Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

DISEÑO MODULAR DE MONITOR DE TEMPERATURA CON CONEXIÓN 3G BASADO EN RASPBERRY PI MODELO B PARA ASIGNACIÓN DE LA CALIDAD DE PRODUCTOS PERECEDEROS, SINCRONIZADO ONLINE

Alex Antonio Díaz Mazariegos
Asesorado por el Ing. Iván René Morales Argueta

Guatemala, octubre de 2014

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

DISEÑO MODULAR DE MONITOR DE TEMPERATURA CON CONEXIÓN 3G BASADO EN RASPBERRY PI MODELO B PARA ASIGNACIÓN DE LA CALIDAD DE PRODUCTOS PERECEDEROS, SINCRONIZADO ONLINE

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

ALEX ANTONIO DÍAZ MAZARIEGOS
ASESORADO POR EL ING. IVÁN RENÉ MORALES ARGUETA

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN ELECTRÓNICA

GUATEMALA, OCTUBRE DE 2014

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Murphy Olympo Paiz Recinos
VOCAL I	Ing. Alfredo Enrique Beber Aceituno
VOCAL II	Ing. Pedro Antonio Aguilar Polanco
VOCAL III	Inga. Elvia Miriam Ruballos Samayoa
VOCAL IV	Br. Narda Lucía Pacay Barrientos
VOCAL V	Br. Walter Rafael Véliz Muñoz
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Murphy Olympo Paiz Recinos
EXAMINADOR	Ing. Otto Fernando Andrino González
EXAMINADOR	Ing. Byron Odilio Arrivillaga Méndez
EXAMINADOR	Ing. José Aníbal Silva de los Ángeles
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

DISEÑO MODULAR DE MONITOR DE TEMPERATURA CON CONEXIÓN 3G BASADO EN RASPBERRY PI MODELO B PARA ASIGNACIÓN DE LA CALIDAD DE PRODUCTOS PERECEDEROS, SINCRONIZADO ONLINE

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica, con fecha de 17 julio de 2014.

Alex Antonio Díaz Mazariegos

Guatemala, 17 de septiembre de 2014

Ing. Carlos Guzmán
Coordinador Área de Electrónica
Escuela de Mecánica Eléctrica
Facultad de Ingeniería

Estimado Ingeniero Guzmán,

Por este medio me dirijo a usted para comunicarle que he asesorado y revisado el trabajo de graduación titulado: " **DISEÑO MODULAR DE MONITOR DE TEMPERATURA CON CONEXIÓN 3G BASADO EN RASPBERRY PI MODELO B PARA ASIGNACIÓN DE LA CALIDAD DE PRODUCTOS PERECEDEROS, SINCRONIZADO ONLINE**", desarrollado por el estudiante **Alex Antonio Díaz Mazariegos**, quien se identifica con número de carnet **200915026**. Luego de comprobar su contenido final, considero que cumple con los requerimientos necesarios, y lo apruebo como trabajo de graduación.

Agradeciendo su amable colaboración, me suscribo atentamente,

Iván René Morales Argueta
Ingeniero Electrónico
Colegiado 12489

Ing. Iván René Morales Argueta
Colegiado 12489

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

Ref. EIME 48. 2014
Guatemala, 22 de SEPTIEMBRE 2014.

Señor Director
Ing. Guillermo Antonio Puente Romero
Escuela de Ingenierìa Mecànica Elèctrica
Facultad de Ingenierìa, USAC.

Señor Director:

Me permito dar aprobación al trabajo de Graduación titulado: DISEÑO MODULAR DE MONITOR DE TEMPERATURA CON CONEXIÓN 3G BASADO EN RASPBERRY PI MODELO B PARA ASIGNACIÓN DE LA CALIDAD DE PRODUCTOS PERECEDEROS, SINCRONIZADO ONLINE, del estudiante Alex Antonio Díaz Mazariegos , que cumple con los requisitos establecidos para tal fin.

Sin otro particular, aprovecho la oportunidad para saludarle.

Atentamente,
ID Y ENSEÑAD A TODOS

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

Ing. Carlos Eduardo Guzmán Salazar
Coordinador Área Electrónica

sro

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

REF. EIME 48. 2014.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto Bueno del Coordinador de Área, al trabajo de Graduación del estudiante; ALEX ANTONIO DÍAZ MEZARIEGOS titulado: DISEÑO MODULAR DE MONITOR DE TEMPERATURA CON CONEXIÓN 3G BASADO EN RASPBERRY PI MODELO B PARA ASIGNACIÓN DE LA CALIDAD DE PRODUCTOS PERECEDEROS, SINCRONIZADO ONLINE, procede a la autorización del mismo.

Ing. Guillermo Antonio Puente Romero

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

GUATEMALA, 1 DE OCTUBRE 2,014.

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

DTG. 582.2014

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Mecánica Eléctrica, al Trabajo de Graduación titulado: **DISEÑO MODULAR DE MONITOR DE TEMPERATURA CON CONEXIÓN 3G BASADO EN RASPBERRY PI MODELO B PARA ASIGNACIÓN DE LA CALIDAD DE PRODUCTOS PERECEDEROS, SINCRONIZADO ONLINE,** presentado por el estudiante universitario **Alex Antonio Díaz Mazariegos,** y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, se autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Murphy Olympo Paiz Recinos
Decano

Guatemala, 24 de octubre de 2014

/gdech

ACTO QUE DEDICO A:

Dios	Por la vida y las oportunidades que me ha dado para llegar a cumplir esta meta.
Mi padre	Jorge Vidal Díaz Alvarado, por el apoyo incondicional, exigencias para la excelencia profesional y consejos que me infunde para actuar con el bien y la humildad.
Mi madre	Hilma Natividad Mazariegos Pérez, por ser la mayor fuente de inspiración para alcanzar mis metas.
Mis hermanos	Alejandra, Mario y en especial a Jorge Luis Díaz, por su apoyo, confianza y ayuda para culminar mis metas con éxito y ser un ejemplo a seguir profesionalmente.
Mi familia	Por estar siempre apoyando y alentando a terminar mis estudios profesionales.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por darme la oportunidad de ingresar a tan prestigiosa institución y brindarme los valores profesionales necesarios para afrontar la vida.
Facultad de Ingeniería	Por acogerme en sus aulas para recibir una educación digna y profesional, para ser competente en los ambientes laborales.
Escuela de Ingeniería Mecánica Eléctrica	Por velar que la enseñanza sea de alta calidad para sobresalir profesionalmente, en especial a los catedráticos.
Ingeniero electrónico Iván Morales	Por compartir sus conocimientos sin egoísmo, por servir de apoyo incondicional en los proyectos y actividades estudiantiles.
Mis amigos y compañeros	Por las experiencias vividas, conocimientos compartidos, los momentos difíciles para salir adelante juntos y sobre todo la amistad incondicional.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
A	Amperio
DB	Base de datos
C	Centímetro
AC	Corriente alterna
DC	Corriente directa
ºC	Grados Celsius
ºF	Grados Fahrenheit
Hz	Hertz
®	Marca registrada
mb	Megabits
MB	Megabytes
mA	Miliamperio
mm	Milímetro
Ω	Ohm
Rx	Receptor
R	Resistencia
GND	Tierra o voltaje de referencia
Tx	Transmisor
V	Volt
VDC	Voltios de corriente directa
Vdd	Voltaje de alimentación positivo
Q	Quetzales (moneda guatemalteca)

GLOSARIO

3G Abreviación de Tercera Generación de transmitir datos y voz a través de la telefonía móvil.

Alambre PTFE Cable de altas especificaciones para condiciones ambientales deplorables construido de un polímero similar al polietileno.

Baudio Es una unidad de velocidad de transmisión de datos utilizada en comunicaciones que denota la cantidad de símbolos transmitidos en un intervalo de tiempo, un baudio puede contener varios bits.

Bit Es la unidad mínima de información que se puede almacenar en memoria o ser transmitida y únicamente puede tener dos estados lógicos, cero o uno.

Binding Es una adaptación de una biblioteca o librería para ser usada en un lenguaje de programación distinto de aquél en el que ha sido escrita sirviendo de acople entre los lenguajes de programación.

CPU	Siglas en inglés de *Central Processing Unit* (significa Unidad Central de Procesamiento) es el componente principal de los dispositivos programables donde se procesan datos y se interpretan las instrucciones contenidas en los programas.
DBMS	Siglas en inglés de *Data Base Managemet System* (significa Sistema Gestor de Bases de Datos) son varios programas que permiten almacenar y manipular datos de forma eficaz y estructurada.
DDL	Siglas en inglés de *Data Definition Language* (significa Lenguaje de Definición de Datos) es un lenguaje asequible artificial para definir y describir los objetos de una base de datos, estructura, relaciones y restricciones, puede consistir en un subconjunto de instrucciones de otro lenguaje informático.
DML	Siglas en inglés de *Data Manipulation Language* (significa Lenguaje de Manipulación de Datos) es un lenguaje artificial con cierto grado de complejidad que permite el manejo y procesamiento del contenido en una base de datos, puede consistir en un subconjunto de instrucciones de otro lenguaje informático.

GPU	Siglas en inglés de *Graphics Processing Unit* (significa Unidad de Procesamiento Gráfico) es un coprocesador dedicado únicamente al procesamiento de gráficos u operaciones de punto flotante, permitiendo así el máximo desempeño en otras funciones al procesador central.
IDE	Siglas en inglés de *Integrated Development Enviroment* (significa Entorno de Desarrollo Integrado) es un software compuesto por un conjunto de herramientas de programación empaquetadas en un programa de aplicación, contiene un editor de código, un compilador, un depurador y un editor de interfaz gráfica.
LED	Siglas en inglés de *Light Emitting Diode* (significa Diodo Emisor de Luz) es un dispositivo semiconductor capaz de emitir una longitud de onda visible dependiendo del dopado del mismo al ser polarizado directamente.
Middleware	Software que asiste a una aplicación para interactuar con otras aplicaciones, software, redes, simplificando el trabajo en lo complejo que es generar las conexiones necesarias en los sistemas distribuidos. De esta forma se provee una solución que mejora la calidad de servicio, seguridad, envío de datos o información.

Python	Es un lenguaje de programación interpretado con énfasis en la sintaxis que favorezca un código legible, soporta orientación a objetos, programación imperativa y programación funcional, además posee una licencia de código abierto.
Perecedero	Poco durable, capacidad de conservar sus propiedades solo durante un determinado intervalo de tiempo.
RAM	Siglas en inglés de *Random Access Memory* (significa Memoria de Acceso Aleatorio) es la memoria donde se cargan las instrucciones que ejecuta el procesador o cualquier unidad de cómputo, en el tiempo real.
Resistencia *pull-up*	Resistencia de polarización conectada a un voltaje positivo para elevar la tensión de entrada para un dispositivo digital.
ROM	Siglas en inglés de *Read Only Memory* (significa Memoria de Solo Lectura) es un medio de almacenamiento en dispositivos electrónicos donde se permite sólo la lectura de información.

Script	Es un programa que puede acompañar a un documento HTML o incrustarse directamente en él o bien un archivo de órdenes para configuraciones o instrucciones en otros programas y se almacena en un documento de texto plano.
Sensor	Dispositivo capaz de transformar un determinado tipo de energía de entrada a una señal eléctrica de salida.
Servicio web	Es una tecnología o sistema de software desarrollado para permitir la interacción entre máquinas o aplicaciones a través de una red de internet utilizando protocolos y estándares de comunicación**.**
Sobrecorriente	corriente máxima que un determinado dispositivo puede soportar durante un intervalo de tiempo y es mayor a la corriente nominal o corriente de funcionamiento.
Tarjeta SD	Dispositivo de almacenamiento masivo. Tipo de tarjeta de memoria desarrollada para cumplir con los requisitos de seguridad de información en los dispositivos electrónicos.
Tarjeta SIM	Es una tarjeta inteligente que almacena la clave de servicio para identificarse en la red utilizada en dispositivos móviles.

Telefonía móvil Sistema de comunicación conocido comúnmente como telefonía celular que consta de una red de comunicaciones o red celular y las terminales o teléfonos móviles que tienen el acceso a la red mencionada.

Termocopla Sensor formado por la unión de dos metales distintos que provee una diferencia de potencial en función de la temperatura a la que se somete.

UART Siglas en inglés de *Universal Asynchronous Reciever-Transmitter* (significa Recepción-Transmisión Asíncrona Universal) encargado de controlar los dispositivos y puertos de comunicación serial.

UMTS Siglas en inglés de *Universal Mobile Telecommunications System* (significa Sistema Universal de Comunicaciones Móviles) es uno de los estándares utilizados por la tecnología 3G.

URL Siglas en inglés de *Uniform Resource Locator* (significa Localizador de Recursos Uniforme) es un conjunto de caracteres que permite la asignación de una dirección única y exclusiva a un recurso que se encuentre en internet.

USB	Siglas en inglés de *Universal Serial Bus* (significa Bus Universal Serial) está definido como un estándar industrial para especificar los cables, conectores y protocolos utilizados en un bus para comunicar y proveer alimentación a otros dispositivos electrónicos.
WCDMA	Siglas en inglés de *Wideband Code Division Multiple Access* (significa Acceso Múltiple por División de Código de Banda Ancha) tecnología que sirve como base para varios estándares de comunicación en 3G.

RESUMEN

En la industria de productos perecederos es importante y fundamental mantener control sobre las variaciones de temperatura en las que se ven expuestos durante la distribución, ya sea que necesiten magnitudes de refrigeración o congelación. Estos valores son significativos para una asignación de calidad y punto de referencia para el consumo, así como también son parámetros que conforman un precio dentro de la evaluación de costos operativos y de mantenimiento para una organización.

En el primer capítulo se presenta la trascendencia del control de la temperatura en productos perecederos para la transportación y comercialización. Se expone una definición de la categoría de estos, los tipos de contenedores, y el concepto de cadena en frío que es el fundamento de un sistema conservación estable y controlado, necesario para garantizar la calidad de un producto.

La tecnología de información y comunicación que actualmente está presente en el medio permite la optimización de procesos industriales que son puntos de referencia dentro de la cadena de valor de una organización, sumado a esto, la creciente demanda en la construcción de dispositivos de hardware con la finalidad de la automatización de procesos.

En el segundo capítulo se presentan las bases del diseño para un monitor de temperatura funcional como sistema portátil.

La interacción de la información en tiempo real es un fenómeno de comunicación que en la actualidad es indispensable para cualquier técnica de control y monitoreo, esto es posible gracias a la disponibilidad de los avances en las telecomunicaciones y la convergencia de tecnologías basadas en estándares.

Por lo tanto, en el tercer capítulo se presenta un diseño integral basado en módulos con fines específicos que dan origen a un sistema embebido de componentes interconectados, que optimiza el proceso de control y evaluación de temperatura dentro de un medio en el cual son transportados productos perecederos.

Finalmente se genera una integración de diferentes tecnologías de hardware y software que dan origen a un sistema funcional que constituye un apoyo eficiente en el control de temperatura, el cual envía toda la información recolectada en el lugar de almacenamiento móvil hacia un servidor central que clasifica y evalúa los parámetros recibidos para posterior asignación de valores de calidad sobre productos monitoreados.

En el capítulo cuatro se detalla la interconectividad de los objetos diseñados y se presentan las tecnologías, protocolos, herramientas de software y base de datos para sustentar el eficiente manejo del flujo de información intercambiada.

OBJETIVOS

General

Diseñar un monitor de temperatura con conexión 3G basado en Raspberry Pi modelo B para asignación de la calidad de productos perecederos, sincronizado *online*.

Específicos

1. Exponer la importancia del control de la temperatura en productos perecederos para la transportación.

2. Presentar las bases del diseño para el sistema de monitoreo de temperatura.

3. Explicar las conexiones del hardware del sistema de monitoreo de temperatura.

4. Diseñar el software del sistema de monitoreo de temperatura con la directriz de estructura servidor-cliente.

INTRODUCCIÓN

El mundo actual se encuentra en la era de las comunicaciones donde la tecnología puede ser utilizada para optimizar procesos, tal es el caso de los monitores de temperatura que apoyan la gestión de control de la calidad monitorizando parámetros transcendentales como las señales térmicas de un entorno determinado.

Las comunicaciones en tiempo real utilizando altas velocidades de transferencia de datos por la red celular 3G de un sistema de monitoreo, son una plataforma de apoyo cuando se presentan escenarios donde se necesita obtener parámetros que proyecten una referencia hacia indicadores estándares. Un ejemplo práctico de estos escenarios es representado por la logística de transporte de productos perecederos, en los cuales el correcto control de temperatura define la calidad al producto como tal.

Actualmente en la región centroamericana hay una serie de empresas que dependen del transporte de productos perecederos en medios terrestres y marítimos, en donde las variaciones de temperatura influyen directamente en la calidad de los productos, esto viene dado por la serie de factores que pueden ocurrir en la ruta de transporte del sitio origen al sitio destino. De acuerdo al promedio internacionalmente aceptado los productos deben conservarse a temperaturas definidas para que se mantengan en condiciones idóneas y calidad específica.

Apoyándose con la tecnología y versatilidad de la Raspberry Pi se detalla el diseño de un sistema de monitoreo de los parámetros de temperatura, con la

finalidad de sincronizar la información con un servidor central que recabe los datos generados en el trayecto del transporte del producto y asignar un nivel de calidad al producto final entregado.

Se diseñará un sistema de monitoreo de temperatura que envíe la información de los parámetros obtenidos a través de la red celular hacia un servidor central remoto, en donde se procesarán los datos y se asignará un indicador de calidad del producto en función de las variantes obtenidas, comparando los resultados con las recomendaciones internacionalmente aceptadas para la conservación de productos perecederos.

1. IMPORTANCIA DEL CONTROL DE LA TEMPERATURA EN PRODUCTOS PERECEDEROS PARA LA TRANSPORTACIÓN

En la actualidad ha tomado una particular prioridad la calidad del transporte alimenticio, gracias a él se puede aprovechar durante todo el año de los alimentos que se demandan. Estos pueden ser elaborados en cualquier parte del mundo y a pesar de ello, llegan en insuperables condiciones a los puntos de venta o consumo como los mercados o distribuidoras. Las empresas que manipulan productos perecederos o semiperecederos, ven necesario conocer el indicador de calidad de los mismos, con el cual pueden organizar la logística de distribución y promover como valor agregado una categorización en función del nivel de los atributos que presente un producto determinado.

1.1. Productos perecederos

Los productos sensibles a la temperatura y los productos perecederos, son aquellos que tienden a la descomposición de una forma rápida y sin mayor dificultad por las condiciones del ambiente en donde se encuentran; por poseer enzimas que favorecen a la fermentación, pueden sufrir ciertas alteraciones al no conservar una temperatura eficiente para la preservación y pueden llegar en malas condiciones, cambiando tanto la estructura física como la estructura orgánica.

Todo producto que degrade la condición de calidad con el tiempo es considerado perecedero y debe manejarse cuidadosa y eficientemente para llegar al consumidor final en condiciones de uso inmejorables.

1.1.1. Transporte de productos perecederos

La temperatura en el momento de la carga debe ser la correspondiente a la exigida para el transporte durante todo el trayecto, hasta el destino final. Los productos deben ser transportados en vehículos con contenedores especialmente acondicionados para ello, cumpliendo las especificaciones para conservar la calidad. Aunque existen varias clases de transporte, todos conllevan en común la baja capacidad de transmisión de calor, impidiendo que la zona donde se transportan los productos cambie de temperatura precipitadamente debido a las condiciones externas.

Las condiciones y normas que deben cumplir el transporte por tierra, mar o aire de productos refrigerados y congelados son (en todos los casos) la conservación de las temperaturas estipuladas durante el trayecto para mantener así, el indicador de calidad. En el caso de los suministros y productos alimenticios congelados se suele señalar además los límites de tolerancia al respecto. Para explícitas provisiones refrigeradas, en ocasiones se demanda la conservación de temperaturas distintas durante el trayecto, en función de la duración del traslado hacia el destino.

Es posible considerar, que en el mismo contenedor, se puedan realizar cargas de diferentes productos, aprovechando al máximo la capacidad del transporte, siempre y cuando compartan la misma temperatura para preservarse y sean compatibles para no alterarse entre sí, las propiedades internas de cada uno.

Las condiciones ideales del transporte para la conservación de la temperatura con bajo margen de error es vital para garantizar seguridad de la calidad del producto al consumidor final.

1.1.2. Temperatura de productos perecederos

El control de la temperatura es uno de los factores más importantes que intervienen en la asignación de calidad del producto. Una vez que el producto es enfriado a la temperatura deseada, la siguiente condición es lograr mantenerlo frío, durante todo el transporte al destino final.

La cadena en frío, o *Cool Chain* es la serie de elementos, actividades y procedimientos para un sistema de conservación estable controlado de manejo, transporte y distribución, necesarios para garantizar la calidad de un producto, desde la adquisición en estado natural o precocido hasta el consumo, o bien desde la elaboración hasta su utilización. La cadena en frío no sólo es ajustable a los alimentos, sino también a múltiples productos sanitarios o farmacéuticos requieren refrigeración o incluso congelación, como es el caso de las vacunas, medicamentos, análisis clínicos, entre otros. La cadena en frío es fundamental para resguardar la calidad de productos perecederos y sensibles a la temperatura y mantenerla es importante por diversas razones, las cuales se relacionan con mantener el indicador calidad del producto. La temperatura tiene una consecuencia inmediata en el grado de respiración de los productos y esta es una indicación del grado de detrimento del mismo, además de que la temperatura acelera el proceso de putrefacción. Si la cadena en frío logra mantenerse a la perfección, ambos factores pueden ser retrasados, dándole una mayor vida útil y manteniendo el indicador de calidad del producto garantizado para el consumidor final.

A continuación se detalla una lista con algunos productos perecederos con la temperatura idónea para la preservación que estos podrán alcanzar, durante el transporte al destino final o bien para la conservación en los puntos bases de distribución:

Tabla I. **Temperaturas aceptadas en productos perecederos**

PRODUCTO(S)	TEMPERATURA MÁXIMA
Cremas heladas	-20ºC
Pescados, moluscos, crustáceos congelados o ultracongelados	-18ºC
Productos ultracongelados	-18ºC
Mantequilla congelada	-10ºC
Resto de productos congelados	-12ºC
Despojos rojos	+3ºC
Mantequilla	+6ºC
Productos de caza	+4ºC
Leche en cisternas	+4ºC
Leche industrial	+6ºC
Productos lácteos refrigerados	+4ºC
Hielo fundente, carne y preparados de carne (excepto despojos rojos)	+7ºC
Aves y conejos	+4ºC
Comidas congeladas	- 18ºC
Comidas refrigeradas con un período de duración inferior a 24 horas	8ºC
Comidas refrigeradas con un período de duración superior a 24 horas	4ºC
Comidas calientes	65ºC
Huevos refrigerados	0 ºC y 15ºC
Huevos congelados	-18ºC

Fuente: http://www.madridsalud.es/temas/transporte_alimentos.php.

Consulta: julio de 2014.

2. BASES DEL DISEÑO PARA EL SISTEMA DE MONITOREO DE TEMPERATURA

Se propone crear una convergencia de tecnología digital, soportado por hardware libre que brinda características de aplicación, tales como precisión, confiabilidad, compatibilidad con diferentes componentes de hardware y software, aplicando diferentes conceptos que interactúan entre sí, para lograr el despliegue de un sistema monitor de temperatura en tiempo real, que envíe parámetros con información susceptible hacia un servidor de aplicación a través de una red WAN y de forma sincronizada, dicha información será persistente en un sistema de bases de datos.

Fundamentándose con la tecnología que brindan las telecomunicaciones (red 3G) y los sistemas de hardware integrados, se puede lograr una interconexión entre ambos para desarrollar un sistema embebido de control y monitoreo remoto de parámetros, que serán enviados a un servidor central para el procesamiento, el cual automatizará los métodos de asignación de calidad de los productos perecederos, evaluando el comportamiento de las condiciones ambientales a las que fueron sometidos durante el trayecto de transporte y comparándolos con valores considerados como ideales para la conservación de la calidad del producto.

2.1. Raspberry Pi Modelo B

Es un dispositivo electrónico u ordenador en placa reducida de bajo costo, desarrollado específicamente para el avance en el estudio tecnológico. Utiliza un sistema operativo basado en Debian, específicamente en Raspbian, que

está optimizando para este hardware, que cuenta además con el conjunto de programas básicos y utilidades, varios paquetes y programas precompilados para una mayor facilidad de instalación y uso de Raspberry Pi.

El modelo más popular en la actualidad de este dispositivo es conocido como Raspberry Pi Modelo B por ser la variante de mayor rendimiento, con 512 MB de RAM, dos puertos USB y un puerto Ethernet 100MB.

Figura 1. **Raspberry Pi modelo B**

Fuente: http://www.raspberrypi.org/product/model-b/. Consulta: julio de 2014.

Este dispositivo cuenta con una serie de pines para diferentes propósitos, como pueden ser los protocolos de comunicación UART (que será utilizado para la comunicación entre Raspberry Pi y el módulo de conexión a la red), I2C, SPI o una salida digital de modulación de ancho de pulso (PWM), además de varios pines de uso general.

Se utilizará este dispositivo como placa base por la capacidad y eficiencia en hardware para interactuar con diferentes dispositivos simultáneamente, especialmente la habilidad para actuar con el protocolo *1-wire* utilizado el pin generador de señal de reloj para mayor precisión en la sincronización y

comunicación, la facilidad en software para la conexión a la red de datos y el bajo consumo en energía que requiere.

A continuación se resaltan las características de la Raspberry modelo B que se utilizará en el sistema:

Tabla II. **Características principales: Raspberry Pi modelo B**

Chip	Procesador de aplicaciones multimedia *Broadcom* BCM2835 SoC *full* HD
CPU	Procesador de aplicaciones de bajo consumo ARM1176JZ-F de 700MHz
GPU	Co-Procesador Multimedia Dual Core, VideoCore IV®
Memoria	512MB SDRAM
Conexión Ethernet	Ethernet 10/100, conector RJ45
USB 2.0	Doble conector USB
Video	HDMI (rev 1.3 & 1.4) / RCA compuesto (PAL y NTSC)
Audio	Conector de 3.5mm, HDMI
Almacenamiento	SD, MMC, SDIO
Sistema operativo	Linux
Consumo energético	700 mA
Costo aproximado	Q 400,00

Fuente: http://downloads.element14.com/raspberryPi1.html. Consulta: julio de 2014.

2.2. Sensor de temperatura DS18B20 (Termocopla)

La presentación termocopla es una versión precableada e impermeabilizada del sensor DS18B20 hecho con un cable de alambre de PTFE para soportar las altas temperaturas. Útil para cuando se necesita para mediciones a distancia, o en condiciones de mucha humedad. Este sensor es un poco más versátil porque se puede utilizar hasta 125 ° C, el límite del propio

sensor debido a que la señal del sensor es digital, no recibe ninguna degradación de la señal, incluso en largas distancias. Estos sensores digitales de temperatura basados en el protocolo *1-wire* son bastante precisos, y pueden dar hasta 12 bits de resolución del convertidor de analógico-digital interno. Trabajan muy bien con cualquier microcontrolador utilizando un único pin digital para la comunicación, e incluso se puede conectar múltiples sensores en el mismo pin, puesto que cada uno tiene un código identificador único de 64 bits preprogramado en la fábrica para diferenciarlos. Se puede utilizar con los sistemas de alimentación de 3.0-5.0V.

Se utilizará este sensor, en la presentación de termocopla por la alta capacidad en el rango de medición de temperatura, la facilidad de conexión, el protocolo *1-wire* en el que está basado utilizando un único pin para la comunicación y la estructura física del mismo para soportar condiciones de uso y ambiente deplorables a las que se pueda someter, siendo flexible y permitiendo alcances de distancia considerables (aproximadamente 1 metro) desde su dispositivo maestro de control.

Figura 2. **Sensor DS18B20 Termocopla**

Fuente: http://blog.thestateofme.com/2013/01/28/ds18b20-rpi/. Consulta agosto de 2014.

Este sensor de temperatura consta de tres pines:

- Pin 1: GND tierra o referencia
- Pin 2: DQ entrada/salida de datos, es la conexión *1-wire* hacia el dispositivo maestro
- Pin 3: Vdd, alimentación (opcional)

El sensor puede operar en modo parásito, en donde el pin de alimentación (3) se conecta al voltaje de referencia (tierra) y la alimentación se obtendrá desde el pin de datos (pin 2) cuando el bus está en alto.

En la siguiente tabla se detallan las características más importantes del sensor de temperatura DS18B20:

Tabla III. **Características del sensor DS18B20 termocopla**

Tensión de alimentación	3,3V a 5,5V
Precisión	± 0,5 °C de -10 °C a +85 °C
Rango de temperatura	-55 hasta 125 ºC (-67 °F a +257 °F)
Resolución	Seleccionable de 9 a 12 de bits
Diámetro del cable	4 mm
Longitud	100 cm aproximadamente
Costo aproximado	Q 80,00

Fuente: http://www.adafruit.com/products/642. Consulta: agosto de 2014.

2.2.1. Comunicación con el sensor de temperatura: Protocolo *1-wire*

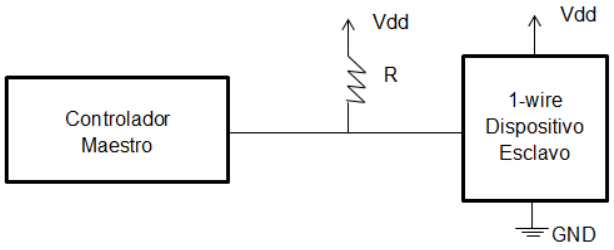
Es una tecnología creada por Dallas Semiconductor y permite la comunicación sin reloj de sincronía, entre varios dispositivos, uno denominado maestro y varios denominados esclavos, utilizando una única terminal de entrada/salida de datos. Los dispositivos esclavos tienen un identificador único

grabado en la ROM por el fabricante, lo que asegura un direccionamiento o comunicación directa con los mismos.

Se debe considerar que para conectar un dispositivo esclavo *1-wire*, se torna necesaria la conexión de una resistencia de *pull-up* aproximadamente de 2,5 a 5 kΩ, enganchada a un voltaje Vdd, comúnmente a 5 V.

A continuación, una representación gráfica de las conexiones necesarias de este protocolo de comunicación:

Figura 3. **Ejemplo de conexión *1-wire***



Fuente: elaboración propia.

2.3. Módulo de conexión a la red RPI GSM/GPRS *Add-on*

Es una creación de hardware libre. El módulo RPI GSM/GPRS *Add-on*, que es una personalización para adaptarse a la interfaz Raspberry Pi modelo B, está basado en el módulo de comunicación GSM/GPRS cuatribanda SIM900. Los comandos de comunicación se pueden enviar a través del puerto serial (UART) de Raspberry Pi, permitiendo así, las funciones de marcar y contestar llamadas, envío y recepción de mensajes y lo más importante para el diseño y aplicación en el sistema de monitoreo de temperatura, la conexión a internet.

Cabe resaltar que las temperaturas que soporta el módulo en funcionamiento, son las temperaturas a las que será sometido para el objetivo de este proyecto. A continuación las características generales del módulo:

Tabla IV. **Características generales: RPI GSM/GPRS *Add-on***

Bandas	850/900/1800/1900 MHz
Tecnología GPRS	GPRS multi-slot clase 10/8GPRS estación móvil B
Potencia nominal RF	Fase 2/2 + Clase 4 (2 W a 850/900 MHz)
	Clase 1 (1 W a 1800/1900MHz)
Interfaz de control	Comandos AT (GSM 07.07, 07.05 y comandos AT mejorados SIMCOM)
Temperatura	-40 ° C a +85 ° C

Fuente: http://wiki.iteadstudio.com/RASPBERRY_PI_SIM900_GSM/GPRS_ADD-ON.
Consulta: agosto de 2014.

Este módulo adquiere su fuente de alimentación de la Raspberry Pi al ser montado directamente sobre esta. Las características eléctricas de operación del módulo son:

Tabla V. **Características eléctricas RPI GSM/GPRS *Add-on***

Parámetro	**Min.**	**Máx.**	**Régimen permanente**	**Dimensional**
Fuente de alimentación	-	5,5	5	VDC
Consumo promedio de corriente	-	-	500	mA
Consumo instantáneo de corriente	-	2	-	A

Fuente: http://wiki.iteadstudio.com/RASPBERRY_PI_SIM900_GSM/GPRS_ADD-ON.
Consulta: agosto de 2014.

A continuación se detalla una tabla de los pines, tanto de la Raspberry Pi modelo B, como del módulo RPI GSM/GPRS *Add-on*, donde se puede observar las conexiones para la interacción serial entre los dispositivos por medio de los pines designados para la comunicación por UART, considerando las siguientes siglas designadas para los nombres en los pines:

- GPIO: *General Purpose Input/Ouput* (pin de propósito general de entrada/salida).
- SDA: *Serial Data* (datos seriales)
- SCL: *Serial Clock* I2C (reloj serial)
- SCLK: *Serial Clock* SPI (reloj serial)
- CE: *Chip Enable* (habilitación de integrado)
- MISO: Master Input-Slave Output (entrada de datos del maestro y salida de datos del esclavo)
- MOSI: Master Output-Slave Input (salida de datos del maestro y entrada de datos del esclavo)
- TxD: Transmisor UART
- RxD: Receptor UART

Tabla VI. **Interfaces en Raspberry Pi**

Número de pin en Raspberry PI	**Nombre del pin**	**Pin en el módulo RPI GSM/GPRS *Add-On***	**Descripción**
11	GPIO0	SIM900-PWERKEY	Encendido vía software
12	GPIO1	SIM900-RST	Reinicio vía software
13	GPIO2		
15	GPIO3		
16	GPIO4		
18	GPIO5		
22	GPIO6		
7	GPIO7		
3	SDA0		
5	SCL0		
24	CE0		
26	CE1		
19	MOSI		
21	MISO		
23	SCLK		
8	TxD	SIM900_RX	Recepción para SIM900 UART
10	RxD	SIM900_TX	Envío para SIM900 UART
2	+5V	-	
1	+3,3V	-	
6	GND	-	

Fuente: http://wiki.iteadstudio.com/RASPBERRY_PI_SIM900_GSM/GPRS_ADD-ON. Consulta: agosto de 2014.

El módulo RPI GSM/GPRS *Add-on* trae además, otros pines macho sobre la placa del dispositivo, tales como:

Tabla VII. **Otras interfaces de conexión en RPI GSM/GPRS *Add-on***

Nombre del pin	Descripción
DR	Confirmación de recepción de datos, puerto serial
DT	Aviso de transmisión de datos, puerto serial
VBAT	SIM900 fuente de 4,2V
G	Tierra o referencia
V	Pin de 3,3V

Fuente: http://wiki.iteadstudio.com/RASPBERRY_PI_SIM900_GSM/GPRS_ADD-ON.
Consulta: agosto de 2014.

El módulo RPI GSM/GPRS *Add-on* trae incorporados en la placa personalizada, *LEDs* indicadores del funcionamiento general:

- Encendido (PWR): cuando existe fuente de alimentación normal a la placa, el led indicador se mantiene encendido.
- Estado (STATUS): cuando el módulo SIM900 opera normalmente, el led indicador se mantiene encendido.
- Red (NET): se utiliza para indicar el estado de la red, el led indicador funciona de la siguiente forma:
    - Apagado: módulo SIM900 no está funcionando.
    - 64ms encendido/800ms apagado*:* módulo SIM900 no encuentra la red.
    - 64ms encendido /3000ms apagado*:* módulo SIM900 registra la red.
    - 64ms encendido /300ms apagado*:* GPRS en comunicación.

La selección de este módulo para el proyecto se debe al bajo costo que presenta, la versatilidad en funcionamiento, la facilidad de instalación en

Raspberry Pi y la sencilla programación que necesita para el envío de telemetría a través de internet.

Figura 4. **Módulo RPI GSM/GPRS *Add-on***

Fuente: http://imall.iteadstudio.com/raspberry-pi-gsm-add-on.html. Consulta: agosto de 2014.

2.3.1. WCDMA: red de conexión del módulo RPI GSM/GPRS *Add-on*

Es la tecnología que se utiliza para conexiones de alta velocidad a internet 3G, en los dispositivos de telefonía móvil, por medio del estándar UMTS.

UMTS es utilizado por los móviles de tercera generación debido a la gran capacidad multimedia, alta eficiencia de acceso a internet y la fiabilidad en la trasmisión de voz comparable a las redes fijas. Permite agregar más usuarios a la red global incrementando la velocidad por usuario móvil.

WCDMA proporciona un mayor ancho de banda espectral, lo que permite mayor transferencia binaria y eficiencia para el transporte de diferentes tipos de servicios en el acceso a la red, ya sea voz o datos. Al formar parte de las

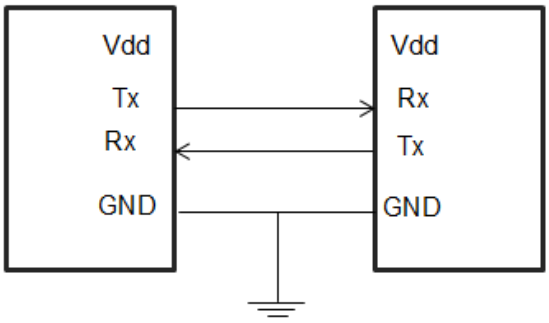
técnicas de acceso múltiple por separación de código, se asigna a los usuarios un código de identificación única, que hace necesario el uso de una tarjeta SIM para identificarse ante la red, pero permite compartir el ancho de banda al no necesitar de separación de frecuencias por usuario.

Actualmente los proveedores de servicio de internet cuentan conexión WCDMA y GSM, tecnologías compatibles con el módulo RPI GSM/GPRS *Add-on*.

2.3.2. Protocolo de comunicación entre el módulo RPI GSM/GPRS *Add-on* y Raspberry Pi: UART

UART es el protocolo de comunicación serial más popular entre dispositivos y puertos, utiliza una línea de comunicación diferente para la transmisión y recepción de datos. Para la sincronización entre dispositivos utiliza un bit de inicio, comúnmente un nivel bajo, y un bit de parada, comúnmente un nivel alto, además de los 8 o más bits de información. El dispositivo transmisor (UART) convierte los *bytes* de información en una secuencia de *bits* para ser transmitidos hacia el dispositivo receptor (UART) que se encargará de ensamblar la secuencia en bytes de información completos. Ambos dispositivos, transmisor y receptor cuentan con un registro de corrimiento, necesario para la conversión de entre las formas serial y paralelo. La transmisión de datos serial es mucho más eficiente en cuantos a costos y hardware comparado con la transmisión en paralelo, los cuales utilizan una cantidad mayor de conexiones. A continuación, una representación gráfica de las conexiones necesarias para este protocolo de comunicación:

Figura 5. **Esquema de UART**



Fuente: elaboración propia.

2.4. Servidores de información

Para lograr que los datos recuperados en el dispositivo modular se conviertan en información que representen un valor cuantitativo en el diseño elaborado, es necesaria la implementación de componentes de hardware y software unificados como servidores de información. Estos dispositivos se convierten en un arreglo de servicio que, dispuestos de forma paralela, brindan soporte al acceso, recepción, manipulación y almacenamiento de los datos, para su posterior análisis, redistribución y control, soportado de forma sincronizada, incluyendo normativas de seguridad que garanticen la integridad y confiabilidad de los datos.

2.4.1. Servidor de aplicación

Un servidor de aplicación comprende una serie de rutinas de software que despliegan servicios, apoyan en la ejecución de algoritmos establecidos para centralizar procesos, reducen la complejidad en la construcción de herramientas de software y optimizan los recursos.

2.4.1.1. Servicio web basado en la transferencia de estado representacional

Dentro de la interfaz que se presenta al cliente se define la tecnología y estándares de los servicios web basados en la transferencia de estado representacional, con el objetivo de brindar un tipo de acceso simple, eficiente, seguro y confiable a través de un URL.

2.4.1.2. Servicio web basado en el protocolo simple de acceso a objetos: SOAP

Utilizado como servidor de aplicación se despliega un servicio web que funcione como sistema de flujo de datos, basado en estándares que simplifican el intercambio de información a través de objetos xml, y que para la implementación presentada funciona como *middleware* entre la interfaz empleada directamente por el cliente y el sistema de bases de datos.

2.4.2. Bases de datos

Una base de datos es un conjunto de datos relacionados entre sí, almacenados de forma estructurada y sistemática con la menor redundancia viable, para el uso posterior por medio de un sistema gestor (DBMS).

Las DB tienen diversos propósitos, como acceso, actualización, ingreso o eliminación de datos de acuerdo a los privilegios de usuario que se le hayan otorgado utilizando la ventaja primordial que varios usuarios pueden acceder a la DB simultáneamente a través de la red, sin interrupción y se convierte más útil y rentable cuando la cantidad de información almacenada asciende.

2.4.2.1. Administración de DB

Por la gran cantidad de información que se puede manejar en una DB y el constante crecimiento de esta, surge la necesidad de contar con un sistema de administración que sirva como interfaz para manipular los datos y tener control sobre los usuarios. La gestión de la DB se lleva a cabo por programas denominados DBSM que permiten la optimización de la información.

Estos gestores permiten que varios usuarios realicen operaciones sobre la DB al mismo tiempo, como puede ser el caso que un usuario opere una consulta, mientras otro usuario realiza una actualización de datos.

2.4.2.2. Tipos y modelos de DB

Existen varios tipos y modelos de DB, pero se considerarán los más importantes para el diseño de este sistema de monitoreo de temperatura.

Los tipos de bases de datos pueden clasificarse de varias formas, en concordancia con el contexto que se esté manipulando y la utilidad de las mismas o las necesidades que satisfagan, por lo que sobresale la DB estática debido a su siguiente característica para este diseño. Las DB estáticas son bases de datos de solo lectura, utilizadas fundamentalmente para almacenar datos que posteriormente se utilizarán para estudiar la conducta de un conjunto de datos a través del tiempo, predecir comportamientos, tomar decisiones y realizar análisis de información.

Los modelos de DB dependen del patrón de administración que se hará con los datos, en este caso se consideran dos modelos, DB transaccionales y

DB relacionales, que se pueden relacionar para aprovechar las características de cada uno.

El modelo DB transaccional se caracteriza porque el único fin es el envío y recepción de datos a altas velocidades. Estas DB son poco habituales y están concentradas en el ámbito de análisis de calidad, datos de producción e industrial. Se debe considerar que el fin único es recoger y rescatar los datos a la mayor tasa de velocidad operable, por lo que la redundancia o duplicación de información no debe ser un problema como con las demás DB, comúnmente para poder aprovecharlas al máximo, se suele permitir cierto tipo de conectividad con las DB relacionales.

El modelo DB relacional se caracteriza porque los datos de las tablas son almacenados con relaciones entre sí, y como cada relación es un conjunto de datos, el orden en el que éstos se almacenen no tiene relevancia o efecto alguno sobre la DB. Al mencionar tablas de datos, se refiere al tipo de modelado de datos donde se guarda la información recogida por un programa y consta de columnas que se denominan campos, estos a su vez contienen datos de un tipo diferente que los demás campos y filas que se denominan registros, los cules contienen datos de los mismos tipos que los demás registros.

En estas DB no deben existir dos tablas con el mismo nombre y la relación entre estas se lleva a cabo por medio claves únicas, conocidas como llaves primarias y foráneas.

- Llaves primarias: son la clave primordial de un registro dentro de una tabla y deben satisfacer la integridad de datos. Es única y elegida entre todos los demás registros que define inequívocamente a todos los demás

atributos de la tabla para especificar los datos que serán relacionados con las demás tablas, por medio de claves foráneas.

- Llaves foráneas: son una referencia a una clave en otra tabla y se ubican en la tabla hija, establece la relación existente en dos tablas y contienen el mismo valor que la clave primaria del registro padre. Las claves foráneas no necesariamente deber ser llaves únicas en la tabla donde se localizan, pero sí a donde hacen referencia.

2.4.3. Servidor de base de datos

Un servidor de base de datos es un conjunto de programas y hardware de almacenamiento masivo, que administra los servicios de base de datos a otros programas, computadoras o dispositivos de información, con base en el modelo de comunicación cliente-servidor.

Este servidor se utiliza para guardar, recuperar, consultar y administrar los datos de una base de datos. Gestiona las actualizaciones de datos, ya sean de nuevo ingreso o eliminación, permite el acceso simultáneo de numerosos servidores, terminales o equipos web utilizando un programa denominado cliente y garantiza la seguridad e integridad de los datos.

Además de las funciones básicas, el software de estos servidores ofrece herramientas para facilitar y optimizar la administración de bases de datos. Algunas de estas funciones son la exportación de datos, la configuración de la jerarquía o permisos de acceso de los usuarios y redundancia de datos para soporte, en caso de fallas.

El servidor contiene el conjunto de datos almacenados sistemáticamente para facilitar las consultas posteriores del usuario, optimizando el tiempo para la

utilización de la información para ser compartida con una infinidad de clientes simultáneamente y de manera segura, considerando que el aumento de la velocidad a la que se puede escribir y recuperar datos en la DB tendrá un efecto significativo en el rendimiento.

Los servidores de datos deben facilitar mecanismos de comunicación eficientes, porque de cómo se envíe la información dependerán parámetros significativos como la velocidad de acceso a los datos. Todos los sistemas o programas gestores considerados cuentan con numerosas configuraciones de protocolos, acoplándose a los existentes y estandarizados de la actualidad como el TCP/IP; es trascendental no sólo el canal de comunicaciones que está disponible para los servidores de datos sino también cómo es transferida la información.

3. HARDWARE DEL SISTEMA DE MONITOREO DE TEMPERATURA

En el sistema requerido se utilizará hardware *Open Source* como base fundamental del monitor de temperatura. El diseño se divide en tres componentes principales, los cuales se representan por: el módulo recolector de temperatura, que hace uso del sensor DS18B20 conectado al pin GPIO4 de la placa base; el módulo integrador, que contiene las interfaces de recepción y transmisión de los datos obtenidos haciendo uso de los protocolos *1-wire* y UART; y el módulo de conexión móvil, que contiene configuraciones personalizadas para que funcione como dispositivo de acceso a la red celular integrado a Raspberry Pi. Finalmente el sistema contempla una fuente de alimentación para los componentes que cumplan especificaciones de demanda de corriente y voltaje fijo, que sean adaptables a las condiciones de baterías del transporte de los productos o la corriente convencional AC de 120V.

Los dispositivos propuestos vienen con pines de fácil implementación para evitar así el diseño o construcción de nuevas interfaces de conexión para el funcionamiento.

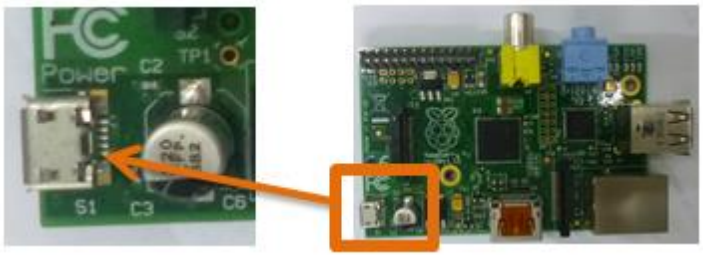
3.1. Fuente de alimentación al monitor de temperatura

Los módulos a utilizar, Raspberry Pi modelo B, RPI GSM/GPRS *Add-on* y el sensor de temperatura DS18B20 pueden operar con 5 VDC de alimentación.

Como el dispositivo RPI GSM/GPRS *Add-on* se conecta por medio de pines hembra a todos los pines macho de Raspberry Pi, incluyendo los de

alimentación de 5 VDC de donde obtiene la fuente de alimentación al igual que el sensor de temperatura, se debe considerar la conexión alternativa de la Raspberry Pi de entrada microusb por medio de un cable que cubra las especificaciones, incluido en el paquete de compra de Raspberry Pi, cumpliendo así las condiciones eléctricas para el funcionamiento.

Figura 6. **Puerto micro-USB de alimentación de voltaje en Raspberry Pi**



Fuente: elaboración propia, en laboratorios de electrónica de FIUSAC.

Si el dispositivo modular fuese montado en contenedores de enfriamiento halados por tráiler, en donde se localiza una batería de DC de 12 o 24 voltios, para la reducción del voltaje a 5 VDC se recomienda utilizar un conversor de voltaje CD-CD que contenga en la salida de voltaje un conector USB.

Figura 7. **Conversor CD-CD con salida USB**

Fuente: http://i.ebayimg.com/00/s/MTYwMFgxNjAw/z/NV4AAOxyLchRsVKZ/$T2eC16hHJHIE9nysfrP7BRsVKZjSfg~~60_57.JPG. Consulta: agosto de 2014.

A continuación se detalla el conversor CD-CD con las especificaciones necesarias para utilizar como fuente de alimentación hacia Raspberry Pi:

Tabla VIII. **Especificaciones de conversor CD-CD con salida USB**

Tipo de rectificación	No síncrona
Voltaje de entrada	7- 24 VDC
Voltaje de salida (fijo)	5 VDC
Corriente de salida	3 A (máx.)
Eficiencia de conversión	96% (máx.)
Frecuencia de conmutación	340 KHz
Rizado de salida	30 mV (máx.)
Regulación de carga	± 0,5%
Regulación de voltaje	± 2,5%
Temperatura de operación	-40 °C a +85 °C
Protección	Contra sobrecorriente
Costo aproximado	Q 20,00

Fuente: http://www.ebay.com/itm/5V-USB-DC-7V-24V-to-5V-3A-Step-Down-Buck-KIS3R33S-Module-Arduino-than-LM2596/181008007744. Consulta agosto de 2014.

Se recomienda este conversor debido a la facilidad de instalación y tamaño, con la seguridad de obtener una salida de voltaje fija de 5 VDC en un conector USB apropiado, sin importar si el voltaje de entrada es variable, además de satisfacer las demandas de corriente nominal para cada dispositivo.

Si el dispositivo modular fuese montado en contenedores inmóviles refrigerados, en donde se tiene acceso a un voltaje de 120 VAC o 240 VAC, puede utilizarse cualquier tipo de conversor de AC-DC que requiera una entrada de voltaje alterno y mantenga una salida fija de 5 VDC a un conector USB, satisfaciendo las demandas de corriente.

A continuación se detalla un adaptador AC-USB con las especificaciones necesarias para utilizar como fuente de alimentación hacia los módulos, haciendo hincapié en que cumple con al menos, 2,7 amperios de corriente directa de salida, compensando la corriente mínima nominal del monitor de temperatura completo:

Tabla IX. **Especificaciones de conversor AC-CD con salida USB**

Tipo de rectificación	No síncrona
Voltaje de entrada	120- 240 VAC, 50/60 Hz
Voltaje de salida (fijo)	5 VDC
Corriente de salida	2 A (máx.)
Temperatura de operación	-40 ° C a +85 ° C
Protección	Contra sobre-corriente y sobrecarga
Costo aproximado	Q 50,00

Fuete: http://www.ebay.com/itm/AC-100-240V-0-5A-DC-5V-2A-US-Plug-USB-Power-Supply-Adapter-Converter-Charger-/370877746122. Consulta: agosto de 2014.

Se recomienda este conversor debido a la facilidad de instalación y tamaño, con la seguridad de obtener una salida de voltaje fija de 5 VDC en un

conector USB apropiado, considerando las dos alternativas tomas de voltaje (120-240 VAC), además de satisfacer las demandas de corriente nominal para cada dispositivo.

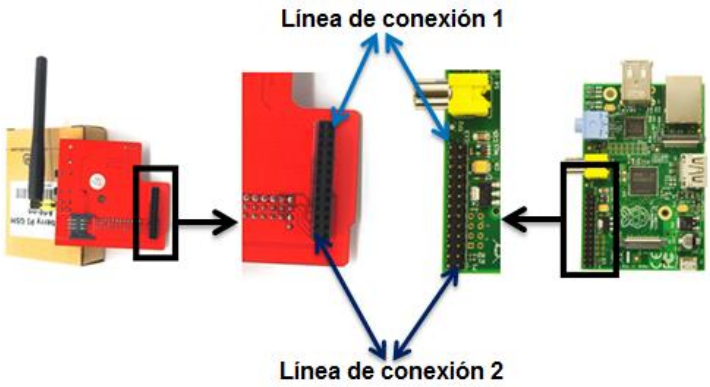
Figura 8. **Conversor AC-DC con salida USB**

Fuente: http://www.ebay.com/itm/AC-100-240V-0-5A-DC-5V-2A-US-Plug-USB-Power-Supply-Adapter-Converter-Charger-/370877746122. Consulta: 14 de julio de 2014.

3.2. Conexión del módulo RPI GSM/GPRS *Add-on* a Raspberry Pi

El módulo RPI GSM/GPRS *Add-on*, por ser una placa personalizada, la conexión se realiza montando este módulo directo sobre los pines macho de la Raspberry Pi, quedando sobre la placa del módulo los puertos o pines libres para su utilización en otras actividades o conexiones con diferentes dispositivos o usos, como sería en el monitor de temperatura, con el sensor DS18B20.

Figura 9. **Conexión del módulo RPI GSM/GPRS *Add-on* a Raspberry Pi**



Fuente: elaboración propia, en laboratorios de electrónica de FIUSAC.

La comunicación entre estos dispositivos de la figura 9 se realiza por medio de UART, utilizando los pines Rx y Tx, conectados en forma cruzada, respectivamente. Se puede observar en la tabla V el diagrama de conexión.

3.3. Conexión del sensor DS18B20 a Raspberry Pi

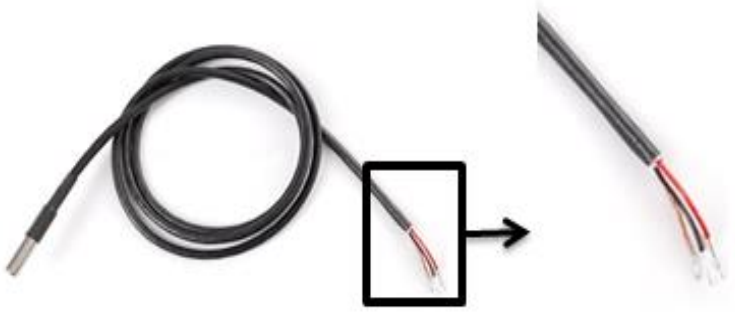
Debido a que módulo RPI GSM/GPRS *Add-on*, en el acoplamiento a Raspberry Pi ocupa todos los pines, la conexión del sensor de temperatura no se logra directamente.

En el módulo RPI GSM/GPRS *Add-on* se encuentran en forma de pines macho, los pines no utilizados de Raspberry Pi y que quedan disponibles para cualquier otra conexión, además de tener disponibilidad de conectores macho de voltaje de alimentación y voltaje de referencia o tierra, que servirán para conectar el sensor de temperatura.

El sensor de temperatura DS18B20 cuenta con tres pines o cables (descritos la sección 2.2) que vienen con un color diferente para identificarlos, que son:

- Rojo: voltaje positivo (3-5.5 VDC)
- Negro: voltaje de referencia o tierra
- Amarillo: datos

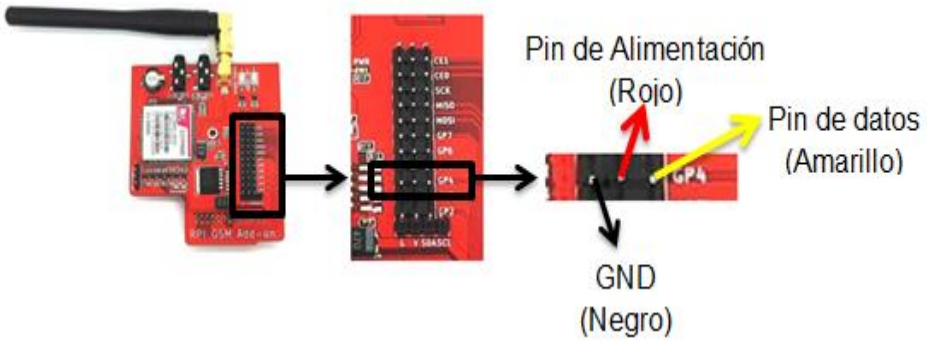
Figura 10. **Pines del sensor de temperatura DS18B20**



Fuente: elaboración propia, en laboratorios de electrónica FIUSAC.

Por consiguiente, la conexión del sensor de temperatura en el módulo RPI GSM/GPRS *Add-on* se detalla en la siguiente figura:

Figura 11. **Conexión del sensor de temperatura DS18B20**



Fuente: elaboración propia, en laboratorios de electrónica de FIUSAC.

El pin de datos del sensor de temperatura es conectado al pin GP4 (pin de propósito general 4) que quede directamente vinculado al GPIO4 de la Raspberry Pi.

Se recomienda utilizar una resistencia de *pull-up*, aproximadamente de 4,7 kΩ, que debe ser conectada entre el pin de datos (amarillo) y el pin de alimentación (rojo).

4. SOFTWARE DEL SISTEMA DE MONITOREO DE TEMPERATURA

En la etapa de análisis y diseño del software que se presenta, se recomienda visualizar la arquitectura establecida tomando como punto de partida la persistencia de los datos, la cual contiene el repositorio de almacenamiento, gestión y manejo de accesos, optimización de consultas, tipos de datos, estructuras necesarias para formar el banco de información que servirá de soporte para crear el valor agregado en la integración de los datos para convertirse en información. Estableciendo las estructuras de datos que se utilizarán, subsecuentemente debe implementase el componente de software que funciona como intermediario y brinda acceso a la capa de aplicación disponible al cliente y que permite de forma segura el intercambio de datos.

Establecida la comunicación entre el *middleware* y la base de datos se despliega la interfaz que atiende las peticiones del componente cliente almacenado en el dispositivo modular que interactúa de forma sincronizada, enviado datos sensibles para el almacenamiento. Dicho componente cliente dispone de una rutina ejecutada que comprende de un algoritmo eficiente que recolecta la temperatura del entorno.

4.1. Creación de la base de datos (DB)

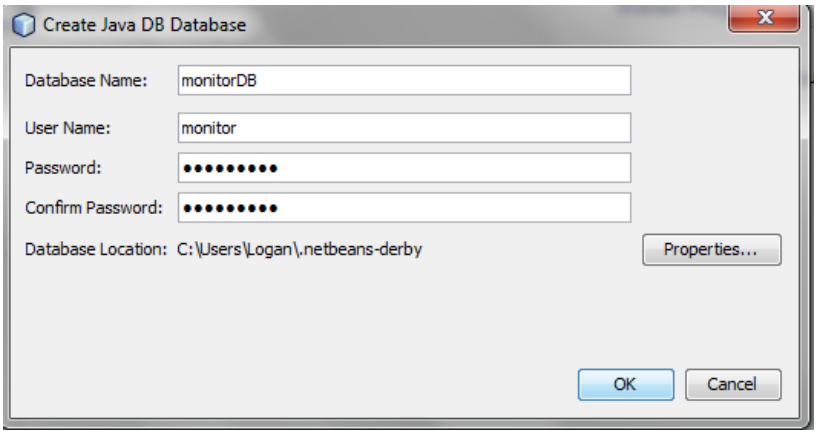
Basado en el tipo de información (liviana), la concurrencia baja (transacciones por segundo), el volumen de tráfico (kilobytes transmitidos) es funcional y eficiente hacer uso de la herramienta DBMS de software libre conocida como Sun Java DB que soporta las características principales de un

gestor de bases de datos formal y administra de forma eficaz las operaciones deseadas en la implementación de la solución propuesta, tales como agregar, consultar, actualizar y eliminar registros de información.

Este gestor de DB es intuitivo, por lo que la creación y manejo de la DB no se tornará complicado y el acceso a la información será de manera rápida y segura.

A continuación se detallan los pasos para la construcción de la DB en Sun Java DB a través del entorno de desarrollo integrado Netbeans. Como primer paso se crea la DB, agregando el nombre que se le asignará, un nombre de usuario para el manejo y un *password* de seguridad.

Figura 12. **Creación de DB**



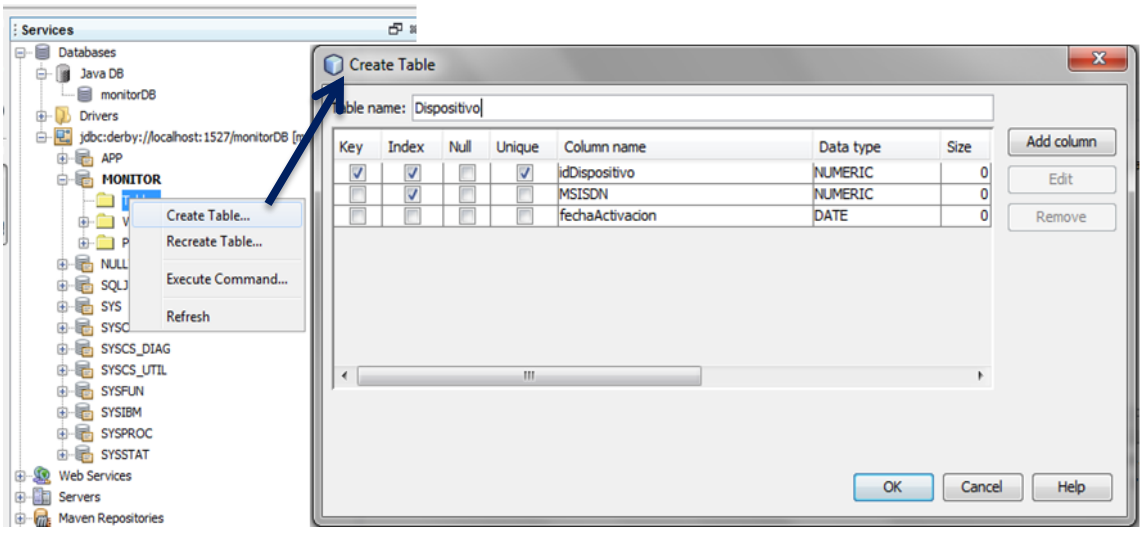
Fuente: elaboración propia, con programa de Sun Java DB.

Una vez asignados los datos anteriores se procede a la elaboración de las tablas a utilizar dentro de la DB con sus respectivos campos y tipo de datos para consolidar el registro deseado en cada tabla y así poder asignar los tipos

de claves de relacionales, ya sean llaves primarias o foráneas para facilitar la consulta de datos.

En la siguiente figura se muestra como se crean las tablas y los campos:

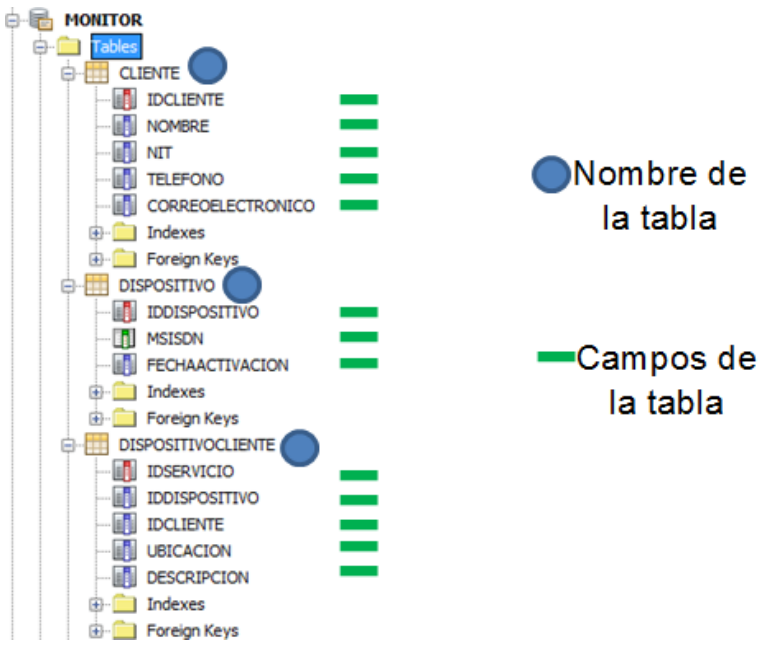
Figura 13. **Creando tablas en DB**



Fuente: elaboración propia, con programa de Sun Java DB.

Después de crear y llenar las tablas con los respectivos datos, quedará una estructura de la DB como la siguiente imagen:

Figura 14. **Estructura de la DB**



Fuente: elaboración propia, con programa de Sun Java DB.

Todas tablas utilizadas en la DB del sistema pueden ser consultadas en el apéndice I.

4.2. Creación del servicio web basado en el protocolo de acceso simple a objetos

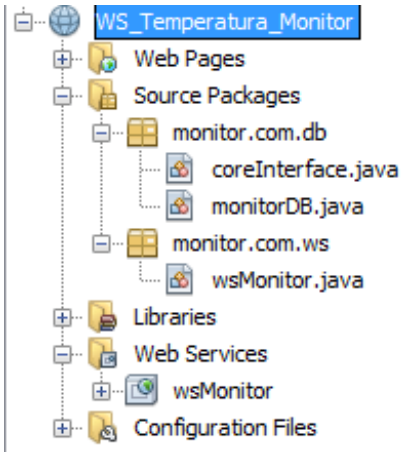
Siguiendo con la directriz de diseño se procede a crear el componente intermediario entre la interfaz desplegada al cliente y el gestor de DB.

El objetivo es disponer de métodos SOAP que interactúen con la base de datos monitorDB y básicamente comprende en generar las estructuras *request* y *response (objetos xml SOAP)* que transporten los campos que identifiquen al

dispositivo que envía el reporte de temperatura. La propuesta del objeto SOAP sugerido puede ser consultada en el apéndice II.

Utilizando como IDE para programar el *web service* a Netbeans 7.2 se debe crear un proyecto web para desplegar el servicio *web* respectivamente. Para ello se sugiere la siguiente estructura:

Figura 15. **Estructura de clases para el servicio web SOAP**



Fuente: elaboración propia, con programa de Netbeans 7.2.

El principio de programación se basa en tres clases que serán nombradas coreInterface, wsMonitor y monitorDB, los métodos de implementación propuestos pueden ser consultados en los apéndices III, IV y V respectivamente.

Para complementar la descripción de las estructura de clases, se puede hacer énfasis en la definición del paquete monitor.com.db que contiene a la clase monitorDB, la cual es la encargada de obtener el recurso de conexión

hacia la base de datos y enviar las instrucciones DML que harán persistente la información recibida.

Se procede a crear la clase coreInterface que realiza directamente el *binding* y que establece la comunicación hacia la base de datos y le pasa de parámetro este objeto a monitorDB.java.

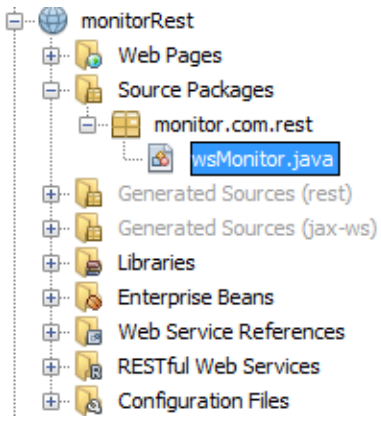
Para finalizar con este componente de software se crea el paquete monitor.com.ws que contiene la clase wsMonitor, en la cual se presenta la interfaz SOAP que recibirá los parámetros que necesitan ser almacenados.

4.3. Creación del servicio web basado en la transferencia del estado representacional

Para presentar una capa de aplicación liviana que permita recuperar los parámetros de información enviados desde el dispositivo modular acerca de la temperatura recolectada, se deben desplegar métodos *get* que interactúen con el script de Python para la transmisión de los valores de temperatura haciendo uso del protocolo TCP/IP (capa 4) y el protocolo HTTP (capa 7).

Utilizando como IDE para programar el w*eb service RESTFul a* Netbeans 7.2 se debe crear un Proyecto Web para desplegar el servicio web respectivamente. Para ello se sugiere la siguiente estructura:

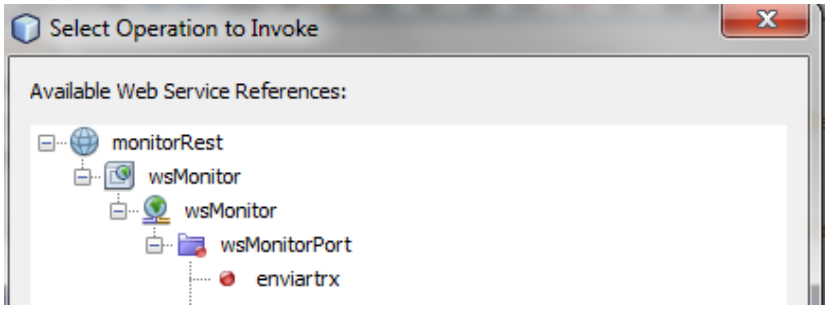
Figura 16. **Estructura del web *service RESTFul***



Fuente: elaboración propia, con programa de Netbeans 7.2.

Es importante mencionar que este componente hace referencia al servicio web SOAP creado en el apartado anterior y el método de implementación puede ser consultado en el apéndice VI.

Figura 17. **Operación para invocar al servicio web SOAP**



Fuente: elaboración propia, con programa de Netbeans 7.2.

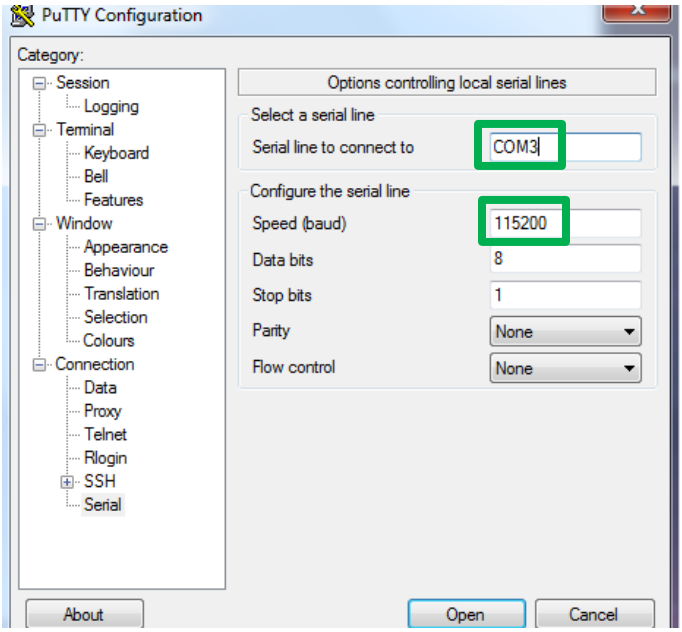
El funcionamiento de la implementación de este servicio puede consultarse en el apéndice VII utilizando una dirección IP local.

4.4. Lectura de parámetros en el monitor de temperatura: código identificador del dispositivo modular y temperatura del ambiente

Continuando con el criterio del diseño y llegando a la etapa final de la estructura del software del sistema, se procede al diseño y creación de la rutina que se ejecutará en el cliente.

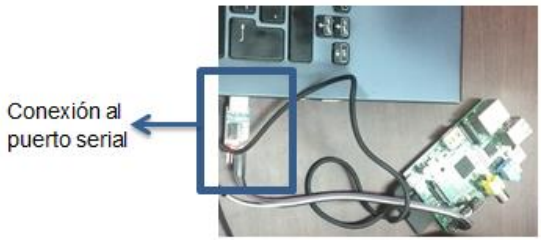
Para comenzar se debe acceder a la Raspberry Pi por medio de la consola serial, utilizando una aplicación ejecutable de un cliente SSH y Telnet de fácil configuración y permite el acceso remoto utilizando la consola como monitor hacia los servidores, conocida como Putty. Para inicializar la comunicación con Raspberry Pi se configura el puerto serial asignando nombre de puerto y la tasa de símbolos por segundo, en este caso COM3 y 115200 baudios respectivamente. Se debe crear primero el puerto serial con la configuración y después se enciende Raspberry Pi.

Figura 18. **Configuración del puerto serial en Putty**



Fuente: elaboración propia, con programa de PuTTY.

Figura 19. **Conexión de Raspberry Pi al puerto serial**

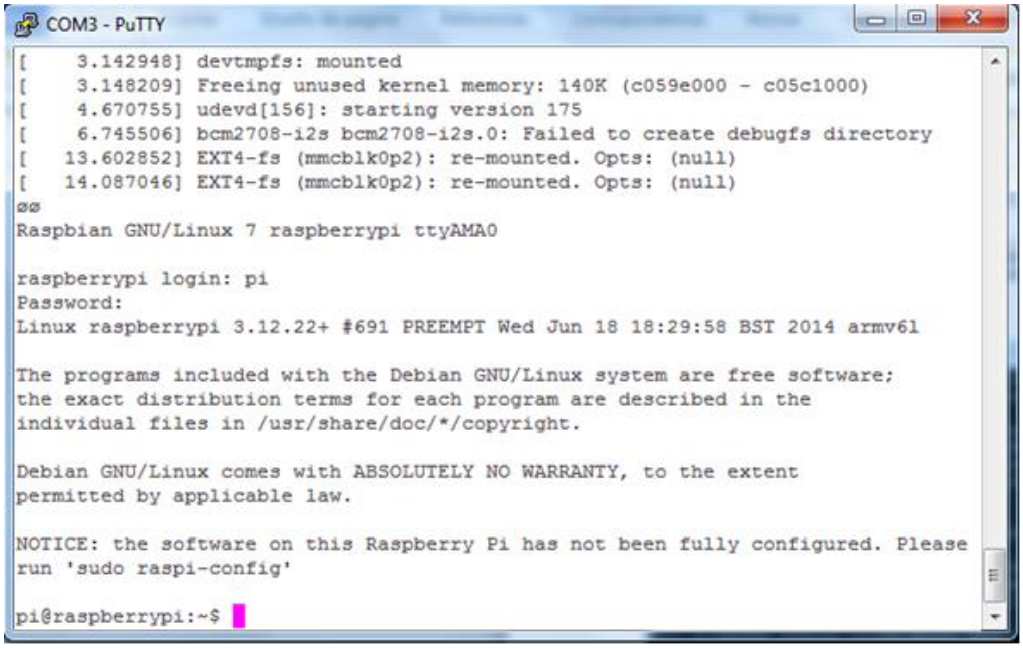


Fuente: elaboración propia, en laboratorios de electrónica FIUSAC.

Se debe desactivar salida TTY del sistema operativo por el puerto serial para evitar la transferencia de datos por todos los demás periféricos conectados en Raspberry Pi que no serán utilizados.

Una vez abierta la conexión con el puerto serial se usará como *boot* a través de consola de Linux por la falta de monitor, y ya existe una interfaz para comenzar la programación.

Figura 20. **Interfaz visual de Linux por consola hacia Raspberry pi**



Fuente: elaboración propia, con programa de PuTTY.

En el programa principal que se ejecuta en el cliente, se enviarán dos parámetros al servidor, el código de identificación único de cada dispositivo monitor de temperatura y el parámetro de temperatura del ambiente.

- Asignación del código de identificación único al monitor de temperatura: el código único identificador del dispositivo servirá para tener un control en la DB de a quién pertenece dicho dispositivo para la clasificación y estructuración en las tablas de control. Este código de identificación se

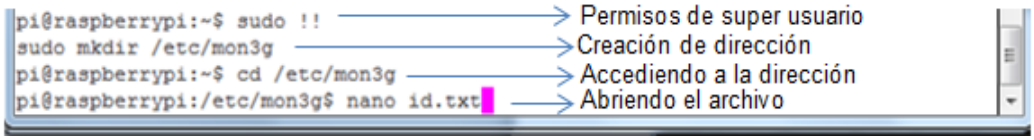
asignará manualmente en la Raspberry Pi para evitar repetición de datos y tener mayor seguridad en el manejo de los clientes que cuenten con el servicio de monitoreo de temperatura en tiempo real.

A continuación se detallan los pasos para crear un archivo que contenga este código de identificación para después ser interpretado por el algoritmo recolector de datos y que enviará a la red 3G.

- Para tener todos los permisos de super usuario en Linux se debe ejecutar el comando *sudo su*.
- Luego se procede a la creación del directorio y la carpeta con el comando *mkdir* en donde se encontrará el archivo de texto plano.
- Para crear el archivo se va a la dirección utilizando el comando *cd* y la dirección asignada.
- Utilizando el editor de texto de Linux conocido como *nano* se abre el archivo con un nombre y la extensión, en este caso se le asigna el nombre de id.txt.

La siguiente figura muestra el orden de la serie de comandos escritos en consola:

Figura 21. **Creación de un archivo de texto en Linux**



Fuente: elaboración propia, con programa de PuTTY.

- Una vez abierto el archivo se rellena con el código de identificación que se le asignará al dispositivo a configurar y se almacena, quedando guardado en la carpeta de la dirección asignada anteriormente.

Figura 22. **Asignación del código identificador único**

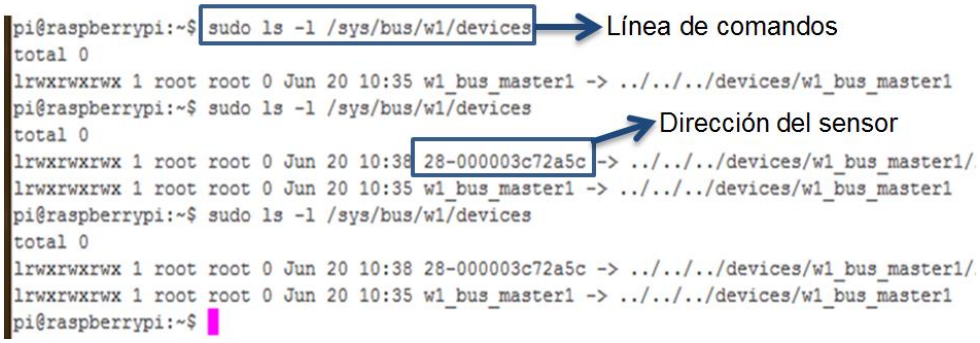
Fuente: elaboración propia, con programa de PuTTY.

- Lectura de temperatura del sensor DS18B20: por el protocolo de comunicación que maneja este sensor es necesario identificarlo con el código programado de fábrica para poder utilizarlo y obtener los parámetros de temperatura. A continuación los pasos para el control del sensor.

- Con el sensor conectado correctamente al pin GPIO4 de Raspberry Pi y la resistencia de *pull-up,* en consola se escribe la siguiente línea de comandos para detectarlo y obtener el código de identificación de fábrica: ls –l /sys/bus/w1/devices.

La siguiente figura muestra como aparece en consola el código del sensor después de ejecutar el comando:

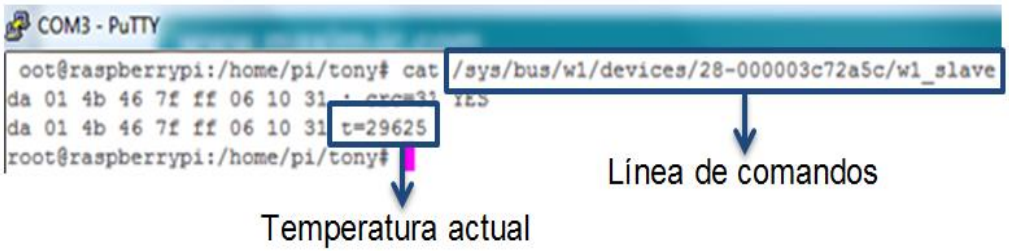
Figura 23. **Detección del sensor de temperatura**



Fuente: elaboración propia, con programa de PuTTY.

Una vez obtenido el código del sensor se utiliza para hacer la lectura actual de la temperatura con la siguiente línea de comandos en consola: /sys/bus/w1/devices/28-000003c72a5c/w1_slave

Figura 24. **Lectura de temperatura en sensor DS18B20 con Raspberry Pi en consola de Linux**



Fuente: elaboración propia, con programa de PuTTY.

- Para obtener el parámetro de temperatura en grados Celsius, se divide en un valor de 1 000, tomando como ejemplo la figura 24, la temperatura actual es:

$$T = \frac{29\,625}{1\,000} = 29{,}625℃$$

4.5. Configuración del módulo RPI GSM/GPRS *Add-on* como *modem* serial

Este módulo por definición viene configurado para realizar una conexión a la red de 2G, por lo que será necesario realizar una configuración para que actúe como *modem* para el monitor de temperatura y así poder conectarse con el servidor de información para enviar la información de los parámetros.

La configuración se realiza mediante un paquete propio de Debian conocido como PPPConfig que permite automatizar la creación de una conexión PPP (*Point to Point Protocol*) a través de un asistente.

La configuración se realiza llamando al paquete PPPConfig desde consola como se muestra en la siguiente figura:

Figura 25. **Ejecutar paquete PPPConfig**

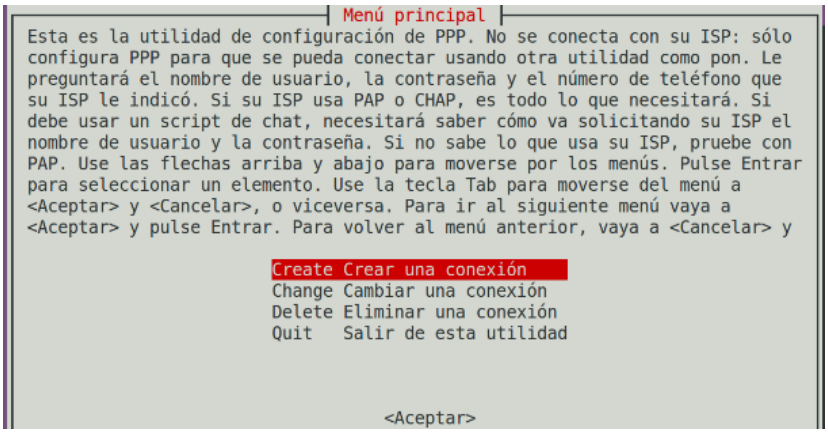
Fuente: elaboración propia, con consola en Ubuntu.

Al ejecutar el comando, abre un menú principal para comenzar la configuración guiada por instrucciones y una descripción de los parámetros a ingresar en cada caso.

A continuación, se muestra el modo de ingreso y los parámetros para que la Raspebrry Pi detecte al módulo RPI GSM/GPRS *Add-on* como un *modem* a través del puerto serial.

- En el menú principal desplegado por el paquete PPPConfig se selecciona: crear una conexión como se muestra en la figura 26:

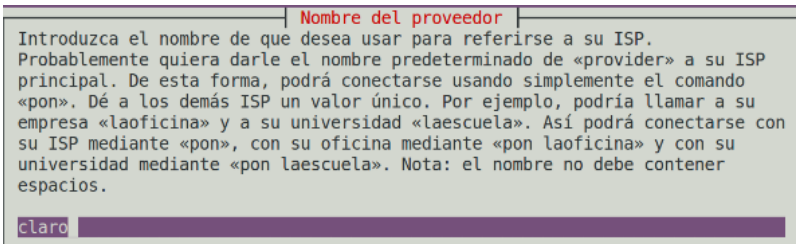
Figura 26. **PPPConfig: crear una conexión**



Fuente: elaboración propia, con programa de PPPConfig.

- El siguiente paso es el ingreso del nombre del proveedor de servicios de internet (ISP), para el monitor de temperatura se utiliza una SIM de Claro, por lo que el nombre del ISP a ingresar es: claro.

Figura 27. **PPPConfig: nombre del ISP**

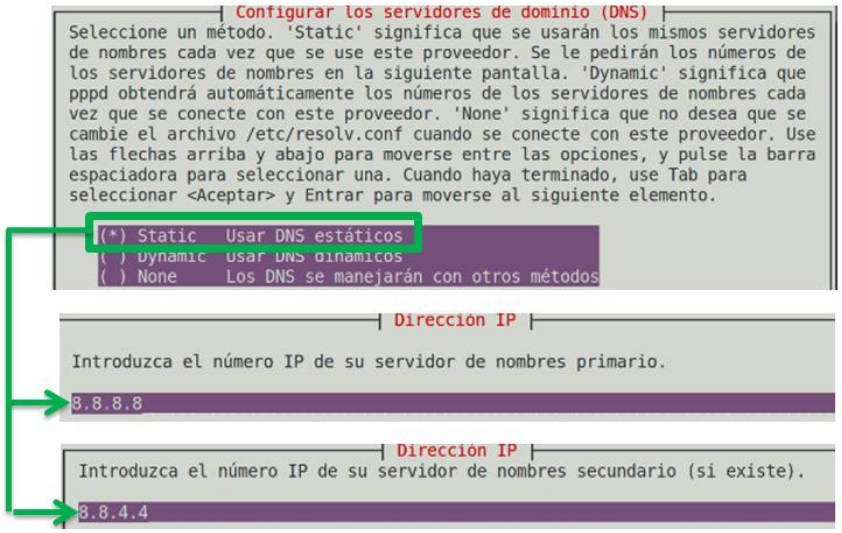


Fuente: elaboración propia, con programa de PPPConfig.

- El siguiente paso es la configuración del Servicio de Nombres de Domino (DNS). Se selecciona un DNS estático para utilizar siempre el mismo servidor y a continuación se ingresan las direcciones IP de este, una

dirección primaria y una secundaria en caso de existir. Se utilizarán las siguientes IP porque son direcciones confiables, ya que son los Servidores de Nombre de Dominio públicos de Google.

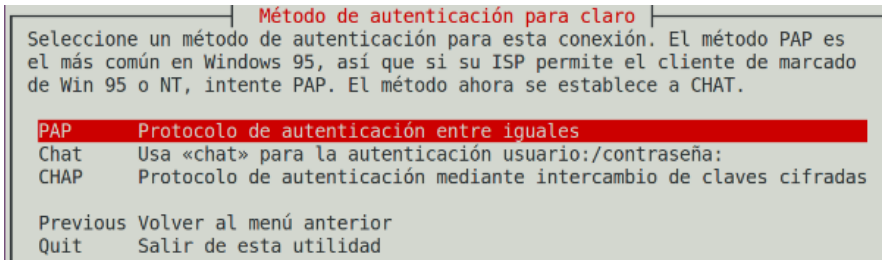
Figura 28. **PPPConfig: configuración DNS**



Fuente: elaboración propia, con programa de PPPConfig.

- El siguiente paso es la selección del método de autenticación a la red del dispositivo cada vez que se conecta, se utilizará el protocolo de autenticación entre iguales (PAP) debido a que es el que se usa en el Proveedor de Servicios de Internet, para este caso, Claro.

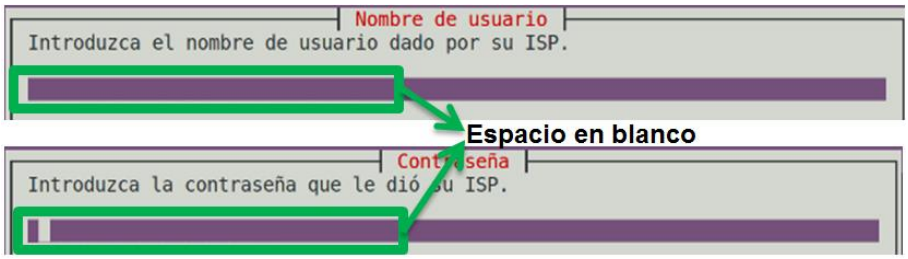
Figura 29. **PPPConfig: método de autenticación**



Fuente: elaboración propia, con programa de PPPConfig.

- El siguiente paso es el ingreso del nombre de usuario y la contraseña, parámetros dados por el ISP, en el caso de Claro Guatemala, estos dos parámetros no son requeridos, por lo que no se escriben en la configuración, pero para que se permita avanzar al siguiente paso, es necesario teclear un espacio en blanco en ambos casos.

Figura 30. **PPPConfig: usuario y contraseña dado por ISP**

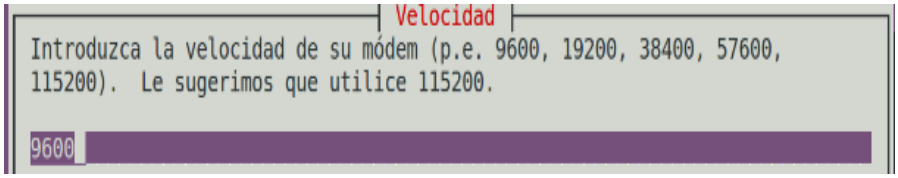


Fuente: elaboración propia, con programa de PPPConfig.

- El siguiente paso es seleccionar la velocidad de comunicación por el puerto serial entre Raspberry Pi y el módulo RPI GSM/GPRRS *Add-on,* este último trae una configuración de fábrica a 9 600 baudios y es la que

se utilizará, puesto que la cantidad de información es mínima y no afecta el proceso de comunicación.

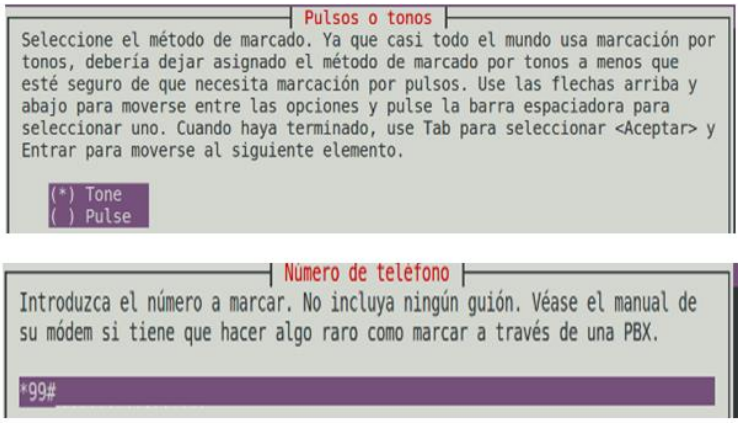
Figura 31. **PPPConfig: velocidad de comunicación del puerto serial**



Fuente: elaboración propia, con programa de PPPConfig.

- El siguiente paso es seleccionar el tipo de marcado y el número del *modem* al cual se llama para adquirir el servicio. Se utilizará marcación por tonos llamando al número ingresado para conectar al servicio.

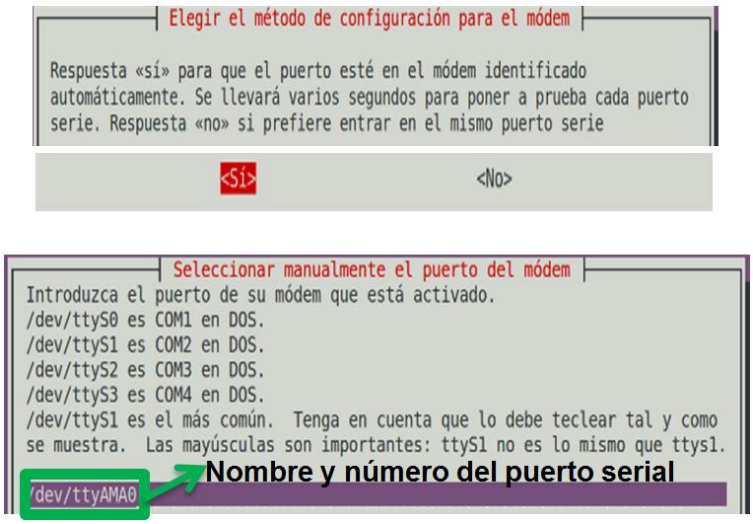
Figura 32. **PPPConfig: método de marcado y número de teléfono**



Fuente: elaboración propia, con programa de PPPConfig.

- El siguiente paso es seleccionar configurar manualmente el puerto del *modem* que se utilizará se debe tomar en cuenta el uso de mayúsculas y el lenguaje en que se trabaja, puesto que en este caso, para nombrar al puerto serial no se usa la terminología COM de Windows, sino el análogo en Linux de AMA y el respectivo número de puerto como se muestra en la siguiente figura.

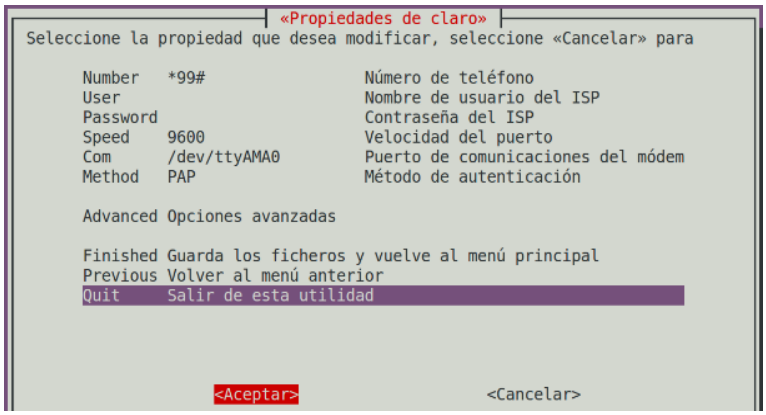
Figura 33. **PPPConfig: Configuración de puerto del *modem***



Fuente: elaboración propia, con programa de PPPConfig.

- Después de realizar todos los pasos anteriores, se muestra una ventana con los parámetros ingresados para una revisión final y si es necesario modificar algún dato para luego guardar los ficheros como se muestra en la siguiente figura.

Figura 34. **PPPConfig: Guardar configuración**



Fuente: elaboración propia, con programa de PPPConfig.

Una vez realizados los pasos anteriores correctamente se generan dos *scripts* (pueden ser consultados en el apéndice VIII y IX) que se almacenan en la tarjeta SD de Raspberry que contienen la configuración y las instrucciones a ejecutar para conectarse al ISP (Claro Guatemala), por medio del módulo RPI GSM/GPRS *Add-on,* que ahora es detectado como *modem* serial.

Dentro del programa principal, descrito en el apartado siguiente se hace el llamado a uno de los *scrpits* almacenado con el nombre de Claro, para ejecutar la rutina de configuración para la conexión a internet, realizada una vez únicamente.

4.6. Creación del programa principal

El programa principal en Python, que al igual que el archivo de texto del código de identificación del dispositivo, se logra agregando una dirección o ruta a la carpeta contenedora, abriendo el archivo y asignando el código principal,

haciendo la salvedad que para este archivo la extensión será .py, por ejemplo: monitor3g.py.

La arquitectura del programa consta de:

- Importación bibliotecas: las bibliotecas son los paquetes con las herramientas básicas para que determinadas funciones del programa principal se ejecuten. Una biblioteca importante en este programa, es la que permita acceder a internet por medio de una dirección web.
- Definición de funciones: son subrutinas que realizan de una serie de instrucciones y devuelven parámetros. Las funciones utilizadas en el programa sirven para:
  - Leer el código de identificación del monitor de temperatura asignado por el usuario.
  - Detección del sensor de temperatura y el código de identificación único.
  - Recoger el parámetro de temperatura actual del sensor de temperatura.
  - Realizar la configuración del módulo RPI GSM/GPRS *Add-on* para que actúe como *modem* serial llamando al *script* generado por PPPConfig.

- Estructura principal: aquí se encuentran los comandos para agregar los módulos del protocolo *1-wire* para el sensor de temperatura y la detección de este. También se encuentra un ciclo infinito (mientras esté encendido el monitor de temperatura) en donde se envía los parámetros de temperatura y el código de identificación del dispositivo, a través de una dirección web, utilizando una función de la biblioteca que permite el acceso a internet.

El código fuente del programa principal puede ser consultado en el apéndice XI.

CONCLUSIONES

1. El sistema de monitoreo de temperatura diseñado, contribuye con la organización de la logística de distribución de productos perecederos con base en el indicador de calidad obtenido en tiempo real.

2. El sistema de monitoreo de temperatura es una opción rentable en la industria, representando una optimización de recursos de tiempo y costo al aumentar la eficiencia en la productividad y confiabilidad de productos perecederos.

3. El sensor de temperatura DS18B20 es un dispositivo electrónico con interfaz de control digital.

4. La interfaz digital del sensor de temperatura DS18B20 garantiza que la señal obtenida no presente atenuación y permite una resolución programable de hasta 12 bits.

5. El uso de servicios web como capa de aplicación, aseguran una implementación basada en protocolos de comunicación estándar que permiten la integración de diferentes tecnologías.

6. Los servicios web permiten un intercambio de información, admitiendo la compatibilidad de servicios y la arquitectura escalable para un sistema transaccional dentro una red de dispositivos.

7. La Raspberry Pi cuenta con una estructura modular que permite la integración de diversos componentes con fines específicos y configurables.

8. La Raspberry Pi permite construir dispositivos de alto desempeño para realizar tareas de optimización de procesos apoyados por tecnología bajo un previo análisis.

9. La comunicación móvil hace uso de la red celular debido a la velocidad de transferencia de datos y la señal existente prácticamente en cualquier ubicación geográfica.

10. El uso del protocolo SOAP, como base en los servicios web, permite la interoperabilidad de mensajes en redes de dispositivos consumiendo objetos transaccionales diseñados apropiadamente para el envío bajo el protocolo TCP/IP.

11. El protocolo SOAP optimiza el ancho de banda durante la transmisión hacia el servidor remoto.

12. El uso de la técnica RESTFul para la creación de un sistema de recursos que pueden ser accedidos de forma remota, permite la creación de una arquitectura de objetos distribuidos.

13. Los objetos creados con la técnica RESTFul pueden ser accedidos por medio de un identificador global dentro de una red haciendo uso del protocolo HTTP.

RECOMENDACIONES

1. En la implementación del monitor de temperatura puede considerarse que durante la instalación, la referencia para las mediciones en el ambiente es el punto medio del arreglo que empaqueta los productos a evaluar para que el parámetro obtenido por el sensor sea lo más próximo al valor real.

2. Considerar que antes de la instalación del monitor de temperatura, se pueden realizar pruebas de funcionamiento para asegurarse que está en buen estado, en condiciones de desempeño máximo y que la conexión al servidor remoto sea satisfactoria y sustentable, para lo cual es necesario contar con un ambiente de pruebas en donde sean certificados los dispositivos para la implementación.

3. Previo a la implementación del monitor de temperatura se puede verificar si los voltajes o corrientes de alimentación a los conversores DC-USB o AC-USB, no exceden el valor máximo de entrada requerido para no causar daños irreparables en los dispositivos electrónicos.

4. Considerar que la arquitectura presentada maneja los principios básicos de confiabilidad, escalabilidad y seguridad para un sistema transaccional por lo que se puede evaluar el tráfico que cursa y contemplar la adaptación de redundancia a nivel de aplicación en el sitio del servidor remoto para cubrir fines específicos de balanceo de carga y disponibilidad de servicios, como contingencia ante fallas.

BIBLIOGRAFÍA

1. ADAFRUIT. *High Temp Waterproof DS18B20 Digital TemperatureSensor* [en línea]. <http://www.adafruit.com/products/642> [Consulta: agosto de 2014].

2. ITEAD STUDIO. *Raspberry Pi SIM900 GSM/GPRS Add-on* [en línea]. <http://imall.iteadstudio.com/raspberry-pi-gsm-add-on.html>. [Consulta: agosto de 2014].

3. LEAL, Eduardo. *Connect Raspberry PI and SIM900* [en línea]. < http://www.raspberrypi.org/forums/viewtopic.php?t=67719&p=494610> [Consulta: julio de 2014].

4. MADRID SALUD. *El transporte de alimentos: un eslabón crucial en la cadena alimentaria* [en línea] <http://www.madridsalud.es/temas/transporte_alimentos.php> [Consulta: julio de 2014].

5. MEMBREY, Peter; HOWS, David. *Learn Raspberry Pi with Linux*. Watkiss, Stewart (rev. tec.). New york: Springer Science+Business Media, 2013. 273 p. ISBN: 978-1-4302-4822-4.

6. MAXIM INTEGRATED. *DS18B20 – HIGH-PRECISION 1-WIRE DIGITAL THERMOMETER* [en línea]. <http://datasheets.maximintegrated.com/en/ds/DS18S20.pdf>. [Consulta: agosto de 2014].

7. SCHMIDT, Maik. *Raspberry Pi A Quick-Start Guide.* Callahan, Ellie (rev. Tec.). 1a ed. Estados Unidos, : Pragmatic Bookshelf. 2012. 30 p. ISBN 978-1-937785-04-8.

8. SILVA LÍMACO, Diego Enrique. *Web Service RESTFul* [en línea]. < http://www.apuntesdejava.com/2010/11/restful-la-forma-mas-ligera-de-hacer.html> [Consulta: septiembre de 2014].

APÉNDICE

I. Tablas de la base de datos.

Dispositivo
id_Dispositivo
MSISDN
fecha_activación

Cliente
id_Cliente
Nombre
Teléfono
e-mail
NIT

DispositivoCliente
id_Cliente
id_Dispositivo
Ubicación
Descripción
id_Servicio

Transc_Dispositivo
id_transacción
id_Dispositivo
fecha_registro
reporte_temperatura

DispositivoConfig
id_configuración
período_reporte
temperatura_transporte
id_Dispositivo

DispositivoEstatus
id_dispositivo
último_reporte
Estado

Fuente: elaboración propia.

II. Objetos SOAP de *Request* y *Response*

```
Response:
<?xml version="1.0" encoding="utf-16"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:enviartrxResponse xmlns:ns2="http://ws.com.monitor/">
      <return>ok</return>
    </ns2:enviartrxResponse>
  </S:Body>
</S:Envelope>

------------------------------------------------------------------------

Request:
<?xml version="1.0" encoding="utf-16"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
 http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <enviartrx xmlns="http://ws.com.monitor/">
      <idDisp xmlns="">1</idDisp>
      <temp xmlns="">-100</temp>
    </enviartrx>
  </soap:Body>
</soap:Envelope>
```

Fuente: elaboración propia, con programa de Netbeans 7.2.

III. Servicio web SOAP: coreInterface

```
package monitor.com.db;

import java.math.BigDecimal;
import java.util.ArrayList;
import javax.sql.DataSource;

public class coreInterface {

    private monitorDB mdb;

    public coreInterface() {
        DataSource roamingds = null;
        String roamingdsName = "monitordb";

        try {
            javax.naming.Context ctx = new javax.naming.InitialContext();
            roamingds = (javax.sql.DataSource) ctx.lookup(roamingdsName);
        } catch (javax.naming.NamingException e) {
            System.out.println("Error: "+ e.getMessage());
        }
        mdb=new monitorDB(roamingds);

    }

    public boolean enviarTrx(int dispositivo,int temp){
     return mdb.enviaTemperatura(dispositivo, temp);
    }
}
```

Fuente: elaboración propia, con programa de Netbeans 7.2.

IV. Servicio web SOAP: wsMonitor

```
package monitor.com.ws;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import monitor.com.db.coreInterface;

@WebService(serviceName = "wsMonitor")
public class wsMonitor {

    private coreInterface manager;
    public wsMonitor(){
    manager=new coreInterface();
    }
    @WebMethod(operationName = "enviartrx")
    public String enviartrx(@WebParam(name = "idDisp") int idDisp,@WebParam(name = "temp")
    int temp) {
        if(manager.enviarTrx(idDisp, temp)){
         return "ok";
        }
        return "err";
    }
}
```

Fuente: elaboración propia, con programa de Netbeans 7.2.

V. Servicio web SOAP: monitorDB

```
package monitor.com.db;

import java.math.BigDecimal;
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.Types;
import java.util.ArrayList;
import javax.sql.DataSource;

public class monitorDB {

    private DataSource ds;

    public monitorDB(DataSource ds) {
        this.ds = ds;
    }

   public boolean enviaTemperatura(int dispositivo,int temp) {
        Connection conn;
        Statement stmt = null;
        ResultSet rs;
        boolean result=true;
        try {
            conn = ds.getConnection();
        } catch (SQLException e) {
            System.out.println("Error: " + e.getMessage());
            return false;
        }
        try {
            stmt = conn.createStatement();
            String query="insert into DISPOSITIVOTRANSACCION (iddispositivo,"+
              "fecharegistro,temperaturareporte)\n" +
                           "values ("+dispositivo+",current_timestamp,"+temp+")";

            stmt.executeUpdate(query);

            try {
                stmt.close();
                conn.commit();
                conn.close();
                return true;
            } catch (SQLException e) {
                System.out.println("Error: " + e.getMessage());
                return false;
            }
```

```
        } catch (SQLException e) {
            System.out.println("Error: " + e.getMessage());
            try {
                conn.close();
            } catch (SQLException e1) {
                System.out.println("Error "+ e1.getMessage());
                return false;
            }
            return false;
        }
    }
}
```

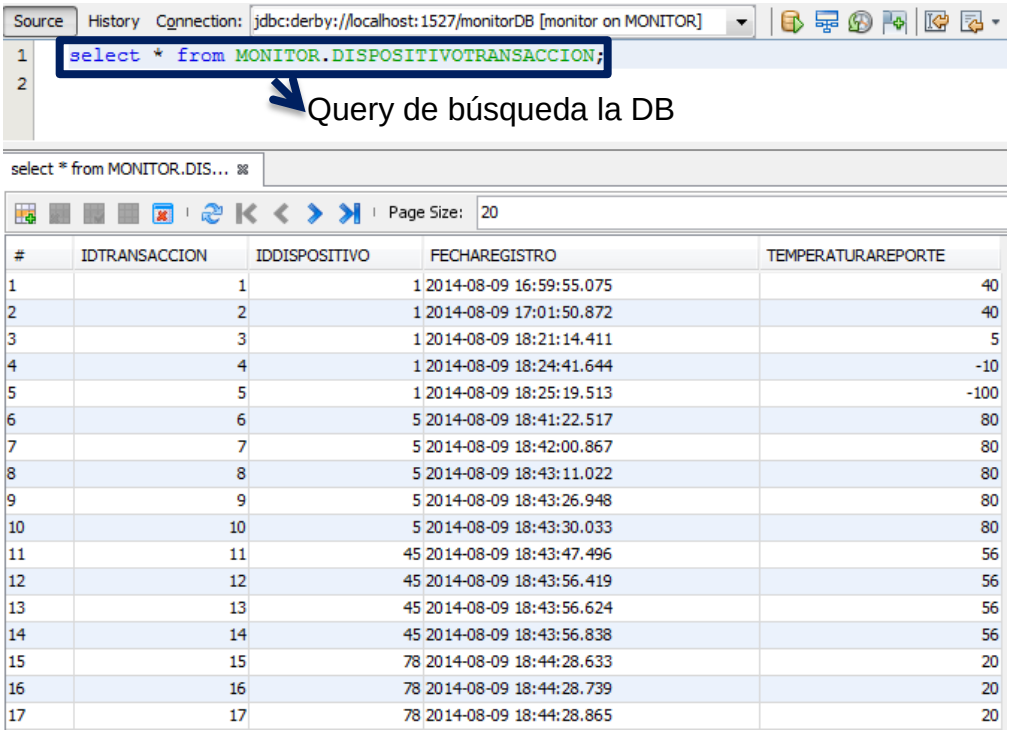
Fuente: elaboración propia, con programa de Netbeans 7.2.

VI. Servicio web RESTFul: wsMonitor

```
package monitor.com.rest;

import javax.ejb.Stateless;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.QueryParam;
import javax.xml.ws.WebServiceRef;
import monitor.com.soap.WsMonitor_Service;

@Stateless
@Path("/monitor")
public class wsMonitor {
    @WebServiceRef(wsdlLocation = "WEB-INF/wsdl/localhost_8080/"+
    "WS_Temperatura_Monitor/wsMonitor.wsdl")
    private WsMonitor_Service service;
    @GET
    public String factorial(@QueryParam("id") int id,@QueryParam("tmp"
    int tmp) {

        return this.enviartrx(id, tmp);
    }

    private String enviartrx(int idDisp, int temp) {
        monitor.com.soap.WsMonitor port = service.getWsMonitorPort();
        return port.enviartrx(idDisp, temp);
    }

}
```

Fuente: elaboración propia, con programa de Netbeans 7.2.

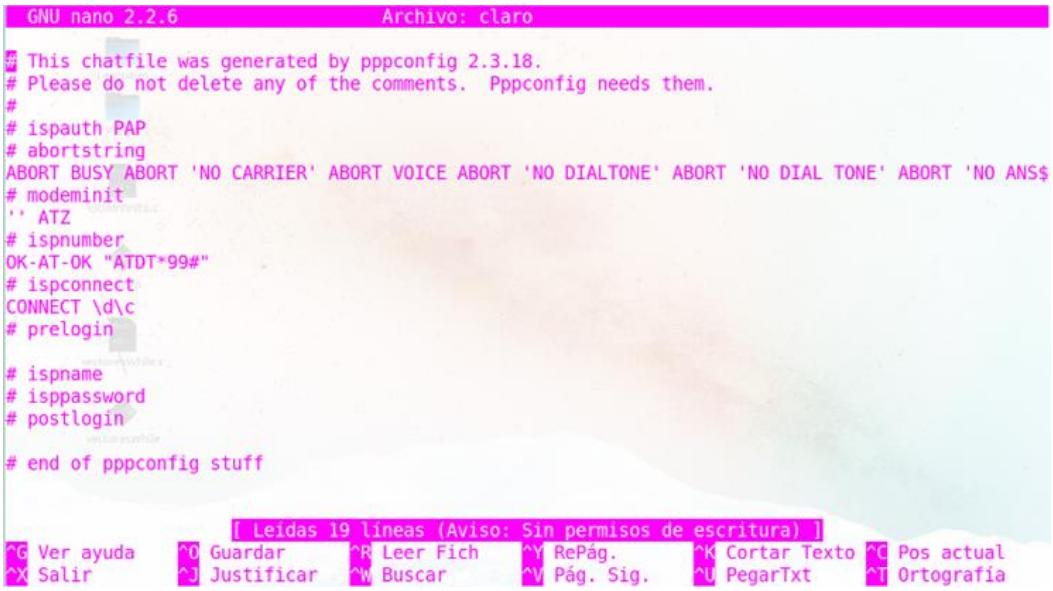
VII. Pruebas de funcionamiento de la DB



#	IDTRANSACCION	IDDISPOSITIVO	FECHAREGISTRO	TEMPERATURAREPORTE
1	1	1	2014-08-09 16:59:55.075	40
2	2	1	2014-08-09 17:01:50.872	40
3	3	1	2014-08-09 18:21:14.411	5
4	4	1	2014-08-09 18:24:41.644	-10
5	5	1	2014-08-09 18:25:19.513	-100
6	6	5	2014-08-09 18:41:22.517	80
7	7	5	2014-08-09 18:42:00.867	80
8	8	5	2014-08-09 18:43:11.022	80
9	9	5	2014-08-09 18:43:26.948	80
10	10	5	2014-08-09 18:43:30.033	80
11	11	45	2014-08-09 18:43:47.496	56
12	12	45	2014-08-09 18:43:56.419	56
13	13	45	2014-08-09 18:43:56.624	56
14	14	45	2014-08-09 18:43:56.838	56
15	15	78	2014-08-09 18:44:28.633	20
16	16	78	2014-08-09 18:44:28.739	20
17	17	78	2014-08-09 18:44:28.865	20

Fuente: elaboración propia, con programa de Sun Java DB.

VIII. *Script* de configuración generado por PPPConfig



Fuente: elaboración propia, con programa de PPPConfig.

IX. *Script* de configuración generado por PPPConfig

Fuente:elaboración propia, con programa de PPPConfig.

X. Pruebas del sensor de temperatura funcionando

Fuente: elaboración propia, con programa de PuTYY.

XI. Código de Python: programa principal en Raspberry Pi

```
import os
import glob
import time
import urllib2, httplib

def leerID(fileName='id.txt'):
    arch = open(fileName,'r')
    lines = arch.readlines()
    f.close()
    return lines[0]

def read_temp_raw(sensor):
    device_file = base_dir + '/' + devices[sensor][:-1] + '/w1_slave
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

def read_temp(sensor):
    lines = read_temp_raw(sensor)
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw(sensor)

    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        return temp_string[0:4]

def conexion():
    os.system('pon claro')

os.system('sudo modprobe wire')
os.system('sudo modprobe w1-gpio')
os.system('sudo modprobe w1-therm')
```

Continuación del apéndice XI.

```
base_dir = '/sys/bus/w1/devices'
f = open(base_dir + '/w1_bus_master1/w1_master_slave_count', 'r');
sensorCount = f.readlines()
sensorCount = [int(l[0]) for l in sensorCount]
f.close()
f = open(base_dir + '/w1_bus_master1/w1_master_slaves', 'r');
devices = f.readlines()
f.close()

while True:
    url = "230.245.147.95/monitorRest/webresources/monitor?id="
    + leerID + "&tmp=" read_temp(sensor)

urllib2.urlopen(url)
time.sleep(10)
```

Fuente: elaboración propia, con programa de Python.