Int.J. Applied Thermodynamics,
Vol.5 (No.2), pp.85-99, June-2002

ISSN 1301-9724

A First Step Towards Unmanned Intelligent Process Management: A Procedure for the Diagnostics and Prognostics of Energy Conversion Plants

T. BIAGETTI, E. SCIUBBA*
Mechanical Engineering Dept., University of Roma 1 - Italy

Abstract

This paper describes the conceptual development and the prototype implementation of an Expert System that deals with the prognosis and diagnosis of a significant subset of the operative faults of a cogeneration power plant. The expert system receives both raw and organised real-time information on the "instantaneous" (actually, averaged over 60-seconds intervals) plant operating conditions from a non-intelligent "plant interface" consisting of a standard plant data acquisition system and of a specific plant simulation software. This solution was adopted in view of possible future applications to industrial plants, where a low number of intrusive sensors is desirable: in this case, the simulator provides the missing data.

The Inference Engine operates on the basis of a set of pre-defined rules that seek possible "faults chains" expressed as combinations of a pre-assigned number of continuously calculated performance indicators, like the air filter output pressure drop, the relative compressor, combustion and turbine efficiency decay with respect to their respective nameplate values, the compressor enthalpy gain and surge conditions, the pollutants concentration in flue gases, the relative pressure losses in both the primary and secondary water circuit, the TTD of the Heat Recovery Boiler, etc.: the complete list includes 29 indices. The rules establish whether a component's behaviour is degraded by examining both the individual indicators and all of their relevant combinations. A graphic window displays a series of icons, one for each indicator, with a refresh rate of one real-time minute. The Expert System enables the user to determine in detail the instantaneous performance conditions. If performance deterioration is detected, it sends a message to the operator and provides some decision support via a customised graphic interface.

The structure of the code is Object Oriented, and each component as well as each flux are represented as the *instance of a class*. Both the reasoning and the controlling actions are taken in the same O-O environment.

The present paper presents the organization of an ES whose prototype version has been nicknamed PROMISE, from the Italian acronym for PROgnostic and Intelligent Monitoring Expert System, defines its goals and discusses both the results of the tests conducted so far and the implications for future applied research in this field.

Key words: Unmanned Intelligent Process Management, Expert System, diagnosis, cogeneration

1. Introduction

It is perhaps not unnecessary to begin this paper by recalling that the process of designing, constructing and operating energy conversion systems is a very complex task. If all three steps are properly implemented, the plant delivers the desired final form of energy in the prescribed amount, operates within the admissible emission limits and performs efficiently at the budgeted cost. If any part of this process breaks down, the plant fails to deliver these benefits. The monetary

* Author to whom all correspondence should be addressed.

losses originated by such failures are important, but even more important is the resource destruction that results in the end (because the reduced productivity must be "made up" by some alternative generation system, so that the "resource effectiveness" of the process is decreased). In view of this, design-and-optimisation methods have been developed that take into due account variable load conditions (Frangopoulos 1990, Munoz and Spakovsky 2000), availability losses due to scheduled and unscheduled maintenance (Frangopoulos 1990) and performance degradation due to wear and fouling of the equipment (Zaleta-Aguilar et al., 2001,). When we consider field conditions, though, we find that the physical operation of the plant is still largely non-automated, in the sense that all non-routine control activities are performed solely by human operators. The "self-diagnosis" enacted by current control systems is very rudimentary, and in fact they are intentionally so designed that the "intelligence" in their responses be zero, because the design philosophy is that a control system must act fast, safely and deterministically, always guided by a "linear" logic protocol to avoid possible misinterpretations on the part of the human plant operator. This does not mean that such control systems are technologically primitive: on the contrary, they usually conjugate the high reliability of fully integrated electronic chips with custom-designed mechanical/electrical sensors and actuators, and are capable of "reading" the instantaneous state of the process and of actively operating to keep this state in line with the pre-set operating point. The starting point of our research is that we ought to go beyond the idea of "leaving all logically non-linear decisions to the human operator", and extend to this field some of the well-proven methods of Artificial Intelligence.

The aim of a "Process Management Protocol" is that of maximising instantaneous power output, burning less fuel, emitting less pollutants, etc., which can be synthetically described as constraining the plant to always maintain a state corresponding to the "best" operating point over a wider range of conditions. To facilitate the practical conduction of the plant, these protocols are also requested to be capable of self-diagnosing their own failures and to provide data to support manual process diagnoses.

Intelligent Process Management Tools (IPMTs) have been conceived to go two (or three) steps further: they are capable of generating an *intelligent diagnosis* of the present state of the plant (and therefore are sometimes called "Health Monitoring Systems"), but must also enact a *prognostic action*, making intelligent estimates of the future state of the plant under the foreseen boundary conditions. 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 human operators with very valuable information about the "optimal" operating curve of the plant in some future period T. The practical implementation of IPMTs will no doubt require some modifications in present design procedures, especially for what sizing and physical assembly of equipment are concerned. It has been estimated (Melli 2001) that by the year 2010, Energy Systems design tools will contain hypertext guides to provide designers with information in the early phases of the design process, sharing both physical and logical data with other databases, including "older" (i.e., present at the time of this writing!) ones. Such tools will include robust analysis capabilities for the modelling of the operation of the plant to be designed, including control system malfunctions and equipment degradation and failure. A limited set of analytic tools that autonomously analyse design performance is in fact presently available (Papalambros and Wilde 1988). A survey of the existing advanced design tools (Privette 1997) shows that the trend is clearly that of implementing "Design Environments" that are fully integrated with CAD systems, that possess analytic capabilities to predict thermal performance and mechanical and auxiliary systems integrity and safety, and can elaborate "raw" data to generate output that can be shared with IPMTs.

Further development of such Design Environments is hoped to support all aspects of design and provide rapid analysis, design suggestions, unambiguous real-time data interpretation, and automatic generation of all design documents. The IPMTs, on their part, will include start-up support, normal operation control and diagnosis, maintenance scheduling, and extended prognostics.

The present paper describes the conceptual development and the actual field implementation of a Diagnostic and Prognostic tool been specifically designed for a gas turbine based cogeneration system.

2. The "Icaro" Cogeneration Plant of Enea-Casaccia

The system for which the Expert Diagnostic/Prognostic tool was developed is an experimental facility on the grounds of the ENEA Laboratories of Casaccia, near Roma, Italy, and consists of a gas turbine-based cogeneration plant called ICARO (*Figure 1*). The turbogas is a GE-Nuovo Pignone PTG-2 turbogas set with 2 MW of nameplate power, and the heat recovery boiler HRB (manufactured by CEFLA) feeds the centralised heating system of the compound.

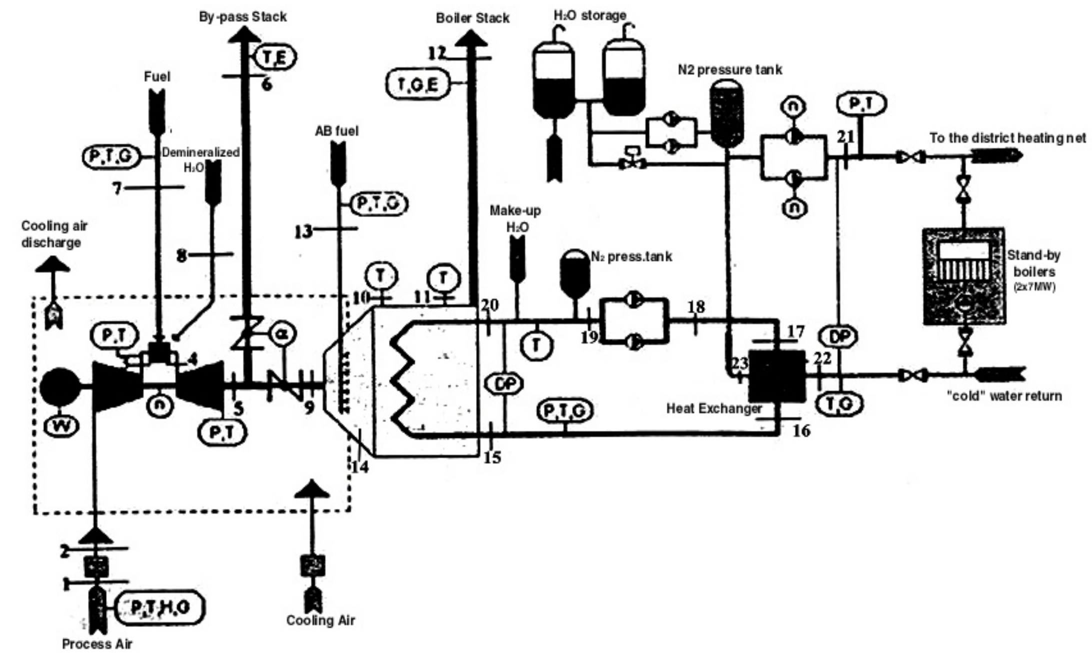

Figure 1. ICARO power plant layout

The thermal load is 5 MW without afterburner and 7 MW with afterburner. The process presently operates on natural gas. In the year 1999/2000, ICARO generated slightly over 4.5 electrical GWh and recovered a total of about 30 thermal GJ (Gülen et al., 2000), with a design electrical efficiency of 0.36. The actual first- and second Law efficiencies of the cogeneration unit, averaged over a 12-months period, amounted to:

$$\eta_{cog,I} = \frac{W_{el} + Q_{rec}}{Q_f} \cong 0.68 \qquad (1)$$

$$\eta_{cog,II} = \frac{W_{el} + E_{Q_{rec}}}{E_{Q_f}} \cong 0.37 \qquad (2)$$

where Q_{rec} is the recovered thermal energy and E indicates an exergy flux. In 2), the exergetic content of the fuel has been assumed to be equal to its lower heating value (45000 kJ/kg), while the exergy of the recovered heat has been computed using an exergetic factor $\alpha = 1-(T_o/T_{rec})$, with $T_o = 288$ K and $T_{rec} = 416$ K. ICARO performed satisfactorily for almost two years, under a wide range of operating conditions, indicating that both the monitoring and the control systems were working properly.

3. Design Specifications for the Expert System

The specific objective of our research project was that of realising an Expert System endowed with both Diagnostics and Prognostic capabilities. Though the specifications called for an immediate application on the ENEA-ICARO cogeneration plant, we decided from the very beginning to place this project in a broader perspective: our work then is in fact a first step towards the implementation of a general IPMT.

The usual design goal of any type of equipment is the minimisation of the resource "consumed" to generate the "product" (material or immaterial) for which the plant is being designed. Restricting our considerations to energy conversion processes, we define a plant availability factor

$$PF = \frac{\text{total equivalent full}}{\text{total number of hours in the year}} - \frac{\text{load operating hours in a year}}{\text{total number of hours in the year}}$$

It is apparent that no energy conversion plant can operate with a 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.

Notice that we separately account here 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.

a) Plant shutdowns due to scheduled maintenance

These are usually related to design problems. In recent years, the MTBF of TG plants has been increased to over 20000 hours (for large heavy duty plants, see Anonymous Staff Writer 1994), with some critical components reaching the 100000 hours.

b) Plant shutdowns due to unscheduled maintenance

Improvements in this area are mainly due to systematic application of maintenance-scheduling techniques (early replacement, on-line monitoring) that call for an anticipated replacement of sensitive equipment performed during scheduled maintenance interventions. Design procedures have also brought an improvement by taking into account possible failures and introducing redundancies, or modifying the configuration of critically sensitive components. As a result, failures due to "wear" of a component have been substantially reduced.

c) Plant shutdowns due to sudden failures.

This is the type of failures of our concern. 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 phenomena "local" to a component that lead to a small but detectable deterioration of its performance. Our efforts may thus be redirected to the early detection of these "performance degradation"-warning signals. The method we propose is indeed an exact transposition to the Artificial Intelligence domain of the activities performed at present by use of "human intelligence": a sufficient number of critical points of 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 is detected, the operator, together 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.

4. The Logic of the Diagnostic/Prognostic System

4.1. The knowledge base (KB)

The KB of the Expert System was designed so that it would be exactly and completely representative of the universe of information and rules within which the human operator ("HO") performs. Its logical format is though somewhat distant from its human counterpart, because it is known from the theory of Artificial Intelligence (Melli 2001, Papalambros and Wilde 1988) that a proper "systematisation" (or logical "pre-conditioning") is mandatory in the construction of a KB. Such a systematisation is performed during the Knowledge Acquisition phase and is aimed at making the resulting KB as logically and accessible as possible. Many of the rules on which HO base their decisions are composite or fuzzy rules, and therefore this manipulation in the early phases of the acquisition is inevitable. The following seven meta-principles were adopted here (list adapted from Biagetti et al., 2002):

a) *The number of possible failures is finite*, and each failure depends at least on a specific "*fault signature*", i.e., on a unique combination of the values of some relevant process parameters;

b) *There are no "sudden" failures*: a "failure" may be logically described as a process of migration of the operative point of the process towards a specific "fault attractor" represented by the corresponding signature;

c) Each one of these migrations takes place over a *characteristic time period* that is a function both of the failing component and of the type of failure;

d) *Failure signatures may be non-local*, in the sense that some of the measurable quantities in points of the plant physically "far" from the component which is experiencing a failure event may display abnormal effects as a consequence of that failure: in these cases we say that the measurables (and the respective indicators) are correlated;

e) A convenient representation of the operating point of the process may be based on a set of *dimensionless indicators*, each defined as the ratio between the actual value of a relevant measurable quantity (temperature, pressure, stress level, frequency, etc.) and the dynamic design value of the same measurable;

f) *Any "failure event" is described by at least one chain of well-defined physical causes.* Only in this sense failures are logically deterministic events;

g) *All failure chains are by force fuzzy*. There are two reasons that justify this rather strong statement. First, the causes are necessary but not sufficient for the failure to happen in practice (e.g., for a first-row statoric blade to fail under thermal fatigue, it is necessary that the impinging gas temperature exceeds a given limit, but it is not true that for a gas temperature slightly above this limit the failure is certain). Second, even the necessity is by force vague (the blade failure may happen even if the gas temperature is decidedly lower than the design limit);

h) Some of the above defined causal chains may be *concurrent*: the same failure may thus be caused by each one of two (or more) chains, or by a certain combination of them.

4.1.1. Knowledge acquisition and classification

This phase was lengthy and difficult. A large number of basic engineering data, not provided by the Plant Designer and by the Plant Constructor, were necessarily substituted by reasonable assumptions. There was though absolutely no problem in the acquisition and systematisation of the plant operative log: we had complete access to 6 months of full operative data, both raw and pre-filtered by the existing control system.

After some initial problems, the custom-made Process Simulator became an irreplaceable source of process data for our experiments. It must be stressed that the simulator does not "replace" the experimentally acquired data, but rather complements them, within the limits of its own simulation capabilities.

The Acquisition/Classification phase considered three main sources:

- For Process Engineering and Mechanical Plant Engineering, direct interviews to Domain Experts, both internal and external to our Research Group;
- For the definition of the Fault Signatures, an accurate study of the relevant technical literature;
- For the actual implementation and the organisation of the ES, we followed a methodology already defined, refined and calibrated in our own previous works (Papalambros and Wilde 1988).

To conveniently classify the acquired Knowledge, we followed a "*bottom-up*" method, starting from the specific failure and trying to establish a general inductive diagnostic procedure. At the risk of oversimplification, it can be said that the Health Monitoring proceeds according to the following three steps:

a) Identification of a sufficient number of proper performance indicators that can fully identify the operating point of the plant (Section 4.1.4);

b) Definition of the fundamental criteria to use in the identification of both "sudden" and "creeping" faults (Sections 4.1.4.1 and 4.1.4.2);

c) Definition of a complete procedure of fault Diagnostics and Prognostics (Sections 4.3 e 4.4).

4.1.2. Knowledge formalisation

A large database exists on TG monitoring. There are international standards (ASME, ISO) that define, both for design and contractual purposes, the number and type of the measurables and the data acquisition procedures. We approached the problem though with a different angle: we are not interested in detecting whether the system is operating within the contractually defined "design" or "off-design" modes, but in monitoring whether the system's operating point is defined by a signature that corresponds to a "correct" or "allowed" state or to an "anomalous" or "failure" state. Therefore, we must include in our analysis a larger number of measurables than those defined -for instance- by standard test protocols. There are several "lists of measurables" in the technical literature (Forsyth and Delaney, 2000, Melli, 2001, Ozgur et. al., 2000), and we considered them as one of the possible expert sources from which to derive our knowledge. Other sources were the systems Designer, the Plant Operator, and Turbomachinery Experts (Biagetti et al., 2002). For contingent reasons, which seem to be of rather general character, see (Sriram, 1997), the Knowledge Acquisition phase of the project was plagued by inconsistencies, logistic problems and scarce co-operation on the part of some of the Domain Experts, but in the end the formalisation of the acquired knowledge was performed as described here below.

4.1.3. Definition of the rreference conditions

To construct "performance indicators" that measure the derangement from "standard operative conditions", 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. If a general "performance function", $f(x_i)$, were known, we could use it to describe the instantaneous performance of the plant as a function of a certain number *N* of process parameters that we call "measurables": x_i (x = p, T, m, w etc.; i=1,2…N). Consequently, the variation in the performance of the plant corresponding to any variation of any subset of the measurables would also be immediately computable:

$$\frac{\partial f}{\partial x_j} = p(x_j) \ \ (j = 1,2,..N)$$

$$\frac{df}{dx} = p(x_1)dx_1 + p(x_2)dx_2 + .. + p(x_N)dx_N \quad (3)$$

The problem is that, obviously, the explicit form of *f* is known only for ideal and extremely simplified processes, of no practical relevance. But in the specific case of a cogeneration plant, we are only interested in the electrical (P) and the thermal (Q) power outputs. Therefore, an empirical approximation to f valid for our purposes may be experimentally determined by computing the sensitivity of P and Q to variations of the process parameters:

$$\frac{\Delta P_{el}}{\Delta T_{air}} \cong \frac{\partial f_1}{\partial T_{air}} \Rightarrow \text{Measured} \quad (4)$$

$$\frac{\Delta Q_{thermal\ \ power}}{\Delta T_{air}} \cong \frac{\partial f_2}{\partial T_{air}} \Rightarrow \text{Measured} \quad (5)$$

$$\frac{\Delta P_{el}}{\Delta p_{air\ \ filter}} \cong \frac{\partial f_1}{\partial p_{filter}} \Rightarrow \text{Measured} \quad (6)$$

$$\frac{\Delta \eta_{tot}}{\Delta T_{air}} \cong \left(\frac{\partial(f_1 + f_2)}{\partial T_{air}}\right)_{constant\ \ fuel\ \ flowrate} \Rightarrow \text{Measured} \quad (7)$$

etc.

On the basis of a field measurement campaign, we can thus re-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 to reduce its cost in terms of resource intensity, by considering as "experimental data" also the results of a properly calibrated plant simulator. For ICARO, a dynamic simulator was available Biagetti et al. (2002) and we treated its output as an equivalent source of knowledge to the plant log-sheets, thus substantially expanding our database. On the basis of the above, we can now describe the acquisition phase of the Expert System:

a) From the plant log-sheets or from the plant simulator PROMISE computes the discrete partial differentials:

$$\frac{\Delta f}{\Delta x_j} \cong \frac{\partial f_1}{\partial x_j} = g(x_j) \quad (8)$$

where g is an interpolating polynomial and j = 1…N. Recall that we are not interested in the exact form of the performance function f, but only to its two "components" $f_P = f_1$ and $f_Q = f_2$;

b) PROMISE constructs then a "discrete total differential" of f with respect to all relevant process parameters:

$$\left(\frac{\Delta f}{\Delta x}\right)_{tot} \cong \frac{d(f_1 + f_2)}{dx} \quad (9)$$

The approximation expressed by equation 9 is improved if we increase the number N of measurables "x_i" and decrease the scanned experimental interval for each Δx_i. In practical applications, one must consider that both the conceptual complexity (the interpretation of the results) and the computational intensity increase as $N/\Delta x_i$. In the end, we decided to use a limited number of indicators (29), and to impose relatively small bounds to the variations of each indicator: this is clearly a compromise between accuracy and computational effort, and is strongly application dependent.

4.1.4. The proposed indicators and the failure detection criteria

TABLE I contains the list of the faults that were included in this version of the ES. The list (which constitutes a part of the Knowledge Base) was constructed by trial-and-error, working in close cooperation with the Plant Manager.

TABLE II reports a list of the indicators, each one defined as a function of process measurables: the suffixes refer to the numbering of *Figure 1*.

Finally, TABLE III lists the adopted "failure detection criteria". The list has been compiled on the basis of a proper combination of expert advice, experimental evidence and analysis of computational results. TABLE III 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 Inference Engine of the Expert System. Notice that all measurables listed in both TABLE II and III are based on "primary" quantities, already monitored by the existing PLC-based control system.

The detection of "sudden failures" and of "creeping failures" seems to require different search strategies: we found though a meta-level "logical similarity" between the two operations, and could therefore implement a very efficient modular procedure (this is an example of a logical embedding procedure, see (Privette 1997)).

4.1.4.1. "Sudden" failures

In this case, the meta-rule is straightforward: if one Fault Signature is detected (third column of TABLE III), a message is immediately sent to the Operator, together with an appropriate warning message and some suggestions on the corrective strategy. Notice that the Fault Signature

TABLE I. LIST OF POSSIBLE FAULTS INCLUDED IN THE KNOWLEDGE BASE

COMPONENT	POSSIBLE FAULT(S)	COMPONENT	POSSIBLE FAULT(S)
Filter	Mass leaks 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 Mass leaks	Heat Recovery Boiler	Fouling
Boiler Stack	Secondary combustion reactions Fouling Mass leaks	Main Shaft	Near-critical vibration frequencies
Turbine	Fouling Choking Excessive inlet temperature	Afterburner	CH_4 injector fouling
Primary Heat Exchanger	Fouling		

TABLE II. THE PROPOSED INDICATORS

AIR FILTER	$I_1 = (p_1\text{-}p_2)/(p_1\text{-}p_2)_d$
COMPRESSOR	$I_2 = c_{p23}\ (T_3\text{-}T_2)/[c_{p23}\ (T_3\text{-}T_2)]_d$
	$I_3 = \eta_C/(\eta_C)_d$
	$I_4 = \beta_C/(\beta_C)_d = p_3/p_{3d}$
	$I_5 = m_3/(m_3)_d$
COMBUSTION CHAMBER	$I_6 = \Delta p_{cc}/\Delta(p_{cc})_d = (p_3\text{-}p_4)/(p_3\text{-}p_4)_d$
	$I_7 = m_8/(m_8)_d$
FUEL INJECTOR	$I_8 = m_7/(m_7)_d$
BY-PASS STACK	$I_9 = (x_{NOX,6})/(x_{NOX,6})_d$
	$I_{10} = (x_{CO,6})/(x_{CO,6})_d$
	$I_{11} = (T_5\text{-}T_6)/(T_5\text{-}T_6)_d$
BOILER MAIN STACK	$I_{12} = (x_{NOX,12})/(x_{NOX,12})_d$
	$I_{13} = (x_{CO,12})/(x_{CO,12})_d$
	$I_{14} = (T_{11}\text{-}T_{12})/(T_{11}\text{-}T_{12})_d$
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	$I_{19} = (T_{10}\text{-}T_9)/(T_{10}\text{-}T_9)_d$
	$I_{20} = m_{13}/(m_{13})_d$
H_2O/H_2O HEAT EXCHANGER, shell side	$I_{21} = (p_{16}\text{-}p_{17})/(p_{16}\text{-}p_{17})_d$
	$I_{22} = U\ (T_{16}\text{-}T_{17})/[U\ (T_{16}\text{-}T_{17})]_d$
H_2O/H_2O HEAT EXCHANGER, tube side	$I_{23} = U\ (T_{23}\text{-}T_{22})/[U\ (T_{23}\text{-}T_{22})]_d$

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)	$I_{29} = \|rms(a_i)/rms(a_{i,})_d\|$

LEGEND FOR TABLE II

Pressure	*P*	Angular velocity (rad/s)	ω
Temperature	*T*	Efficiency	η
Mass flow rate	*M*	Compression ratio	β
Molar concentration	*X*	Design Conditions	$(\ )_d$
Heat Exchange Coefficient	*U*	Specific Heat	c_p
Vibration amplitude	*A*	Root Mean Square Norm	*rms*

TABLE III. FAILURE DETECTION CRITERIA

(For the interpretation of the attributes "high" and "low", see Section 4.2 here below)

COMPONENT	TYPE OF FAILURE	FAULT SIGNATURE
Filter	Fouling	I_1 high$\cup m_2$ low
Compressor	Fouling	(I_3 low$\cup I_4$ low)$\cap$(I_3 low$\cup$I2 high)
Compressor	Malfunctioning	I_2 high$\cup C_{lomp}$ high
Compressor	Choking	(I_4 low$\cup I_5$ high) $\cap$ (I_4 low$\cup T_3$ low)
Compressor	Excessive T_3	I_2 high
Compressor	Stall	I_4 low$\cup I_5$ low
Combustion Chamber	Fouling	I_6 high
Combustion Chamber	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 low
Electrical Generator	Unbalance	$I_{18} \neq 1$
By-pass Stack	Emissions	$I_9 > 1 \cup I_{10} > 1$
By-pass Stack	Heat Losses	I_{11} high
Boiler Stack	Emissions	$I_{12} > 1 \cup I_{13} > 1$
Boiler Stack	Heat Losses	I_{14} high
Turbine	Excessive T_4	I_{16} high
Turbine	Fouling	(I_{15} low$\cup T_5$ high$_)$ $\cap$(I_{15} low$\cup p_5$ high$_)$
Turbine	Choking	I_{17} high$\cup p_4$ high
Afterburner	Fouling	I_{19} low$\cup I_{20}$ low
Afterburner	Losses	I_{19} low
Primary Heat Exchanger	Fouling	I_{21} high$\cup I_{22}$ low
Primary Heat Exchanger	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
Main Pump	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	$I_{29,i} \neq 1$

is “interpreted” in a fuzzy sense (Section 4.2 below): a “fault signature” is assumed to exist not when the values of the indicators are exactly within their pre-assigned respective deterministic fault ranges, but when there is a (fuzzy) likeliness that they are in that range.

4.1.4.2. Creeping failures

We talk of a “creeping fault” when the instantaneous signature S_j is not yet within the failure range, but its values show an unequivocal tendency to approach that range. In other words, if a single indicator varies in time in such a way that at a time $t_o+\tau$ in the future its value will exceed its threshold value, then PROMISE assumes that there is some fuzzy likeliness that at time $t_o+\tau$ 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, $\left|\frac{dS_j}{dt}\right|$ 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 both cases, we again adopt a fuzzy approach.

4.2. Fuzzy fault detection criteria

A Diagnostic system cannot operate solely in a binary mode: its warning signals cannot be based on a rigid threshold value beyond which a fault is certain, and below which a fault has zero probability. This is a clear case of “vagueness”: we need to instruct our ES to consider that a fault may happen “somewhat below” the threshold, and may not happen “somewhat above” it. Fuzzy Logic is a very powerful technique capable of dealing with cases in which vague knowledge constitutes a substantial portion of the available knowledge. There are two considerations that constitute a necessary premise to the fuzzy procedure we are going to discuss in this Section:

1) As odd as it may seem, Fault Chains are not exactly represented by strict causal propositions, like <”IF event a AND event b...AND event n” THEN “Fault K = true”>: the final event of interest (the fault) has an aleatory character that cannot be captured by a purely deterministic description and cannot be conveniently expressed simply by probability theory. An “absolutely certain fault event” does not exist *a priori*, but neither does its logical opposite, the “absolute certainty of the absence of a fault event”. Both come obviously into existence when they happen at a certain time *t*, and have therefore an *a posteriori* probability value of either (1, 0) OR (0, 1) respectively: but our method has to be able to predict a fault *before* it happens, and from a purely probabilistic formulation, the fault *probability* is equal to zero until it suddenly “jumps” to one. Fuzzy Logic solves this problem by assigning a *likelihood* to the event “fault K = true”, or, to put it into AI terms, by assigning non-binary truth-values to it. Thus, we say that, in a certain range of values taken by indicator I_j, the likelihood of “fault K = true” is low (not necessarily equal to zero), and that in another range the likelihood is high (not necessarily equal to one). These considerations are quantified by the technique described here below.
2) Fuzzy truth-values have nothing to do with probability. A “fuzzily unlikely” fault may well happen, reaching a probability equal to 1 while its fuzzy truth-value is still below - say- 0.2. The opposite is true as well: a fuzzily likely fault (high fuzzy truth-value) may never happen, and maintains a zero probability forever.

With reference to *Figure 2*, Fuzzy Logic assigns a likelihood to the event “fault K = true” that grows (in this case, linearly) from I_j = 0.95 to I_j = 1.05. This means that the higher 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 be NOT realised: the final truth value of Fault K is a linear combination (in this simple 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 (Sriram 1997) 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 not necessarily linear: actually, S-shaped curves (*Figure 3*) give better results;

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.

4.3. The diagnostic mode

This Section describes in detail the procedure employed by PROMISE when in Diagnostic Mode. The sequel of the operations, which also constitutes the code logical flow chart (*Figure 4*), clearly mimics the behaviour of a human operator:

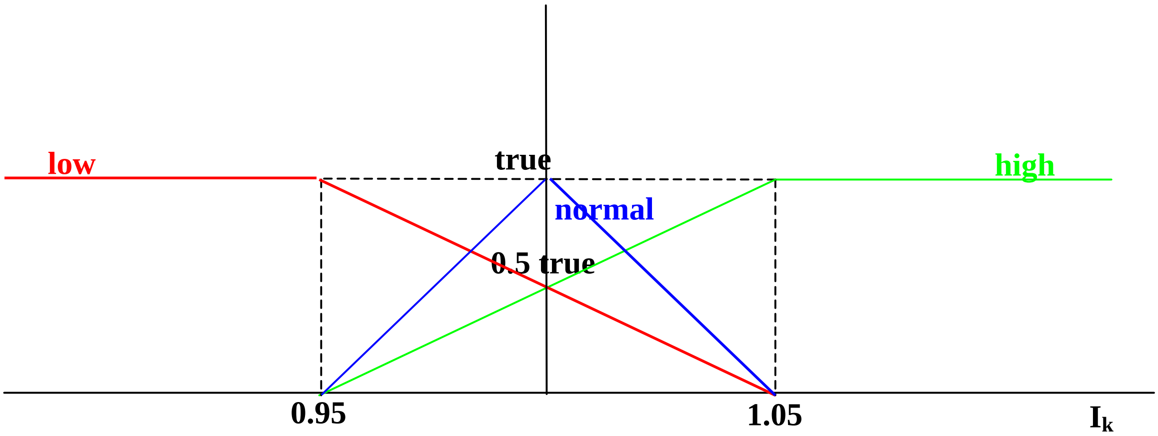

Figure 2. Fuzzy truth-values in a critical interval

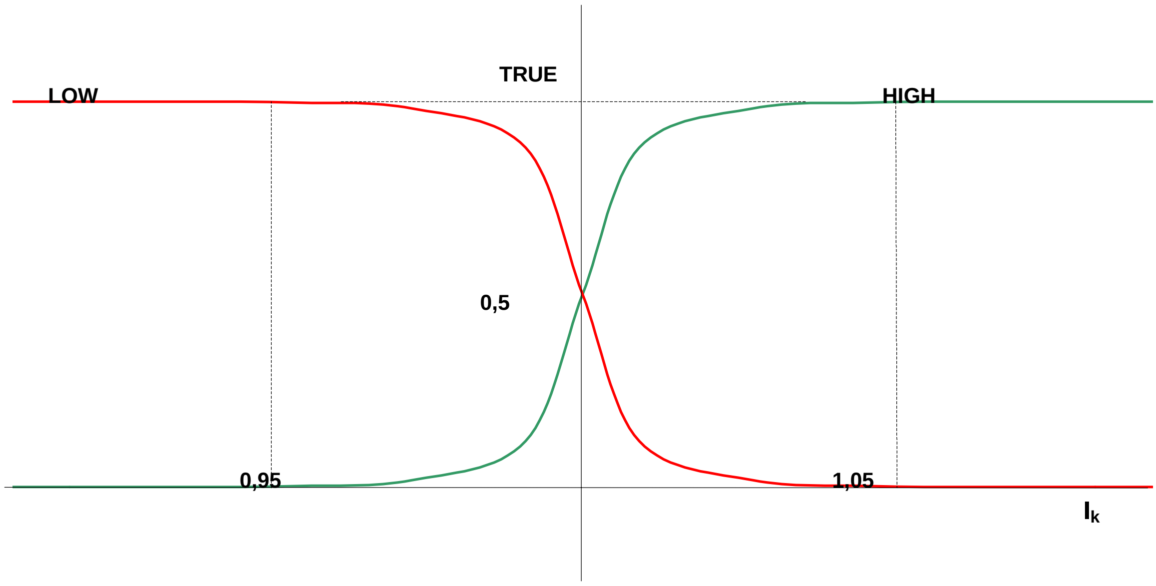

Figure 3. S-shaped curves

a) The ES compares the instantaneously measured data with their respective "expected values". The code must be exactly instructed on how often to execute this comparison (the user must specify a desired time interval for the scanning) and must have some way of knowing the expected value for each indicator: this is where a Process Simulator is needed;

b) If the *k*-th measurable (or indicator) is outside of the specified tolerance range with respect to its own expected value, PROMISE activates a procedure to check the component to which the measurable *k* is directly related;

c) PROMISE controls whether the detected event appears in one of the Fault Signatures contained in its KB, or whether the event may displace the operating point in such a way that at least one of these signatures becomes active. To do this, the ES monitors a certain number of related relevant parameters M_j, to check whether they are in a "dangerous" range and whether their gradients are "excessive" with respect to their expected values (also contained in the KB);

d) If no possible Fault Chain (*Figures 5,6,7*) is detected, PROMISE informs the human operator of the anomalous behaviour of the *k-th* measurable, keeps monitoring the relevant component closely for a pre-assigned period of time, and takes no further action;

e) Otherwise, PROMISE decides what remedial action must be undertaken (choosing from a list contained in its KB), and whether it is possible to wait until the next scheduled maintenance intervention.

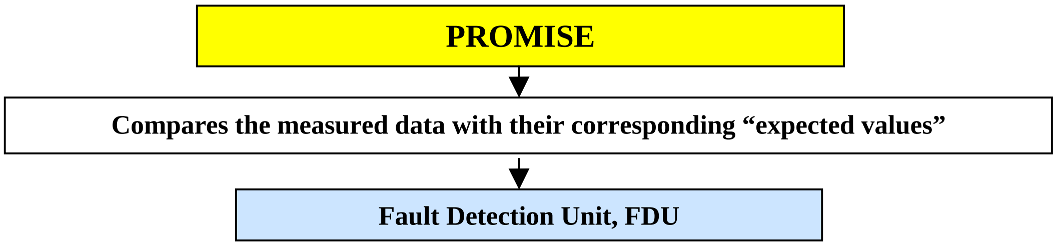

Figure 4. Diagnostic Mode

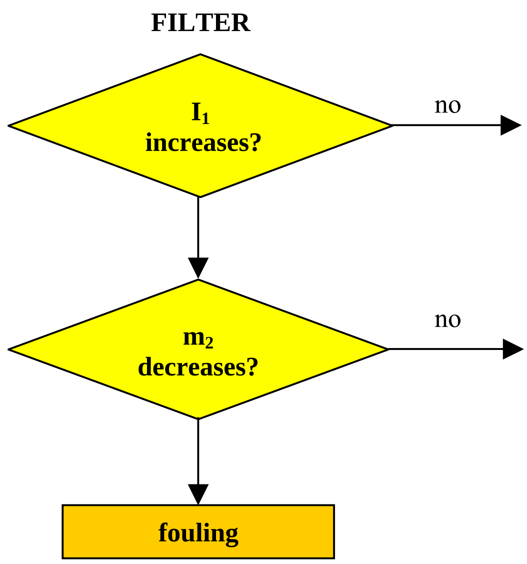

Figure 5. Filter Fault Chain

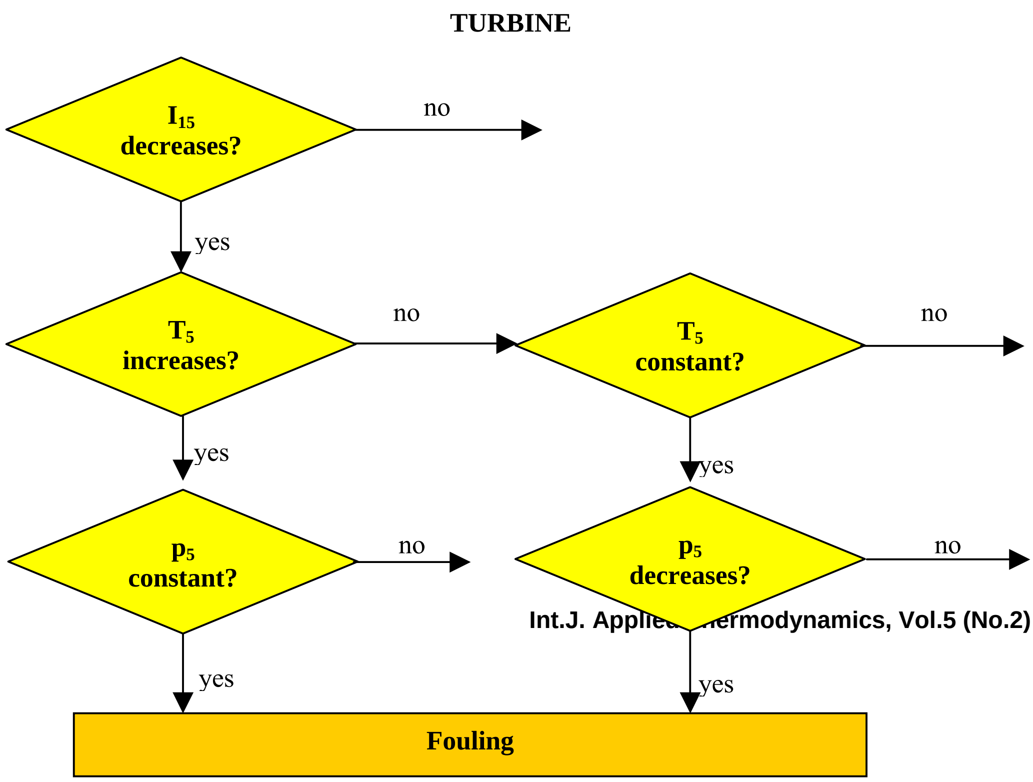

Figure 6. Turbine Fault Chain

MAIN PUMP

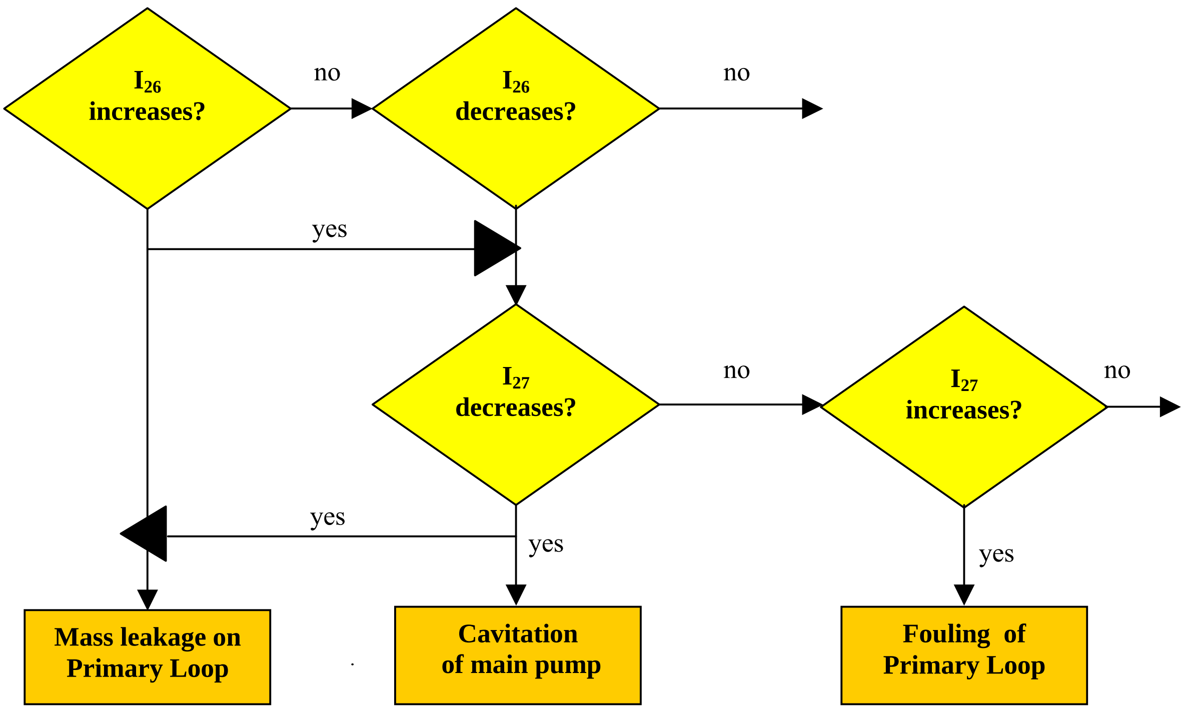

Figure 7. Main Pump Fault Chain

4.4. The prognostic mode

The Prognostic activity is performed in accordance to a paradigm that is very similar to that employed for the Diagnostic mode, thanks to the logical embedding strategy adopted in the construction of the Expert System. The steps taken by PROMISE (*Figure 8*) are the following:

f) The ES compares each one of the instantaneously measured gradients with its corresponding "expected value". The frequency of this operation (how often this comparison must be performed) is specified by the user; the expected value for the gradient of each indicator usually requires on-line use of a suitable Process Simulator;

g) PROMISE projects the value of each measurable to time t+Δt, using a proper multi-point extrapolation over the last two or three time steps. If the k-th projection falls outside of its tolerance range at t+Δt, PROMISE activates a procedure to check the component to which the measurable *k* is directly related;

h) The ES controls whether the foreseen variation leads, within a specified "incubation time", to one of the known Fault Signatures (the exact extension of this incubation period is prescribed in the KB, and is different for different components). The code at this point classifies the event "trend of the *k-th* measurable" (or indicator) as either "dangerous and possibly related to an excessive performance derangement trend", or "indifferent". In practice, the code checks other additional *M* relevant parameters (or indicators) to verify whether their values and those of their gradients are similar to any of the known fault signatures;

i) If no Fault Signature is detected, PROMISE informs the human operator of the anomalous behaviour of the *k-th* measurable, keeps monitoring the relevant component closely for a pre-assigned period of time, and takes no further action;

j) Otherwise, PROMISE decides what remedial action must be undertaken (choosing from a list contained in its KB), and whether it is possible to wait until the next scheduled maintenance intervention.

It is important to remark that the results of each prognostic step are used by the ES to "re-assess" its diagnostics at the successive time step. The idea is 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 entirely heuristic, has no theoretical justification, and its implementation requires some prior knowledge of the probability statistics of each fault chain: but its application produced encouraging results for all cases examined here.

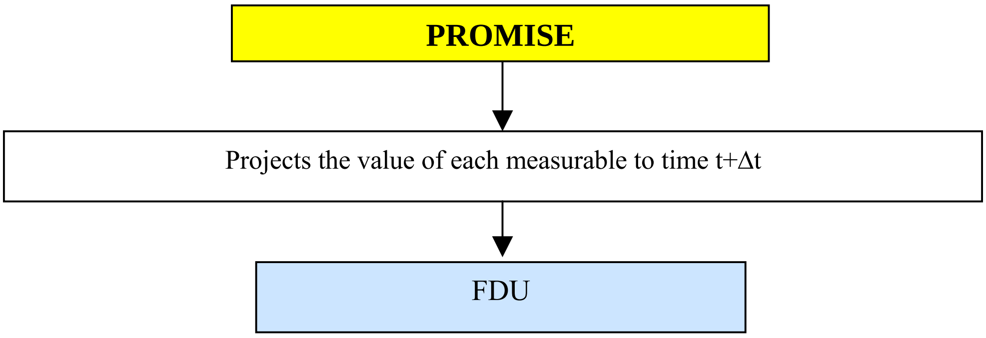

Figure 8. Prognostic Mode

4.5 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 or more indicators were varied manually, so that the ES 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, never misdiagnosing, and correctly predicting even related failures that had not been specified in advance. When artificial time-sequences that would indicate a slow derangement of one or more measurables from their nominal values were inserted, PROMISE always predicted the correct "creeping fault", thus performing its prognostics correctly. Once the code had been thus satisfactorily calibrated, three fault conditions were chosen and the respective "logs" were submitted to the ES:

A) Gradual and abnormal increase of the Air Filter pressure drop. The increase amounted to 30% of the nominal value within two successive days (48 hours) with a linear behaviour ("ramp").

B) Gradual and abnormal increase of the gas temperature at Combustion Chamber outlet. The increase amounted to about 10% of the nominal value within ten minutes, again with a linear ramp.

C) Gradual and abnormal decrease of the Main Pump flow rate. The decrease amounted to 30% of the nominal value within ten minutes. 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 Biagetti et al., 2002 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 minutes of the run, both diagnosis and prognosis of the filter were active. Starting from the third minute, PROMISE calculates the indicators' trend using the previous values. After 480 minutes 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 minutes of the run, both diagnosis and prognosis of the turbine were active. Starting from the third minute, the indicators' trends were computed. After 5 minutes 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 minutes of the run, both diagnosis and prognosis of the turbine were active. Starting from the third minute, the indicators' trends were computed. After 12 minutes the prognosis was interrupted, and a "Main Pump Cavitation" warning message was activated.

4.6. The next step

In its present version, PROMISE is not capable of planning a *restoring intervention,* i.e., to suggest a list of actions to take to re-establish correct plant behaviour. Such a task is very complex, represents a higher-level activity, and would need a separate ES for its implementation (Biagetti et al., 2002).

Future developments include:

1. The implementation of a remote monitoring and diagnostic system, i.e., of a system with the ability to readily access operating data, transmit them to a central location, perform diagnostic evaluation automatically or semi-automatically, report on the operating conditions of a component and offer recommendations for attaining a better overall performance. This could be achieved by an off-site monitor (CLIENT supervisor) detecting anomalous conditions and communicating the relevant data to on on-site monitor (SERVER) that performs the diagnostic and prognostic actions (*Figure 9*).
2. The addition of a "secondary" KB to improve data quality and precision.
3. The direct connection of the ES with the actual data acquisition system.

4.7. A note on shell-dependency

The original version of PROMISE (Biagetti et al., 2002) has been implemented under the G2® shell, and it is currently available only in this format. To extend its portability, a second version of the code, implemented in ACCESS® will be soon made available. As all AI codes, PROMISE has an intrinsically limited portability, and its syntactical structure (its source-code) is strongly dependent on the environment within which it has been developed. In fact, the graphical interface is at present hard-linked to (and only available under) the G2 shell, but this fact, though perhaps annoying, is inessential, as the appearance of the screen displays can be in principle reproduced under any commercial operative system.

5. Conclusions

When we undertook the development of a Diagnostic & Prognostic tool for Plant Intelligent Health Monitoring, we had four goals in sight:

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 ES working under the rules defined above;
4. To field test an α-version of such an ES on a proper set of actual plant data.

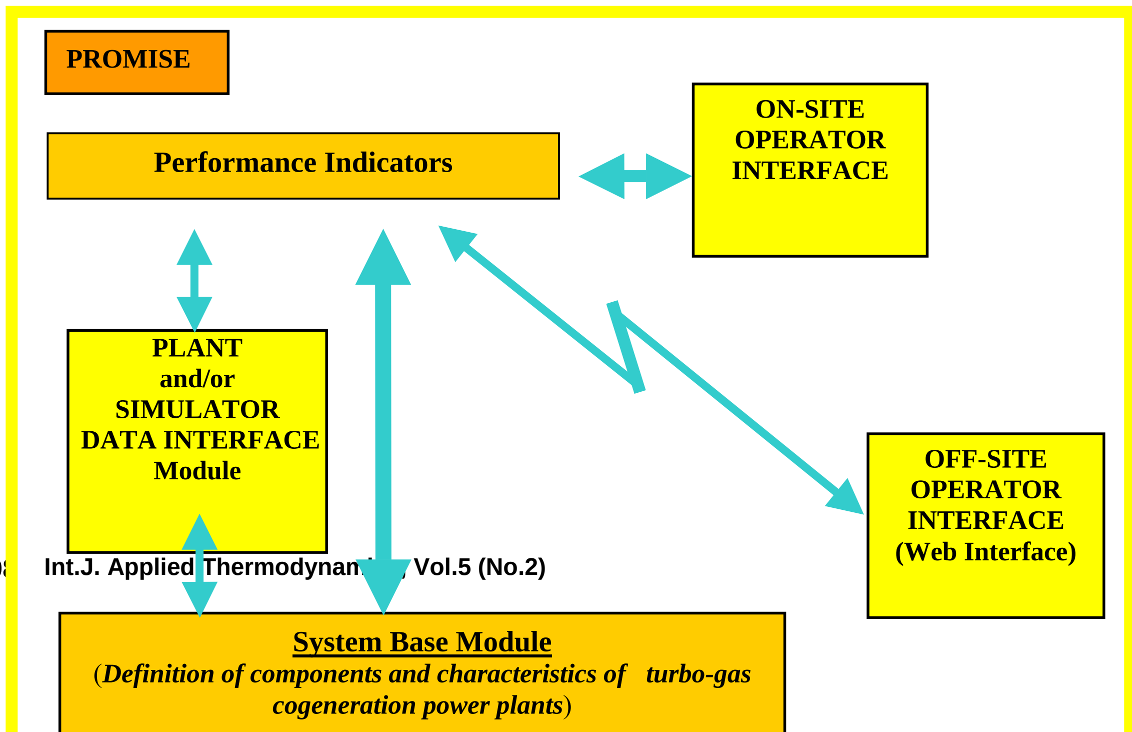

Figure 9. Remote monitoring system

All goals have been achieved. Though 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 (no aliasing etc.). 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 ES: 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

Anonymous staff writer: *Frame 7 and Frame 9 EC models bridge the gap*, MPS, July 1994

Biagetti, T. et al., 2002, **"**PROMISE: a tool for the on-line intelligent performance prediction of cogeneration plants." *Proceedings of ECOS'02-* Berlin, July2002

Gülen, S.C., Griffith, P. R., Paolucci, S., 2000, "Real time on-line performance diagnostics of heavy-duty industrial gas turbines" ASME paper 2000-GT-312,

Crescenzi, T., Mazzei, D., 2000, "Analysis of the operating data of the energy generation system in the Casaccia Research Centre" ENEA Technical Report, July 2000 (in Italian)

DOE Fossil Energy Program: *Tomorrow's Turbines*, www.fe.goe.gov, 2001

Forsyth, G., Delaney, J., 2000, "Designing diagnostics Expert Systems for long-term supportability, *ASME paper 2000-GT-31*,

Frangopoulos, C. A., 1990, "Intelligent functional approach: a method for analysis and optimal synthesis-design-operation of complex system." *Proc. Florence World Energy Research Symposium* Florence, Italy, May 28-June 1, 1990

Melli, R., 2001, *Private communication*

Munoz, J.R., Von Spakovsky, M. R., 2000, "The use of decomposition for the large scale thermoeconomic synthesis/design optimisation of highly coupled, highly dynamic energy systems-theory and application." *Proc.IMECE 2000,ASME-AES*,

Ozgur, D. et al., 2000, "Remote monitoring and diagnostics system for GE heavy-duty gas turbines" *ASME paper 2000-GT-314*

Papalambros, P. Y., Wilde D. J., 1988, *Principles of optimal design: Modeling and Computation*, Cambr. U.Press

Privette, R. M., *Computer-aided process design Trends in the power industry,* in R.Boehm ed. *"Developments in the design of Thermal Systems",* CUP 1997

Roemer, M. J., Kacprzynski, G. J., 2000, "Advanced diagnostics and prognostics for gas turbine engine risk assessment" *ASME paper 2000-GT-30*

Sciubba, E., Melli, R., 1998, *Artificial Intelligence in Thermal Systems Design: Concepts and Applications*, Nova Science, NY.

Sriram, R. D., 1997, *Intelligent Systems for Engineering*, Springer NY.

Tsalavoutas, A. et al., 2000, "Combining advanced data analysis methods for the constitution of an integrated gas turbine condition monitoring and diagnostic system" *ASME paper 2000-GT-34*

Zaleta-Aguilar, A., Royo, J., Valero, A., 2000, "Thermodynamic Model of the Loss Factor applied to Steam Turbines" *Int. J. of Applied Thermodynamics.* ,Vol.4, No.3