Tecnológico de Costa Rica
Escuela de Ingeniería Electrónica

Comunicación e interfaz gráfica para visualizar y almacenar mediciones de variables físicas de un sistema de paneles solares en el Centro de Producción ICE Barranca

Informe de Proyecto de Graduación para optar por el título de Ingeniero en Electrónica con el grado académico de Licenciatura

Pablo César Sulecio Varela

Cartago, Noviembre de 2011

INSTITUTO TECNOLOGICO DE COSTA RICA

ESCUELA DE INGENIERIA ELECTRONICA

PROYECTO DE GRADUACIÓN

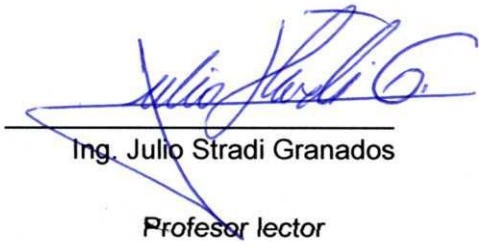
TRIBUNAL EVALUADOR

Proyecto de Graduación defendido ante el presente Tribunal Evaluador como requisito para optar por el título de Ingeniero en Electrónica con el grado académico de Licenciatura, del Instituto Tecnológico de Costa Rica.

Miembros del Tribunal

Ing. Néstor Hernández Hostaller

Profesor lector



Ing. Julio Stradi Granados

Profesor lector

Ing. Francisco Navarro Henríquez

Profesor asesor

Los miembros de este Tribunal dan fe de que el presente trabajo de graduación ha sido aprobado y cumple con las normas establecidas por la Escuela de Ingeniería Electrónica

Cartago, 21 noviembre 2011

Declaro que el presente Proyecto de Graduación ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado bibliografía, he procedido a indicar las fuentes mediante las respectivas citas bibliográficas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

Cartago, 21 de noviembre 2011

Firma del autor

Nombre completo del autor:
Pablo César Sulecio Varela

Céd: 1-1162-0352

Resumen

El Instituto Costarricense de Electricidad está adoptando nuevas políticas para la reducción de consumo de hidrocarburos para la producción de electricidad. Debido a esto se ha planteado la idea de generar electricidad por medio de energía solar. Un primer paso fue la implementación de un plan piloto el cual consiste en la instalación de un sistema fotovoltaico en el Centro de Producción ICE Barranca. El objetivo es estudiar el rendimiento de dicho sistema para analizar la factibilidad de su implementación a gran escala. En el presente informe se expone una deficiencia del sistema en la parte de monitoreo de las variables físicas involucradas en el sistema, tales como temperaturas, variables eléctricas, variables climatológicas y rendimientos. El problema es que el sistema de monitoreo no permite que los funcionarios del ICE puedan realizar análisis a partir de una base de datos de mediciones. Por esta razón fue necesario implementar un sistema que logre registrar las mediciones de las variables físicas del sistema fotovoltaico.

En el informe se presenta los recursos con que se cuenta y a partir de estos buscar posibles soluciones. Con el análisis de estos recursos se llegó al planteamiento de tres alternativas de las cuales se escogió la más económica y la más simple sin que esta deje de ser eficiente.

La solución consiste en la realización de un software que se comunica vía Ethernet con el dispositivo de monitoreo del sistema fotovoltaico. En dicho dispositivo se encuentran almacenas temporalmente las mediciones de las variables (estas se actualizan cada 3 segundo). El software adquiere las mediciones y las almacena en disco duro para que puedan ser usadas y analizadas por los funcionarios. En el informe se describe paso a paso como se desarrolló la aplicación, también se expone las limitaciones que se tuvieron y como se trataron.

Palabras clave: Dispositivo Sunny WebBox, .Net, C#. Clases, métodos.

Abstract

The new ICE policy of reduce oil consumption had led in the search of alternative power resources, such as solar energy. A first step was to implement a pilot project which involves the installation of a photovoltaic system in the Centro de Producción ICE Barranca. The main purpose of the pilot project was to analyze the solar energy source performance. The photovoltaic system monitors physic variables such as temperature, electric and weather variables and yields. The problem was that the monitoring system did not pursuit a measurement database, hence ICE technician could not analyze performances. For this reason, it was necessary to implement a system that recorded electric and weather measurements.

This report outlines the available resources for finding a solution. It was found three solutions; from those it was chosen the cheapest and the simplest solution still being efficient.

The solution consisted in an implementation of software that establishes communication with a device via Ethernet. The device temporarily stores measurements and these are updated every 3 seconds. The software acquires and stores the measurements so they can be analyzed by the technicians. The report describes step by step how the application was developed, it also exposes the limitations and how they were handled.

Keywords: Sunny WebBox device, .NET, C#, class, method.

A mi querida difunta abuela Albertina Peraza Mata,
cuya pasión por los animales, compasión por los enfermos y el desapego a los cosas materiales; me motivan a contribuir a la sociedad de forma positiva

Agradecimiento

Agradezco primeramente a Dios, que por él todas las cosas son posibles, a mi querida madre Vera Varela por su gran sacrificio y cuyo apoyo incondicional me ayudó a sobrepasar muchas barreras, a mi padre Francisco Sulecio por estar pendiente en el desarrollo del proyecto, a mi abuelo, Wenceslao Varela por sus consejos sabios, a mi Padrastro Oscar Rodríguez, a mis hermanos José Sulecio, Andrés Rodríguez, Laura Rodríguez, Luis Sulecio, los cuales han hecho de mi una mejor persona. y en alguna manera eso ayudó en la culminación de mi carrera. A todos los familiares cercanos un agradecimiento especial por ayuda y apoyo durante toda mi vida. A mis compañeros y amigos de carrera por estar ahí en los malos y buenos momentos. A todos mis profesores por compartir su conocimiento y por su pasión por formar buenos profesionales. A mi profesor asesor Francisco Navarro y profesores lectores, Néstor Hernandez y Julio Stradi por su esfuerzo y dedicación en esta última etapa de mi carrera.

Agradezco a Nolan Rojas y a todos los funcionarios del Centro de Producción ICE Barranca por hacerme sentir como en familia y brindarme su apoyo durante la realización del proyecto.

Agradecimiento especial a la familia Brais Espinoza; Abdalab, Mauren, Abigail y Brenda, por acogerme en su hogar y hacerme sentir como en casa.

INDICE GENERAL

Capítulo 1: Introducción 1

1.1 Problema existente e importancia de su solución 3

1.2 Solución seleccionada 5

Capítulo 2: Meta y Objetivos 6

2.1 Meta 6

2.2 Objetivo general 6

2.3 Objetivos específicos 6

Capítulo 3: Marco teórico 8

3.1 Descripción del sistema fotovoltaico 8

3.2 Antecedentes Bibliográficos 12

3.3 Descripción de los principales principios computacionales relacionados con la solución del problema 12

3.3.1 Llamada a procedimiento remoto (RPC): 12

3.3.2 Lenguaje de programación C#: 14

3.3.3 ¿Por qué programar en C#? 17

3.3.4 La biblioteca webBox.RPC.dll 18

3.3.5 Ambiente de desarrollo integrado SharpDevelop: 19

Capítulo 4: Procedimiento metodológico 20

4.1 Reconocimiento y definición del problema 20

4.2 Obtención y análisis de información 21

4.3 Evaluación de las alternativas y síntesis de una solución 22

4.3.1 Alternativa 1 22

4.3.2 Alternativas 2 26

4.3.3 Alternativa 3 (solución escogida) 28

4.4 Implementación de la solución 31

4.5 Reevaluación y rediseño 37

Capítulo 5: Descripción detallada de la solución (Explicación del diseño) 39

5.1 Análisis de soluciones y selección final 39

5.2 Descripción del software (por módulos) 41

5.2.1 Declarar proceso en paralelo 42

5.2.2 Declarar clases y variables 43

5.2.3 Enviar petición de envío de mediciones 45

5.2.4 Traducir y extraer mediciones del paquete recibido 48

5.2.5 Actualizar el valor de la medición en la interfaz gráfica 48

5.2.6 Guardar datos y tiempos en memoria RAM 49

5.2.7 Guardar datos de la memoria RAM a disco duro 50

5.2.8 Mostrar mensaje de que se cerró la conexión 52

5.2.9 Realizar y mostrar Interfaz gráfica" 52

5.2.10 Graficar datos de la memoria RAM 54

5.2.11 Graficar datos guardados en disco duro 55

5.2.12 Abrir y cerrar conexión 55

Capítulo 6: Análisis de Resultados 56

6.1 Resultados 56

6.2 Análisis 77

Capítulo 7: Conclusiones y recomendaciones 80

7.1 Conclusiones 80

7.2 Recomendaciones 81

8. Bibliografía 82

ANEXO 1 83

ANEXOS 2 84

INDICE DE FIGURAS

Figura 3.1 Paneles solares sobre el techo de una estructura dentro del ICE barranca 8
Figura 3.2 Conexión física de los paneles solares 9
Figura 3.3 Vista de los 3 inversores Sunny Boy A, B y C 9
Figura 3.4 Dispositivo Sunny WebBox [5]..... 10
Figura 3.5 Comunicación del sistema de monitoreo..... 11
Figura 3.6 Pasos para realizar una llamada a procedimiento remoto. Los stubs están sombreados. [1] 13
Figura 4.1 Comunicación del SMA OPC Server con un sistema fotovoltaico de forma remota [4]..... 23
Figura 4.2 Comunicación del SMA OPC Server con varios sistemas fotovoltaico en diferentes ubicaciones..... 23
Figura 4.3 Diagrama de bloques general de la solución 1..... 24
Figura 4.4 Diagrama de bloques de la solución 2..... 27
Figura 4.5 Diagrama de bloques general de la solución 3..... 29
Figura 4.6 de una cadena de caracteres en formato JSON 32
Figura 4.7 Ventana principal de la aplicación “Real Time WebBox Values . 34
Figura 4.8 Variables que se miden al inversor SunnyBoy A..... 34
Figura 4.9 Botón para visualizar la gráfica de valores de velocidad de viento en el tiempo..... 35
Figura 4.10 Gráfica generada por la aplicación correspondiente a mediciones de potencia total generado por el sistema fotovoltaico desde 10:13 am hasta 11:13 am..... 35
Figura 4.11 Vista de acercamiento de la figura 4.7 de tres botones..... 36
Figura 4.12 Ventana para graficar mediciones de días pasados..... 37
Figura 5.1 Diagrama de flujo de la aplicación Real Time WebBox Value..... 40
Figura 5.2 Ventana que avisa que la configuración regional y de idioma no es la correcta..... 41
Figura 5.3 Diagrama de flujo ineficiente de dos procesos. 42

Figura 5.4 Contenido del archivo txt que se genera 51
Figura 6.1 Temperatura ambiente en los paneles en función del tiempo 63
Figura 6.2 Irradiación solar en función del tiempo 63
Figura 6.3 Temperatura del módulo Sunny SensorBox en función del tiempo 64
Figura 6.4 Velocidad del viento en función del tiempo 64
Figura 6.5 Potencia total producida por el sistema fotovoltaico en función del tiempo 64
Figura 6.6 Rendimiento del sistema solar en función del tiempo..... 65
Figura 6.7 Rendimiento del día actual en función del tiempo 65
Figura 6.8 Temperatura en el inversor A en función del tiempo 65
Figura 6.9 CO2 ahorrado en función del tiempo..... 66
Figura 6.10 Energía producida por el inversor A en función del tiempo 66
Figura 6.11 Frecuencia de la señal de tensión en la salida del inversor A en función del tiempo 66
Figura 6.12 Corriente efectiva suplida por el inversor A en función del tiempo 67
Figura 6.13 Corriente directa generada por un arreglo de paneles solares medida en la entrada del inversor A, en función del tiempo. 67
Figura 6.14 Potencia en el inversor A entregada a la red en función del tiempo 67
Figura 6.15 Tensión directa generada por un arreglo de paneles, entregada al inversor A, en función del tiempo 68

INDICE DE TABLAS

Tabla 4.1 Precios de software..... 25
Tabla 5.1 Lista de clases implementadas en la aplicación Real Time WebBox Value 44
Tabla 5.2 Clasificación por grupos y definición de las variables que se miden 47
Tabla 5.3 Requerimientos de la aplicación 53
Tabla 6.1 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador; medición 1 a la medición 20 57
Tabla 6.2 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador medición 21 a la medición 40 58
Tabla 6.3 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador medición 41 a la medición 60 59
Tabla 6.4 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador medición 61 a la medición 80 60
Tabla 6.5 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador medición 81 a la medición 100 61
Tabla 6.6 Datos a partir de las muestras tomadas de las tablas 6.1 a la 6.5. 62
Tabla 6.7 Valores de mediciones y tiempos de la variable Potencia Total. .. 69
Tabla 6.8 Mediciones y tiempos de la variable Rendimiento Diario. 69
Tabla 6.9 Mediciones y tiempos de la variable Rendimiento 70
Tabla 6.10 Mediciones y tiempos de la variable Irradiación. 70
Tabla 6.11 Mediciones y tiempos de la variable Temperatura Ambiente 71
Tabla 6.12 Mediciones y tiempos de la variable Temperatura del Módulo ... 71
Tabla 6.13 Mediciones y tiempos de la variable Velocidad de Viento..... 72
Tabla 6.14 Mediciones y tiempos de la variable potencia generada por el inversor A. 72
Tabla 6.15 Mediciones y tiempos de la variable energía diaria del inversor A 73

Tabla 6.16 Mediciones y tiempos de la variable CO2 ahorrado por el inversor A. 73
Tabla 6.17 Mediciones y tiempos de la variable frecuencia eléctrica del inversor A. 74
Tabla 6.18 Mediciones y tiempos de la variable corriente alterna en la salida del inversor A. 74
Tabla 6.19 Mediciones y tiempos de la variable corriente directa en la entrada del inversor A 75
Tabla 6.20 Mediciones y tiempos de la variable tensión alterna en la salida del inversor A 75
Tabla 6.21 Mediciones y tiempos de la variable tensión continua en la entrada del inversor A. 76
Tabla 6.22 Mediciones y tiempos de la temperatura del inversor A 76

Capítulo 1: Introducción

El Instituto Costarricense de Electricidad (ICE) es una empresa gubernamental encarga de brindar servicios de electricidad y telecomunicaciones en el territorio costarricense. El ICE cuenta con diversos tipos de plantas de generación eléctrica, dentro de las cuales se puede nombrar: plantas eólicas, geotérmicas, termoeléctricas, hidroeléctricas y solares. Las plantas hidroeléctricas son las mayores suplidoras de electricidad en el país, sin embargo, en verano, los causes de los ríos disminuyen, imposibilitando la generación suficiente de energía, es por esta razón que el ICE ha incorporado plantas termoeléctricas que generan energía para amortiguar el déficit de potencia de las plantas hidroeléctricas.

Las plantas termoeléctricas funcionan con combustible y como se mencionó, solo funcionan en casos de emergencia; lamentablemente el gasto de combustible de estas plantas es muy alto y además la emisión de CO_2 es altamente contaminante.

Dentro de las políticas del ICE de abaratar costos y de tener responsabilidad socio ambiental, se ha planteado la idea de producir y vender electricidad producida por sistemas fotovoltaicos. Esta solución es la más viable, desde la perspectiva económica, ya que el costo de producción disminuiría considerablemente y además bajaría la contaminación por emanación de CO_2.

Debido a que el ICE quiere instalar sistemas fotovoltaicos, se realizó un plan piloto. Este consiste en un sistema de paneles solares el cual abastece de electricidad únicamente a la infraestructura del Centro de Producción ICE Barranca. Esto con el propósito de estudiar la eficiencia de los paneles solares, para evaluar que tan factible es utilizar esta tecnología.

El sistema de paneles solares instalado en el Centro de Producción Barranca consiste en varios paneles, que en combinación producen 10 kW. El sistema posee varios módulos o etapas las cuales se en listan y explican a continuación:

- Inversores: este módulo convierte la corriente directa, generada por los paneles, a corriente alterna. Esta señal es acondicionada para que tenga una tensión de línea de 208 V a 60 Hz y debe estar en fase con la señal eléctrica de la red proveniente de las subestaciones. Esto con el propósito de que la señal eléctrica abastezca de electricidad toda la infraestructura del ICE de Barranca.

- Banco de baterías: En la noche los paneles solares no producen electricidad (debido a la ausencia de luz), es por ello que hay un banco de baterías que produce electricidad. Se dice entonces que este sistema es el encargado de proporcionar energía eléctrica permanentemente. Este módulo también se encarga de cargar el banco de baterías cuando se es de día.

- Sistema de monitoreo: este sistema consiste en una estación meteorológica llamada Sunny SensorBox esta mide temperatura, radiación solar y velocidad del viento. Estas mediciones se realizan en el lugar donde se está ubicado los paneles solares. También se cuenta con un módulo que mide todas las variables eléctricas de todo el sistema (tensiones, corriente, potencia eficiencia etc.). Todos estos datos son enviados de forma serial a un módulo, éste consiste en una estación web la cual presenta todas las mediciones en una página web. Dicho módulo se llama WebBox

- Estación web (Sunny WebBox): Este módulo es el encargado de presentar todas las mediciones realizadas en una página web. Con este sistema un usuario puede acceder a las mediciones desde una computadora con conexión a internet.

Los empleados del ICE encargados de este plan piloto, tenían inconvenientes a la hora de tomar lecturas de las mediciones; estas se pueden acceder por medio de

una página web, sin embargo la página no cuenta con una interface gráfica la cual permita una visualización eficiente de las mediciones. Esta página web presenta las siguientes desventajas:

a) Solo se puede visualizar un grupo de variables a la vez.

b) La estación web no cuenta con un sistema de almacenamiento por lo cual no permite llevar un registro de las mediciones realizas.

c) Este sistema es dependiente del servicio de internet, quiere decir, que si el servicio falla no hay manera de realizar las lecturas.

Llevar un registro de las mediciones y tener una visualización adecuada de ellas es de suma importancia a la hora de analizar la eficiencia de los paneles solares. Esta dificultad impedía el estudio de la factibilidad de esta nueva alternativa.

1.1 Problema existente e importancia de su solución

El Centro de Producción Barranca del ICE ha instalado un sistema de generación de energía solar, como plan piloto para estudiar la factibilidad de este recurso a la hora de generar electricidad para que se disminuya la producción por medio de la combustión.

Para estudiar la factibilidad de este recurso es de suma importancia poder estudiar y analizar las variables físicas involucradas en todo el sistema, debido a que, por ejemplo, el ángulo de incidencia de radiación con el panel y la temperatura son variables físicas que modifican la eficiencia de los paneles.

El sistema de paneles solares instalados en el Centro de Producción ICE Barranca funciona correctamente y cuenta con un sistema de monitoreo el cual permite visualizar todas las medidas de las variables físicas climatológicas y eléctricas del sistema, estas variables son enviadas a una estación web, la cual permite que un usuario pueda visualizar las mediciones desde un computador con conexión a internet.

A pesar de contar con este recurso, los operarios de la subestación se quejaban de que la página web no contaba con una visualización más eficiente y amigable de las mediciones realizadas por el sistema.

A continuación se en listan las dificultades que los funcionarios tenían con el sistema antes del nuevo sistema implementado:

d) La estación web no contaba con un sistema de almacenamiento, por lo cual no permitía llevar un registro de las mediciones realizadas.

e) Este sistema es dependiente del servicio de internet, quiere decir, que si el servicio falla no había manera de realizar las lecturas.

f) Se tenía dificultades a la hora de querer ubicar los paneles en otro lugar debido a que el lugar debe contar con conexión a internet

1.2 Solución seleccionada

Debido a que la página web donde se acceden los datos de las mediciones no es eficiente, se diseñó un sistema con el que se logró visualizar y guardar, en una base de datos, todas las mediciones del sistema. En la figura 4.5 se muestra un esquema de la solución que se plantea.

Se realizó un enlace por medio de cable Ethernet para comunicar la estación Web (WebBox) con un computador.

Al computador se le instaló un software que cuenta con los siguientes módulos:

a) Manejador de Protocolo y tráfico de datos: Este módulo opera el protocolo de comunicación entre la computadora y la estación Web para el envío y recepción de datos.

   Debido a que se cuenta con diversas variables física, este módulo se encarga de clasificar un dato de medición. Se debe tener una función que mida el tiempo que transcurre desde que se envía una petición de información hasta la llegada de información requerida. Esto con el propósito de detectar fallas.

b) Administrador de la base de datos: Se debe contar con una base de datos en la cual se lleve un registro de las mediciones y la hora en la que se realizó dicha medición. Este módulo genera archivos donde se muestren las mediciones realizadas en un periodo específico.

c) Módulo HMI (Human Machine Interface): Se diseñará una interfaz gráfica para que un usuario pueda visualizar de forma efectiva todas las variables físicas medidas. Las variables también se podrán observar en gráficas, donde se visualice su comportamiento con el transcurrir del tiempo.

   La interface gráfica debe ser entendida fácilmente a primera vista por los usuarios.

Capítulo 2: Meta y Objetivos

2.1 Meta

Brindar a los funcionarios del ICE Barranca las herramientas necesarias para que puedan monitorear y estudiar el rendimiento de los paneles solares con el objetivo de evaluar la factibilidad.

Indicador: Los funcionarios pueden ver todas las mediciones presentes y pasadas de forma clara, sin ningún tipo de retraso.

2.2 Objetivo general

Elaborar un sistema que muestre en una interface gráfica las mediciones climatológicas y eléctricas de un sistema de paneles solares instalados en la subestación del ICE en Barranca.

Indicador: Las mediciones se tienen que actualizar en un tiempo máximo de 7 s.

2.3 Objetivos específicos

1. Diseñar el módulo Manejador de Protocolo y Tráfico de Datos.

   Indicador: Se enviará a la Estación Web una petición de envío de un grupo de variables físicas 100 veces, cada variable debe de tener un acierto mayor o igual al 90% y una latencia máxima de 1s para la recepción de un dato.

2. Realizar el módulo Administrador de la base de datos.

   Indicador: Se leerán 1000 mediciones de cada variable. Estas se guardarán en disco duro a partir de ellas se generará una gráfica y los resultados deben ser coherentes.

3. Diseñar el módulo de software HMI.

   Indicador: Se tiene que visualizar correctamente todas las mediciones realizadas durante el día, y también mediciones efectuadas en algún periodo en el pasado; el periodo debe de estar después de la fecha de la primera medición. Se debe visualizar la medición actual de cada variable; y la actualización de cada variable se debe dar en un tiempo menor a 7s, esto dentro de un periodo de por lo menos 20s.

Capítulo 3: Marco teórico

3.1 Descripción del sistema fotovoltaico

El sistema consta de un conjunto de paneles solares tal y como se aprecia en la figura 3.1 estos paneles están ubicados sobre un techo en el Centro de Producción de Barranca del ICE. El conjunto consiste en 6 filas de 10 paneles

Figura 3.1 Paneles solares sobre el techo de una estructura dentro del ICE barranca

En medio de los paneles se encuentra instalado un módulo llamado “Sunny SensorBox”, este mide variables climatológicas (radiación solar, velocidad del viento, temperatura del ambiente y temperatura del módulo).

Cada dos filas de paneles se conectan en paralelo, para conformar 3 conjuntos de dos filas. De cada conjunto salen dos cables con una tensión directa; tal como se ve en la figura 3.2. Estos cables se conectan a los inversores. Los inversor llamados (“Sunny Boy”) son los encargados de tomar la tensión directa y transformarla a tensión alterna. Cada inversor, aparte de transformar tensión, también se encarga de medir diferentes variables físicas, tales como: tensión

directa, tensión alterna, corrientes, temperatura etc.… Los inversores se muestran en la figura 3.3

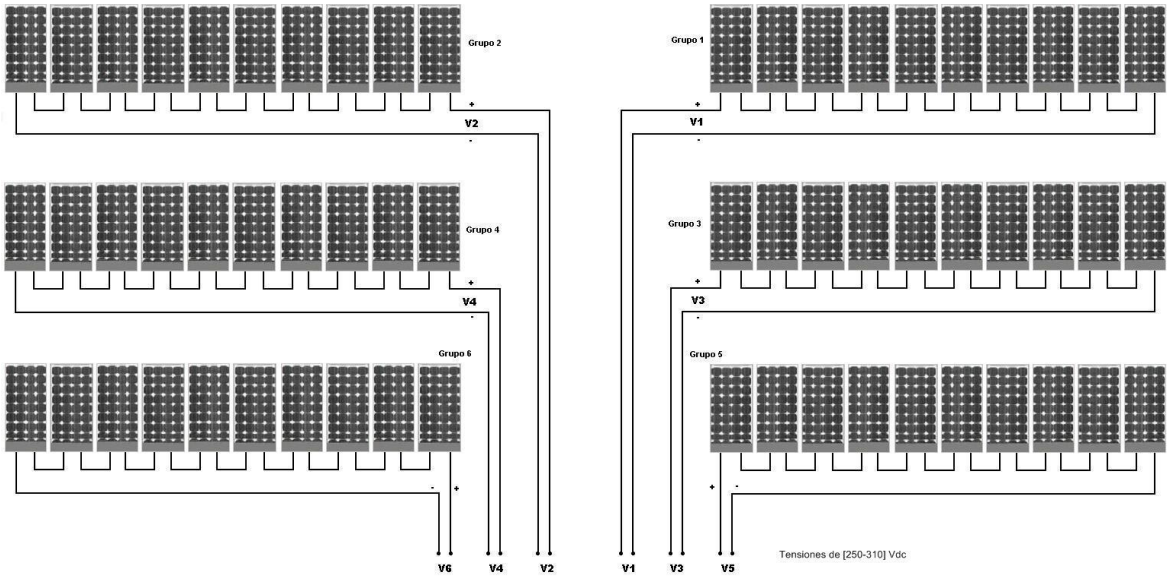
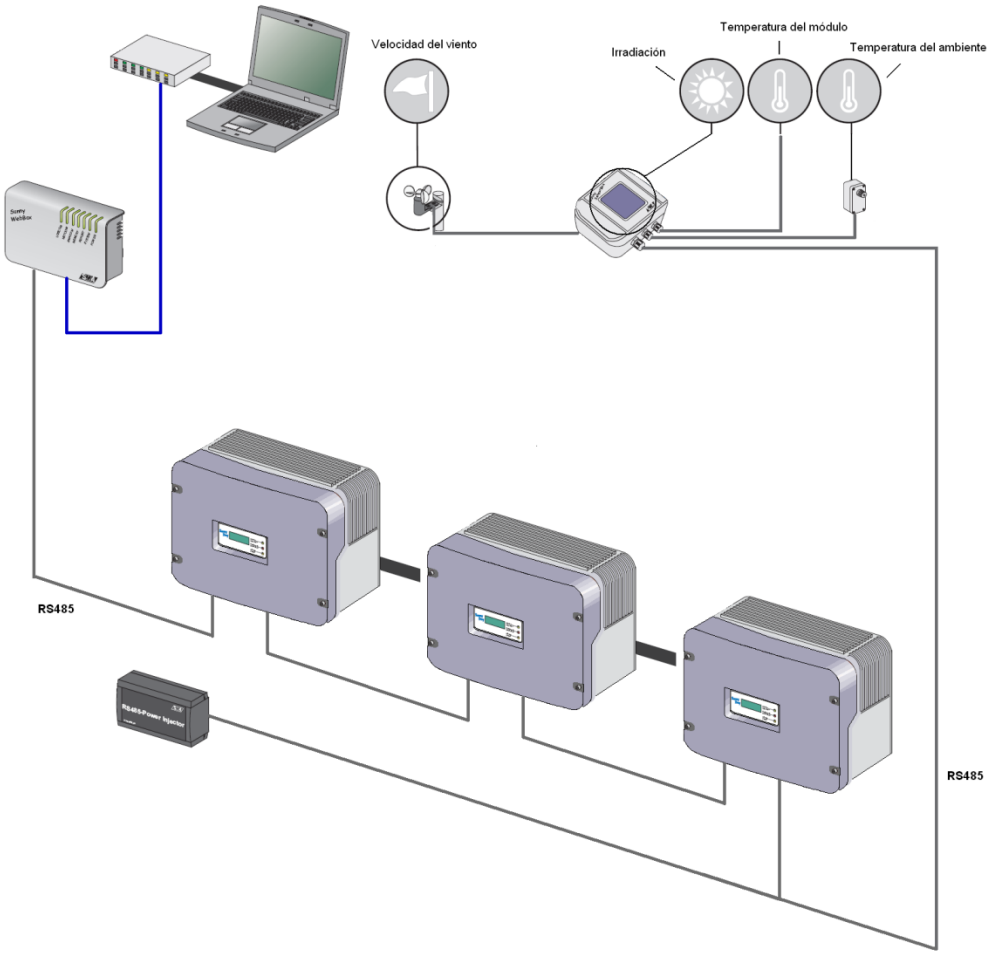


Figura 3.2 Conexión física de los paneles solares

Figura 3.3 Vista de los 3 inversores Sunny Boy A, B y C

El sistema cuenta con un dispositivo llamado “Sunny WebBox” este se conecta con el módulo que mide las variables climatológicas (el Sunny SensorBox) y se conecta también con los inversores (Sunny Boy A, B y C) a través de una comunicación RS-485 ver figuras 3.4 y 3.5. Se conecta a estos dispositivos con la intención de extraer y guardar todas las mediciones realizadas, en su memoria. Por medio de su puerto Ethernet las mediciones puedan ser accedidas por medio de una conexión a internet.

Figura 3.4 Dispositivo Sunny WebBox [5]



Paint

Figura 3.5 Comunicación del sistema de monitoreo

La Web Box cuenta con la facilidad de poder comunicarse vía HTTP o vía UDP por medio del protocolo RPC, esta opción permite a un usuario realizar un programa para obtener datos guardados en la memoria de la Web Box y así poder manipular estos datos a conveniencia.

3.2 Antecedentes Bibliográficos

Investigando sobre la funcionalidad del Sunny WebBox se encontró un manual para programadores el cual brinda información pertinente para realizar transferencia de datos desde el dispositivo WebBox a una computadora, dicho material se presenta en el informe en el Anexo 2.
Toda la información y documentación sobre la WebBox se puede bajar del sitio Web de “ASM América” cuya dirección se puede ver en [5]. En la parte download se puede bajar documentos y manuales del dispositivo.

La empresa SMA brinda una herramienta para establecer comunicación entre un computador y el WebBox. La información Se puede encontrar referencia en el sitio Web “ASM América” en la dirección de la referencia [4]

3.3 Descripción de los principales principios computacionales relacionados con la solución del problema

3.3.1 Llamada a procedimiento remoto (RPC):

La llamada a procedimiento remoto RPC por sus siglas en ingles Request Procedure Call, es una técnica utilizada en redes computacionales para realizar aplicaciones donde una computadora ocupa recursos de otra computadora.

Esta técnica consiste en que un programa hace llamadas a funciones no localmente, sino que hace llamadas a funciones ubicadas en un host remoto. El host que realiza la llamada se le denota como “cliente” y el host que recibe la llamada se conoce como servidor.
Una vez que el servidor recibe una llamada, este debe de correr la función dentro de su sistema operativo, con lo que se obtiene un valor de retorno, el cual será enviado al dispositivo que realizó la llamada.

A continuación se citan los pasos que propone TANENBAUM, Andrew S en su libro "*Redes de Computadoras"* [1] que se dan en una llamada a procedimiento.

"El paso 1 consiste en que el cliente llame al stub del cliente. Ésta es una llamada a procedimiento local, y los parámetros se colocan en la pila de la forma tradicional. El paso 2 consiste en que el stub del cliente empaca los parámetros en un mensaje y realiza una llamada de sistema para enviar dicho mensaje. El empaquetamiento de los parámetros se conoce como marshaling. El paso 3 consiste en que el kernel envía el mensaje desde la máquina cliente a la máquina servidor. El paso 4 consiste en que el kernel pasa el paquete entrante al stub del servidor. Por último, el paso 5 consiste en que el stub del servidor llame al procedimiento servidor con parámetros sin marshaling. La respuesta sigue la misma ruta en la dirección opuesta."[1]

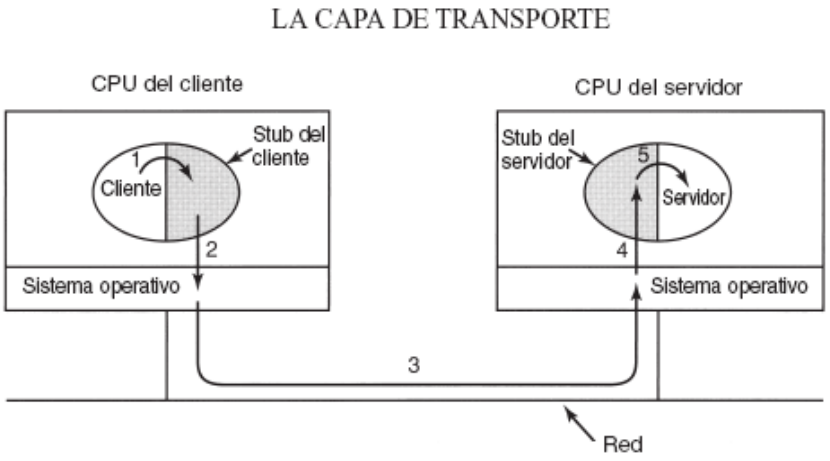


Figura 3.6 Pasos para realizar una llamada a procedimiento remoto. Los stubs están sombreados. [1]

3.3.2 Lenguaje de programación C#:

Desde los años 80´s el leguaje C y C++ han sido lenguajes predilectos a la hora de desarrollar aplicaciones comerciales; este lenguaje tiene la gran ventaja de proporcionar control al programador permitiendo el uso de punteros y muchas funciones de bajo nivel. Por otro lado se cuentan con programas que no proporcionan tanto control pero que permiten gran facilidad y rapidez a la hora de realizar una aplicación tal como Microsoft Visual Basic. Esta característica no se logra con lenguajes como C/C++ que obliga al programador a gastar mucho tiempo en una aplicación.
Programadores en C/C++ no pasarían a una programación fácil como Visual Basic, por el miedo a perder control del bajo nivel.

La comunidad de programadores ocuparía un lenguaje que brindara características de control de bajo nivel y rápida programación, además un leguaje con el cual se pueda realizar aplicaciones Web, XML y tecnologías emergentes. Para facilitar una programación fácil a los usuarios de C/C++ sin problemas de transición y además bridar a los pocos expertos un aprendizaje fácil emergió el lenguaje C#.

Este leguaje fue presentado por Microsoft en la Professional Developer´s Conference en el verano del 2000 en Orlando Florida. C# extrae las mejores ideas de lenguajes como C, C++ y Java con las mejoras de productividad de .NET Framework de Microsoft.
[2]

A continuación se enlista las diferencias existentes entre los lenguajes C++ y C#.

Herencia: una clase puede heredar una implementación de una única clase base. También una clase o una interfaz pueden implementar múltiples interfaces.

Los arreglos: La sintaxis para declarar arreglos en C# es diferente a la de C++. Los símbolos “[]” aparecen inmediatamente después del tipo de variable en el caso de C#.

El tipo bool: En C# no se pueden convertir variables bool a otro tipo de variables, específicamente, un bool no se puede convertir en un int.

La variable long: En C# el tipo long es de 64 bits mientras que en C++ es de 32 bits.

El tipo struct: En C# las clases y estructuras son semánticamente diferentes. Una estructura es in tipo de valor, mientras que una clase un tipo de referencia.

La instrucción switch: A diferencia de la instrucción switch en C++, C# no permite hacer saltos desde una etiqueta case hacia otra.

El tipo delegate: Delegados son aproximadamente iguales a los punteros de funciones en C++.

Las directivas del preprocesador son usadas para compilación condicional. Es por eso que en C# no son usados los archivos de cabecera.

C# contiene un manejador de excepciones, utilizando las instrucciones finally.

Operadores C#: C# maneja operadores adicionales tales como is y typeof. También ofrece diferentes funcionalidades en los operadores lógicos.

La manera general de estructurar un programa en C#: namespaces, clases, strucs, delegates y enumerations.

Parámetros de métodos: En C# se maneja los parámetros ref y out los cuales son usados en vez de punteros, para pasar parámetros por referencia.

Los punteros son permitidos en C# pero solamente cuando se trabaja en modo unsafe.

Los operadores sobrecargados son manejados diferente en C#.

Las cadenas en C# son diferentes a las de C++.

En C# existe la palabra clave foreach, que permite hacer iteraciones a través de arreglos y colecciones.

No existen variables ni métodos globales en C#: Métodos y variables deben ser declarados dentro de una estructura o una clase.

Los archivos de cabecera o la directiva #include no se usan en C#: La directiva using es usada para hacer referencias a tipos.

Las variables locales en C# no pueden ser usadas antes de ser inicializadas.

Destructores: En C# no se tiene control sobre cuando un destructor es llamado por que los destructores son llamados automáticamente por el colector de basura.

Constructores: Similar a C++, si no se provee un constructor a una clase en C#, por defecto se crea automáticamente un constructor. El constructor que se crea por defecto inicializa todos los campos de sus variables.

C# no maneja campos de bit.

El servicio de entrada/salida y el formato se basa en la biblioteca de tiempo de ejecución del .NETFramework.

En C# los parámetros de los métodos no pueden tener valores por defecto. Se debe de tener métodos sobre cargados si quiere esa característica.
[3]

3.3.3 ¿Por qué programar en C#?

Una de las ventajas de este lenguaje es que se puede usar más de 40 leguajes de programación diferentes, es la llamada integración de lenguajes. Además si se es programador de C++ será muy fácil aprender a programar en C#.

Toda la plataforma .NET está hecha en C#, así que es de gran utilidad trabajar con este lenguaje si se desarrolla una aplicación web. Además .NET brinda las herramientas necesarias para trabajar con una interfaz gráfica basada en Windows.

La plataforma .NET es eficiente y versátil es por ello que muchos fabricantes de sistemas embebidos han proporcionado a sus clientes bibliotecas basadas en .NET las cuales brindan herramientas necesarias para desarrollar aplicaciones que complementan alguna función del sistema embebido; por ejemplo una aplicación en una PC que se comunique con el sistema embebido para hacer transferencia de datos importantes; utilizando la web.

En el caso de la implementación de este proyecto se cuenta con la biblioteca webBox.RPC.dll basada en .NET la cual brinda funciones para comunicarse con el dispositivo Sunny WebBox.

3.3.4 La biblioteca webBox.RPC.dll

El ensamblador webBox.RPC.dll ayuda a realizar servicios de procedimientos RPC en la SMA Sunny WebBox usando C# o cualquier otro lenguaje .NET

El ensamblado consiste en varias clases y una de las principales y más usadas clases son:

a) SMA.WebBox.RPC.Procedures: se utiliza para crear cadenas de comandos JSON para un procedimiento WebBox disponible.

b) SMA.WebBox.RPC.Connection: este clase contiene dos clases las cuales son útiles para establecer comunicación con la computadora y el dispositivo WebBox, para que se lleve a cabo los procedimientos.

c) SMA.WebBox.RPC.GUIAdapter: después de recibir una respuesta el dato devuelto debe convertirse en un tipo donde se pueda manipular, por ejemplo convertirlo en un string o a un int para ser manipulado por el programador

También se cuenta con la clase SMA.WebBox.RPC.Form.SelectChannelsForm la cual brinda al programador la facilidad de crear formas en una interfaz gráfica. La forma brinda la opción de elegir el canal o el dispositivo del cual se obtendrán mediciones.

3.3.5 Ambiente de desarrollo integrado SharpDevelop:

El lenguaje de programación C# como todo lenguaje ocupa de una herramienta en la cual se pueda escribir líneas de código (editor) y así contar con diversas herramientas que nos ayudaran a realizar nuestro proyecto. Para el presente proyecto se utiliza el SharpDevelop. Este es un entorno de desarrollo integrado (IDE) que brinda herramientas muy útiles en el desarrollo de alguna aplicación; como por ejemplo el gestor de interfaz gráfica o también llamado diseñador de Windows Form, y herramientas para depurar etc. El SharpDevelop es un software libre por lo que se puede descargar en internet sin ningún problema.

Capítulo 4: Procedimiento metodológico

4.1 Reconocimiento y definición del problema

El sistema de paneles solares cuanta con un sistema de monitoreo de diferentes variables físicas; los datos de estas mediciones se guardan en un dispositivo llamado Sunny WebBox, este se encarga de guardar los datos y permite visualizar la medición actual de las variables, por medio de una conexión a internet desde una computadora. La desventaja de esto es que los datos no podían ser manipulados para el análisis de estos.

Antes de implementar el sistema si se quería hacer un análisis de los datos, un usuario tendría que observar los datos de la página web para anotarlos manualmente. Evidentemente este es un trabajo tedioso y casi imposible ya que se cuentan con alrededor de 40 variables, las cuales se actualizan cada 4 segundos. El usuario tendría que estar todo un día realizando mediciones. El hacer esta actividad es casi inconcebible tomando en cuenta el desarrollo tecnológico que hay actualmente.

Definición del problema: No existía un método adecuado el cual permitiera el análisis de datos de las mediciones realizadas en el sistema fotovoltaico.

4.2 Obtención y análisis de información

El coordinador de mantenimiento del Centro de Producción ICE de Barranca comentó lo importante que era tener un sistema de monitoreo del sistema fotovoltaico el cual brindara varias opciones para el manejo y manipulación de las variables que se estaban midiendo. Por ejemplo se quiere poder calcular, a partir de las mediciones, la eficiencia de la generación de energía por parte de los sistemas solares, cosa que en el pasado no se realizaba debido a que las mediciones no se podían guardar en la memoria del computador.

En el centro de producción se le asignó a uno de los técnicos tomar lecturas de las mediciones, por lo menos dos veces al día, con el fin de verificar el funcionamiento correcto del sistema fotovoltaico. El sistema fotovoltaico a medio día con el cielo despejado logra producir una potencia entre los 8000 y 9000 W, aproximadamente, si el técnico toma la lectura de potencia a medio día con el cielo despejado y esta mide menos de 8000W, se puede sospechar que el sistema tiene una avería. Nótese que si existiera una avería y el técnico no pudo realizar una lectura de potencia al medio día con clima despejado; no pudo entonces percatarse de un mal funcionamiento del sistema. Este era otro inconveniente al no tener un registro de mediciones realizadas.

Los administrativos de la planta plantearon como solución el pedirle a la empresa (la que instaló el sistema fotovoltaico) que añadiera un sistema adicional que permitiera llevar un registro de las mediciones y que estas se pudieran manipular. Se le solicitó a la empresa que instalara ese sistema pero por motivos adversos la empresa no puedo proporcionar lo que se le pidió.

En una reunión con el coordinador de mantenimiento y encargado de los paneles comentó que la compañía SMA, la cual es la fabricante de los módulos del sistema fotovoltaico, ofrece un producto el cual consiste en una interface que permite realizar comunicación entre el dispositivo que contiene las mediciones y una computadora, por ende permitiría realizar fácilmente una aplicación que realice el registro de las mediciones. La adquisición de este producto ayudaría en la solución del problema; la desventaja es que el valor ronda en 1.200.000 colones. La compra de este producto aumentaría el coste total del proyecto.

4.3 Evaluación de las alternativas y síntesis de una solución

4.3.1 Alternativa 1

Los fabricantes de los productos del sistema fotovoltaico SMA proveen de varias alternativas para el diseño de SCADAS y transferencia de datos entre diferentes dispositivos; ejemplo de ello es el SMA OPC server. Este es un software que se instala en una computadora de escritorio y se encarga de establecer la comunicación entre el WebBox y alguna aplicación en un dispositivo o computador.; Esta aplicación debe ser compatible con el OPC.

En las figuras 4.1 y 4.2 se puede apreciar como un OPC server brinda comunicación entre la WebBox y posibles clientes.

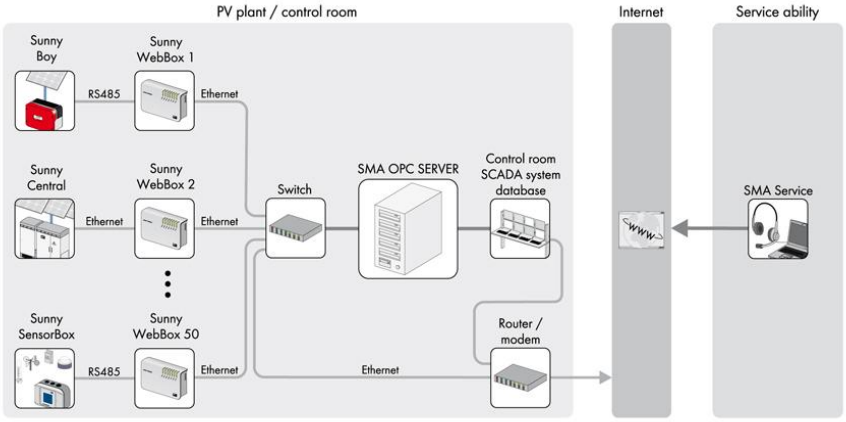


Figura 4.1 Comunicación del SMA OPC Server con un sistema fotovoltaico de forma remota [4]

Como se mencionó anteriormente el OPC server establece comunicación entre el sistema de monitoreo (webBox) y un computador, quiere decir que esta interfaz puede extraer datos del dispositivo y por medio de una aplicación, manipularlos. Se investigó que dicha aplicación puede ser implementada por medio del ambiente LabView. Esta herramienta es un ambiente en el cual se pueden programar aplicaciones de forma gráfica, haciendo que la programación sea fácil. Dicho software tiene la ventaja de contar con rutinas pre diseñadas que interactúan con el OPC server.

Se planteó una solución donde se haría el uso del OPC server y el LabView por las facilidades que estas herramientas brindan.

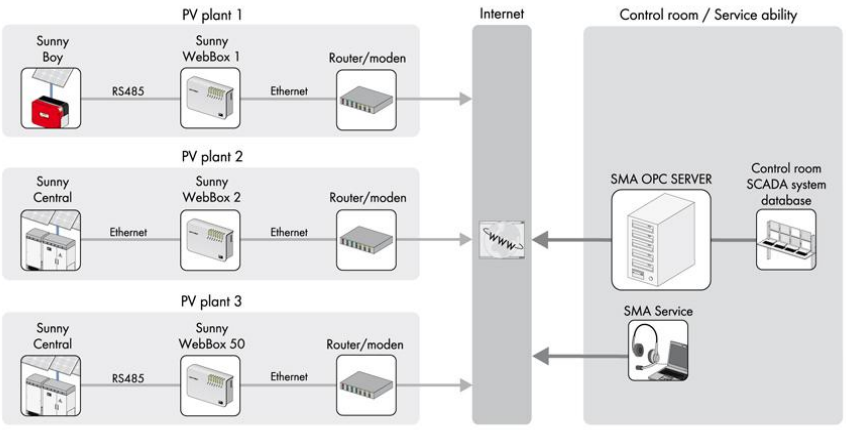


Figura 4.2 Comunicación del SMA OPC Server con varios sistemas fotovoltaico en diferentes ubicaciones.

Solución 1: Establecer comunicación entre el dispositivo WebBox y una computadora a través de una red LAN usando el OPC server; el cual permite llevar datos desde la WebBox a la computadora. Los datos recibidos se manipulan con una aplicación realizada en LabView. También dicha aplicación cosiste en una interfaz gráfica para que el usuario pueda tomar lectura de las mediciones del sistema y podrá ejecutar diferentes funciones para el análisis de las mediciones.

En la figura 4.3 se puede ver el diagrama general de bloques de esta solución.

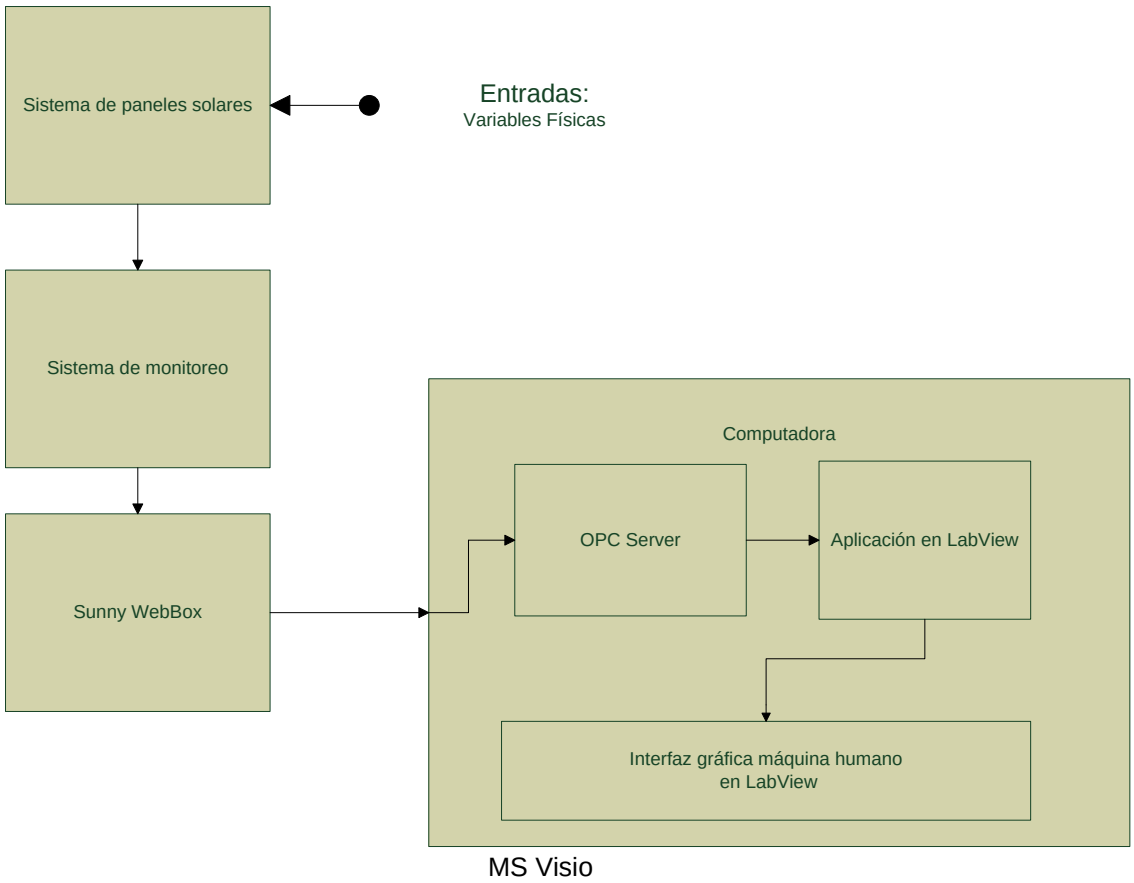


Figura 4.3 Diagrama de bloques general de la solución 1

Desventaja de la solución 1:

La implementación de esta solución parece ser sencilla una vez que se adquieran los software requeridos, sin embargo el precio de estos excede el presupuesto permitido. En la tabla 4.1 se puede ver el costo aproximado de la implementación esta solución.

Tabla 4.1 Precios de software

Artículo	**Precio [¢]**
OPC Server	2.200.000
LabView	1.400.000
Total	3.600.000

MS Word

El sistema del ICE para comprar productos de este tipo es bastante lerdo, por lo tanto la aprobación de la compra de este producto podría durar meses, es más, la estimación de este tiempo no se puede calcular, en el mejor de los casos duraría 2 meses, tiempo que afectaría la fecha de conclusión del proyecto. Por estas desventajas se procedió a buscar más alternativas.

Ventajas de la solución 1

La utilización del server permitiría establecer comunicación entre los dispositivos sin tener que preocuparse por el protocolo de comunicación.

Utilizar LabVIEW permite una programación más rápida, característica que en los lenguajes convencionales no se tiene.

4.3.2 Alternativas 2

El dispositivo de monitoreo WebBox tiene la ventaja que se puede comunicar con diferentes dispositivos por medio de varios puertos de salida: modem, puerto para tarjeta de memoria y un puerto para conexión Ethernet. Sabiendo esta característica se pensó en diseñar un módulo que se conecta con la WebBox.

Solución 2:

Se evaluó la alternativa de diseñar un sistema inalámbrico que logre extraer datos del WebBox y los envíe a un transmisor para que sean transferidos a una computadora y en la computadora realizar una aplicación que guarde y muestre las mediciones de las variables físicas. Dicha aplicación tendría otras funciones tales como presentar un registro de mediciones de forma gráfica. El diagrama de bloques se muestra en la figura 4.4.

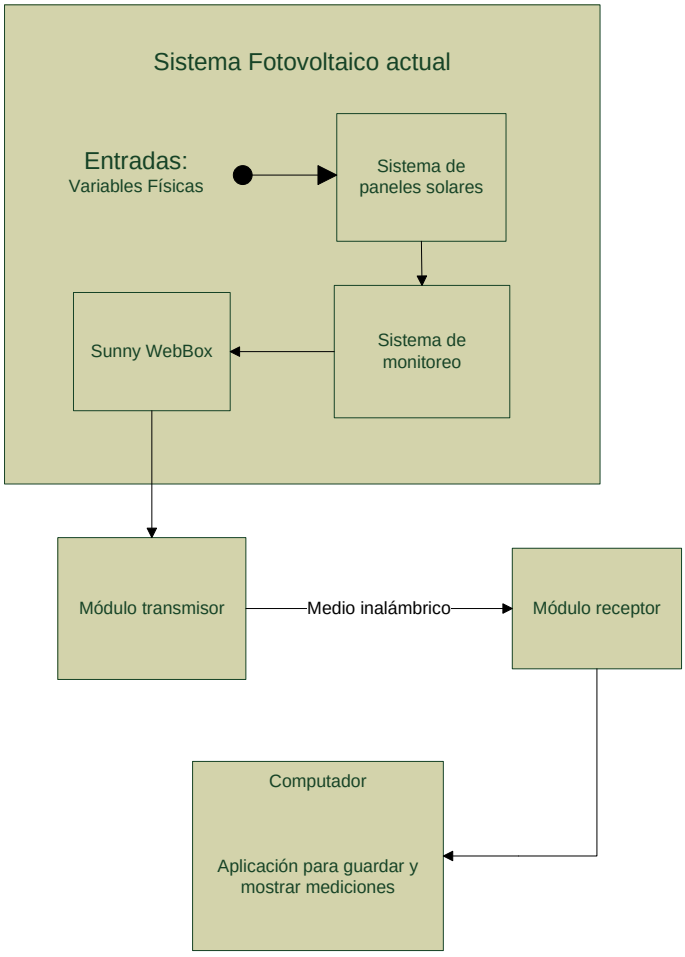
Para dicha solución se pretendió diseñar un módulo el cual se encarga de comunicarse con la WebBox para obtener datos de ella, después de obtener datos este módulo mandaría los datos de forma inalámbrica a un módulo receptor.

El módulo receptor se encarga de recibir datos del transmisor para que sean transferidos a un computador; en el cual se le instalará una aplicación la cual cumpla las siguientes características:

- Permite recibir datos del módulo receptor.
- Se encarga de crear un registro de mediciones.
- Despliega una interfaz gráfica la cual muestra de forma eficaz las mediciones del sistema y además realiza gráficas de las mediciones.

Desventajas de la solución 2:

El lugar donde se instalará el sistema está expuesto a interferencia electromagnética esto se debe a que contiguo al cuarto del sistema fotovoltaico se encuentra una subestación eléctrica. Esta situación es de importante atención ya que la interferencia afecta considerablemente los sistemas de comunicación inalámbrica. Quiere decir que a la hora de hacer el diseño hay que implementar una solución para que la interferencia no afecte la comunicación. Esto complica más el trabajo; entonces se cuestiona si realmente un sistema inalámbrico es imprescindible. El coordinador de mantenimiento expresa que la computadora remota (donde se leen las mediciones) estará a una distancia corta y en un lugar fijo por tanto un sistema inalámbrico no es necesario.



MS Visio

Figura 4.4 Diagrama de bloques de la solución 2

4.3.3 Alternativa 3 (solución escogida)

En el manual de usuario de la Sunny WebBox se encontró que el dispositivo cuenta con una colección de servicios de procedimiento que pueden ser accedidos desde una terminal remota por medio del protocolo RPC a través de una red local o una conexión RAS (Remote Access Service). El formato para el intercambio de datos es el formato JSON por sus siglas en ingles JavaScript Object Notation. Estos servicios de procedimiento permiten que un desarrollador de software realice una aplicación en cualquier lenguaje. Esta es una gran ventaja ya que el desarrollador podría escoger el lenguaje de programación más conveniente.

Para mayor información se puede referir al manual del protocolo RPC (ver anexo 2) proporcionado por la empresa SMA. Con el manual un programador podrá establecer comunicación con el sistema de monitoreo de forma adecuada.

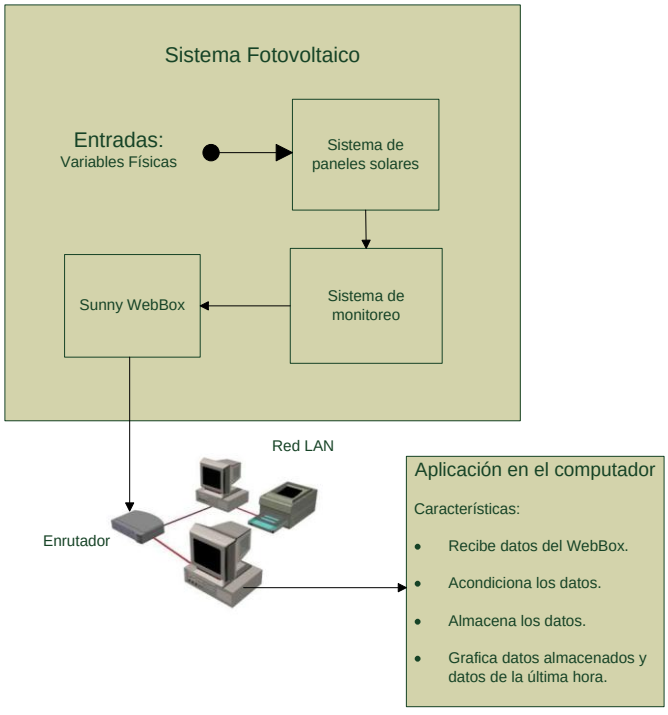
Analizando las características anteriores se procedió a idear una solución que se describirá a continuación:

Solución 3:

Realizar una aplicación en algún lenguaje de programación que logre comunicarse con el sistema de monitoreo a través de la red local; con el propósito de transferir datos desde el dispositivo a la computadora y que estos datos (mediciones) sean guardados en disco duro y que a demás sean mostrados en una interfaz gráfica. Ver figura 4.5

La aplicación debe tener las siguientes características o funciones:

a) Establecer comunicación con el dispositivo WebBox y un computador
b) Función que establezca peticiones de envío de datos desde el WebBox, de forma continua.
c) Que reciba los datos de la WebBox, los reconozca y que los traduzca de manera adecuada.
d) Almacenar los datos recibidos (mediciones) en el disco duro de la medición.
e) Interfaz gráfica donde se puedan leer todas la mediciones del sistema de monitoreo.
f) La interfaz gráfica cuenta con la opción de realizar gráficas de las mediciones en función del tiempo.



MS Visio

Figura 4.5 Diagrama de bloques general de la solución 3

Desventajas:

Todo el protocolo de comunicación y las características que tiene la aplicación, mencionadas anteriormente, deben ser implementadas programando en un nivel medio de programación; por ejemplo C++, C#, JAVA etc. Esto se ve como desventaja tomando en cuenta que hay que presentar atención en detalles que no se tomarían en cuenta en un lenguaje de programación a nivel muy alto, LabVIEW por ejemplo. También el tiempo de implementación y depuración aumentaría.

Se pueda dar el caso que se requiere aprender un lenguaje nuevo, esto podría ocurrir si un lenguaje es el más apto para el tipo de aplicación desarrollado. En tal caso se tendría que invertir tiempo en estudio e investigación.

Ventajas:

Como se tiene que diseñar todo el software, no se tendría que comprar ningún paquete computacional; evitándose el tiempo en que se dura en hacer un tedioso trámite para realizar una compra. Y por ende se economiza una cantidad considerable de dinero ver tabla 4.1

Al programar en un nivel intermedio se tendría a disposición todo los recursos de la computadora esto permite al programador contar con una gran variedad de opciones para la implantación de la aplicación.
Se puede contar con bibliotecas que realizan funciones sofisticadas, por ejemplo bibliotecas para graficar, o para realizar interfaces gráficas elaboradas.

4.4 Implementación de la solución

En la sección 4.3 se presentaron tres alternativas de solución al problema planteado; la solución 1 fue descartada debido a que se tendría que comprar dos paquetes computacionales de elevado costo; además se tomó en cuenta que el trámite para realizar una compra de tal magnitud requiere de mucho tiempo, debido al ineficiente sistema de compras que maneja la empresa.

Se analizó una segunda solución la cual involucraba diseño de hardware que involucraba comunicación inalámbrica. Esta solución no tuvo mucha aceptación ya que se tomaba en cuenta el hecho de que en el lugar donde se implementaría el diseño, estaba expuesto a interferencia electromagnética, factor que dificultaría una comunicación inalámbrica entre los dispositivos. El coordinar de mantenimiento y yo llegamos a la conclusión de que un sistema inalámbrico no era realmente necesario. Si no se tiene un sistema inalámbrico entonces convendría mas utilizar las opciones para la transferencia de datos que dispone el dispositivo WebBox, ejemplo de una opción es el software de interface con el protocolo RPC, que viene instalado en la WebBox

El software de interface con el protocolo RPC permite que un programador desarrolle una aplicación en una computadora y esta se comunique con el dispositivo WebBox mediante la red local interna (LAN) de la empresa utilizando el puerto 80 o sea una conexión Ethernet.

Explicando con más detalle: el protocolo RPC permite que una aplicación en el computador, mande una cadena de caracteres hacia el WebBox (“petición”), el software de interface de la WebBox reconoce dicha cadena como una instrucción la cual será ejecutada ahí mismo, después de la ejecución, el dispositivo responde enviando una cadena hacia el computador (“respuesta”); la respuesta contendrá información relativa a la petición. Por ejemplo: si se realiza una petición de envío de datos correspondientes a mediciones climatológicas, el dispositivo WebBox enviará al computador una respuesta donde se incluye las últimas

mediciones realizadas por el sistema, la respuesta también contiene unidades y otros códigos que se utilizan para verificación.

El tipo de formato de la cadena que se envía y se recibe se le denomina JSON (JavaScript Object Notation) este consiste en arreglos y objetos tal y como se muestra en la figura 4.6.

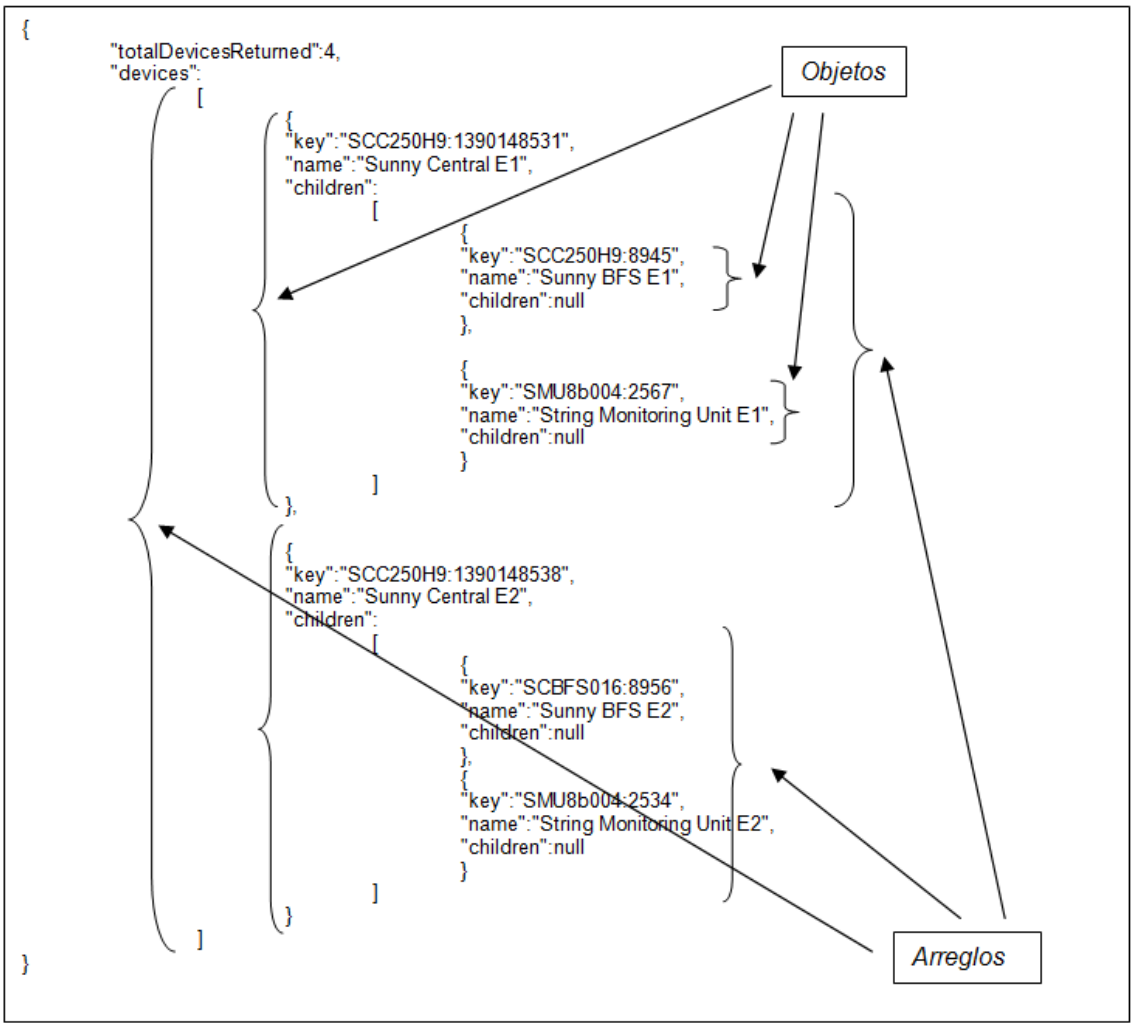


Figura 4.6 de una cadena de caracteres en formato JSON

Este formato es conocido como un formato liviano para el intercambio de datos, tiene la característica de que es fácil para el humano leerlo y escribirlo; de igual forma para un computador es fácil de interpretarlo y generarlo. El formato JSON es completamente independiente del lenguaje de programación y sus convenciones son interpretadas fácilmente por los programadores de la familia de lenguajes C, incluyendo C++, C#, Java, JavaScript, Perl, Python y muchos otros.

Se puede apreciar que la posibilidad que brinda el fabricante para transferir datos por medio del protocolo RPC puede ser bien aprovechada si se tiene conocimiento del lenguaje de programación más apto. Inclusive se puede notar que se puede escoger dentro de una gran variedad de lenguajes.

La principal enmienda es el realizar una transferencia de datos al computador y que se puedan guardar; la solución 3 permitiría realizar esta acción en su totalidad

La alternativa 3 tiene como ventaja de que no se requiere comprar ningún software computacional ni ningún dispositivo electrónico; esta es la principal ventaja por lo que se escogió la alternativa 3 como solución.

La aplicación realizada se llama “Real Time WebBox Values” y su interfaz tiene un aspecto como el que se muestra en la figura 4.7. Se puede notar que en esta ventana principal se pueden ver las mediciones de las diferentes variables físicas.

En la parte superior se observan 4 figuras las cuales simbolizan las mediciones climatológicas actuales y el resto de mediciones correspondes a mediciones de los 3 inversores A, B y C. La figura 4.8 corresponde a las variables de los inversores y es un acercamiento de la figura 4.7

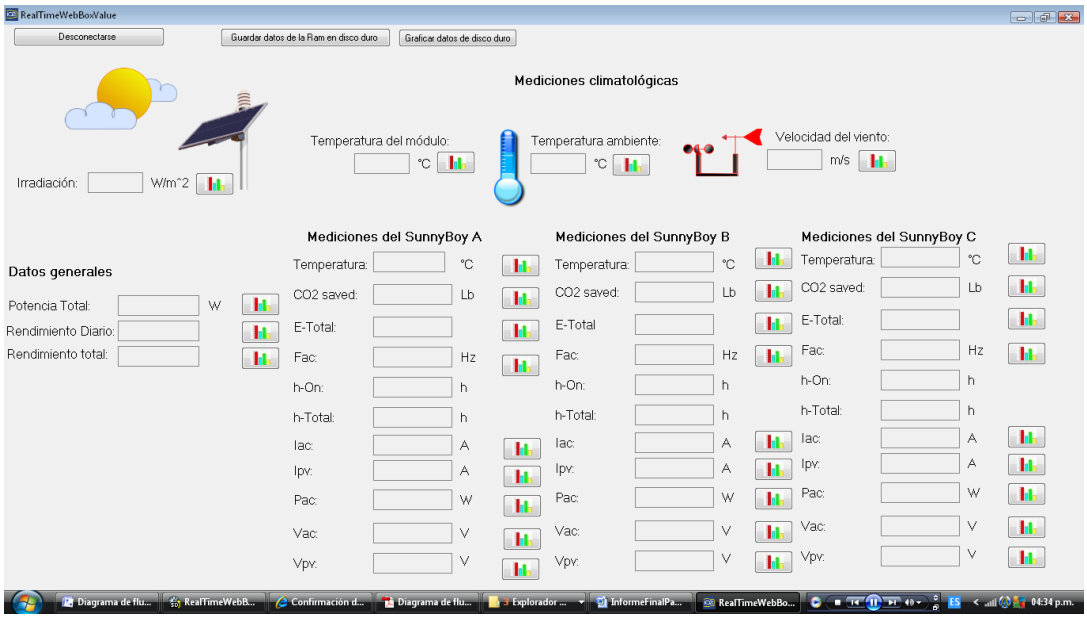


Figura 4.7 Ventana principal de la aplicación “Real Time WebBox Values

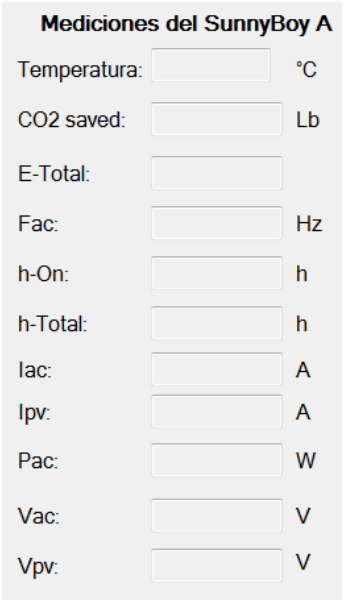


Figura 4.8 Variables que se miden al inversor SunnyBoy A

De la figura 4.7 se puede apreciar que existe un botón al lado derecho de las cajas de texto, de cada medición (botones con colores rojo verde y amarillo). Ver figura 4.9.

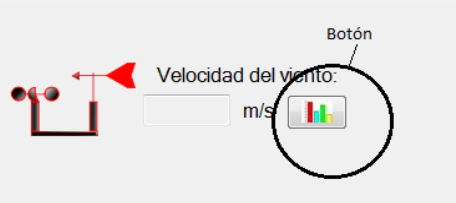


Figura 4.9 Botón para visualizar la gráfica de valores de velocidad de viento en el tiempo

Al presionar este botón aparecerá una sub ventana la cual muestra una gráfica de los valores de la medición escogida en función del tiempo. El intervalo de tiempo es el de la última hora, por ejemplo si fueran las 13:45 pm y se presiona el botón que está al lado derecho de la caja de texto de potencia total; aparecerá una gráfica de los valores de mediciones realizadas desde las 13:00 hasta las 13:45, por supuesto mediciones de potencia total. Ejemplo de la gráfica realizada por la aplicación se puede apreciar en la figura 4.10.

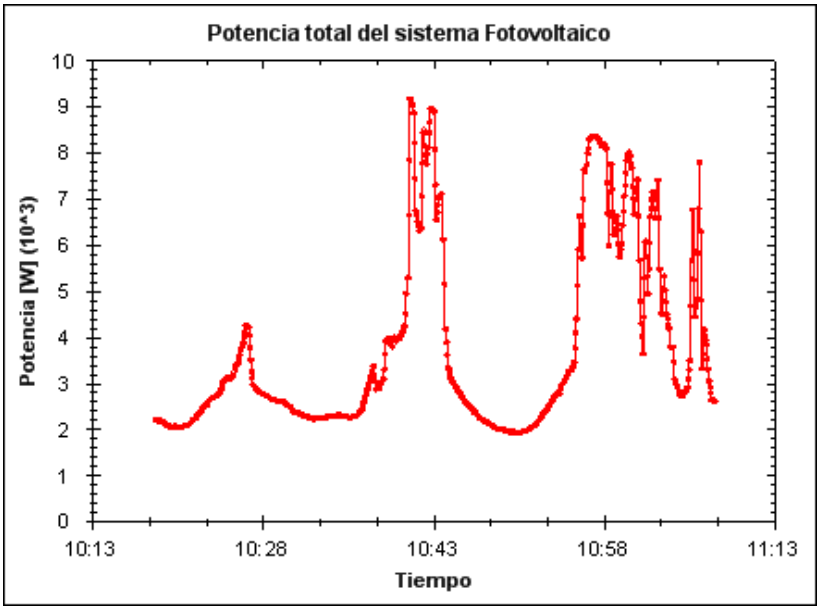


Figura 4.10 Gráfica generada por la aplicación correspondiente a mediciones de potencia total generado por el sistema fotovoltaico desde 10:13 am hasta 11:13 am.

En la parte más superior de la ventana principal aparecerán 3 botones, ver figura 4.11. El primer botón corresponde a la opción de desconectar/conectar esto permite desconectarse del dispositivo WebBox; hecho que provocaría que se interrumpiera la transferencia de datos, por tanto se dejaría de mostrar mediciones en la ventana principal. Esta opción es útil ya que si otro dispositivo quiere acceder al WebBox se necesita que este no se esté comunicando con otro dispositivo. Esto se deba a que solo un dispositivo se puede conectar.

Figura 4.11 Vista de acercamiento de la figura 4.7 de tres botones

El segundo botón (Guardar datos de la RAM a disco duro) permite guardas las mediciones que se encuentran en memoria principal y las salva en el disco duro en un archivo de texto (txt) una vez guardado en disco duro se libera memoria principal.

Se explicó que se cuenta con una opción que consiste en graficar datos de la última hora, sin embargo también se cuenta con la opción de graficar mediciones de días pasados. El usuario al presionar el botón (Graficar datos del disco duro) aparece una ventana en la cual se escoge la variable y el intervalo de tiempo que se quiere graficar. Tal como se ilustra en la figura 4.12. Si el intervalo de tiempo incluye un día donde no se tomo mediciones, la aplicación lanzará un mensaje donde se indica que no se encontró ningún archivo correspondiente al día que no se tomaron mediciones.

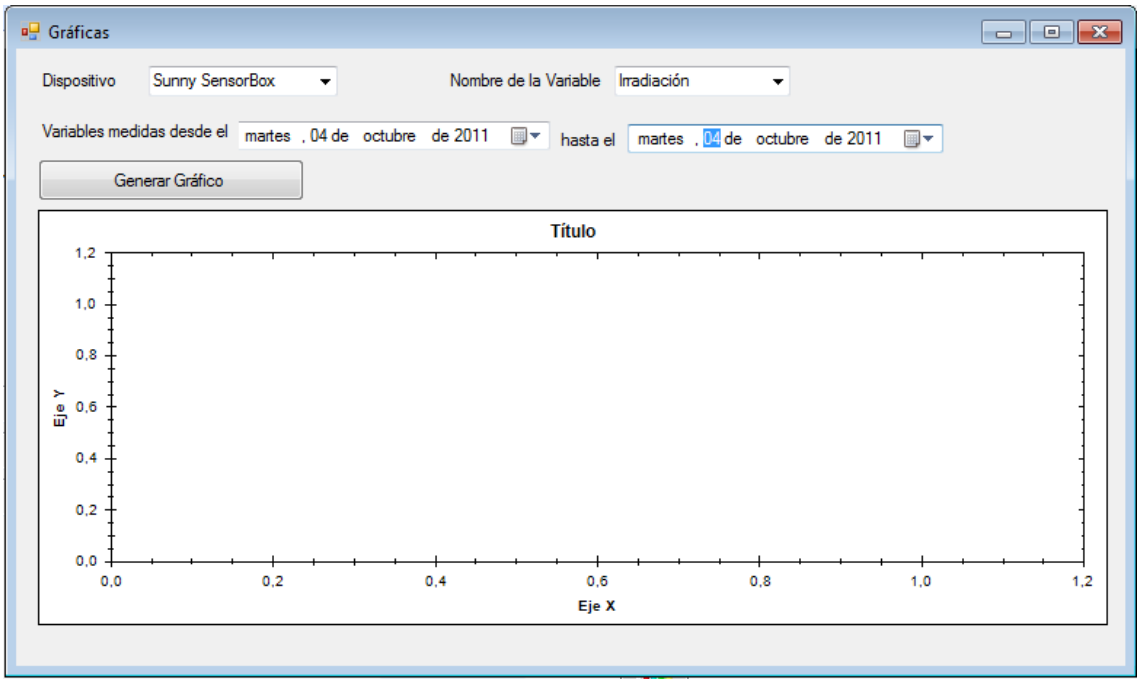


Figura 4.12 Ventana para graficar mediciones de días pasados

4.5 Reevaluación y rediseño

La implementación soluciona el problema planteado sin embargo hay consideraciones que se deben tomar por ejemplo si al sistema se le agrega otro inversor el programa no lo reconoce y por tanto no muestra mediaciones que se le pueden hacer a este dispositivo. De igual forma sucede si un inversor es remplazado por otro. Para remediar esto se tendría que cambiar las líneas de código de la aplicación. Por esta razón sería bueno realizar dentro de la aplicación un módulo que detecte los dispositivos instalados en el sistema fotovoltaico

Es importa encender una alarma en caso de que alguna variable física se salga del rango de operación correcta, por ejemplo si la temperatura excede los 50□ C se podría tener la sospecha de algún mal funcionamiento del sistema o inclusive un posible incendio. Actualmente el sistema no que quedó con un sistema de alarmas pero si es algo que se debería implementar.

El ICE cuenta con otros sistemas fotovoltaicos en diferentes regiones del país los cuales no tienen implementado el sistema de registro de datos (“Real Time WebBox Values”), por lo que sería útil hacerle modificaciones a esta aplicación para que se comunique con todos los sistemas fotovoltaicos del ICE.

Capítulo 5: Descripción detallada de la solución (Explicación del diseño)

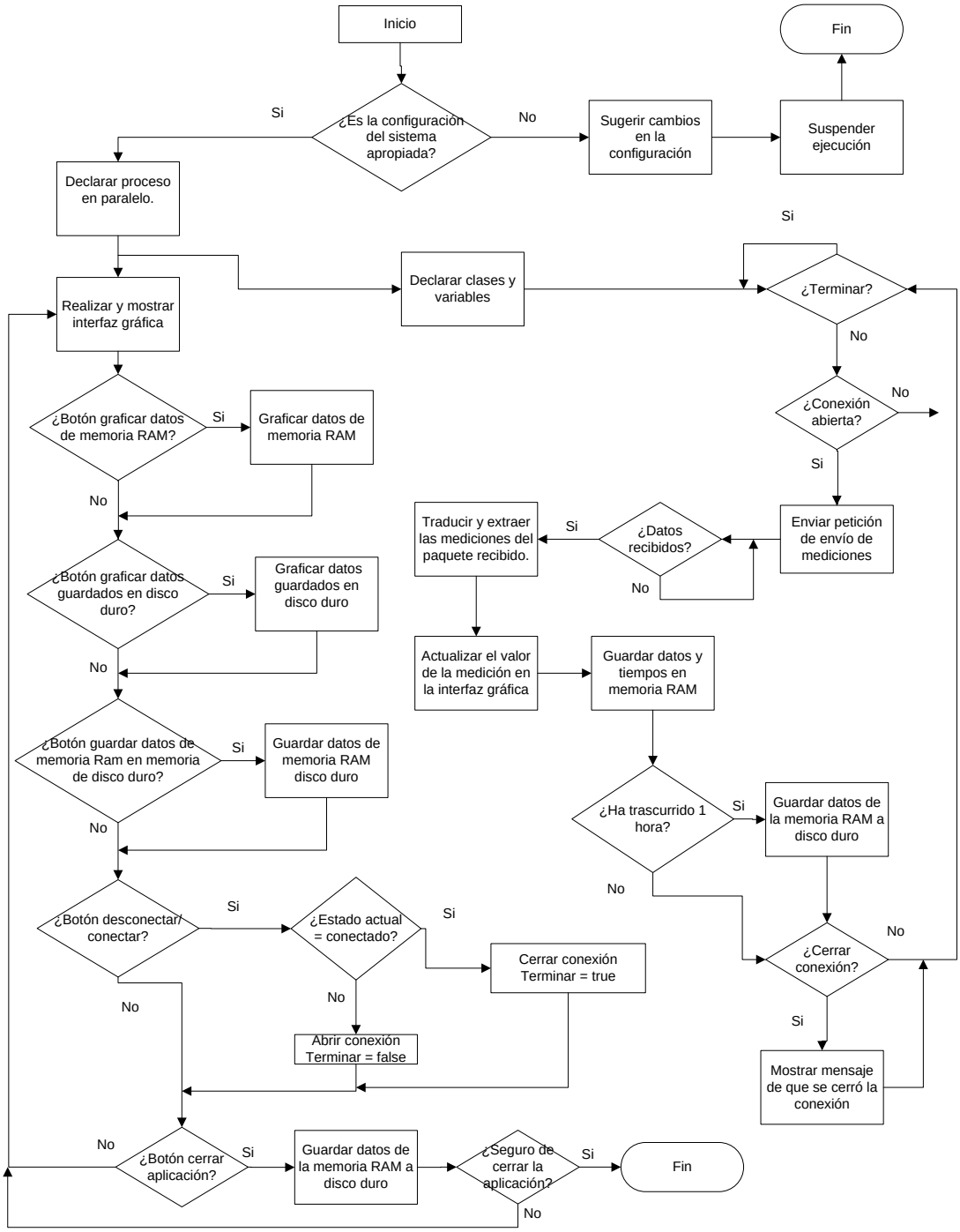
5.1 Análisis de soluciones y selección final

Antes de empezar la implementación de la aplicación se tuvo que escoger el lenguaje de programación. Investigando se analizó que el lenguaje C# presenta varias ventajas sobre otros lenguajes, que hacen su escogencia la mas predilecta; las ventajas se enlistan a continuación:

a) La filosofía de este lenguaje es tratar de unificar los lenguajes más populares en uno solo. Verdadera ventaja ya que el lenguaje C# se asemeja en gran manera con el lenguaje C++ (lenguaje muy conocido por los estudiantes de electrónica del TEC)
b) Este lenguaje cuanta con una gran variedad de bibliotecas que ayudan a simplificar funciones tediosas.
c) Muchas bibliotecas se consiguen de forma gratuita, están bajo el concepto de software libre.
d) El uso de punteros es opcional, no es realmente necesario utilizar punteros, esto facilita la programación.
e) Existe un ambiente de desarrollo integrado (IDE) libre (gratis), que permite la programación y la gestión de gráficos para tener una programación más sencilla.
f) El fabricante del dispositivo WebBox (SMA) desarrolló una biblioteca en este lenguaje, la cual permite el manejo del protocolo RPC con el WebBox. Esta vendría siendo la característica más ventajosa de este lenguaje

El diseño de la aplicación se realizó por medio de diseño modular, se implementaba un módulo, se verificaba que estuviera correcto por medio de una aplicación sencilla y se procedía a unirlo al resto de la aplicación.

En la figura 5.1 se muestra el diagrama de flujo de la aplicación. Dicho diagrama se explica con detalle en las siguiente sub secciones. Cada una de las secciones corresponde a cada uno de los módulos que se presentan en el diagrama general de flujo; se recomienda al lector hacer referencia a este diagrama para tener un mejor entendimiento de los módulos.



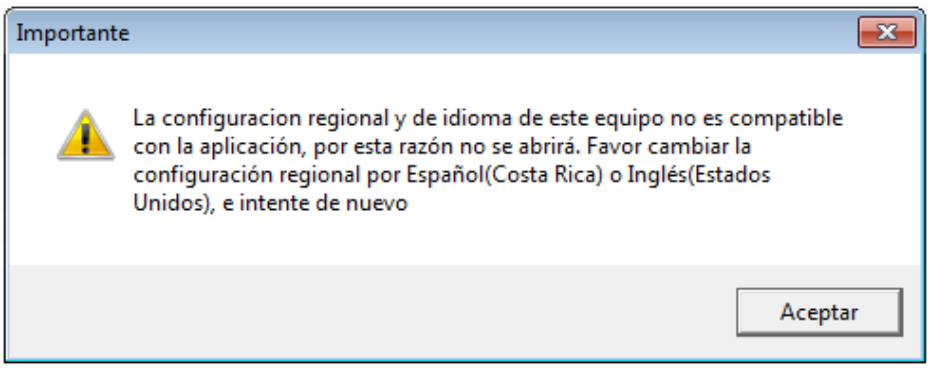
MS Visio

Figura 5.1 Diagrama de flujo de la aplicación Real Time WebBox Value

5.2 Descripción del software (por módulos)

Si nos referimos a la figura 5.1 se aprecia que al inicio del diagrama se verifica que el sistema tenga la configuración apropiada. Este paso es de suma importancia ya que uno de los métodos para graficar requiere que el formato de los números decimales sea con la coma decimal y el formato de la fecha y hora sea el que se utiliza en Costa Rica (dd/MM/aaaa) para fecha, y (hh:mm:ss tt) para la hora. Un formato que también se podría utilizar es el usado en Estados Unidos.

Si el sistema operativo tiene otra configuración, la aplicación lanza una pequeña ventana donde aparece un mensaje de error y además se recomienda que se cambie la configuración regional y de idioma por la de Es-CRC o En-USA. La ventana se aprecia en la figura 5.2. El programa no se cargará al menos que no se cambie a la configuración apropiada.



Paint

Figura 5.2 Ventana que avisa que la configuración regional y de idioma no es la correcta.

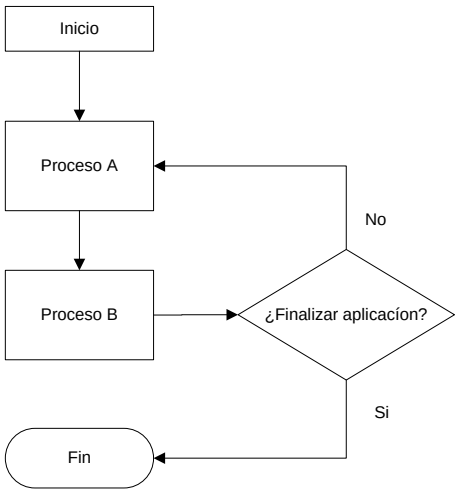
Una vez que la configuración regional y la de idioma es la apropiada se sigue la línea de flujo y se llega al módulo, que declara un proceso que se va a ejecutar en paralelo, a esto también se le llama proceso en hilo.

5.2.1 Declarar proceso en paralelo

En la figura 5.1 se observa que si la condición“¿Es la configuración apropiada?” es si, se pasa al módulo “Declarar proceso en paralelo” El cual se explica a continuación.

Este módulo permite que la computadora realice dos procesos simultáneamente; técnica que es de suma importancia ya que en el proceso de desarrollo se notó que la aplicación, sin procesos en paralelo, se tornaba extremadamente lenta. Esto por el siguiente motivo:

La aplicación cuenta con dos procesos principales, nombrémoslos como Proceso A y B. El primero es el encargado de actualizar la interfaz gráfica y de atender las acciones realizadas por el usuario (por ejemplo el hacer click a un botón). El otro (Proceso B) es el encargado de enviar solicitudes de datos, recibirlos, almacenarlos guardarlos y mostrarlos en la interfaz gráfica. Debido al tipo de aplicación estos dos procesos se tienen que realizar de forma continua es decir que una alternativa sería que estos dos procesos se realicen en un lazo o ciclo como se ve en la figura 5.3.



MS Visio

Figura 5.3 Diagrama de flujo ineficiente de dos procesos.

Nótese que si un proceso no se ha terminado no se puede empezar el otro, esto no sería un factor negativo si ambos procesos se realizan de forma rápida; tal que el usuario no notara el cambio de un proceso al otro. El problema es que el proceso B dura mucho en realizarse (aproximadamente 3 segundos) esto se debe a que hay que recibir 40 mediciones del WebBox; y para obtener cada medición se requiere de un proceso largo, el proceso se explica en el módulo “Enviar petición de datos” y las mediciones de tiempos se muestran en la tablas 6.7 a la 6.22.

El problema surgía cuando se estaba ejecutando el proceso B, y el usuario cliqueaba un botón; se notó que el programa no respondía a esta acción y se debe a que las acciones de las pulsaciones de botón se evalúa en el Proceso A, y este no se ejecutaba sino hasta que el proceso B terminara. Se buscó una solución y la más eficiente fue la de realizar estos dos procesos en paralelo. Para ello se utilizó la biblioteca “System.Threading” la cual permite declarar procesos en paralelo. Se implementó esta técnica y el programa se corría eficientemente.

Volviendo al diagrama de flujo principal se ve que después del módulo “Declarar proceso en paralelo” el flujo se divide en dos partes estos corresponden a los dos procesos en paralelo; uno es el proceso A el cual inicia con el módulo “Realizar y mostrar interfaz gráfica” y el proceso B que empieza con el módulo “Declarar clases y variables”.

5.2.2 Declarar clases y variables

En el desarrollo de la aplicación se encontró que varios procesos compartían muchas características por lo que se planteó realizar clases de aquellos métodos que se parecían entre si (más que todo compartían funciones). Se recurrió a implementar clases con el propósito de ahorrar tiempo al escribir menos líneas de código.

Dichas clases de nombran y explican en la tabla 5.1. Cada una de las clases se explicarán adelante, con la explicación del módulo que las contiene

Tabla 5.1 Lista de clases implementadas en la aplicación Real Time WebBox Value

Nombre de clase	Descripción de la clase	Relación entre clases
Nodo	Implementa un nodo con puntero o referencia	Utilizada en la clase ListaEnlazable para crear la lista
ListaEnlazable	Crea una lista enlazada y además incluye varias funciones para manipular la lista	Hereda funciones de la clase Manejador de datos
ManejadorDeArchivosTxt	Realiza funciones para trabajar con archivos “txt”.	Es una clase padre de la clase ListaEnlazable
GráficasDelDia	Contiene funciones para realizar graficas de alguna medición	Utiliza la clase ListaEmlazable
Gráficas	Permite realizar gráficos de mediciones almacenadas en disco duro, se cargan datos de disco duro en una lista enlazable	Utiliza la clase ListaEmlazable

5.2.3 Enviar petición de envío de mediciones

Este módulo fue el primero que se implementó debido a que este representa la columna vertebral de la aplicación. Lo más importante es lograr obtener datos de la WebBox y lograr manipularlos mediante rutinas de programación.

Revisando documentación del funcionamiento de la WebBox se encontró que se cuenta con una biblioteca la cual posee una colección de funciones útiles para proporcionar comunicación entre la computadora (la aplicación) y el WebBox.

Una de las funciones que se utilizó fue “Coneccion.JsonRequest()” que pertenece a la biblioteca WebBox.RPC.dll, esta es proporcionada por el fabricante del dispositivo. Esta función se encarga de enviar una cadena al dispositivo WebBox; dicha cadena corresponde a una petición de procedimiento. O sea que en el dispositivo se tendrá que ejecutar un procedimiento y después enviar al computador una respuesta, que dependerá del tipo de procedimiento.

Las respuestas que devuelve el WebBox por lo generan son mediciones de variables de cierto grupo. En total hay cuatro grupos de variables que se especifican en la tabla 5.2

Entonces por ejemplo si se ejecuta la siguiente línea de código:

```
result = Coneccion.JsonRequest(Procedures.GetPlantOverview());
```

El computador envía una petición al WebBox de obtener el grupo de mediciones resumen. Con esto el dispositivo devuelve un paquete de datos en formato Json conteniendo las mediciones de tres variables (Potencia total, Rendimiento diario y Rendimiento Total) según la tabla 5.2. Dicho paquete de datos se guarda en la variable “result” que es de tipo JsonObject.

Para obtener los otros tres grupos de mediciones de variables se utilizan otros tres procedimientos enlistados en la tabla 5.2

Se puede apreciar que el grupo de mediciones eléctricas resumen consta de tres variables el grupo de mediciones climatológicas contarían 4 variables y el grupo de los inversores tiene 16 variables de medición de las cuales solo se presentan 11. De la tabla nótese que el procedimiento para adquirir mediciones del alguno de los inversores se necesita especificar el número de dicho inversor dentro de los paréntesis de la función, se vería así: “Procedures.GetProcessData('' *número_inversor*'')”

Tabla 5.2 Clasificación por grupos y definición de las variables que se miden

Grupo de mediciones	**Nomenclatura de la variable**	**Descripción de la variable**	**Procedimiento para extraer las mediciones**
Mediciones de las variables eléctricas resumen	Potencia Total	Potencia producida por los tres inversores	Procedures.GetPlantOverview()
	Rendimiento diario	Energía total producida durante el día	
	Rendimiento total	Energía total producida desde que se instalaron los paneles	
Mediciones de las variables climatológicas	Irradiación	Irradiación del sol	Procedures.GetProcessData("SENS0500:3087")
	Temperatura ambiente	Temperatura ambiente	
	Temperatura del módulo	Temperatura del módulo que mide las variables climatológicas	
	Velocidad del viento	Velocidad del viento donde se ubican los paneles	
Mediciones de variables proporcionas por los inversores A, B y C	Temperatura	Temperatura del inversor	Procedures.GetProcessData("*inversor*")
	CO2 saved	CO2 ahorrado	
	E-Total	Energía total producida por un inversor	
	Fac	Frecuencia AC en la salida del inversor	
	h-On	Horas de funcionamiento del inversor durante el día	
	h-total	Horas totales de funcionamiento	
	Iac	Corriente alterna producida en la salida del inversor	
	Ipv	Corriente directa producida en un arreglo de paneles (entrada del inversor)	
	Pac	Potencia producida en la red CA	
	Vac	Tensión alterna producida en la red	
	Vpv	Tensión directa del arreglo de paneles	

5.2.4 Traducir y extraer mediciones del paquete recibido

Una vez que en el computador recibe un paquete de datos correspondiente a mediciones de variables del alguno de los cuatro grupos, se debe de extraer del paquete la magnitud y la unidad de las mediciones para que posteriormente otros módulos utilicen esta información. Este proceso consiste básicamente en manipular la cadena de caracteres en formato Json y de esta obtener los datos necesarios, para guardarlos en variables tipo string. La variable string permite presentar datos en la interfaz gráfica y también existe la posibilidad de transformar la variable string a un tipo de variable flotante para aplicar operadores matemáticos, o a la hora de graficar datos.

5.2.5 Actualizar el valor de la medición en la interfaz gráfica

Partiendo del hecho de que ya las mediciones recibidas están guardadas en memoria principal en un tipo de variable string, se procede a actualizar este valor en la interfaz gráfica. Este procedimiento es delicado ya que el refrescamiento de la interfaz gráfica (Proceso A) se ejecuta de forma paralela con el Proceso B y dentro de este proceso se está ejecutando el módulo que actualiza el dato en la interfaz gráfica; entonces se podría dar conflictos cuando el Proceso A está refrescando la caja de texto y simultáneamente se está escribiendo un valor a la caja.

Para evitar conflictos dentro de este módulo se imprentó un método el cual consiste en evaluar si la caja de texto está lista para ser escrita con el nuevo dato, si no está lista entonces este espera a que lo esté; entonces se escribe en la caja de texto

5.2.6 Guardar datos y tiempos en memoria RAM

Unos de los principales requerimientos era que las mediciones pudieran ser guardadas en memoria para poder realizar análisis.

Para lograr tal acometido se escogió guardar las mediciones en una lista enlazada de datos, este es el método más eficaz ya que no se sabe con exactitud el número de mediciones que se van a guardar y guardar en memoria principal es relativamente más rápido que estar guardando en disco duro.

Para implementar la lista enlazada se creó una clase llamada “ListaEnlazada” esta clase tiene muchas funciones, alguna de ellas es realizar una lista enlazada y aplicarle operaciones como por ejemplo insertar un nuevo elemento a la lista. Cada elemento de la lista, o nodo, contiene dos tipos de datos, uno corresponde a un dato tipo string que es donde se encuentra la mediación y el otro es de tipo DataTime donde se guarda la hora y fecha del instante donde llegó la medición al computador. Guardar el tiempo es necesario ya que se requiere graficar estos datos en función del tiempo.

En la aplicación se hizo una instancia de la clase ListaEnlazable por cada variable de medición; a cada instancia se le daban atributos tales como nombre y unidades de la variable.

5.2.7 Guardar datos de la memoria RAM a disco duro

Se decidió guardar los datos de forma continua en memoria RAM ya que esta tiene tiempo de escritura y lectura más bajo. Caso contrario sería guardar los datos en disco duro, este método implicaría hacer accesos al disco duro continuamente, hecho que produciría que el disco duro se dañara y la aplicación se tornaría muy lenta. Pese a que guardar en disco duro requiere mucho tiempo si es necesario guardar los datos en un sistema de almacenamiento permanente. Si la medición se guardase solamente en memoria principal obviamente estos datos se perderían a la hora de cortar la alimentación y además la memoria se saturaría y dejaría de guardar datos. Por estos motivos se decidió guardar los datos de memoria RAM en disco duro cada hora y cada vez que el usuario lo quisiera. También si se quisiera cerrar la aplicación esta guardaría los datos antes de que el programa se cerrara. Una vez que la aplicación salve los datos estos se borran de la memoria principal, con el propósito de no saturar la memoria.

El método, que se encarga de guardar datos de RAM a disco duro, se encuentra dentro de la clase ListaEnlazable, recuerde que esta clase contiene una lista enlazada de datos correspondientes a una variable. Entonces si se llama al método de guardar en disco duro desde la clase ListaEnlazable; el método guarda las mediciones correspondientes a dicha clase.

Las mediciones se guardan en el disco duro en un archivo txt. En la figura 5.4 se muestra como se guarda la información en el archivo txt.

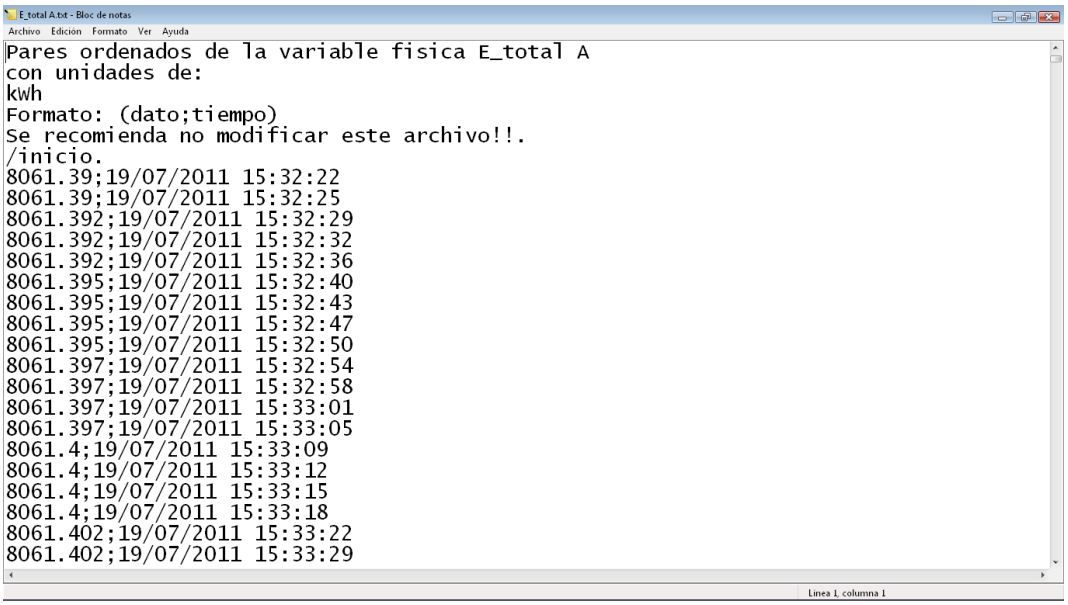


Figura 5.4 Contenido del archivo txt que se genera

Se puede notar que los datos se guardan en pares ordenados separados por punto y coma y además se especifica la variable y las unidades. Es importante remarcar que cada archivo generado corresponde a mediciones de una variable física en un día determinado; quiere decir que para un día nuevo se tendrá un archivo nuevo.

La clase ListaEnlazable hereda funciones de la clase ManejadorDeArchivosTxt precisamente esta clase se encarga de gestionar todo lo relevante a los archivos txt. Cuando se requiera guardar datos, esta clase se encarga de crear el directorio donde se guardar el archivo txt, este directorio consiste en carpetas organizadas por año, mes y día. Entonces por ejemplo si se van a guardar datos de la variable potencia total medida el 19/07/2011, se crea el directorio “C:\Variables Sunny\2011\Julio\19 Martes” y dentro de la carpeta “19 Martes” se guarda el archivo llamado “potencia total”. Si se requiere guardar más datos ese mismo día la clase detecta que ya existe el archivo y lo que se hace es agregar más información al archivo.

Este método se realiza mediante un proceso en paralelo de forma que si se están guardando muchas variables a disco duro la aplicación no se verá afectada.

5.2.8 Mostrar mensaje de que se cerró la conexión

Si el usuario quiere cerrar la aplicación aparecerá un mensaje de confirmación en el cual se le preguntado al usuario si realmente quiere cerrar la aplicación. También se guardarán las mediciones y se cerrará la conexión del dispositivo WebBox.

5.2.9 Realizar y mostrar Interfaz gráfica"

El proceso A empieza con este módulo y es el encargado de presentar la interfaz gráfica.

La interfaz gráfica se realizó con el gestor de formas de Windows, herramienta proporcionada por el IDE SharpDevelop. Dicha interfaz se construyó basándose en una lista de requerimientos la cual se planteo a partir de los objetivos propuestos. La lista de requerimientos se observa en la tabla 5.3, en ella se muestra la descripción de lo que se hizo en la interfaz gráfica partiendo de los requerimientos; además se muestran los módulos correspondientes.

Dichos módulos se muestran en el diagrama de flujo principal y se explican en las secciones siguientes.

La ventana principal consiste principalmente en botones que corren subrutina y visualización de todas las mediciones realizadas por el sistema.

Tabla 5.3 Requerimientos de la aplicación

Requerimiento	Descripción en la interfaz gráfica	Módulo correspondiente
Mostrar las mediciones de todas las variables	Todas las mediciones se visualizan en la ventana principal de la aplicación	Interfaz gráfica y presentación de las mediciones
Presentar gráficas de mediciones en función de tiempo de la última hora	Se cliquea el botón que se ubica al lado de cada medición; esto despliega una gráfica de la última hora	Graficar datos de la RAM
Guardar mediciones en disco duro	Se cliquea el botón (Guardar datos de RAM a disco duro), con esto se genera un archivo txt de cada una de las variables; conteniendo en él, mediciones y tiempo de la realización de la medición. Después el archivo se guarda en disco duro	Guardas datos de la memoria RAM a disco duro.
Presentar gráficas de mediciones guardadas en disco duro	Se cliquea el botón (Graficar datos de disco duro); se abre una ventana donde se escoge el intervalo de tiempo que se quiere graficar	Graficar datos guardados en disco duro
Conectar/Desconectarse de la WebBox	Se desconecta de la WebBox; hecho que provoca la interrupción de recepción de datos	Envío y ,recepción de datos

La interfaz gráfica proporciona símbolos y representaciones gráficas que ayudan a tener una mejor comprensión del uso de la aplicación.

En la figura 5.1 se puede notar que el módulo anteriormente descrito forma parte de un ciclo que se repite hasta la finalización del programa. Dentro de este ciclo la aplicación también evalúa si el usuario presionó uno de los botones. Si se presionó un determinado botón se ejecutará una subrutina o módulo.

Por ejemplo, si se vuelve al diagrama de flujo y si se presiona el botón de “Graficar datos de RAM” se ejecuta el módulo correspondiente, el cual se explica a continuación.

5.2.10 Graficar datos de la memoria RAM

Uno de los requerimientos según la tabla 5.3 es realizar gráficas de las mediciones más actuales, estas serian las mediciones de la última hora. La realización de dicha tarea fue sencilla gracias a la existencia de una biblioteca llamada ZedGraph.dll.

Se implementó una clase llamada “Gráficas_del_día” la cual posee una función cuyo parámetro de entrada es una variable de tipo ListaEnlazable; implica que a la función le entra la lista enlazada de los datos de las mediciones que se quieran graficar; y también los atributos nombre de variables y unidades; parámetros importantes a la hora de rotular la gráfica.

Dentro de la clase “Gráficas_del_día” se utiliza la clase “zedGraphControl1.GraphPane” dicha clase se encarga de realizar la gráfica, lo único que el programador deberá hacer es, darle los parámetros adecuados. Esto serian una lista de pares ordenados (dato;tiempo) titulo de la gráfica y títulos de los ejes. Los pares ordenados provienen de la lista enlaza la cual es un parámetro de entrada, como se explicó anteriormente.
Dicha clase se instanciará por cada gráfica que se quiera generar.

5.2.11 Graficar datos guardados en disco duro

Al darse un botón se instancia una clase la cual se encarga de presentar una ventana donde aparecen varias opciones, tal como se aprecia en la figura 4.12. En dicha ventana el usuario escoge el grupo de variables y la variable en específico que se quiere graficar; también se cuenta con la opción de elegir el periodo de tiempo que se quiere graficar. La clase utiliza toda esta información para realizar una búsqueda de los archivos, aquí es donde se usa la función “CargarTxtFileValuesEnLista” esta es una función de la clase ListaEnlazable y lo que hace es buscar el archivo según la fecha inicial y fecha final del periodo y también por el nombre de variable. Una vez encontrado el archivo se cargan los pares ordenados del archivo txt en una lista enlazada. La lista se manda a la función graficar y entonces se genera el gráfico; el título del gráfico y el titulo de los ejes son proporcionados por el archivo.

5.2.12 Abrir y cerrar conexión

El módulo abrir y cerrar conexión se corre cada vez que el usuario oprime el botón correspondiente o cuando la aplicación se cierra, dicho procedimiento se realiza por medio de la función “Coneccion.Close()”; es parte de la biblioteca WebBox.RPC.dll:

Capítulo 6: Análisis de Resultados

6.1 Resultados

A continuación se presentan mediciones que ayudan a validar el objetivo 1 en las tablas de la 6.1 a la 6.5 se muestran el tiempo que demora un paquete de datos en llegar al computador. En las tablas se aprecia que se cuanta con 5 paquetes de datos o 5 grupos de datos los cuales se detallaron en la tabla 5.2.

En las tablas se enlistan 1000 mediciones de latencia de cada paquete y en la tabla 6.6 se muestran las latencias promedio de cada paquete, también se muestran cálculos como taza de transferencia de datos, tiempo promedio por dato

Tabla 6.1 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador; medición 1 a la medición 20

Paquete de 16 datos	**Paquete de 16 datos**	**Paquete de 16 datos**	**Paquete de 3 datos**	**Paquete de 6 datos**
SunnyBoy A	**SunnyBoy B**	**SunnyBoy C**	**Over View**	**Estación meteorológica**
Latencia [s]	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**
1,4843750	1,2031250	0,6093750	0,46875	0,8437500
0,5000000	0,5468750	0,5937500	0,9375	0,3906250
0,5000000	1,0468750	1,5937500	0,484375	1,2500000
1	0,5312500	0,5156250	0,5625	0,3593750
0,5625000	0,5312500	0,5625000	0,59375	0,4062500
0,5625000	1,0312500	1,4375000	0,921875	0,3906250
1,1718750	0,5781250	0,5312500	0,515625	0,8593750
0,5468750	1,1250000	0,9687500	0,6875	0,3750000
0,9843750	1,0937500	0,5468750	0,484375	0,3906250
0,5156250	0,5312500	0,5156250	0,890625	0,4062500
1,1875000	0,5781250	1,0156250	0,65625	0,8906250
1,0468750	1,1875000	0,6093750	0,421875	0,4218750
0,5156250	0,5156250	0,5937500	0,5	0,4843750
1,4375000	1	1,4062500	0,5	0,9062500
0,5625000	0,5156250	0,5000000	1,015625	0,4687500
0,5468750	0,7500000	0,5156250	0,5	0,3906250
1,4843750	1,0625000	1,0937500	0,53125	0,5000000
0,5937500	0,7343750	0,5312500	0,46875	0,9218750
0,5156250	1,0625000	1,3906250	1,234375	0,3750000
0,9531250	0,5312500	0,5000000	0,53125	0,4375000

Tabla 6.2 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador medición 21 a la medición 40

Paquete de 16 datos	**Paquete de 16 datos**	**Paquete de 16 datos**	**Paquete de 3 datos**	**Paquete de 6 datos**
SunnyBoy A	**SunnyBoy B**	**SunnyBoy C**	**Over View**	**Estación meteorológica**
Latencia [s]	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**
0,5468750	0,7343750	0,5312500	0,46875	0,3593750
0,5156250	0,9687500	1,1562500	0,6875	1,0625000
0,9843750	0,5468750	0,5468750	0,46875	0,3593750
0,7500000	0,8125000	1,2031250	0,984375	0,4062500
1,0156250	0,9531250	0,5000000	0,484375	0,8281250
0,6093750	0,5312500	0,9531250	0,5625	0,6875000
0,6875000	0,9531250	1,0312500	0,53125	0,4218750
1,1093750	0,5625000	0,5468750	0,78125	0,3750000
0,6250000	1,0312500	0,5000000	0,703125	0,8593750
0,9687500	1,0781250	1	0,453125	0,3593750
1,4375000	0,5312500	0,5781250	0,5	0,4062500
0,5781250	0,5937500	0,5625000	0,5	0,3906250
1,2187500	1,1875000	1,2031250	1,03125	0,8906250
0,5156250	0,5312500	0,6093750	0,5	0,4218750
0,5937500	1,1406250	1	0,515625	0,7656250
0,9531250	0,6562500	0,5000000	0,484375	0,3437500
0,5468750	0,5156250	0,7500000	0,640625	0,9062500
0,9218750	1,0468750	1,0625000	0,890625	0,4062500
0,5156250	0,5468750	0,5312500	0,5	0,6250000
0,5937500	0,5000000	1,4687500	0,53125	0,7968750

Tabla 6.3 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador medición 41 a la medición 60

Paquete de 16 datos	**Paquete de 16 datos**	**Paquete de 16 datos**	**Paquete de 3 datos**	**Paquete de 6 datos**
SunnyBoy A	**SunnyBoy B**	**SunnyBoy C**	**Over View**	**Estación meteorológica**
Latencia [s]	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**
1,4843750	1	0,5312500	0,46875	0,4218750
0,5312500	0,5312500	0,5625000	1,109375	0,3906250
0,5156250	1,3437500	1,2968750	0,578125	0,3906250
0,9843750	0,5000000	0,5625000	0,4375	1,4531250
0,7500000	0,5000000	0,5000000	0,515625	0,3593750
1,0156250	1,2343750	0,9843750	0,515625	0,4062500
0,6093750	0,5781250	0,5468750	0,984375	0,5312500
0,5781250	0,5468750	0,9531250	0,640625	0,8750000
1,5156250	1,0156250	0,5781250	0,46875	0,4062500
0,5000000	0,5625000	0,8281250	0,546875	0,4062500
0,5468750	1,4843750	1,0937500	1,109375	0,4062500
1,5000000	0,5468750	0,5781250	0,484375	0,7187500
0,5781250	0,5312500	0,7500000	0,484375	0,4218750
0,9687500	1,4531250	0,9687500	0,515625	0,3750000
0,5468750	0,5312500	0,5156250	0,5	0,4375000
0,5156250	0,5000000	0,9531250	0,9375	0,7343750
1,0156250	0,9687500	0,5625000	0,515625	0,4375000
0,5937500	0,5468750	0,8281250	0,46875	0,7656250
1,1875000	0,9531250	1,0781250	0,640625	0,8437500
1,0468750	0,5625000	0,7343750	0,46875	0,4218750

Tabla 6.4 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador medición 61 a la medición 80

Paquete de 16 datos	**Paquete de 16 datos**	**Paquete de 16 datos**	**Paquete de 3 datos**	**Paquete de 6 datos**
SunnyBoy A	**SunnyBoy B**	**SunnyBoy C**	**Over View**	**Estación meteorológica**
Latencia [s]	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**
0,5000000	0,7031250	1,0312500	1,015625	0,3906250
0,6250000	0,9843750	0,5468750	0,5	0,4218750
1,0781250	0,5468750	0,5156250	0,75	1,3125000
0,5937500	0,7656250	0,9218750	0,46875	0,3906250
0,9687500	0,8437500	0,6406250	0,953125	0,4375000
0,5625000	0,5156250	0,5312500	0,46875	0,3906250
0,7500000	1,0312500	1,3906250	0,484375	0,8593750
1,0625000	0,5312500	0,6093750	0,640625	0,4531250
0,7812500	1,1250000	0,5781250	0,46875	0,3906250
1,6250000	1,0937500	1,1718750	1	0,8593750
2,2656250	0,9375000	0,5000000	0,671875	0,4843750
0,9218750	1,0312500	1,0468750	0,453125	0,4375000
2,6562500	0,5468750	0,5000000	0,5	0,4375000
2,8750000	0,5156250	0,5781250	0,515625	0,8593750
1,5937500	0,9218750	1,0312500	1	0,3593750
0,8281250	0,6406250	0,6093750	0,53125	0,3906250
1,0937500	0,5625000	1,4687500	0,515625	0,4062500
0,7187500	1,4843750	0,6718750	0,46875	0,8750000
0,5625000	0,5312500	0,5468750	0,609375	0,4218750
1	0,5312500	1	1,265625	0,7656250

Tabla 6.5 Mediciones de tiempo en el que un tipo de paquete de datos llega al computador medición 81 a la medición 100

Paquete de 16 datos	**Paquete de 16 datos**	**Paquete de 16 datos**	**Paquete de 3 datos**	**Paquete de 6 datos**
SunnyBoy A	**SunnyBoy B**	**SunnyBoy C**	**Over View**	**Estación meteorológica**
Latencia [s]	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**	**Latencia [s]**
0,5625000	1,2656250	0,5312500	0,453125	0,3593750
0,5156250	0,5312500	0,5312500	0,53125	0,8906250
0,9687500	1,0312500	1,0312500	0,46875	0,4218750
0,5781250	0,5781250	0,5156250	1,0625	0,5312500
0,5625000	0,7500000	1,4843750	0,578125	0,9218750
1,4531250	1,0625000	0,5468750	0,5	0,3593750
0,5625000	0,9218750	0,5937500	0,53125	0,4218750
1	1,0625000	1,3437500	0,46875	0,3906250
0,5156250	0,5468750	0,5468750	1,21875	0,9843750
0,8125000	0,5000000	0,5468750	0,5	0,4531250
1,0156250	1	0,9531250	0,53125	0,3750000
0,5468750	0,5468750	0,6406250	0,703125	0,4062500
0,5468750	1,0156250	1,1093750	0,984375	1,3593750
1,2031250	0,7656250	1,2187500	0,609375	0,3593750
0,5156250	0,5156250	1,5781250	0,53125	0,4375000
1,0156250	1,0156250	0,5156250	0,515625	0,7812500
0,8437500	0,7656250	0,5156250	0,9375	0,4062500
0,5625000	2,6250000	1,0625000	0,421875	0,6562500
0,9687500	2,3125000	0,5625000	0,515625	0,3437500
0,5468750	0,9062500	0,8125000	0,5	0,8593750

Tabla 6.6 Datos a partir de las muestras tomadas de las tablas 6.1 a la 6.5.

Tiempos de recepción de diferentes paquetes de datos	Numero de muestras	Promedio de tiempo de la recepción t_R [s]	Desviación estándar	Número de datos por paquete	Taza de datos TD [datos/s]	Tiempo promedio para un dato t_{dato} [ms]	Porcentaje de acierto [%]
Paquete del inversor SunnyBoy A	100	0,87703125	0,439730151	16	18,24336362	54,81445313	100
Paquete del inversor SunnyBoy B	100	0,8303125	0,361021699	16	19,26985322	51,89453125	100
Paquete del inversor SunnyBoy C	100	0,80203125	0,317751449	16	19,94934736	50,12695313	100
Paquete con los datos de mediciones resumen	100	0,63359375	0,218551215	3	4,734895191	72,85040505	100
Paquete con datos de la estación meteorológica	100	0,5740625	0,258883936	4	27,871529 67	35,87890625	100

Se presentan las gráficas generadas por la aplicación Real time WebBox values, estas gráficas se realizaron a partir de mediciones guardadas en el disco duro, el numero de datos por gráfica es de aproximadamente 1000, son datos obtenidos en una hora y 15 minutos aproximadamente. En el eje “y” se ve la hora de la medición

Mediciones climatológicas:

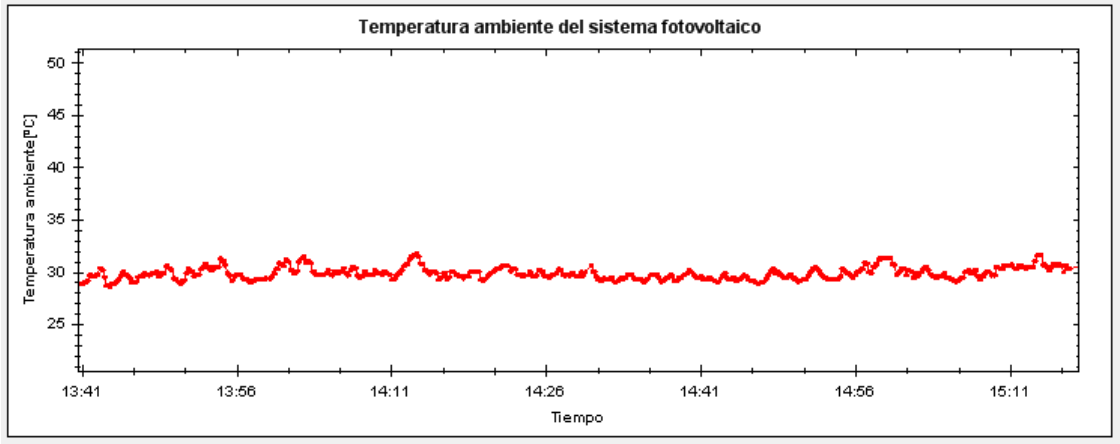


Figura 6.1 Temperatura ambiente en los paneles en función del tiempo

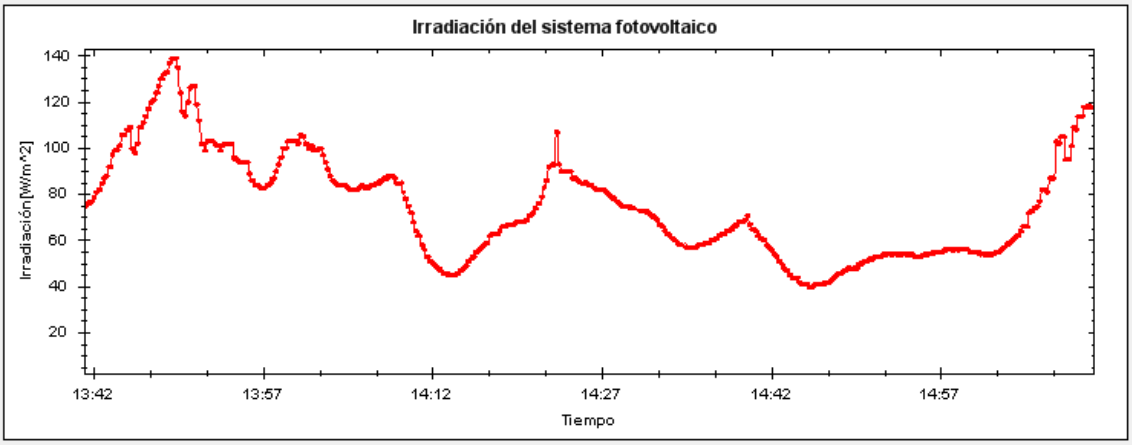


Figura 6.2 Irradiación solar en función del tiempo

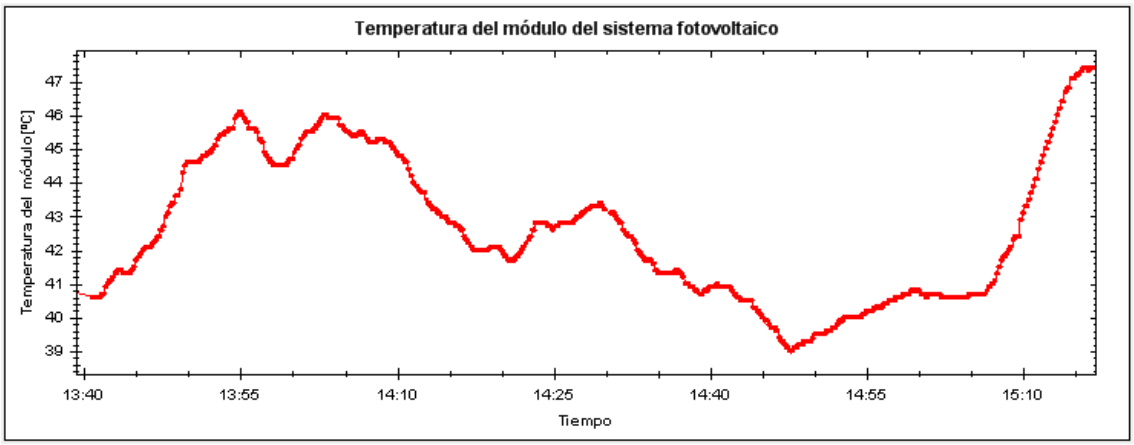


Figura 6.3 Temperatura del módulo Sunny SensorBox en función del tiempo

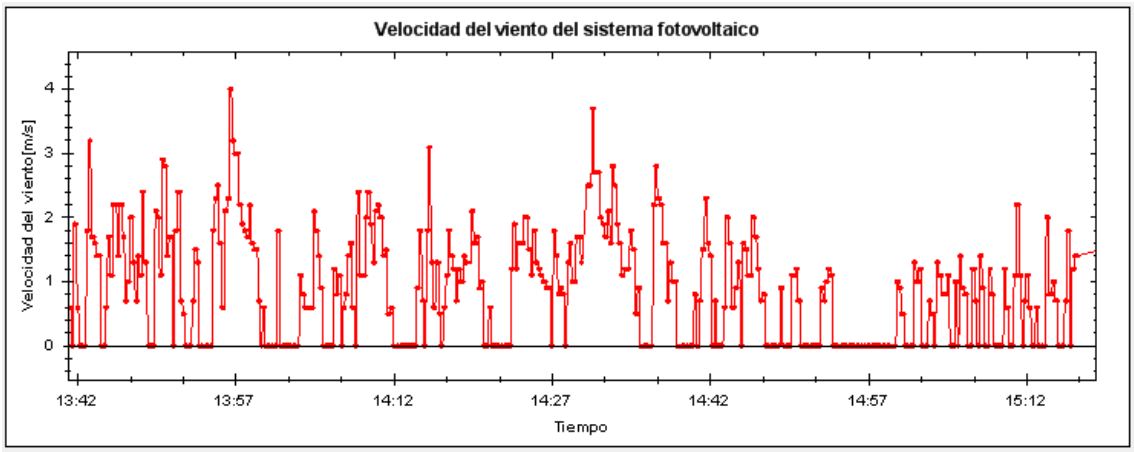


Figura 6.4 Velocidad del viento en función del tiempo

Mediciones correspondientes a las variables resumen:

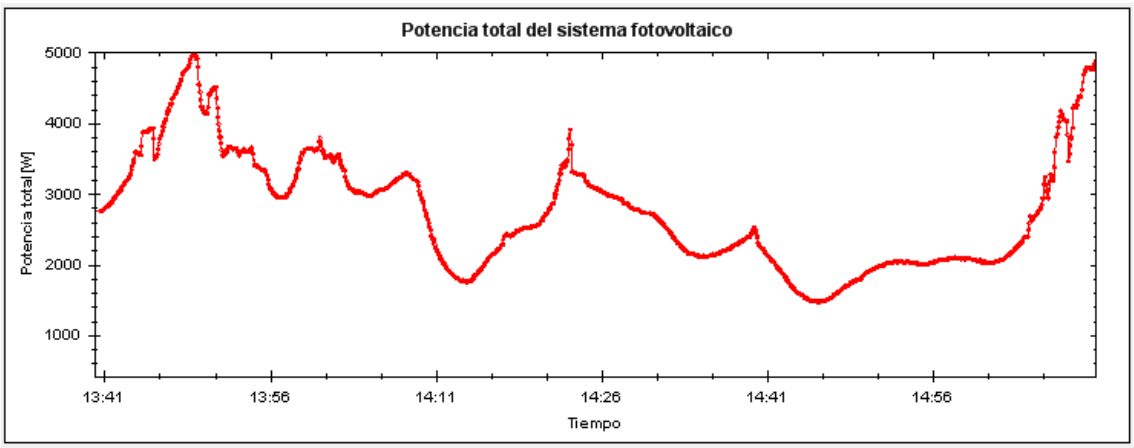


Figura 6.5 Potencia total producida por el sistema fotovoltaico en función del tiempo

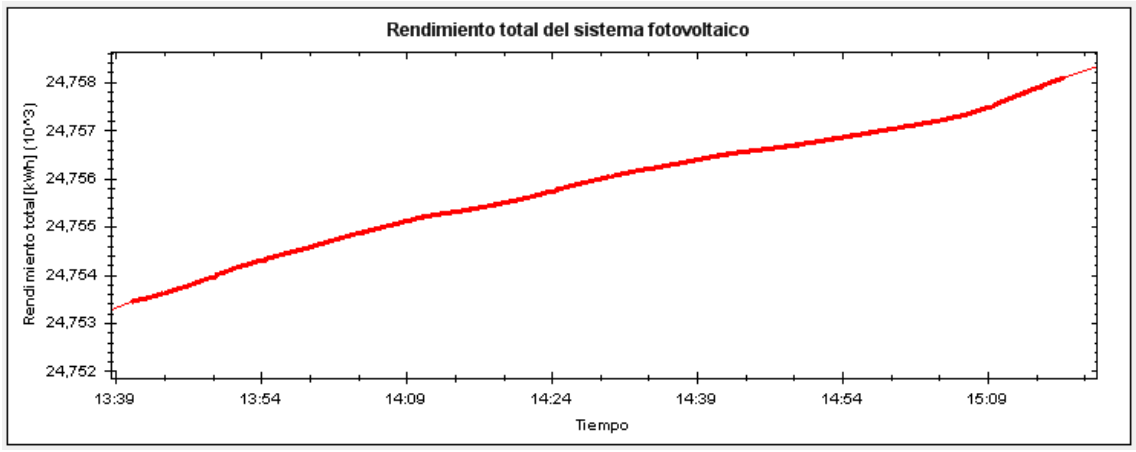


Figura 6.6 Rendimiento del sistema solar en función del tiempo

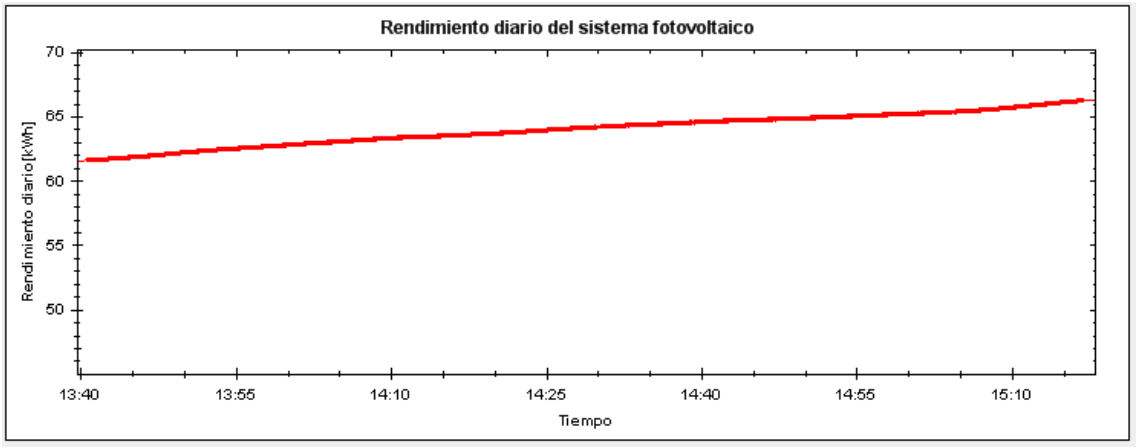


Figura 6.7 Rendimiento del día actual en función del tiempo

Mediciones las variables físicas tomadas del inversor A

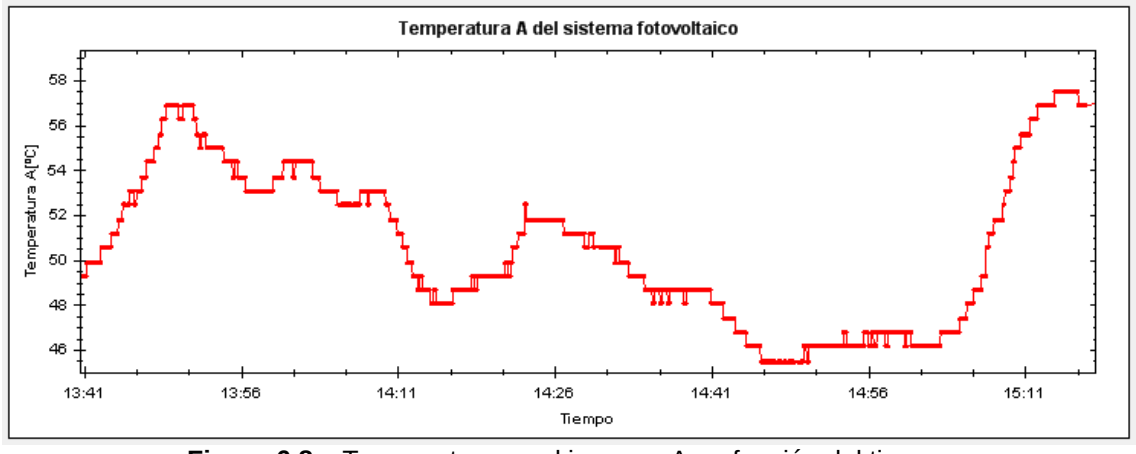


Figura 6.8 Temperatura en el inversor A en función del tiempo

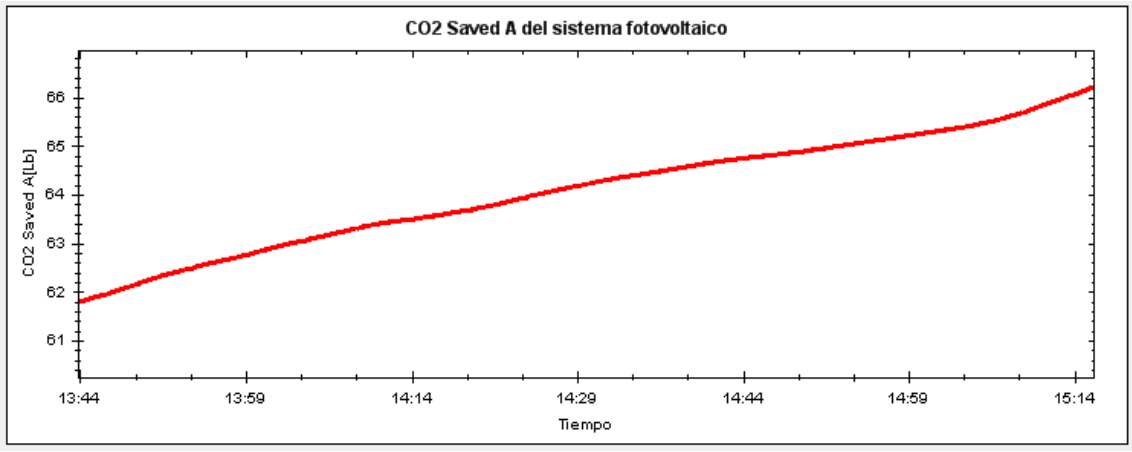


Figura 6.9 CO2 ahorrado en función del tiempo

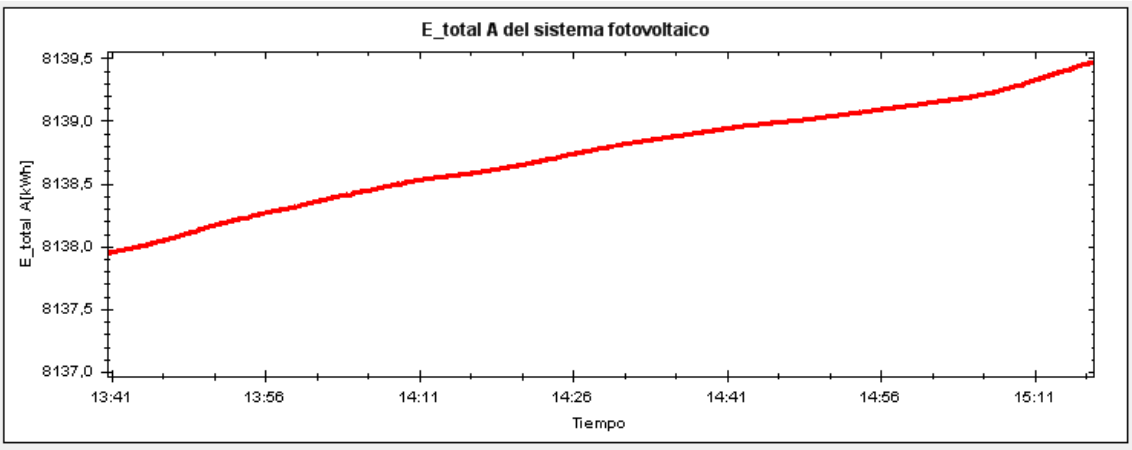


Figura 6.10 Energía producida por el inversor A en función del tiempo

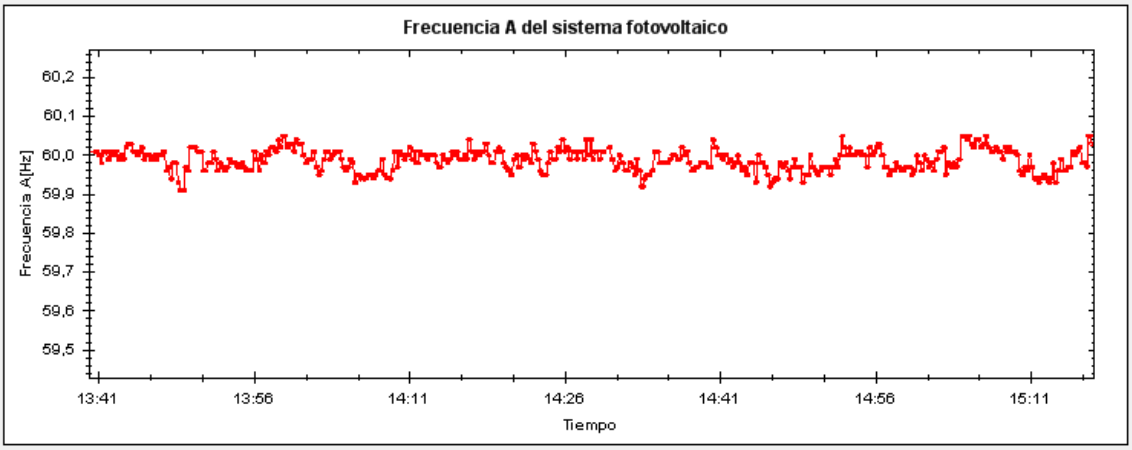


Figura 6.11 Frecuencia de la señal de tensión en la salida del inversor A en función del tiempo

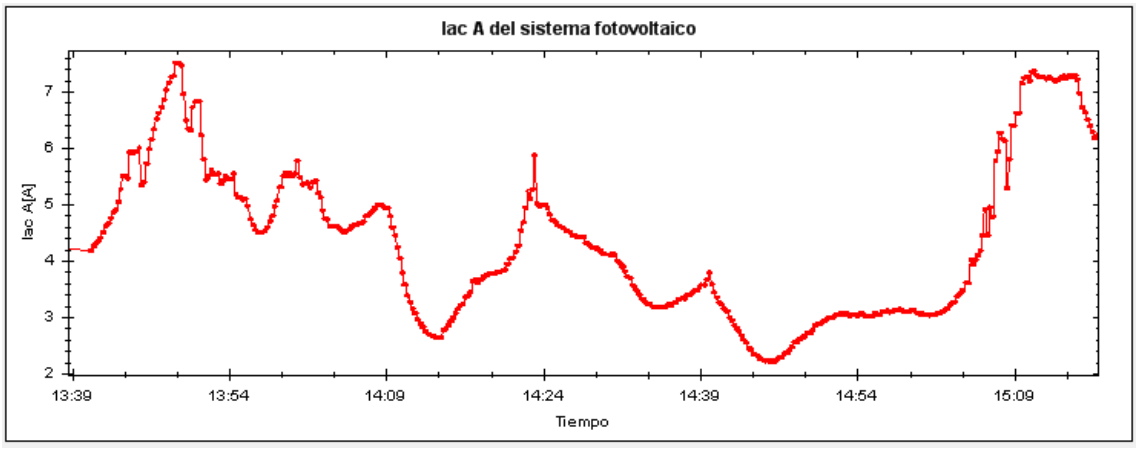


Figura 6.12 Corriente efectiva suplida por el inversor A en función del tiempo

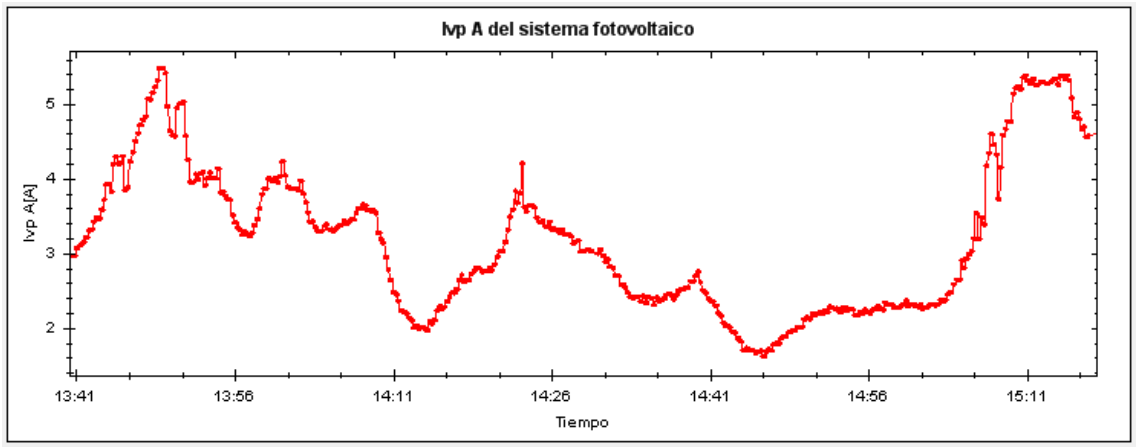


Figura 6.13 Corriente directa generada por un arreglo de paneles solares medida en la entrada del inversor A, en función del tiempo.

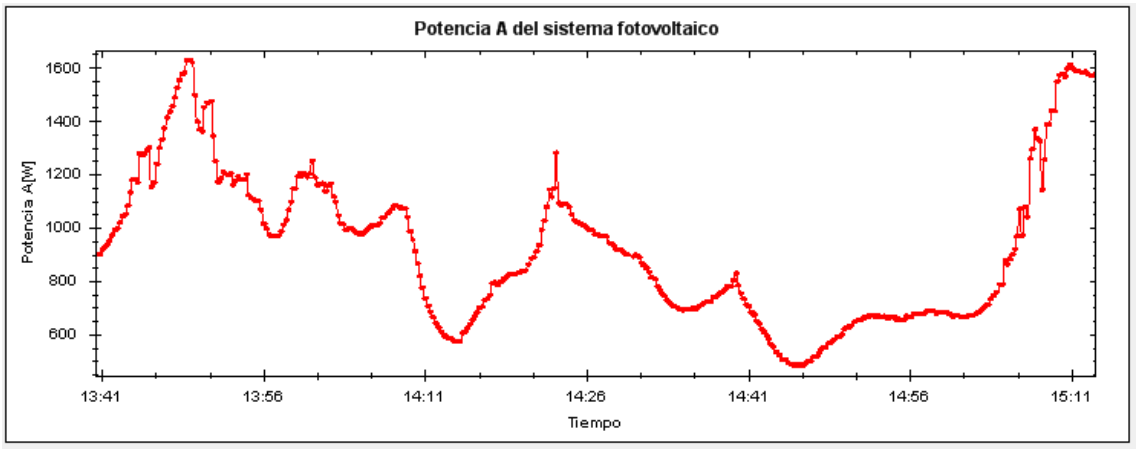


Figura 6.14 Potencia en el inversor A entregada a la red en función del tiempo

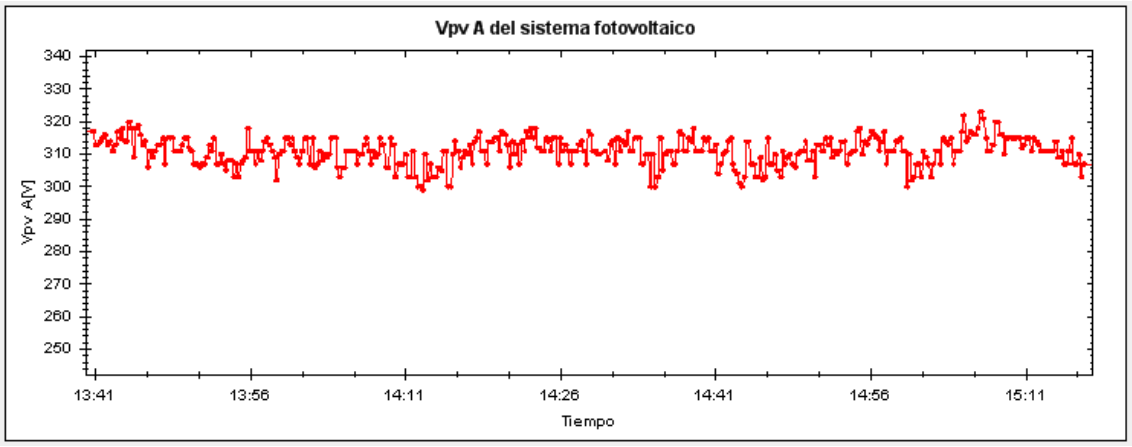


Figura 6.15 Tensión directa generada por un arreglo de paneles, entregada al inversor A, en función del tiempo

Las siguientes mediciones correspondes a tiempo de actualización de la medición presentada en la interfaz gráfica, se tomaron las mediciones de todas las variables. También se muestra el valor de la medición

Tabla 6.7 Valores de mediciones y tiempos de la variable Potencia Total.

Hora de la medición	Valor de la medición [W]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
01:41:02 p.m.	2814		
01:41:06 p.m.	2823	4	Si
01:41:09 p.m.	2833	3	Si
01:41:13 p.m.	2833	4	Si
01:41:16 p.m.	2846	3	Si
01:41:20 p.m.	2853	4	Si
01:41:24 p.m.	2853	4	Si

Tabla 6.8 Mediciones y tiempos de la variable Rendimiento Diario.

Hora de la medición	Valor de la medición [Kw]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
01:41:56 p.m.	61,708		
01:42:00 p.m.	61,713	4	Si
01:42:04 p.m.	61,718	4	Si
01:42:08 p.m.	61,723	4	Si
01:42:11 p.m.	61,723	3	Si
01:42:14 p.m.	61,727	3	Si
01:42:18 p.m.	61,735	4	Si

Tabla 6.9 Mediciones y tiempos de la variable Rendimiento

Hora de la medición	Valor de la medición [W]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
01:42:58 p.m.	24753,577		
01:43:02 p.m.	24753,581	4	Si
01:43:05 p.m.	24753,581	3	Si
01:43:09 p.m.	24753,586	4	Si
01:43:13 p.m.	24753,59	4	Si
01:43:17 p.m.	24753,595	4	Si
01:43:20 p.m.	24753,595;	3	Si

Tabla 6.10 Mediciones y tiempos de la variable Irradiación.

Hora de la medición	Valor de la medición [W/m^2]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
01:43:01 p.m.	87		
01:43:05 p.m.	88	4	Si
01:43:08 p.m.	88	3	Si
01:43:12 p.m.	88	4	Si
01:43:16 p.m.	92	4	Si
01:43:19 p.m.	92	3	Si
01:43:23 p.m.	92	4	Si

Tabla 6.11 Mediciones y tiempos de la variable Temperatura Ambiente

Hora de la medición	Valor de la medición [ºC]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
01:44:00 p.m.	28.93		
01:44:03 p.m.	28.93	3	Si
01:44:06 p.m.	28.93	3	Si
01:44:11 p.m.	29.13	5	Si
01:44:14 p.m.	29.13	3	Si
01:44:17 p.m.	29.13	3	Si
01:44:21 p.m.	29.13	4	Si

Tabla 6.12 Mediciones y tiempos de la variable Temperatura del Módulo

Hora de la medición	Valor de la medición [ºC]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
01:44:57 p.m.	41,73		
01:45:02 p.m.	41,73	5	Si
01:45:05 p.m.	41,83	3	Si
01:45:08 p.m.	41,83	3	Si
01:45:12 p.m.	41,83	4	Si
01:45:16 p.m.	41,93	4	Si
01:45:19 p.m.	41,93	3	Si

Tabla 6.13 Mediciones y tiempos de la variable Velocidad de Viento

Hora de la medición	Valor de la medición [m/s]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:02:00 a.m.	0.9		
10:02:03 a.m.	1.2	3	Si
10:02:07 a.m.	1.2	4	Si
10:02:11 a.m.	1.2	4	Si
10:02:14 a.m.	1.2	3	Si
10:02:18 a.m.	1.2	4	Si
10:02:21 a.m.	1.2	3	Si

Tabla 6.14 Mediciones y tiempos de la variable potencia generada por el inversor A.

Hora de la medición	Valor de la medición [W]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:02:01 a.m.	951		
10:02:04 a.m.	951	3	Si
10:02:08 a.m.	951	4	Si
10:02:11 a.m.	951	3	Si
10:02:15 a.m.	956	4	Si
10:02:19 a.m.	956	4	Si
10:02:22 a.m.	956	3	Si

Tabla 6.15 Mediciones y tiempos de la variable energía diaria del inversor A

Hora de la medición	Valor de la medición [kWh]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:25:01 a.m.	8132.701		
10:25:05 a.m.	8132.705	4	Si
10:25:09 a.m.	8132.705	4	Si
10:25:12 a.m.	8132.705	3	Si
10:25:16 a.m.	8132.705	4	Si
10:25:20 a.m.	8132.709	4	Si
10:25:24 a.m.	8132.709	4	Si

Tabla 6.16 Mediciones y tiempos de la variable CO2 ahorrado por el inversor A.

Hora de la medición	Valor de la medición [Lb]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
08:39:58 a.m.			
08:40:02 a.m.	13822.95	4	Si
08:40:06 a.m.	13822.954	4	Si
08:40:09 a.m.	13822.954	3	Si
08:40:13 a.m.	13822.954	4	Si
08:40:18 a.m.	13822.954	5	Si
08:40:21 a.m.	13822.954	3	Si

Tabla 6.17 Mediciones y tiempos de la variable frecuencia eléctrica del inversor A.

Hora de la medición	Valor de la medición [Hz]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:02:01 a.m.	60.01		
10:02:04 a.m.	60.01	3	Si
10:02:08 a.m.	59.99	4	Si
10:02:11a.m.	59.99	3	Si
10:02:15 a.m.	59.99	4	Si
10:02:19 a.m.	59.99	4	Si
10:02:22 a.m.	59.99	3	Si

Tabla 6.18 Mediciones y tiempos de la variable corriente alterna en la salida del inversor A.

Hora de la medición	Valor de la medición [A]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:02:01 a.m.	4.376		
10:02:04 a.m.	4.376	3	Si
10:02:08 a.m.	4.376	4	Si
10:02:11 a.m.	4.399	3	Si
10:02:15 a.m.	4.399	4	Si
10:02:19 a.m.	4.399	4	Si
10:02:22 a.m.	4.399	3	Si

Tabla 6.19 Mediciones y tiempos de la variable corriente directa en la entrada del inversor A

Hora de la medición	Valor de la medición [A]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:02:01 a.m.	3.309		
10:02:04a.m.	3.309	3	Si
10:02:08 a.m.	3.309	4	Si
10:02:11 a.m.	3.309	3	Si
10:02:15 a.m.	3.231	4	Si
10:02:19 a.m.	3.231	4	Si
10:02:22 a.m.	3.231	3	Si

Tabla 6.20 Mediciones y tiempos de la variable tensión alterna en la salida del inversor A

Hora de la medición	Valor de la medición [V]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:25:01 a.m.	215.4		
10:25:05 a.m.	215.6	4	Si
10:25:09 a.m.	215.6	4	Si
10:25:12 a.m.	215.6	3	Si
10:25:16 a.m.	215.6	4	Si
10:25:20 a.m.	215.6	4	Si
10:25:24 a.m.	215.6	4	Si

Tabla 6.21 Mediciones y tiempos de la variable tensión continua en la entrada del inversor A.

Hora de la medición	Valor de la medición [V]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:26:00 a.m.	308;26		
10:26:04 a.m.	308	4	Si
10:26:08a.m.	306	4	Si
10:26:11 a.m.	306	3	Si
10:26:15a.m.	306	4	Si
10:26:19 a.m.	305	4	Si
10:26:23 a.m.	305	4	Si

Tabla 6.22 Mediciones y tiempos de la temperatura del inversor A

Hora de la medición	Valor de la medición [ºC]	Tiempo en actualizarse [s]	¿Tiempo en actualizarse menor a 7s?
10:26:00 a.m.	48.7		
10:26:04 a.m.	48.7	4	Si
10:26:08a.m.	48.7	4	Si
10:26:11 a.m.	48.7	3	Si
10:26:15 a.m.	48.7	4	Si
10:26:19 a.m.	48.7	4	Si
10:26:23 a.m.	48.7	3	Si

6.2 Análisis

Uno de los objetivos era lograr una transferencia de datos rápida, para ello se propuso que un dato debería tardar no más de un segundo desde el momento que se hizo la petición.

En las tablas de la 6.1 a la 6.5 se muestra el tiempo que demora un paquete de datos en llegar al computador. En las tablas se aprecia que se cuanta con 5 paquetes de datos o 5 grupos de datos los cuales se detallaron en la tabla 5.2.
En las tablas se enlistan 1000 mediciones de latencia de cada paquete y en la tabla 6.6 se muestran las latencias promedio de cada paquete, también se muestran cálculos como taza de transferencia de datos y tiempo promedio por dato

Para cada paquete de datos se calcularon latencias promedios y desviaciones estándar seguidamente a partir de estos resultados se procedió a calcular el número de datos que se manda por segundo “TD” (tasa de datos) para esto se utilizó la ecuación 6.1

$$TD = \frac{Numero\ de\ datos}{latencia\ promedio} \qquad 6.1$$

Se determinó otro parámetro el cual consiste en el inverso de TD, este parámetro nos da un estimado del tiempo que dura un dato en llegar a la computadora, dicho parámetro se calcula según la ecuación 2.

$$t_{dato} = \frac{1}{TD} \qquad 6.2$$

Los datos calculados se presentan en la tabla 6.6; de esta, se analizan los resultados de los tiempos promedio t_{dato}. Este tiempo es la duración de un dato en

llegar a la computadora desde su petición. De estos datos se ve que el mayor tiempo promedio es de 73 milisegundos y en el indicador se propuso que esta latencia debería ser no mayor a un segundo además se nota que el acierto fue de un 100 %, por tanto el objetivo de diseñar un módulo Manejador de Protocolo y Tráfico de Datos se cumple. Se dice entonces que en la interfaz gráfica se muestran las mediciones climatológicas y eléctricas del sistema fotovoltaico de forma eficiente.

De los archivos txt guardados en el disco duro se tomaron más de 1000 mediciones para generar las gráficas de cada variable. Estas gráficas corresponden a las figuras 6.1 a la 6.15.

De estas gráficas se puede suponer que no se presentan datos erróneos ya que cada una de las mediciones muestran valores coherentes, si una medición fuera errónea se notaría en la gráfica por medio de un dato pico que se sale del rango normal. Si se observan las gráficas se puede observar que las gráficas de irradiación, corriente y potencia presentan un comportamiento muy similar; esto tiene sentido ya que estas variables tienen relaciones proporcionales; esto da una pista de que las mediciones son correctas.

Una gráfica que presenta valores muy aleatorios es la gráfica de velocidad de viento, esto se ve en la figura 6.4, cosa que es de esperar ya que esta variable suele tener este comportamiento. Caso contrario es la temperatura que en teoría no tendría que presentar cambios bruscos. Según las gráficas obtenidas (figuras 6.1, 6.3 y 6.8 correspondientes a temperaturas), este comportamiento se comprueba por lo que se puede decir que los datos son coherentes.

Con el análisis anterior se podría decir que el acierto de las mediciones guardadas en el archivo txt tienen un acierto mayor al 90 % por lo que se puede afirma que el módulo “Administrador de la base de datos” realiza lo requerido. También se puede decir que las mediciones quedan registradas de forma correcta.

El módulo “HMI” tiene como requerimiento que se puedan ver todas las mediciones de las variables de forma clara y ademas el tiempo de actualizacion debe ser menor a 7 segundos. Para esto se midieron y anotaron tiempos de actualizacion de todas las variables, en un periodo de 20 segundos

Las medidas de tiempo se lograron, implentando una rutina dentro de la aplicación tal rutina toma la hora donde hubo un cambio de medición, entonces para calcular el tiempo se hace una resta de la hora final menos la hora inicial, las mediciones se presentan en las tablas de la 6.7 a la 6.22; además se lee el valor de la medicion y se comprueba se este es correcto. Hablamos de que un dato es correcto si este tiene un valor coerente conrespecto a mediciones anteriores.

Todas las mediciones anotadas corresponden a mediciones realizadas el 26 de julio del 2011 y las mediciones se realizaron en un intervalo de 20 minutos. En la columna derecha se indica si el tiempo de actualizacion fue menor a 7 segundos.

Al analizar los resultados se nota que el tiempo de actualización que predomina es el de 3 y 4 segundos y en casos esporaticos aparecen tiempos de 5 segundos,que es el mayor tiempo medido, se nota que aun así se cumple el objetivo que todo los tiempo fueron menores a 7 segundos. Es por ello que se dice que el módulo HMI fue implemetadode forma correcta

Capítulo 7: Conclusiones y recomendaciones

7.1 Conclusiones

a) El tiempo promedio de transferencia de un dato es de 73 milisegundos y puesto que es menor a un segundo, se cumple que el módulo manejador de Protocolo y Tráfico de Datos funciona correctamente.

b) Se graficaron mediciones guardadas en el disco duro y las mediciones son coherentes. Se dice entonces que el módulo Administrador de la base de datos se implemento de forma requerida.

c) El tiempo de actualización de las mediciones en la interfaz gráfica es menor a 7 segundos; parámetro que valida el cumplimiento de que el módulo de software HMI.

d) Los objetivos específicos se cumplen por eso se afirma que los funcionarios del Centro de Producción ICE Barranca pueden ver todas las mediciones presentes y pasadas sin ningún tipo de retraso.

e) Con el nuevo sistema implementado se puede analizar las variables físicas medidas

f) .NET proporciona una amplia gama de bibliotecas que facilitan la programación en C#.

g) El desarrollo de clases es de gran utilidad cuando se tiene que implementar funciones semejantes varias veces.

7.2 Recomendaciones

a) Cuando la aplicación se esté corriendo no es recomendado cerrarla ya que esta interrumpiría la recepción de datos.

b) La aplicación implementada “Real Time WebBox Values” funciona exclusivamente para los dispositivos del sistema fotovoltaico, si se adiciona otro dispositivo, por ejemplo otro inversor, la aplicación no reconocerá este dispositivo, por lo que se deberá modificar el código de la aplicación.

c) El ICE cuenta con diversos sistemas fotovoltaicos en diversas partes del país y estos no cuentan con la aplicación ““Real Time WebBox Values” por lo que aconseja modificar la aplicación para que se pueda comunicar con todos los sistemas fotovoltaicos del ICE y así tener un monitoreo mas centralizado.

8. Bibliografía

[1] TANENBAUM, Andrew S. *Redes de Computadoras.* Pearson Educación. 2003. pp. 912.

[2] FERGUSON, Jeff. *La Biblia de C#.* Ediciones Anaya Multimedia. 2003.

[3] Microsoft. 2011. MSDN Library. (Sitio Web). Consultado 07 set. 2011. Disponible en http://msdn.microsoft.com/en-us/library/yyaad03b(v=vs.71).aspx

[4] SMA Solar Technology. 2011. (Sitio Web). Consultado 05 jun. 2011. Disponible en http://www.sma-america.com/en_US/products/software/sma-opc-server.html.

[5] SMA Solar Technology. 2011. SUNNY WEBBOX RPC User Manual. Remote Procedure Call Description Interface a dAPI Definition (en línea). Consultado 05 jun. 2011. Disponible en http://www.sma-america.com/en_US/products/monitoring-systems/sunny-webbox.html

ANEXO 1

HOJA DE INFORMACIÓN

Información del estudiante:

Nombre: Pablo César Sulecio Varela

Cédula o No. 1-1162-0352

Carné ITCR: 200123911

Dirección de su residencia en época lectiva: 350 m norte de Palí Tres Rios Cartago

Dirección de su residencia en época no lectiva: 350 m norte de Palí Tres Rios Cartago

Teléfono en época lectiva: 2279-0985

Teléfono época no lectiva: 2279-0985

Email: pablo.sulecio@gmail.com

Información del Proyecto:

Nombre del Proyecto: Comunicación e interfaz gráfica para visualizar y almacenar mediciones de variables físicas de un sistema de paneles solares en el Centro de Producción ICE Barranca

Profesor Asesor: Francisco Navarro Henríquez

Horario de trabajo del estudiante: Lunes a Viernes de 7am a 5pm

Información de la Empresa:

Nombre: Centro de Producción ICE Barranca

Zona: Puntarenas, Cantón Central, Barranca, Barrio Santa Lucia.

Dirección: 250 m Norte de Arrocera el Porvenir

Teléfono: **:** 2663-0147

Fax: 2663-1878

Actividad Principal: Producción de electricidad

ANEXOS 2

Sunny WebBox RPC

Remote Procedure Call Description

Interface and API Definition