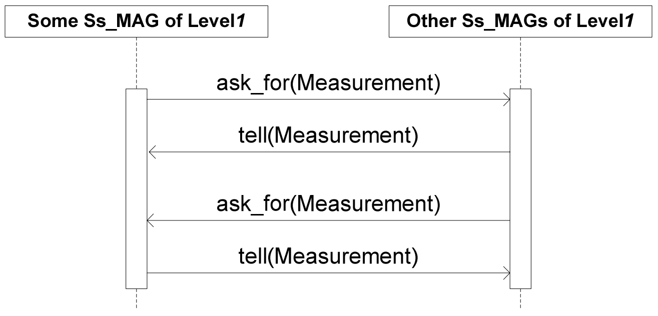


Fig. 8 Collaboration protocol among Ss_MAGs of level*1*

IV. Case Study: Aircraft Brake System

Fig. 9 shows a physical illustration of a hypothetical aircraft brake-system (ABS). The main function of the ABS is to slow down the aircraft during the taxiing and landing phases and achieve safe retardation in the case of rejected take-off. The brake function of the ABS is supported by anti-skid and an optional selection between auto-brake in which the pilot pre-arms the rate of deceleration before the landing phase and manual brake in which the pilot applies the brake manually by depressing two pedals.

Basic components of the ABS include valves, sensors and three redundant pressure lines; normal, alternative and accumulative lines. The components are arranged in two sub-systems; Left-side Wheel Brake (RWB) sub-system and Right-side Wheel Brake (RWB) sub-system. Passive fault-tolerant controlling is implemented to control faults of the ABS. Initially the brake is pressured by the normal line; should this fail, it is isolated and the alternative line is activated. Should the alternative line fail, it too is isolated and the accumulative line is activated. A pressure line may fail due to a drop in the pressure to less than 1300 PSI or due to a fault of a basic component of that line. A brake system control unit (BSCU) is incorporated to control the ABS.

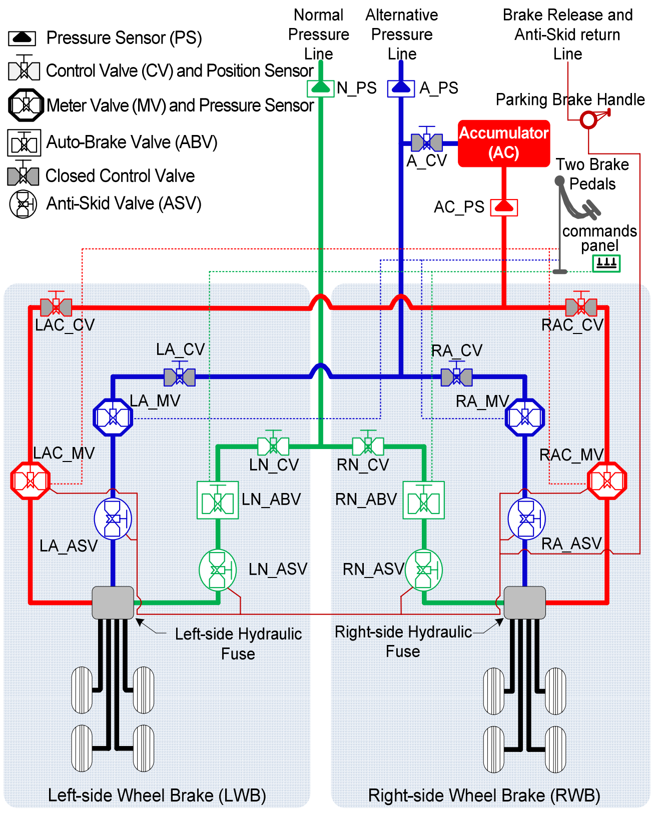


Fig. 9 Physical illustration of an aircraft brake system

During flying the hydraulic pressure at the lines is monitored by the BSCU. Once the pressure drops close to 1300 PSI, a warning light “BRAKE SOURCE” illuminates and the pilot is advised by a message of the Engine Indication and Crew Alerting System (EICAS) to switch to another brake line.

ABS has three modes: normal, alternative and accumulative modes, according to the operative line. Over the three modes, different functions are delivered. For example, the optional selection between the auto and manual brake is only possible during the normal mode. Table III abstracts the deliverable functions over the three mode of the ABS.

TABLE III
FUNCTIONS PROVIDED OVER EACH MODE OF THE AIRCRAFT BRAKE SYSTEM

Mode	Deliverable functions
Normal	1- Auto-Brake. 2- Manual-Brake. 3- Anti-skid. 4- Parking Brake.
Alternative	1- Manual-Brake. 2- Anti-skid. 3- Parking Brake.
Accumulative	1- Manual Brake. 2- Parking Brake.

According to Prometheus [19], Fig. 10 shows the multi-agent system that is deployed to monitor the aircraft brake system. Two agents monitor the two sub-systems and appear as LWB_MAG and RWB_MAG and one agent monitors the entire system, appearing as ABS_MAG. The multi-agent system is implemented by Jason interpreter, an extended version of AgentSpeak programming language [20].

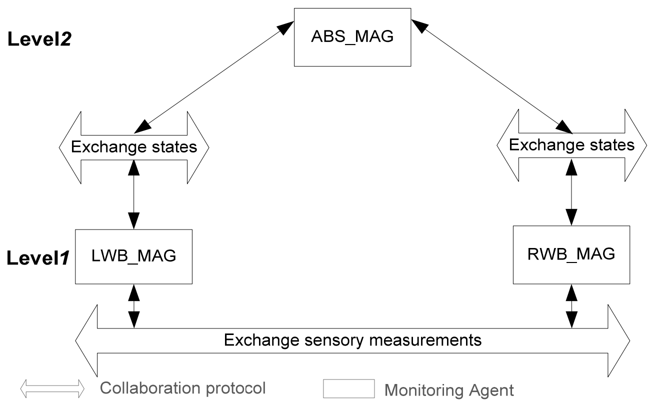


Fig. 10 Multi-agent system to monitor the aircraft brake system

Table IV shows the state-transition table of the LWB sub-system. Together Table I, Table II and Table IV in addition to Fig. 5 contribute to achieve the monitoring experiment and demonstrate the ability of the monitor to deliver the intended three safety tasks.

In the context of the experiment, failure of the normal line is simulated by injecting a fault of the anti-skid valve RN_ASV, such that the brake system transits to the alternative mode. The monitoring expression that verifies the occurrence of this event is as follows:

$$\mathrm{RN\_ASV\_OP} > \mathrm{RN\_ASV\_C} + 50$$
OR
$$\mathrm{RN\_ASV\_OP} < \mathrm{RN\_ASV\_C} - 50;$$

The expression is verified true when the pressure measured at the output of the anti-skid valve of the normal line (RN_ASV_OP) is less or greater than the pressure commanded by the BSCU to the anti-skid valve (RN_ASV_C). Plus and minus 50 PSI is added as possible bias of the sensors.

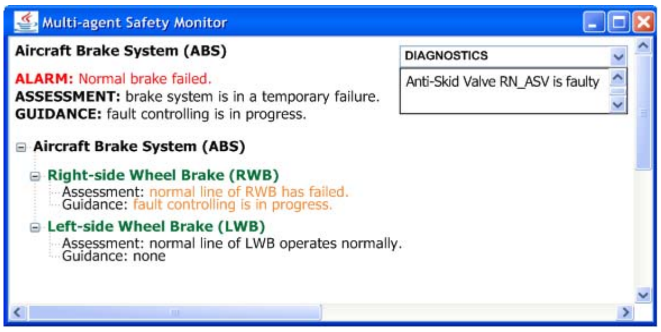
TABLE IV
STATE-TRANSITION TABLE OF THE LWB SUB-SYSTEM OF THE AIRCRAFT BRAKE SYSTEM

Current state	Conditions	Event	Alarm	Controlling	Diagnosis	New state
LWB_NM_EFS	**Assessment:** LWB operates normally. **Guidance:** none.	LN_ASV_OP > LN_ASV_C + 50 OR LN_ASV_OP < LN_ASV_C - 50;	normal brake failed	- LA_CV_C = 1; - LN_CV_C = 0;	needed	LWB_NM_FS
		ABS_NM_TDFS == true	none	- LA_CV_C = 1; - LN_CV_C = 0;	not_needed	LWB_AM_EFS
LWB_NM_FS	**Assessment:** normal line of LWB has failed. **Guidance:** fault controlling is in progress.	ABS_NM_TDFS == true	none	not_needed	not_needed	LWB_AM_EFS
LWB_AM_EFS	**Assessment:** LWB is pressured by the alternative line **Guidance:** apply manual brake.	LA_ASV_OP > LA_ASV_C+ 50 OR LA_ASV_OP < LA_ASV_C- 50;	alternative brake failed	- LAC_CV_C=1; - LA_CV_C = 0;	needed	LWB_AM_FS
		ABS_AM_TDFS == true	none	- LAC_CV_C=1; - LA_CV_C = 0;	not_needed	LWB_ACM_EFS
LWB_AM_FS	**Assessment:** the alternative line of the RWB has failed. **Guidance:** fault controlling is in progress.	ABS_AM_TDFS == true	none	not_needed	not_needed	LWB_ACM_EFS
LWB_ACM_EFS	**Assessment:** LWB is pressured by the accumulative line. **Guidance:** apply manual brake.	LAC_MV_OP > LAC_MV_C + 50 OR LAC_MV_OP < LAC_MV_C - 50;	Accumulative brake failed	impossible	needed	LWB_ACM_FS
LWB_ACM_FS	**Assessment:** LWB fails there is no brake on the right-side landing gear **Guidance:** no brake is available.	none	none	not_needed	not_needed	none

Once the occurrence of the above expression is verified, agent RWB_MAG perceives its state-transition table (Table II) and achieves the following procedure:

- From the relevant ALARM attribute, agent RWB_MAG quotes "normal brake failed" and alarms the pilot.
- From the relevant CONTROLLING attribute, agent RWB_MAG opens valve RA_CV and closes valve RN_CV, to switch to the alternative line.
- From the relevant DIAGNOSIS attribute, agent RWB_MAG verifies the need for a diagnostic process. At this point and before applying the corrective measures, agent RWB_MAG updates the symptoms of the relevant diagnostic model (Fig. 5) with the relevant measurements.
- From the relevant NEW STATE attribute, agent RWB_MAG transits to a new state which is the failure state RWB_NM_FS. From this state, the pilot is provided with assessment, "normal line of RWB has failed" and guidance, "fault controlling is in progress".
- Agent RWB_MAG communicates state RWB_NM_FS to the parent agent (ABS_MAG).

Since a diagnostic process is needed, then before launching a new monitoring cycle, agent RWB_MAG retrieves the position of the relevant diagnostic model (Fig. 5). For the purpose of diagnosis, agent RWB_MAG exploits a diagnostic algorithm that couples between blind-depth-first and heuristic traverses. Fig. 11 shows alarm information announced to the operator on the given conditions.



Aircraft Brake System (ABS)
ALARM: Normal brake failed.
ASSESSMENT: brake system is in a temporary failure.
GUIDANCE: fault controlling is in progress.
Anti-Skid Valve RN_ASV is faulty
Aircraft Brake System (ABS)
Right-side Wheel Brake (RWB)
Assessment: normal line of RWB has failed.
Guidance: fault controlling is in progress.
Left-side Wheel Brake (LWB)
Assessment: normal line of LWB operates normally.
Guidance: none

Fig. 11 Operator's Interface after the failure of the normal line of RWB

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

- As the current state is the ABS_NM_EFS, the received state results in verifying the occurrence of "LWB_NMFS == true OR RWB_NMFS == true".
- From the relevant NEW STATE attribute, agent ABS_MAG transits to a new state which is the temporary degraded/failure state ABS_NM_TDFS. From this state the pilot is provided with assessment, "brake system is in a temporary failure", and guidance, "fault controlling is in progress" (as shown in Fig. 11).
- Agent ABS_MAG communicates the new state to the children (RWB_MAG and LWB_MAG).

When agents RWB_MAG and LWB_MAG receive the messages, each achieves a certain procedure, as follows:

Agent RWB_MAG perceives the state-transition table (Table II) and achieves the following procedure:

- While the current state is the RWS_NM_FS, the received state results in verifying the occurrence of ABS_NM_TDFS == true.
- As the relevant ALARM, CONTROLLING and DIAGNOSIS attributes require no action then from the relevant NEW STATE attribute, agent RWB_MAG transits to a new error-free state RWB_AM_EFS. From this state the pilot is provided with assessment, "RWB is pressured by the alternative line", and guidance, "apply manual brake".
- Agent RWB_MAG communicates that state to the parent agent (ABS_MAG) and launches a monitoring cycle.

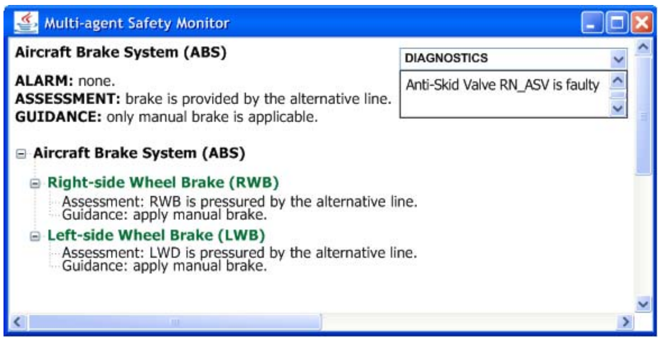
Agent LWB_MAG perceives its state-transition table (Table IV) and achieves the following procedure:

- While the current state is the LWS_NM_EFS, the received state results in verifying the occurrence of ABS_NM_TDFS == true.
- As the relevant ALARM attribute shows "none", no alarm is released.
- From the relevant CONTROLLING attribute, agent LWB_MAG opens valve LA_CV and closes valve LN_CV, to switch to the alternative line.
- As the relevant DIAGNOSIS attribute holds "not_needed", no action is taken.
- From the relevant NEW STATE attribute, agent LWB_MAG transits to a new state which is the error-free state LWB_AM_EFS. From this state the pilot is provided with assessment, "LWB is pressured by the alternative line" and guidance, "apply manual brake".
- Agent LWB_MAG communicates that state to the parent agent (ABS_MAG) and launches a monitoring cycle.

When agent ABS_MAG receives messages sent by agents LWB_MAG and RWB_MAG, it accordingly perceives its state-transition table (Table I) and achieves the following procedure:

- While the current state is the ABS_NM_TDFS, the received state results in verifying the occurrence of LWB_AM_EFS == true AND LWB_AM_EFS == true.
- From the relevant NEW STATE attribute, agent ABS_MAG transits to a new state which is the error-free state ABS_AM_EFS. From this state the pilot is provided with assessment, "LWB and RWB are pressured by the alternative line" and guidance, "only manual brake is applicable".

After achieving the above procedures, the alternative mode of the entire brake system would be launched without asking the pilot to switch to another model. That includes the automatic isolation of the normal line by closing the control valves LN_CV and RN_CV and activation of the alternative line by opening the controlling valves LA_CV and RA_CV. The pilot is updated with the new conditions (activation of the alternative line) as shown in Fig. 12. The interface informs the pilots about the current conditions and also advises them to apply manual brake as the only applicable option.



Aircraft Brake System (ABS)

ALARM: none.
ASSESSMENT: brake is provided by the alternative line.
GUIDANCE: only manual brake is applicable.

Anti-Skid Valve RN_ASV is faulty

Aircraft Brake System (ABS)

Right-side Wheel Brake (RWB)
Assessment: RWB is pressured by the alternative line.
Guidance: apply manual brake.

Left-side Wheel Brake (LWB)
Assessment: LWD is pressured by the alternative line.
Guidance: apply manual brake.

Fig. 12 Operator's Interface after controlling the failure of the normal line of RWB

V. Conclusion

This paper proposed an on-line safety monitor based on a multi-agent system and knowledge derived from design models and safety assessment model of the monitored system. Agents exploit that knowledge to deliver a range of safety tasks which have been briefly discussed in the context of a case study of an aircraft brake system. The monitor can detect symptoms of failure on process parameters as violations of simple constraints, or deviations from more complex relationships among process parameters, and then diagnose the causes of such failures. With appropriate 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, i.e. assessment and guidance. 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 a research issue remains to be investigated. The quality of the monitoring tasks and the correctness of the inferences drawn by the monitor depend mainly on the validity of sensory measurements. The validation of the sensory measurements is, therefore, an area for further research.

Acknowledgment

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

References

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

[2] 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*, vol. 10, no. 5, 2006, pp. 319-335.

[3] Y. S. Ng, and R. Srinivan, "Multi-agent based collaborative fault detection and identification in chemical processes," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 6, 2010, pp. 934-949.

[4] A. A. Mohamed and O. Basir, "An Adaptive Multi-Agent Approach for Distributed Alarm Correlation and Fault Identification," in *Proc. Parallel and Distributed Computing and Networks (PDCN 2010)*, Innsbruck, Austria, 2010.

[5] M. Rollo, P. Novak, J. Kubalik, and M. Pechoucek, "Alarm Root Cause Detection System," in Camarinha-Matos, L. M., editor, *Emerging Solutions for Future Manufacturing Systems*. New York: Springer, 2004, pp. 109-116.

[6] K. H Cho and J. T. Lim, "Multiagent Supervisory Control for Anti fault Propagation in Serial Production Systems,", *IEEE Transactions on Industrial Electronics*, vol. 48, no. 2, April, 2001, pp. 460-466.

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

[8] BEA, "Safety investigation into the accident on 1 June 2009 to the Airbus A330-203, flight AF447," [online], *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 August 2011].

[9] HSE, "Better alarm handling, HSE Information Sheet" [online]. *UK: Health and Safety Executive, chemical sheet No 6*, 2000. Available: http://www.hse.gov.uk/pubns/chis6.pdf, [Accessed 13th February 2011]

[10] Y. Papadopoulos, "Model-based system monitoring and diagnosis of failures using state-charts and fault trees," *Reliability Engineering and System Safety*, vol. 8, no. 3, 2003, pp. 325-341.

[11] Wallace C. J., Jajn G. J. and Mcarthur S. D. J., 2011. Multi-agent system for nuclear condition monitoring. In *Proc. 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)*, Taipei, Taiwan, 2nd of May 2011.

[12] A. F. 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.

[13] E. Mangina, "Intelligent agent-based monitoring platform for applications in engineering," *International Journal of Computer Science & Applications*, vol. 2 no. 1, pp. 38-48.

[14] A. Dheedan and Y. Papadopoulos, "Model-Based Distributed On-line Safety Monitoring," in *Proc. The Third International Conf. on Emerging Network Intelligence (EMERGING 2011)*, Lisbon, Portugal, November 20-25, 2011, pp. 1-7.

[15] H. P. Feiler, P. D. Gluch, J. J. Hudak, "The Architectural Analysis and Design Language (AADL): An Introduction," *USA: Software Engineering Institute and Carnegie Mellon University, CMU/SEI-2006-TN-001*, 2006. Available: http://i12www.ira.uka.de/~engelc/lehre/seminarSS07/material/aadlIntro.pdf [Accessed November 7th 2010].

[16] H. P. Feiler and A. Rugina, "Dependability Modelling with the Architecture Analysis and Design Language (AADL)" [online], *USA: Software Engineering Institute and Carnegie Mellon University, CMU/SEI-2007-TN-043*, 2007. Available: ftp://ftp.sei.cmu.edu/pub/documents/07.reports/07tn043.pdf [Accessed November 7th 2010].

[17] M. Breen. Experience of using a lightweight formal specification method for a commercial embedded system product line. *Requirements Engineering Journal*, vol. 10, no. 2, 2005, pp. 161–172.

[18] S. D. J. McArthur, E. M. Davidson, J. A. Hossack, J. R. McDonald, "Automating power system fault diagnosis through multi-agent system technology," in *Proc. 37th Annual Hawaii International Conference on System Sciences (HICSS'04)*, Big Island, Hawaii, 5-8 Jan 2004, pp. 1-4.

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

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