10

Wind Farms Sensorial Data Acquisition and Processing

Inácio Fonseca[1], J. Torres Farinha[1] and F. Maciel Barbosa[2]
[1]*Institute Polytechnic of Coimbra*
[2]*Engineering Faculty of Porto University & INESC Porto*
Portugal

1. Introduction

In this chapter is introduced the issues involved in the Wind Farms Sensorial Data Acquisition and Processing. This chapter is organized in five sub chapters summarized afterwards. The first sub chapter is the introduction. The second sub chapter makes an overview of a wind maintenance system, describing in detail the software related to the acquisition system, the information system and other software. This sub chapter explains also the operation of the acquisition system, including algorithms, hardware and firmware details. The third sub chapter deals with algorithms that manage the results of a methodology presented in the second sub chapter, with the objective to illustrate the operation of the system. The penultimate sub chapter will present results including simulation and real operation of the system, data details for clock synchronization protocols with improved changes, acquisition time and a SVM (Support Vector Machines) classifier applied to sensorial wind data. Finally we will make the chapter conclusions and present the references used in this chapter.

The contribution of this chapter is in the design of the architecture proposed with emphasis for synchronous data acquisition in different geographic points. An improvement for PTP (Precision Time Protocol) is included to achieve fast time convergence in the initial phase of a clock synchronization setup. The control and setup of acquisition timings also play an important role in the system behaviour. This chapter also includes different alternatives for this subject.

Given the current energy framework and global climate change, the emphasis on renewable energy has grown a lot. One of the most important renewable energies is from wind that has given great contribution for this new paradigm. There are, however, many aspects that must be considered and are related to its framework as an energy environmentally friendly. This growth in wind farms has the effect of the increase in diversity of the type of equipment in wind turbines. Moreover, the average life of each wind generator and readiness of this kind of technology means that there is a legacy of equipments for different ages and maintenance needs.

An information system for maintenance, called SMIT (Terology Integrated Modular System) is used as a general base to manage the assets and for the strategic lines to the evolution of

maintenance management, which incorporates on-condition maintenance modules, and the support to the research and development done around this theme.
The use of open source software in many institutions and organizations is increasing. However, a balance should be considered between the software cost and the costs of its technical support and reliability. This way had shown to be a good option to integrate with SMIT, not only as the base for wind generators but as the base technology to other applications.
The SMIT system is based on a TCP/IP network, using a *Linux* server running a *PostgreSQL* database and *Apache* Web Server with PHP, and *Octave* and *R* software for numerical analysis. This maintenance system for wind systems uses also special low cost hardware for data acquisition on floor level. The hardware uses a distributed TCP/IP network to synchronize SMIT server master clock through Precision Time Protocol. The development of maintenance management models for multiple wind equipments is important, and will allow countries to be more competitive in a growth market. For on-condition monitoring, the algorithms are based on Support Vector Machines and time series analysis running under *Octave* and *R* open source software's.
A wind turbine is a complex system with several components changing constantly and supporting strong forces. By consequence, it can experience many problems, such as vibrations, electrical failures and many other kinds of faults (Joseph & Gutowski; 2008; Caselitz & Giebhardt; 2002; Hameed et al.; 2009; Scheffer & Girdhar; 2004; Durstewitz et al.; 2005). Additionally, wind farms are usually far from cities and from companies that support their maintenance. Technical assistance is expensive and the combination of on-condition maintenance with the best practices of operational research to minimize distance costs is extremely important. The main objective is to implement a maintenance plan using on-condition maintenance through on-line data instrumentation, acoustic techniques, vibration techniques (Fonseca et al.; 2009, 2008), infrared images, stress measurement, zero crossing current analysis and artificial intelligence, in a coherent and synergetic way. Another strategic objective of this work is to build the entirely system with open source and low cost hardware (MicroChip; 2010; Opensource; 2010). The reliability of actual microcontrollers and open source software is of great importance for market acceptance.

2. Wind maintenance system

2.1 Global overview

Fig. 1 gives an idea of the system design that consists of a server (SMIT), connected by TCP/IP connectivity to an Ethernet-CAN gateway. The data acquisition devices are interconnected in the CAN (Controller Area Network).
The SMIT system uses a Linux Server running *Apache* Web Server and PostgreSQL database (Open-source; 2010). This module/software is responsible for saving information in a structured way. Network connections can be made by fiber optics, UTP cables, Wireless, Satellite link or HSDPA/GSM technologies.
Data acquisition can be done using special low cost hardware made for this project, based on microcontrollers with special firmware. The methodology adopted relies on the use of low cost components and devices, to create a data acquisition system over IP networks. The basic idea consists on distributing a master clock among different field equipments, to ensure the synchronous acquisition of the different data collection points. The SNTP (Simple Network Time Protocol) (Group; 2010) and PTP protocols (PTP; 2010) are used to implement

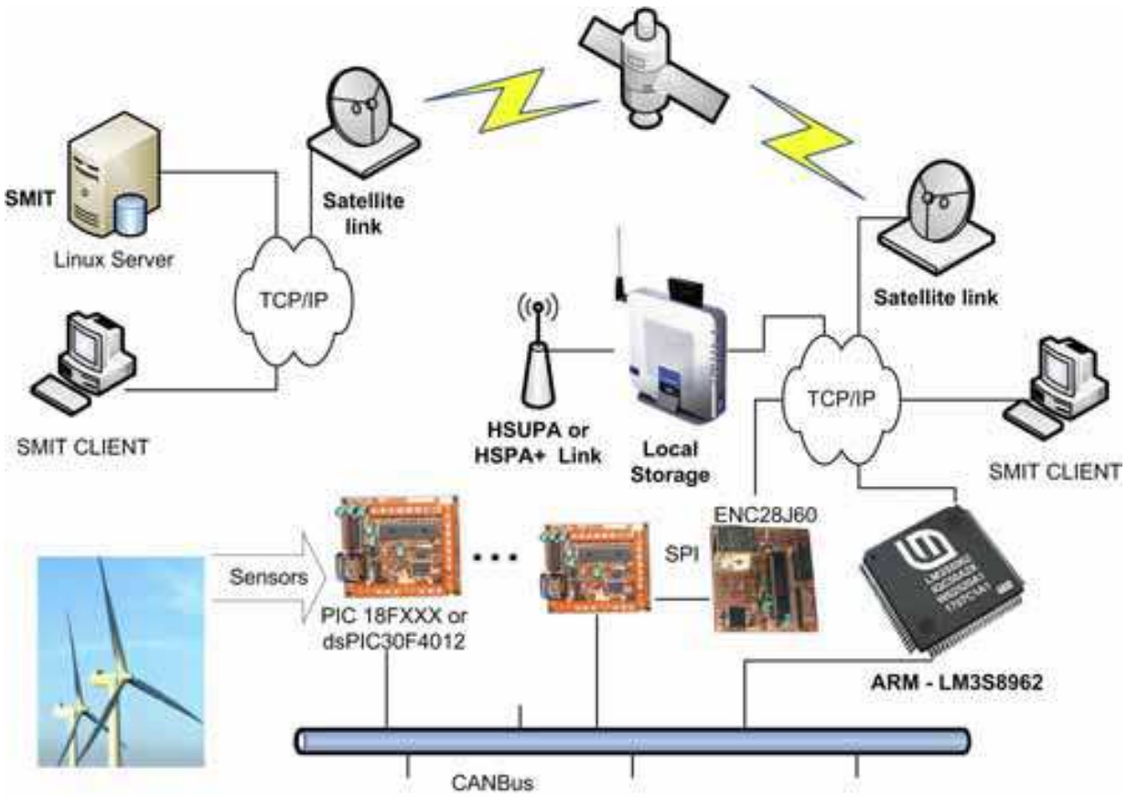


Fig. 1. An Integrated System for Maintenance of Wind Systems (Fonseca; 2010).

a set of control techniques in order to achieve clock synchronization. The basic structure of the system uses data collecting devices connected through a CAN network. One of the devices, with CAN and Ethernet connectivity, coveys the acquired information and relays it into the SMIT server. Simultaneously, this master node controls the data acquisition sequence, as well as the clock synchronization with the SMIT server. The integration of the developed hardware and software modules implies the data flow from the acquisition nodes to the server, which sends time references to the master device, including the reference clock signal. SMIT server uses a TCP/IP server for reception of data from acquisition points, using UDP (User Datagram Protocol) with acknowledgement.
Fig. 2 shows the flow information between different components.

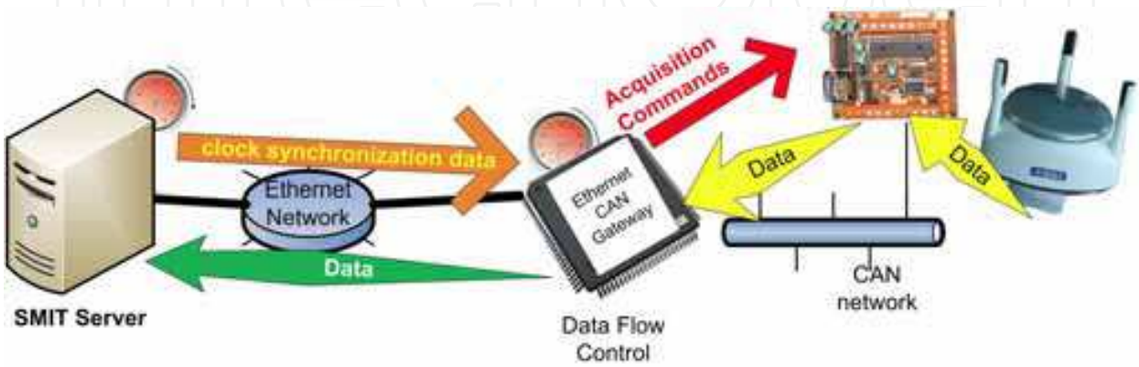


Fig. 2. Information flow between different SMIT's components (Fonseca; 2010).

The SMIT can also incorporate any hardware with IPv4 connectivity. Tests were performed with USB 6251 National Instruments Acquisition board (Hardware and Software; 2010), using LabView 8.5 and Beckhoff BC9100 (Wago & Beckhoff; 2010). These two tests addresses the industrial equipment usually used in some instrumentation systems of wind farms (Hameed et al.; 2010).
The SMIT server runs the SNTPD or the PTPD.

2.2 Software technology

The SMIT is supported by *Slackware Linux* distribution (Open-source; 2010), version 13.0 or 12.1.
The PostgreSQL has many features, for example, stored procedures are able to be written in PL/pgSQL, as native language and optional by third party, PL/PHP, PL/R, etc. SMIT uses PL/pgSQL to implement database procedures, where the main logic of the program is located. SMIT's database uses 149 tables and 156 PL/pgSQL stored procedures. *PostgreSQL* version is 8.2.3 and 7.4.16.
For remote access it is used an *Apache* Server running PHP, versions 2.2.4 and 5.2.1, respectively. Some parts of the maintenance system are available by web browser for maintenance intervention requests and information exchange by web services (Technologies; 2010).
The SMIT's users can interact with the system using a Windows interface developed in Delphi version 7 and print reports through Crystal Reports version 9.0 or PHP enabled reports both stored in a special database table in a compressed format, allowing easily new reports uploading.
The system is able to perform automatic installation of new versions in Windows systems. System administration problems with passwords are implemented using CPAU (Joeware; 2010). This is an independent software command line tool for starting processes in an alternate security context, enabling software installation, without problems with the Windows Administrator password.
In the security field, the database has been changed to protect the knowledge through encryption at the stored procedures level, configuration files and database storage. A special tool has been developed to translate a SQL database definition file to encrypt/decrypt the PL/pgSQL procedures, using lex and bison Unix utilities while the encryption algorithm is self made by the authors. All the connections to the database are made using SSL sockets, implemented through OpenSSL (Hardware and Software; 2010). The PHP distribution has been also patched to permit encryption of PHP files and storage in the SMIT Linux Server.
For numerical data processing, the SMIT integrates the Octave distribution version 3.0.3 and version 2.8.0 of R (Eaton and Gentleman; 2010)]. By default, the SMIT incorporates some algorithms (described in the next sections).
Finally, as described above, the system can be integrated with third party software via web services technology (Technologies; 2010), using any language supporting this technology.
Fig. 3 shows the type of relationship between SMIT server and SMIT client in terms of TCP ports and the type of information exchanged.

2.3 Hardware

The low cost acquisition system presented here includes PIC18F2685, ENC28J60 for Ethernet connectivity, a dsPIC30F4012 for high speed acquisition (MicroChip; 2010), a board using

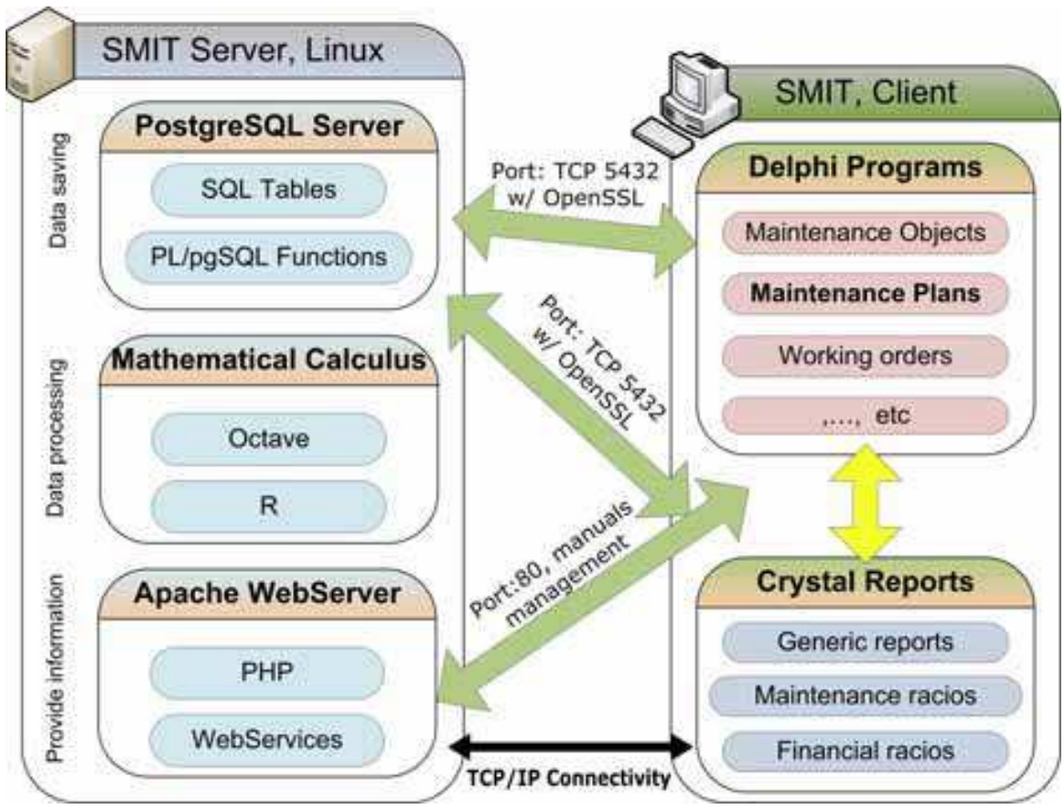


Fig. 3. Relational Diagram, SMIT Server *versus* SMIT Client (Fonseca; 2010).

Microchip digital potentiometers to implement a Butterworth low pass filter of 4th order with cut-off frequency between 100 Hz and 50 kHz and two cascade amplifiers with gain range: between 0.1 and 10. The frequency and gain are programmed by software using Programmable Gain Amplifier (PGA) and, finally, a board based on the Luminary microcontroller LM3S8962 (Luminary; 2010), an ARM-Cortex-M3 architecture with support for Ethernet packet time stamping in hardware, with two interfaces Ethernet at a 10/100 Mbps full/half duplex and a CAN 2.0 was built. All the programming tools for this microcontroller has been constructed based on GNU GCC toolchain for ARM-Cortex-M3 version 4.3.3, Binutils 2.19.1 and newlib-1.17.0 under Gygwin. This is the most expensive component of the system; however, it is priced by 12 Eur/unit and, in addition, an operational board runs with few elements. This board will be a gateway between the traffic from the CAN network and the Ethernet network (Fig. 2). Special interfaces for industrial sensors are also addressed to be used. Discrete analog signals and switching signals can be acquired. Analog signals, current and voltage are acceptable, if compatible with industrial standard 4-20mA current loop and 0-10V voltage. PIC software uses Microchip TCP Stack (MicroChip; 2010) and ARM uses lwip TCP Stack (Dunkels; 2003).

Relating to Fig. 2, the following blocks must be considered from hardware point of view:

- **Ethernet-CAN gateway** - Two systems can be used: ARM-Cortex-M3, LM3S8962, runs PTP client for clock synchronization; PIC18F2685+ENC28J60 running SNTP client for clock synchronization;
- **Acquisition points (slaves)** - PIC18F2685 for low velocity acquisition and dsPIC30F4012 for high speed acquisition connected in CAN network. ARM-Cortex- M3,

LM3S8962 and PIC18F2685+ENC28J60 connected in the Ethernet network if high demanding acquisition is required. Signal conditioning must be done according to the sensors used.

2.4 Execution and configuration of the acquisition system

In short, according to Fig. 2, the Ethernet-CAN gateway is responsible for:

1. Collect CAN network setup parameters and acquisition timings/periods table from SMIT server, to control acquisition points connected in CAN network;
2. Run PTP client receiving clock synchronization information;
3. Generate acquisition commands for acquisition points in CAN network;
4. Collect data from CAN network relaying it with SMIT server using Ethernet network.

Fig. 11, represents the firmware flow for this device showing the main functions described.
For the acquisition commands to be temporally precise, it is necessary to measure the propagation time in the CAN bus, depending on the number of bytes of the CAN message sent. This procedure can be made on-line or off-line, and Fig. 4 gives an idea of the measuring process. The CAN acquisition message sent by the gateway to ask slaves to acquire data should be sent at (Figs. 4 and 5):

$$time_{propagation} = \frac{T_1 + T_2 + T_3}{2} \approx \frac{T_1 + 0 + T_1}{2} = T_1$$
$$t = time_{requested} - time_{propagation} \quad (1)$$

The CAN slaves, while in setup mode, will auto-baud the communication velocity until a valid CAN message is received. After this stage they will start the normal cycle, waiting for a message asking for an acquisition and forwarding packets for measuring CAN propagation delay time, or receiving messages for firmware upload.
Fig. 5 shows examples of real data measured in the system, to compute how must time previous to the acquisition commands should be sent to the CAN bus by the Ethernet-Can gateway. Fig. 6 shows the precise time when messages are sent to acquire. In this case (one message with one byte to send), the CAN message should be sent 310 μs earlier. Absolute error is about 2,5 μs which gives a relative error of 0,8.

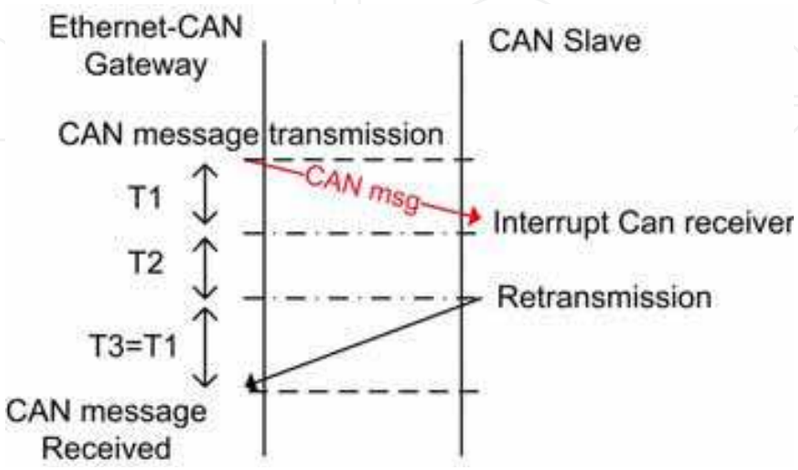


Fig. 4. Measuring delay in the CAN bus transmission media (Fonseca; 2010).

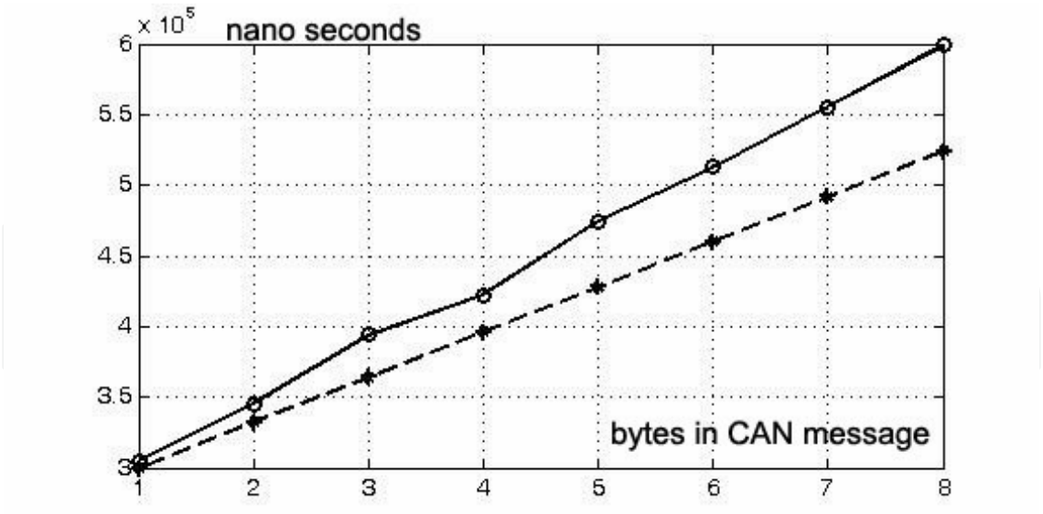


Fig. 5. Dash: Theoretical travel time (nano seconds) for transmitting extended CAN messages with 1 byte to 8 bytes of length. Solid: The delay time measured in the real system between Ethernet - CAN gateway and one CAN Slave (Fonseca; 2010).

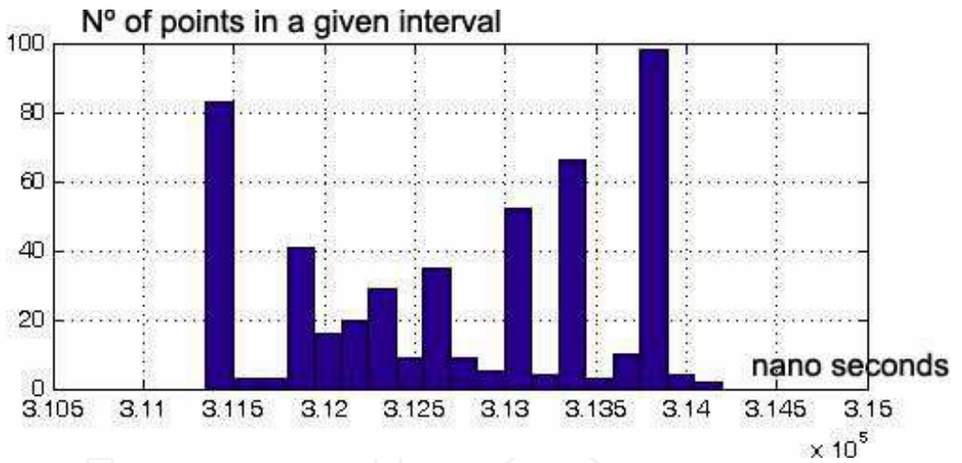


Fig. 6. Sampling time measured (nano seconds) in the Ethernet-CAN gateway after sending the acquisition CAN message with one byte to slaves (Fonseca; 2010).

The acquisition timings are saved in SMIT´s database where the global acquisition network is designed and stored. At its first step, a SMIT user will configure the acquisition network, choosing the boards on the field. The setup for CAN network is also stored in the database. Under this perspective it is very easy to change CAN network velocity by changing the parameters associated to the CAN network (Fig. 7, Fig. 8 and Fig. 9). Each slave in the CAN bus will have a different CAN ID, to receive different messages. The CAN ID for each card is always a multiple of 2. This limits the number of slaves in the CAN bus, but always lets to send a message that addresses all slaves, with acquisition instants coincident in time. At the second step, the information related to frequency sampling is indicated, and also the temporal interval like the starting and final acquisition date. These parameters are downloaded by the gateway board from SMIT server to control the sampling rate in the CAN bus, generating control signals for the I/O boards, as described.

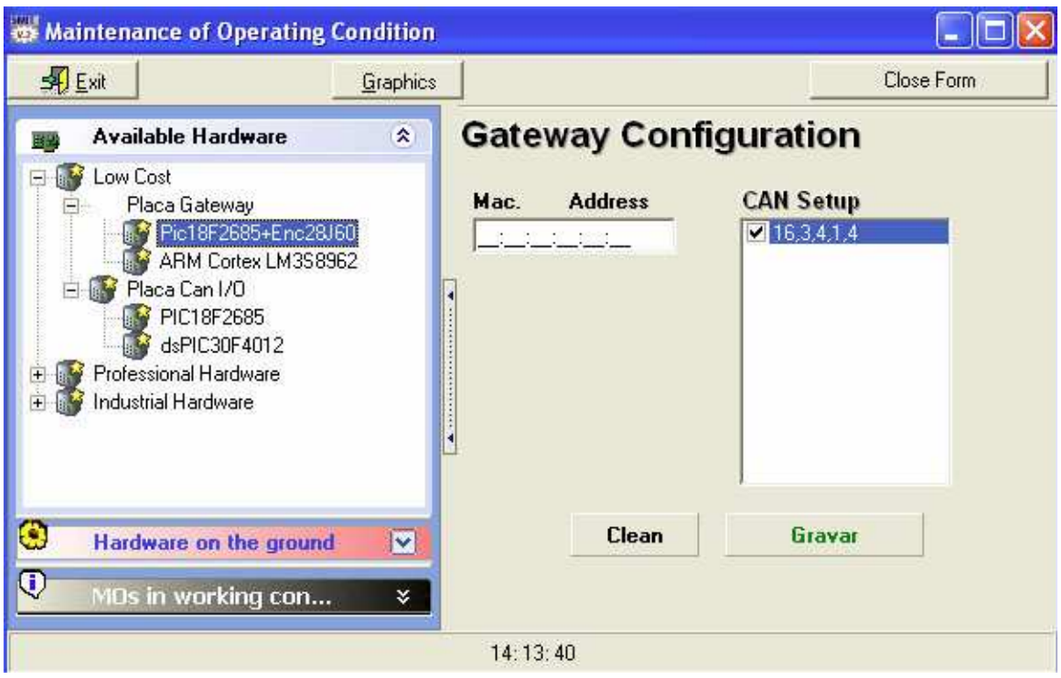


Fig. 7. Choosing hardware - in low cost mode a gateway is always needed as also as an I/O CAN board (Fonseca; 2010).

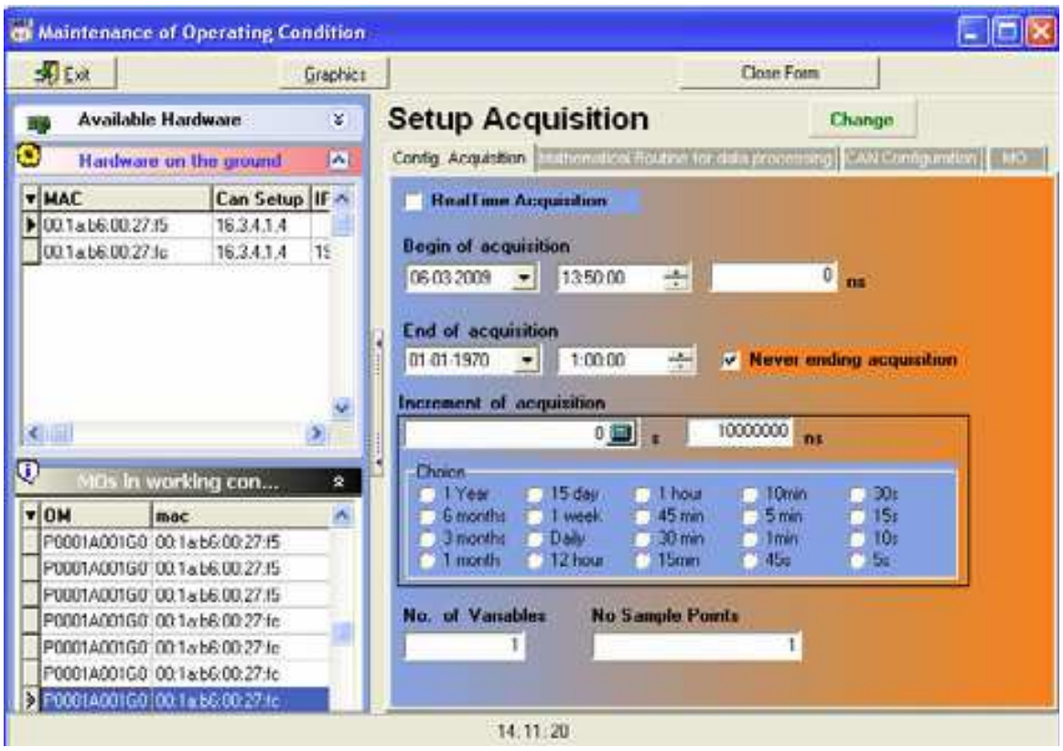


Fig. 8. Programming the acquisition temporization for each low cost I/O board in the SMIT´s on-condition module (Fonseca; 2010).

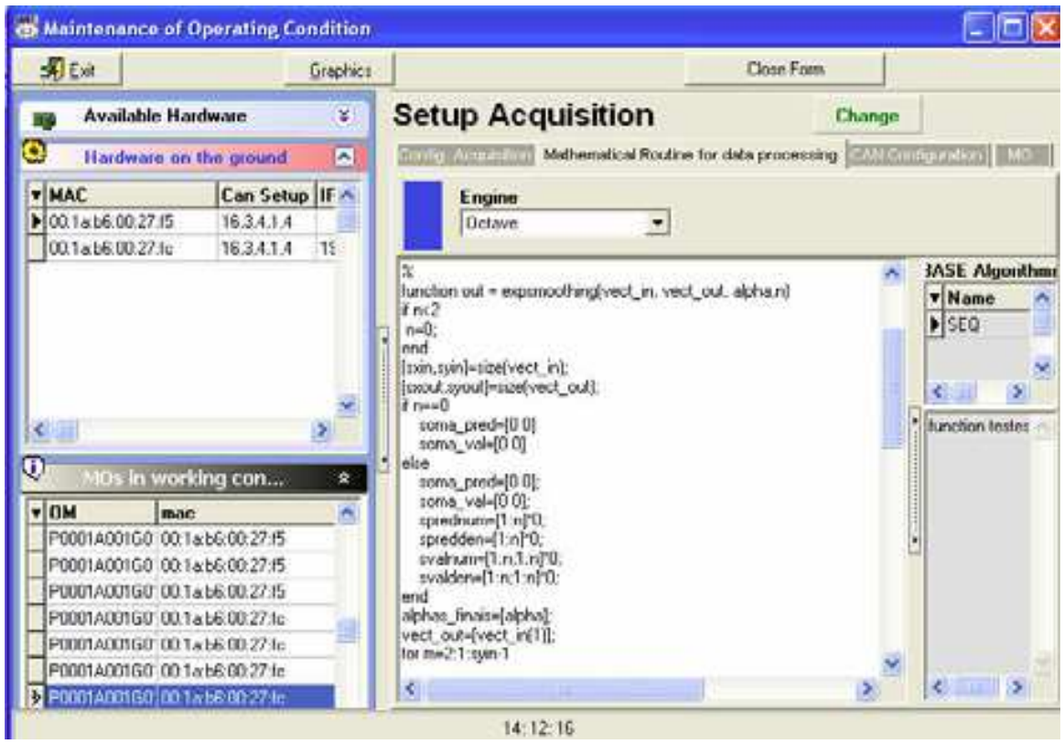


Fig. 9. Octave script using time series algorithms, for numerical data analysis (Fonseca; 2010).

2.5 Acquisition system operation

Acquisition timings are stored in SMIT's database, as also as the acquisition network topology. SMIT's user must configure the acquisition network topology, choosing the hardware that he wants to put on the ground (Fig. 7). The gateway board will use DHCP protocol to acquire an IP number, and receives the CAN setup parameters from the SMIT's database. This feature allows the user to change operation speed of the CAN industrial network in a simple way, altering the parameters stored in the database, as can be seen in Fig. 7. In the second step, the information related to frequency sampling is indicated, and also the temporal interval like the starting and final acquisition date (Fig. 8). These parameters are important to the gateway board. After this setup, the gateway will control the acquisition nodes (I/O boards), by sending acquisition commands to the CAN bus.

Fig. 2 shows a broad outline of how information flows through the system. The CAN-gateway is responsible by collecting data from acquisition modules connected to the CAN network. As can be seen, there are three types of information that can flow: a) information of clock synchronization in the Ethernet network; b) data acquired in the CAN network and then forwarded through the Ethernet network to the SMIT server using UDP packets with acknowledgment; c) acquisition timings sent by acquisition command messages to the industrial CAN network. The CAN bus devices are restricted to the global acquisition rate. If the number of nodes together generate data with a flow rate higher than the CAN network speed, the nodes must be divided by different CAN networks. Another goal is to synchronize the acquisition boards through time propagation from the SMIT server to PIC micro-controllers in the CAN bus using SNTP or PTP running in a cooperative way in the Ethernet-CAN gateway (Group; 2010; PTP; 2010).

Under this feature it is possible to ensure that different devices placed in different wind turbines perform signal acquisition at the same time. This aspect makes possible the comparison among the same data in different wind turbines and it is guaranteed that the gap between the acquisition times is less than 40 micro-seconds. The CAN slaves, in setup mode, will auto-baud the communication velocity until a valid CAN message is received. After this stage they will start the normal cycle, waiting for a message asking for an acquisition and forwarding packets for measuring CAN propagation delay time, or receiving messages from firmware.
Fig. 10 shows the information flow between SMIT server and an Ethernet-CAN gateway. In order to make possible the communication it was created on the server SMIT a UDP server on port 9945, responsible for managing all communications between the two systems. This UDP server tells to the gateways, the SMIT IP, acquisition tables and also collects the field data.
The Ethernet-CAN gateway operation is described in flowchart of Fig. 11. Two different operations are presented, one for setup mode and the second for normal operation. This flowchart presents the Gateway behaviour. Another important feature of the gateway is how it guards and relays the data information to the SMIT server. The gateway saves the information from the CAN nodes in a FIFO. For each node, the information saved is described in Table 1.

Msg size	ID CAN (4 bytes)	seconds (4 bytes)	nano-seconds (4 bytes)	Data

Table 1. Data Message saved on Gateway FIFO, for each CAN node.

If real time is required for any node, the data received from the CAN bus by gateway is not saved in its internal FIFO, but relayed to SMIT server immediately.
When the size of the FIFO reaches the maximum permissible limit of a UDP packet, the data is packed and transmitted to SMIT server. If the FIFO has not received any information in the last two minutes, the stored information is sent to the SMIT server, leaving the FIFO empty.
Each device generates the information described in Table 1, after several devices have sent their data, the FIFO will contain the information organized according to Table 2.

Equipment 1	...	Equipment n

Table 2. Information saved in the FIFO, collected from the acquisition nodes.

2.6 Synchronous data acquisition using PTP

PTP protocol works in the following way: each slave synchronizes with the RTC of the master through a set of specific messages (Sync, Delay Request, Follow Up and Delay Response). Sync messages are sent at a periodic rate of $T_{Sync} = 2$ *seconds* (Fig. 12) (PTP; 2010; Luminary; 2010; Correll & Barendt; 2006).
The master clock (SMIT) is described by two variables, seconds and nanoseconds, i.e., Ms: Mns. Each clock cycle, the nanoseconds variable is incremented and normalized to the seconds variable. The slave (Ethernet-CAN gateway), a microcontroller operating at a frequency rate of f_{no} =40Mhz (clock period $T_{no} = 25ns$), needs to keep two variables Ns:Nns in absolute synchronization with Ms:Mns variables. The maintenance of these variables is

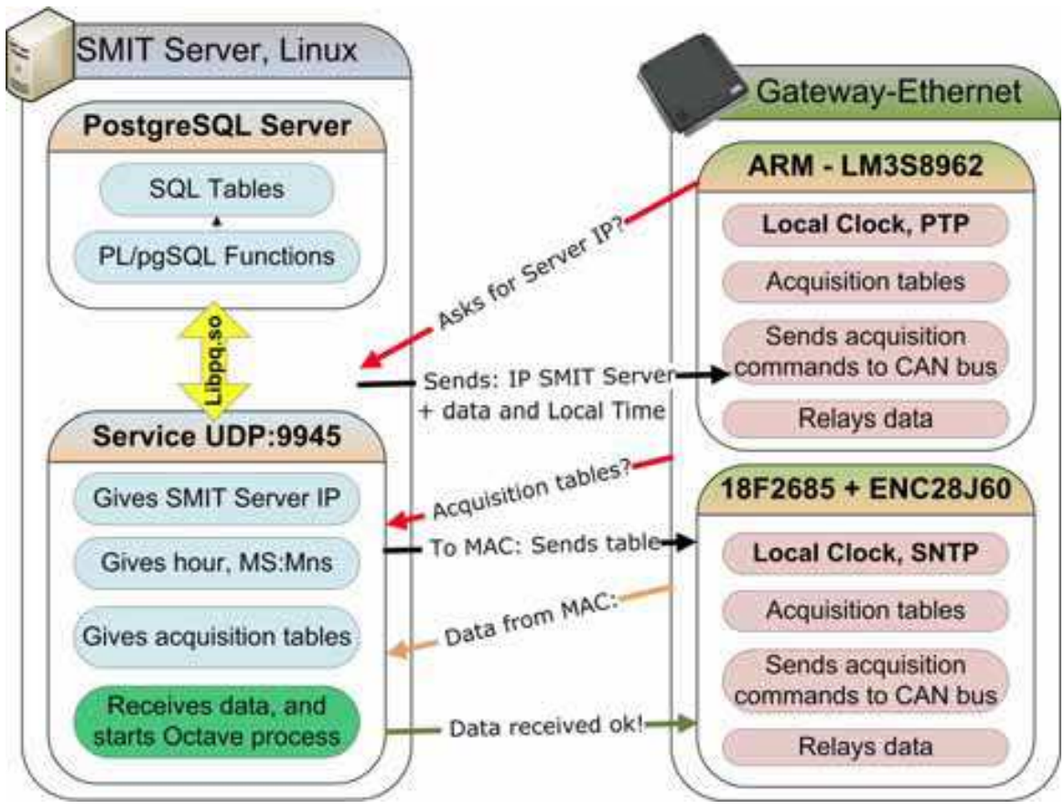


Fig. 10. Relational Diagram. SMIT server Gateway (Fonseca; 2010).

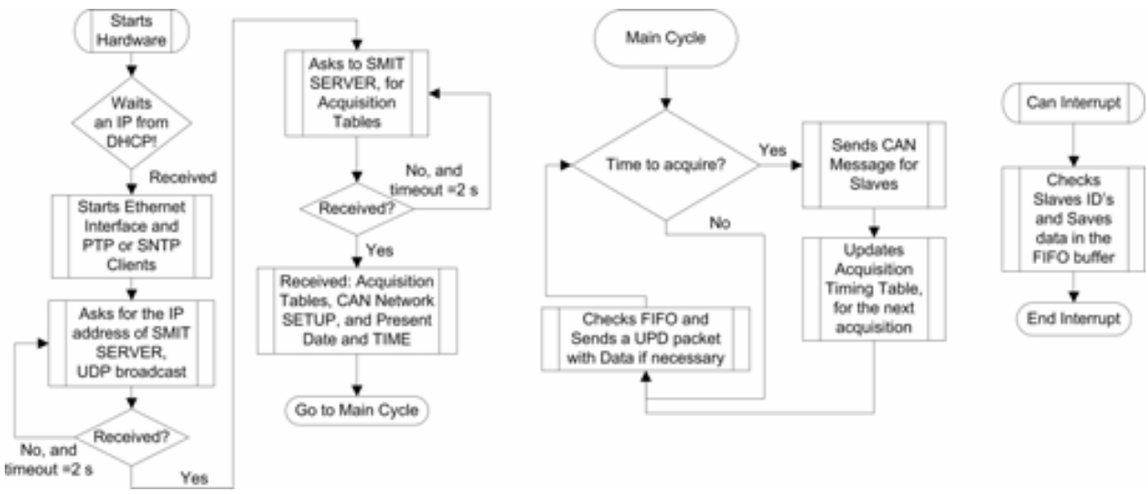


Fig. 11. Left: Ethernet-CAN gateway in setup mode. Right: Normal operation for acquisition control and data relaying (Fonseca; 2010).

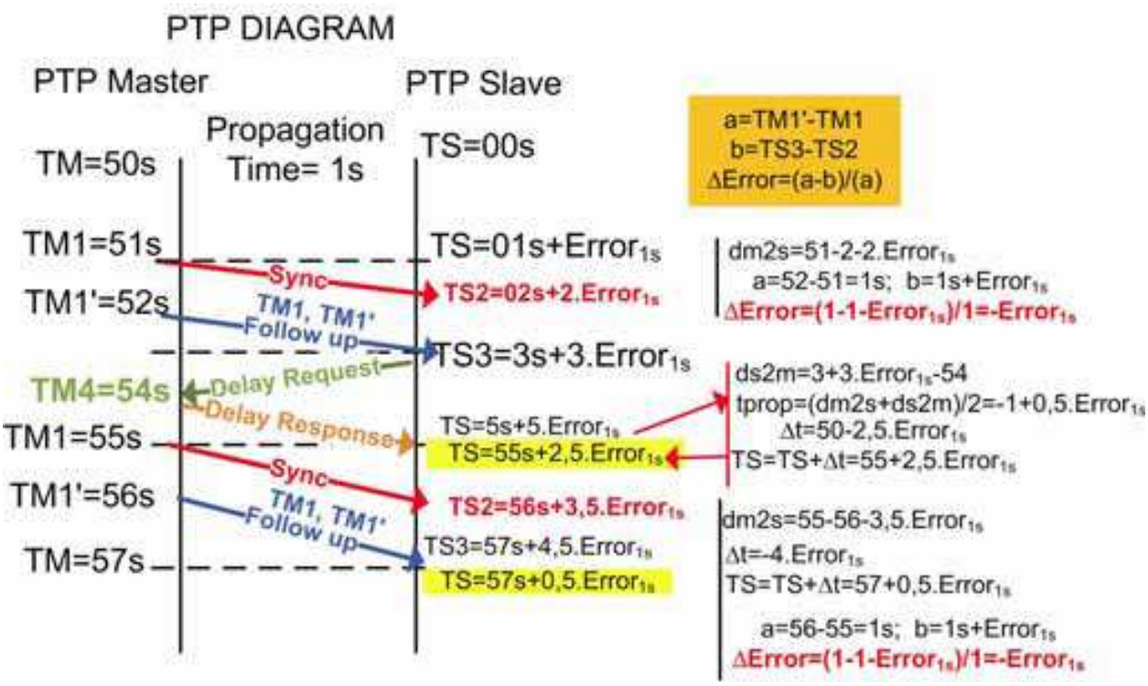


Fig. 12. PTP normal messages diagram (Fonseca; 2010).

usually done through Tick Interrupt System with a period of 10ms that is an adequate value for this timing. Thus, the value of Ns:Nns is incremented by 10 ms (10.000.000ns). In microcontrollers, when this interruption occur, it is necessary to plan a hardware counter (CNT_TICK), as described, with the value CNT_TICK_MAX = 400.000 = 10ms/25ns. The counter starts at zero and counts up until the limit (400,000), generating an interruption (to carry out maintenance on Ns:Nns), and restarting at zero again. To determine the correct time it is necessary to read the value of CNT_TICK and add it to the variables Ns:Nns. The theoretical accuracy of this system will be, in the limit, of the order of 25ns, whose value is impossible because it is the execution time of an instruction. Assuming differences in the frequency of oscillation of both clocks, which is perfectly acceptable with reference to the sender, without loss of generality the model of this differential can be designed and the clock error of the receiver defined by:

$$T_{no} = 25.T_m + \Delta T_{no} = 25ns + \Delta T_{no} \tag{2}$$

For a ΔT_{no} positive it means that the period of the slave is higher than the master. As a consequence, it will have an increase slowly over time, i.e. introduces a negative error in counting the clock given by:

$$\begin{aligned} Error_{10ms} &= -CNT\_TICK\_MAX.\Delta T_{no} \\ Error_{1s} &= 100.Error_{10ms} \end{aligned} \tag{3}$$

As the master frequency is Tm = 1ns that means: $\Delta T_{no} = T_{no} - 25.T_m$. From this description, it is clear that is necessary to require $\Delta T_{no} \rightarrow 0$ and change the value CNT_TICK_MAX. This control is not more than a software PLL function. The parameter CNT_TICK_MAX should be controlled and modified when calculating the error Δt in the reception of Follow Up

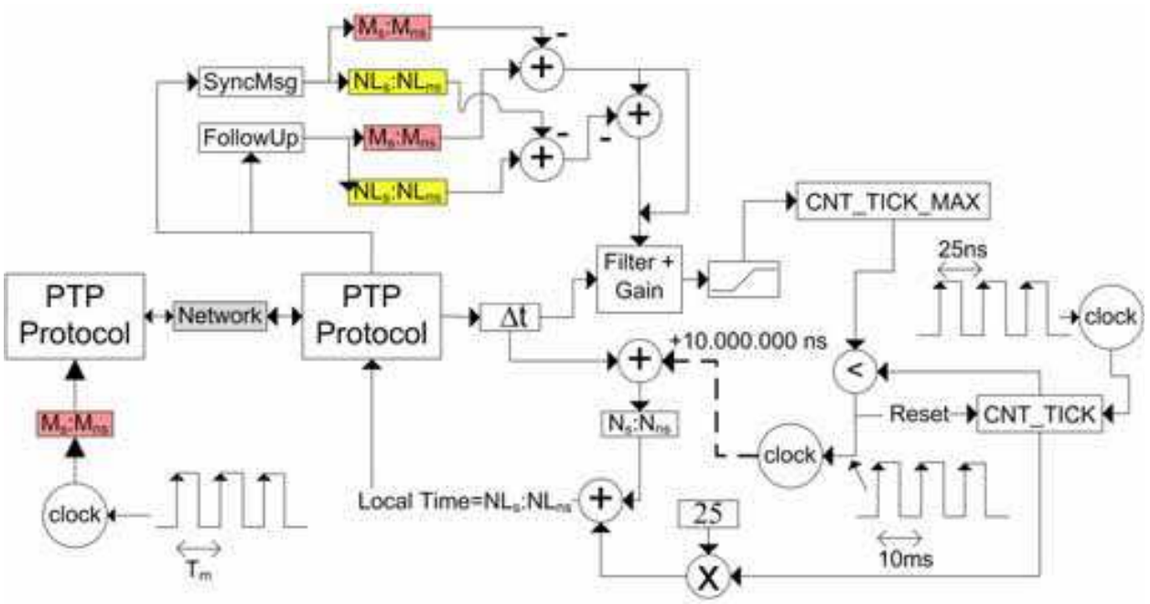


Fig. 13. Diagram to update local Slave clock with SMIT Master clock, using PTP protocol (Fonseca; 2010).

messages (if not send a Delay Request message) or at the reception of Delay Response messages, or both. This update of the base period of System Tick Interruption is directly related to the PLL operation. One additional note refers that the value of CNT_TICK_MAX has the following limits: it cannot be negative nor exceed the maximum allowable size of the microcontroller register. This means applying a threshold to the value calculated before it is reflected in the CNT_TICK_MAX parameter.

Experiences have also been made with two modifications in the PTP protocol, in particular changing the messages Sync and Follow Up to include the value of the master clock time (Fig. 12). Fig. 13 shows a block diagram of the text described, for the modifications made to PTP master software. The result shows a faster initial convergence due to decreased of initial error, but in subsequent periods is normal.

3. Relevant algorithms for wind turbine maintenance

In the literature many authors are working on this subject (Caselitz and Giebhardt; 2002; Durstewitz et al.; 2005; Hameed et al.; 2010). The data acquired from wind acquisition systems usually presents irregular values.

In this section will be presented two methodologies: the first one, a SVM classifier and the second one, prediction of the deterioration of operating conditions of the equipment by time series.

3.1 SVM classification

The data acquired from wind acquisition systems, usually presents irregular values. The system (on-condition classifiers) have to be very accurate, otherwise, a false error condition would be triggered. For example, when the wind break system is activated it is expected

that the power curve differs from the relation published by the manufacturer; however, the inner axis rotation versus active power should maintain the relation, because the break will only change the rotation speed and not the relation between rotation speed and produced power. The same arguments are valid for the *pitch* system.

The first algorithm used for detecting an uncharacteristic operation will use Support Vector Machines (SVM) classifiers (Cauwenberghs & Poggio; 2000; Suykens et al.; 2002; Kecman; 2001; Zhizheng & YouFu; 2009). The goal is to decide if data measured by the sensors shall be processed in detail or just stored. If the SVM classifier indicates an anomalous situation, a detailed processing will be performed related to the sensors already installed. The SVM classifier uses $\vec{x}_{wind-turbine}$, based on equation 4, using RBF as kernel mapping function.

$$\vec{x}_{wind-turbine} = \begin{bmatrix} Wind\ Velocity \\ Wind\ Direction - Nacelle\ Direction \\ Rotor\ Shaft\ Speed \\ Generator\ Shaft\ Speed \\ Active\ Power \\ Break\ System\ State \\ Pitch\ Angle \end{bmatrix} \tag{4}$$

Due to the unavailability of measuring some of these variables, including alignment of the nacelle, the angle of the rotor blades and condition of the brake rotor, two alternatives can be used based on two vectors for classification (to determine what the best) given by equations 5 and 6

$$\vec{x}_{wind-turbine} = \begin{bmatrix} Wind\ Velocity \\ Active\ Power \end{bmatrix} \tag{5}$$

or

$$\vec{x}_{wind-turbine} = \begin{bmatrix} Rotor\ Shaft\ Speed \\ Generator\ Shaft\ Speed \\ Active\ Power \end{bmatrix} \tag{6}$$

The change described above facilitates the analysis and application of the classifier, resulting in the reduction of detectable abnormal situations. As a corollary of this assertion, for example, it is not possible to determine if the brake system works properly.

Taking in account the goal, it is only necessary to use a classifier with two states, damaged or in good operation, built on the power curve of the experimental wind turbine. Of course, there is the need to allocate a tolerance, ie, the power curve is given by a ”band” as can be seen in Fig. 14. Within the two curves the equipment is still considered in “proper operation”.

One suggestion for future developments might go through considering various parts of the operating condition, namely, good condition, reasonable condition, almost damaged and broken, thus making a smoother transition between areas of good functioning and malfunctioning. The problem lies in defining the transition rates between different states of damage.

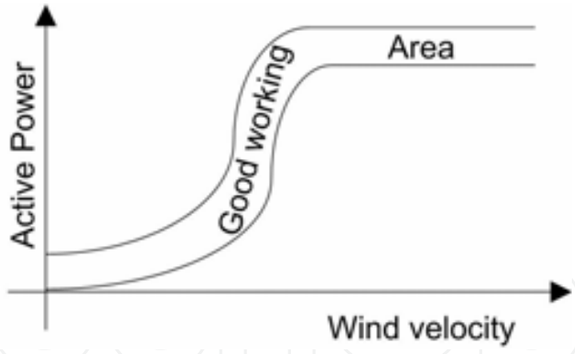


Fig. 14. Power Curve Area considered in proper operation (equation 5) (Fonseca; 2010).

3.2 Degradation curve estimation using time series

The condition of any equipment can be described using indicative variables, including the evolution of its values. A trend analysis, together with a specification of allowable limits for these parameters will permit to infer the evolution of the degradation of equipment operation. Thus an appropriate methodology is to use time series to analyze the evolution of these parameters using prediction.

Another purely statistical hypothesis would be through the Weibull distribution to characterize the reliability of equipment, supported by the calculation of probability of failure. However, this approach goes beyond the scope of this chapter because it requires a history with a considerable temporal dimension, whose data is not available.

The algorithm to forecast the degradation curve uses time series analysis techniques.

The quality of forecasting will be measured by the following error indicators: MSE (Mean Squared Error), TIC (Theil Inequality Coefficient), STD (Standard Deviation), and MAE (Mean Absolute Error).

Several methods, based on time series, were used to estimate the curve of degradation of equipment. These methods can be applied to various characteristics, such as temperature, pressure, etc.. The methods were ARRSE (Adaptive-Response-Rate Single Exponential Smoothing), ES (Exponential Smoothing), HW (Holt-Winters or Double Exponential Smoothing), HWSas (Holt-Winters with Seasonality), ESMSE (Exponential Smoothing, proposed method), ARMA (Autoregressive Moving Average), SVR-RBF (Support Vector Regression with RBF Kernels), SVR-LIN (Support Vector Regression with Linear Kernels) (details on (Fonseca et al.; 2009)). All this methods are of scientific value-added. In particular, the ”exponential smoothing” modified to adapt the parameter α. The basic method of Exponential Smoothing is driven by the following equation:

$$\hat{y}[k] = \alpha.y[k] + (1-\alpha).\hat{y}[k-1] \tag{7}$$

The value for α can be obtained minimizing the MSE, conducting to a non-linear optimization problem, usually solved by the Levenberg-Marquardt algorithm.

It is possible to continuously update α value (results will be labeled with ESMSE) and introducing an estimation of α with an implicit error, considering that the old forecasting values are independent of α; equation 8 represents the update. However, we will use the mean of equation 8 with the equation 8 substituting $\hat{y}\,[i-1] = y[i-1]$ (MSE minimization).

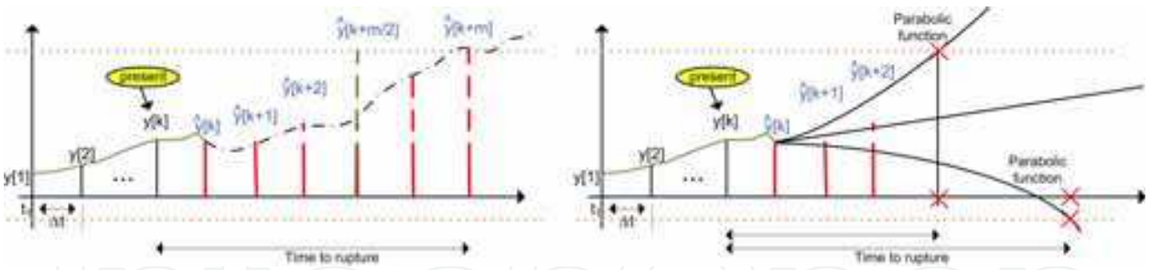


Fig. 15. Left: Prediction to rupture using time series prediction with *m* steps. Right: Using steps 1 to 3 with parabolic interpolation (Fonseca; 2010).

$$\alpha = \frac{\sum_{i=n}^{k}\left(y[i+1]-\hat{y}[i-1]\right).\left(y[i]-\hat{y}[i-1]\right)}{\sum_{i=n}^{k}\left(y[i]-\hat{y}[i-1]\right)^2},\ n \geq 2 \tag{8}$$

Fig. 15 presents two methodologies to determine the time until rupture. Both techniques use upper and lower limits, indicating the area of good functioning. At left, the technique uses the prediction until *m* steps ahead, while the technique on the right shows an alternative based on interpolation. As the number of points to predict (between 1 to 3) is lower, the reliability of prediction increases.

4. Experimental results

This section presents experimental results based on the system described in the previous sections.

4.1 SVM classification

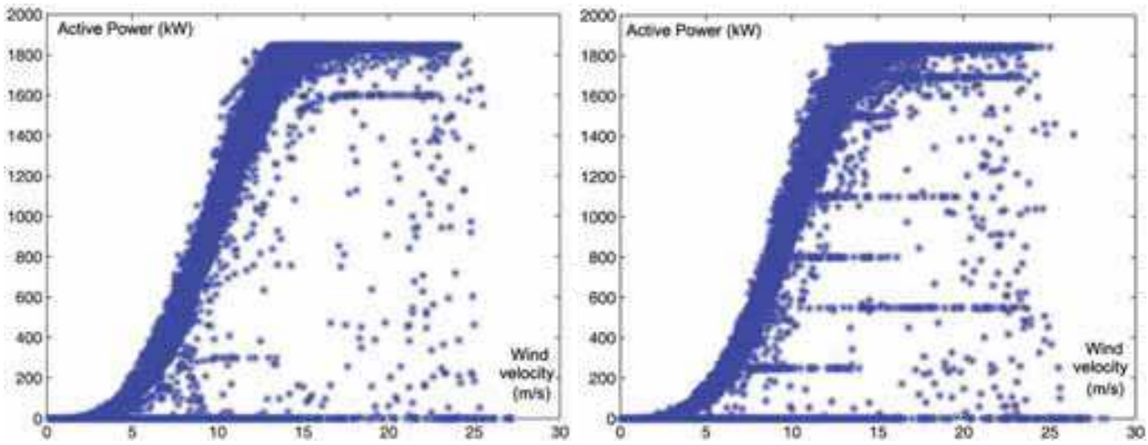
In section 3.1 was exposed a method, from a macroscopic point of view, to determine the operational status of a wind turbine. The methodology presented here uses values of several variables, which are subject to a classification of state (good / bad service). This section presents the results of this methodology that has been validated with real data from a wind turbine from north of Portugal.

Figs. 16 and 17 present the classification power curve according to the equations 5 and 6.

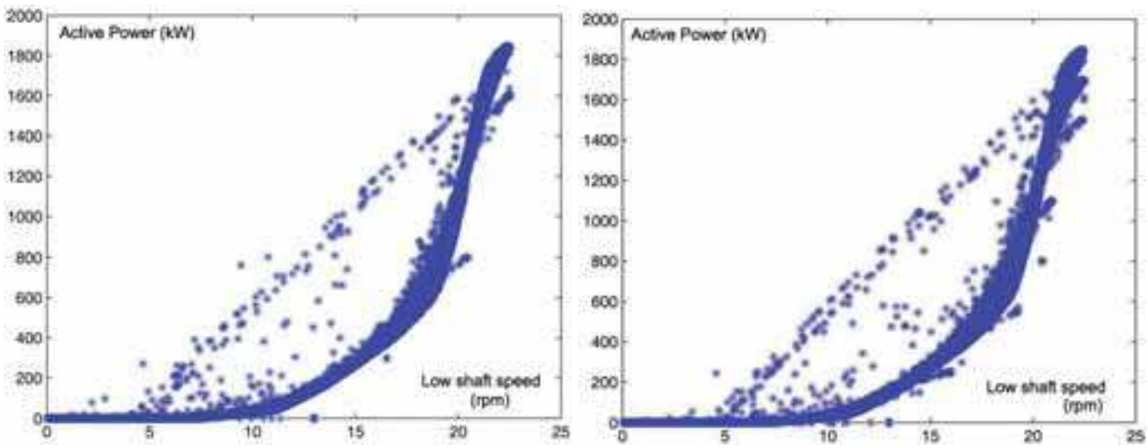
The first case, Fig. 16, uses the relationship between wind speed and produced power, which does not have into account the limitations imposed by the activation of the wind turbine brakes or by the *pitch* angle of the rotor blades. Even a malfunction, the ratio normally used does not take, as a consequence, an incorrect classification of the good points of functioning as a moment of breakdown.

The second case, Fig. 17, uses the rotor shaft velocity *versus* active power produced. This relationship includes the influences of either the pitch angle of the rotor blades or brake, or the power limitations set by the operator. As can be seen, there are many points that violate the relationship of the power curve but, in this case, are classified as good points of operation due no violation of the relationship in which they are classified (this is the best method for the data available).

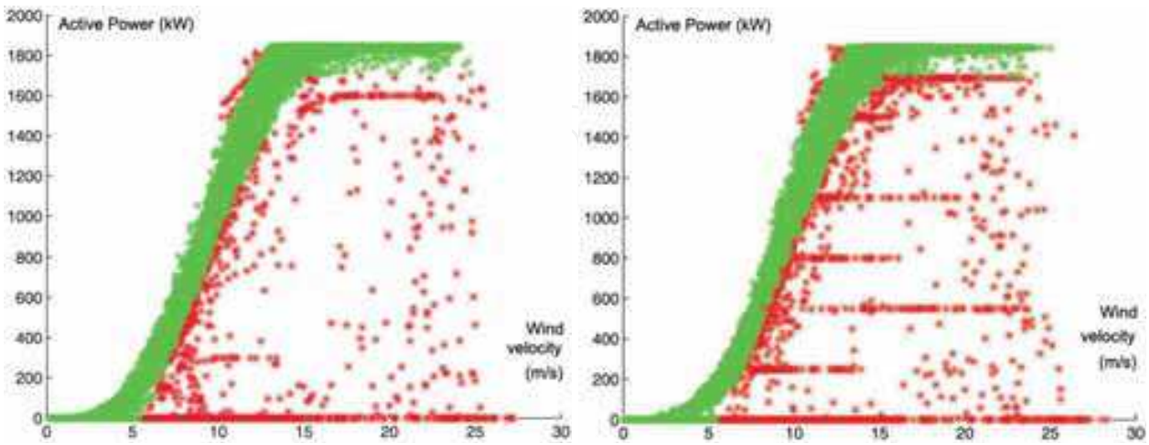
In the case of wind turbine n. 8 (Fig. 16) can also be seen that the power curve seems to work orderly in operations of 1100, 800, 550 and 220 kW. Although, there is no indication about the moments and the respective reasons in which the power production is restricted.



(a) Power Curve. Left: wind turbine n. 3; right: wind turbine n. 8;

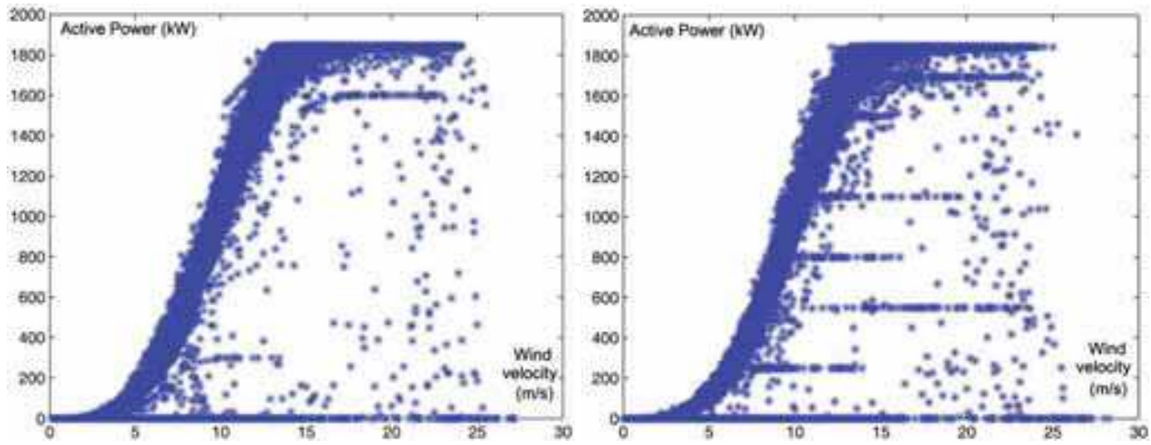


(b) Active power *versus* low shaft velocity;

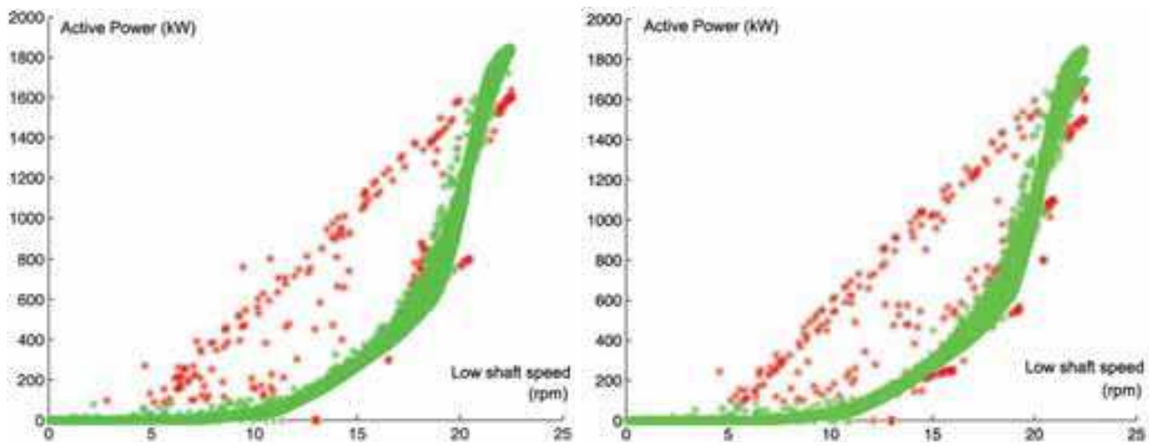


(c) SVM classification for Power curve;

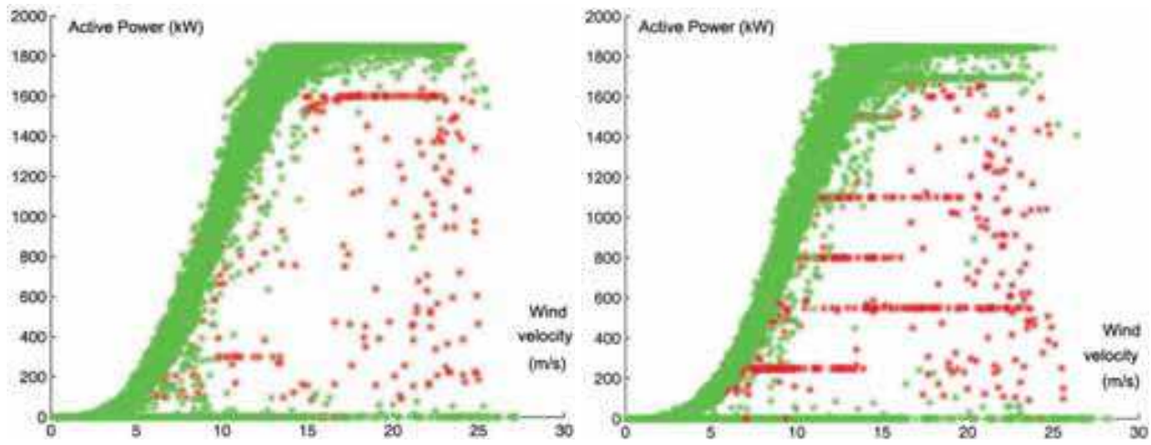
Fig. 16. SVM classification of a wind farm, according to equation 5. Red: breakdown. Green: good operation (Fonseca; 2010).



(a) Power curve. Left: Wind turbine n. 3; right: wind turbine n. 8;



(b) SVM classification for Active power *versus* low shaft speed;



(c) SVM classification of Fig. 17(b) transposed to the power curve;

Fig. 17. SVM classification of a wind farm, according to equation 6. Red: breakdown. Green: good operation (Fonseca; 2010).

Additionally, the second method classifies identically to the first. To overcome this problem it is necessary to ″relax″ restrictions on the right graph of Fig. 17(b). The data indicated belongs to a wind farm in which there is not relevant information to determine surely to what situation applies.

4.2 PTP synchronization

This section presents results for PTP Synchronization. Experiences have also been made with two modifications in the PTP protocol, in particular changing the messages Sync and Follow Up to include the value of the master clock time. Fig. 12 shows a block diagram of the text described, for the modifications made to PTP master software. The results show a faster initial convergence due to the decrease of initial error, but in subsequent periods is normal.

Matlab simulation and experimental setup has been used to collect results, and the experimental work confirms theoretical simulation. The error update of CNT_TICK_MAX used in experiments is based on the mean of last two error samples, times 0.6. A 90000 threshold has been applied on this error.

Fig. 18 outline the simulation through large seconds. It can be observed the convergence of the clocks and, after 50 timing messages, at the end, a maximum deviation of 50 ns exists. In this experiment it was considered two decimal accuracy places in parameter CNT_TICK_MAX; its final value was estimated at 434782.61 for a real value of 434782.608696. At the end, the clock slave was late 8 ns. Delay Request messages are sent randomly between 2 and 30 TSync. The same simulation, changing the precision of the parameter CNT_TICK_MAX to zero decimal places, gives a final estimation of 434783.0. The clock slave was, after simulation, late in 1783 ns. The final error was about 4000ns.

Fig. 19 shows the simulation results for two decimal precision places on CNT_TICK_MAX (similar to Experience 1), but the Slave only sends one Delay Request message at the beginning. The result is a constant absolute error in the difference between clocks. The precision is equal to experience 1, i.e., about 50 ns; CNT_TICK_MAX was 434782.61. Fig. 20 shows a real experiment after long seconds.

4.3 Time series

According to the above in section 3.2, this section presents the results obtained with different forecasting methods based on time series analysis. The methods were validated with simulated time series and a Mackey-Glass series (one of the references when studying this problem) (Teo et al.; 2001).

Fig. 21 shows the simulation results for the methods. The sign chosen to develop the tests has Gaussian noise and is described mathematically by:

$$S(t)=\begin{cases} t*0.2+gaussian\_noise*0.5, if\ t\le 200 \\ 0+gaussian\_noise*0.8, if\ 200<t\le 400 \\ t*0.2+gaussian\_noise*0.8, if\ 400<t\le 500 \\ t*0.6+gaussian\_noise*0.6, if\ 500<t\le 600 \\ t*0.4+gaussian\_noise*0.5, if\ 600<t\le 800 \\ t^2*0.004+gaussian\_noise*0.5, if\ 800<t\le 1000 \end{cases} \quad (9)$$

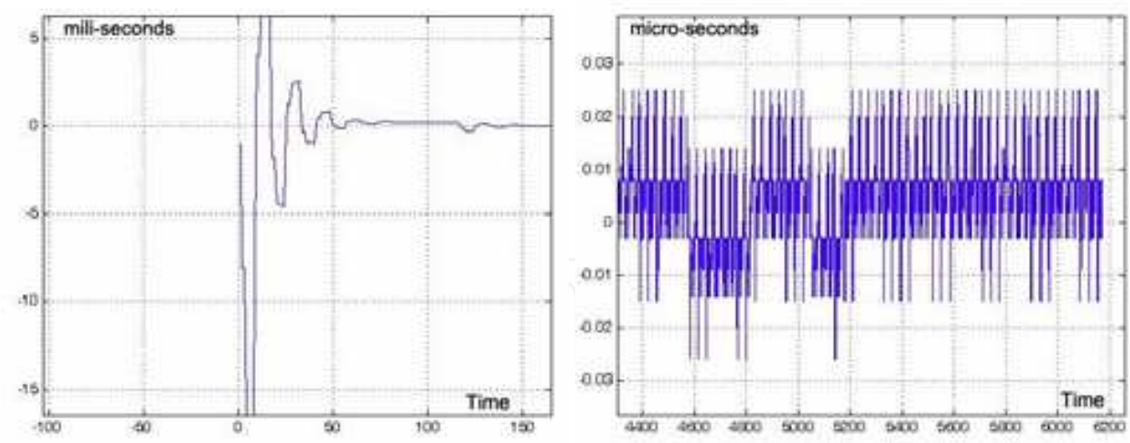


Fig. 18. Experience 1: Difference between the Master and Slave clocks (microseconds). Left: initial moments. Right: final instant (Fonseca; 2010).

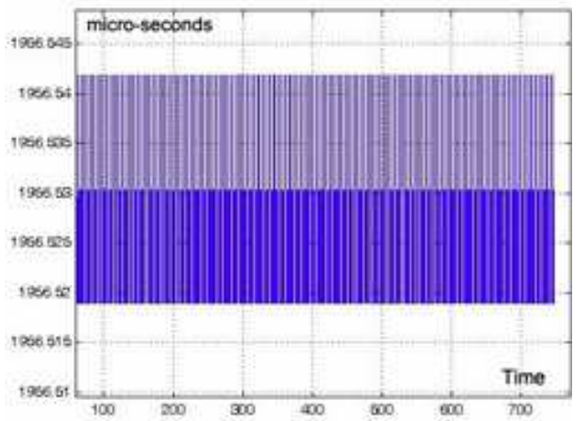


Fig. 19. Experience 2: Difference between Master and Slave clocks (Fonseca; 2010).

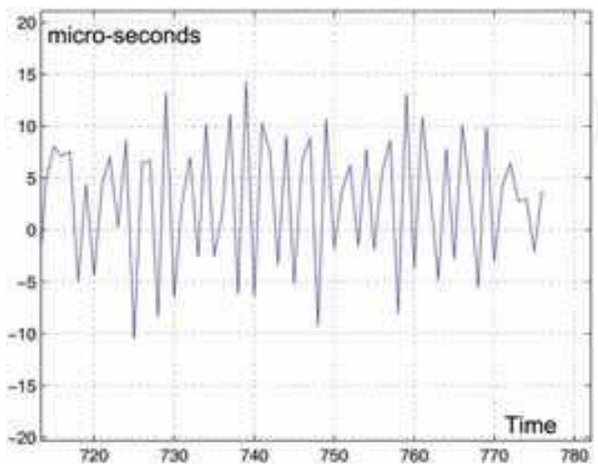


Fig. 20. Experience 3: Difference between Master and Slave clocks (microseconds), at final position, for a real system, using an ASUS Switch (Fonseca; 2010).

Alg. \ Mtr.	MSE	TIC	STD	MAE
ARRSE	17.2324	0.0486	4.1416	1.0677
ES(0.5)	20.4305	0.0531	4.5070	1.3335
HW	20.3014	0.0527	4.5005	1.1765
HWSAS	23.2613	0.0564	4.8172	1.9004
ESMSE(0.5,0)	**15.6385**	**0.0466**	**3.9500**	**0.9850**
ESMSE(0.5,5)	18.9677	0.0514	4.3471	1.1795
ARMA(2,2)	30.4376	0.0644	5.5143	1.2391
SVR-RBF	83.9986	0.1094	9.0960	3.5345
SVR-LIN	61.0381	0.0905	7.7881	1.5086

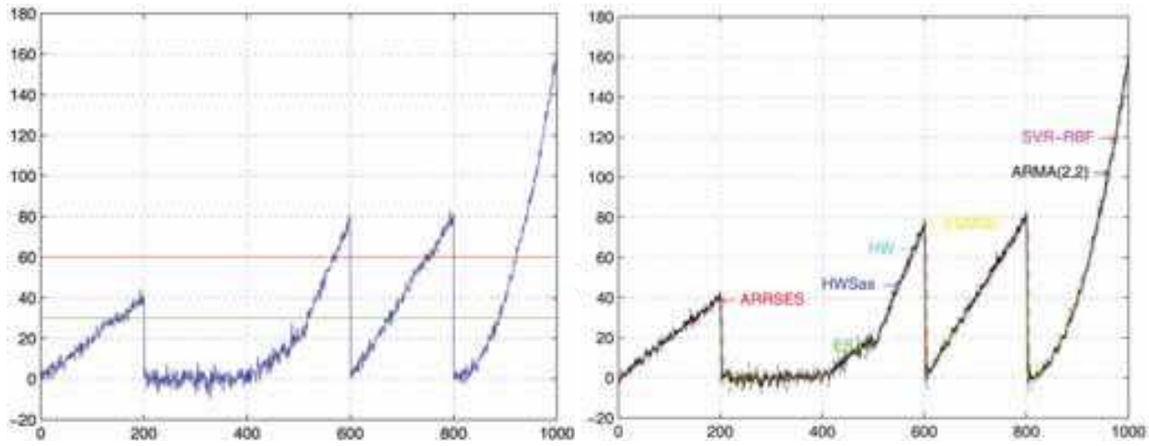
Alg. \ Mtr.	MSE	TIC	STD	MAE
ARRSE	1.3644	0.0087	0.8202	0.9596
ES(0.5)	3.7066	0.0186	2.1118	1.7807
HW	3.0305	0.0155	1.8419	1.2670
HWSAS	12.1969	0.0278	3.6379	3.2466
ESMSE(0.5,0)	**1.2975**	**0.0084**	0.8116	**0.9121**
ESMSE(0.5,5)	1.9957	0.0131	1.4572	1.2820
ARMA(2,2)	1.6631	0.0095	**0.8083**	1.0895
SVR-RBF	86.6412	0.0913	11.5840	6.5191
SVR-LIN	2.6590	0.0612	8.3743	1.5896

Table 3. Prediction results for the series of Fig. 21. Left: overall results, right: results of the range 800-1000.

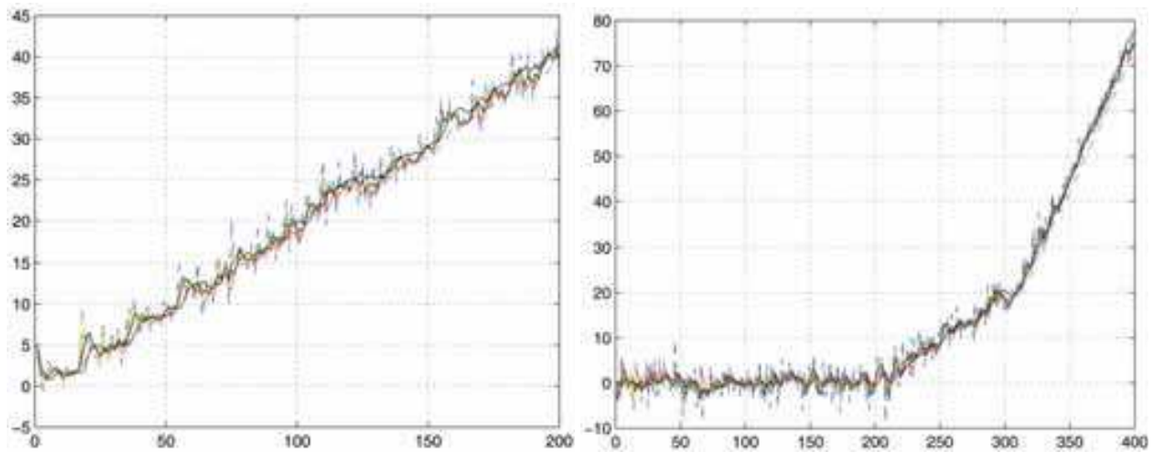
The signal purpose is to simulate the divergence of certain parameter values out of the indicated tolerance, checking how the behaviour of different algorithms is.
The parameters used in each algorithm were as follows: ARRSE(0.5,0.2); ES(0.5); HW(0.5,0.2); HWSAS(0.5,0.2, 0.2,200); ESMSE(0.5,0); ARMA(2,2); SVR-RBF(30,10) and SVR-LIN(30,10) (Fonseca et al.; 2009).
From analysis of Table 3, it appears that the ESMSE method presents the best results in most of the metrics considered, to predict the values of the next instant. However, the method does not maintain the best performance in the case of prediction of values in time later in the future and is also accompanied by most methods.
Fig. 22 shows, at left, the variation of α, when using all past samples to recalculate according to equation 8. On the right we have the same variation of α, but considering only the error of the last five samples. The interpretation of the results is simple: in the first case the α reflects the total historical errors, and when there is a high prediction error, undergoes a great change, whose influence is decisive for its value; in the other case, with a ”history” of five samples, the α varies more dynamically to compensate recent errors.
In a second simulation, from a reference series, uses a signal called differential equation of Mackey-Glass, whose time series is obtained after incorporating differential equation:

$$\frac{dx(t)}{dt} = \frac{A.x(t-\tau)}{1+x^{C}(t-\tau)} - B.x(t) \tag{10}$$

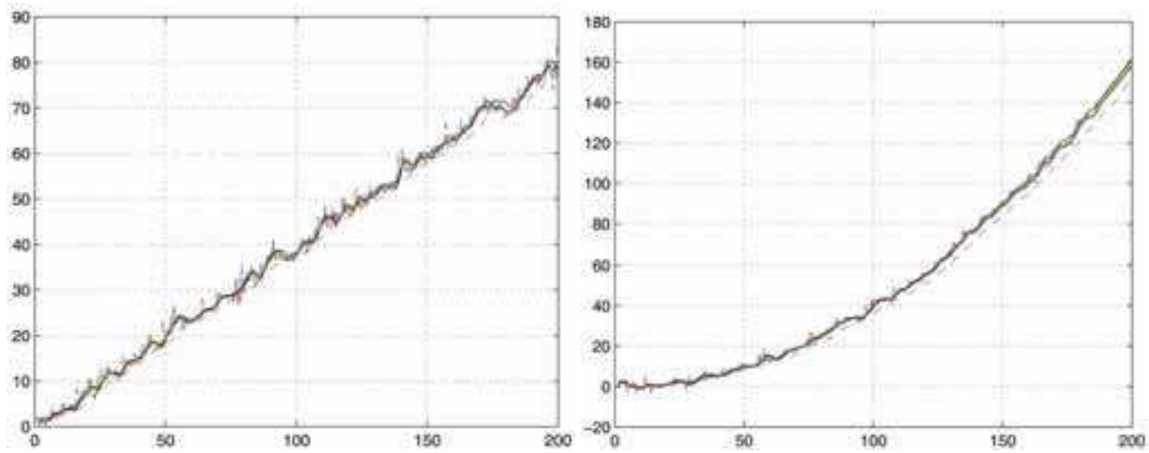
In experiments described in the literature (Teo et al.; 2001), is used A = 0.2, B = 0.1, C = 10, τ =17 and, as initial conditions, $x(0)$ =1.2 and $x(-\tau)$ =0 to $0 \leq t < \tau$ together with the Runge-Kutta method of fourth order with unit step, to calculate the series values. This differential equation was used at first-hand for analysis of blood concentration and analysis of patients with leukaemia (Teo et al.; 2001).
Fig. 23 and Table 4 present the results obtained by different methods outlined. The results of two experiences, forecasting of $\hat{y}[k]$ and $\hat{y}[k+2]$ are shown. There is deterioration in the quality of the forecasts with the increment of the steps to the future. However, in this case, unlike the previous series, the ESMSE method shows better performance for higher values to forecast the future.



(a) Left: original signal.Right: graph of the performance of various methods;



(b) Performance on the time period 0-200 (left) and 200-600 (right);



(c) Performance on the time period 600-800 (left) and 800-1000 (right);

Fig. 21. Comparative study of different methods (Fonseca; 2010).

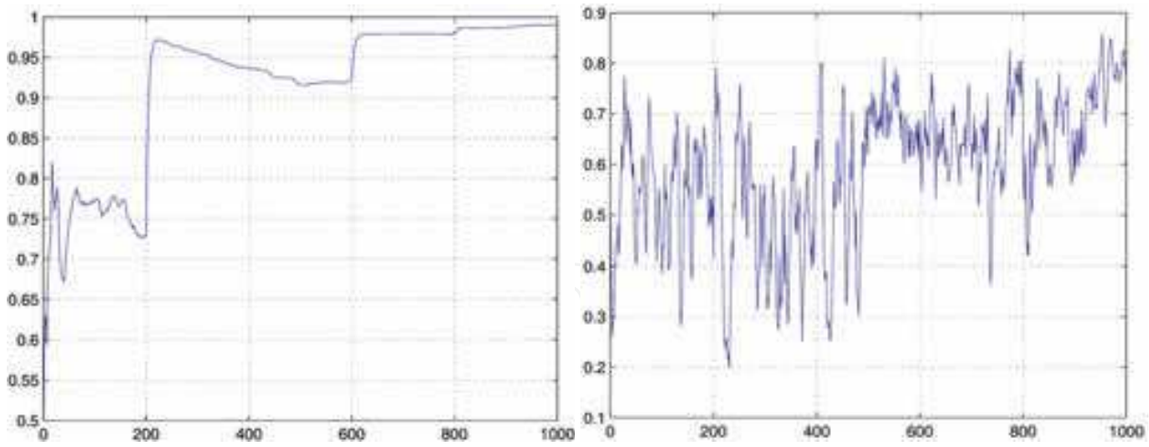


Fig. 22. Left: ESMSE(0.5,0), right: ESMSE(0.5,5). Variation of α for the prediction of the series $\hat{y}\,[k] \approx y[k+1]$ of Fig. 21 whose performance is in Table 3 (Fonseca; 2010).

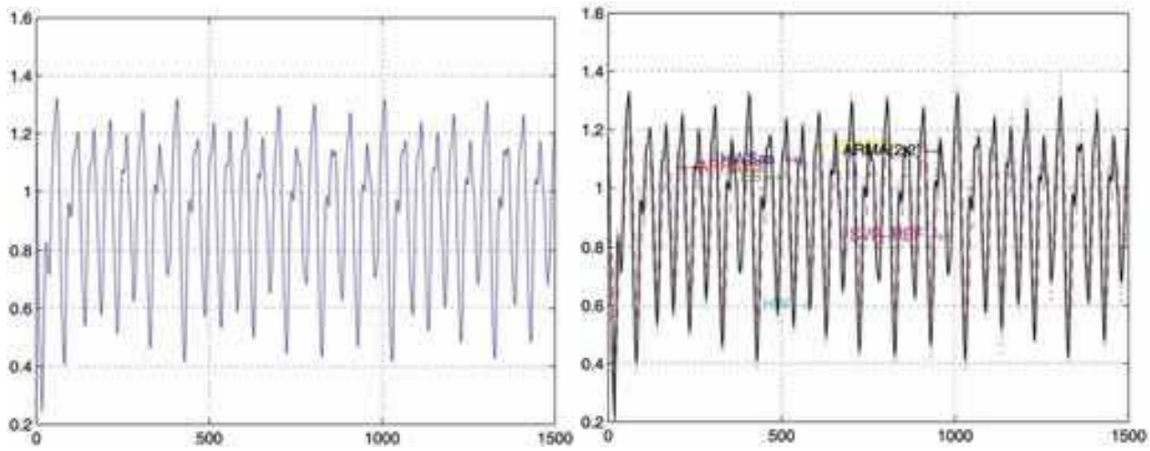


Fig. 23. Results of time series algorithms applied to the series *Mackey-Glass*: forecast $\hat{y}\,[k] \approx y[k+1]$ (Fonseca; 2010).

Alg. \ Mtr.	MSE	TIC	STD	MAE
ARRSE	0.0025	0.0261	0.0496	0.0376
ES(0.5)	0.0041	0.0339	0.0646	0.0543
HW	0.0029	0.0285	0.0544	0.0434
HWSAS	0.0037	0.0321	0.0612	0.0496
ESMSE(0.5,0)	0.0015	0.0202	0.0385	0.0320
ESMSE(0.5,5)	0.0014	0.0198	0.0378	0.0315
ARMA(2,2)	**0.0012**	**0.0185**	**0.0353**	**0.0290**
SVR-RBF	0.0300	0.0922	0.1724	0.1537
SVR-LIN	0.0104	0.0540	0.1017	0.0928

Alg. \ Mtr.	MSE	TIC	STD	MAE
ARRSE	0.0049	0.0368	0.0701	0.0482
ES(0.5)	0.0055	0.0392	0.0747	0.0614
HW	0.0052	0.0378	0.0721	0.0564
HWSAS	0.0151	0.0645	0.1234	0.0996
ESMSE(0.5,0)	**0.0026**	**0.0272**	**0.0519**	**0.0412**
ESMSE(0.5,5)	0.0026	0.0270	0.0516	0.0407
ARMA(2,2)	0.0035	0.0313	0.0594	0.0433
SVR-RBF	0.0470	0.1155	0.2161	0.1903
SVR-LIN	0.0183	0.0713	0.1353	0.1137

Table 4. Prediction results for the *Mackey-Glass* series. Left: forecast to $\hat{y}\,[k] \approx y[k+1]$; right: forecast to $\hat{y}\,[k+2] \approx y[k+3]$.

5. Conclusions

This chapter described a wind maintenance system with all the components, from software to hardware, in which the main objective is to lower maintenance costs through on-condition maintenance based on on-line data acquisition, and the use of open-source software and low cost hardware. An acquisition synchronization system was also presented using PTP hardware with time stamping facility and the related control system. Finally were briefly presented two algorithms to perform on-condition monitoring based on SVM and Time Series Analysis. One proposed method for time series analysis was modified. The ESMSE method is suitable to use on degradation estimation and to be used also on microcontrollers too.

6. References

Caselitz, P. and Giebhardt, J. (2002). Advanced condition monitoring for wind energy converters, *Proc. European Wind Energy Conference, Nice, France*.

Cauwenberghs, G. and Poggio, T. (2000). Incremental and decremental support vector machine learning, *NIPS*, pp. 409–415.

Correll, K. and Barendt, N. (2006). Design considerations for software only implementations of the IEEE 1588 precision time protocol, *In Conference on IEEE 1588 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*.

Dunkels, A. (2003). Full tcp/ip for 8-bit architectures, lwip - light-weight ip implementation, *In Proceedings of the first international conference on mobile applications, systems and services (MOBISYS 2003)*.

Durstewitz, M., Hahn, B. and Rohrig, K. (2005). *Advanced Maintenance and Repair for Offshore wind farms using fault prediction and Condition Monitoring Techniques,* E.U. final report of project NNE5/2001/710,
http://www.iset.uni-kassel.de/osmr/.

Eaton, J. W. and Gentleman, R. (2010). Octave and R, *Scientific open-source software*. www.gnu.org/software/octave/ and
www.r-project.org/.

Fonseca, I. (2010). *Maintenance of Wind turbines using IP networks (Phd Thesis in Portuguese)*, Porto University,
http://www.fe.up.pt.

Fonseca, I., Farinha, J. T. and Barbosa, F. M. (2008). A computer system for predictive maintenance of wind generators, *Proceedings of the 12th WSEAS International Conference on COMPUTERS*, WSEAS, Heraklion, Greece, pp. 928–933.

Fonseca, I., Farinha, J. T. and Barbosa, F. P. M. (2009). On-condition maintenance for wind turbines, *IEEE Bucharest Power Tech Conference*.

Group, N. W. (2010). SNTP, *Simple Network Time Protocol*.
www.cis.udel.edu/˜mills/ database/rfc/rfc4330.txt.

Hameed, Z., Ahn, S. and Cho, Y. (2010). Practical aspects of a condition monitoring system for a wind turbine with emphasis on its design, system architecture, testing and installation, *Renewable Energy* 35(5): 879 – 894.

Hameed, Z., Hong, Y., Cho, Y., Ahn, S. and Song, C. (2009). Condition monitoring and fault detection of wind turbines and related algorithms: A review, *Renewable and Sustainable Energy Reviews* 13(1): 1 – 39.

Hardware and Software (2010). Open-source software and commercial hardware, *National, Vaisala, Codegear, OpenSSL*.
ni.com;
vaisala.com;
codegear.com;
openssl.org;.

Joeware (2010). Program for context switching, *CPAU*.
www.joeware.net/freetools/ tools/cpau/index.htm.

Joseph, F. and Gutowski, T. (2008). TurbSim: Reliability-based wind turbine simulator, *IEEE International Symposium on Electronics and the Environment, San Francisco USA*. http://web.mit.edu/ebm/www/Publications/Joe%20Foley% 20IEEE%202008.pdf.

Kecman, V. (2001). *Learning and Soft Computing*, MIT Press, Cambridge.
http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=3720.

Luminary, C. (2010). ARM cortex processors, *Micro-controllers*.
www.luminarymicro.com.

MicroChip (2010). Micro-controllers, *Microchip Technology Inc.*
http://www.microchip. com.

Open-source (2010). Open-source software, *Slackware, FreeBSD, PostgreSQL, Apache and PHP*.
www.slackware.com;
www.freebsd.org;
www.postgresql.org;
www. apache.org;
www.php.net.

PTP (2010). National Institute of Standards and Technology, *Precision Time Protocol*.
http: //ieee1588.nist.gov/, and an implementation of PTPD in
http://ptpd. sourceforge.net/.

Scheffer, C. and Girdhar, P. (2004). *Practical Machinery Vibration Analysis and Predictive Maintenance*, Elsevier,
http://www.elsevier.com/wps/find/bookdescription.editors/702923/ description#description.

Suykens, J., Gestel, T. V., Brabanter, J. D., Moor, B. D. and Vandewalle, J. (2002). *Least Squares Support Vector Machines*, World Scientific Publishing Co., England. http://books.google.pt/books?id=g8wEimyEmrUC&printsec= frontcover&source=gbs_v2_summary_r&cad=0#v=onepage&q&f=false.

Technologies, Z. (2010). Php-webservices, *PHP manual pages*.
http://www.php.net/ manual/en/refs.webservice.php.

Teo, K. K.,Wang, L. and Lin, Z. (2001). Wavelet packet multi-layer perceptron for chaotic time series prediction: Effects of weight initialization school of electrical and electronic engineering, *Springer-Verlag Berlin Heidelberg LNCS 2074*, pp. 310–317.

Wago and Beckhoff (2010). Programmable logic controller, *Automation companies*.
http:// www.wago.us/;
http://www.beckhoff.com/.
Zhizheng, L. and YouFu, L. (2009). Incremental support vector machine learning in the primal and applications, *Neurocomput.* 72(10-12): 2249–2258.

Data Acquisition

Edited by Michele Vadursi

ISBN 978-953-307-193-0
Hard cover, 344 pages
Publisher Sciyo
Published online 28, September, 2010
Published in print edition September, 2010

The book is intended to be a collection of contributions providing a birdâ€™s eye view of some relevant multidisciplinary applications of data acquisition. While assuming that the reader is familiar with the basics of sampling theory and analog-to-digital conversion, the attention is focused on applied research and industrial applications of data acquisition. Even in the few cases when theoretical issues are investigated, the goal is making the theory comprehensible to a wide, application- oriented, audience.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

F. Maciel-Barbosa, Inácio Fonseca and J. Torres Farinha (2010). Wind Farms Sensorial Data Acquisition and Processing, Data Acquisition, Michele Vadursi (Ed.), ISBN: 978-953-307-193-0, InTech, Available from: http://www.intechopen.com/books/data-acquisition/wind-farms-sensorial-data-acquisition-and-processing-

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen