Desarrollo de un sistema de monitoreo continuo y gestión integral de defectos en motores de inducción

Faults detection and remote monitoring system for induction motors

José Ignacio Terra[1] *Juan Pablo Fossati*[2]

Entregado: Agosto 2010 Aceptado: Septiembre 2010

Resumen: *Este artículo se enmarca dentro del proyecto realizado por el CITEM de la Universidad de Montevideo, para la empresa ALFEX S.A., siendo el objetivo del mismo, el desarrollo de un sistema automatizado de monitoreo remoto y diagnóstico de defectos en motores de inducción1. Este sistema implementa la metodología de diagnóstico basada en MCSA (Análisis de la Firma de Corriente del Motor), que permite detectar indicios de fallos tales como rotura de barras del rotor, daños en los rodamientos, excentricidad del rotor y cortocircuitos en los bobinados del estator 2 3.*
A su vez, permite a los usuarios verificar el estado de sus motores vía web y dispara distintos tipos de alarmas en caso de detectar indicios de fallo.

Palabras claves: motores de inducción, monitoreo y diagnóstico, web, tratamiento de señales

Summary: *This article describes the design and implementation of a fully automated system for faults detection and remote monitoring of induction motors, developed by CITEM at Montevideo University, for the company ALFEX S.A. The system uses a methodology for faults detection based on MCSA (Motor Current Signature Analysis), which is sensitive to several faults such as: broken rotor bars, bearing damages, rotor eccentricity and shorted turns. Furthermore, the system triggers different kinds of alarms whenever a fault is detected and its website allows users to check the state of each motor.*

Key words: induction motors, remote monitoring, automated diagnosis, signal processing

1. Introducción

1.1. Antecedentes.-En una instancia previa se desarrolló conjuntamente con la empresa ALFEX S.A. una metodología de diagnóstico de fallas en motores asíncronos de inducción, realizándose numerosas pruebas a nivel de laboratorio e industrial, la cual fue presentada en la edición número 6 de esta memoria 4. La metodología presentada en ese artículo es el punto de partida para el desarrollo de este proyecto.

[1] Ing. Telemático; Docente de Física e Investigador; Universidad de Montevideo. jterra@um.edu.uy
[2] Ing. Industrial; Docente de Física e Investigador; Universidad de Montevideo. jfossati@um.edu.uy

1.2. Motivación.-Los motores de inducción de más de 100 HP son, en la mayoría de los casos, equipos imprescindibles para llevar a cabo un proceso productivo. Esto implica que al momento de la ocurrencia de un desperfecto que tenga como consecuencia la parada no programada del equipo y el proceso involucrado, se incurre en una serie de costos elevados, más allá de los de reparación o reemplazo, y además dificultan el cumplimiento de las exigencias de calidad. En la actualidad, existen numerosas metodologías para el monitoreo y detección de defectos en este tipo de motores como 5, 6, 7, 8, 9 y 10, entre otros. Pero ninguno de éstos reúne en un sistema automatizado y versátil las capacidades de:

- monitorear continuamente y de forma remota. Únicamente la sensorización debe ser instalada en planta
- detección predictiva de los cuatro tipos de fallos más comunes
- monitoreo del grado de avance del defecto
- permitir a los usuarios acceder a los diagnósticos vía web
- enviar alarmas de defectos vía correo electrónico y mensajes de texto (SMS)

La detección predictiva de los fallos permite actuar de forma programada para detener la máquina en un momento oportuno y realizar la reparación necesaria.

2. Desarrollo.-Se desarrolló un sistema automatizado que monitorea a todos los motores almacenados en la base de datos. El mismo obtiene en tiempo real la velocidad de giro del motor, la frecuencia de alimentación y la corriente de fase, para luego estudiar las componentes de frecuencia asociadas a los distintos defectos. El monitoreo se hace procesando continuamente, lecturas de la corriente consumida por cada motor. Cada lectura es un conjunto de muestras tomadas a una frecuencia de muestreo de 1 kHz durante diez segundos (10 s), resultando en un total de diez mil muestras por lectura.

Cada vez que la aplicación detecta un nuevo indicio de falla, envía un mensaje de texto de alarma a todos los usuarios registrados en la empresa a la que pertenece el motor, informando sobre la ocurrencia del mismo. Además, cuando las muestras no pueden ser procesadas de manera satisfactoria, se envía un correo electrónico con las muestras a los usuarios registrados como "Expertos MCSA" para que éstos puedan definir qué es lo que está ocurriendo con el motor en cuestión.

A su vez, se implementaron servidores de base de datos y web. Permitiendo ingresar datos de las empresas, secciones, motores, dispositivos de muestreo y usuarios en la base de datos, vía web. Asimismo, este medio permite a los usuarios acceder al estado de los motores correspondientes a la empresa para la que trabajan o a la totalidad de ellos, dependiendo del tipo de usuario que los solicite.

Por último, se diseñó y construyó un dispositivo de muestreo que permite acceder a las muestras de corriente a través de internet. La *Figura 1* muestra la vista general del sistema implementado.

ISSN 1510-7450

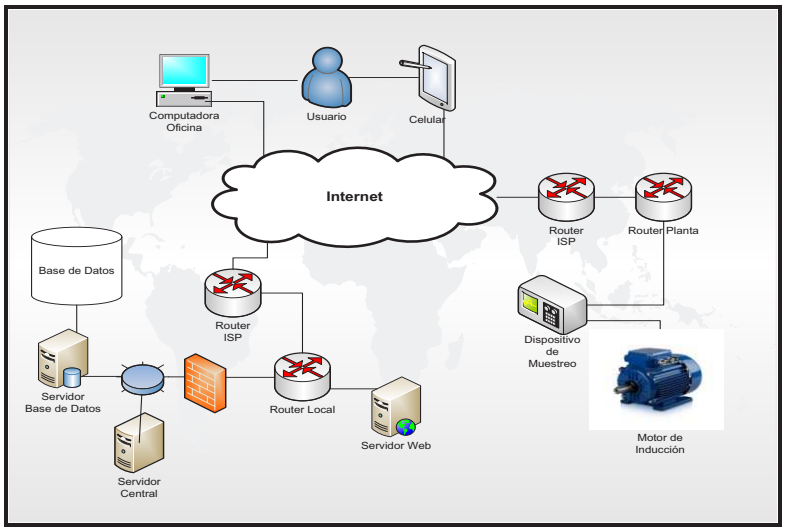

Figura 1. Vista general del Sistema.

2.1. Arquitectura.- Si bien en el prototipo, los tres servidores (de base de datos, aplicaciones y web) se ejecutan en una misma computadora, los mismos fueron diseñados para ejecutarse en una arquitectura de servidores distribuidos de modo de ofrecer una mejor calidad de servicios. La aplicación central fue implementada utilizando el lenguaje de programación *Java*. Se eligió sistema de gestión de base de datos orientas a objetos (OODBMS) debido a la complejidad de los datos que almacena el sistema y el costo, en términos de procesamiento, que esto implica si se opta por el modelo relacional. Se utilizó el servidor web Apache Tomcat y las páginas web se implementaron con el lenguaje *JSP*.

Los dispositivos de muestreo se construyeron con un módulo *RabbitCore*, que se programa con el lenguaje *Dynamic C*, y pinzas amperimétricas para que la instalación del mismo sea fácil y no requiera la parada de los motores. El costo de construcción de estos dispositivos fue un factor muy importante a la hora del diseño, debido a que se requiere uno por cada motor. La aplicación central accede a las lecturas realizadas por cada dispositivo a través de internet utilizando el protocolo *FTP*.

Los usuarios únicamente necesitan un navegador web para acceder a todas las funcionalidades del sistema.

2.1.1. Aplicación Central.-La aplicación central se encarga de gestionar constantemente a todos los dispositivos de muestreo que se encuentran en línea. En régimen, el mismo solicita periódicamente lecturas a los dispositivos, procesa las mismas, almacena los resultados en la base de datos, detecta si hay algún indicio de fallo y, en caso afirmativo, dispara la alarma correspondiente al tipo y magnitud del defecto detectado.

Resumidamente, la metodología de diagnóstico utilizada se basa en analizar periódicamente, en el dominio de la frecuencia, la corriente consumida en cada fase del motor. Cada defecto induce componentes específicas en la corriente del estator, cuyas magnitudes y frecuencias difieren según el nivel de asimetría (magnitud) y naturaleza del defecto que las origina. De este modo, para detectar la ocurrencia de defectos, la aplicación analiza cada una de estas componentes específicas y las compara con módulos de referencia almacenados en la base de datos. Cuando el módulo de la nueva componente supera al de referencia, multiplicado por un coeficiente que

varía según el tipo de defecto, el sistema detecta el indicio de falla, almacena el diagnóstico en la base de datos y dispara la alarma correspondiente.

Una de las herramientas utilizadas en la metodología MCSA es la Transformada Rápida de Fourier (FFT), que es el algoritmo más conocido, debido a su eficiencia, para calcular la Transformada Discreta de Fourier (DFT) a partir de un vector de muestras. La DFT es un vector que contiene los módulos de las componentes de frecuencia de la señal muestreada. En principio, la metodología de detección de fallas no fue desarrollada pensando en un sistema totalmente automatizado, sino como una herramienta que proporcione la información necesaria para que un técnico realice el diagnóstico. Al pasar a la automatización, hay características de las DFT obtenidas a partir de las muestras, que dificultan precisamente dicha automatización. Una de estas características es que la potencia del armónico principal afecta al resto de los armónicos cercanos, agregando una componente constante a los mismos. En consecuencia se hace más difícil establecer un umbral a partir del cual se busquen las componentes específicas, dado que el umbral debería variar según la corriente consumida por el motor. Para solucionar este problema, se agregó un paso a la metodología. Luego de utilizar el armónico principal para calcular el consumo de corriente y frecuencia de alimentación exacta, se procesan nuevamente las muestras para eliminar el armónico de la transformada. La *Figura 2* muestra las DFT con y sin la presencia del armónico fundamental en el rango de frecuencias entre 13 Hz y 37 Hz aproximadamente. En los círculos A y B se muestran otras dos ventajas de este cambio. En A, se observa que el armónico principal afecta el módulo de las otras componentes de frecuencia. Dos componentes que parecían tener el mismo módulo, resultaron en una con aproximadamente la mitad del módulo que la otra. Mientras que en B, con la nueva DFT se puede distinguir una componente con baja potencia que no era posible distinguir con la DFT original. La aplicación central trabaja solamente con la DFT modificada para analizar las componentes específicas de defecto.

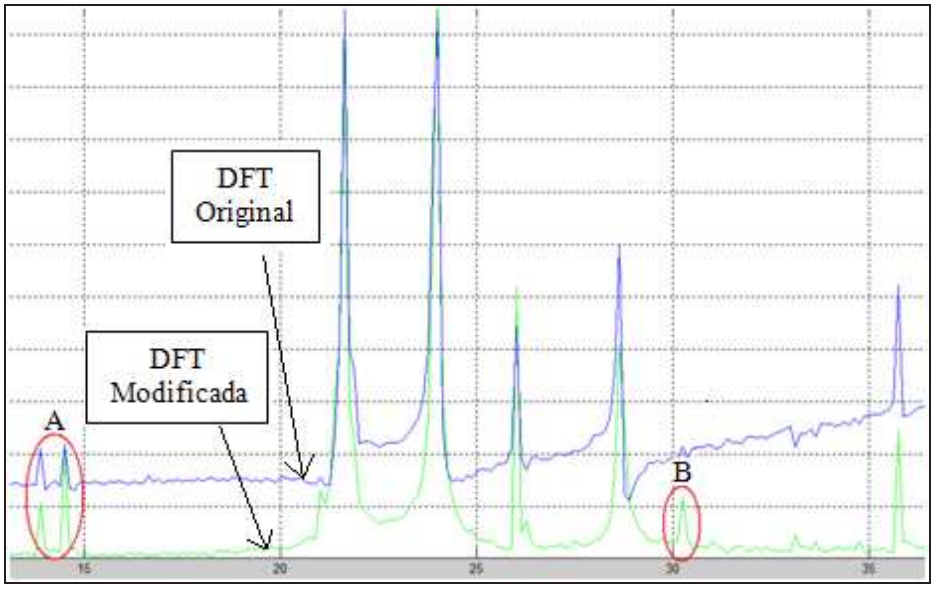

Figura 2. Diferencias entre DFT con y sin la presencia del armónico fundamental

A su vez, todos los valores se normalizan con el armónico fundamental, para que los módulos de las componentes específicas no dependan del consumo de corriente, que varía según la carga mecánica.

Para aumentar la versatilidad del sistema en términos de modelos de motores que pueden ser monitoreados y, además, reducir los costos de instalación tanto de tiempo como de dinero, se

ISSN 1510-7450

implementó un algoritmo para el cálculo de los módulos de referencia asociados a cada uno de los componentes específicos de frecuencia mencionados anteriormente. Las referencias deben ser calculadas únicamente al momento de la instalación y luego de cada reparación.

La *Figura 3* representa el flujo de procesos de la aplicación central trabajando en régimen. Es decir, cuando ya se calcularon todas las componentes de referencia y el motor se encuentra en buen estado.

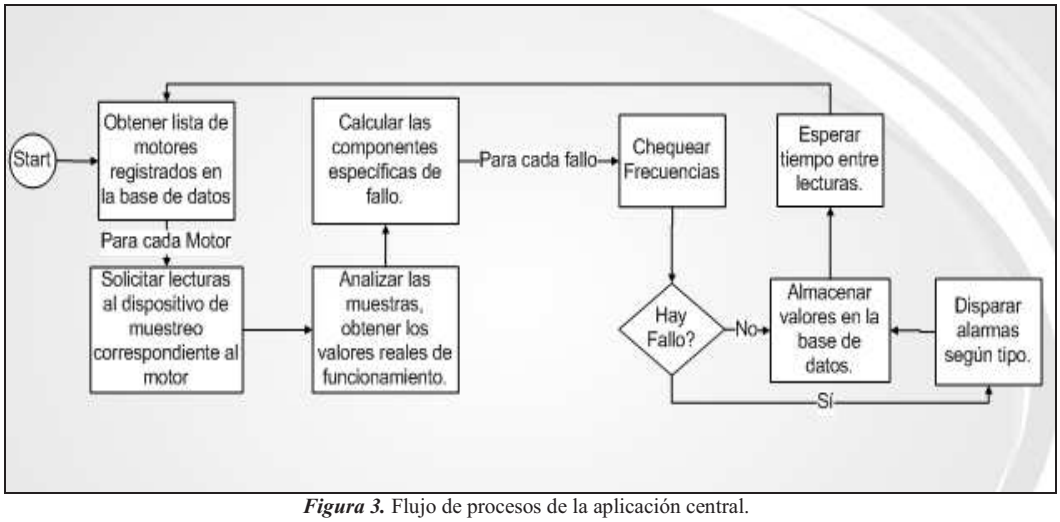

Figura 3. Flujo de procesos de la aplicación central.

2.1.2. Dispositivo de Muestreo.-Para elegir el equipamiento necesario para construir el dispositivo se tuvo en cuenta que el mismo incluyera las siguientes funcionalidades:

- pinzas amperimétricas, imprescindibles para que la técnica sea no invasiva
- conversor Analógico/Digital (A/D) para convertir la señal de voltaje proveniente de las pinzas
- conexión de red y soporte de protocolos de transferencia de datos
- capacidad de muestrear a altas frecuencias (1000 Hz)
- capacidad de almacenamiento para los archivos de lectura

La metodología de diagnóstico fue aplicada anteriormente obteniendo las muestras con una consola de entradas analógicas conectadas a conversores A/D. Esta consola se conecta a la placa madre de cualquier computadora personal a través de un slot PCI y funciona con un software propietario para procesar las muestras. Sin embargo, lo anterior tiene un costo elevado (del orden de los miles de dólares, sin contar las pinzas), teniendo en cuenta que se precisa uno por motor monitorizado por el sistema.

Considerando lo anterior y la complejidad del diseño y construcción de una tarjeta compatible con el estándar PCI en conjunto con el software que la soporte, se optó por estudiar opciones de microcontroladores en lugar de una computadora personal con tarjeta externa para el muestreo. Se eligió un módulo de la línea *Rabbit* 11 por razones de costo, calidad, lenguaje de programación (modificación de C) y el soporte de todas las funcionalidades listadas anteriormente. Específicamente, se optó por el modelo *RCM4000* (Módulo RabbitCore 4000), que soporta una serie de características y conectividad a internet que permiten diseñar sistemas integrados que los usuarios pueden monitorizar remotamente.

A la hora del diseño, se partió de las especificaciones de muestreo establecidas por la metodología de detección de fallas, fijadas en 5000 Hz durante 8 segundos. Éstas resultan en 40000 muestras que deben ser almacenadas en memoria secundaria. Teniendo en cuenta la capacidad de almacenamiento de los microcontroladores, se realizaron pruebas comparando las DFTs obtenidas con distintas frecuencias de muestreo. La *Figura 4* muestra las DFTs obtenidas con frecuencias de muestreo 5000 Hz y 1000 Hz. La metodología indica que algunos defectos pueden detectarse con componentes específicas dentro de la ventana de 0 - 500 Hz, en la práctica se enfoca el análisis en las frecuencias menores a 200 Hz. Por esto, según el *Teorema de Muestreo*, la frecuencia de muestreo mínima posible es igual a 1000 Hz si se quiere analizar frecuencias de hasta 500 Hz. A su vez, es al menos cinco veces superior al valor de las frecuencias más estudiadas, mejorando la apreciación de la DFT para esos valores. En la *Figura 4* se puede ver que las diferencias entre las DFTs obtenidas con las distintas frecuencias, solamente pueden ser apreciadas en los niveles bajos o "ceros" del espectro, los cuales no son considerados para el análisis.

Finalmente, puesto que con este cambio se redujo sustancialmente el tamaño de la lectura, es posible aumentar el tiempo de muestreo con el objetivo de mejorar la apreciación de la DFT y su resolución frecuencial. Se optó por un tiempo de muestreo de 10 segundos debido a que resulta en una resolución frecuencial de 0.10 Hz.

Con estos cambios, la lectura de cada fase de los motores se redujo de 40000 a 10000 muestras sin pérdida de información relevante.

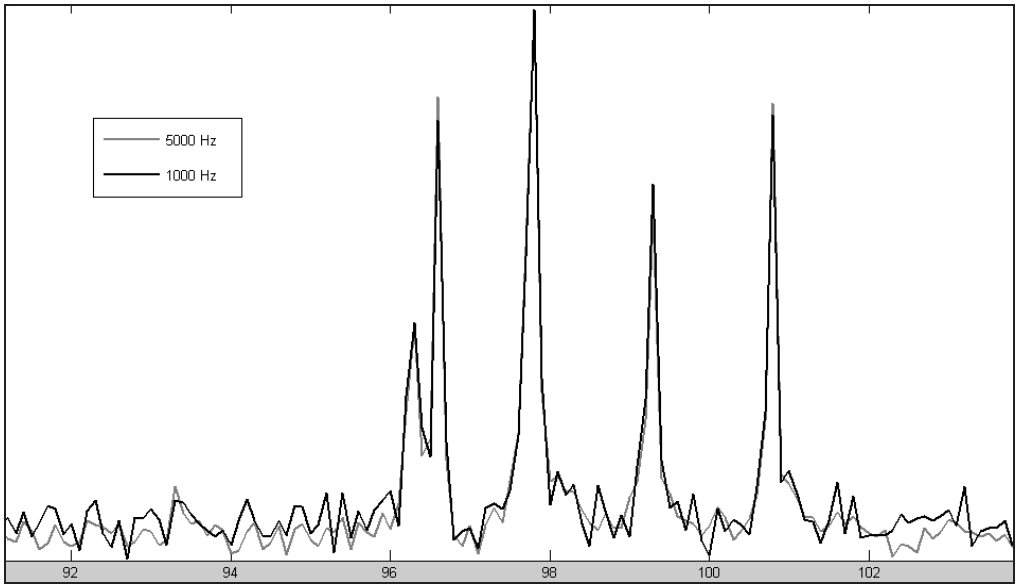

Figura 4. DFTs obtenidas con frecuencias de muestreo 1000 Hz y 5000Hz.

El programa del dispositivo fue diseñado para que éste ejecute un servicio FTP que permita acceder a los archivos almacenados en la memoria *Nand Flash* con el sistema de archivos FAT16 y para que realice una lectura de la corriente por minuto.

Como en cualquier sistema embebido, los programas implementados para el RCM4000 interactúan directamente con el hardware, es decir que no hay ningún sistema operativo que lo gestione. Esto imposibilita ejecutar dos programas simultáneos. Asimismo, Dynamic C[3] no

[3] Lenguaje de programación para la línea de microcontroladores *RabbitCore*

ISSN 1510-7450

soporta la programación *multihilos*. Lo que sí ofrece son los "*costates"* (o *coestados*). Los coestados son estructuras para programar distintas tareas (o estados), los mismos se agregan dentro de un bucle y existen una seria de condiciones para que éstos se detengan, resuman, "cedan el paso" a la ejecución de otros coestados, entre otras12.

En el software del dispositivo se implementaros dos coestados: el primero procesa las peticiones que llegan al puerto FTP (21 por defecto). Cada vez que se comienza a procesar una operación, bloquea al otro coestado hasta finalizar (ejemplo, si está enviando un archivo a la aplicación central); el segundo, es el que llama al método "*muestrear()*", a diferencia del anterior, este sí tiene una condición que lo bloquea temporalmente y es la de "*waitfor(DelaySec(40))*", dejando inactivo al coestado durante los segundos necesarios (en este caso 40) para que se realice una lectura por minuto.

Las lecturas se almacenan en archivos con el nombre "lecXXX.txt" (XXX es un entero de tres cifras que se autoincrementa cada vez que se realiza una lectura), que cumplen con el siguiente formato:

"*dd/mm/aaaa hh:mm:ss-->xxxx, xxxx, xxxx, xxxx,……., xxxx, xxxx, xxxx,-->ttttt"*

- *dd*. Día del mes
- *mm*. Mes del año
- *aaaa*. Año
- *hh*. Hora
- *mm*. Minuto
- *ss*. Segundo
- *xxxx*. Valor de la muestra antes de aplicarle las constantes de calibración (0-2047)
- *ttttt*. Tiempo (en milisegundos) en que se tomaron las muestras. Durante el período de experimentación, se utilizó para calcular la frecuencia de muestreo efectiva. Actualmente, es un dato redundante que se utiliza para corroborar que la lectura se ejecutó exitosamente.

Otra restricción del RCM4000, es que el conversor A/D solamente convierte voltajes positivos[4] y su precisión depende del rango de valores de entrada posibles. El rango debe ser configurado en el programa antes de que se tome la medida, dependiendo del rango de corriente que se quiere medir y la constante de conversión de la pinza amperimétrica. A su vez, puesto que las pinzas generan una señal de voltaje directamente proporcional a la corriente medida, los semiciclos negativos generan voltajes negativos. Por ende, se agregó un circuito simple de acoplamiento entre las pinzas y la entrada del conversor. Este circuito le suma a la señal proveniente de las pinzas, una componente de continua cuyo valor es igual al valor medio del rango de conversión del RCM4000.

2.1.3. Servidor Web.-Las páginas web son la única parte gráfica del sistema, a través de la cual, dependiendo del tipo, los usuarios pueden ingresar diversos parámetros, entre ellos empresas, secciones, motores, dispositivos de muestreo y nuevos usuarios a la base de datos. A su vez, permite consultar el diagnóstico de los motores y confirmar la recepción de alarmas.
El puerto por defecto del servidor *Apache Tomcat* es el 8080, para que la dirección web del sistema sea más familiar, se debe configurar el DNAT[5] (Traducción de la Dirección de Red de Destino) en el router local.

[4] También convierte valores de corriente entre 4 - 20 mA.
[5] Mecanismo que traduce la dirección IP y puerto públicos del router a otra dirección IP y puerto de la red privada. Permitiendo que un host perteneciente a la red privada, sea visible desde internet.

Al momento de creación de la base de datos, se agrega localmente el primer usuario de tipo administrador. Una vez ingresado éste, todos los datos serán ingresados a través del servicio web.

2.1.4. Servidor de Base Datos.- El sistema requiere una base de datos (DB) para almacenar los datos de las empresas, secciones, motores, dispositivos de muestreo y usuarios. A su vez, periódicamente[6] la aplicación central procesa las lecturas de cada uno de los dispositivos de muestreo y almacena los valores relevantes en la DB.

En la actualidad, la arquitectura más común para soportar el acceso concurrente a información por parte de distintas aplicaciones (es el caso de este sistema), se implementan con la arquitectura Cliente-Servidor con un servidor de base de datos. La mayoría de estos sistemas utilizan RDBMS (Sistemas de Gestión de

Bases de Datos Relacionales), que almacenan los datos principalmente en tablas (*relaciones*), mientras que el software está implementado con un leguaje orientado a objetos. Esto genera una ineficiencia por el hecho que se debe convertir (*mapear)* los objetos con los que trabaja el software a filas (*tuplas)* del modelo relacional; probablemente en más de una tabla con el uso de claves foráneas. La diferencia entre ambos modelos de datos, conocida como "*impedance mismatch*", implica una pérdida de performance del sistema en conjunto debido a la necesidad del mapeo.

Se optó por un motor de base de datos orientado a objetos. La decisión se justificó con lo mencionado en el párrafo anterior y los siguientes puntos:

- la complejidad de las estructuras de datos (objetos) utilizados por la aplicación central para procesar la variedad de datos temporales (por ejemplo las DFTs).
- la necesidad de utilizar los métodos de estas estructuras (objetos) relativamente seguido.

Los sistemas de gestión de bases de datos orientadas a objetos (OODBMS) son el resultado de la combinación entre principios de programación orientada a objetos y de gestión de base de datos. Incluyen las técnicas de programación del paradigma orientado a objetos, así como también las propiedades de transacciones para bases de datos.

2.1.5. Software para Lecturas Manuales.- Para los casos en que el seguimiento de los motores no se haga de forma automatizada, se implementó un programa para procesar las muestras localmente y enviar los resultados a la base de datos a través de internet. Este software está compuesto por el módulo para el procesamiento de las lecturas de la aplicación central y una interfaz gráfica amigable para facilitar su uso.

Es importante tener acceso a internet para realizar el diagnóstico debido a que se requieren los valores de referencia del motor que están almacenados en la base de datos. Si no se cuenta con una conexión al momento de la recolección de datos, se debe almacenar las lecturas en el ordenador y solicitar el diagnóstico más tarde, cuando se disponga de la conexión. El programa no se diseñó con una base de datos local para obligar a los usuarios a concentrar todos los datos en la base de datos principal del sistema y así facilitar el seguimiento de la totalidad de los motores ingresados en el sistema.

3. Verificación.- En primer lugar, se verificó cada módulo de la arquitectura por separado y luego se realizaron ensayos con el sistema funcionando en su totalidad**.**

[6] Por defecto, una vez por minuto. No es una variable global del sistema, sino que se configura en cada dispositivo y puede depender del servicio contratado.

ISSN 1510-7450

3.1. Banco de Ensayos.- Se montó el siguiente banco de ensayos en el laboratorio de la Universidad de Montevideo.

Figura 5. Banco de Ensayos

3.2. Ensayos Aplicación Central.- En los ensayos documentados en 1, se muestra que de treinta lecturas realizadas, no fue posible el procesamiento automatizado en solamente un caso, verificando la eficacia del software. A su vez, en el mismo documento se indica que inspeccionando los valores temporales de la aplicación central en modo *debug*, se verificó que los valores de frecuencias y módulos calculados coincidieron con los calculados utilizando la herramienta MATLAB[7].

3.3. Ensayos Dispositivo de Muestreo.- Alcanzar la precisión de las lecturas realizadas por el dispositivo de muestreo, necesaria para la detección de defectos fue el último objetivo en ser cumplido.

La *Figura 6* verifica la precisión de la frecuencia de muestreo al coincidir la frecuencia del armónico fundamental con la de alimentación. La *Figura 7* por su parte, comprueba que la apreciación del conversor A/D es suficiente como para detectar las componentes de baja potencia, mostrando como ejemplo una de las componentes específicas del defecto de excentricidad. Si bien ambas figuras corresponden a ensayos realizados en el laboratorio, ya se han realizado lecturas a nivel industrial y el dispositivo funcionó correctamente a pesar de la presencia de interferencias electromagnéticas superiores a las que se pueden encontrar a nivel de laboratorio. La precisión es posible gracias al conversor A/D de 11 bits del módulo *RabbitCore 4000*, que en conjunto con las pinzas amperimétricas resulta en un error máximo de 0.220 A tomando cuando se trabaja con muestras de corriente de 1000 A (error relativo 0.022 %).

3.4. Servidor Web y Servidor de Base de Datos.- Se testeó el sistema utilizando la interfaz web y todas las funcionalidades respondieron exitosamente. Las mismas dependen de consultas, altas, bajas y modificaciones de la base de datos, por lo que verifican a su vez el funcionamiento del servidor de base de datos.

[7] MATLAB es la herramienta utilizada por la metodología de detección de fallas manual.

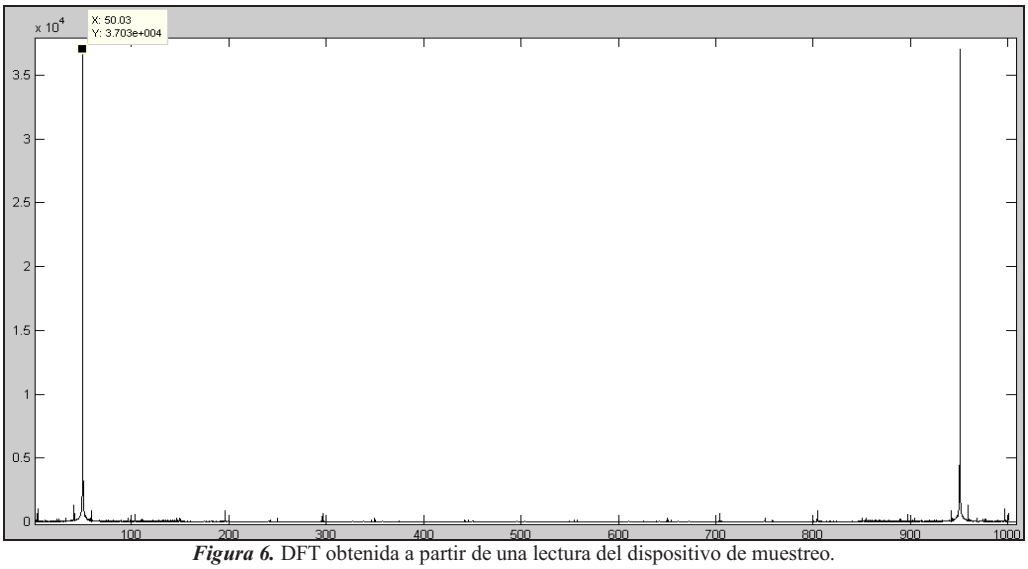

Figura 6. DFT obtenida a partir de una lectura del dispositivo de muestreo.

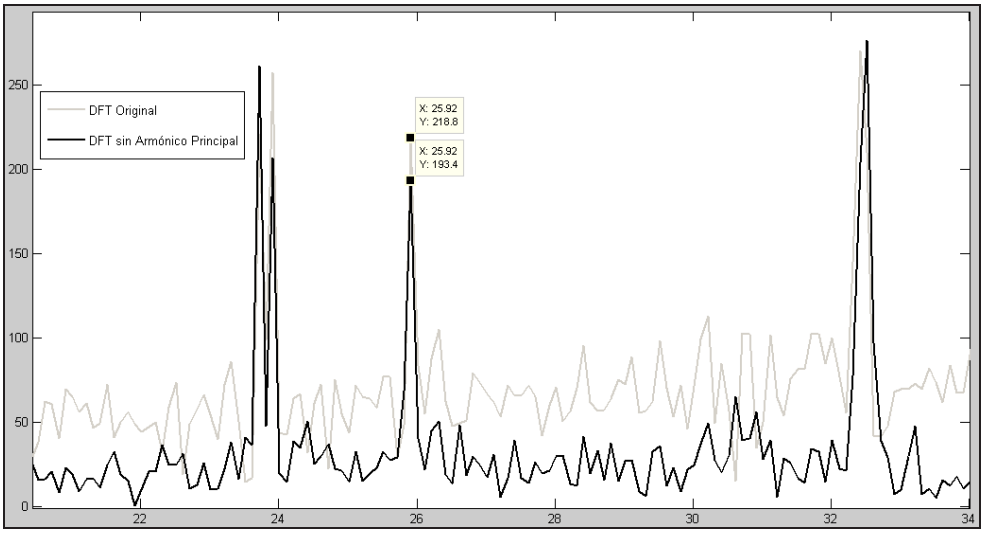

Figura 7. Componente específica de Excentricidad.

4. Conclusiones.- En este artículo se presentó el diseño e implementación de: un software que automatiza la metodología MCSA y gestiona los distintos tipo de alarmas; un servidor web para administrar el sistema y mostrar el estado de los motores a todos los usuarios interesados; una base de datos para almacenar todos los datos relevantes; y un dispositivo de muestreo económico y confiable que permite el monitoreo remoto y continuo de los motores ingresados en el sistema.

Si bien hay características del diseño que aún no han sido testeadas, por ejemplo el monitoreo de varios motores concurrentemente, los resultados obtenidos hasta el momento son muy alentadores y permiten responder positivamente a las preguntas sobre la viabilidad de un sistema de monitoreo continuo y remoto para el diagnóstico y gestión integral de defectos en motores de inducción.

ISSN 1510-7450

5. Referencias

1. Terra, José Ignacio. *Desarrollo de un Sistema de Monitoreo Continuo y Gestión Integral de Fallos en Motores de Inducción.* Tesis de Grado, Universidad de Montevideo, Abril 2010.
2. Castelli, Marcelo. *Desarrollo de una nueva metodología de monitoreo y diagnóstico de motores de inducción*. Tesis de Doctorado, Universidad de Navarra, Abril 2010.
3. Fossati, Juan Pablo. *Análisis de defectos en Motores de Inducción*. Tesis de Grado, Universidad de Montevideo, Marzo 2008.
4. Castelli, Marcelo; Fossati, Juan Pablo; Andrade, Marcos; *Metodología de monitoreo, detección y diagnóstico de fallos en motores asíncronos de inducción.* Memoria de Trabajos de Difusión Científica y Técnica. Número 6; Facultad de Ingeniería – Universidad de Montevideo, octubre de 2008.
5. Nejjari, Hamid; Benbouzid, Mohamed El Hachemi; *Monitoring and Diagnosis of Induction Motors Electrical Faults Using Current Park's Vector Pattern Learning Approach.* IEEE Transactions on Industry Applications. 2000, Vol. 36, No. 3.
6. Bangura, J.F; Demerdash, N.A; *Improvement of Monitoring and Diagnosis of Broken Bars/End-Ring Connectors and Airgap Eccentricities of Squirrel-Cage Induction Motors in ASDs Using a Time-Stepping Coupled Finite Element-State Space Technique*. IEEE International Conference IEMD, 1999.
7. Ammar, M.; Abdesselam, L.; Ahcen, B; *On-line monitoring and diagnosis of broken rotor bars in induaction motors;* IEEE, ELECO International Conference, Noviembre 2009.
8. Ordaz-Moreno, A.; de Jesus Romero-Troncoso, R.; Vite-Frias, J.A.; Rivera-Gillen, J.R.; Gracia-Perez, A.; *Automatic Online Diagnosis Algorithm for Broken-Bar Detection on Induction Motors Based on Discrete Wavelet Transform for FPGA Implementation.* IEEE Transactions on Industrial Electronics, 2008, Vol.55, No. 5.
9. Bellini, A.; Filippetti, F.; Franceschini, G.; Tassoni, C.; Passaglia, R.; Saottini M.; Tontini, G.; Giovannini, M.; Rossi, A.; *On-field experience with online diagnosis of large induction cage failures using MCSA.* IEEE Transactions on Industry Applications, 2002, Vol. 38, No 4.
10. Jung, Jee-Hoon; Lee, Jong-Jae; Kwon, Bong-Hwan; *Online Diagnosis of Induction Motors Using MCSA.* IEEE Transactions on Industrial Electronics, 2006, Vol. 53, No. 6.
11. RabbitCore Modules, disponible en http://www.rabbit.com, 23 de julio 2010.
12. Documentación RCM4000, disponible en http://www.rabbit.com/products/rcm4000/docs.shtml , 26 de julio 2010.