Available online at www.sciencedirect.com
SCIENCE DIRECT®
Energy 29 (2004) 2553–2572
ENERGY
www.elsevier.com/locate/energy

Automatic diagnostics and prognostics of energy conversion processes via knowledge-based systems

Tatiana Biagetti, Enrico Sciubba *

Mechanical Engineering Department, University of Roma 1, Via Eudossiana 18, 00184 Rome, Italy

Abstract

This paper presents a critical and analytical description of an ongoing research program aimed at the implementation of an expert system capable of monitoring, through an Intelligent Health Control procedure, the instantaneous performance of a cogeneration plant. An application has been tested on a real plant, located on the grounds of the ENEA-Casaccia Energy Laboratories. The expert system, denominated PROMISE as the Italian acronym for *PRO*gnostic and *I*ntelligent *M*onitoring *E*xpert *S*ystem, generates, in real time and in a form directly useful to the plant manager, information on the existence and severity of faults, forecasts on the future time history of both detected and likely faults, and suggestions on how to control the problem.

The expert procedure, working where and if necessary with the support of a process simulator, derives from real-time data a list of selected performance indicators for each plant component. For a set of faults, pre-defined with the help of the plant operator, proper rules are defined in order to establish whether the component is working correctly; in several instances, since one single failure (symptom) can originate from more than one fault (cause), complex sets of rules expressing the combination of multiple indices have been introduced in the knowledge base as well.

Creeping faults are detected by analyzing the trend of the variation of an indicator in a pre-assigned interval of time. Whenever the value of this ''discrete time derivative'' becomes ''high'' with respect to a specified limit value, a ''latent creeping fault'' condition is prognosed.

The expert system architecture is based on an object-oriented paradigm. The knowledge base (facts and rules) is clustered: the chunks of knowledge pertain to individual components. A graphic user interface (GUI) allows the user to interrogate PROMISE about its rules, procedures, classes and objects, and about its inference path. The paper also presents the results of some tests conducted on the real plant.

* Corresponding author. Dipartimento di Meccanica e Aeronautica, Universita di Roma 1, ''La Sapienza'', via Eudossiana 18, 00184 Rome, Italy. Tel.: +39-06-44585-244; fax: +39-06-44585-249.
E-mail address: enrico.sciubba@uniroma1.it (E. Sciubba).

0360-5442/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.energy.2004.03.031

Nomenclature

p	pressure, Pa
T	temperature, K
m	mass flow rate, kg/s
x	molar concentration
U	heat exchange coefficient, W/(m^2 K)
a	vibration amplitude, m
w_{compr}	compressor work
ω	angular velocity (rad/s)
η	efficiency
β	compression ratio
$(\,)_{\mathrm{d}}$	design conditions
Rms	root mean square norm
c_{p}	specific heat, J/(kg K)
fl	(fuzzy) degree of belonging

1. Introduction

The process of designing, constructing and operating energy conversion systems is a very complex task. If any part of this process breaks down, the plant fails to deliver its benefits. The monetary losses originated by such failures are indeed important, but even more important is the resource destruction that constitutes the end result of the fault (because the reduced productivity must be "made up" by some alternative generation system). All modern design methods contain procedures that take into due account variable load conditions (off-design operation), availability losses due to scheduled and unscheduled maintenance and performance degradation due to wear and fouling of the equipment. The "self-diagnosis" enacted by current control systems is though very rudimentary, and actually they are intentionally designed so that the "intelligence" in their responses is zero: the design concept being that a control system must act fast, safely and deterministically, and that its logic must be "linear", to facilitate the interpretation of its response by human plant operators [3].

Intelligent process management tools (IPMTs) have been conceived to go two steps further [9,12]: they are not only by definition capable of producing an intelligent diagnosis of the present state of the plant (such systems are sometimes called "Health Monitoring Systems"), but also to enact a prognostic action, making intelligent estimates of the future state of the plant under the foreseen boundary conditions [7]. Finally, they can use design, operation and load-scheduling data, together with other relevant external information (like for instance local weather forecasts or projected operating load curves of similar plants in the same "fleet") to provide operators with valuable information about the "optimal" operating curve of the plant in some future period T [8]. The practical implementation of IPMTs will no doubt require some modifications in the present design procedures, especially for what sizing and physical assembly of equipment

are concerned. The present paper describes the development and the actual field implementation of a diagnostic and prognostic tool, specifically designed for a gas turbine-based cogeneration system; its development constitutes though a useful paradigm for different applications.

Let us define the plant availability factor PF as the ratio of the total equivalent full load operating hours in a year and the total number of hours in the year. It is apparent that no energy conversion plant can operate with PF equal to 1, due to three orders of reasons:

(a) plant shutdowns due to scheduled maintenance;
(b) plant shutdowns due to unscheduled maintenance;
(c) plant shutdowns due to sudden failures.

It is useful for our purposes to separately account for events of type "b", that imply the replacement of a component for which an early failure has been prognosed, and events of type "c", in which the replacement is done after the failure has forced a plant shutdown.

Our study specifically concentrates on *plant shutdowns due to sudden failures.* Strictly speaking, "*sudden catastrophic failures"* rarely happen as such, and when they do, they are obviously by definition unforeseeable. But extensive field studies have conclusively shown that most of the failures we call "sudden" are in reality caused by a series of component-localised phenomena that lead to a (usually very small but still significant) deterioration of its performance. Our efforts may thus be redirected to the early detection of these "performance degradation"-warning signals. The method to follow is in principle straightforward: a sufficient number of "critical points" in the process are monitored in real time, and a specific series of performance decay indicators are computed. As soon as one of these "creeping faults" has been detected, the operator, working under tight co-operation with the designer and the plant manager decides whether to execute an immediate shutdown to fix the fault, or to wait until the next scheduled maintenance intervention.

2. The general conceptual layout of a diagnostic/prognostic system

In the language of artificial intelligence (AI), we say that a procedure is enacted by an "Agent". In the following description, the agent is our expert system: but it is easy to recognise a high degree of similarity between the individual steps of the procedure and the actions that a human operator would take when executing the same task. An interesting and explicit comparison is made in [10]: our scope here is to show that both the procedures in its entirety and each one of its single steps is feasible at the present level of AI technology. We shall separately describe the diagnostic and the prognostic procedures, but will show later (Section 3.3.5 here below) that they both admit a meta-procedure, i.e., they can be embedded in a single code.

2.1. A diagnostic system

A possible (non-unique) procedure for an automatic diagnostic system consists of the following steps:

(1) The intelligent agent ("IA") must identify in real time (in practice, at sufficiently small time intervals) the operational state of the process. This requires that IA be endowed with an

efficient interface with a process data collection system, and to possess an object-like representation of the interconnection between the individual components of the process. Such a representation consists for IA of a vector of length N containing an ordered set of *measurables*, i.e., of process parameters that identify the state (mass flow rates, pressures, temperatures, etc.).

(2) IA must compare, at each time step, the detected operational state with the expected one. To do this, IA must have access either to a pre-determined operational schedule of the process, or to a reliable process simulator that provides it with such a "design datum".

(3) If the value of the kth measurable differs from the corresponding "design datum" by more than a preset tolerance, IA activates a monitoring-and-control procedure on the component to which this measurable pertains.

(4) IA verifies whether the "failure" just detected appears in one of the "fault chains" contained in its knowledge base. If it does, then IA proceeds to step 5 here below. If it does not, IA activates a sub-procedure to monitor k for a prescribed period of time, and notifies the (human) plant operator of this action.

(5) If the event "kth measurable out of range" belongs to one or more fault chains known to IA, the agent launches a monitoring-and-control procedure on all measurables $i,j,\ldots,p$ that appear together with k in the detected fault chains.

(6) If a fault chain is indeed identified as "active", IA will: a—notify the plant operator; b—consult its knowledge base to search for remedial actions (e.g., adjustment of other process parameters to compensate for the derangement in k); c—decide whether it is possible to wait for the next scheduled maintenance intervention or a repair/substitution is immediately necessary.

2.2. A prognostic system

A possible, and also non-unique, procedure for an automatic prognostic system consists of the following steps:

(1) The IA must compare at each time step (or with a pre-determined time-sampling procedure) the operational state of the process.

(2) IA *projects* the detected operational state forward in time, founding this projection on the most recent time history (2 or more previous time steps) of the process.

(3) If the projected value of the kth measurable at $t + \Delta t$ activates one of the known fault signatures, or if it shows an undesirable trend in the time history of x_k (e.g., "$\mathrm{d}x_k/\mathrm{d}t$"too high" according to some norm), IA activates a monitoring-and-control procedure on the component to which this measurable pertains.

(4) IA also launches a monitoring-and-control procedure on all measurables $r,s,\ldots,z$ that are related to k (i.e., whose values are known to be functionally linked to the value of x_k).

(5) Otherwise, IA keeps monitoring x_k for a pre-defined time interval, and notifies the plant operator of this action.

6) If IA estimates that a fault chain may be "activated" by an excessive variation of x_k, it will: a—notify the plant operator; b—consult its knowledge base to search for remedial actions (e.g., adjustment of other process parameters to compensate for the derangement in

x_k); c—decide whether it is possible to wait for the next scheduled maintenance intervention or a repair/substitution is immediately necessary.

Notice the remarkable analogy between the steps of the diagnostic and those of the prognostic procedure: we shall return to this point in Section 3.3.5.

3. Theoretical and practical aspects of the implementation of the intelligent agent

For the expert (or intelligent) agent to be in fact "expert" and "intelligent" in performing his task, its knowledge base (KB) must be as "complete" and "exact" as possible. Here, these adjectives take a meaning much more precise than in common language:

- "Complete" means that there must exist a one-to-one mapping of all rules and information available to the human operator and this KB.
- "Exact" means that this mapping is logically consistent, i.e., that no logical chain of induction correctly derived from the KB contradicts any of the rules and information available to the human operator.

3.1. The meta-rules of failure detection

It is known from AI theory [10,11] that it is convenient to re-organise, wherever possible, the knowledge bits acquired during the knowledge acquisition phase, because such a systematisation goes in favour of the transparency and the accessibility of the "built-in-logic" of the expert system. In the case in point, we are dealing with "failures" of a system, and we have found it useful to construct our KB on the basis of the following seven meta-rules:

(1) There exists a finite number of possible types of failure, and for each one of them there exists at least one specific signature, i.e., a unique combination of the process parameters.
(2) There are no sudden failures: every possible failure is "forewarned" by a drifting of the point representative of the operational state of the plant, on a path that leads to a specific attractor in the state space (the failure point).
(3) Each one of these "drifting" processes has a characteristic time scale that depends both on the component and on the type of failure.
(4) A convenient way to represent such a drifting is that of employing a proper set of dimensionless indicators, each defined as the ratio of the instantaneous value of a measurable of interest to its "design" value. Notice that such a design value is in reality a time-dependent quantity: it is the value expected for the same instantaneous operative conditions but without any derangement.
(5) The process of "failure formation" is described by at least one "fault chain", i.e., an ordered list of the immediate causes of the failure. There may be more than one chain (see point 6 here below). Each chain though has at least two fuzzy aspects: first, the "causes" it contains are necessary, but not sufficient (for example, for a creep failure in a first row statoric blade in a gas turbine, it is necessary that the gas temperature at turbine inlet be

higher than a certain design limit; but once the temperature exceeds this limit, failures are not certain). Second, even this necessity is affected by some degree of uncertainty (for example, a blade failure may happen even if the gas temperatures are below the design limit).

(6) Some of the fault chains may be concurrent. That is, the same failure stem from one or the other or from a combination of two (or more) fault chains.

(7) Many of the fault signatures are non-local: the values of measurables detected at locations physically remote from the point where the failure actually takes place may be affected by the drifting process mentioned in point (2). In this case, we say that these measurables (and the indicators constructed on them) are correlated with the ones immediately affected by the failure.

3.2. Formalisation of the fault signatures and choice of the fault indicators

For the monitoring of all energy conversion plants (e.g. "efficiency", "mechanical output", "thermal output"), and especially for gas turbines and their derivates (combined and cogenerating plants), a very extended database is available. There is a body of international industrial standards, often validated by Governmental Agencies, which regulates even the fine details of the type and tolerance of the measurables. Our approach here is though rather different: we are not interested in the abidance by contractual specifications, but rather in a (continuous) monitoring of whether the system operates within a certain number of admissible states (Fig. 1 represents a very simple case—a compressor plant—in which all the admissible states belong to one

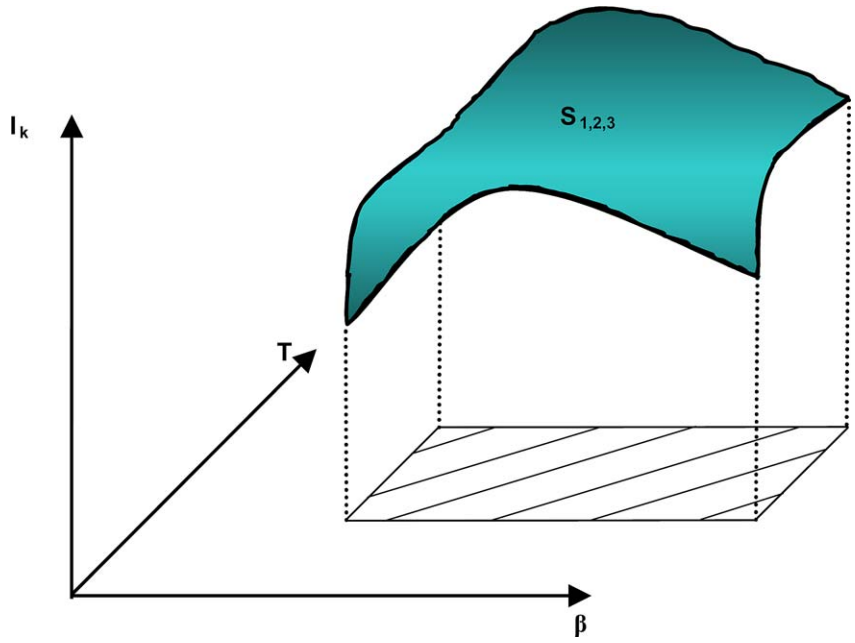

Fig. 1. $S_{1,2,3}$ is the three-dimensional acceptable "operating field" for a compressor in the space (*efficiency* I_k, *compression ratio* β, *temperature* T). Values outside of $S_{1,2,3\ w}$ correspond to faulty operation.

single continuous set). Therefore, the various sets of measurables defined by International Standards do not suffice for our purpose: in fact, our KB complements them with other knowledge bits derived from design handbooks, operators manuals and interviews with field experts. Strictly speaking, each type of plant has its own proper "Indicators list": Table 1 reports the list of the failures to be diagnosed/prognosed and Table 2 the indicators adopted in the application discussed in Section 4 here below (the plant layout is represented in Fig. 2).

3.3. From the mathematical to the operational problem position

In this section, we describe the process that leads from the mathematical problem position to its practical implementation into a set of AI procedures. Such a process is general, and applies with very little modifications to a wide range of similar problems. However, because of its very nature, it is best described not in an abstract "state space", but with reference to some specific process.

Table 1
List of possible faults included in the KB

Component	Possible fault(s)	Component	Possible fault(s)
Filter	Leakage Fouling	Secondary heat Exchanger	Fouling
Compressor	Stall Choking Fouling Excessive exit temperature Malfunctioning	Primary loop	Fouling
Primary Combustion Chamber	Fouling Excessive pressure losses CH_4- or H_2O valve failure	Secondary loop	Fouling
Primary fuel Injector	Fouling Failure	Main pump	Cavitation Malfunctioning
By-pass stack	Secondary combustion reactions Fouling Leakage	Heat recovery boiler	Fouling
Boiler stack	Secondary combustion reactions Fouling Leakage	Main shaft	Near-critical vibration frequencies
Turbine	Fouling Choking Excessive inlet temperature	Afterburner	CH_4 injector fouling
Primary heat exchanger	Fouling		

Table 2
The proposed indicators

Air filter	$I_1 = (p_1 - p_2)/(p_1 - p_2)_d$
Compressor	$I_2 = c_{p23}(T_3 - T_2)/[cp_{23}(T_3 - T_2)]_d$
	$I_3 = \eta_C/(\eta_C)_d$
	$I_4 = \beta_C/(\beta_C)_d = p_3/(p_3)_d$
	$I_5 = m_3/(m_3)_d$
Combustion chamber	$I_6 = \Delta p_{cc}/\Delta(p_{cc})_d = (p_3 - p_4)/(p_3 - p_4)_d$
	$I_7 = m_8/(m_8)_d$
Fuel injector	$I_8 = m_7/(m_7)_d$
By-pass stack	$I_9 = (X_{NOX,6})/(X_{NOX,6})_d$
	$I_{10} = (X_{CO,6})/(X_{CO,6})_d$
	$I_{11} = (T_5 - T_6)/(T_5 - T_6)_d$
Boiler main stack	$I_{12} = (X_{NOX,12})/(X_{NOX,12})_d$
	$I_{13} = (X_{CO,12})/(X_{CO,12})_d$
	$I_{14} = (T_{11} - T_{12})/(T_{11} - T_{12})_d$
Turbine	$I_{15} = \eta_T/(\eta_T)_d$
	$I_{16} = T_4/(T_4)_d$
	$I_{17} = m_4/(m_4)_d$
Electrical generator	$I_{18} = \omega_t/(\omega_t)_d$
Afterburner	$I_{19} = (T_{10} - T_9)/(T_{10} - T_9)_d$
	$I_{20} = m_{13}/(m_{13})_d$
H_2O/H_2O heat exchanger, shell side	$I_{21} = (p_{16} - p_{17})/(p_{16} - p_{17})_d$
	$I_{22} = U(T_{16} - T_{17})/[U(T_{16} - T_{17})]_d$
H_2O/H_2O heat exchanger, tube side	$I_{23} = U(T_{23} - T_{22})/[U(T_{23} - T_{22})]_d$
Primary hydraulic loop	$I_{24} = (p_{20} - p_{15})/[p_{20} - p_{15}]_d$
Secondary hydraulic loop	$I_{25} = (p_{21} - p_{22})/[p_{21} - p_{22}]_d$
Main pump	$I_{26} = m_{19}/(m_{19})_d$
	$I_{27} = (p_{19} - p_{18})/(p_{19} - p_{18})_d$
Heat recovery boiler	$I_{28} = U(T_{10} - T_{11})/[U(T_{10} - T_{11})]_d$
Shaft (vibrations)	$I_{29} = \left[\mathrm{rms}(a_i)/\mathrm{rms}(a_{i,})_d\right]$

3.3.1. The mathematical formulation

Define the "performance function" Π_P of an energy conversion process **P** as the deterministic mathematical relation between the instantaneous process output(s) and a set of N process parameters that we call the measurables: Π_P can be thought of as an operator that, applied to the vector **X** of measurables, generates the output vector **Y** (the mapping performed by the operator on the quantity enclosed within the brackets $\langle . \rangle$).

$$\begin{aligned} &\Pi_P\langle \mathbf{X} \rangle = \mathbf{Y} \\ &\mathbf{X} = (x_i) = (p_j, T_j, m_j, b_j, \ldots) \\ &\mathbf{Y} = (y_k) = (P_1, P_2, P_3, \ldots) \end{aligned} \tag{1}$$

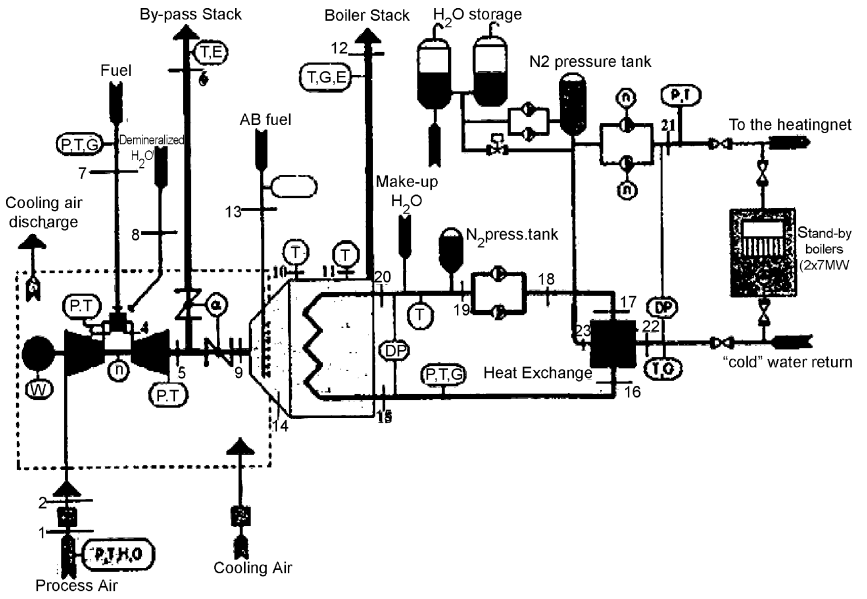

Fig. 2. ICARO power plant layout (from [1]).

Now, denote as $\mathbf{X}'$ a deranged operational state, in which some of the measurables have taken values slightly (but detectably) different from their "design datum". We can formally compute the new outputs by:

i Computing the new functional value assumed by the operator $\boldsymbol{\Pi}_P$:

$$\Pi_P\langle \mathbf{X}' \rangle \approx \Pi_P\langle \mathbf{X} \rangle + \left|\frac{\mathrm{d}\Pi}{\mathrm{d}X}\right|^{\mathrm{T}} \langle \mathbf{X} - \mathbf{X}' \rangle + \mathrm{O}(\mathrm{X} - \mathrm{X}')^2 \tag{2}$$

where $|\mathrm{d}\Pi/\mathrm{d}X| = |\mathrm{d}\Pi/\mathrm{d}x_j|$ represents the term-by-term derivative of a vector and not the total differential, and "O" means "of the order of".

ii Applying the modified operator to the new vector of measurables:

$$\Pi_P\langle \mathbf{X}' \rangle = \mathbf{Y}' \tag{3}$$

Such an exact mathematical approach is, however, not applicable in practice, because we do not know the exact form of Π_P except in very few, *ideal*, cases, of no practical interest (the so-called "textbook cases"). Fortunately, there is only a very limited number of outputs that we want to control, or, in other words, the vector $\mathbf{Y}$ represents a small set of length M[1]. Therefore, we can proceed as follows:

[1] M usually assumes values between 1 and 3 for energy conversion plants, and grows to order 10 in some chemical processes or in heat exchangers networks.

i separately consider M "components" of Π_P, each one pertaining to one of the outputs[2]:

$$\Pi_P = \begin{vmatrix} \Pi_{P_1} \\ \Pi_{P_2} \\ \cdots \\ \Pi_{PM} \end{vmatrix} \tag{4}$$

ii directly measure, by numerical- or field experiments, the differentials in Eq. (2)):

$$\frac{\mathrm{d}\Pi_k}{\mathrm{d}x_j} \approx \frac{\Delta P_k}{\Delta x_j} \tag{5}$$

iii substitute (4) and (5) in (3), making use of the discrete equivalents of (1) and (2), and obtain:

$$\begin{aligned} \Pi_P\langle X'\rangle &= \begin{vmatrix} \Pi_{P_1} \\ \Pi_{P_2} \\ \cdots \\ \Pi_{PM} \end{vmatrix}^{\mathrm{T}} \langle X'\rangle \approx \Pi_P\langle X\rangle + \left|\frac{\mathrm{d}\Pi_P}{\mathrm{d}X}\right|^{\mathrm{T}} \langle X' - X\rangle \\ &= \begin{vmatrix} \Pi_{P_1} \\ \Pi_{P_2} \\ \cdots \\ \Pi_{PM} \end{vmatrix}^{\mathrm{T}} \langle X\rangle + \begin{vmatrix} \dfrac{\mathrm{d}\Pi_{P_1}}{\mathrm{d}X} \\ \dfrac{\mathrm{d}\Pi_{P_2}}{\mathrm{d}X} \\ \cdots \\ \dfrac{\mathrm{d}\Pi_{P_M}}{\mathrm{d}X} \end{vmatrix}^{T} \langle X' - X\rangle \end{aligned} \tag{6}$$

in which the terms $\mathrm{d}\Pi_{P_i}/\mathrm{d}X$ are to be expanded by expressions like (5).

Notice that:

(1) The first term on the right-hand side of 6 is the performance function at design point, and is therefore known (or computable, if we are performing a numerical experiment).
(2) Each one of the individual components of the transposed vector in the second term on the right-hand side of 6 can be measured (or computed) by means of Eq. (5)).
(3) The last term in $\langle..\rangle$ brackets in 6 represents the variation in the vector of measurables, and is provided by the process monitoring system.

[2] Each one of these sub-matrices may span a different set of measurables. This property, which is important in practice, is not formally exploited here to simplify the exposition.

For all practical purposes, Eq. (6) may be therefore considered as an experimental correlation between the derangements in the relevant measurables and the process outputs. If we define a tolerance level on the output variation for each one of the outputs, we can then use 6 to check whether the detected derangement in the measurables is acceptable or not, i.e., if it must be considered a failure or not. The problem seems to be deterministically solved: let us assume for the moment that it is indeed so, and see how we can translate Eq. (6) into a logical diagnostic procedure.

3.3.2. The operational procedure

Each sub-unit of the plant is subject to certain types failures, and the number of these types is finite (see Sections 3.1 and 3.2). These failures are identified by a derangement of certain outputs from their expected value, and are detected by monitoring the variations of (usually a small number of) some of the measurables of the process. Therefore, our IA must possess a knowledge of what combination of variations in the measurables leads to what kind of failure. Denoting by ΔI_k the event "variation of measurable k" and by F_j the jth type of failure, we call the causal link

$$\Delta I_k \rightarrow F_j \tag{7}$$

a *fault chain*. There may be different types of chains, as mentioned in Section 3.1:

(a) simple fault chain: $\Delta I_A \rightarrow F_I$. Failure of type "$I$" is causally linked to the variation of only one indicator.
(b) composite fault chain: $(\Delta I_a \cup \Delta I_b \ldots \Delta I_z) \rightarrow F_I$. Failure of type "$I$" is causally linked to the simultaneous variation of z indicators.
(c) Multiple fault chain: $(\Delta I_a \cap \Delta I_b \cap \ldots \Delta I_z) \rightarrow F_I$. Failure of type "$I$" is causally linked to the variation of either one of z indicators.
(d) concurrent fault chain: $(\Delta I_a \cap \Delta I_b \cap \ldots \Delta I_z) \cup (\Delta I_\alpha \cup \Delta I_\beta \cup \ldots \Delta I_\zeta) \rightarrow F_I$. Failure of type "$I$" is causally linked to the simultaneous appearance of the composite and multiple events $a \ldots z$ and $\alpha \ldots \zeta$.

There are obviously several possibilities for (d), irrelevant to our discussion (they are given by the possible permutations between the $\cup$ and the $\cap$ operators).

The strategy for developing an intelligent diagnostic agent is now clear: once it is established which types of failures we want IA to "diagnose", all we have to do is to insert into its KB the following information:

(1) a list of failures (IA's *universe of events*);
(2) a list of indicators, together with all of the rules necessary to acquire them, compute their derivatives, etc. (IA's *universe of perception*);
(3) a set of rules of the type $a \ldots d$ above, linking possible combinations of perceptions (fault chains) with possible failure events (IA's *inferential engine*).

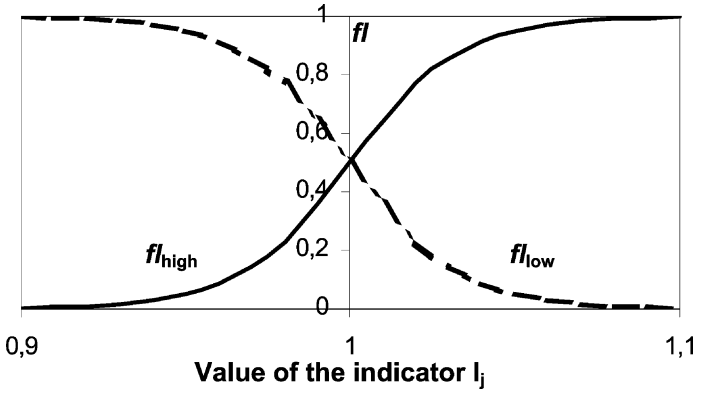

Fig. 3. Representation of the likeliness for the value of an indicator I_j to be "too high" or "too low". "fl" is the degree of fuzzy likeliness: fl = 1 ⇒ true; fl = 0 ⇒ false.

3.3.3. Fuzzy decisions

Both for logical and practical reasons,[3] a purely deterministic diagnosis as the one outlined in the previous section is not acceptable. As discussed at length in [1,2], a failure F_i is not certain even if its "necessary" fault chains defined in Section 3.3.2 are active; by converse, the absence of the same failure is not certain once the same chains are inactive. *The failure event is therefore devoid of a classical causal link*: the arrow in Eq. (7) becomes a fuzzy arrow, and the realisation of the event "failure" attains a continuous set of values of *likelihood* in the surrounding of the tolerance limit for the variation of the indicators. In words, we say that our field experience convincingly shows that:

(a) Once a fault chain, consisting of a certain combination of variations in one or more of the indicators, is detected, the corresponding failure may not happen even if the norm of such a variation has exceeded the tolerance limit.
(b) By converse, a failure may happen while its corresponding fault chain is still below its tolerance limit.

Mathematically, we express such a multi-valued logic by means of a fuzzy representation: while in classical logic an event A (existence of a failure) either belongs to ($A \subset S$) a set S (existence of the corresponding fault chain) or it does not ($A \not\subset S$), in fuzzy logic A may belong to S with a degree of certainty continuously varying between 0 (A certainly $\not\subset S$) to 1 (A certainly $\subset S$). Fig. 3 shows a representation of a fuzzy function: depending on the percentual derangement of I_k from its design value, the likelihood of failure may be "low" in varying degrees, $L(I_k/I_{0,k})$, and "high" in varying degrees, $H(I_k/I_{o,k})$. A reasonable measure of the actual likelihood of failure is provided by some properly weighted combination of L and H:

$$\text{Likelihood}(F) = w_L(\Delta I_k)^* L(\Delta I_k) + w_H(\Delta I_k)^* H(\Delta I_k) \tag{8}$$

[3] The main practical reason is that a controller based on a simple on/off logic is likely to generate operational instabilities.

With reference to Fig. 3, fuzzy logic thus assigns a likelihood to the event "fault K = true" that grows from $I_j = 0.95$ to $I_j = 1.05$. This means that higher the I_j, the higher is the likelihood of a fault event to be realised. At the same time and in the same interval of values for I_j, there is a decreasing likelihood that the fault is NOT realised: the final truth value of fault K is a linear combination (in other case, the sum) of the two truth values for the given measured value of indicator j. If the fault signature consists of more than one indicator, proper fuzzy algebra rules [10,11] dictate how to correctly combine the individual truth values to reach a joint (or global) truth-value for the composite signature. Notice that:

(a) The curves expressing the likelihood are S-shaped curves, which allow for a better field tuning.
(b) In the limit case of an extremely steep S-shaped curve, we recover the binary logic formulation (fault K either TRUE or FALSE).
(c) The method is the exact translation of the mental process that expert operators follow when assessing the "danger" that a fault actually may happen.

In our implementation, we have always adopted a linear combination, in which the weights are the very same percentual variations of ΔI_k. With this reasoning, a failure is no longer deterministically certain as soon as its fault chains are active: the diagnostic IA, after detecting the existence of one or more fault chains, will display to the plant operator the likelihood (between 0 and 1) of an immediate failure event. To be able to intervene, IA must also be able to de-fuzzify its calculation. In our case, we have enacted this capability by a set of rules that assign a threshold value to each fuzzy failure: if the failure is, say, 85% certain, it is assumed to "happen". Lowering this threshold makes the diagnosis more "cautious" (it may signal as failures many events that are not), while increasing it makes it more precise but also more risky (it may not signal an actually detected failure event).

3.3.4. Filters

A direct implementation of the IA endowed with the fuzzy logic protocol would result in a failure. The reasons are of different order, and are all related to the problem of the "quality" of the acquired data. In this section we provide a simplified description of the possible faults in the data acquisition protocol and discuss some remedial actions.

(1) The first and most obvious problem is that the data may be non-exact, i.e., affected by systematic and/or casual errors in the data acquisition system.
(2) A second problem is noise, which also can be random or systematic.
(3) Statistical independence must also be assured: there must not be spurious (i.e., system-induced) cross-correlations in the signals.
(4) The data must be statistically relevant: there must be a correlation equal (or very near) to 1 between the measurable and the generated signal.
(5) For rapidly varying signals, aliasing must be absent.

These five problems pertain to the field of data acquisition system design, and we shall not delve into them here: in other words, we shall assume that our data are exact, congruent, statistically independent and relevant, noiseless and devoid of aliasing (or aliasing-corrected).

There is though a higher level at which consistency of the input signal must be checked: this is the system (or process) level, which implies a deeper "knowledge" of the physical details of the process. To better analyse this aspect of the problem, let us first construct a *data influence matrix* DIM, defined as follows:

- $\mathrm{DIM}_{i,j} = 0$ if a variation of the ith measurable is known not to affect the jth indicator.
- $\mathrm{DIM}_{i,j} = 1$ if a variation of the ith measurable is known to affect the jth indicator.
- $\mathrm{DIM}_{i,j} = f$. where $f\{f \subset \Re,\ 0 < f < 1\}$ is the degree of correlation of the ith measurable and the jth indicator.

A first useful filter for this higher-level monitoring action is then the following filtering rule #1: if IA detects a variation of an indicator I_j for which $\mathrm{DIM}_{i,j}$ is "high" (here, a threshold $f > 0.7$ has been imposed) and simultaneously the ith measurable does not display a significant variation, a data-checking procedure is activated and the plant operator is alerted of possible data incongruency.

Let us further consider another matrix, the *Indicator Correlation Matrix,* ICM, defined as follows:

- $\mathrm{ICM}_{i,j} = g$ (rational number between 0 and 1) where g is the degree of correlation between the ith and the jth indicator.

A second filter can then be defined, by filtering rule #2: if IA detects a simultaneous variation of two indicators, I_i and I_j, whose correlation g in ICM is lower than a pre-set limit (again, $g < 0.7$ has been chosen here), then the corresponding fault chains are disregarded, and the plant operator is alerted of the possible mismatch.

Since both DIM and ICM are constructed on the basis of empirical knowledge (gathered either by numerical or by physical experiments), the position of these two constraints is a rather simple way of introducing higher-level knowledge into the IA.

3.3.5. Diagnosis vs. prognosis

If one considers the logical flowchart of the diagnostic and prognostic procedures (Sections 2.1 and 2.2 above), it appears clearly that *the prognostic activity is really a projection of the diagnostic one*: in more precise terms, one can say that the prognostic is a diagnostic action performed not on the real state $\mathbf{S}_t$ of the system, but on a virtual one representing IA's estimate of a future state $\mathbf{S}_{t+\Delta t}$. This virtual state is constructed on the basis of the present state (as an initial condition) and of the foreseen operative conditions (as boundary conditions and independently imposed external constraints). Thus, there is no need to implement a separate prognostic paradigm: the same logic can be used on the two states, the present one and the future, "virtual" one. This represents a substantial simplification to the structure of the IA, and makes its implementation much easier. Examples of external constraints are the ambient T and p, the operating load, etc.

4. A practical application

4.1. The "ICARO" cogeneration plant at the ENEA-CASACCIA site

The energy conversion process for which the expert diagnostic/prognostic tool was developed is an experimental facility on the grounds of the ENEA Laboratories of Casaccia, near Roma, consisting of cogeneration plant called ICARO (Fig. 2) based on a GE-Nuovo Pignone PTG-2 turbogas set with 2 MW of nameplate power and on a heat recovery boiler HRB that feeds the heating system of the compound. The thermal load is 5 MW without afterburner and 7 MW with afterburner. The process operates on natural gas, but its adaptation to H_2-operation is underway. In 1999/2000, ICARO generated slightly over 4.5 electrical GWh and recovered a total of about 30 thermal GJ [5]. The design electrical efficiency amounts to 0.36, but the field-measured value was substantially lower, about 0.23 [1]. The actual first- and second law efficiencies of the cogeneration unit, measured over a 12-month period, amounted to 0.68 and 0.37, respectively [1,2]. After start-up, ICARO performed satisfactorily indicating that both the monitoring and the control systems were working properly.

4.2. Definition of the reference conditions

To construct "performance indicators" on which the derangement from "standard operative conditions" is measured, it is necessary to accurately and completely define such reference conditions. The plant operating manual and the design specifications provided by the designer and by the constructor define only a very limited set of operating points. We must perforce recur to some form of "logical extrapolation" based on an intelligent comparison between the measured data and a set of proper theoretical operating curves. On the basis of our experimental findings we can construct by interpolation the response of the plant to variations of the operative conditions. It is possible to enhance the usefulness of this computational process, and its efficiency in practical terms, by considering as "experimental data" also the results of a properly calibrated plant simulator. For ICARO, a dynamic simulator was available [4,6], and we treated its output as an equivalent source of knowledge as the plant log-sheets, thus substantially expanding our database. At this point, we can describe the acquisition phase of the expert system:

(a) From the plant log-sheets or from the plant simulator PROMISE computes a projection of the expected variation of performance caused by an incremental variation of each relevant parameter.
(b) PROMISE constructs then a "discrete total differential" that represents its estimate of the plant performance under the given operational data.

Now, this approximation becomes more accurate if we increase the number of measurables and decrease the scanned experimental interval for each measurable. We decided to use a limited number of indicators (29 see Table 2), and to impose relatively small bounds to the variations of each indicator: this is clearly a compromise between accuracy and computational effort, and can be modified in future applications.

Table 3
Failure detection criteria (for the interpretation of the attributions "high" and "low", see Sections 3.3.3 and 3.3.4)

Component	Type of failure	Indicator-based causal chain
Filter	Fouling	I_1 high $\cup$ m_2 low
Compressor	Fouling	(I_3 low $\cup$ I_4 low) $\cap$ (I_3 low $\cup$ I_2 high)
	Malfunctioning	I_2 high $\cup$ W_{compr} high
	Choking	(I_4 low $\cup$ I_5 high) $\cap$ (I_4 low $\cup$ T_3 low)
	Excessive T_3	I_2 high
	Stall	I_4 low $\cup$ I_5 low
Combustion chamber	Fouling	I_6 high
	CH_4/H_2O valve malfunctioning	(I_7 high $\cup$ I_8 constant) $\cap$ (I_7 constant $\cup$ I_8 high)
CH_4 injector	Fouling	I_8 high
Electrical generator	Balance fault	$I_{18} \neq 1$
By-pass stack	Emissions	$I_9 > 1 \cup I_{10} > 1$
	Heat losses	I_{11} high
Boiler stack	Emissions	$I_{12} > 1 \cup I_{13} > 1$
	Heat losses	I_{14} high
Turbine	Excessive T_4	I_{16} high
	Fouling	(I_{15} low $\cup$ T_5 high) $\cap$ (I_{15} low $\cup$ p_5 high)
	Choking	I_{17} high $\cup$ p_4 high
Afterburner	Fouling	I_{19} low $\cup$ I_{20} low
	Losses	I_{19} low
Primary heat exchanger	Fouling	I_{21} high $\cup$ I_{22} low
	Mass leakage	I_{21} high
Secondary heat exchanger	Fouling	I_{23} low
Primary loop	Fouling	I_{24} high
Secondary loop	Fouling	I_{25} high
Main pump	Cavitation	I_{26} low $\cup$ I_{27} low
	Malfunctioning	(I_{26} high $\cup$ I_{27} low) $\cap$ (I_{26} low $\cup$ I_{27} high)
Heat recovery boiler	Fouling	I_{28} low
Shaft (vibrations)	Unbalance, wear	$I_{29,i} \neq 1$

4.3. The proposed Indicators and the failure detection criteria

Table 3 lists the "failure detection criteria" adopted in PROMISE. The list has been complied on the basis of a proper combination of expert advice, experimental evidence and analysis of computational results. Table 3 represents thus already a meta-level knowledge: in fact, it is a simple task to derive from its entries an operative flow chart for the ES. Notice that all measurables listed in both Tables 2 and 3 are based on "primary" quantities, already monitored by the existing PLC-based control system. PROMISE applies two different strategies for the detection of "sudden failures" and of "creeping failures, as described here below.

4.3.1. "Sudden" failures

In this case, the meta-rule is straightforward: if one fault chain is detected (third column of Table 3), a message is immediately sent to the operator, with an appropriate warning message and, if possible, with some suggestions on the strategy to take. The fault chain is "interpreted"

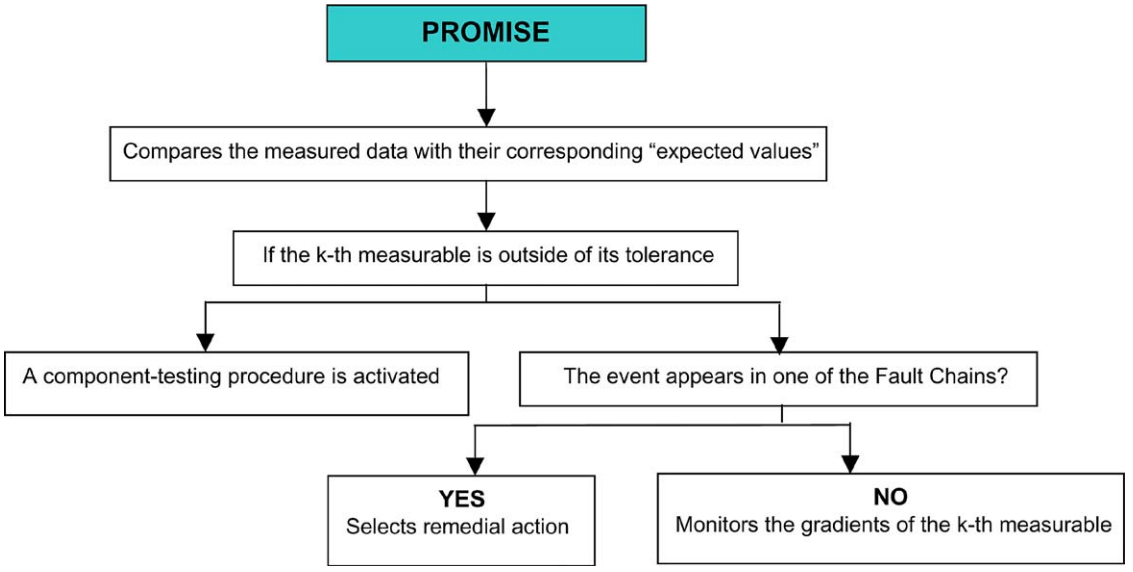

Fig. 4. Logical flow-chart of the PROMISE package.

in a fuzzy sense (Section 3.3.3): a "fault signature" is assumed to exist not when the values of the indicators are exactly within the respective fault range, but when there is a (fuzzy) likelihood that they are "near" that range.

4.3.2. Creeping failures

We talk of a "creeping fault" when the instantaneous signature S_j shows a clear tendency to approach the failure range. In other words, if single indicator varies in time in such a way that at a time $t_0 + \tau$ in the future its value will exceed its threshold value, then PROMISE assumes that there is the possibility that, at time $t_0 + \tau$ a fault event will be realised (the same applies for a chain consisting of more than one indicator). In mathematical terms, if for a certain S_j, $\left|dS_j/dt\right|$ appears to be "abnormally growing" with respect to a limit value, then a warning is displayed. Obviously, we need a higher-level knowledge to establish both the limit value of the signature and the "excessive" value of its time derivative. In either case, we again adopt a fuzzy approach. It is important to remark that the results of each prognostic step are used by the IA to "re-assess" its diagnostics at the successive time step: the idea being that, if the trend displayed by a certain indicator I_k is " safe", then the fuzzy likelihood of a sudden fault is decreased by a heuristic factor depending on that trend. This criterion is obviously not theoretically justifiable, and its implementation requires some prior knowledge of the probability statistics of each fault chain, but its implementation for the simple cases examined here gave encouraging results (Fig. 4).

4.4. Some remarks on shell-dependency

The present version of PROMISE has been implemented under the G2® shell, and it is currently available only in this format. To extend its portability, we are working on an ACCESS® version of the code. As all AI codes, PROMISE is of quite limited portability, and its syntacti-

cal structure (its source-code) is strongly dependent on the environment within which it has been developed. Even the graphical interface is at present hard-linked to the G2 shell, but this is inessential, as the appearance of the screen displays can be reproduced under any commercial operative system.

5. Conclusions

When we undertook the development of a diagnostic and prognostic tool for plant intelligent health monitoring [1], we had four goals in mind:

1. To acquire a sufficient amount of knowledge (in the design-, operation- and plant management domain) to compile a list of design-, operation- and management rules an artificial plant manager could use to control the plant under a broad operative range.
2. To classify the acquired knowledge, possibly clustering it in separate but interrelated knowledge areas.
3. To implement, test and train a prototype knowledge-based IA working under the rules defined above.
4. To field test an α-version of such an IA on a proper set of actual plant data.

All goals have been achieved. Broadly (even if not exactly) speaking, we can classify the activities performed in the course of the study into: knowledge acquisition and codification, training and calibrating of the code, and field validation: this classification is rather sharp and does not capture the vast amount of overlapping that characterised the single tasks, but provides a good global framework that encompasses the entire activity.

5.1. Training and calibration

PROMISE was field-tested on the very same plant for which it was designed. This testing started with a series of calibration runs, in each one of which the value of one of the indicators (or more, if requested by the specific fault chain) was (were) varied manually, so that the IA would "read" a time-series of "measured data" which we knew would lead to a failure in one or more components. The code performed satisfactorily, always diagnosing the correct fault and never misdiagnosing. Additionally, in few cases it was possible to insert artificial time-sequences that would indicate a slow derangement of one or more measurables from their nominal values: in all cases, PROMISE predicted the correct "creeping fault", thus performing its prognostics correctly. Once the code had been thus satisfactorily tested, three fault conditions were chosen and the respective "logs" were submitted to the IA:

(A) Abnormal increase of the air filter pressure drop. The increase amounted to 30% of the nominal value within two successive days (48 h) with a linear behaviour ("ramp").
(B) Abnormal increase of the gas temperature at combustion chamber outlet. The increase amounted to about 10% of the nominal value within 10 min, again with a linear behaviour.

(C) Abnormal decrease of the main pump flow rate. The decrease amounted to 30% of the nominal value within 10 min. A linear behaviour was again specified.

For all of these "faults" there were no experimental data: the "artificial" process log was created using the plant simulator. The results are presented in [1,2] and can be synthetically described as follows:

Fault (A) Expected *action*: starting from nominal conditions, the prognosis of "filter fouling" ought to be activated as soon as the measured value in input reaches 105% of the design value.

Actual action: during the first 2 min of the run, both diagnosis and prognosis of the filter were active. Starting from the third minute, PROMISE calculated the indicators' trend using the previous values. After 480 min the prognosis was interrupted, and a "filter fouling" warning message was activated.

Fault (B) Expected *action*: starting from nominal condition, the prognosis of "turbine excessive inlet temperature" ought to be activated as soon as the measured value in input reaches 105% of the design value.

Actual action: during the first 2 min of the run, both diagnosis and prognosis of the turbine were active. Starting from the third minute, the indicators' trends were computed. After 5 min the prognosis was interrupted, and a "turbine excessive temperature" warning message was activated.

Fault (C) *Expected action*: starting from nominal condition, either the prognosis of "main pump cavitation" or that of "primary loop fouling" ought to be activated (because PROMISE also has access to additional information that allows it to distinguish between the two failures) as soon as the measured value in input decreases below 70% of the design value.

Actual action: during the first 2 min of the run, both diagnosis and prognosis of the turbine were active. Starting from the third minute, the indicators' trends were computed. After 11.7 min, the prognosis was interrupted, and a "main pump cavitation" warning message was activated.

5.2. Future developments

Though entirely satisfactory, these results must be considered as preliminary. The knowledge imparted to the code is, in fact, quite elementary, and most, if not all, of the complications related to signal acquisition and processing have been by-passed by proper choice of the operative situations the code has been called to work on. No data analysis has been included: all data have been assumed to be exact, congruent, statistically relevant and independent, and physically relevant (Section 3.3.4). A real Intelligent Health Monitor must possess some capability to screen the data and perform a preliminary data congruency analysis. This is at present negated by the very structure of our IA: it is likely that an independent expert system must be specifically implemented for this task, and that its "filtered" data must then be analysed by PROMISE. To this extent, new (logically and hierarchically different) knowledge must be acquired,

classified and imparted to the expert assistant. This is a lengthy and complex task, which requires a substantial investment of resources for its completion.

References

[1] Biagetti T, Sciubba E. PROMISE: a tool for the on-line intelligent performance prediction of cogeneration plants. Proceedings of the ECOS02-Berlin, July. 2002, p. 463–71.
[2] Biagetti T, Sciubba E. A first step towards unmanned intelligent process management: a procedure for the diagnostics and prognostics of energy conversion plants. IJAT, April 2002:85–99.
[3] Can Gülen S, et al. Real time on-line performance diagnostics of heavy-duty industrial gas turbines. ASME paper 2000-GT-312, 2000.
[4] Casella FC, Maffezzoni FC. Dynamic Simulation of a gas turbine-based cogeneration plant. Proceedings of the X TESEC-Genoa, June (in Italian) 2001.
[5] Crescenzi T, Mazzei D. Analysis of the operating data of the energy generation system in the Casaccia Research Centre. ENEA Technical Report, July 2000 (in Italian).
[6] DOE Fossil Energy Program. Tomorrow's Turbines, 2001. See also: http://www.fe.goe.gov.
[7] Forsyth G, Delaney J. Designing diagnostics expert systems for long-term supportability. ASME paper 2000-GT-31, 2000.
[8] Ozgur D, et al.. Remote monitoring and diagnostics system for GE heavy-duty gas turbines. ASME paper 2000-GT-314, 2000.
[9] Roemer MJ, Kacprzynski GJ. Advanced diagnostics and prognostics for gas turbine engine risk assessment. ASME paper 2000-GT-30, 2000.
[10] Sciubba E, Melli R. Artificial intelligence in thermal systems design: concepts and applications. New York: Nova Science; 1998.
[11] Sriram RD. Intelligent systems for engineering. New York: Springer; 1997.
[12] Tsalavoutas A, et al. Combining advanced data analysis methods for the constitution of an integrated gas turbine condition monitoring and diagnostic system. ASME paper 2000-GT-34, 2000.