A distributed Architecture to implement a Prognostic Function for Complex Systems

Xavier Desforges[1], Mickaël Diévart[2], Philippe Charbonnaud[1] and Bernard Archimède[1]

[1]*Univeristé de Toulouse, INPT, ENIT, Laboratoire Génie de Production, 65016 Tarbes, France*

xavier.desforges@enit.fr
philippe.charbonnaud@enit.fr
bernard.archimede@enit.fr

[2]*Aéroconseil, 31703 Blagnac, France*

mickael.dievart@aeroconseil.com

ABSTRACT

The proactivity in maintenance management is improved by the implementation of CBM (Condition-Based Maintenance) and of PHM (Prognostic and Health Management). These implementations use data about the health status of the systems. Among them, prognostic data make it possible to evaluate the future health of the systems. The Remaining Useful Lifetimes (RULs) of the components is frequently required to prognose systems. However, the availability of complex systems for productive tasks is often expressed in terms of RULs of functions and/or subsystems; those RULs provide information about the components. Indeed, the maintenance operators must know what components need maintenance actions in order to increase the RULs of the functions or subsystems, and consequently the availability of the complex systems. This paper aims at defining a generic prognostic function of complex systems aiming at prognosing its subsystems, functions and at enabling the isolation of components that needs maintenance actions. The proposed function requires knowledge about the system to be prognosed. The corresponding models are detailed. The proposed prognostic function contains graph traversal so its distribution is proposed to increase the calculation speed. It is carried out by generic agents.

1. INTRODUCTION

The implementation of the Condition-Based Maintenance (CBM) recommendations usually leads to the improvement of the equipment availability (Jardine, Lin and Banjevic, 2006; Scarf, 2007). The CBM actions are planned and led according to the health status of equipments. Monitoring, diagnostic and prognostic functions assess these statuses. The development of health assessment functions has often been considered as a downstream activity in the design process of complex systems with few allocated means. This has often led to a lack of collaboration with upstream activities and to centralized deployment in light computational modules although those functions have to process numerous pieces of data of different kinds. The consequences are increasing rates of useless replacements of devices, with the increasing complexity of the systems. Those replacements are not only costly but may also cause additional damage to the system.

Therefore, health assessment functions now become a major issue for complex system designers. Among those functions, the prognostic function aims at defining the future health of the system that contributes to plan productive tasks or maintenance tasks. Among the difficulties leading to the implementation of prognostic functions in complex systems, there are the numerous hardware or software components, devices, functions or subsystems of complex systems. Those equipments are designed, manufactured, assembled by different industrial partners (OEMs, suppliers, subcontractors, etc.). Each partner has a part of the needed knowledge to carry out the prognosis of the complex system. However, some pieces of this knowledge are parts of the own know-how of the partners and so they cannot be shared.

To tackle this difficulty, a decentralized/distributed architecture can be proposed. Indeed, such architectures enable the implementation of the Remaining Useful Lifetime (RUL) assessment and prognostic functions closer to components, devices, functions or subsystems. Therefore, each OEMs, suppliers or subcontractors can provide RUL assessment and prognostic functions for their equipments. Nevertheless, those functions have to collaborate in order to ensure the convergence of the prognostic process of

complex systems. Indeed, the union of local prognoses is not the global prognosis. To illustrate that point, let us consider a system made of a power supply and a computer. If the RUL of the power supply is lower than the one of the computer, the computer will not probably be able to carry out its activity beyond the RUL of the power supply. Agents that carry out RUL assessment and prognostic function can be used to ensure this collaboration. An agent is defined as a self-contained problem-solving computational entity able, at least, to perform most of its problem-solving tasks, to interact with its environment, to perceive its environment and to respond within a given time, to take initiatives when it is appropriate (Jennings and Wooldridge, 1995).

The aim of this article is to present an architecture for implementing a distributed prognostic function for complex systems. Firstly, the interest of distributed prognostic function is discussed. To implement prognostic function, knowledge about the complex system is necessary. Then the paper describes the principles of the prognostic function for complex systems. The notion of Time before Out of order (TBO) is introduced. Then the paper shows the proposed architecture that is based on the multi-agent system concept with generic agents and how to split up the modeled knowledge between the agents.

2. DISTRIBUTED PROGNOSIS

One aim of the Prognostic and Health Management (PHM) is to assess the ability of complex systems to carry out future tasks from diagnostic and prognostic results and the definition of the constraints of the future tasks. Roemer, Byington, Kacprzynski and Vachtsevanos (2007) advise that diagnostic and prognostic algorithm should be processed as close as possible to the monitored components and that the produced data should be then exploited by ascending the hierarchical structure of the complex system. Therefore, bringing the PHM into operation requires the implementation of prognostic functions.

If Vachtsevanos and Wang (2001) consider that the prognostic activity consists in assessing a RUL once an early detection of failure have been made, Lebold and Thurston (2001) consider that it is a reliable assessment of the RUL of a system or a device. From these studies it appears that the assessment of the RUL is the keystone of the prognostic activity. Indeed, the data it provides are used as decision support for maintenance planning and proactive maintenance (Iung, Monnin, Voisin, Cocheteux and Levrat, 2008) or for e-maintenance (Muller, Crespo Marquez and Iung, 2008).

Several studies have been led dealing with the design of prognostic functions of devices. Several techniques are described in (Vachtsevanos, Lewis, Roemer, Hess and Wu, 2006). Nevertheless, in the case of complex systems, the set of the RULs of the devices may not be enough to be a suitable decision support for maintenance or for productive tasks planning purposes. The sets of RULs shall therefore be processed. In complex systems, the number of RULs can be so huge that the only reasonable way to process them is distributed. Another good reason in using distributed architecture is that it enables implementations of prognostic processes as close as possible to the monitored devices as Roemer *et al*. (2007) advise it. Works dealing with distributed prognosis are quite recent and several ways to distribute the prognostic processes were already proposed.

In (Voisin, Levrat, Cocheteux and Iung, 2010) the prognosis is considered as a business process whose activities can be distributed in a context of e-maintenance. The mentioned distribution is made according to different actors located on different sites.

Saha, Saha, and Goebel (2009) propose an architecture made of several agents that can communicate between each other. An agent diagnoses a device and when it detects a fault it switches to the prognostic mode and informs a base station. The base station plans tasks, can reinitialize the processes of agents if errors are detected, it manages the accesses to resources like the ones to an external database and it manages the availability of agent in terms of computation load.

Dragomir, Gouriveau, Zerhouni and Dragomir (2007) present an architecture for health assessment that consists of two levels: the local level corresponds to the components and global level that is associated to the complex system. In this architecture, each local agent brings into operation several known prognostic methods according to the available knowledge about the monitored component. The global agent collects the health assessment data from the local agents and computes a health assessment for the system thanks to a neural network.

Takai and Kumar (2011) propose a decentralized prognoser for discrete event systems where local agents generate prognoses that are sent to the other agents. Then the agents cooperate in order to converge to a prognosis of the system thanks to an inference engine.

The sets of RULs shall also be processed according to knowledge as mentioned in (Saha *et al*., 2009).

3. KNOWLEDGE MODELING

During the design stage of a complex system, different kinds of knowledge are elaborated. Among them, the structural knowledge, the functional knowledge and the behavioral knowledge are required to implement prognostic functions (Reiter, 1992; Chittaro and Ranon, 2003). HAZard and OPerability (HAZOP) methodology, that is a process hazard analysis technique, enable to study not only the hazards of a system, but also its operability problems, by exploring the effects of any deviations from design conditions (Dunjo, Fthenakis, Vilchez and Arnaldos, 2010).

This methodology enables to identify functions and interconnections.

3.1. The functional knowledge modeling

The functional knowledge modeling aims at providing the sets of components that implements the functions of the complex system from the users point of views. Knowing the RUL of a function will help to plan future missions of the system and/or the maintenance actions it needs.

Therefore, the functional knowledge modeling consists in defining functions as sets of components or devices, which we call all "devices". Functions can also be made of functions. Complex systems can also be divided into subsystems. In that case, a subsystem can be considered as a set of functions. Thus, a complex system is made of subsystems. A subsystem is made of functions. A function is made of devices and/or functions.

Three types of functions must be considered for the computation of prognostic of functions.

Simple functions are functions that fail if one their entities fails (devices or functions) at least.

For reliability purposes, complex systems contain functions with redundancies. These functions are carried out by at least two entities (devices and/or subfunctions) that bring into operation the same activities, services... For example, we suppose that a flight control function of an aircraft is made of three functions we call "flight controllers". If one or even two flight controllers fail, the flight control can still carry out its task. However, if two flight controllers fail, there is no more redundancy. That is why we consider redundancies as functions called redundancy functions. Those functions are the only entities included in the functions with redundancies.

Subsystems are considered as sets of functions that are not included in other functions. Thus, subsystems can be considered as simple functions. The prognostic of the complex system can then be assessed from the prognostic of its subsystems.

The modeling of this knowledge can be done thanks to a UML (Unified Modeling Language), which is an object oriented modeling language, class diagram shown in figure 1.

3.2. The structural knowledge modeling

The structural modeling aims at representing the direct interactions between devices and their failure modes mainly in order to propagate the effects of failures (Worn, Langle, Albert, Kazi, Brighenti, Revuelta Seijo, Senior, Sanz-Bobi and Villar Collado, 2004).

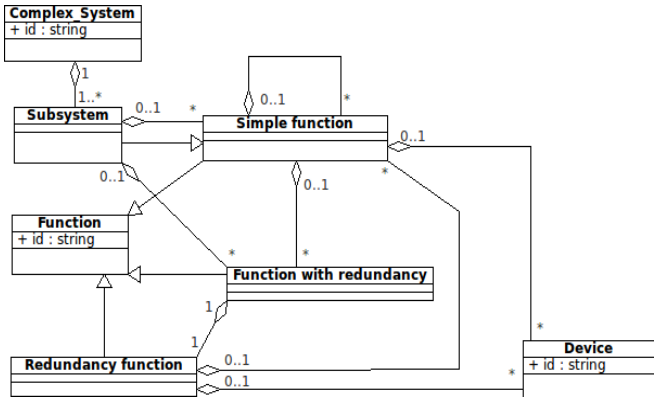


Figure 1. Functional knowledge model.

Failure Modes and Effects Analysis (FMEA) or HAZOP studies enable to collect the necessary knowledge for structural modeling. Indeed, those studies enable to identify what happens to other devices when one or several devices fail.

Therefore the structural knowledge can be modeled thanks to a set S_1 of arcs $D_{i,M_n} \rightarrow D_{j,M_{oo}}$ with $i \neq j$, where an arc means that the device D_j will be out of order (mode M_{oo}) if the failure mode M_n of a device D_i occurs. Let us note that the mode M_n can be the mode M_{oo}. However, some particular cases exist. For example, a laptop uses a power supply function. Let us simplify by assuming that the battery and the electricity distribution network carry out this function. If only the battery or the electricity distribution network fails, the computer still operates normally. That is why, the cases where a function F_k fails or becomes out order makes components become out of order must also be considered. Thus, the structural model must also represent a set S_2 of arcs with the same meaning $F_{k,M_p} \rightarrow D_{l,M_{oo}}$ with $D_l \notin F_k$. So, the structural model consists of the sets S_1 and S_2. Those sets of arcs represent a graph where the nodes are the failure modes of the devices and the functions of the complex system.

The out of order mode (M_{oo}) is quite relevant because it indicates that the origin of the predicted failure of an entity is not the entity itself.

3.3. The behavioral knowledge modeling

The behavioral modeling mainly aims at defining the dynamical behavior of a system. Behavioral models can be used to detect degradation and to analyze their trends in order to define the RUL of the monitored device.

The behavioral models used to prognose a device can be achieved thanks to three approaches (Byington, Roemer, Watson and Galie, 2003): experience-based, evolutionary and/or statistical trending or model based. The implemented behavioral models to prognose a complex system can so be numerous and of various kinds that it is difficult to consider

all of them. They also require design knowledge of devices, functions or subsystems that may reveal the know-how of their providers. Nevertheless, the most important things are what they contribute to produce: the RULs of the devices. We then assume that a monitoring layer made of one or several agents provides the RULs to the proposed prognostic function. The monitoring layer agents can so bring into operation the most suitable techniques to assess RULs of devices.

3.4. RUL modeling

In order to be processed by a prognostic function, a RUL has to assess a duration T between the instant t_0, at which it is calculated, and the predicted instant t_0+T at which the device will fail according to a given failure mode. Thus, a RUL must contain four entities (Voisin *et al.*, 2010) that are:

- the involved device,
- the involved failure or degraded mode,
- the instant at which it was calculated,
- the duration.

RULs are assessments, so fields can be added to deal with uncertainty or confidence level. However, the proposed prognostic function of complex system does not take into account any kind of uncertainty representations. The aim of this paper is to propose a principle for prognosing a complex system that can be distributed into generic agents. Handling uncertainty of RULs would likely lead to implement different processes into the tasks described in section 4.

The RULs that the monitoring layer provides are the base of the proposed prognostic function for complex systems but this function also needs the functional and structural knowledge.

4. PROGNOSTIC FUNCTION FOR COMPLEX SYSTEMS

This section is dedicated to the proposed generic principle for prognosing complex systems from RULs and from the modeled functional and structural knowledge. We assume that the monitoring layer sends to the Complex System Prognostic Function (CSPF) each RUL that it computes for each failure mode of the devices unless the out of order mode. The CSPF is divided into three main tasks that are:

1. the computation of the RUL of the device for which a RUL has been received,
2. the computation of the RUL of the devices that are interconnected (directly or not) to the device for which a RUL has been received,
3. the computation of the RUL of functions from the former task,

The process of the CSPF starts when a RUL is received from the monitoring layer.

4.1. Computation of the RUL of a device (task 1)

The RUL received by the CSPF from the monitoring layer is noted $RUL(D_i, M_j, t_k, T_l)$ where D_i is the device, M_j is the predicted failure mode, t_k is the instant at which the RUL was computed and T_l is remaining lifetime such as t_k+T_l is the predicted instant at which the failure will likely occur.

When a RUL $RUL(D_i, M_f, t_k, T_l)$ is received at the instant t, it is recorded and it replaces the last stored RUL for D_i with the failure mode M_j $RUL(D_i, M_f, t_{k-1}, T_{l-1})$ if $t_k > t_{k-1}$ else the task stops. If $t_k > t_{k-1}$, the RUL of D_i is then defined thanks to its last recorded RULs for all its failure modes. These RULs are noted $RUL(D_i, M_j, t_{kj}, T_{lj})$. The new RUL of D_i becomes $RUL(D_i, M_p, t, T_p)$ where p correspond to the failure mode for which:

$$T_p = \min_j \left(T_{lj} + t_{kj}\right) - t \quad (1)$$

Then this RUL is compared to the last recorded RUL for the device noted $RUL(D_i, M_q, t_{kq}, T_{lq})$ if $t + T_p < t_{kq} + T_{lq}$, $RUL(D_i, M_p, t, T_p)$ becomes the new RUL of the device D_i. It is stored and replaces $RUL(D_i, M_q, t_{kq}, T_{lq})$ and then, if at least one arc starts from the node D_{i,M_p}, the task 2 is processed else the task 3 is processed. If $RUL(D_i, M_p, t, T_p)$ does not become the new RUL of the device D_i the CSPF stops.

4.2. Computation of the RUL of the devices that are interconnected (task 2)

This task consists in propagating the new $RUL(D_i, M_p, t, T_p)$ in the graph described by the arcs $D_{k,M_{nk}} \rightarrow D_{j,M_{oo}}$ where the devices will likely be out of order earlier than previously predicted because of this new RUL. We must here introduce the notion of Time Before Out of order (TBO). This notion explains that a device will likely become out of order because of the failure M_p of the device D_i. This notion is meaningful for maintenance because it enables to localize the devices for which maintenance actions will be necessary. The TBO so contains five entities:

- the involved device,
- the device for which the RUL has generated the TBO,
- the failure mode of the device for which the RUL has generated the TBO,
- the instant at which the TBO was computed,
- the remaining time before the out of order mode occurs.

If the prognostic function handles uncertainty, TBOs must also contain fields dealing with this notion. That is not the case in this paper.

This second task does not consist of the computation of the new RULs of the interconnected devices but of their new TBOs. Two cases are considered:

- one for the arcs $D_{i,M_p} \rightarrow D_{j,M_{oo}}$,

- one for the arcs $D_{n,M_{oo}} \rightarrow D_{m,M_{oo}}$.

For all the arcs $D_{i,M_p} \rightarrow D_{j,M_{oo}}$, *RUL*($D_i$, M_p, t_{kp}, T_p) is compared to the last recorded TBOs of the devices D_j for which TBOs are noted *TBO*(D_j, D_x, M_{qx}, t_{kqx}, T_{qx}) with $j \neq x$. This comparison is made at the instant *t*. If $T_p + t_{kp} < T_{qx} + t_{kqx}$, then the new TBO of D_j becomes *TBO*(D_j, D_i, M_p, *t*, T_{pt}) with $T_{pt} = T_p - (t - t_{kp})$. This new TBO is recorded and replaces the previous stored one and it is propagated in the graph from the node $D_{j,M_{oo}}$ otherwise the propagation in the graph from the node $D_{j,M_{oo}}$ is stopped.

For all the other arcs $D_{n,M_{oo}} \rightarrow D_{m,M_{oo}}$ the TBO of the device D_n noted *TBO*(D_n, D_i, M_p, t_{pt}, T_{pt}) is compared to the last recorded TBO of D_m noted *TBO*(D_m, D_j, M_q, t_{qt}, T_{qt}). This comparison is made at the instant *t*. If $T_{pt} + t_{pt} < T_{qt} + t_{qt}$, then the new TBO of D_m becomes *TBO*(D_m, D_i, M_p, *t*, *T*) with $T = T_{pt} - (t - t_{pt})$. This new TBO is recorded and replaces the previous stored one and it is propagated in the graph from the node $D_{m,M_{oo}}$ otherwise the propagation from the node $D_{m,M_{oo}}$ is stopped.

This tasks ends when there is no more TBO to propagate. Then the prognostic of the functions must be done by the CSPF from the RULs and TBOs that were updated.

4.3. Computation of the RUL of the functions (task 3)

According to section 3.1, three types of functions must be considered for the computation of their prognoses: simple functions, functions with redundancies and redundancy functions.

The failure mode of a function is directly linked to the failure mode of one of its devices and/or to the missing service carry out by one of its subfunctions. That is why we only consider the TBOs of the functions instead of their RULs. The TBO of a function contains the same fields as the ones of the TBOs for devices except the involved function instead of the involved device.

The TBO of a function is computed if, at least, one RUL of one its devices has been modified or if one TBO of one of its entities (devices or functions), noted *X*, has been modified by the CSPF.

For a simple function F_j, if the RUL *RUL*(D_i, M_p, t_p, T_p) of one of its device has been modified its *TBO*(F_j, D_k, M_l, t_l, T_l) is modified if $t_p + T_p < t_l + T_l$ is verified then it becomes *TBO*(F_j, D_i, M_p, *t*, *T*) with $T = T_p - (t - t_p)$.

For a simple function F_j, if the TBO of one of its functions or of its devices *TBO*(X_q, D_i, M_p, t_p, T_p), where X_q denotes either the function or the device, has been modified its *TBO*(F_j, D_k, M_l, t_l, T_l) is modified if $t_p + T_p < t_l + T_l$ is verified then it becomes *TBO*(F_j, D_i, M_p, *t*, *T*) with $T = T_p - (t - t_p)$.

The new TBO is recorded and replace the previous stored one.

For functions with redundancies, the TBOs and/or RULs of their entities included in their redundancy functions are considered. For an entity that is a device D_i, we consider its RUL *RUL*(D_i, M_p, t_p, T_p) or its TBO *TBO*(D_i, D_x, M_q, t_q, T_q) and the value Tt_i that is computed with the relationship (2):

$$Tt_i = \min(t_p + T_p, t_q + T_q) \quad (2)$$

If an entity is a function F_j with *TBO*(F_j, D_x, M_q, t_q, T_q), the value Tt_j is computed with the relationship (3):

$$Tt_j = t_q + T_q \quad (3)$$

The *TBO*(F_{wr}, D_y, M_s, *t*, *T*) of a function with redundancies is computed from (4):

$$T = \max_i(Tt_i) - t \quad (4)$$

where D_y and M_s are the device and its failure mode for which the RUL or TBO that have the greatest value *Tt* and *t* is the instant at which the TBO has been computed. The new TBO is recorded and replace the previous stored one.

For a redundancy function, the TBOs and/or RULs of their entities are considered. For an entity that is a device D_i, we consider its *RUL*(D_i, M_p, t_p, T_p) or *TBO*(D_i, D_x, M_q, t_q, T_q) and the value Tt_i, that is also computed with the relationship (2). For a function entity F_j with *TBO*(F_j, D_x, M_q, t_q, T_q), the value Tt_j is computed with the relationship (3) too. The *TBO*(F_r, D_y, M_s, *t*, *T*) of a redundancy function is computed from (5):

$$T = \max_i{}^{n}(Tt_i) - t \quad (5)$$

where D_y and M_s are the device and its failure mode for which the RUL or TBO that have the *n*th greatest value *Tt* and *t* is the instant at which the TBO has been computed (generally *n*=2). The new TBO is recorded and replace the previous stored one.

If the TBO of a function F_k has changed and if its linked to a device by an arc $F_{k,M_p} \rightarrow D_{l,M_{oo}}$, which is in fact an arc $F_{k,M_{oo}} \rightarrow D_{l,M_{oo}}$, the task 2 is then processed with the same procedure as the one for arcs $D_{n,M_{oo}} \rightarrow D_{m,M_{oo}}$.

The TBOs of the functions and RULs of the devices are the elements of the prognostic.

4.4. Experimental results

The CSPF was successfully tested. In order to illustrate the results it provides, we propose the case study of the figure 2 where the arcs represent the structural knowledge and the boxes the functional knowledge.

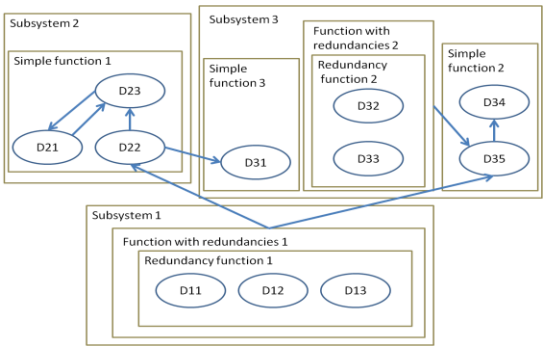


Figure 2. Case study.

In this system, only one mode of failure is considered for each device and the effect of this failure is supposed to be the same as the out of order mode. That is why only one kind of arcs is represented in Figure 2. However, one case of system with devices having two failure modes with different effects has also been successfully tested.

The table 1 shows an overview of the results provided by the CSPF for a simulated scenario. In this table the first column is the rank of reception of a RUL, the second column is the identifier of the device for which the monitoring layer has emitted a RUL, the third column is the date (which is the sum $t+T$ of the fields contained in the received RUL) at which the device will probably fail. The other columns of the table contain the dates (the sums $t+T$) of the RULs or TBO of devices and functions that are modified by the CSPF because of the received RUL. Dates in red mean that the date ($t+T$) of RUL's device is earlier than the date of its TBO.

The proposed CSPF is processed on-line each time a new RUL is received but it always leads to reduce the dates ($t+T$) of the RULs and TBOs of devices and functions. Thus, it is a pessimistic approach of the prognostic of the complex system. In that case, we can consider that the prognostic made on-line is dedicated to control operators. However, the CSPF can be run off-line for maintenance operators. The T values of the TBO must so be set to very great values. Then the CSPF is then run for all the RULs of each device. Thus, the maintenance operators have so indications about the devices that need maintenance actions. Once a device have been replaced or fixed, its RUL must be set at new values. In such cases, the T value of the RUL of the replaced or fixed device may be set to a value equal to its MTBF (Mean Time Between Failures) or MTTF (Mean Time To Failure). The T values of the TBO are then set to very great values. Then the CSPF is then run for all the RULs of each device.

However, The CSPF requires graph traversal and it can so be a long process. One way to reduce the computation time is to distribute the CSPF.

5. DISTRIBUTION OF THE CSPF

The proposed distribution of the CSPF consists of several agents that all process the CSPF. Assuming that there are few interconnections between subsystems, we propose one agent by subsystem in order to reduce the number of the sent messages between the agents. The agents have to be implemented into different computing platforms to increase the computation of the CSPF. Thus, the architecture can be represented as shown in figure 3.

received RULs			Devices, functions and subsytems																			
rank	device	date of RUL	D11	D12	D13	**RF1**	**FWR1**	**SS1**	D21	D22	D23	**SF1**	**SS2**	D31	**SF3**	D32	D33	**RF2**	**FWR2**	D34	D35	**SF2**	**SS3**
initial values			1000	1000	1000	**1000**	**1000**	**1000**	1000	1000	1000	**1000**	**1000**	1000	**1000**	1000	1000	**1000**	**1000**	1000	1000	**1000**	**1000**
1	D11	980	980																				
2	D12	990		990		**990**																	
3	D13	975			975	**980**	**990**	**990**	990	990	990	**990**	**990**	990	**990**					990	990	**990**	**990**
4	D21	960							960		960	**960**	**960**										
5	D22	985								985				985	**985**								**985**
6	D35	995																					
7	D33	970															970	**970**					
8	D33	965															965	**965**					
9	D31	970												970	**970**								**970**
10	D32	960														960		**960**	**965**	965	965	**965**	**965**
11	D23	995																					
12	D11	950	950			**975**																	
13	D13	955			955	**955**																	
14	D32	970																					
15	D33	940															940	**940**					
16	D12	965		965			**965**	**965**	965	965	965	**965**	**965**	965	**965**								
17	D32	945														945			**945**	945	945	**945**	**945**
18	D12	940		940		**950**	**955**	**955**	955	955	955	**955**	**955**	955	**955**								
19	D34	940																		940		**940**	**940**
20	D35	955																					
21	D31	950												950	**950**								
22	D21	950							950		950	**950**	**950**										
23	D22	980																					
24	D13	935			935	**940**	**950**	**950**		950													
25	D35	930																		930	930	**930**	**930**

Table 1. Example of results provided by the CSPF

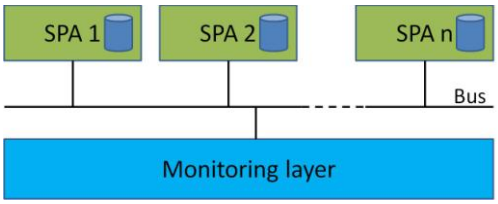


Figure 3. Distributed architecture scheme.

The SPAs are the Subsystem Prognostic Agent. They contain a database in which the functional and structural knowledge are represented as well as the structural interconnections between the subsystems. In this architecture the monitoring layer sends the RULs of the devices to the SPA that prognoses the subsystem to which the device belongs.

In the proposed distribution of the CSPF, the knowledge is distributed to the SPAs. The SPAs are generic agents because they all process the same tasks but their results depend on the knowledge modeled in their databases. The prognostic of the complex system is made of the RULs of the devices and the TBOs of the functions that are recorded by the SPAs.

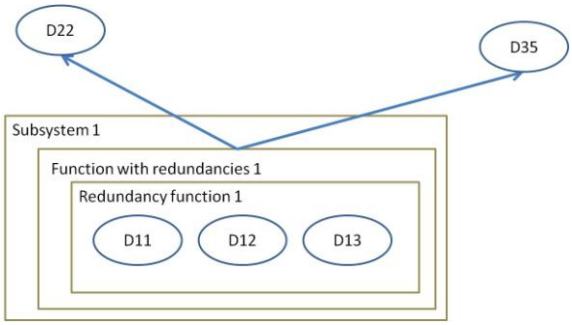
Thus the proposed architecture is also quite scalable. Indeed, adding a device or a new function consists mainly in adding functional and structural descriptions in the SPA of the subsystem it belongs to and, perhaps, some arcs for the structural models of the other SPAs. However, they do not need to modify the algorithms processed by the SPAs.

Assuming the case study of the figure 2, three SPAs are implemented.

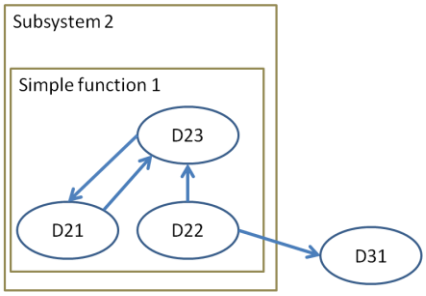
The parts of knowledge that are modeled in the databases of the SPAs are described in figure 4 (4.1 for SPA1, 4.2 for SPA2 and 4.3 for SPA3).

From the structural knowledge, an SPA knows to what SPA a TBO must be sent thanks to the identifier of the external devices.

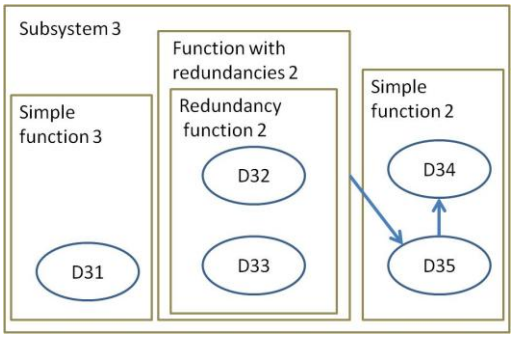
The communication between the SPAs can be modeled thanks to an UML sequence diagram as shown in figure 5 where the monitoring layer is considered as a single agent but it could made of several ones. Two SPAs are represented: the one that receives the RUL and one that represent the other SPAs. The task 1 is processed only once when a "New_RUL" message is received by a SPA. The "Modified_TBO" messages are emitted by the SPA from task 2 or task 3. Those messages indicate to the SPA that receives it what device is impacted by the TBO. Thus, when a SPA receives such a message, it processes the task 2 and the task 3. Even if it is distributed, the CSPF can be quite long to execute and "New_RUL" or "Modified_TBO" messages can be received while a SPA is running. So those messages have to be stored in a kind of buffer. The *t* values, which are fields of RULs and TBOs, can be used to sort the messages by increasing date.



(4.1)



(4.2)



(4.3)

Figure 4. Modeled knowledge in SPAs .

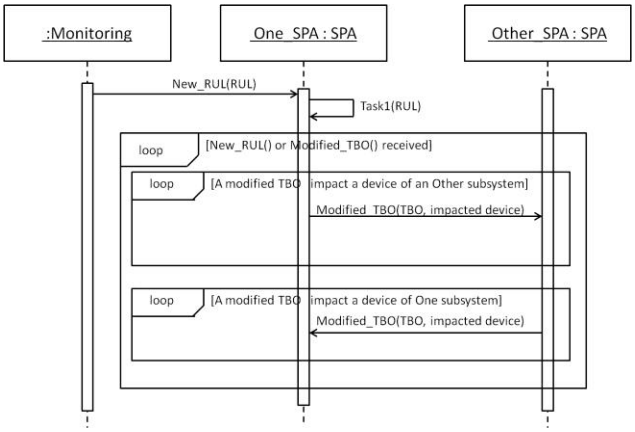


Figure 5. Sequence diagram.

6. CONCLUSION

This paper presented a generic algorithm to carry out the prognosis of complex system from the RULs of its devices. This approach requires functional and structural knowledge of the complex systems whose models were given. Requirements for the functional modeling were detailed. As the proposed prognostic principle requires graph traversal, its distribution into generic agents in order to reduce its computation time was presented. The distribution of the functional and structural models into the prognostic agents was proposed. The principle of prognosis provides online pessimistic results but it can be run off-line for more optimistic results. So one can consider that online process is dedicated to control operators (TBOs of functions) and that off-line process is dedicated to maintenance (TBOs of functions and RULs of devices).

The distributed simulation platform is under development. It uses a middleware to bring into operation the communication between the monitoring layer agents and SPAs. This platform shall enable to compare the centralized approach (with one SPA) and the distributed approach with several SPAs.

Another perspective will consist of the definition of functional and structural model to assess TBOs of devices and functions even when RULs of devices are increasing (when $t+T$ is increasing). Eventually, the problem of uncertainty of RULs could be addressed for prognosing complex systems.

REFERENCES

Byington, C., Roemer, M.J., Watson, M., Galie, T. (2003). *Prognostic enhancements to gas turbine diagnostic system*s, Proceedings of IEEE Aerospace Conference, vol. 7, pp. 3247-3255.

Chittaro, L., Ranon, R. (2003). Hierarchical model-based diagnosis based on structural abstraction, *Artificial Intelligence*, vol. 155, pp. 147–182

Dragomir, O., Gouriveau, R., Zerhouni, N., Dragomir, F. (2007). *Framework for a distributed and hybrid prognostic system,* Proceedings of 4th IFAC Conference on Management and Control of Production and Logistics.

Dunjo, J., Fthenakis, V., Vilchez, J.A., Arnaldos, J. (2010). Hazard and operability (HAZOP) analysis. A literature review, *Journal of Hazardous Materials*, vol. 173, pp. 19–32.

Engel, S., Gilmartin, B., Bongort, K., Hess, A. (2000). *Prognostics, the real issues involved with predicting life remaining,* Proceedings of the IEEE Aerospace Conference, vol. 6, pp. 457-469.

Iung, B., Monnin, M., Voisin, A., Cocheteux, P., Levrat, E. (2008). Degradation state model-based prognosis for proactively maintaining product performance, *CIRP Annals - Manufacturing Technology*, vol. 57, pp.49–52.

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

Jennings, N.R., Wooldridge, M. (1995) Applying agent technology, *Applied Artificial Intelligence*, vol. 9, pp. 357-369.

Lebold, M., Thurston, M. (2001) *Open standards for condition-based maintenance and prognostics systems*, Proceedings of the 5th annual maintenance and reliability conference (MARCON 2001).

Muller, A., Crespo Marquez, A., Iung, B. (2008). On the concept of e-maintenance: Review and current research, *Reliability Engineering and System Safety*, vol. 93, pp. 1165–1187.

Reiter, R. (1992). *A theory of diagnosis from RST principles,* Readings in model-based diagnosis, Morgan Kaufmann Publishers, pp. 29-48.

Roemer, M., Byington, C., Kacprzynski, G.J., Vachtsevanos, G. (2007). An overview of selected prognostic technologies with reference to an integrated PHM architecture. *Technical Report, Impakt Technologies.*

Saha, B., Saha, S., Goebel, K. (2009). *A distributed prognostic health management architecture*, Proceedings of the Conference of the Society for Machinery Failure.

Scarf, P. (2007). A Framework for Condition Monitoring and Condition Based Maintenance, *Quality Technology & Quantitative Management*, vol 4, pp. 301-312.

Takai, S. Kumar, R. (2011). Inference-Based Decentralized Prognosis in Discrete Event Systems, *IEEE Transactions on Automatic*, vol. 56, pp.165-171.

Vachtsevanos, G., Wang, P. (2001). *Fault prognosis using dynamic wavelet neural networks.* Proceedings of AUTOTESTCON IEEE Systems Readiness Technology Conference, pp. 857-870.

Vachtsevanos, G., Lewis, F. L., Roemer, M., Hess, A., Wu, B. (2006). *Intelligent fault diagnosis and prognosis for engineering system.* Hoboken, NJ: John Wiley & Sons, Inc.

Voisin, A., Levrat, E., Cocheteux, P., Iung, B. (2010). Generic prognosis model for proactive maintenance decision support: application to pre-industrial e-maintenance test bed, *Journal of Intelligent Manufacturing*, vol. 21, pp. 177–193.

Worn, H., Langle, T., Albert, M., Kazi, A., Brighenti, A., Revuelta Seijo, S., Senior, C., Sanz-Bobi, M.A., Villar Collado, J. (2004). Diamond: distributed multi-agent architecture for monitoring and diagnosis. *Production Planning and Control*, vol. 5, pp. 189-200.