An embedded distributed tool for transportation systems health assessment

M. Dievart[1], P. Charbonnaud[1], X. Desforges[1]

1: Université de Toulouse, INPT, Ecole Nationale d'Ingénieurs de Tarbes, Laboratoire Génie de Production, 47 avenue d'Azereix, 65016 TARBES France

Abstract: This article presents an embedded distributed tool for health assessment of complex systems. The presented architecture is based on a solving method for embedded technical diagnostics and prognostics. This tool provides services enabling the evaluation of the health status of complex systems. Diagnostic services provide information for the maintenance decision support system that leads to reduce the periods of unavailability and determine if their future mission can be carried out. The diagnostic and prognostic functions are detailed and the exchanged data are specified. An example shows the feasibility of the proposed architecture and demonstrates the correctness of the developed algorithms.

Keywords: Distributed diagnosis, prognosis, health status, embedded systems, transportation systems.

1. Introduction

Transportation systems are the objects of new regulations in terms of environment, goods and people safety, and the needs of new services have consequences on the complexity of embedded systems. To face this increasing complexity, multiple functionalities of the resources are embedded and deployed into networks of functions achieved by Line Replaceable Units (LRU). Faulty LRUs are replaced when the vehicle is at its base and repaired in the maintenance workshops, while the repaired system carries on with its mission. The increasing number of functionalities of the embedded systems contributes to raise the possession and acquisition costs leading the resources customers to optimize their availability rate.

Bringing into operation the Condition-Based Maintenance (CBM) recommendations usually improve the equipment availability [1, 2]. Indeed, the CBM depends on the effectiveness of the system state provided by monitoring and diagnostic functions. They are carried out in particular from on line data that an embedded centralized diagnostic function generally process. However, in the case of system of systems also called complex system, the identification of the faulty components is difficult using centralized architectures. After the mission, the maintenance operators must collect information by interactions with the embedded diagnostic system, in order to isolate possible faulty LRUs, and to apply troubleshooting procedures. The drawbacks of such architectures are related to the numerous pieces of information to process, which might be wrong. The automated diagnostic processes combine these errors and lead to useless removals of LRUs. Those removals are costly and increase the risk of damaging the system.

The technical diagnosis of transportation systems provides to the maintenance operators a list of LRUs that should be replaced. When the diagnosis is done online, the maintenance operators prepare the intervention sooner, reducing the duration and the costs of maintenance actions. The main task of a diagnostic function is to deliver an advice on a set of faulty components and to determine the severity of the fault. A difficulty in diagnosing such complex systems is due to their numerous kinds of functions integrating different technologies (electronics, data processing, mechanics, hydraulics…). Thus, the implemented diagnostic techniques must be adapted to the knowledge available about the system. During its use, various faults may impact the resource. Those faults degrade its operating modes. Three types of faults are considered: *Cataleptic*, *Permanent* and *Fugitive*.

Different implementations of the monitoring and diagnostic functions associated to the LRUs exist. The LRU hosts its monitoring and diagnostic functions, the diagnostic function can also be hosted in another platform or, the monitoring and diagnostic functions can totally be distributed in different platforms.

In the case of aircrafts, the Centralized Maintenance System (CMS) provide a list of likely faulty LRUs for the maintenance operator. This list is established according to information provided by the built-in test equipments that collect monitoring information from the LRUs and generate tests if they are needed. CMS correlates data to provide a "pre-diagnosis" of the LRUs. The flight warning system provides to the cockpit crew information on aircraft failed functions [3, 4].

Complex systems can be considered as sets of systems that depend more or less on each other. A system implements one or several functions. For safety purposes, functions can be redundant as well as the LRUs implementing them. That explains why several models are necessary to classify the different operating modes of the LRUs and their health status.

In this paper, a decentralized/distributed health assessment is proposed for reducing the number of useless removals of LRUs and for providing information about the availability of the system. For applications to systems of systems, monitoring and diagnostic functions can be implemented closer to the LRUs thanks to the multi-agent approach. In the case of a distributed approach, a collaborative mechanism between diagnostic agents have to enable the convergence of the local diagnoses towards a set of accused LRUs which should ideally correspond to the true faulty ones. Furthermore, prognostic agents are in charge of evaluating the Remaining Useful Lifetime (RUL) of systems and functions of the complex system from the RUL of LRUs provided by a monitoring layer. The Health Status is computed from the results provided by the diagnostic and prognostic agents.

2. System of systems modeling

Generally, the system is analyzed from different models in order to obtain a satisfying representation for diagnosis purpose. This analysis enables to collect the available knowledge on the complex systems. Many of them are helpful to design their diagnostic functions. In the literature [5-8], these models can be functional, structural, behavioral and teleological. They enable to model the behavior of components, of functions and of their interactions according to normal or degraded modes. In other studies, models are added to evaluate the diagnosis confidence [9].

A Complex System (*CS*) can be defined by a finite set of *m* system $\sum_i$. $CS = \{\sum_1, \sum_2,.., \sum_m\}$. A system $\sum_i$ can be defined as a set of *n* function $F_{i,j}$. $\sum_i = \{F_{i,1}, F_{i,2},..,F_{i,n}\}$. A function $F_{i,j}$ can be defined as a set of *k* LRUs implementing this function. $F_{i,j} = \{LRU_{i,j,1}, \ldots, LRU_{i,j,k}\}$. If a LRU contribute to the implementation of more than one function, this one should be duplicated as many times as it contributes to the implementation of function. If an LRU contributes to the implementation of 3 functions, it must exist three LRUs of that kind when defining the system. The fact that these LRUs are equivalent must be declared in the knowledge base. After defining the hierarchical decomposition of the system, a system modeling has to be formalized for ensuring system diagnostic.

In this paper, the necessary System Knowledge to diagnose the complex system are collected in the set *SK* made of four types of knowledge: functional, structural, behavioral and topological.

The Functional Description (*FD*) is the set of functions ensured by every system. *FD* represents links between LRUs, functions and system.

The Structural Description (*SD*) is dedicated to the identification of the set of LRUs and of physical connections between them. *SD* introduce predicate *CONNECT(X,Y)* that means that X is connected to Y. $S : CONNECT(LRU_{i,j,q}, LRU_{p,r,s})$ with *q* and *s* respectively one of the LRUs implementing $F_{i,j}$ and $F_{p,r}$.

The behavioral models are used in order to identify the relevant indicators that are used to generate symptoms for the various faults that may affect the LRUs. They help to classify the faults of the LRUs from their symptoms. For diagnostic purpose, the knowledge *BM* provides the relationships between the symptoms, the LRUs and their faults.

The Topological Dependencies (*TD*) determines the proximities of components that may be the origin of indirect failures or faults of a LRU due to the proximity of another failed one. *TD* introduce predicate *TOPO(X,Y)* that means that *X* is close to *Y* and that some faults of X may affect the functioning of Y. *TD*: $TOPO(LRU_{i,j,q}, LRU_{p,r,s})$ with q and s respectively one of the LRUs implementing $F_{i,j}$ and $F_{p,r}$.

Finally, $SK = SD \cup FD \cup TD \cup BM$

3. Embedded health assessment

The OSA-CBM project [10] defines a Health Assessment (HA) layer whose primary function determines the health status of a monitored system, subsystem, equipment or component in terms of fault, failure, availability. The health assessment module should take into account diagnoses, trends in the health history, operational status and loading, and the maintenance history. The HA layer provides the Health Status (HS) of a monitored entity.

Prognostics and Health Management (PHM) is defined in [11] as the phase involved with predicting future behavior, including RUL, in terms of current operating state and the scheduling of required maintenance actions. According to [12], diagnosis and prognosis are processes of assessment of a system's health. Diagnosis is an assessment about the current and past health of a system based on observed symptoms, and prognosis is an assessment of the future health.

The different results given by diagnostic and prognostic functions must be considered as a decision support to operate appropriately on the system. In the proposed approach, the HA is introduced as a combination of the results of the diagnostic and prognostic functions.

Ideally, the diagnosis identifies a set Δ_2 of failed function and locates their causes, *i.e.* a set Δ_1 of faulty LRUs from a set of symptoms *S* and a set of tests *T*. This leads to the next relationship where *Diag* is the diagnostic function:

$$(\Delta_1, \Delta_2) = Diag(SK, S, T)$$

The function *Diag* can be implemented thanks to two sub-functions: *Diag1* and *Diag2*. The function *Diag1* allows, starting from a set of symptoms and a set of tests, to identify the set of faulty LRUs of the system $\Delta_1 = Diag1(SK, S, T)$, where $\Delta_1 = \{AB(LRU_{i,j,k}),\ldots,AB(LRU_{p,r,s})\}$ and the function *Diag2* enables to locate the set of failed function from tests and the set of failed LRUs: $\Delta_2 = Diag2(SK, \Delta_1, T)$ where $\Delta_2 = \{AB(F_{i,j}),\ldots,AB(F_{p,r})\}$. AB(.) denotes either a faulty LRU or a failed function.

The prognostic function, that completes the diagnostic function, has been the subject of many studies since the end of the 90s. Prognostic allows determining the RUL of a component or a system [13]. In [14], the prognostic is defines as the capability to provide early detection of the precursor and/or incipient fault condition (very “small” fault) of a component, and to have the technology and means to manage and predict the progression of this fault condition to component failure. A global point of view of the studies realized on prognostic is proposed in [15]. The RUL can either be determined by indicators extrapolation or from failure probability of the component of the system like the Mean Time To Failure (MTTF) or the Mean Time Between Failures (MTBF). The RUL is an assessment of the remaining operational time until the system becomes unable to complete its mission successfully according to the conditions of operation. This means that the prognostic function requires a historic of the current and future mission with their constraints. This knowledge allows determining thresholds beyond which the LRU cannot carry out the mission in nominal mode. In short term, the data about the current mission are used online and in long-term, the data about the future mission are used when the system is in the maintenance area. Of course, this threshold is not useful when determining the likelihood of failure. The RUL of an LRU is ensured by the monitoring layer. Let us note $RUL(LRU_{i,j,k})$ the RUL of the $LRU_{i,j,k}$, $TBAB(LRU_{i,j,k})$ the Time Before Abnormal Behavior of the $LRU_{i,j,k}$. and $TBAB(F_{i,j})$ the TBAB of the function $F_{i,j}$, As it exists dependencies between LRUs and RUL are the data generated by the monitoring, TBAB is introduced and can be defined by : $TBAB(LRU_{i,j,k}) = Min(RUL(LRU_{i,j,k}), TBAB(LRU_{i,j,k}), TBAB_{struc}(LRUs))$

where $TBAB_{struc}(LRUs)$ is the set of TBAB of upstream structurally dependent LRU of the $LRU_{i,j,k}$.

Therefore, for a given function $F_{i,j}$ without redundant LRUs, $RUL(F_{i,j}) = Min(TBAB(LRU_{i,j,1}), \ldots, TBAB(LRU_{i,j,k}))$. Let us considered now the case of a function $F_{i,j}$ implemented by *A* redundant LRUs. And this function is not considered as failed if at least *B* of *A* LRUs are not failed. In this case, $TBAB(F_{i,j})$ is equal to the B^{th} maximum RUL among the *A* TBAB of each LRUs implementing $F_{i,j}$. For example, if a function $F_{i,j}$ is ensured by $LRU_{i,j,1}$, $LRU_{i,j,2}$ and $LRU_{i,j,3}$, and the function is not considered as failed if at least 2 of this 3 LRUs are not failed. Given $RUL(LRU_{i,j,1}) = 5UT$, $RUL(LRU_{i,j,2}) = 10UT$ and $RUL(LRU_{i,j,3}) = 15UT$, $TBAB(F_{i,j}) = 10$ UT.

Ideally, the prognosis identifies the set $\prod_2$ of the RUL of each function implementing the system of systems and the set $\prod_1$ of RUL of each LRU from a set of extrapolated data *ED* and a set of threshold *Th*. This leads to the next relationship where *Prog* is the prognostic function:

$$(\prod_1, \prod_2) = Prog(SK, ED, Th)$$

The function *Prog* can be implemented thanks to two sub-functions. The function *Prog1* allows, starting from a set of extrapolated data and a set of threshold, to identify the set of TBAB of the LRUs of a function $\prod_1 = Prog1(SK, RUL, TBAB)$, where $\prod_1 = \{TBAB(LRU_{i,j,k}),\ldots, TBAB(LRU_{p,r,s})\}$ and the function *Prog2* enables to locate the set of failed function from tests and the set of failed LRUs: $\prod_2 = Prog2(SK, \prod_1)$ where $\prod_2 = \{TBAB(F_{i,j}),\ldots, TBAB(F_{p,r})\}$. TBAB (.) denotes either the TBAB of a LRU or the TBAB of a function.

In this paper, the health management of the system is ensured by a diagnostic function and a prognostic function. Data produced by these functions, list of RULs, faulty components and failed functions, are differently interpreted according to the current end user of the system. This implies the use of different kinds of histories and knowledge, including safety limits of the system defined by the operator.

Considering *HS(X)* the Health Status of *X*, the health status of an LRU can be defined by:

$$\begin{aligned} &HS(LRU_{i,j,k}) = \\ &= (TBAB(LRU_{i,j,k}) \wedge \neg AB(LRU_{i,j,k})) \\ &\vee (TBAB(LRU_{i,j,k}) \wedge AB(LRU_{i,j,k})) \end{aligned}$$

Considering $\varDelta_1^{F_{i,j}}$ the list of failed LRUs implementing the function $F_{i,j}$ and $\varPi_1^{F_{i,j}}$ the list of the RUL of LRUs implementing $F_{i,j}$ the health status of $F_{i,j}$ can be defined as: $HS(F_{i,j}) = (\varDelta_1^{F_{i,j}} \wedge \varPi_1^{F_{i,j}})$. The health status of the complex system is also defined as: $HS(SC) = (\varDelta_2 \wedge \varPi_2)$ and can be described as a use case diagram available in figure 1.

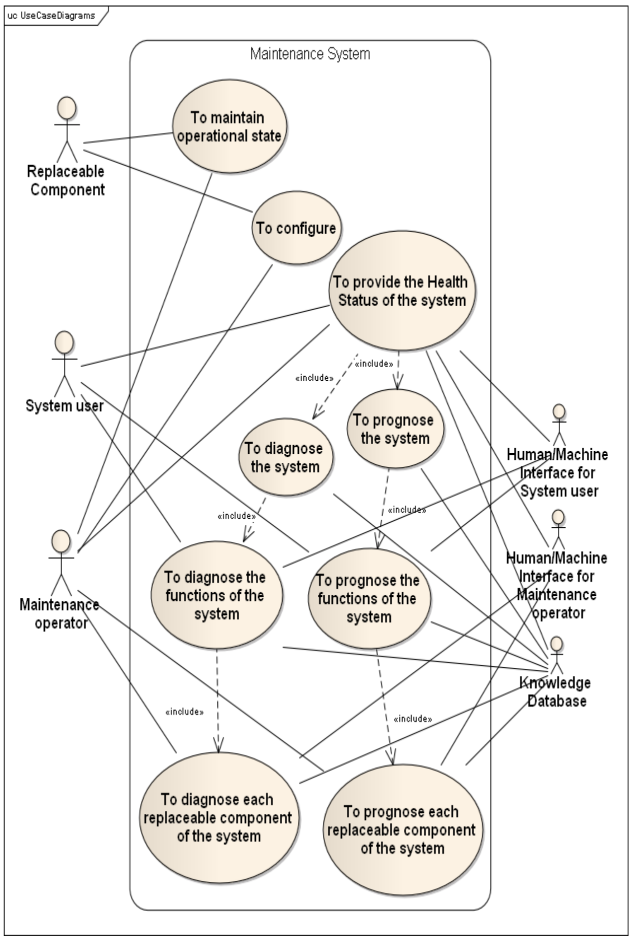
uc UseCaseDiagrams
Maintenance System
Replaceable Component
System user
Maintenance operator
Human/Machine Interface for System user
Human/Machine Interface for Maintenance operator
Knowledge Database
To maintain operational state
To configure
To provide the Health Status of the system
To diagnose the system
To prognose the system
To diagnose the functions of the system
To prognose the functions of the system
To diagnose each replaceable component of the system
To prognose each replaceable component of the system
«include»

Figure 1: Use case diagram of a maintenance system

4. A distributed architecture for health assessment

Distributed approaches of Information and Communication Technologies often provide satisfying solutions to face complexity. The diagnostic function was implemented in a distributed structure according to the multi-agent system concept. The agents of the structure cooperate and exchange data whatever the language used to model the information they contain is. This implementation requires data and models that have been collected and organized. In the case of complex systems, some works [16-18] show the feasibility to implement an embedded distributed diagnostic function with or without cooperation between its elements.

In figure 2, the architecture presented is based on a distributed implementation. Local diagnostic agents cooperate to diagnose the system. A middleware makes it possible to implement the services provided by the agents. The architecture consists of several LRUs gathered into several functions denoted LRU layer. Each LRU is observed by a monitoring function. The monitoring layer represents the monitoring functions designed by the suppliers. The monitoring functions send their symptoms to Diagnostic Agents (DAs). Using the cause-symptom relations, each DA determines the set of faulty LRUs among the LRUs that implement a function. One or more databases (KB) contain the set *SK* that supports the activities of the DAs. A Human/Machine Interface (HMI) displays the failed functions of the system for the production operators and the LRUs to replace or to fix for the maintenance operators. If the collaboration is correct the global diagnosis of the system in terms of faulty LRUs is the union of local diagnoses.

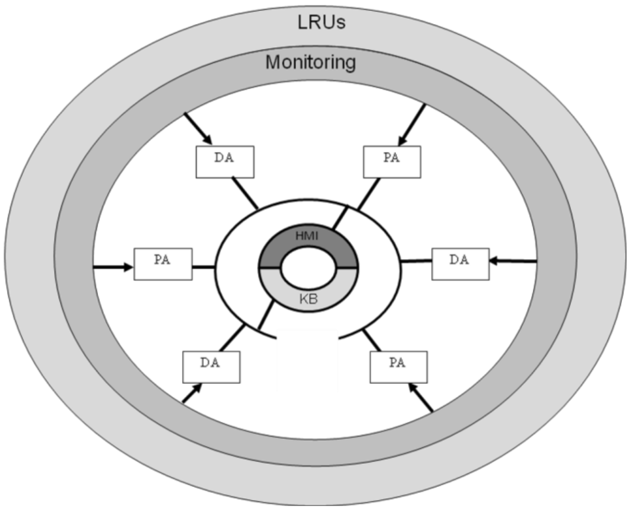

Figure 2: Distributed approach of Health Assessment

Several types of data are considered: a “symptom” is generated by the monitoring layer, a “faulty LRU message” is generated by the function “To propagate symptom” of a DA. A “known symptom” is has at least one cause identified by an analysis done at the system design stage and stored in KB. An “unknown symptom” has no cause identified in KB.

The activities of the diagnostic function are carried out by cooperation between the DAs. The reception of a symptom by a DA starts the process. The corresponding DA begins by inserting the received symptom in the chronologic list according to the date the monitoring layer. Then, the DA checks the type of symptom received. If the symptom is known, the database returns the cause of failure of the LRU and the DA declares it as known, otherwise, the cause of failure of the LRU is declared as unknown. It also manages the list of symptoms, chronologically. Δ_1 and Δ_2 are updated according to the type of the symptom and the answers of the database dealing with a status of the symptom (known or unknown) in the case of a failure symptom. Then, the DA searches structural dependencies with the failed LRU thanks to KB. If there is at least one dependent LRU, "propagation symptoms" are sent to the DAs that diagnose those LRUs which may not operate correctly and are declared "out of order". During this stage, the DA updates Δ_1 and Δ_2 according to the chronological list of symptoms. The status and the timestamp of the LRU are updated and data are recorded and displayed. Supervision of the DAs is ensured during "the fault propagation" function. If a DA did not confirm that it receives the message during the propagation task, the diagnostic process declares the agent as failed.

The evolution of the status of a LRU is shown in figure 3. An LRU is declared as "OK" at the beginning of the process. If an "unknown symptom" is received, the status switches to "unknown failure" (see (1) in the figure). If a symptom is received and this symptom (or set of symptom) is known, the status of the LRU is defined as "known failure" (see (2) in the figure). The reception of a Faulty LRU message only allows switching the status of the LRU from "OK" to "Out of Order" (see (3) in the figure). Four different values describe the state of a LRU:

- "OK" when the LRU is not faulty,
- "UF" when the LRU is faulty but the cause is unknown (Unknown Failure),
- "KF" when the cause of its failure is known (Known Failure),
- "OO" when the LRU does not work in nominal mode or is failed because of the failure of another LRU (Out of Order).

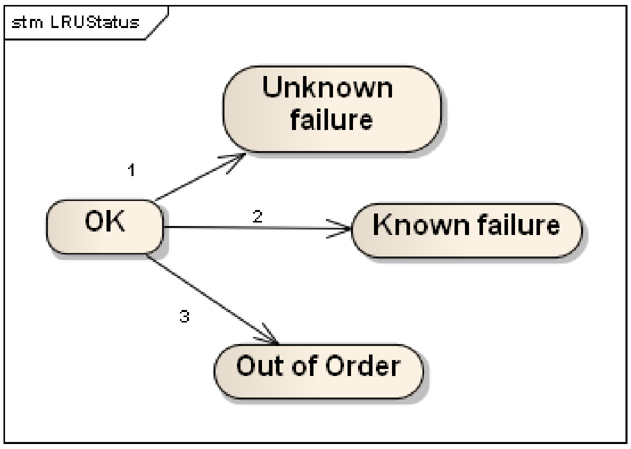

Figure 3: Status of a LRU

In our case, the status of an LRU evolved differently. The status of the LRU is estimated by the DA that received data whose causality is not respected (according to apparition timestamp). As it is possible to challenge the order of reception of the data (symptoms and Faulty_LRU_Messages), the evolution of the status of a LRU is different as shown in figure 4.

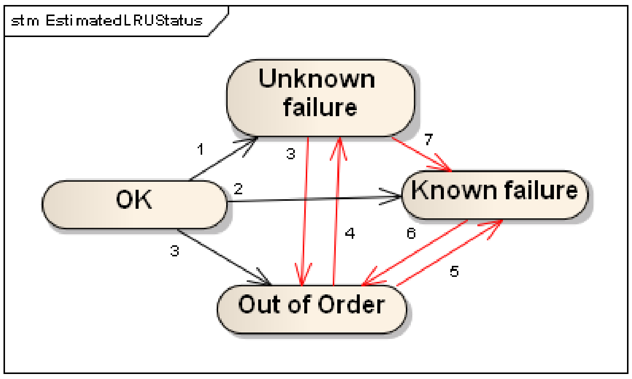

Figure 4: Estimated LRU status by DA

The red arrows denote that the causality was not respected. Arrows number 4 (in the case of an unknown failure) and 5 (in the case of a known failure) in figure 4 mean that a faulty LRU message has been diagnosed before a symptom. Arrows numbers 3 and 6 in figure 4 mean that a symptom has been diagnosed before a faulty LRU message. The arrow number 7 denotes that a symptom or a set of symptoms was firstly declared has unknown but, because of the symptom(s) received afterward, it becomes a known failed situation. When a LRU is subjected to a known or an unknown failure, the known failure is privileged in the results. Whatever the kind of failure is known or unknown, the LRU must be removed.

Therefore, there are two different ways to process the diagnosis depending on data received by the DA (symptom or Faulty_LRU_Message). Whatever the data (symptom or Faulty_LRU_message) the diagnostic function has to deal with, the results are recorded in the blackboard of the DA. Every time the DA receives a symptom or a Faulty_LRU_message, the old diagnostic result is duplicated and one copy is updated according to the data received. The blackboard of a DA contains the evolution of the status of the LRUs and of its implemented function and the data (symptom or Faulty_LRU_message). From this historic, it is possible to track the evolution of the status of each LRUs and each function of the system according to received data. The algorithm of the function ***ToDiagnose****(Symptom)* describes the behavior of the diagnostic function when a DA receives a symptom sent by the Monitoring layer.

Algorithm ***ToDiagnose****(Symptom)*

Return the results in the blackboard of the DA

```
1. Begin
2. Verify the symptom chronology
3. Define L the list of data ever diagnosed that appeared after symptom received
4. Query the database for knowing if the failure is identified
5. Change the status of the LRU incriminated by the symptom according to database answer and LRU current status (OK to UF, OK to KF OR UF to KF)
6. While L is not empty
7.     ToDiagnose(L.data)
8. EndWhile
9. Query the database for structural dependencies
10. While it exists dependencies
11.     Send(Faulty_LRU_Message) to DA
12. EndWhile
13. End
```

The algorithm of the function ***ToDiagnose****(Faulty_LRU_Message)* describes the behavior of the diagnostic function when receiving a Faulty_LRU_Message sent by another DA.

Algorithm ***ToDiagnose****(Faulty_LRU_Message)*

Return the results in the blackboard of the DA

```
1. Begin
2. Verify the Faulty_LRU_Message chronology
3. Define L the list of data ever diagnosed that appeared after Faulty LRU Message received
4. Change the status of the LRU incriminated by the Faulty LRU Message (OK to OO)
5. While L is not empty
6.     ToDiagnose(L.data)
7. EndWhile
8. Query the database for structural dependencies
9. While it exists dependencies
10.     Send(Faulty_LRU_Message) to DA
11. EndWhile
12. End
```

The activities chain involved when a symptom is generated is showed in figure 5

.

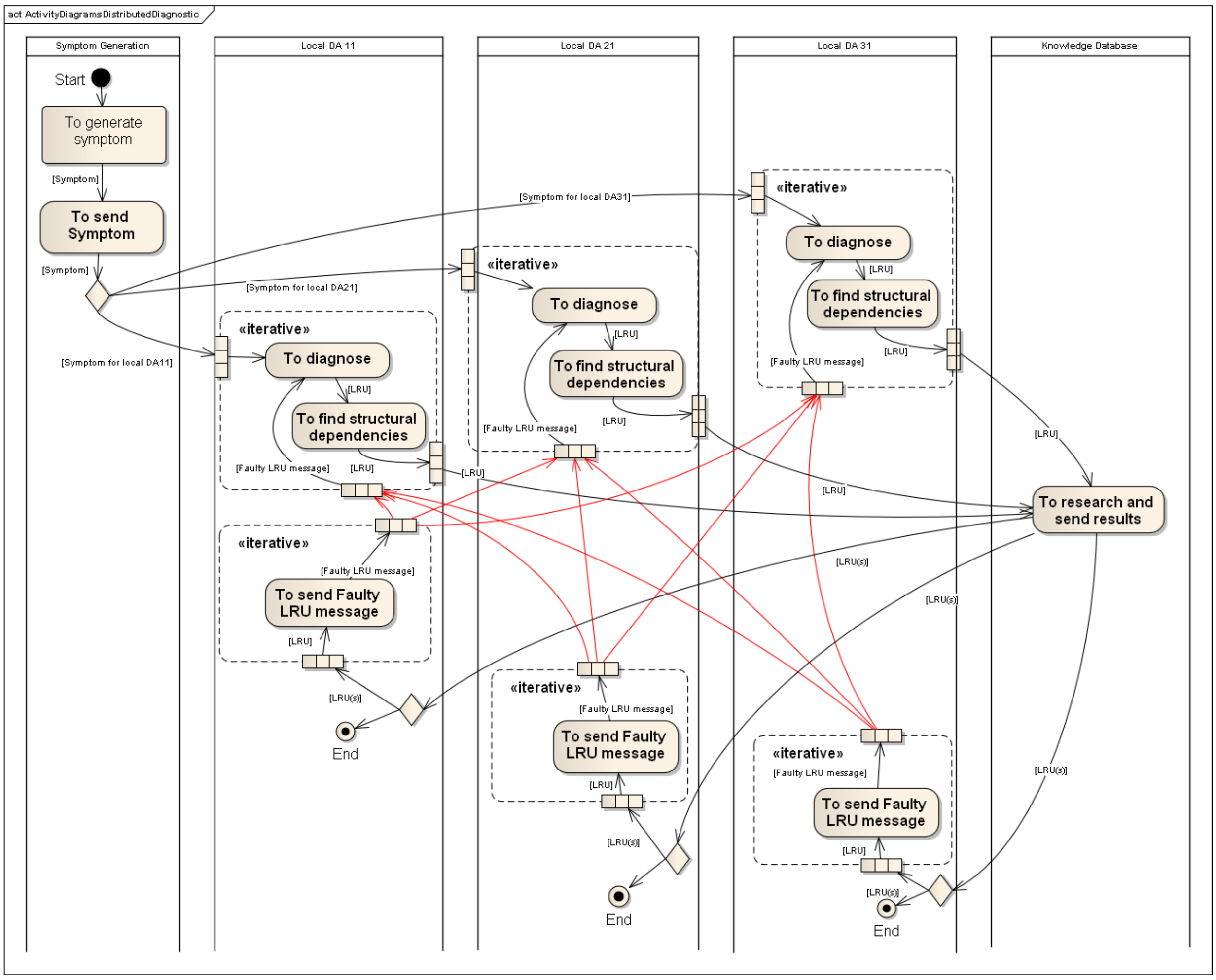

Figure 5: Activity Diagram of the distributed diagnosis system

The ***Send*** function sends Faulty_LRU_Message to the DA in charge of diagnosing the LRU incriminated by the Faulty_LRU_Message and launch the algorithm "***ToDiagnose****(Faulty_LRU_Message)*" of this DA.

Aiming at prognosing the LRUs and the functions, the monitoring layer sends RUL of each LRU periodically to the corresponding Prognostic Agents (PAs). Each PA is prognosing one function of the system and uses KB. Figure 6 shows the activity diagram of a PA. The reception of a RUL or a TBAB_message by a PA starts its process. A TBAB_message is a message send by a PA to a PA. A TBAB_message is composed of the number of the upstream LRU monitored by the PA that sends the TBAB_message, the calculated TBAB of this LRU and the number of the LRU the upstream LRU is structurally connected to. Finally, the TBAB_message is send to the PA that computes the downstream LRU. The involved PA updates the Time Before Abnormal Behavior (TBAB) of this LRU in the list it manages for each LRU implementing the function it prognoses. When a PA receives for the first time a TBAB_message from another PA that prognoses an upstream LRU, it records the TBAB_message. This record is updated every time it receives another TBAB_message from the same PA that prognoses the same upstream LRU. The PA defines the TBAB of the corresponding function. Then, it sends the TBAB (encapsulated in TBAB_Message) of this LRU to the PAs prognosing functions carried out by LRUs depending on the LRU for which the TBAB has been received. These PAs may then update the TBAB of their prognosed functions. As the RUL is defined to be the data that define remaining useful lifetime of a LRU processed by the monitoring layer, TBAB is defined to denote the same concept but take into account the structural dependencies of the LRU. It is to note that TBAB may be equal to RUL if the RUL sends by the monitoring is minor than the TBAB of upstream LRUs among the structural dependencies.

Therefore, there are two different ways to process the prognosis depending on data received by the PA (RUL or TBAB_message). Whatever the data (RUL or TBAB_message) the prognostic function has to calculate, the results are recorded in the blackboard of the PA. Every time the PA receives a RUL or a TBAB_message, the old prognostic result is duplicated and one copy is updated according to the data received. The blackboard of a PA contains the evolution of the TBAB of the LRUs and its function implemented and the data (RUL or TBAB_message). For each TBAB of the LRUs, the data (RUL or TBAB_message) from which the TBAB has been computed are recorded. From this historic, it is possible to track the evolution of the TBAB of each LRUs and each function of the system according to received messages.

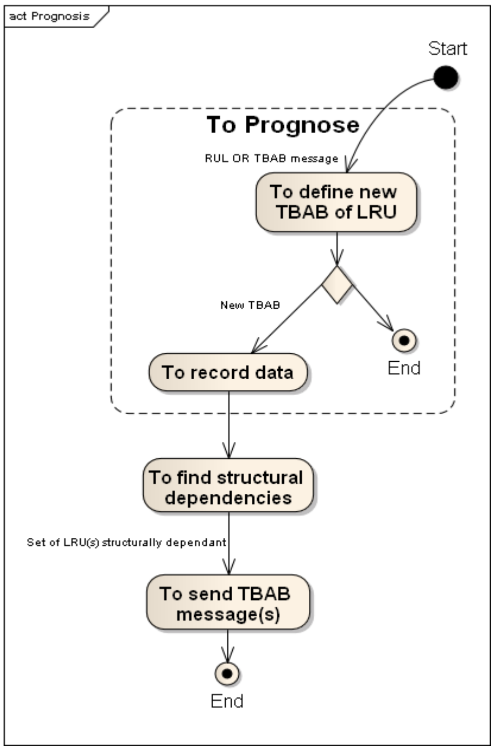

Figure 6: Activity Diagram of PA

The algorithm of the function ***ToPrognose****(RUL)* describes the behavior of the prognostic function when a PA receives a RUL sent by the Monitoring layer.

Algorithm ***ToPrognose****(RUL)*

Return the results in the blackboard of the PA

```
1. Begin
2. Update RUL value of the LRU
3. Compute the TBAB of the LRU
4. If TBAB of the LRU change
5.     Query the database for structural dependencies
6.     While it exists dependencies
7.         Send(TBAB_Message) to PA
8.     EndWhile
9. EndIf
10. End
```

The ***Send*** function sends TBAB_Message to the PA in charge of prognosing the LRU incriminated by the TBAB_Message and call the function ***ToPrognose****(TBAB_Message)* of this PA.

The algorithm of the function ***ToPrognose****(TBAB_Message)* describes the behavior of the prognostic function when receiving a TBAB_Message sent by another PA.

Algorithm ***ToPrognose***(*TBAB_Message)*

Return the results in the blackboard of the PA

```
1. Begin
2. If the LRU incriminated by the TBAB_Message
   have not ever received a TBAB_message with the
   number of this upstream LRU
3.       Create TBAB_Message record for the
         incriminated LRU
4. Else
5.       Update TBAB_Message record for the
         incriminated LRU
6. EndIf
7. Compute the TBAB of the LRU
8. If TBAB of the LRU change
9.       Query the database for structural
         dependencies
10.      While it exists dependencies
11.            Send(TBAB_Message) to PA
12.      EndWhile
13. EndIf
14: End
```

The sequence of activities involved when a RUL is generated is similar to the one involved when a symptom is generated (see figure 5). For the prognosis, RULs are generated by the monitoring layer to PAs and TBAB_Message are exchanged between PAs.

Both functions (To diagnose and to prognose) are activated when the HA system is started, but the diagnostic and prognostic functions are only processed when they receive a message, otherwise, they stand by. Diagnostic and prognostic sessions are finished when all the DAs and PAs are in sleeping mode.

5. Example of implementation

An example of implementation of the distributed HA function onto a system made of 2 functions both ensured by 3 LRUs is given in figure 7. This structure leads to the HA function to be implemented by 2 DAs and 2 PAs. DAs and PAs receive data generated by the monitoring layer respectively symptoms (full arrows between monitoring and health assessment layers) and RULs (dotted arrows between monitoring and health assessment layers). DAs communicate between them and KB (bold dotted arrows inside the health assessment layer). PAs do it between them and KB (bold arrows inside the health assessment layer) too. In fact, the communication between Agent of the same nature (DA or PA) is processed according to the structural dependencies between LRUs and the description of diagnostic and prognostic functions. DAs communicate with each other thanks to Faulty_LRU_Message and PAs thanks to TBAB_Message.

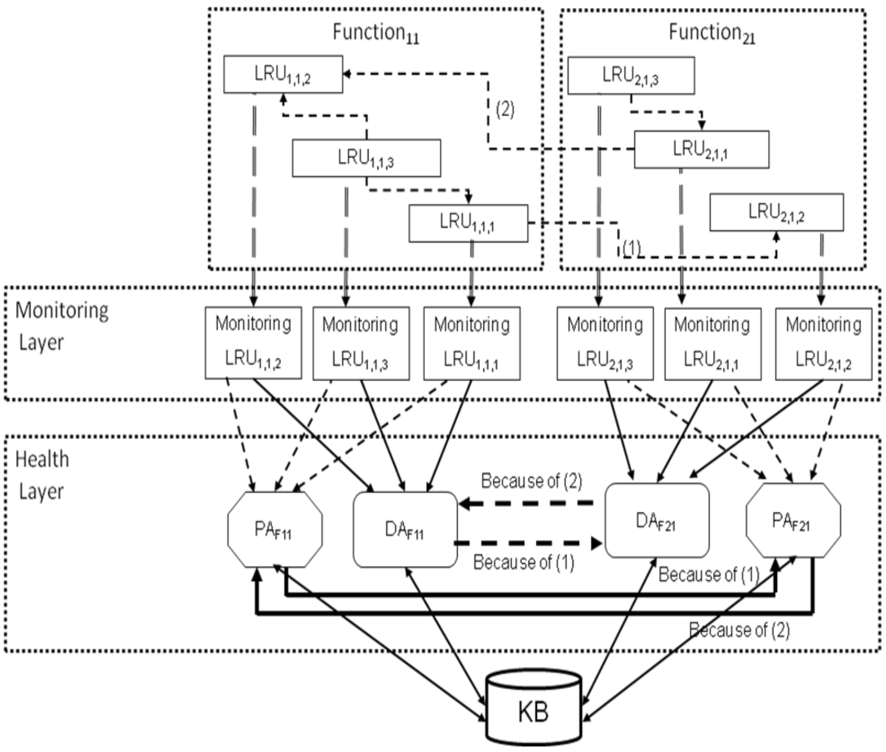

Figure 7. An example of implementation of a distributed health assessment function

Each time a DA receives a symptom, it copies *Δ1* and *Δ2* in its own blackboard and updates them with the result computed from the new symptom. This carries out the tracking of the evolution of the diagnostic process and this enables the diagnostic process to be non-monotone. If the DA receives a "propagation symptom", this one is used to refine the diagnosis. To explain this activity, let us considered the example with $LRU_{1,1,1}$, $LRU_{1,1,2}$ and $LRU_{1,1,3}$ that are structurally dependant. *SD* (the only considered knowledge for this example) contains knowledge about the system:

CONNECT($LRU_{1,1,1}, LRU_{1,1,2}$),

CONNECT($LRU_{1,1,2}, LRU_{1,1,3}$). Each LRU has its proper monitoring periods for its symptoms. For example, if we consider one symptom per LRU, we suppose that the symptom emitted from the monitoring of $LRU_{1,1,1}$ can be send to the DA every 5 minutes, the one of $LRU_{1,1,2}$ every minute and the one of $LRU_{1,1,3}$ every second. Let us consider that only $LRU_{1,1,1}$ failed, $LRU_{1,1,2}$ and $LRU_{1,1,3}$ should be defined as "out of order". DAs receive the symptoms when it is detected by the monitoring layer. Firstly, the DA receives $S_{1,1,3,1}$. So, $LRU_{1,1,3}$ is declared has failed. Then, the DA receives $S_{1,1,2,1}$. Because of the structural dependencies, $LRU_{1,1,2}$ is declared failed and $LRU_{1,1,3}$ is updated to an "out of order" status. Finally, the DA receives $S_{1,1,1,1}$. Because of the structural dependencies, $LRU_{1,1,1}$ is declared "failed" and $LRU_{1,1,2}$ and $LRU_{1,1,3}$ are declared as "out of order". The evolution of the status of these 3 LRUs according to the order of reception of symptoms is described in the chronogram in figure 8.

The state of the LRU changes and the timestamps of its changes are updated. At the end of the HA process, a list of faulty LRUs with their causes, their dated changes of status and a list of failed functions are available. The causes of faulty LRUs are described in terms of sentences from FMEA studies or of faulty LRUs for "OO" LRUs. The causes of failed functions are described in terms of faulty LRUs.

Considering the example as described in figure 7 with this failure: $LRU_{2,1,1}$ failed and the cause of the failure of $LRU_{2,1,1}$ is known to be a power failure. After symptom generation, the diagnosis result is given by:

Δ_1 = $LRU_{2,1,1}$ (status: KF, cause: power failure, timestamp: 2009/07/05 15h26min56s) & $LRU_{1,1,2}$ (status: OO, cause $LRU_{2,1,1}$ failure, timestamp: 2009/07/05 15h26min57s)

Δ_2 = $Function_{11}$ (status: OO, cause: $function_{12}$ failure, timestamp: 2009/07/05 15h26min58s) & $Function_{21}$ (status: KF, cause: $LRU_{2,1,1}$ failure, timestamp: 2009/07/05 15h26min57s)

In each set Δ_1 and Δ_2 the LRUs or functions that do not work in a nominal mode are listed with their current states, the causes of their failures and the timestamps of their changes of state.

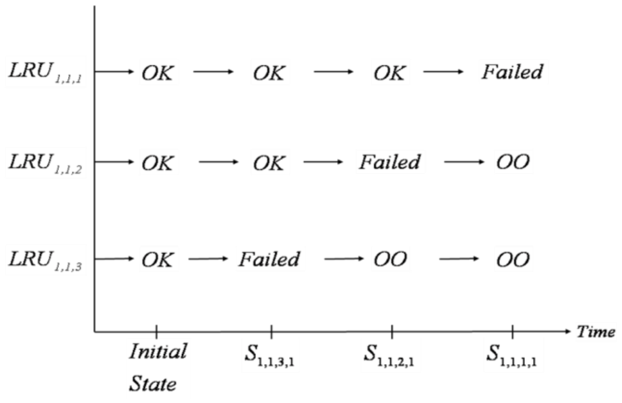

Figure 8: Chronogram of the evolution of the status of LRUs

Considering the same example as described in figure 7 with these RUL values at the beginning of the prognosis process:

RUL ($LRU_{1,1,1}$) = 10 TU (Time Unit), RUL ($LRU_{1,1,2}$) = 90 TU, RUL ($LRU_{1,1,3}$) = 30 TU, RUL ($LRU_{2,1,1}$) = 20 TU, RUL ($LRU_{2,1,2}$) = 40 TU, RUL ($LRU_{2,1,3}$) = 50 TU.

After prognosis process, the prognosis is given by:

Π_1 = RUL($LRU_{1,1,1}$) = 10 TU (Time Unit), RUL ($LRU_{1,1,2}$) = 90 TU, RUL ($LRU_{1,1,3}$) = 30 TU, RUL ($LRU_{2,1,1}$) = 20 TU, RUL ($LRU_{2,1,2}$) = 40 TU, RUL ($LRU_{2,1,3}$) = 50 TU.

Π_2 = RUL ($F_{1,1}$) = 10 TU, RUL ($F_{2,1}$) = 20 TU.

6. Conclusion

The distributed diagnostic and prognostic functions, presented in this paper, implement a distributed health assessment for systems of systems. The health assessment is here based on a monitoring layer that provides symptoms and estimations of the remaining useful lives of the components of the system.

The proposed diagnosis is a non-monotonic process that permits to take into accounts the events detected by the monitoring layer and time stamped before their asynchronous transmissions.

All the pieces of diagnostic and prognostic data and their timestamps are recorded to enable performance evaluation at the end of the health assessment session. Performance indicators (speed of convergence, data flow, and computational load...) may therefore be evaluated from the timestamps of the different data that are exchanged between the agents or stored. Further works will deal with performance evaluation of the proposed structure compared to centralized structures.

7. Acknowledgement

The project was supported by Airbus France.

8. References

[1] Jardine, A., Lin, D. and Banjevic, D.: *"A review on machinery diagnostics and prognostics implementing condition-based maintenance"*, Mechanical Systems and Signal Processing, vol 20, pp. 1483-1510, 2006.

[2] Scarf, P.: *"A Framework for Condition Monitoring and Condition Based Maintenance",* Quality Technology & Quantitative Management, vol 4, pp. 301-312, 2007.

[3] Ramohalli, G.: *"The honeywell on-board diagnostic and maintenance system for the boeing 777",* Digital avionics systems conference, 11th IEEE/AIAADigital Avionics Systems Conference, pp. 485-490, 1992.

[4] Byington, C., Kalgren, P., Johns, R. and Beers, R.: *"Embedded diagnostic/prognostic reasoning and information continuity for improved avionics maintenance",* AUTOTESTCON. IEEE Systems Readiness Technology Conference. Proceedings, pp. 320-329, 2003.

[5] Abu-Hanna, A., Benjamins, R., and Jansweijer, W.: *"Device understanding and modeling for diagnosis"*, *IEEE Intelligent Systems and Their Applications* Vol. 6(2), pp. 26-32, 1991.

[6] Chittaro, L. and Ranon, R.: *"Hierarchical model-based diagnosis based on structural abstraction"*, *Artif. Intell.,* Vol. 155(1-2), pp. 147-182, 2003.

[7] Keuneke, A.: *"Device representation-the significance of functional knowledge"*, IEEE Intelligent Systems and Their Applications, Vol. 6(2), pp. 22-25, 1991.

[8] Chittaro, L.; Guida, G.; Tasso, C. and Toppano, E.: *"Functional and teleological knowledge in the multimodeling approach for reasoning about physical systems: a case study in diagnosis Systems",* Man and Cybernetics, IEEE Transactions on, vol 23, pp. 1718-1751, 1993.

[9] Bonarini, A. and Sassaroli, P.: *"Opportunistic Multimodel Diagnosis with Imperfect Models"*, Information Sciences, vol 103, pp. 161-185, 1997.

[10] Mathew, A. D., Zhang, S., Ma, L., Earle, T. and Hargreaves, D. J.: *"Reducing maintenance cost through effective prediction analysis and process integration"*, Advances in Vibration Engineering, vol. 5(2), pp. 87-96, 2006.

[11] Vachtsevanos, G., Lewis, F.L., Roemer, M., Hess, A. and Wu, B.: *"Intelligent fault diagnosis and prognosis for engineering systems"*, John Wiley & Sons Inc, ISBN: 978-0- 471-72999-0, 2006.

[12] Mathur, A., Cavanaugh, K., Pattipati, K., Willett, P. and Galie, T.: *"Reasoning and modeling systems in diagnosis and prognosis"*, Component and Systems Diagnostics, Prognosis, and Health Management, Proc. SPIE, vol. 4389, pp. 194-203, 2001.

[13] Luo, J.; Namburu, M.; Pattipati, K.; Qiao, L.; Kawamoto, M. and Chigusa, S.: *"Model-based prognostic techniques",* AUTOTESTCON 2003. IEEE Systems Readiness Technology Conference. Proceedings, pp. 330-340, 2003.

[14] Engel, S.; Gilmartin, B.; Bongort, K. and Hess, A.: *"Prognostics, the real issues involved with predicting life remaining",* Aerospace Conference Proceedings, 2000 IEEE, Proc. IEEE International Conference on Aerospace, vol 6, pp. 457-469, 2000.

[15] Roemer, M.; Byington, C.; Kacprzynski, G. J. and Vachtsevanos, G.: *"An Overview of Selected Prognostic Technologies with Reference to an Integrated PHM Architecture",* Impakt Technologies, 2007.

[16] Biteus, J.: *"Distributed Diagnosis and Simulation Based Residual Generators",* Dept. of Electrical Engineering, LiUTEK- LIC-2005:31, Thesis No. 1176, 2005.

[17] Heck, Florentin, Thomas, Längle and Heinz, Woern: *"A Multi-Agent Based Monitoring and Diagnosis System for Industrial Components",* Proceedings of the DX '98, pp. 63-69, 1998.

[18] Wörn, H., Längle, T., and Albert, M.: *"Multi-Agent Architecture for Monitoring and Diagnosing Complex Systems",* The Fourth International Workshop on Computer Science and Information Technologies, Patras, Greece, 1998.

8. Glossary

CS:	Complex System
DA:	Diagnostic Agent
FD:	Functional Description
HS:	Health Status
KB:	KnowledgeBase
KF:	Known Failure
LRU	Line Replaceable Unit
OO:	Out of Order
PA:	Prognostic Agent
RUL:	Remaining Useful Lifetime
SD:	Structural Description
SK	System Knowledge
TBAB:	Time Before Abnormal Behavior
TD:	Topological Description
UF:	Unknown Failure