UNIVERSIDAD NACIONAL DE CATAMARCA
FACULTAD DE TECNOLOGÍA Y
CIENCIAS APLICADAS

INGENIERÍA EN INFORMÁTICA

TRABAJO FINAL

Sistema Embebido de Telemetría para la Dirección de Tecnologías de la Información y Comunicación de la Universidad Nacional de Catamarca

AUTOR:
Morti, Ricardo Maximiliano

DIRECTOR**:**
Ms. Ing. Aranda, Marcos Darío

Catamarca, Febrero 2019

Agradecimientos

A mis padres, por todo su sacrificio para darme una buena educación que me permitió tener la maravillosa posibilidad de estudiar una carrera de grado.

A mis familiares, amigos y compañeros, por estar ahí cuando los necesité.

A mi director de tesis, Ms. Ing. Marcos Darío Aranda, por su apoyo y apropiada guía durante el proyecto.

Índice/Tabla de Contenidos

Índice de Figuras

Índice de Tablas

Resumen

El Presente trabajo final tuvo como objetivo principal, el desarrollo de un Sistema Embebido que será utilizado en el data center (sala de servidores) de la Dirección de Tecnologías de la Información y Comunicación de la Universidad Nacional de Catamarca con el fin de monitorear la temperatura, humedad y el funcionamiento de los equipos de aire acondicionado a distancia.

El Proyecto consistió en el desarrollo del Sistema Embebido de Telemetría, utilizando como base la plataforma de hardware Intel Galileo, esta plataforma, permitió la conexión y el manejo de los componentes como un sensor de temperatura y humedad, un sensor de corriente, y también admite compatibilidad con plataformas del tipo modulo y modulo escudo, que son placas pre fabricadas que facilitan su conexión y uso con otras plataformas, en este proyecto, se usó un módulo tipo escudo GSM/SMS y un módulo con bobina de relé.

El sistema permitirá la medición y almacenamiento de los datos recolectados por los sensores y la consulta de estos datos a través de mensajes de texto SMS, como la plataforma Intel Galileo, posee un sistema operativo embebido en base Linux, permitió hacer uso de herramientas de internet como un servidor web, y brindar al personal del data center la opción de una aplicación web para consultar los datos recolectados a través de internet.

Por otro lado, se agregó al sistema la capacidad de analizar los datos obtenidos para evaluar la situación del data center, y emitir alertas si detecta valores no deseados, por ejemplo, si la temperatura es muy alta, o si un equipo de aire acondicionado dejara de funcionar, estas situaciones serán informadas al personal.

Así, se logró brindar una forma automática y confiable de medir datos con telemetría, y proveer un dispositivo que permite mantener informado al personal en todo momento sobre el estado de la sala de servidores para tomar una rápida acción si se presenta una emergencia.

CAPITULO 1

1 Introducción

En la actualidad, gracias a los grandes avances de las tecnologías informáticas y de comunicación, el manejo de información es de vital importancia en el correcto funcionamiento de organismos tanto públicos como privados, para asegurar que los datos utilizados estén seguros y protegidos, lo primordial, es mantener la integridad física de los dispositivos que hacen uso de esa información.

Es por esto que la seguridad siempre ha sido (y será) una de las prioridades más altas a cumplir para mantener el buen y correcto funcionamiento del equipo informático de cualquier establecimiento, y no es para menos, estos equipos tienen almacenadas grandes cantidades de información, que como ya se ha mencionado, los organismos, dependen mucho del manejo eficiente de la misma.

La telemetría es una tecnología que permite la medición remota de magnitudes físicas y el posterior envío de la información hacia el operador del sistema y si dicha información está relacionada con datos necesarios para mantener un control de la integridad del equipamiento informático, el personal a cargo del área, estaría al tanto de cualquier irregularidad o percance que pudiera presentarse, actuando rápidamente acorde a la situación, disminuyendo la probabilidad de perdida de información o daños en el hardware. Esto no solo significaría que la infraestructura en general estará más protegida, también facilitará la labor del personal encargado, brindando la seguridad y tranquilidad que la organización busca en lo respecto a la protección de la información que almacenan.

Entonces, se exploró el concepto de Internet de las cosas (Internet of things, abreviado IoT) que se refiere a la interconexión digital de objetos cotidianos con Internet [1]. Alternativamente, Internet de las cosas es la conexión de Internet con más “cosas u objetos” que personas. También se suele conocer como Internet de todas las cosas o Internet en las cosas.
Siguiendo esta ideología, es posible conseguir soluciones a problemas de comunicación, mediante la conexión a internet de los objetos que deseamos conocer su estado actual.
Con respecto al Hardware del proyecto, se utilizó la tecnología Intel Galileo, que es la primera placa basada en arquitectura Intel diseñada para ser compatible tanto en software como en hardware con la tecnología Arduino, la cual se enfoca en acercar y facilitar el uso de la electrónica y programación de sistemas embebidos en proyectos multidisciplinarios**.**

En este proyecto de Trabajo Final, se aborda el desarrollo de un sistema de Telemetría utilizando sensores y módulos ensamblados en una placa Intel Galileo, que realizan un monitoreo constante en el área donde se encuentren alojados los equipos de infraestructura informática a cargo de la D.T.I.C[1], tales como servidores, racks, switches, routers y cualquier otro elemento que se deseé tener bajo supervisión. Los datos que serán recolectados por

[1] Dirección de Tecnologías de la Información y Comunicación

los sensores, son almacenados y podrán ser consultados en tiempo real por el personal encargado a través de un servicio web.
De esta manera, podemos tener lecturas tales como, la temperatura, humedad de la sala, tensión eléctrica y estado de funcionamiento de equipos de aire acondicionado en tiempo real, y poder establecer acciones correctivas y preventivas si estos valores se salen de la norma a fin de proteger los equipos y la infraestructura montada.

Para una mayor comprensión del desarrollo llevado a cabo, se dividió este trabajo en capítulos, los cuales se describen a continuación:

- Capítulo 1, “Introducción”: Se presenta el trabajo propuesto, el problema planteado, la justificación y las aspiraciones.
- Capítulo 2, “Marco Teórico”: Se introducen los sustentos teóricos sobre las tecnologías y herramientas utilizadas en el desarrollo.
- Capítulo 3, “Desarrollo del hardware del Sistema embebido de Telemetría”: Se describe el proceso de desarrollo del hardware empleado para el dispositivo de telemetría.
- Capítulo 4, “Desarrollo del Software del Sistema embebido de Telemetría”: Se describe el proceso de desarrollo del software instalado en el Sistema de Telemetría, su firmware y pagina web.
- Capítulo 5, “Fase de pruebas del Sistema”: Se describe el proceso de pruebas de funcionamiento del sistema previo a la implementación.
- Capítulo 6, “Resultados Alcanzados”: Se describen los resultados alcanzados con el desarrollo de este trabajo final.
- Capítulo 7: “Conclusiones”: Se describen las conclusiones del presente trabajo, poniendo en manifiesto si se cumplieron los objetivos iniciales.

1.1 Planteamiento del Problema

Uno de los problemas más comunes en la seguridad informática, más específicamente, en los data center (Sala de servidores de la D.T.I.C. donde se encuentra el hardware), es que debe estar operativo las 24 horas del día, y no es posible contar siempre con la supervisión humana para garantizar que ante cualquier contingencia, habrá alguien para solucionarla, y es por razones claras: las personas deben volver a sus hogares a descansar.

Debido a esto, habrá momentos donde el Data Center, no tendrá supervisión inmediata del personal, tales como: horario nocturno, vacaciones, días no laborales (feriados, fines de semana, asuetos), esta ausencia inevitable del personal supone correr un riesgo ante posibles problemas críticos que pudieran presentarse, y si la empresa u organismo interesado, necesita un control permanente de sus instalaciones, se debe afrontar un gasto adicional en empleados que estén disponibles todo el día.

Estas cuestiones nos llevan al problema que cuando el personal no está presente en el lugar por los motivos mencionados arriba, no se tiene conocimiento de la situación actual del Data Center.

Se busca que el Sistema de Telemetría, brinde información constante, actualizada y accesible desde cualquier lugar sobre el estado actual de las instalaciones, también emitirá

alertas vía mensajes de texto SMS, de esta forma, ante un evento de carácter urgente, el personal podrá actuar apropiadamente.

1.2 Justificación

En Catamarca, la Seguridad Informática tanto del software como la del hardware está comenzando a ser aceptada como área importante en los organismos públicos y privados, es por eso que el desarrollo de este Proyecto será para el beneficio de la D.T.I.C., brindando la posibilidad al personal a cargo, de conocer en tiempo real datos sobre el estado actual del data center, como la temperatura, humedad de la sala, tensión eléctrica y el funcionamiento de los equipos de aire acondicionado, de esta manera, podrán actuar de manera rápida ante cualquier problema que pueda presentarse en el lugar.

1.3 Alcance

El Presente Trabajo Final estará enfocado a:

Brindar apoyo y soporte en las tareas de mantenimiento, supervisión y manejo del Data Center a cargo de la D.T.I.C.

Procesamiento de la información recolectada por el sistema de telemetría.

1.4 Objetivos

1.4.1 Objetivo General

- Desarrollar e implementar un Sistema Embebido de Telemetría, el cual tendrá sensores que medirán la temperatura, humedad, y funcionamiento de equipos de Aire Acondicionado (mediante consumo de corriente), en el data center a cargo de la D.T.I.C, los datos recolectados por los sensores podrán ser consultados por el personal de la D.T.I.C mediante mensaje de texto o un servicio web, también, el sistema emitirá alertas si se detectan valores no deseados, y se podrá hacer funcionar un equipo de aire acondicionado de emergencia si es necesario.

1.4.2 Objetivos Específicos

- Hacer un relevamiento de los datos que se necesitan medir en el Data Center.
- Identificar los componentes de Hardware más adecuados para realizar el monitoreo de datos.
- Implementar en la placa Intel Galileo, los sensores que se va a utilizar para medir los datos solicitados.
- Implementar en la placa Intel Galileo el módulo GSM/GPRS que permitirá el envío de alertas vía SMS.
- Implementar el módulo de relé/relay que accionara un equipo de aire acondicionado de emergencia de ser necesario, ante falla de funcionamiento del equipo principal.

- Implementar el código del sistema de telemetría utilizando un lenguaje de programación C++ y el entorno de desarrollo.
- Realizar Análisis, diseño, desarrollo y prueba del Sistema Embebido de Telemetría.
- Utilizar los conocimientos adquiridos en HTML, PHP, phpMyAdmin para recolectar los datos obtenidos de los sensores que serán utilizados en un Servicio Web.
- Utilizar los conocimientos adquiridos en HTML, PHP para desarrollar el servicio Web, donde se podrán consultar los datos recolectados.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

CAPITULO 2

2 Marco Teórico

Los Conceptos teóricos que usa como soporte este Trabajo Final serán citados a continuación:

2.1 Internet de la Cosas

Desde la llegada de Internet a nuestras vidas, éste ha evolucionado de una forma veloz y apabullante, desde aquellos rudimentarios módems de 56 KB hasta las rapidísimas y eficaces líneas de fibra óptica actuales. A día de hoy podemos conectar a Internet nuestros móviles, impresoras, Smart TV, cámaras vía IP, GPS y multitud de dispositivos electrónicos que poseen esta funcionalidad.

Desde hace ya unos años se empezó a hablar del **IoT** (Internet de las cosas), no siendo más que una apuesta de futuro y comenzando a ser en la actualidad una absoluta realidad.

Cada vez es más frecuente encontrarse con nuevos dispositivos capaces de conectarse a Internet y permitir al usuario un control y manejo de forma remota desde cualquier parte del mundo, pero esto no ha hecho más que comenzar.

2.1.1 Definición

Podríamos definir el Internet de las cosas como la consolidación a través de la red de redes de una "red" que alojase una gran multitud de objetos o dispositivos, es decir, poder tener conectada a esta todas las cosas de este mundo como podrían ser vehículos, electrodomésticos, dispositivos mecánicos, o simplemente objetos tales como calzado, muebles, maletas, dispositivos de medición, biosensores, o cualquier objeto que nos podamos imaginar.

Como la gran mayoría de los avances tecnológicos el objetivo que pretende alcanzar esta tecnología es hacer más cómodas nuestras vidas así como proporcionar una mayor seguridad en diversos ámbitos. Pongamos un ejemplo: Imaginemos una nevera que fuese capaz de avisarnos cuando perdiese temperatura o indicarnos que algún alimento ha caducado o se ha pasado, o un escritorio que dejará constancia de donde se ha dejado cada cosa, o que pudiésemos saber dónde se encuentra cada objeto, que nos pertenezca, en cada momento, o controlar toda nuestra vivienda desde un smartphone o PC (desde la puerta de entrada, hasta la cadena de WC), o que estemos avisados de los alérgenos y su concentración en el aire en una pulserita para alérgicos, o millones de escenarios por descubrir. Estos son algunos ejemplos de todo lo que podría permitirnos desempeñar el Internet de las cosas.

A diferencia de algunas tecnologías mucho más populares entre las masas, el Internet de las cosas no ha encontrado su foco de explosión en el mercado del consumo. Quizás la tecnología está aún demasiado verde, o quizás los grandes del sector no han visto la oportunidad correcta para abalanzarse encima. Aun así hemos visto como Apple y Google han dado algunos pasos discretos con tecnologías como Home Kit y Android @Home. [1]

Es en el Sector privado donde el Internet de las Cosas se está haciendo cada vez más popular, estas son algunas de las áreas donde se aplica:

- **La industria de producción en masa**: la maquinaria que se encarga de controlar los procesos de fabricación, robots ensambladores, sensores de temperatura, control de producción, todo está conectado al Internet en cada vez más empresas lo que permite centralizar el control de la infraestructura.
- **Control de infraestructura urbana**: control de semáforos, puentes, vías de tren, cámaras urbanas. Cada vez más ciudades implementan este tipo de infraestructuras basadas en el Internet de las Cosas que permiten monitorear el correcto funcionamiento de sus estructuras además de adaptar más flexiblemente su funcionamiento ante nuevos eventos.
- **Control ambiental**: una de las áreas en las que está teniendo más éxito el Internet de las cosas, pues permite acceder desde prácticamente cualquier parte a información de sensores atmosféricos, meteorológicos, y sísmicos.
- **Sector salud**: cada vez más clínicas y hospitales alrededor del mundo confían en sistemas que les permiten al personal de salud monitorear activamente a los pacientes de manera ambulatoria y no invasiva.

2.2 Telemetría

La **telemetría** es una tecnología que permite la medición remota de magnitudes físicas y el posterior envío de la información hacia el operador del sistema.

Un sistema de telemetría funciona normalmente mediante comunicación inalámbrica pero también se puede realizar a través de otros medios como: teléfono, redes de ordenadores, enlace de fibra óptica, entre otros. Es utilizada en áreas muy diversas que va desde el automovilismo, aviación, astrología, pasando por la agricultura, industria de petróleo, medicina y hasta biología.

La telemetría tiene como objetivo permitir la medición de magnitudes físicas o químicas, conocer los estados de los procesos y del sistema, así como controlar de manera remota el funcionamiento, corregir los errores y enviar la información recabada hacia un sistema de información para su uso y provecho. [2][3]

2.3 C++

C++ es un lenguaje de programación diseñado en 1979 por Bjarne Stroustrup. La intención de su creación fue extender al lenguaje de programación C. Posteriormente se añadieron facilidades de programación genérica, incluidas en otros lenguajes pero que C no soportaba, al menos directamente, como son las llamadas clases y objetos, principios usados en la programación actual.

Para ello, rediseñó C, ampliando sus posibilidades pero manteniendo su mayor cualidad, la de permitir al programador en todo momento tener controlado lo que está haciendo, consiguiendo así una mayor rapidez que no se conseguiría en otros lenguajes.

C++ pretende llevar a C a un nuevo paradigma de clases y objetos con los que se realiza una comprensión más humana basándose en la construcción de objetos, con características

propias solo de ellos, agrupados en clases. Es decir, si se quisiera hacer un programa sobre animales, se crearía una clase llamada animales, en la cual cada animal, por ejemplo un pato, sería un objeto, de tal manera que se ve el intento de esta forma de programar por ser un fiel reflejo de cómo los humanos (en teoría) manejamos la realidad

Actualmente existe un estándar, denominado ISO C++, al que se han adherido la mayoría de los fabricantes de compiladores más modernos.

Una particularidad del C++ es la posibilidad de redefinir los operadores, y de poder crear nuevos tipos que se comporten como tipos fundamentales. [19]

El nombre "C++" fue propuesto por Rick Mascitti en el año 1983, cuando el lenguaje fue utilizado por primera vez fuera de un laboratorio científico. Antes se había usado el nombre "C con clases". En C++, la expresión "C++" significa "incremento de C" y se refiere a que C++ es una extensión de C. [20]

2.3.1 Ventajas y Desventajas

A continuación, se listan las ventajas y desventajas de utilizar C++ como lenguaje de programación.

Ventajas

- Lenguaje de programación orientado a objetos.
- Es muy potente en lo que se refiere a creación de sistemas complejos, un lenguaje muy robusto.
- Actualmente, puede compilar y ejecutar código de C, ya viene con librerías para realizar esta labor.
- Es un lenguaje muy flexible que permite programar con múltiples estilos.
- Existen compiladores de C++ para diferentes sistemas operativos, lo cual representa una ventaja en cuestión de portabilidad. Destacando compatibilidad con la plataforma Arduino.

Desventajas

- Elaborar un sistema en C++ es como construir un rascacielos: tiene buen soporte y es robusto, pero si existen errores en los pisos inferiores toda la parte superior se viene abajo terriblemente.
- Uso de DLLs (librerías dinámicas) muy complejo.
- Carece de instrucciones de entrada/salida, de instrucciones para manejo de cadenas de caracteres, con lo que este trabajo queda para la librería de rutinas, con la consiguiente pérdida de transportabilidad.

Se decidió el uso de este lenguaje sobre otros gracias a su amplio repertorio de librerías y compatibilidad con Arduino contando con un IDE[2] basado en C++ que brinda todas las herramientas necesarias para trabajar con la plataforma Intel Galileo.

[2] Integrated Development Environment

2.4 MySQL

Es un sistema de administración de bases de datos (por sus siglas en ingles DBMS, Database Management System) para bases de datos relacionales, fue creada por la empresa sueca MySQL AB. MySQL.

Existen muchos tipos de bases de datos, desde un simple archivo hasta sistemas relacionales orientados a objetos. Como base de datos relacional, utiliza múltiples tablas para almacenar y organizar la información. MySQL fue escrito en C y C++ y destaca por su gran adaptación a diferentes entornos de desarrollo, permitiendo su interactuación con los lenguajes de programación más utilizados como PHP, Perl y Java, así como su integración en distintos sistemas operativos.

También es muy destacable, la condición de open source de MySQL, que hace que su utilización sea gratuita e incluso se pueda modificar con total libertad, pudiendo descargar su código fuente. Esto ha favorecido muy positivamente en su desarrollo y continuas actualizaciones, para hacer de MySQL una de las herramientas más utilizadas por los programadores orientados a Internet. [4]

2.4.1 Características

En un principio, MySQL no poseía elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, despertó el interés en los desarrolladores de páginas web con contenido dinámico, justamente por su simplicidad.

Poco a poco los elementos de los que carecía MySQL se fueron incorporando tanto por desarrollos internos, como por desarrolladores de software libre.

Los beneficios y desventajas de utilizar MySQL son:

Ventajas

- MySQL es Open Source.
- Velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento.
- Bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Facilidad de configuración e instalación.
  Soporta gran variedad de Sistemas Operativos
- Baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que está.
- Su conectividad, velocidad, y seguridad hacen de MySQL Server altamente apropiado para acceder bases de datos en Internet

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Desventajas

- Al ser de Software Libre, muchas de las soluciones para las deficiencias del software no están documentadas ni presentan documentación oficial.
- Muchas de sus utilidades tampoco presentan documentación.

2.4.2 SQL

SQL (por sus siglas en inglés Structured Query Language; en español lenguaje de consulta estructurada) es un lenguaje específico del dominio utilizado en programación, diseñado para administrar, y recuperar información de sistemas de gestión de bases de datos relacionales. Una de sus principales características es el manejo del álgebra y el cálculo relacional para efectuar consultas con el fin de recuperar, de forma sencilla, información de bases de datos, así como realizar cambios en ellas.

Originalmente basado en el álgebra relacional y en el cálculo relacional, SQL consiste en un lenguaje de definición de datos, un lenguaje de manipulación de datos y un lenguaje de control de datos. El alcance de SQL incluye la inserción de datos, consultas, actualizaciones y borrado, la creación y modificación de esquemas y el control de acceso a los datos. También el SQL a veces se describe como un lenguaje declarativo, también incluye elementos procesales. [22]

Características Generales de SQL

SQL es un lenguaje de acceso a bases de datos que explota la flexibilidad y potencia de los sistemas relacionales y permite así gran variedad de operaciones.

Es un lenguaje declarativo de "alto nivel" o "de no procedimiento" que, gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros —y no a registros individuales— permite una alta productividad en codificación y la orientación a objetos. [23]

De esta forma, una sola sentencia puede equivaler a uno o más programas que se utilizarían en un lenguaje de bajo nivel orientado a registros. SQL también tiene las siguientes características:

- **Lenguaje de definición de datos**: El LDD de SQL proporciona comandos para la definición de esquemas de relación, borrado de relaciones y modificaciones de los esquemas de relación.
- **Lenguaje interactivo de manipulación de datos**: El LMD de SQL incluye lenguajes de consultas basado tanto en álgebra relacional como en cálculo relacional de tuplas.
- **Integridad:** El LDD de SQL incluye comandos para especificar las restricciones de integridad que deben cumplir los datos almacenados en la base de datos.
- **Definición de vistas**: El LDD incluye comandos para definir las vistas.
- **Control de transacciones**: SQL tiene comandos para especificar el comienzo y el final de una transacción.
- **SQL incorporado y dinámico**: Esto quiere decir que se pueden incorporar instrucciones de SQL en lenguajes de programación como: C++, C, Java, PHP, Cobol, Pascal y Fortran.

- **Autorización**: El LDD incluye comandos denominados de Control LCD para especificar los derechos de acceso a las relaciones y a las vistas.

2.5 HTML

HTML, sigla en inglés de **HyperText Markup Language** (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros. Es un estándar a cargo del World Wide Web Consortium (W3C) o Consorcio WWW, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación.

Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la World Wide Web (WWW). Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado. [21]

El lenguaje HTML basa su filosofía de desarrollo en la diferenciación. Para añadir un elemento externo a la página (imagen, vídeo, script, entre otros.), este no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene solamente texto mientras que recae en el navegador web (interpretador del código) la tarea de unir todos los elementos y visualizar la página final. Al ser un estándar, HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma (estándar) por cualquier navegador web actualizado.

2.5.1 Estructura del Documento HTML

Los documentos escritos en HTML constan del texto mismo del documento y las **tags** o **etiquetas** que le dan formato. Por ejemplo: *<etiqueta> texto del documento </etiqueta>*. La etiqueta del principio activa la instrucción, y la última (que será la del principio precedida del signo /) la desactiva. Como HTML no sigue fielmente los estándares del SGML (lenguaje de marcado generalizado estándar), no todas las etiquetas tienen principio y final. [24]

Las etiquetas que describen la estructura general de un documento y dan una información sencilla sobre él son: **<HTML> <HEAD> <TITLE> <BODY>**. Estas etiquetas no afectan a la apariencia del documento y solo interpretan y filtran los archivos HTML

- **<HTML>:** Limitan el documento e indica que se encuentra escrito en este lenguaje.
- **<HEAD>:** Especifica el prólogo del resto del archivo. Son pocas las etiquetas que van dentro de ella, destacando la del título.
- **<TITLE>:** Es utilizada por los marcadores del navegador para identificar el contenido de la página. Solo puede haber un título por documento, preferiblemente corto aunque significativo, y no caben otras etiquetas dentro de ella. En head no hay que colocar nada del texto del documento.
- **<BODY>:** Encierra el cuerpo principal del documento, el contenido.

La estructura y características de la página, como tipo de letra, color, colores de fondo, etc. se hace definiendo tales características en código CSS.

2.6 PHP

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML, Ejemplo de ¡Hola Mundo¡ en PHP embebido en código HTML:

```
<!DOCTYPE HTML>
<html>
   <head>
      <title>Ejemplo</title>
   </head>
   <body>
      <?php
         echo "¡Hola, soy un script de PHP!";
      ?>
   </body>
</html>
```

En lugar de usar muchos comandos para mostrar HTML (como en C o en Perl), las páginas de PHP contienen HTML con código incrustado que hace "algo" (en este caso, mostrar "¡Hola, soy un script de PHP!). El código de PHP está encerrado entre las etiquetas especiales de comienzo y final **<?php** y **?>** que permiten entrar y salir del "modo PHP".

Lo que distingue a PHP de algo del lado del cliente, como Javascript, es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá el resultado de ejecutar el script, aunque no se sabrá el código subyacente que era. El servidor web puede ser configurado incluso para que procese todos los ficheros HTML con PHP, por lo que no hay manera de que los usuarios puedan saber qué se tiene debajo de la manga.

Lo mejor de utilizar PHP es su extrema simplicidad para el principiante, pero a su vez ofrece muchas características avanzadas para los programadores profesionales.

Aunque el desarrollo de PHP está centrado en la programación de scripts del lado del servidor, se puede utilizar para muchas otras cosas. [26]

2.6.1 Sintaxis

Las variables se prefijan con el símbolo del dólar **$** y no es necesario indicar su tipo. Las variables, a diferencia de las funciones, distinguen entre mayúsculas y minúsculas. Las cadenas de caracteres pueden ser encapsuladas tanto en dobles comillas como en comillas simples, aunque en el caso de las primeras, se pueden insertar variables en la cadena directamente, sin necesidad de concatenación.

Los comentarios se pueden escribir bien con dos barras al principio de la línea, o con una almohadilla. También permite comentarios multi-línea encapsulados en **/* */**.

En cuanto a las palabras clave, PHP comparte con la mayoría de otros lenguajes con sintaxis C las condiciones con **if**, los bucles con **for** y **while** y los retornos de funciones. Como es habitual en este tipo de lenguajes, las sentencias deben acabar con punto y coma **(;)**.

2.6.2 Características

PHP como todo lenguaje de programación, posee características que bridan beneficios por sobre otros, y también desventajas. [26]

Beneficios

- Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- Es considerado un lenguaje fácil de aprender, ya que en su desarrollo se simplificaron distintas especificaciones, como es el caso de la definición de las variables primitivas, ejemplo que se hace evidente en el uso de php arrays.
- El código fuente escrito en PHP es invisible al navegador web y al cliente, ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con **MySQL** y **PostgreSQL.**
- Capacidad de expandir su potencial utilizando módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su sitio web oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Tiene manejo de excepciones (desde PHP5).
- Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar, aun haciéndolo, el programador puede aplicar en su trabajo cualquier técnica de programación o de desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del patrón de diseño Modelo Vista Controlador (MVC), que permiten separar el tratamiento y acceso a los datos, la lógica de control y la interfaz de usuario en tres componentes independientes.
- Debido a su flexibilidad ha tenido una gran acogida como lenguaje base para las aplicaciones WEB de manejo de contenido, y es su uso principal.

Desventajas

- Como es un lenguaje que se interpreta en ejecución, para ciertos usos puede resultar un inconveniente que el código fuente no pueda ser ocultado. La ofuscación

es una técnica que puede dificultar la lectura del código pero no necesariamente impide que el código sea examinado.

- Debido a que es un lenguaje interpretado, un script en PHP suele funcionar considerablemente más lento que su equivalente en un lenguaje de bajo nivel, sin embargo este inconveniente se puede minimizar con técnicas de caché tanto en archivos como en memoria.
- En las versiones previas a la 7, las variables no son tipificadas, lo cual dificulta a los diferentes IDEs ofrecer asistencias para el tipificado del código, aunque esto no es realmente un inconveniente del lenguaje en sí. Esto es solventado por algunos IDEs añadiendo un comentario con el tipo a la declaración de la variable.

2.7 Sistema Embebido

Un sistema embebido se puede definir como una combinación de hardware y firmware (software) diseñado para realizar una función específica [5]. En algunos casos estos sistemas embebidos son componentes dentro de un sistema de mayor escala.

Cada sistema embebido es único, y el hardware como así también el firmware son altamente especializados para el dominio de la aplicación. Los sistemas embebidos se han convertido en una parte inevitable de cualquier producto o equipamiento dentro de varios dominios como son las telecomunicaciones, la industria automotriz, equipamientos médicos, artículos del hogar, y muchos más.

Las tecnologías embebidas están vinculadas a nuestras tareas diarias y en muchos casos somos inconscientes o ignorantes del poder, el confort o la seguridad que los sistemas embebidos nos brindan [6].

El diseño de un sistema embebido para realizar una tarea específica está en contraste directo con una computadora personal. A pesar de que ambos sistemas están compuestos por hardware, software y componentes mecánicos, una computadora personal no está diseñada para realizar una tarea específica, sino que es capaz de realizar muchas tareas diferentes. Generalmente se utiliza el término “computadora de propósito general” para mantener esta diferencia clara.[7]

A continuación se muestra en la tabla 2-1 una breve comparación entre los sistemas embebidos y los sistemas de propósito general:

Sistemas de Propósito General	Sistemas Embebidos
Combinación de hardware genérico y un Sistema Operativo de Propósito General para ejecutar una variedad de aplicaciones.	Combinación de hardware de propósito específico y un Sistema Operativo Embebido para ejecutar un conjunto especifico de aplicaciones.
Contiene un Sistema Operativo de Propósito General.	Puede o no contener un Sistema Operativo para su funcionamiento.
Aplicaciones alterables (programables) por el usuario, con la posibilidad de reinstalar el sistema operativo y también sumar o quitar	El firmware es pre-programado y es inalterable por el usuario final. Existen excepciones para sistemas que soportan

aplicaciones	sistemas operativos.
Performance es la clave para la selección del sistema.	Los requerimientos específicos de la aplicación (como performance, consumo de energía, uso de memoria, etc.) son los factores claves de decisión.

Tabla 2-1 - Comparación entre Sistemas Embebidos y Sistemas de Propósito General

2.7.1 Arquitectura

La arquitectura de un sistema embebido es una abstracción del dispositivo embebido, es decir es una generalización del sistema sin mostrar información detallada de su implementación como código fuente o diseño de circuitos de hardware. A nivel arquitectura, los componentes de hardware y software son representados como la composición de elementos que interactúan.

La arquitectura de un sistema embebido se puede usar para resolver varios desafíos que se presentan durante el diseño de un nuevo sistema. Entre ellos podemos encontrar:

- Definición y captura del diseño del sistema
- Limitación de costos
- Determinar la integridad del sistema

Sin definir o conocer ningún detalle interno de implementación, la arquitectura de un sistema embebido se presenta como la primera herramienta para ser analizada y usada como un plano que define las opciones, infraestructura y restricciones del diseño.

Una forma de introducir los elementos de alto nivel en el diseño de sistemas embebidos es mediante un Modelo de Sistemas Embebidos, en la Figura 2-1, se muestra un diagrama representativo de este Modelo.

Figura 2-1 - Modelo de Sistemas Embebidos
[Fuente: Embedded Systems Arquitecture - A Compresive Guide for Engineers and Programmers]

El modelo es una representación en capas (modular) de la arquitectura de sistema el cual permite visualizar y agrupar los componentes del sistema como capas. El modelo en capas permite observar las posibles combinaciones de componentes de hardware y software que se pueden utilizar para diseñar un sistema embebido. [7][9]

2.7.2 Componentes

Por definición, todos los sistemas embebidos contienen un procesador y software. Con el fin de almacenar ese software, debe existir un lugar donde pueda ser alojado el código ejecutable y almacenamiento temporario para la manipulación de datos durante la ejecución.

Según la definición de sistema embebido, el hardware y software son los dos componentes principales de los sistemas embebidos.

- **Hardware**
  Como otros sistemas electrónicos, un sistema embebido requiere una plataforma de hardware sobre la cual correr. El hardware será basado en un microprocesador o microcontrolador. El hardware del sistema embebido también contendrá otros elementos como memoria e interfaces de entrada/salida.
- **Software**
  El software del sistema embebido es escrito para realizar una función particular. Generalmente es escrito en formato de alto nivel y luego compilado para proveer un código que pueda ser cargado en una memoria no volátil en el hardware.

En un sistema embebido todos los componentes de hardware residen sobre una placa, también llamada Placa de Circuitos Impresa (PCB). Todo el hardware en una placa embebida se encuentra alojado en la capa de hardware del Modelo de Sistemas Embebidos.

Los componentes de hardware que constituyen la mayoría de las placas pueden ser clasificados dentro de cinco categorías:

- Unidad de Procesamiento Central (CPU)
- Memoria
- Dispositivos de Entrada
- Dispositivos de Salida
- Buses

Estas cinco categorías están basadas en los elementos definidos por el modelo de Von Neumann, una herramienta que se puede utilizar para la arquitectura de hardware de cualquier dispositivo electrónico. El modelo de Von Neumann es el resultado de un trabajo publicado en 1945, el cual define los requerimientos de una computadora de propósito general. Debido a que los sistemas embebidos son un tipo de sistema de computación, este modelo puede ser aplicado para entender el hardware del sistema embebido [7].

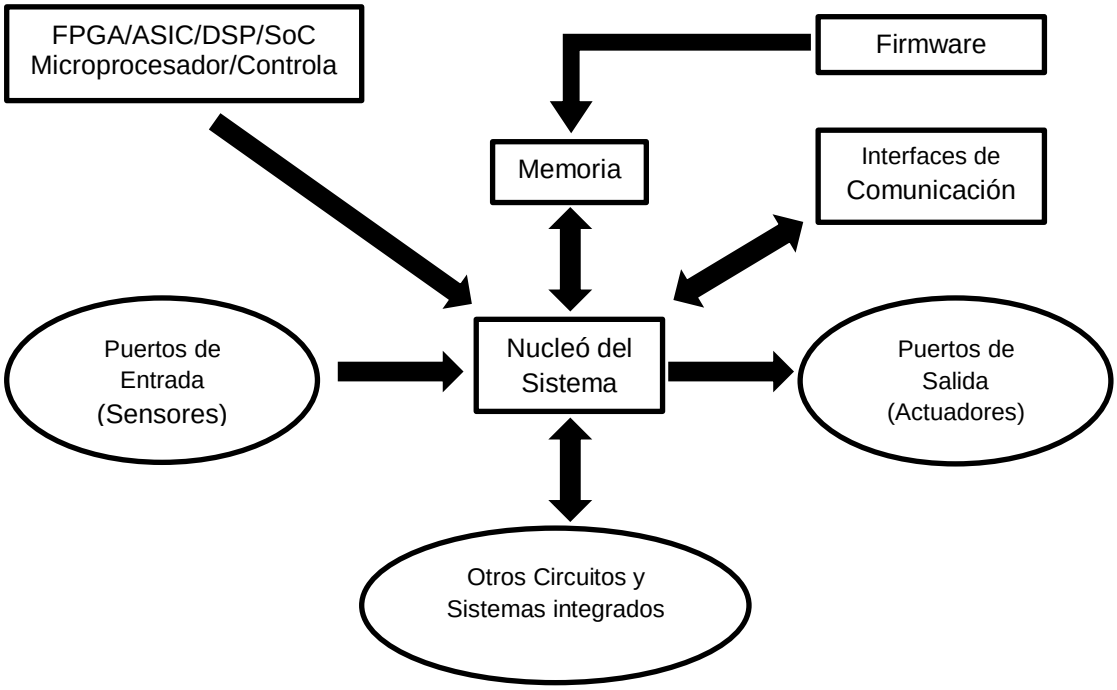


Figura 2-2 - Elementos de un Sistema Embebido

[Fuente: *Introduction to Embedded Systems.*]

- **Núcleo del Sistema**

Un sistema embebido típico contiene un chip controlador, que actúa como el cerebro principal del sistema. El núcleo de un sistema embebido puede encontrarse en cualquiera de las siguientes categorías:

- Procesadores de Propósito General
  - Microprocesadores
  - Microcontroladores
  - Procesadores de Señal Digital (DSPs)
- Circuito Integrado de Aplicación Especifica (ASICs)
- Dispositivos Lógicos Programables (PLDs)

Los procesadores son las principales unidades funcionales de un sistema embebido y son los responsables de procesar instrucciones y datos. Los sistemas embebidos están diseñados alrededor de un procesador principal. La complejidad del procesador principal usualmente determina si este es clasificado como un microprocesador o un microcontrolador. Generalmente, los microprocesadores contienen un conjunto mínimo de memorias y componentes de Entrada/Salida integrados al chip. [6]

Los procesadores embebidos pueden ser separados dentro de varios grupos llamados arquitecturas. Lo que diferencia un grupo de otro es el conjunto de instrucciones que el procesador puede ejecutar. Los procesadores pueden ser considerados de una misma arquitectura si pueden ejecutar el mismo conjunto de instrucciones.

Si se examinan los sistemas embebidos se puede observar que están construidos en base a algunas de los núcleos mencionados anteriormente. Casi en el 80% de los casos los sistemas embebidos son basados en procesadores/Microcontroladores. [7]

- **Firmware**

El Firmware hace referencia al algoritmo de control y configuraciones que el desarrollador del sistema coloca en la memoria y código del sistema embebido, el firmware es una parte fundamental de un sistema embebido. El Software embebido puede ser desarrollado siguiendo los siguientes métodos:

- Escribir el programa en un lenguaje de programación de alto nivel como C/C++ usando un entorno de desarrollo integrado (IDE).
- Escribir el programa en lenguaje Ensamblador usando las instrucciones soportadas por el procesador/controlador.

El conjunto de instrucciones para cada familia de procesadores/controladores es diferente y el programa escrito en cualquiera de los métodos mencionados anteriormente debe ser convertido a un código de maquina entendible por el procesador antes de ser cargado en la memoria.
Para principiantes en el campo de software embebido es recomendable usar un lenguaje de alto nivel para el desarrollo del firmware. La razón de esto es: escribir código en un lenguaje de alto nivel es fácil, le código es altamente portable lo que significa que el código puede ser usado para correr en diferentes procesadores/controladores con una pocas modificaciones.
El proceso de desarrollo de software embebido en lenguaje ensamblador es tedioso y consume mucho tiempo. Para ello, el desarrollador necesita conocer todo el conjunto de instrucciones del procesador/controlador. Un programador que escribe el programa usando lenguaje ensamblador escribe el programa según su punto de vista. Por lo tanto, el software es dependiente del desarrollador, lo que resulta difícil para otra persona entender el código escrito en ensamblador si no fue bien documentado. [7]

- **Memoria**

La memoria es una parte importante de un sistema embebido. Es la responsable de alojar el algoritmo de control y otros detalles de configuración importantes. Todo sistema embebido requiere de una memoria para contener el algoritmo de control o un Sistema Operativo y la aplicación diseñada para correr sobre este, una memoria de datos para contener variables y datos temporarios durante la ejecución de tareas, y una memoria para contener datos no volátiles modificables por la aplicación. Los requerimientos de un sistema embebido en términos de memorias son exclusivamente dependientes del tipo de sistema embebido y las aplicaciones para las cuales fue diseñado [7].

- **Sensores y Actuadores**

Los sistemas embebidos son básicamente diseñados para regular una variable física o manipular el estado de algún dispositivo. Normalmente un sistema embebido consta con una serie de salidas y entradas necesarias para comunicarse con el mundo exterior. Los Sensores conectados a los puertos de entrada de un sistema embebido detectan los

cambios en las variables de entrada y los Actuadores conectados a los puertos de salida del sistema embebido controlan algunas variables de acuerdo con los cambios en la entrada. De aquí que un sistema embebido puede ser visto como un sistema reactivo, es decir, que su funcionamiento depende de la continua interacción con un determinado ambiente el cual determina las posibles respuestas del sistema.

2.7.3 Clasificación

En base a su complejidad y los requerimientos de rendimiento del sistema, es posible clasificar los sistemas embebidos en tres grupos:

- **De escala pequeña**

Sistemas embebidos de aplicación simple y cuyos requerimientos de rendimiento no son de tiempo crítico. Usualmente, se construyen en base a controladores de bajo rendimiento y costo. Estos sistemas pueden o no contener un sistema operativo para su funcionamiento.

- **De escala media**

Sistemas embebidos con requerimientos de complejidad media en cuanto a hardware y software. Se construyen en base a procesadores de rendimiento medio. Usualmente contienen un sistema operativo embebido (sistema operativo de propósito general o sistema operativo de tiempo real) para su funcionamiento.

- **Sofisticados**

Sistemas embebidos con altos requerimientos de complejidad en hardware y software caen dentro de este grupo. Son aplicados en aplicaciones de misión crítica que demanda alto rendimiento. Se construyen en base a procesadores de alto rendimiento. Pueden contener múltiples controladores.

2.7.4 Sistema Operativo Embebido

Un Sistema Operativo (SO) es una parte opcional del stack de software de un sistema embebido. Los SO pueden ser usados sobre cualquier procesador al cual haya sido portado. Como se puede observar en la Figura 2-3 el SO se encuentra sobre la capa de hardware [19].

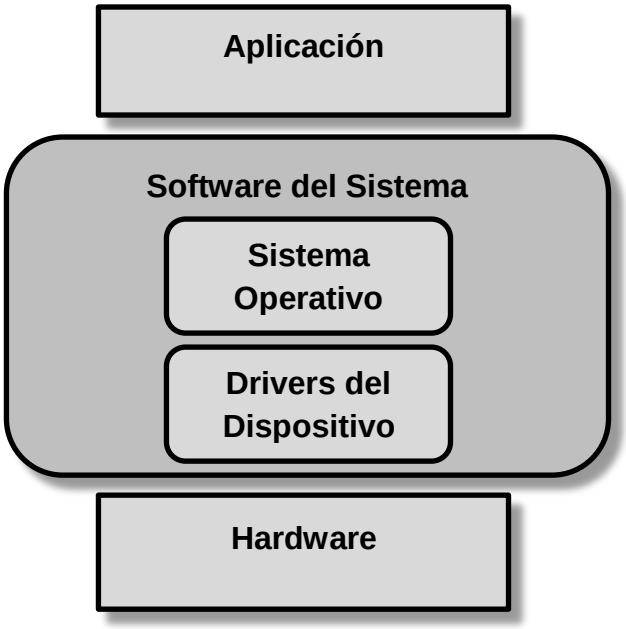


Figura 2-3 - SO y el Modelo de Sistema Embebidos
[Fuente: Embedded Systems Arquitecture - A Compresive Guide for Engineers and Programmers]

El SO es un conjunto de librerías de software que tienen dos propósitos principales en un sistema embebido: proveer una capa de abstracción para el software en la parte superior del sistema operativo para que dicho software sea menos dependiente del hardware, y la gestión de los diferentes recursos de hardware y software del sistema para asegurar que todo el sistema funciona de manera eficiente y fiable. Mientras los SO varían en que componentes poseen, todos los SO tienen al menos un kernel. El kernel es un componente que contiene las funcionalidades principales del SO, entre ellas podemos incluir las siguientes:

- Gestión de Procesos.
- Gestión de Memoria.
- Gestión de Entradas / Salidas.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

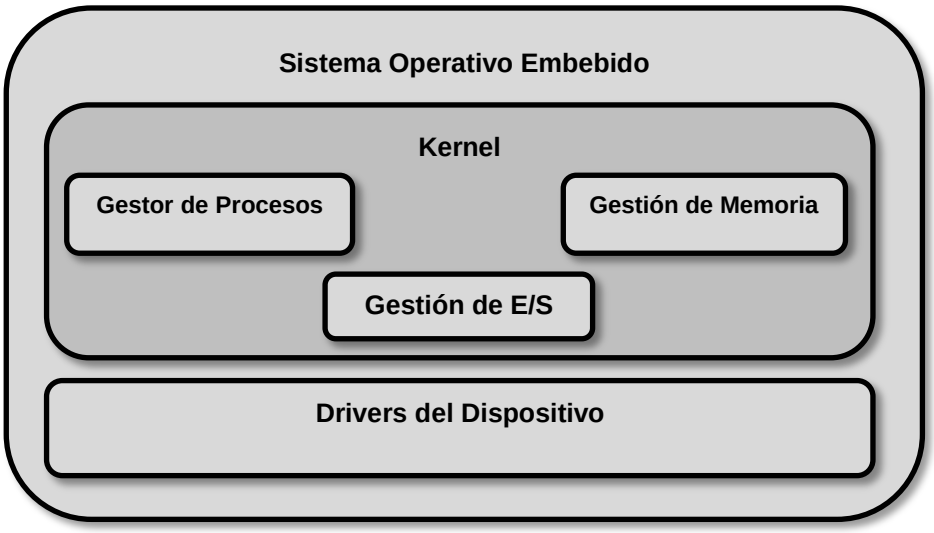


Figura 2-4 - Modelo General de un Sistema Operativo
[Fuente: Embedded Systems Arquitecture - A Compresive Guide for Engineers and Programmers]

2.8 Ingeniería del Software

El proceso de ingeniería de software se define como “un conjunto de etapas parcialmente ordenadas con la intención de logra un objetivo, en este caso, la obtención de un producto de software de calidad”. El proceso de desarrollo del software “es aquel en el cual las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para su uso operativo.” Concretamente “define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo”. [15]

Existen varios modelos de desarrollo a seguir. A continuación, se explicará los modelos de Metodologías para el desarrollo de software elegidos en el presente trabajo final.

2.8.1 Modelo en Cascada

También llamado **modelo secuencial** o ciclo de vida de un programa (denominado así por la posición de las fases en el desarrollo que parecen caer en cascada “por gravedad” hacia las siguientes fases, ver Figura 2-5), es un enfoque metodológico que ordena rigurosamente las etapas de proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperara a la finalización de la etapa anterior. Al final de cada etapa, el modelo está diseñado para llevar a cabo una revisión final, que se encarga de determinar si el proyecto está listo para avanzar a la siguiente fase [16]. Las fases del modelo son:

1) **Análisis de requisitos:**

En esta fase se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. Es importante señalar que en esta etapa se debe consensuar todo lo que se requiere del sistema y será aquello lo que seguirá en las siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software de una manera.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

2) Diseño del Sistema:

Descompone y organiza el sistema en elementos que puedan elaborarse por separado. Es conveniente distinguir entre diseño de alto nivel o arquitectónico y diseño detallado. El primero de ellos tiene como objetivo definir la estructura de la solución (una vez que la fase de análisis ha descrito el problema) identificando grandes módulos (conjuntos de funciones que van a estar asociadas) y sus relaciones. Con ello se define la arquitectura de la solución elegida. El segundo define los algoritmos empleados y la organización del código para comenzar la implementación.

3) Diseño del programa

Es la fase en donde se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario así como también los análisis necesarios para saber qué herramientas usar en la etapa de Codificación.

4) Codificación

Es la fase en donde se implementa el código fuente, haciendo uso de prototipos así como de pruebas y ensayos para corregir errores. Dependiendo del lenguaje de programación y su versión se crean las bibliotecas y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.

5) Pruebas

Los elementos, ya programados, se ensamblan para componer el sistema y se comprueba que funciona correctamente. Se buscan sistemáticamente y se corrigen todos los errores antes de ser entregado al usuario final.

6) Implementación o Verificación del programa

Es la fase en donde el usuario final o el cliente ejecutan el sistema, y se asegura que cubra sus necesidades.

7) Mantenimiento

Es la operación y el mantenimiento del software ya que al utilizarlo como usuario final puede ser que no cumpla con todas nuestras expectativas.

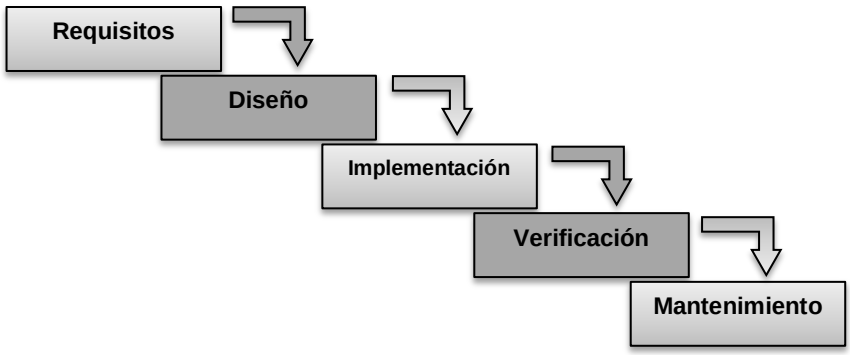


Figura 2-5 - El Modelo en Cascada

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

[Fuente: El proceso Unificado del desarrollo del Software]

2.8.2 El Proceso Unificado de Desarrollo de Software (RUP)

Actualmente, la demanda de software potente y complejo genera que los desarrolladores tengan que afrontar problemas en la coordinación de múltiples cadenas de trabajo. Por lo tanto, es necesario de contar con un proceso que integre las múltiples facetas del desarrollo y permita a los desarrolladores trabajar de forma coordinada. Es necesario un método común, un proceso que:

- Proporcione una guía para ordenar las actividades de un equipo.
- Dirija las tareas de cada desarrollador por separado y del equipo como un todo.
- Especifique los artefactos que deben desarrollarse.
- Ofrezca criterios para el control y la medición de los productos y actividades del proyecto.

El Proceso Unificado de Desarrollo se puede ver como una solución al problema del software.

El Proceso Unificado es un proceso de desarrollo de software. Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. El Proceso Unificado es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software y diferentes tamaños de proyectos.

El Proceso Unificado utiliza el Lenguaje Unificado de Modelado (UML) para generar todos los esquemas de un sistema software. UML es una parte esencial del Proceso Unificado. Sin embargo, los aspectos que definen al Proceso Unificado se resumen en tres conceptos claves: dirigido por casos de usos, centrado en la arquitectura, e iterativo e incremental. Lo cual hace único al Proceso Unificado [8][17]

- **Orientado a Casos de Usos**

Un sistema software se desarrolla para dar servicios a sus usuarios. Por lo tanto, para poder construir un sistema de forma exitosa es necesario conocer lo que los futuros usuarios necesitan y desean.

Este tipo de interacción se le conoce como caso de uso. Un caso de uso es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado. Los casos de usos representan los requerimientos funcionales. El conjunto de casos de usos del sistema constituyen el modelo de casos de usos el cual describe la funcionalidad total del sistema. Los casos de usos son una herramienta que no solo sirven para especificar los requerimientos de un sistema sino también guían su diseño, implementación y prueba, es decir, guían el proceso de desarrollo. En base al modelo de casos de usos se crean una serie de modelos de diseño e implementación que llevan a cabo los casos de usos [8][17].

- **Centrado en Arquitectura**

La arquitectura de un sistema software se describe mediante diferentes vistas del sistema en construcción. El concepto de arquitectura software incluye los aspectos estáticos y

dinámicos más significativos del sistema. La arquitectura es una vista del diseño completo con las características más importantes resaltadas, dejando los detalles de lado

Arquitectura: Conjunto de decisiones significativas acerca de la organización de un sistema software, la selección de los elementos estructurales a partir de los cuales se compone el sistema, las interfaces entre ellos, su comportamiento, sus colaboraciones, y su composición.

Los casos de uso y la arquitectura están profundamente relacionados. Los casos de uso deben encajar en la arquitectura, y a su vez la arquitectura debe permitir el desarrollo de todos los casos de uso requeridos, actualmente y a futuro. El arquitecto desarrolla la forma o arquitectura a partir de la comprensión de un conjunto reducido de casos de uso fundamentales o críticos (usualmente no más del 10 % del total). En forma resumida, podemos decir que el arquitecto:

- Crea un esquema en borrador de la arquitectura comenzando por la parte no específica de los casos de uso (por ejemplo la plataforma) pero con una comprensión general de los casos de uso fundamentales.
- A continuación, trabaja con un conjunto de casos de uso claves o fundamentales. Cada caso de uso es especificado en detalle y realizado en términos de subsistemas, clases, y componentes.
- A medida que los casos de uso se especifican y maduran, se descubre más de la arquitectura, y esto a su vez lleva a la maduración de más casos de uso.

Este proceso continúa hasta que se considere que la arquitectura es estable [18].

- **Iterativo e Incremental**

Es práctico dividir el esfuerzo de desarrollo de un proyecto de software en partes más pequeñas o mini proyectos. Cada mini proyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en el flujo de trabajo, y los incrementos a crecimientos en el producto.

Las iteraciones deben estar controladas. Esto significa que deben seleccionarse y ejecutarse de una forma planificada. Los desarrolladores basan la selección de lo que implementarán en cada iteración en dos cosas: el conjunto de casos de uso que amplían la funcionalidad, y en los riesgos más importantes que deben mitigarse.

En cada iteración los desarrolladores identifican y especifican los casos de uso relevantes, crean un diseño utilizando la arquitectura seleccionada como guía, para implementar dichos casos de uso. Si la iteración cumple sus objetivos, se continúa con la próxima. Sino deben revisarse las decisiones previas y probar un nuevo enfoque.

Beneficios del enfoque Iterativo:

- La iteración controlada reduce el riesgo a los costes de un solo incremento.
- Reduce el riesgo de retrasos en el calendario atacando los riesgos más importantes primero.

- Acelera el desarrollo. Los trabajadores trabajan de manera más eficiente al obtener resultados a corto plazo.
- Tiene un enfoque más realista al reconocer que los requisitos no pueden definirse completamente al principio

➢ El ciclo de vida del proceso unificado

El Proceso Unificado se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo constituye una versión del sistema.

➢ Fases

Cada ciclo consta de cuatro fases: **inicio, elaboración, construcción, y transición.** La Figura 2-6, muestra estas fases y su descripción.

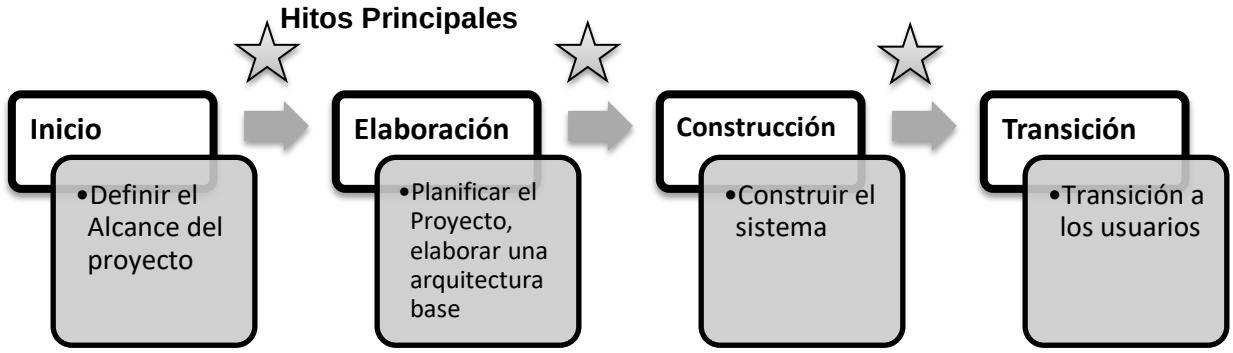


Figura 2-6 - Fases e Hitos de un Proyecto
[Fuente: El proceso Unificado del desarrollo del Software]

Cada fase se subdivide en iteraciones (ver Figura 2-7). En cada iteración se desarrolla en secuencia un conjunto de disciplinas o flujos de trabajos.

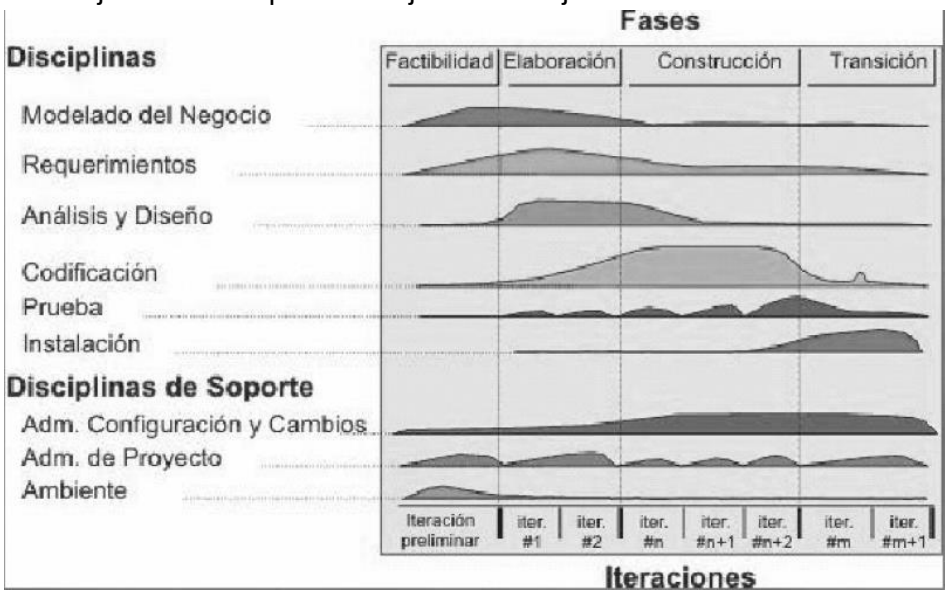


Figura 2-7 - Disciplinas Básicas del proyecto
[Fuente: El proceso Unificado del desarrollo del Software]

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

- **Disciplinas**

Cada disciplina es un conjunto de actividades relacionadas (flujos de trabajo) vinculadas a un área específica dentro del proyecto total. Las más importantes son:
Requerimientos, Análisis, Diseño, Codificación, y **Prueba.**
El agrupamiento de actividades en disciplinas es principalmente una ayuda para comprender el proyecto desde la visión tradicional en cascada. La Figura 2-8, muestra las relaciones entre las Fase, Iteraciones y las Disciplinas.

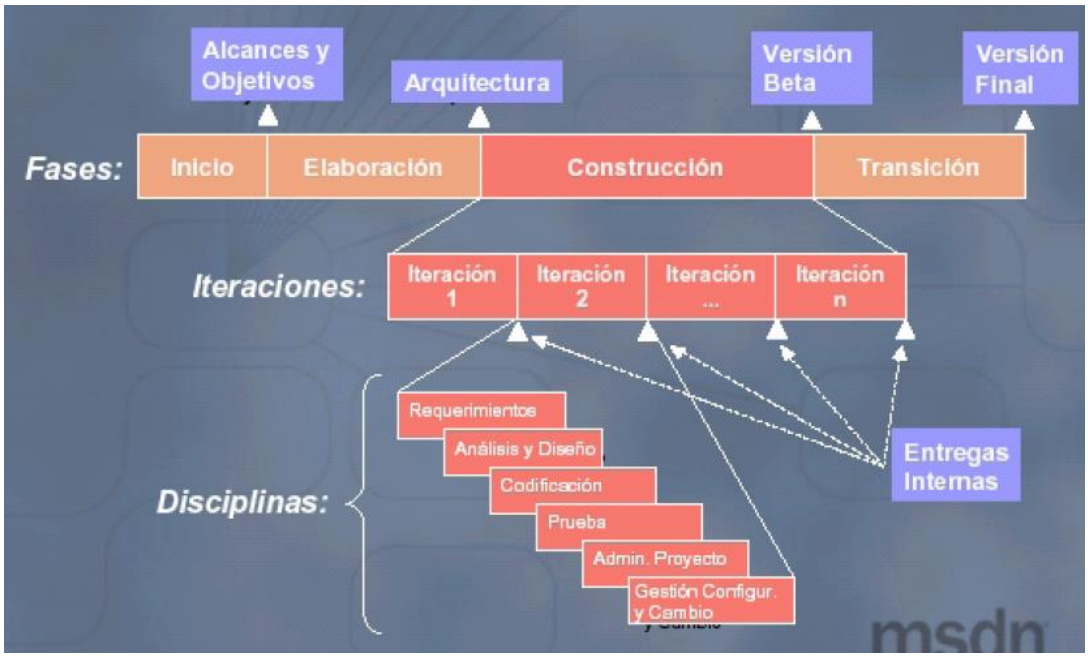


Figura 2-8 - Fases, Iteraciones y Disciplinas
[Fuente: El Proceso Unificado del desarrollo del Software]

Cada disciplina está asociada con un conjunto de **modelos** que se desarrollan. Estos modelos están compuestos por **artefactos**. Los artefactos más importantes son los modelos que cada disciplina realiza: **modelo de casos de uso, modelo de diseño, modelo de implementación, y modelo de prueba** (ver Figura 2-9)**.**

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

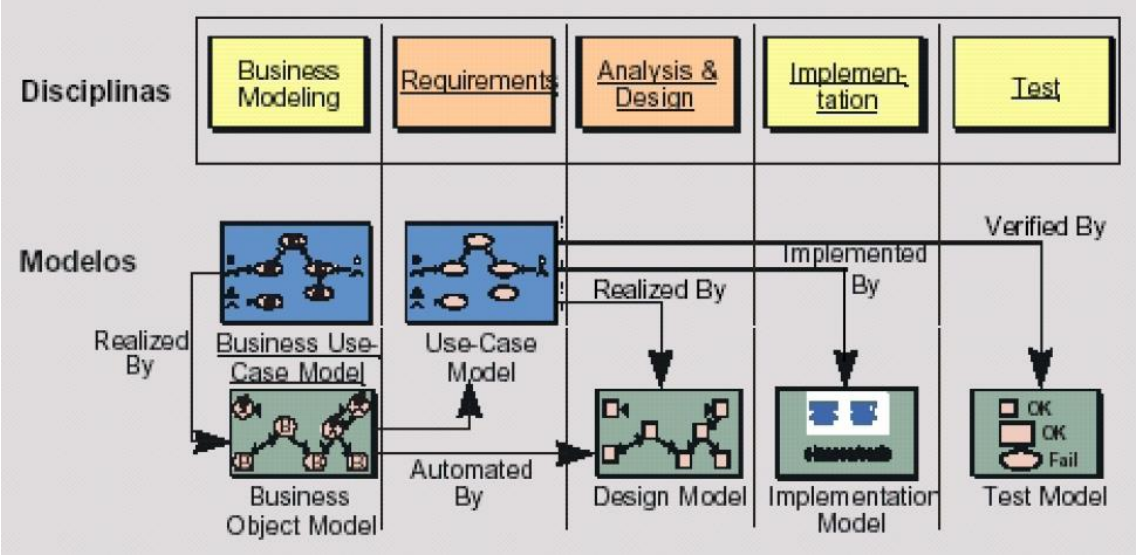


Figura 2-9 - Modelos Producidos en las Disciplinas
[Fuente: El Proceso Unificado del desarrollo del Software]

- Un modelo de casos de usos, con todos los casos de uso y su relación con los usuarios.
- Un modelo de diseño que define la estructura estática del sistema en forma de subsistemas, clases e interfaces y los casos de usos reflejados como colaboraciones entre los subsistemas, clases e interfaces.
- Un modelo de implementación, que incluye componentes y la correspondencia de las clases con los componentes.
- Un modelo de prueba, que describe los casos de prueba que verifican los casos de usos.
- El sistema también debe tener un **modelo del dominio** o **modelo del negocio** que describa el contexto del negocio en el que se halla el sistema.

➢ **Modelo de Dominio**

Un modelo de dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabaja el sistema.
Los objetos o clases del dominio se obtienen de una especificación de requisitos. Las clases del dominio aparecen en tres formas:

- Objetos que representan cosas manipuladas por el negocio, como pedidos, cuentas y contratos.
- Objetos del mundo real y conceptos de los que el sistema debe hacer un seguimiento, como la aviación enemiga, misiles y trayectorias.
- Sucesos que ocurrirán o han ocurrido, como la llegada de un avión y su salida.

El modelo del dominio se describe mediante diagramas de UML y muestran al cliente, usuarios y desarrolladores las clases del dominio y como se relacionan.

El objetivo del modelado del dominio es comprender y describir las clases más importantes del contexto del sistema [8][18].

- **Modelo del Negocio**

El modelado del negocio es una técnica para comprender los procesos de negocio de la organización.

Este modelo, está soportado por dos tipos de modelos de UML: **el modelado de casos de uso,** y **modelos de objetos**.
Un Modelo de Casos de Uso del Negocio describe los procesos de negocio de una empresa en términos de casos de uso del negocio y actores del negocio que se corresponden con los procesos del negocio y los clientes respectivamente.
Al igual que el modelo de casos de uso para un sistema software, el modelo de casos de uso del negocio presenta un sistema (en este caso, el negocio) desde la perspectiva de su uso, y esquematiza como proporciona valor a sus usuarios.
El modelo de casos de uso del negocio se describe mediante diagramas de casos de uso.
Un modelo de objetos del negocio describe como cada caso de uso del negocio es llevado a cabo por parte de un conjunto de trabajadores que utilizan un conjunto de entidades del negocio y de unidades de trabajo, cada realización de uno de estos casos de uso del negocio puede mostrarse en diagramas de interacción y diagramas de actividad.
Las entidades del negocio representan algo que los trabajadores toman, manipulan, inspeccionan, producen o utilizan en un negocio.
Una unidad de trabajo es un conjunto de esas entidades que conforma un todo reconocible para el usuario final.
La técnica de modelado de negocio identifica entidades y trabajadores que participan en la realización de los casos de uso del negocio.
Los trabajadores identificados en el modelo de negocio se utilizan como punto de partida para derivar un primer conjunto de actores y casos de uso del sistema.

El modelado del negocio y el modelado del dominio se parecen en muchos aspectos. Se puede pensar en el modelado del dominio como una variante simplificada del modelado del negocio, en la cual nos centramos solo en las “cosas”, es decir, en las clases del dominio y entidades del negocio que necesitan utilizar los trabajadores.[18]

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

CAPITULO 3

3 Hardware del Sistema Embebido

En este capítulo, se explicara detalladamente el proceso de desarrollo del hardware sistema de telemetría, sus componentes bases y su configuración.

Para el desarrollo del dispositivo del Sistema de Telemetría que permita la lectura de los datos obtenidos por los sensores, la comunicación por SMS/GSM y la aplicación web, es necesario realizar el desarrollo de un sistema embebido, capaz de interconectar todos los componentes, a la vez que pueda actuar como un servidor web.

El siguiente diagrama de bloques (Ver Figura 3-1) muestra la relación entre los componentes utilizados:

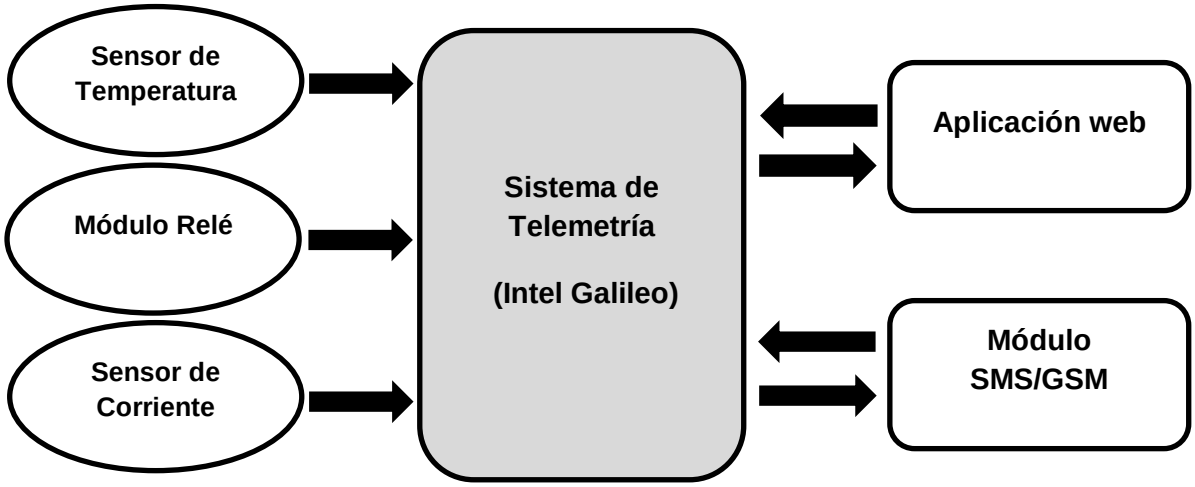


Figura 3-1 - Diagrama de Bloques del Sistema

A Continuación se describirá el procedimiento que se utilizó para desarrollar el sistema embebido de telemetría, empezando por la plataforma Intel Galileo, explicando la razón por la que se la eligió, las configuraciones iniciales de la misma, luego, se procederá a explicar los componentes que se van a utilizar en el montaje del sistema, sensores y módulos de comunicación, sus configuraciones iniciales y conexiones, finalmente, se explicará los elementos de software utilizados para crear el firmware, su lógica, y funcionamiento.

3.1 Intel Galileo Generación 1

Es una familia de placas de desarrollo compatibles con Arduino que trabajan bajo arquitectura Intel, esta placa ofrece las mismas posibilidades que los modelos Raspberry PI o Arduino, es una pizarra en blanco que permite a los desarrolladores crear infinitos proyectos.

Gracias a su gran conectividad, potencia de proceso, y el uso de SDKs muy sencillos permite crear software que, por ejemplo, conecte a Internet cualquier dispositivo, un

dispositivo que se ilumine cada vez que realizamos una acción o sensores de distintas magnitudes tanto analógicas como digitales. Es una herramienta muy interesante de cara a domótica (Automatización del hogar) y para proyectos que necesiten un PC pero de bajo consumo y coste.

La placa Intel Galileo es la primera de una gama de placas de desarrollo compatibles con Arduino basadas en arquitecturas Intel. Esta placa de desarrollo ejecuta un sistema operativo Linux libre que contiene las librerías de software de Arduino, lo que le permite ofrecer una mayor escalabilidad y reutilizar el software ya existente, llamados “bocetos”, y la instalación de software compatible, mediante gestor de paquetes de Linux.

Se puede programar la Intel Galileo desde los sistemas operativos Mac OS, Microsoft Windows y Linux de sus equipos. Además, estas placas también han sido diseñadas para ser compatibles, a nivel de software y de hardware, con el ecosistema Arduino Shield.

Figura 3-2 - Plataforma Intel Galileo

La placa Intel Galileo está diseñada para soportar módulos que operan tanto a 3.3V o 5V. El voltaje principal de funcionamiento de la Galileo es 3.3V. De todas formas, un jumper en la placa permite la trasformación a 5V en los pines de entrada/salida. Esto provee soporte a los Módulos de 5V de la plataforma Arduino como configuración por defecto, si se desea, se puede deshabilitar el jumper para proveer de 3.3V a los pines de entrada/salida.

La placa Intel Galileo posee 14 pines de entrada/salida de información digital (D0-D13) y 6 pines para entrada/salida de información Analógica (A0-A5). Además de la compatibilidad con el hardware y software de Arduino, la placa Galileo incluye varios puertos de E/S estándar para PC y características para expandir sus usos y capacidades más allá del ecosistema de módulos de Arduino. Un puerto mini-PCI Express, puerto 100mb Ethernet, puerto Micro-SD, puerto RS-232 serial, puerto para host USB, puerto para cliente USB y 8 Mb de memoria RAM Flash, vienen por defecto en la placa. [10]

En la tabla 3-1, se compara las características de la plataforma Intel galileo con otras placas de desarrollo.

Característica	Arduino UNO	Intel Galileo
SoC	Atmel ATmega328P	Intel Quark SoC X1000
Arquitectura CPU	AVR	x86 Quark
Núcleos	1	1

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Frecuencia	16 MHz	400 Mhz
Tipo de RAM	SRAM	DDR3
Tamaño de RAM	2 KB	256 MB
PCIe	N/A	1 mini
USB	N/A	1
Almacenamiento	32 KB Flash	8 MB Flash
Ranuras Flash	N/A	Hasta 32 gb por microSD
Comunicación Red	N/A	ETH (10/100)
GPIO	22	20
Analogico	10-bit ADC	12-bit ADC
Sistema Operativo	N/A	Linux

Tabla 3-1 - Comparativa Arduino UNO e Intel Galileo

Por lo mencionado y descripto anteriormente, se eligió esta plataforma para el desarrollo del proyecto.

Las características detallas de la Placa Intel Galileo serán listadas a continuación en la Tabla 3-2:

Detalles de la Arquitectura y características soportadas	
Procesador compatible con el conjunto de instrucciones(ISA) 400MHz 32-bit Intel® Pentium	16 KByte L1 cache
	512 KBytes of on-die SRAM embebida
	Simple de programar: Hilos simples, un solo núcleo , velocidad constante
	CPU compatible con ACPI, soporta estados de sueño(sleep)
	Un reloj de tiempo real integrado (RTC), con la opción de usar una batería de tipo moneda de 3V para operar entre los ciclos de escondidos.
Conector Ethernet 10/100	
Ranura completa PCI Express mini-card, con características compatibles PCIe 2.0	Funciona con mitad de tarjetas mini-PCIe con placa convertidor opcional
	Provee conexión de puerto host USB 2.0 con conector mini-PCIe.
USB 2.0 conector Host	Soporta hasta 128 USB dispositivos end point.
Conector USB client, usado para programación	Más allá del puerto de programación - es un dispositivo totalmente compatible con USB 2.0
Cabecera JTAG estándar 10-Pin para debugging	
Botón de reboot para reiniciar el procesador	
Botón de reset para reiniciar el sketch y cualquier escudo conectado	
Opciones de almacenaje	Memoria Flash SPI de 8 Mbyte cuyo principal propósito es almacenar el firmware (o el bootloader) y el ultimo sketch. Entre 256 Kbyte y 512 Kbyte está dedicado al almacenaje del sketch. La carga sucede automáticamente desde la PC de desarrollo, así que no se requiere acción alguna, al menos que se esté agregando una actualización al firmware.

	512 Kbyte de SRAM embebida habilitada por el firmware por defecto.
	256 Kbyte de DRAM habilitada por el firmware por defecto.
	Tarjeta micro SD opcional que ofrece hasta 32Gbyte de almacenaje.
	Almacenaje USB que funciona con cualquier controlador compatible con USB 2.0
	11 Kbyte EEPROM puede ser programado a través de la librería EEPROM

Tabla 3-2 - Especificaciones detalladas de la Plataforma Intel Galileo G1

3.1.1 Configuración de la Placa Intel Galileo

Antes de empezar a desarrollar con la plataforma Intel Galileo, hay una serie de pasos que se debe seguir, los que serán detallados a continuación:

Encendido de la Placa Intel Galileo

Para encender la Placa Intel Galileo debemos conectar el cable de alimentación (1), esperar unos segundos y luego conectamos mediante un cable mini-USB a USB, usando el puerto mini-USB client en la placa (2), y el USB a nuestra computadora (Ver Figura 3-3).

Nota: Es muy importante conectar la alimentación primero y luego de unos segundos esperar que la luz del Client-USB se encienda indicando que podemos realizar la conexión USB a la computadora, de esta manera, evitaremos daños al hardware.

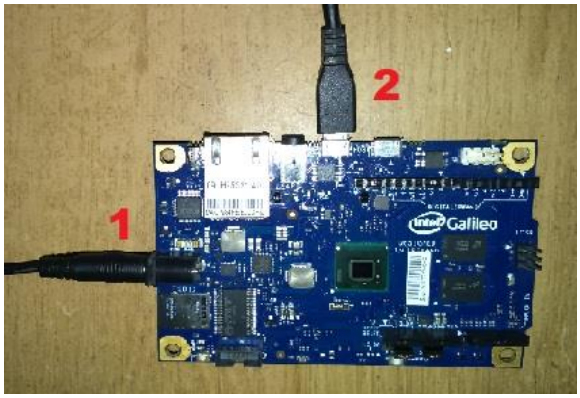


Figura 3-3 - Conexión de la Placa Intel Galileo a la PC

Actualización del Firmware

Como primer paso, debemos descargar la herramienta Intel Galileo Firmware Updater, es necesaria para actualizar el firmware base de la placa Intel Galileo a su versión más reciente, luego descargamos los drivers de la página oficial de Intel, los instalamos en nuestra PC, también debemos contar con una versión actualizada de Java Runtime Enviroment (JRE) y Java Enterprise Edition JDK.

Una vez que la computadora reconozca la conexión de nuestra placa, ejecutamos la herramienta Firmware Updater, luego, seleccionamos el puerto donde está conectada la

placa Intel Galileo, nos mostrará la versión actual de la placa, y la versión por la cual se va a actualizar(ver Figura 3-4).

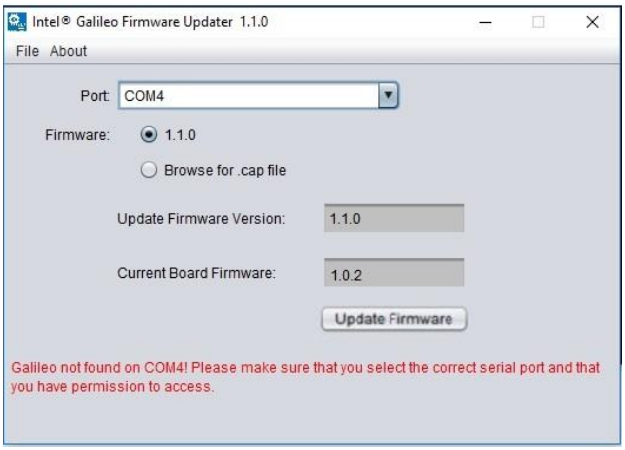


Figura 3-4 - Actualización del Firmware de la placa Intel Galileo

3.1.2 Instalación del S.O. Linux Yocto

La placa Intel Galileo cuenta con un sistema operativo almacenado en su memoria flash, que tiene instalados las funciones básicas necesarias para el funcionamiento de la misma, para poder hacer uso de programas y utilidades más complejos, debemos instalar una imagen de un SO en una micro-SD de al menos 2 GB, de la cual iniciará la placa Intel Galileo.

Primero descargamos la versión del sistema operativo en base Linux Yocto 1.0.4, la imagen que vamos a utilizar (ver Figura 3-5) fue creada por el usuario de la comunidad Intel Galileo AlexT, y contiene herramientas de desarrollo, así también como el gestor de paquetes OPKG, lo que nos permitirá instalar los programas que necesitemos más adelante, este aspecto será abordado en el capítulo 4 de este trabajo final.

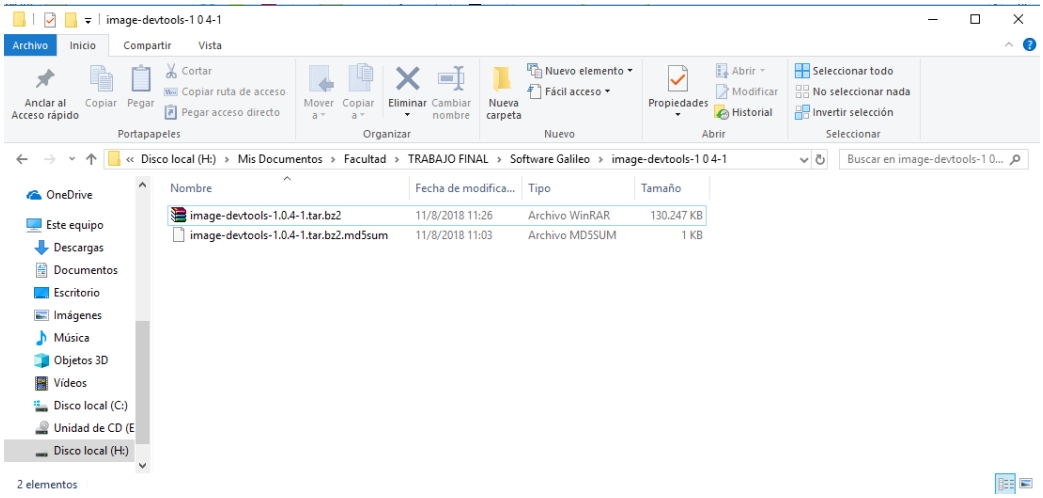


Figura 3-5 - Imagen Linux Yocto 1.0.4

La imagen estará comprimida en un archivo RAR que debemos extraer para después, copiar los archivos extraídos en el directorio raíz de la tarjeta SD (ver Figura 3-6).

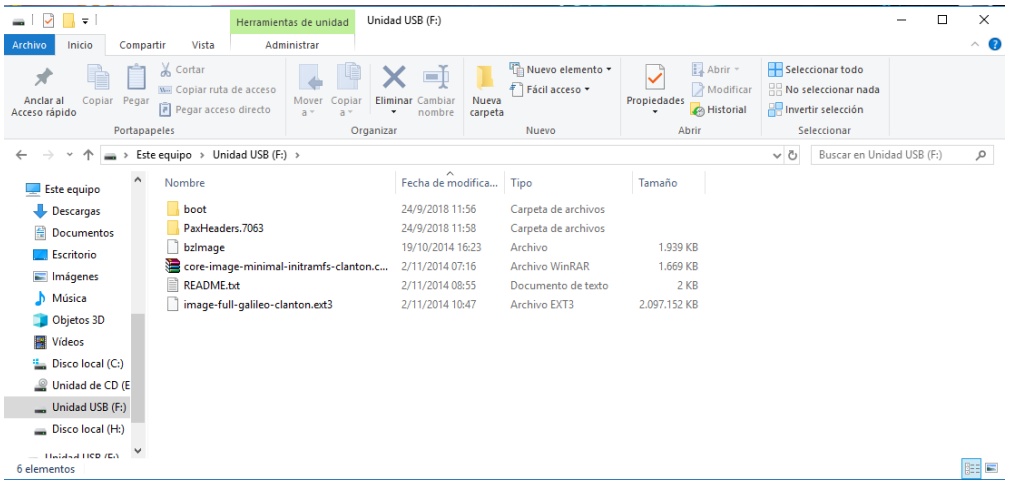


Figura 3-6 - Directorios en la tarjeta SD, de la imagen Linux Yocto

3.2 Sensores de temperatura y corriente

A continuación se explicará el proceso de configuración y conexión con la placa Intel Galileo del Hardware elegido para realizar las mediciones de temperatura y corriente dentro de la sala de servidores.

3.2.1 Sensor de temperatura y humedad DHT 11

Para la medición de las variables de temperatura y humedad se eligió, el sensor de temperatura y humedad DHT11 el cual presenta un sensor complejo y a la vez de uso simple con una salida para señal digital calibrada.

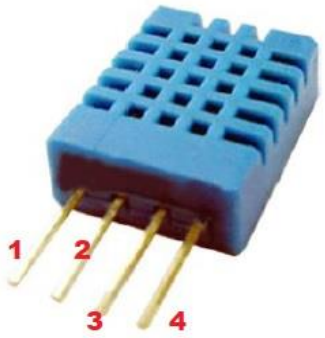


Figura 3-7 - Sensor de temperatura y humedad DHT 11

El sensor DHT 11 posee un microcontrolador de 8 bit que ofrece una respuesta rápida y resistente a interferencias externas, una memoria interna del tipo OTP (One Time Programable, programable una sola vez) la cual es usada para almacenar los coeficientes de calibración que el sensor utiliza en el proceso de detección de señales, esto brinda lecturas rápidas y fiables con poca alimentación de voltaje. El Sensor DHT11 opera en los rangos de 0-50°C con una precisión de ±2°C para la temperatura, y de 20-90% HR(Humedad Relativa) con precisión de ±5% HR. [11]

Estos valores de operación son aceptables para el proyecto, además del bajo costo del sensor mismo, se concluyó que este dispositivo sería adecuado para el proyecto.

En la tabla 3-3, se enlistan las especificaciones técnicas del sensor DHT11.

Parámetros	Condiciones	Mínimo	Típico	Máximo
Humedad				
Resolución		1%RH	1%RH	1%RH
			8 Bit	
Repetitividad			±1%RH	
Precisión	25°C		±4%RH	
	0-50°C			±5%RH
Intercambiabilidad	Intercambiable por Completo			
Rango de Medición	0°C	30%RH		90%RH
	25°C	20%RH		90%RH
	50°C	20%RH		80%RH
Tiempo de Respuesta (En segundos)	1/e (63%)25°C, 1m/s Air	6 S	10 S	15 S
Histéresis			±1%RH	
Estabilidad a largo plazo	Típica		±1%RH/año	
Temperatura				
Resolución		1°C	1°C	1°C
		8 bit	8 Bit	8 Bit
Repetitividad			±1°C	
Precisión		±1°C		±2°C
Rango de Medición		0°C		50°C
Tiempo de Respuesta (En segundos)	1/e (63%)	6 S		30 S

Tabla 3-3 - Especificaciones detalladas del sensor de temperatura y humedad DHT 11

3.2.1.1 Comunicación

El sensor DHT 11 es compatible tanto en hardware como en software con la familia de placas Arduino, incluyendo la Intel Galileo, en hardware el sensor posee 4 pines, el 1Pin se conecta a la alimentación de 5v, el 2Pin envía la señal digital al MCU(Micro Computer Unit, Unidad de Micro Computación) que en este caso, será la Intel Galileo se explicará más adelante su conexión, el 3Pin no tiene uso, el 4Pin debe ser conectado a la masa (tierra, GND) de la placa Galileo (ver Figura 3-8).

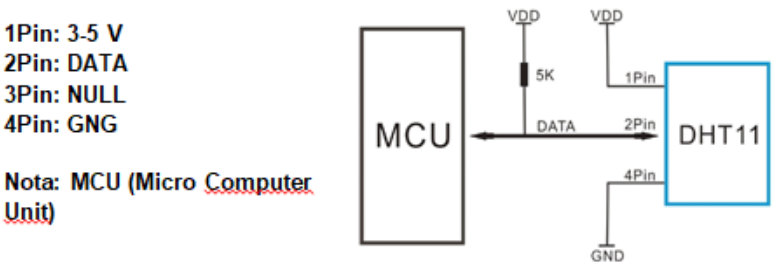


Figura 3-8 - Diagrama del circuito de conexión básico del sensor DHT 11

Para la conexión entre el sensor y la placa Intel Galileo, se consideró que el Microprocesador de la Intel Galileo, opera a una velocidad superior que el microcontrolador

dentro del sensor, el flujo de entrada y salida de datos entre ambos dispositivos, debe ser regulado por un diodo conectado en paralelo al 2Pin, y así mismo, a una resistencia de 5 KΩ, el diodo, divide la conexión en dos caminos, uno de entrada y otro de salida de datos. El flujo de datos que envía el sensor, se conectara a la entrada digital 2, y el flujo que recibe, será conectado a la entrada digital 3 de la Intel Galileo (ver Figura 3-9).

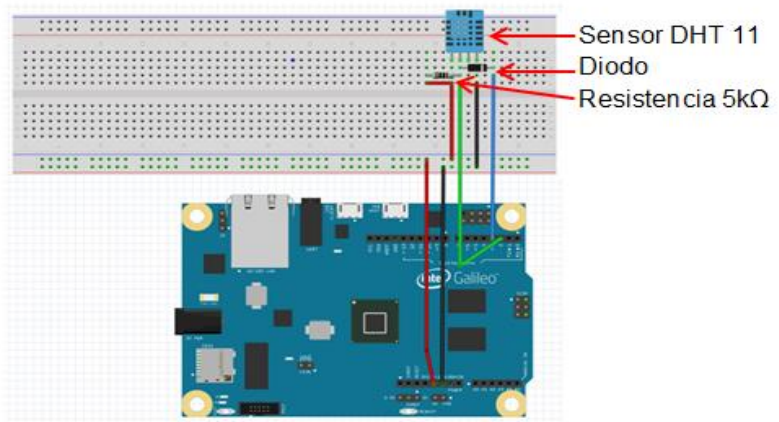


Figura 3-9 - Esquema de conexión del sensor DHT 11

En el aspecto del Software, entre las librerías de Arduino, existe una librería que se puede descargar desde el IDE de Arduino, utilizando el gestor de librerías de la misma aplicación, sin embargo, esta librería no está optimizada para el funcionamiento con la placa Intel Galileo por defecto, se procedió a descargar desde la comunidad Intel Galileo, la versión de esta librería modificada para ser compatible con la placa.

3.2.2 Sensor de Corriente SCT-013-000

El Sensor de Corriente SCT 013 se nos presenta como un sensor de corriente alterna no invasivo (de suspensión) que utiliza el mismo principio de funcionamiento que un transformador de corriente y una pinza amperimétrica.

Figura 3-10 - Sensor de Corriente alterna SCT-013-000

Existen diferentes tipos de sensores de corriente alterna SCT-013 que se pueden organizar en dos grupos. Los que proporcionan una corriente o los que proporcionan un voltaje. La

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

gran diferencia entre ellos es que en los primeros no viene incluida una resistencia de carga y en los segundos sí. [12]

La resistencia de carga, tiene como función, convertir la corriente en un voltaje limitado que podamos medir, por ejemplo, con Arduino y Galileo.

Modelo	**SCT-013-000**	SCT-013-005	SCT-013-010	SCT-013-015	SCT-013-020
Corriente de Entrada	**0-100A**	0-5A	0-10A	0-15A	0-20A
Tipo de Salida	**0-50mA**	0-1V	0-1V	0-1V	0-1V
Modelo	SCT-013-025	SCT-013-030	SCT-013-050	SCT-013-050	SCT-013-000v
Corriente de Entrada	0-25A	0-30A	0-50A	0-60A	0-100A
Tipo de Salida	0-1V	0-1V	0-1V	0-1V	0-1V

Tabla 3-4 - Lista de Modelos del Sensor SCT 013

Por las necesidades de este proyecto de trabajo final, se eligió el modelo SCT-013-000, ya que el rango de medida de corriente de entrada está dentro de los valores esperados de consumo de un equipo de aire acondicionado del tipo slipt, el cual, como se mencionó, se pretende monitorear con el sistema de telemetría. A continuación, en la tabla 3-5, se encuentran las especificaciones técnicas del sensor de corriente SCT-013-000.

Indicaciones técnicas	
Instalación suspendida en el cable de salida principal	
Características	
Resistencia al fuego	UL94-V0
Estandarización	GB1208-2006
Temperatura de operación	-25°C~+70°C
Temperatura de almacenamiento	-30°C~+90°C
Voltaje de operación	660V
Rango de frecuencia	50Hz-1KHz
Fuerza dieléctrica	3,5KV 50Hz 1min
Parámetros eléctricos	
Entrada nominal	100A
Entrada Máxima	120A
Salida nominal	50mA
Relación de vueltas	1:2000
Precisión	±1%
Linealidad	≤0,2
Error de Fase	
Máxima resistencia de muestreo	10 Ω
Peso	50g

Tabla 3-5 - Especificaciones detalladas del sensor de corriente alterna SCT-013-000

- **Principio de funcionamiento**

En su interior encontraremos los 3 componentes básicos de un transformador: devanado primario, devanado secundario y núcleo ferromagnético.

Cuando una corriente circula por el devanado primario, gracias al efecto de la inducción magnética, en el devanado secundario se produce una intensidad de corriente proporcional a la que pasa por el devanado primario.

Así es como funciona a grandes rasgos un transformador de intensidad.

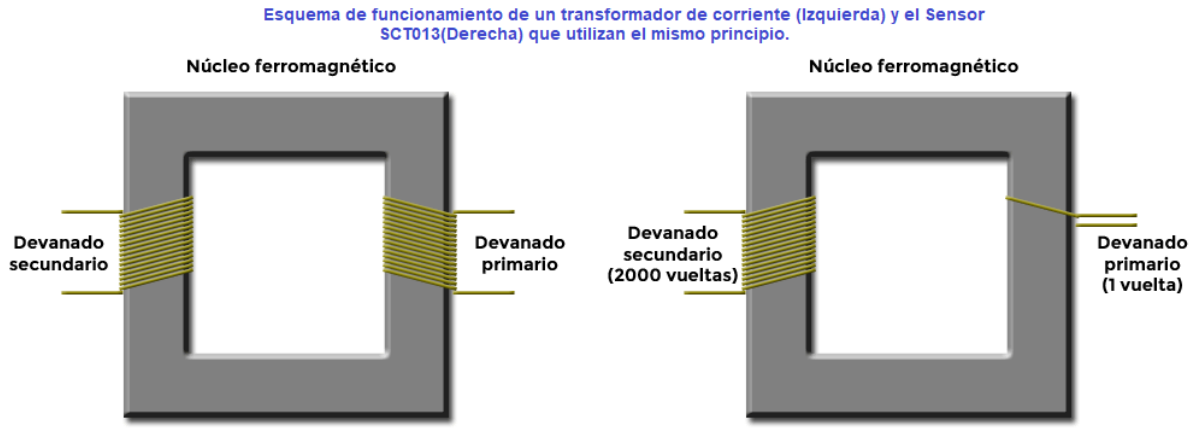


Figura 3-11 - Esquema de funcionamiento de un transformador de corriente

Un factor importante dentro de los transformadores de corriente es el número de espiras o número de vueltas que da el cable al núcleo ferromagnético. Conociendo estos datos y la corriente que circula por uno de los devanados se puede calcular la corriente por el otro devanado. Esto es debido a que guardan la siguiente relación:

$$\frac{N_p}{N_s} = \frac{I_s}{I_p} = \frac{V_p}{V_s}$$

A esta fórmula se le llama relación de transformación. Relaciona el número de espiras del primario (N_p), del secundario (N_s), las intensidades del primario (I_p), del secundario (I_s), el voltaje del primario (V_p) y del secundario (V_s).

En el caso del sensor de corriente alterna SCT-013 el devanado primario es el cable del aparato que queremos medir y el número de vueltas sería uno. El devanado secundario tiene 2.000 vueltas (ver Figura 3-11). [27]

- **Conexión y comunicación**

El sensor posee una ficha del tipo Jack 3,5 mm para la transmisión de los datos que recolecta.

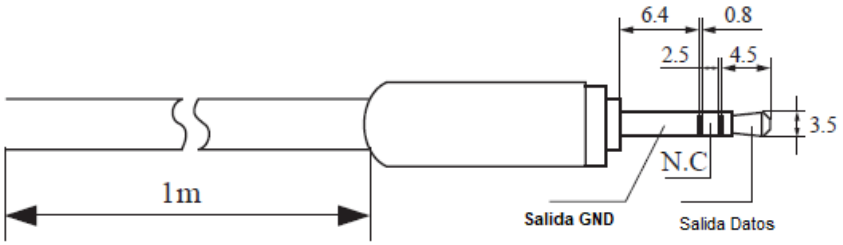


Figura 3-12 - Diagrama esquemático estándar de un conector Jack 3,5mm

Si bien, la Galileo también posee una entrada para fichas de este tipo, no es posible conectar directamente el sensor a la placa Intel Galileo, ya que según los datos técnicos del

sensor, este emite por su salida, una señal analógica que forma una sinusoidal, los valores de esta onda sinusoidal son tanto positivos como negativos, y ni Arduino, ni Galileo, pueden leer voltajes negativos.

Para poder realizar la conexión, utilizamos un amplificador operacional LM358, es un dispositivo amplificador electrónico de alta ganancia acoplado en corriente continua, se lo utiliza para manejar una gran variedad de rangos de voltaje, en nuestro caso, lo usaremos como inversor para que anule las ondulaciones negativas de la señal de datos del sensor SCT-013-000 permitiendo la conexión con Galileo (ver Figura 3-13).

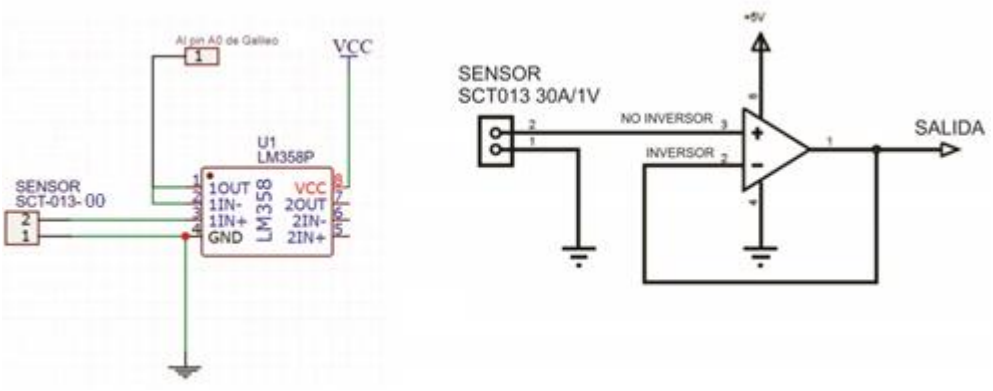


Figura 3-13 - Circuito de conexión del sensor SCT-013-000 y el operacional LM358

Con el operacional integrado al circuito, debemos conectar la señal de salida resultante del operacional a un pin analógico de la Galileo, usaremos el pin A0.

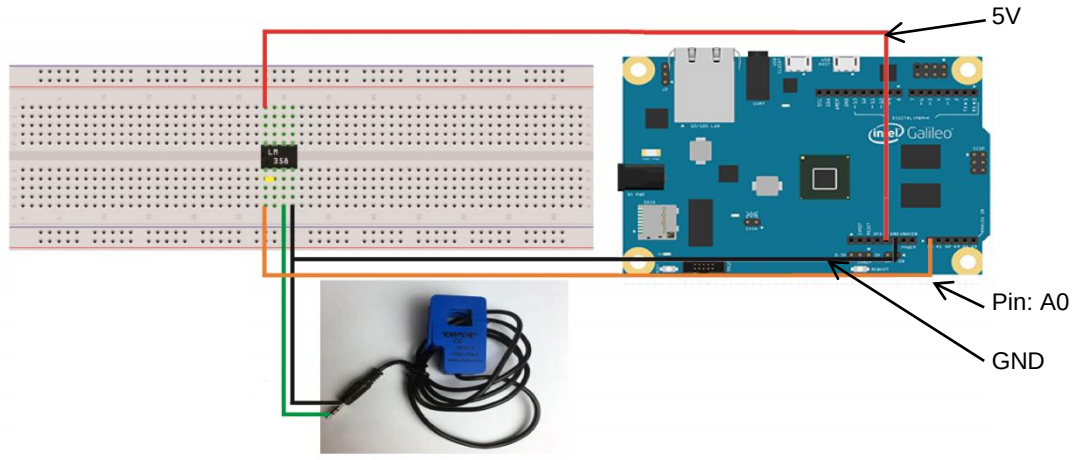


Figura 3-14 - Esquema de conexión del Sensor SCT-013-000 a la placa Intel Galileo

Una vez terminada la conexión, proseguimos a instalar la librería EmonLib en el IDE de Arduino, la cual fue creada para convertir los datos analógicos de diferentes sensores de corriente, entre ellos, el sensor SCT-013-000, a valores enteros que podemos utilizar en nuestro proyecto de trabajo final, es decir, nos facilita las fórmulas matemáticas para la conversión de estos datos.

Para instalar la librería, abrimos el IDE de Arduino (ver Capitulo 4, sección 4.2) y nos vamos a **Programa>Incluir Librería>Gestionar Librerías,** luego en la barra de búsqueda, colocamos "EmonLib", en los resultados, elegimos la librería, y le damos a instalar (ver Figura 3-15).

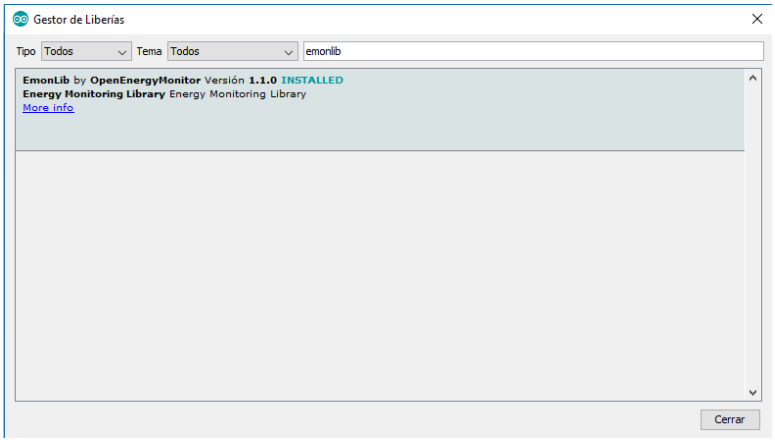


Figura 3-15 - Instalación de librería EmonLib en el IDE de Arduino

3.3 Módulo SIM800C GSM/GPRS

Para el envío de alertas del sistema a través de mensajes de texto GSM, se eligió el módulo SIM800C GSM/GPRS (Sistema Global para comunicaciones Móviles/Servicio general de paquetes vía radio), el modelo a utilizar es una placa del tipo escudo (Shield) que se conecta en la parte superior de la plataforma Intel Galileo.

Figura 3-16 - SIM 800C GSM/GPRS Shield

El SIM800C es un módulo GSM/GPRS en cuatribanda que funciona en frecuencias de GMS850MHz, ESGM900MHz, que son utilizadas actualmente por las operadoras de telefonía celular, volviendo al módulo compatible con cualquier operadora que esté trabajando en la región.[13]

En la tabla 3-6, se encuentran las especificaciones técnicas del módulo SIM 800C.

Información	
Modelo	SIM800C
GSM	850,900,1800 y 1900MHz
BT	(necesita soporte de Software)
FLASH	SIM800C (24 Mbit) SIM800C32 (32 Mbit)
RAM	32Mbit

Característica	Implementación
Fuente de alimentación	3,4V~4,4V
Ahorro de energía	Consumo típico de energía en modo sueño es 0,88mA (BS-PA-MFRMS=9)
Frecuencia de bandas	• Cuatribanda: GSM850, EGSM900, DCS 1800, PCS 1900. El SIM800C puede buscar las 4 frecuencias automáticamente. La frecuencia de bandas puede ser establecida con el comando AT, "AT+CBAND". • Obedece a GSM fase 2/2+
Poder de transmisión	• Clase 4 (2W) a GSM 850 y EGSM 900 • Clase 1 (1W) a DCS 1800 y PCS 1900
Conectividad GPRS	• GPRS slot multi-clase 12（defecto） • GPRS slot multi-clase 1~12 (opcional)
Rango de temperatura	• Operación normal: -40°C~+85°C • Temperatura de almacenaje: -40°C~+90°C
Datos GPRS	• Enlace de descarga de datos GPRS : máximo. 85.6 kbps • Enlace de subida de datos GPRS: máximo. 85.6 kbps • Esquema de codificación: CS-1, CS-2, CS-3 and CS-4 • Protocolo PAP para conexión PPP • Protocolo TCP/IP integrado. • Soporta PBCCH (Packet Broadcast Control Channel)
USSD	• Soporta USSD (Unstructured Supplementary Services Data)
SMS	• MT, MO, CB, Texto y modo PDU. • Almacenaje de SMS: en tarjeta SIM
Interface SIM	Soporta tarjeta SIM: 1,8V , 3V
Antena externa	Antena de tipo almohadilla
Características de Audio	Modos del codec de lenguaje: • Media tasa (ETS 06.20) • Tasa completa (ETS 06.10) • Tasa complete mejorada (ETS 06.50 / 06.60 / 06.80) • Tasa multi-adaptativa (AMR) • Cancelación de eco • Supresión de ruido
Puerto serial y puerto USB	**Puerto Serial:** • Por defecto, Puerto serial completo para modem • Puede usar comandos AT o flujo de datos • Soporta RTS/CTS handshake por

	hardware y encendido/apagado por control de flujo de software • Habilidad multiplex de acuerdo al protocolo GSM 07.10 Multiplexer • Autobaudio soporta tasas de baudio desde 1200 bps a 115200bps • Firmware actualizable **Puerto USB:** • USB_DN y USB_DP • Puede ser utilizado para debugging y actualización de firmware
Manejo de directorio telefónico	Soporta directorios del tipo: SM, FD, LD, RC, ON, MC
Herramientas de aplicación SIM	GSM 11.14 lanzado 99
Características físicas	Tamaño: 17.6*15.7*2.3mm Peso: 1,3g
Actualización de firmware	Puerto serial modem o USB (se recomienda USB)

Tabla 3-6 - Especificaciones detallas del módulo SIM800c GSM/GPRS

Por su compatibilidad con Galileo, fácil conexión a la plataforma, diseño amigable que nos permite seguir utilizando todos los pines de la placa y precio accesible, se decantó por el uso de este módulo, ya que cumple con las necesidades básicas requeridas del proyecto para la comunicación del sistema vía SMS.

3.3.1 Conexión y Comunicación

Antes de conectar el módulo SIM800c a la plataforma Intel Galileo, se tuvo que cambiar de posición los jumper de comunicación de dicho modulo (ver Figura 3-17) para que pueda transmitir y recibir datos a la Intel Galileo. La configuración por defecto del módulo, se utiliza en las placas Arduino las cuales pueden crear comunicaciones seriales mediante software usando librerías, la placa Intel galileo, no es compatible con estas librerías, porque estas conexiones son creadas por defecto usando hardware, sin la necesidad de estas.

Figura 3-17 - Posición de jumpers de SIM800C

Con los jumpers en la posición correcta, acoplamos la antena al módulo SIM800C, se conectó el cable de la antena a la ficha de GSM, después, se procede a conectar el módulo a la plataforma Intel Galileo, asegurando que los pines del escudo, coincidan con los pines

de la placa Galileo, se aplicó un poco de fuerza hasta que el modulo quede firme en su lugar, se tomando precaución de no dañar ambas placas durante el proceso (ver Figura 3-18).

Figura 3-18 - Conexión SIM800C con Intel Galileo

Con respecto al software necesario, las librerías que provee el IDE de Arduino, utilizan la comunicación serial mediante software, y como ya se ha mencionado, la placa Intel Galileo, no es compatible con estas características, la comunicación entre ambas placas se logró utilizando el serial por defecto de Galileo, a través de este, se envían los comandos AT necesarios para utilizar las funciones del módulo SIM800C.

3.3.2 Conjunto de comandos Hayes (comandos AT)

El conjunto de comandos Hayes es un lenguaje desarrollado por la compañía *Hayes Communications* que prácticamente se convirtió en estándar abierto de comandos para configurar y parametrizar módems. Los caracteres «AT», que preceden a todos los comandos, significan «Atención», e hicieron que se conociera también a este conjunto de comandos como **comandos AT**. [14]

La lista total de comandos AT es muy extensa, para los fines prácticos de este informe de proyecto de Trabajo Final, se abordará los aspectos básicos de la sintaxis, para después explicar solamente los comandos AT utilizados en el proyecto.

Convenciones y abreviaciones

A continuación se listara las abreviaturas a utilizar para hacer referencia a los elementos involucrados en el proceso de comunicación del módulo:

- ME(Mobile Equipment)
- MS(Mobile Station)
- TA(Terminal Adapter)
- DCE(Data Comunication Equipment, también puede referirse a un modem FAX o placa FAX)

Además, el equipo que controla el otro extremo de la comunicación serial, en nuestro caso la placa Intel Galileo, será referido con los siguientes términos:

- TE(Terminal Equipment)
- DTE(Data Terminal Equipment) o simplemente, la Aplicación, se refiere al código que se está ejecutando en el sistema embebido.

Sintaxis

El prefijo “AT” o “at” o “aT” o “At”, debe ser incluido al comienzo de cada línea de comando. Para terminar una línea, se coloca <CR> (Carácter de fin de línea).

Los comandos son seguidos de una respuesta que incluye generalmente:

“<CR><LF><respuesta><CR><LF>”

<CR> y <LF> juntos significan salto de línea.

Todos los comandos AT pueden dividirse sintácticamente en tres categorías: **“básico”, “parámetro S”** y **“extendido”.**

- **Sintaxis básica**

Los comandos AT tienen el formato de **“AT<x><>n”** o **“AT&<x><n>”**, donde “<x>” es el comando, y “<n>” es/son el/los argumento/s para ese comando. Un ejemplo de esto es “ATE<n>”, el cual indica al DCE si los caracteres recibidos deben ser repetidos al DTE dependiendo del valor de “<n>”, “<n>” es opcional, y se usara un valor por defecto si se excluye.

- **Sintaxis parámetro S**

Estos comandos AT tienen el formato **“ATS<n>=<m>”**, donde “<n>” es el índice del registro **S** a establecer, y “<m>” es el valor a asignar. “<m>” es opcional; si no se encuentra, entonces se asigna un valor por defecto.

- **Sintaxis Extendida**

Estos comandos pueden operar en muchos modos, en la tabla 3-7 se explican estas operaciones y sus respectivas respuestas.

Categoría	Sintaxis	Descripción
Comando de Prueba	AT+<x>=?	El ME retorna la lista de paramentos y el rango de valores establecidos con su correspondiente comando de Escritura o por procesos internos.
Comando de Lectura	AT+<x>?	Este comando retorna el conjunto de valores actuales del parámetro o los parámetros.
Comando de Escritura	AT+<x>=<…>	Este comando establece los valores de los parámetros definidos por el usuario.
Comando de Ejecución	AT+<x>	El comando de ejecución lee los parámetros de las no-variables afectadas por los procesos internos del equipo GSM.

Tabla 3-7 - Tipos de comandos AT y sus respuestas

Lista de Comandos AT utilizados

- **AT+CMGF:** Establece el modulo en formato SMS, para él envió y recibimiento de mensajes de texto en código ASCII.
- **AT+CMGD:** Borra mensajes de texto almacenados en la tarjeta SIM.
- **AT+CMGR:** Lee mensajes de texto.
- **AT+CMGS:** Envía un mensaje de texto.

Para una descripción más detallada de estos comandos, sus variables y respuestas, ver **Anexo 1**.

3.4 Módulo Relé/Relay

En primera instancia, para el manejo de las conexiones de los equipos de aire acondicionado en su encendido y apagado cuando el usuario lo requiera a través de las funciones del Sistema Embebido de Telemetría, se empleó el módulo de dos canales, versión 2R1B fabricado por Keyestudio, consiste en un módulo con dos bobinas de relés modelos Songle SRD-05VVDC-SL-C soldadas.

Figura 3-19 - Módulo relé 2 canales SRD-05VDC-SL-C

El módulo cuenta con 6 pines de conexión, dos de ellos cerrados mediante jumper (para uso general y configuración por defecto el datasheet recomienda no remover este jumper).

Las bobinas de relé cuentan con 3 borneras cada una para conectar los elementos que se desean controlar.

Se debe conectar las borneras **A** y **B**, si se desea que el circuito este cerrado (fluya corriente), y el relé lo abra al activarse, si se desea que el circuito este abierto (sin flujo de corriente) y el relé lo cierre al activarse, se debe conectar las borneras **A** y **C** (ver Figura 3-19). Para nuestro proyecto, un equipo de aire acondicionado, estará conectado con el circuito cerrado (**A y B**) en el relé número 1, y el otro en el relé número 2 con el circuito abierto (**A y C**), de esta manera se buscó alternar el funcionamiento entre ambos.

Su conexión a la placa Galileo se realizó con el pin IN1 al pin digital 4 el pin IN2 al pin digital 5, estos pines son los que envían las señales de activación de los relés número 1 y numero 2 respectivamente. El pin VCC se conectó con los 5V de la placa Galileo y el pin GND a tierra (ver Figura 3-20).

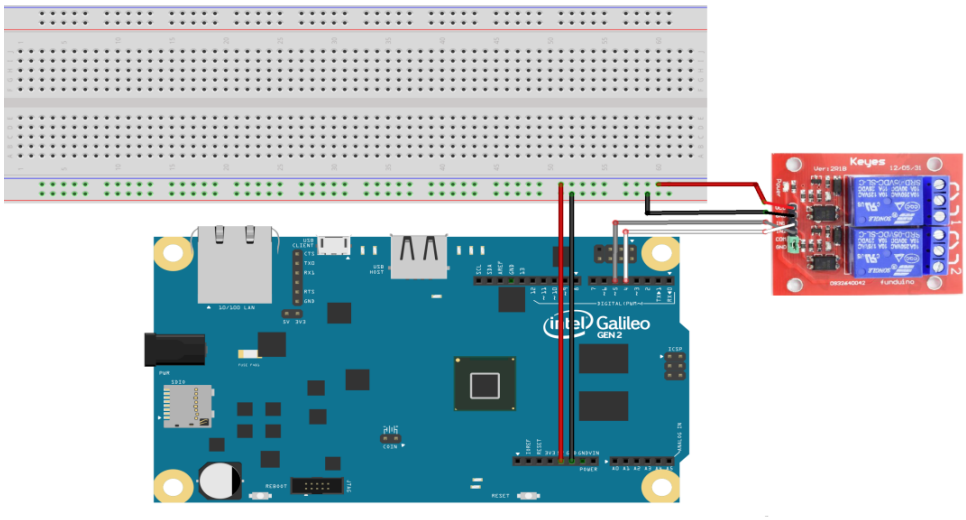


Figura 3-20 - Esquema de Conexión del Módulo rele 2 canales SRD-05VDC-SL-C

Las especificaciones técnicas del módulo rele SRD-05VDC-SL-C seran mencionadas en la tabla 3-8.

Sensibilidad de la bobina	Código de Voltaje de la Bobina	Voltaje Nominal (VDC)	Corriente Nominal (mA)	Resistencia de la bobina (Ω)±10%	Consumo de Poder (W)	Voltaje Pull-IN (VDC)	Voltaje Drop-OUT (VDC)	Voltaje Máximo Admisible (VDC)
SRD(Alta Sensibilidad)	03	03	120	25	Abt. 0.36W	75% Max.	10% Min.	120%
	05	05	71.4	70				
	06	06	60	100				
	09	09	40	225				
	12	12	30	400				
	24	24	15	1600				
	48	48	7.5	6400				
SRD(Estándar)	03	03	150	20	Abt. 0.45W	75% Max.	10% Min.	110%
	05	05	89.3	55				
	06	06	75	80				
	09	09	50	180				
	12	12	37.5	320				
	24	24	18.7	1280				
	48	48	10	4500	Abt.0.51 W			

Tabla 3-8 - Especificaciones detallas del módulo relé 2 canales SRD-05VDC-SL-C

Con respecto al software necesario, no se necesitó ninguna librería o configuración previa para utilizar el módulo de relé, ya que este responde perfectamente a las funciones por defecto de Arduino para activar o desactivar pines conectados.

3.5 Ensamblaje de los componentes

En la primera versión de este sistema de telemetría, se utilizó una placa de PCB (Bifenilos Policlorados) para soldar tanto el sensor de temperatura DHT11 como una ficha Jack hembra de 3,5mm para la conexión del sensor de corriente Alterna SCT-013, así también como la resistencia de 5KΩ, el diodo y el amplificador operacional LM358, por ultimo unas fichas del tipo clema (también conocida como bornera, ficha de empalme) para conectar los cables, a los pines de la plataforma Intel Galileo.

El diagrama del circuito utilizado para soldar los componentes está representado en la Figura 3-21.

Los colores que se emplearon en los cables de conexiones se indican en la Tabla 3-9

Color	Descripción	Pin de Conexión en Galileo
Negro	GND	GND
Rojo	5V	5V
Amarillo	Datos Sensor SCT-013-000	A0
Verde	Datos IN DHT11	D2
Azul	Datos OUT DHT11	D3
Gris	Datos IN Relé 2	D5
Blanco	Datos IN Relé 1	D4

Tabla 3-9 - Tabla de Colores en cables de conexión

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

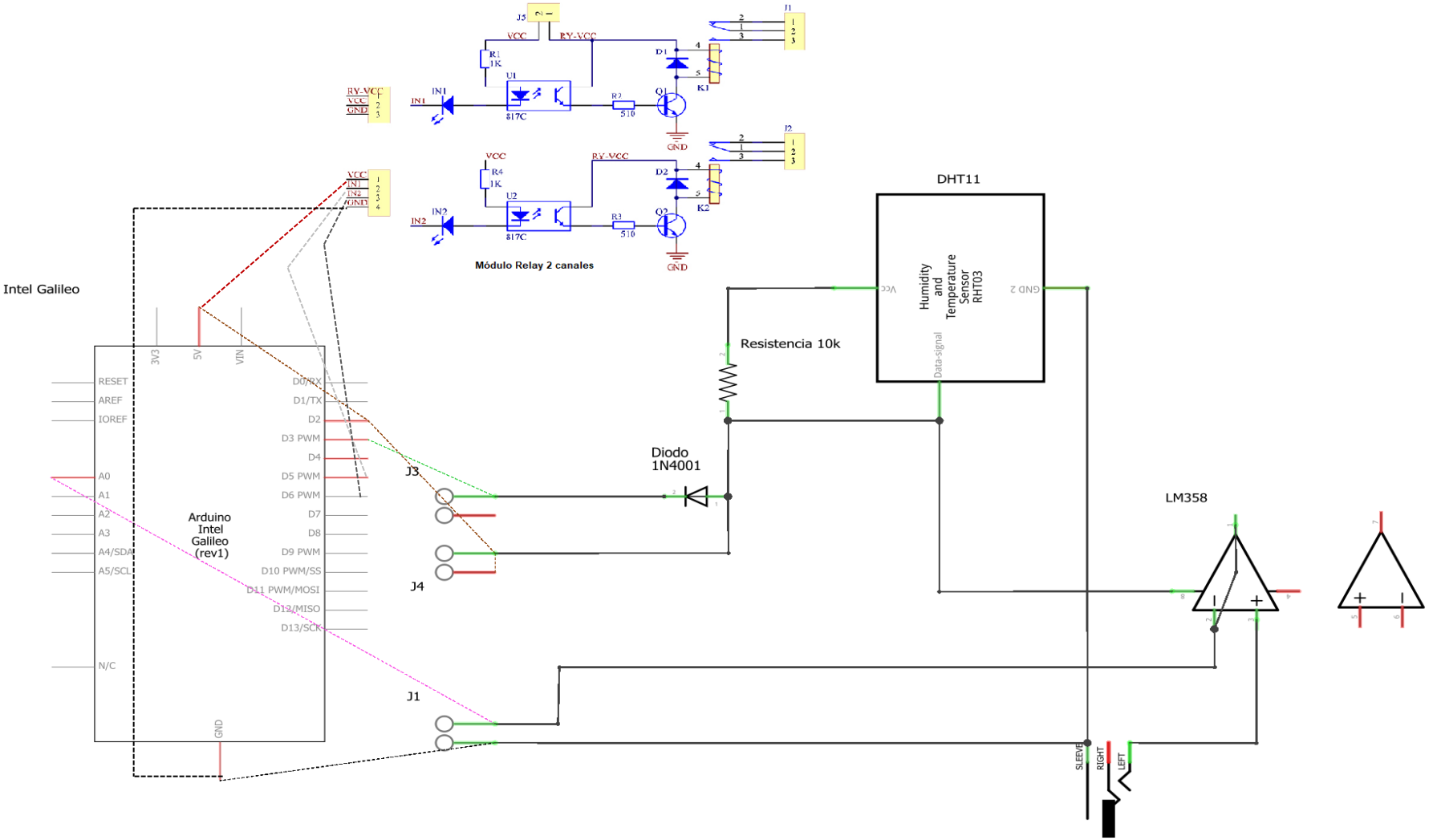


Figura 3-21 - Diagrama Esquemático

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

En la Figura 3-22, se muestra el resultado de los componentes solados a la PCB perforada, junto a las conexiones de los cables y sus colores.

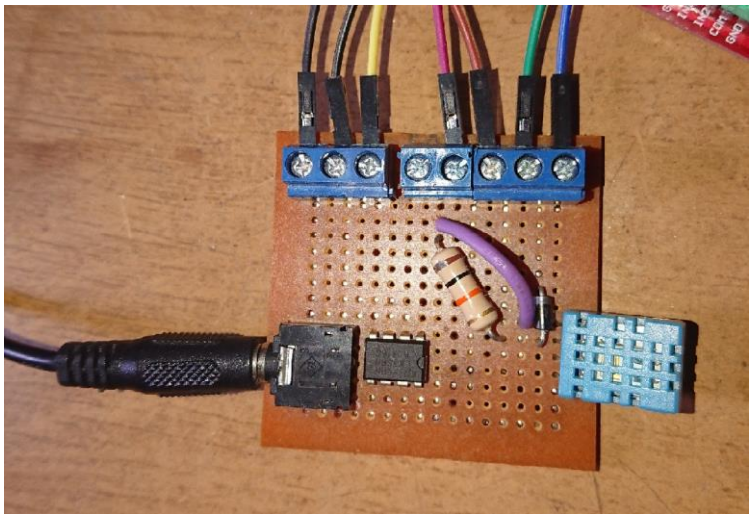


Figura 3-22 - Conexión de componentes en PCB perforada

Para el montaje de los elementos, se eligió utilizar un Gabinete de plástico, ya que brindará la protección y el aislamiento que los componentes del sistema necesitan, también el material del gabinete nos permite realizar cambios en su estructura como rendijas de ventilación, orificios de acceso a los puertos de conexión y acceso a los botones de reinicio o también es posible realizar encargos con medidas y características personalizadas. El modelo del Gabinete puede apreciarse en la Figura 3-23.

Figura 3-23 – Gabinete Plástico

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Una vez terminadas las modificaciones en el gabinete, se colocó un piso de policarbonato en la base del mismo para dar más libertad en la ubicación de los componentes, como se puede observar en la Figura 3-24.

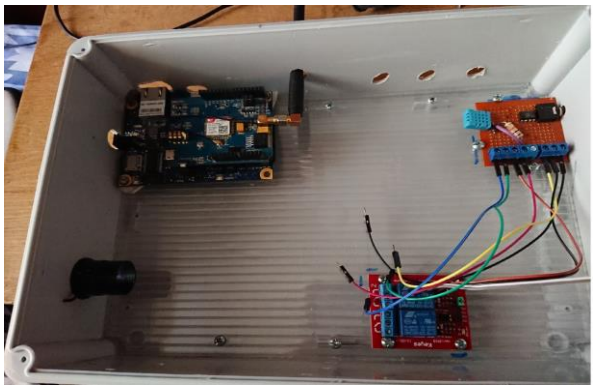


Figura 3-24 - Organización de los componentes dentro del gabinete

Se realizó las conexiones del cableado como muestra la Figura 3-25.

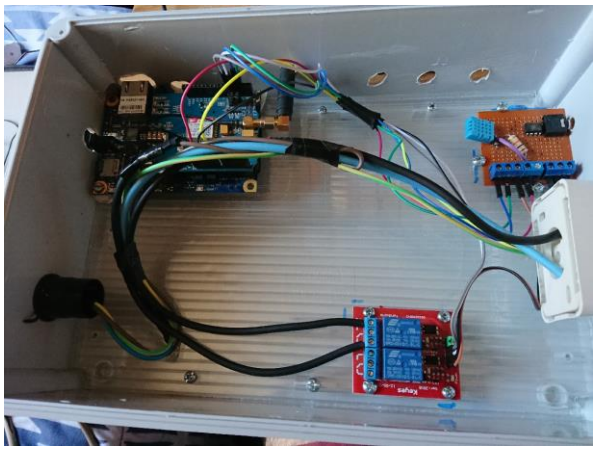


Figura 3-25 - Cableado del dispositivo

Finalmente, se agregó el botón de Reset para el reinicio manual del sistema.

Figura 3-26 - Botón de Reset en el gabinete

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

CAPITULO 4

4 Software del Sistema Embebido

En este capítulo, primero se describirá el proceso de descarga y configuración del software necesario para el sistema y sus componentes, luego se explicara el desarrollo del firmware del Sistema embebido de Telemetría.

4.1 Instalación de Paquetes en Linux Yocto 1.0.4

Una vez funcionando el SO en nuestra placa Intel Galileo, procedemos a instalar los programas que necesitamos para la comunicación Web del sistema de Telemetría, para hacer esto, necesitamos hacer uso de la consola del SO Linux Yocto, primero debemos conectar la placa con un cable ETH (ver figura 4-1) y la computadora a una Red LAN que tenga salida a Internet.

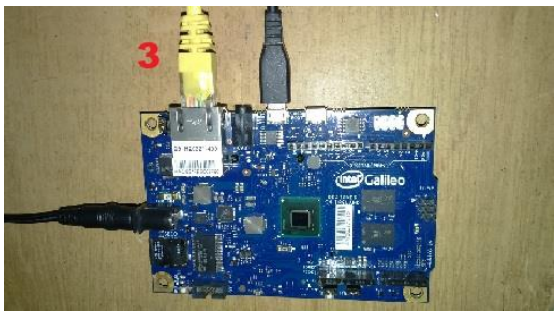


Figura 4-1 - Conexión ETH de la placa Intel Galileo.

Utilizaremos el programa llamado PuTTY en su versión 0.7, para abrir una comunicación de tipo serial SSH (Secure Shell) entre nuestra PC y el SO Linux Yocto dentro de la placa Intel Galileo. Una vez instalado el programa, lo ejecutaremos, en su ventana de configuración, debemos colocar la dirección IP que tiene asignada la placa Intel Galileo en la red LAN, el puerto por defecto para una comunicación serial SSH es el 22(ver figura 4-2).

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

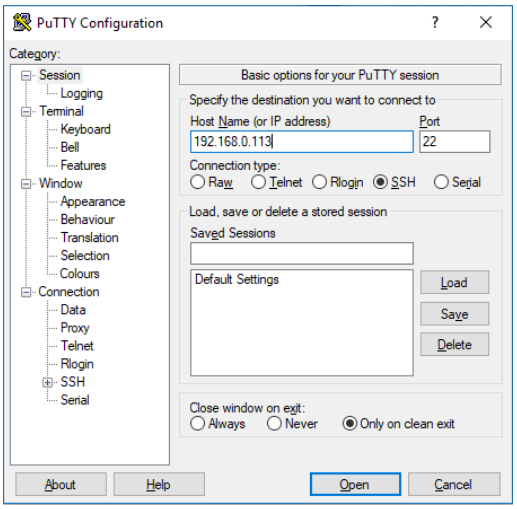


Figura 4-2 - Configuración de PuTTY 0.70 para comunicación SSH

Una vez establecida de manera exitosa la comunicación con la placa Intel Galileo, pudimos acceder a la consola del SO, lo primero que debemos introducir en la consola es el usuario y la contraseña, por defecto, “root” en ambos campos, hecho esto, se logró hacer uso de los comandos de consola, el primer comando a introducir es **opkg update**, y cuando terminó su ejecución, introducimos **opkg upgrade**, estos comandos descargaran la última versión de la base de datos del repositorio de paquetes y actualizará los paquetes actualmente instalados de Linux Yocto, ahora podremos instalar los paquetes de los programas que vamos a necesitar. La Figura 4-3, muestra la consola de Linux después de introducidos los comandos, en nuestro caso, este paso se realizó previamente.

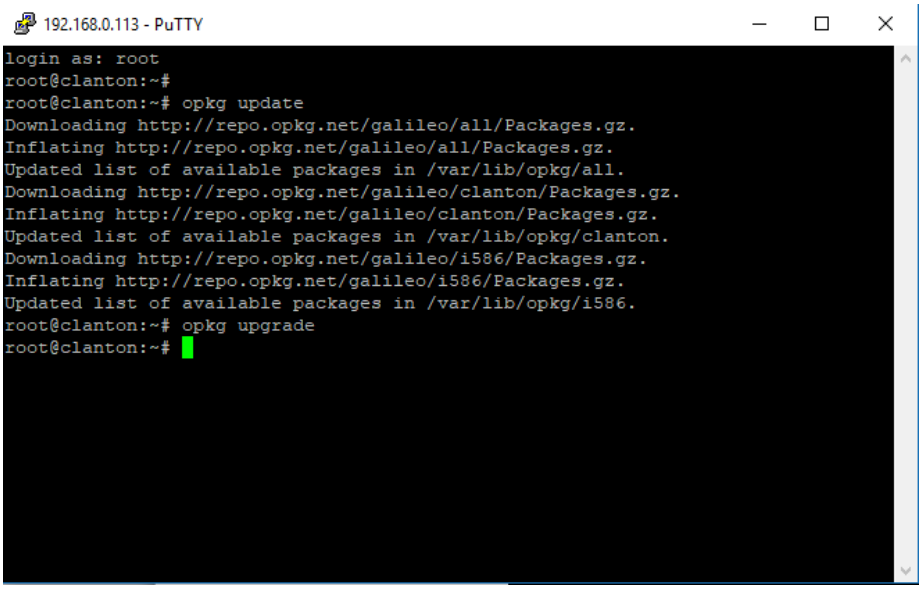


Figura 4-3 - Consola Shell del SO Linux Yocto.

4.1.1 Instalación servidor Apache HTTP

El servidor Apache, es un servidor web HTTP (Hyper Text Transfer Protocol) de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y

otras, que implementa el protocolo HTTP/1.1. Para instalar Apache y todos sus archivos necesarios, debemos introducir en la consola de Linux Yocto los siguientes comandos preferiblemente en el orden que son listados a continuación, luego de introducir un comando, debemos esperar que terminen las descargas e instalaciones necesarias para después, introducir el siguiente.

- ***opkg install apache2*** *:* Instala servidor Apache HTTP en su última versión
- ***opkg install apache2-dbg*** *:* Instala archivos de debugging
- ***opkg install apache2-dev*** : Instala archivos de apoyo a desarrolladores
- ***opkg install apache2-doc*** *:* Instala archivos de documentación
- ***opkg install apache2-scripts*** *:* Instala scripts de configuración para el servidor

En la Figura 4-4, se puede apreciar la consola después de introducir los comandos, como este paso ya fue realizado anteriormente, la consola devuelve el mensaje que los paquetes ya están actualizados e instalados.

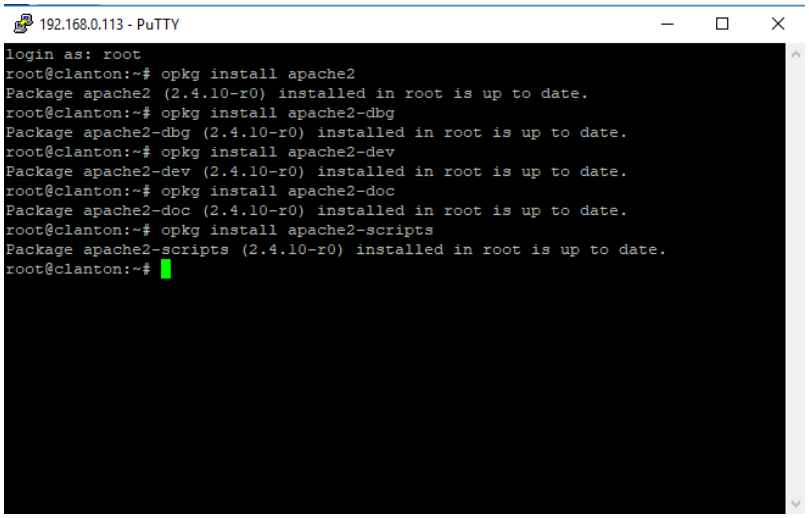


Figura 4-4 - Comandos para Instalación de Apache HTTP

Al finalizar la ejecución de todos los comandos, debemos introducir las siguientes líneas en la consola para ejecutar Apache y configurarlo para que inicie durante el arranque del sistema operativo.

- cd /etc/init.d/ : Accedemos al directorio donde se encuentra el ejecutable de Apache
- ./apache2 start : Iniciamos Apache2 y queda configurado para ejecutarse en el arranque del SO.

A continuación, se tuvo que corregir una configuración por defecto de Apache que no es compatible con el Linux Yocto y de no hacerlo, no se ejecutará el servidor Apache. El error consistía en una mala ubicación de un directorio dentro de los archivos de la tarjeta SD. Para solucionar el problema, se utilizó el programa WinSCP, el cual nos permite mediante una conexión SCP (Secure Copy Protocol o Simple Communication Protocol es un medio de transferencia segura de archivos informáticos entre un host local y otro) acceder al contenido de la micro SD, mientras se está ejecutando el Linux Yocto.

Ejecutamos el programa WinSCP, introducimos la dirección IP de la placa Intel Galileo, seleccionamos el protocolo en SCP, el puerto 22, en usuario y contraseña se coloca en ambos campos 'root', y abrimos la conexión (ver Figura 4-5).

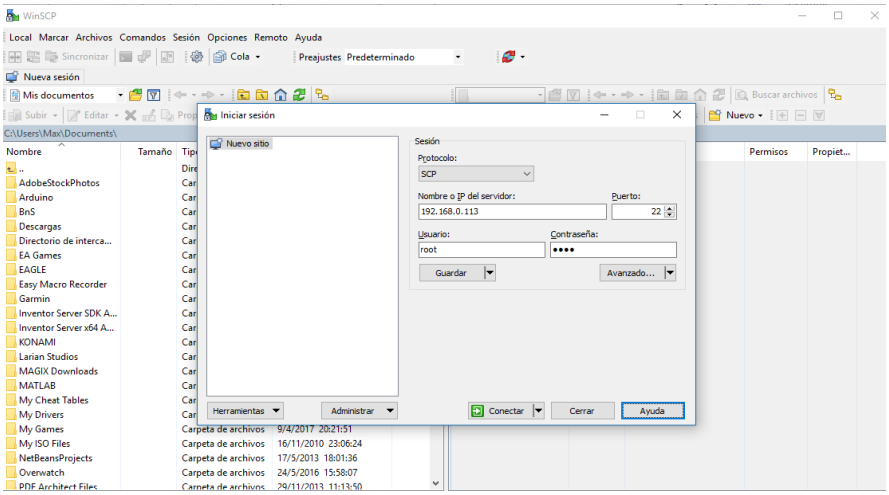


Figura 4-5 - Configuración de WinSCP para abrir comunicación SCP

Utilizando la navegación del programa, nos dirigimos al directorio raíz del SO Linux Yocto, la carpeta causante del problema es ***var***, se encuentra por defecto en el directorio raíz como un acceso directo, lo que limita el acceso de algunos programas como Apache a la misma, porque Apache necesita acceder a los archivos de tipo registro (log) que allí se encuentran, la solución es mover todo el contenido de la carpeta ***var***, al directorio raíz, la figura 4-6 muestra cómo se debe ver el directorio raíz, luego de mover el contenido de la carpeta **var**.

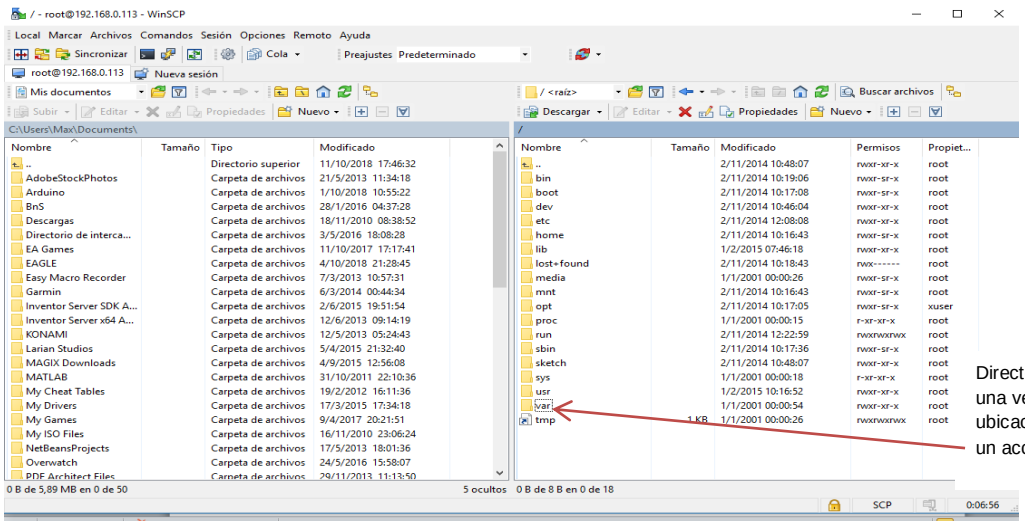


Figura 4-6 - Directorios en la carpeta Raíz del SO Linux Yocto

4.1.2 Instalación MySQL

Para poder instalar el conjunto de paquetes MySQL, se procedió de manera similar a los paquetes de Apache, en la consola se debe introducir los siguientes comandos:

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

- **opkg install mysql5:** Instala el paquete MySQL 5.
- **opkg install mysql5-server:** Instala el conjunto de paquetes de MySQL 5 para servidores.
- **opkg install mysql5-client:** Instala el conjunto de paquetes de MySQL 5 para clientes.
- **opkg install mysql5-dbg:** Instala el conjunto de paquetes de MySQL5, para debugging
- **opkg install libmysqlclient16:** Instala las librerías del cliente MySQL, en su versión 16.
- **opkg install libmysqlclient-dbg:** Instala las librerías del cliente MySQL para debugging.
- **opkg install libmysqlclient-dev:** Instala las librerías del cliente MySQL para desarrolladores.
- **opkg install libmysqlclient-r-dev:** Instala las librerías del cliente MySQL para desarrolladores.
- **opkg install libmysqlclient-r-staticdev:** Instala las librerías del cliente MySQL para desarrolladores.
- **opkg install libmysqlclient-r16:** Instala las librerías del cliente MySQL, para desarrolladores en su versión 16.
- **opkg install libmysqlclient-staticdev:** Instala las librerías del cliente MySQL, para desarrolladores.

Se muestra en la Figura 4-7, los resultados de algunos de los comandos, luego de sus ejecuciones.

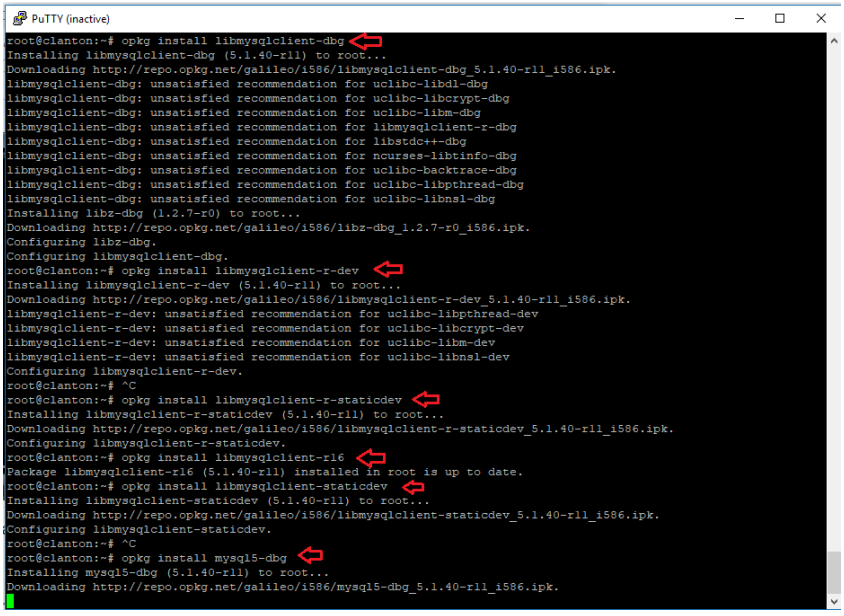


Figura 4-7 - Comandos para la instalación de MySQL

Los mensajes de error que nos muestra la consola, fueron debidos a que el instalador trato de crear y darle permisos al usuario “mysql”, Yocto Linux, por defecto, no permite modificaciones a los usuarios, también, el servicio MySQL, no puede iniciar, por mala

configuración de los sockets de conexión, para solucionar estos problemas, se tuvo que crear manualmente al usuario "mysql" y luego configurar manualmente el socket, para esto se realizaron los siguientes pasos, primero se ejecutaron estos comandos en la consola:

- **useradd -m -r -s /bin/true mysql:** Crea el usuario mysql
- **chown -R mysql /var/mysql:** Concede permisos al usuario mysql en ese directorio
- **chown -R mysql /var/lib/mysql:** Concede permisos al usuario mysql en ese directorio

Luego, para corregir el error de la configuración del socket de conexión, se modificó el archivo ubicado en la siguiente ubicación: **/etc/my.cnf**

En la Figura 4-8, se muestra como quedo el archivo de configuración de MySQL, luego de corregir la ubicación de los socket.

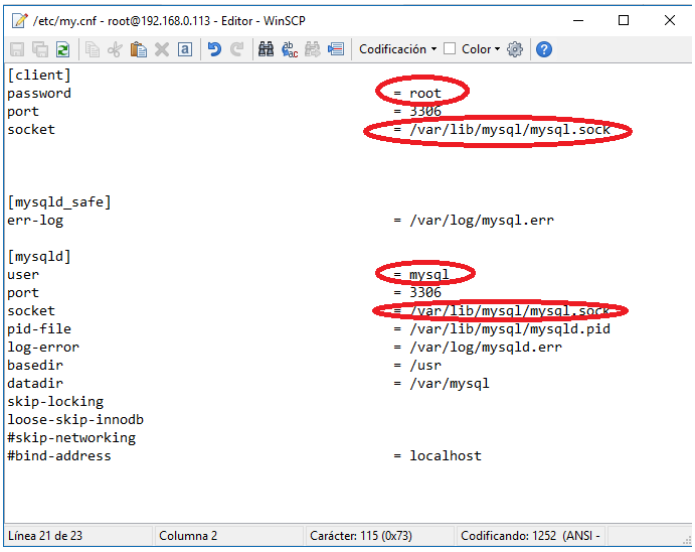


Figura 4-8 - Archivo de Configuración de MySQL

Finalmente, se ejecutó el siguiente comando para que MySQL pudiera crear los directorios e iniciar los servicios correspondientes que antes no pudo, y comenzar a funcionar:

- **mysql_install_db:** Instala e inicia todos los servicios MySQL

➢ Creación de la Base de datos en MySQL

Con la aplicación MySQL funcionando en la plataforma Intel Galileo, ejecutamos el comando:

- **mysql:** Activa el modo gestor de base de datos MySQL, ahora se puede ejecutar comandos SQL para interactuar con las bases de datos.

Se creó nuestra propia base de datos llamada **temperatura_humedad**, para esto se introdujo los siguientes comandos SQL, el comando CREATE, crea la base de datos, el comando USE, la selecciona para manipularla.

- **CREATE DATABASE temperatura_humedad;**

- **USE tempratura_humedad;**

Finalmente para crear la tabla que vamos a usar para los datos recolectados por el sensor de temperatura y humedad DHT11, se creó una tabla **reportes** (ver tabla 4-1)**,** con las siguientes características:

Tabla	**REPORTES**	
Columnas		
Nombre	**Tipo**	**Descripción**
id	INT(4)	Número de identificación de la fila, autoincremental. **Valor clave**
temperatura	varchar(4)	Valor de temperatura recolectado por el sensor.
humedad	varchar(4)	Valor de humedad recolectado por el sensor.
hora	time	Hora que se registró el valor
fecha	date	Fecha que se registró el valor

Tabla 4-1 - Descripción de la tabla Reportes en MySQL

- **Comando SQL:**

CREATE TABLE reportes (id int(4) NOT NULL AUTO_INCREMENT, temperatura VARCHAR(4) NOT NULL, humedad VARCHAR(4) NOT NULL, hora TIME NOT NULL, fecha DATE NOT NULL, PRIMARY KEY (id));

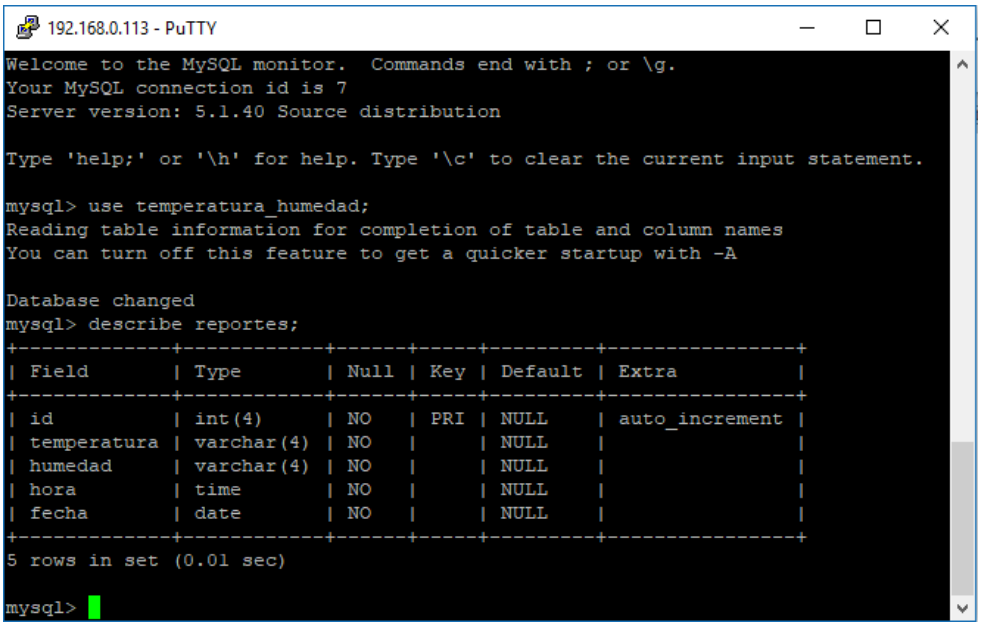


Figura 4-9 - Tabla reportes creada en MySQL

4.2 Instalación del Entorno Arduino IDE

Para poder hacer uso del IDE de Arduino, debemos descargar de la página de Arduino en su sección de Software la última versión compatible con el Sistema Operativo utilizado, una

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

vez hecho esto, ejecutamos el archivo descargado Arduino-(versión)-(sistema Operativo).exe, para este proyecto, Arduino-1.8.5-windows.exe y seguimos las instrucciones de instalación.

Se puede observar (ver Figura 4-10), que al iniciar un nuevo archivo de código, el IDE agregará por defecto en las líneas del mismo, dos funciones que son **void setup ()** y **void loop()**, estas funciones son de carácter obligatorio, y deben estar presente en cualquier código que se deseé implementar utilizando plataformas de la familia Arduino, incluida la Intel Galileo.

La función **setup(),** es llamada cuando el sketch inicia, Se utiliza para inicializar variables, modos a los pin, comenzar a usar librerías, etc. La función **setup()**, solo se ejecutara una vez durante cada inicio, o reseteo de la plataforma.[11]

La función **loop(),** es una función bucle, y se comporta exactamente como uno, repitiéndose constantemente mientras la plataforma esté en funcionamiento, se utiliza para ejecutar el código principal, que hará uso de todas las funciones de la placa y los componentes conectados[11]

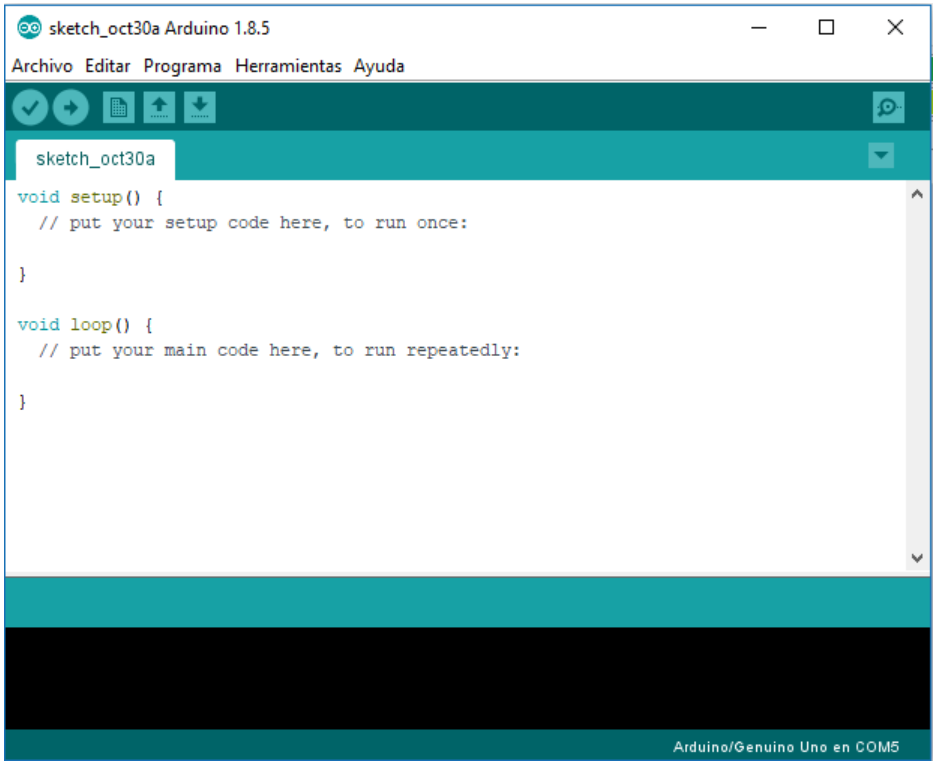


Figura 4-10 - Arduino IDE 1.8.5

Ahora debemos descargar la familia de placas Intel x86 en el entorno de desarrollo, para que el IDE reconozca nuestra Intel Galileo, para hacer esto, debemos ir a **Herramientas>Placa>Gestor de Tarjetas**, luego en la barra de búsquedas escribimos **Intel**, finalmente seleccionamos la opción **Intel i586 Boards** y procedemos a la instalación (ver Figura 4-11).

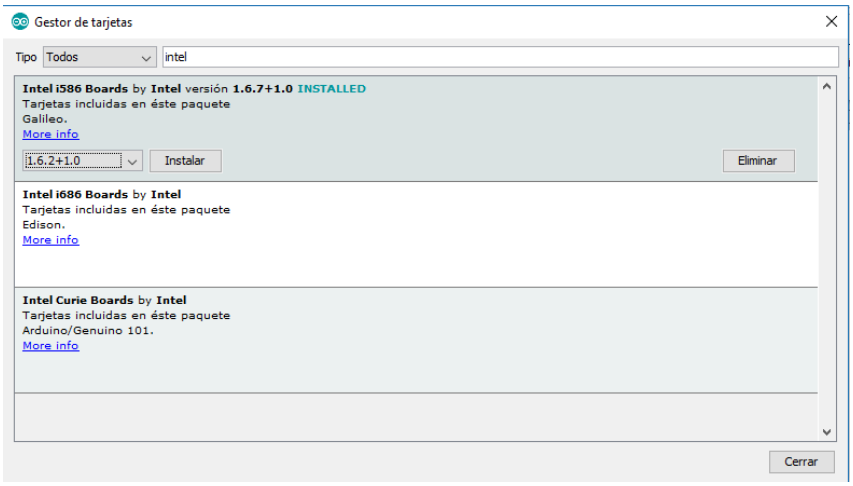


Figura 4-11 - Instalación de placas Intel i586 en el IDE de Arduino

La ultima configuración en el IDE de Arduino para poder trabajar con la plataforma Intel Galileo la realizamos en **Herramientas>placas>Intel® Galileo**, de esta manera queda configurado el compilador del IDE de Arduino para trabajar con nuestra placa Intel Galileo (ver Figura 4-12)

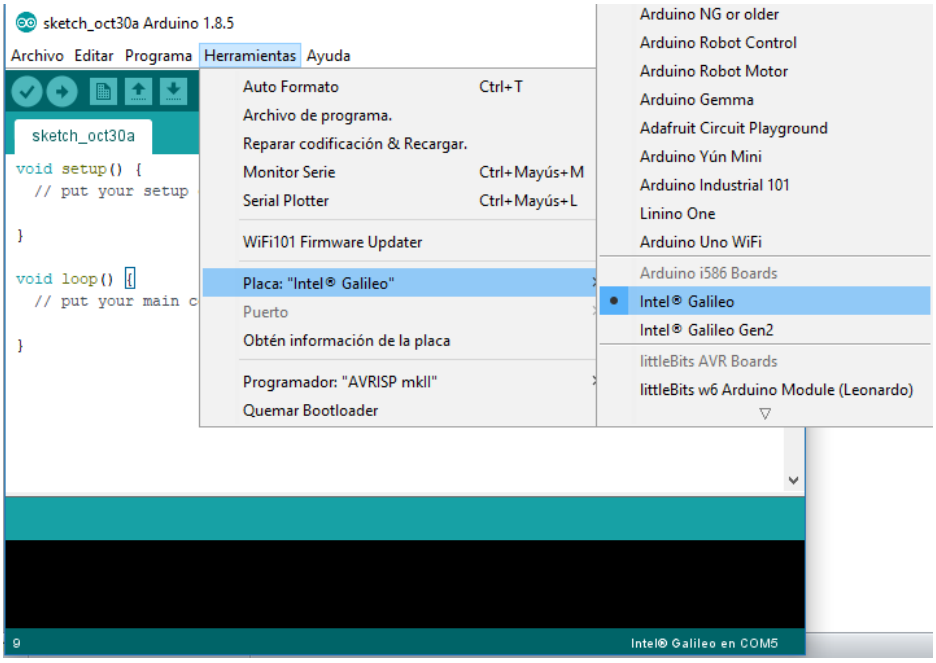


Figura 4-12 - Selección de la placa Intel Galileo en el IDE de Arduino

4.3 Sistemas embebidos de Telemetría

El firmware, es la parte lógica del sistema embebido que se encargará de procesar los datos de los sensores, enviar las alertas de mensajes de texto, recibir e interpretar los mensajes de texto de los usuarios para responder adecuadamente y finalmente, administrar la aplicación web del sistema, en la cual se mostrará los datos de los sensores.

A Continuación se explicará en detalle el proceso de desarrollo del firmware utilizado en el Sistema embebido de Telemetría.

El firmware y su estructura lógica, puede ser graficado en una estructura de capas (Figura 4-13).

- **Aplicación**: Esta capa es donde se encuentra la lógica principal del sistema embebido, el sketch cargado por el IDE Arduino, es el encargada de la recopilación de los datos de los sensores, manejo del módulo SMS/GPRS y él envió de alertas.

- **Software del Sistema:** En esta capa, se encuentra el sistema operativo Linux Yocto, y es la base del sistema, aquí se encuentran los drivers y las funciones básicas del funcionamiento de la Intel Galileo, también provee funciones de servidor web con Apache Web Server y soporte de lenguaje HTML, PHP y almacenamiento de base de datos en MySQL.

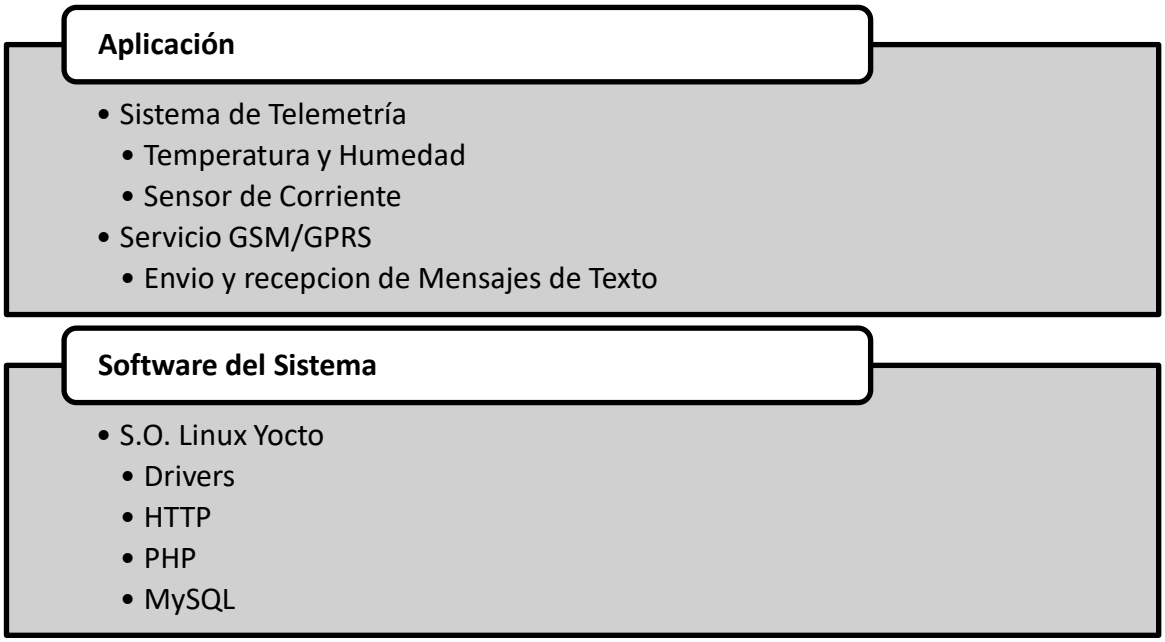


Figura 4-13 - Arquitectura del Firmware

4.3.1 Aspecto General

Para describir algunas de las características generales de la aplicación se siguieron las directrices del estándar “IEEE Recomended Practice for Software Requirements Specification ANSI/IEEE 830 1998”.

❖ **Alcance**

El desarrollo es motivado por la necesidad de un sistema embebido con la capacidad de recolectar datos del estado del data center a cargo de la D.T.I.C. de la U.N.Ca. , y que estos datos puedan ser consultados en cualquier momento para tomar las medidas correspondientes en caso de emergencias.

4.3.2 Descripción General

En el siguiente apartado de describen los detalles generales del sistema embebido. Se explican sus principales características, funciones y otros factores involucrados en su desarrollo.

❖ Perspectiva del Producto

El firmware se preparó para poder operar sobre una plataforma de hardware que soporte el Sistema Operativo Linux Yocto embebido.

❖ Funciones del Sistema

Las principales funcionalidades que del Sistema Embebido brindará son:

- Mostrar un menú por mensaje de texto con las funciones del sistema.
- Consultar Temperatura y Humedad del ambiente por mensaje de texto SMS.
- Consultar Temperatura y Humedad del ambiente por aplicación web.
- Consultar temperatura y humedad del ambiente recolectados en otros días por aplicación web.
- Consultar la dirección IP del dispositivo en la red de área local a la cual está conectado por mensaje de texto SMS.
- Consultar si se desea monitorear un aire acondicionado.
- Dejar de monitorear el equipo de aire acondicionado si recibe la orden de hacerlo.
- Consultar el estado del equipo de aire acondicionado por mensaje de texto SMS.
- Consultar el estado del equipo de aire acondicionado por aplicación web.
- Enviar alerta por mensaje de texto, si la temperatura es demasiado alta en sala.
- Enviar alerta por mensaje de texto, si el equipo de aire acondicionado deja de funcionar.
- Brindar en la aplicación web un servicio de sesiones por cuentas de usuario.
- Permitir acceso a todas las funciones de la aplicación web si se inició sesión correctamente.
- Permitir visualizar, ingresar o modificar los números de celular con al cual se comunica el sistema por mensajes de texto SMS.
- Permitir modificar los datos de inicio de sesión.
- Permitir alternar el funcionamiento de 2 quipos de aire acordonado por mensaje de texto.

❖ Características de los Usuarios

El sistema tiene como usuario objetivo al personal a cargo del data center de la D.T.I.C., se espera que el/los encargado/s, tenga/n conocimiento técnico para poder operar el sistema embebido, reconocer su estructura, las conexiones que deben realizarse y comprender los valores que se obtienen de los sensores para tomar las acciones que se requieran ante contingencias.

❖ **Restricciones**

El/los encargado/s del data center, debe/n proveer acceso a la red de área local y a internet para que el sistema pueda usar las funciones web.

Por cuestiones de seguridad, ya que el sistema podrá encender o apagar equipos de aire acondicionado, cuya integridad debe ser protegida, el sistema embebido, solo enviará y responderá a los mensajes de texto SMS a un número de celular en particular, programado previamente en el firmware, el mismo podrá actualizarse mediante mantenimiento del sistema.

4.3.3 Requisitos Específicos

En este apartado se presentaran los requisitos funcionales, que deberán ser satisfechos por el firmware y la aplicación web. Todos los requisitos que serán listados, son esenciales, significa que el sistema embebido debe cumplirlos para considerarse óptimo.

➢ **Requisitos Funcionales**

- El sistema permitirá almacenar 3 números de celular, uno perteneciente al administrador de la sala, y los otros dos a personal auxiliar.
- Al inicio, el sistema deberá enviar un mensaje de texto al administrador solicitando si se desea monitorear el aire acondicionado para activar las alertas, y a todos los números almacenados en su memoria para informar que está en funcionamiento.
- El sistema no responderá a mensajes de texto que no provengan de los números de celular registrados en el mismo.
- El sistema no responderá a mensajes de texto que no contengan las instrucciones que se indicaron para su funcionamiento.
- El sistema comenzará a monitorear el aire acondicionado conectado si recibe la orden de mensaje del administrador.
- El sistema dejará de monitorear el aire acondicionado conectado si recibe la orden específica por mensaje de texto SMS del administrador.
- El sistema deberá responder al mensaje de texto SMS que solicita el menú de opciones, con otro mensaje de texto SMS con el menú de opciones en su contenido.
- El sistema deberá responder al mensaje de texto SMS que solicita la temperatura y humedad del ambiente con un mensaje de texto SMS con los datos recolectados en ese momento por el sensor de temperatura y humedad en su contenido.
- El sistema deberá responder al mensaje de texto SMS que solicita el estado del equipo de aire acondicionado con un mensaje de texto SMS con el estado actual de los aires acondicionados en su contenido.
- El sistema deberá responder al mensaje de texto SMS que solicita la dirección IP del dispositivo con un mensaje de texto SMS con la dirección IP que tienen asignado el sistema en la red de área local, si está conectado a una.
- El sistema deberá enviar un mensaje de texto SMS a todos los números registrados informando en modo de alerta, que la temperatura alcanzo valores no deseados y repetirá el envío cada 30 minutos hasta que la temperatura alcance valores aceptables.

- El sistema deberá enviar un mensaje de texto SMS a todos los números registrados informado en modo de alerta, que el equipo de aire acondicionado dejo de funcionar y solicitando también, al administrador, en otro mensaje una orden específica, si se desea encender el aire acondicionado de repuesto.
- El Sistema repetirá el envío del mensaje de texto SMS de alerta sobre el estado del aire acondicionado cada 30 minutos, si no recibe respuesta del administrador.
- El sistema deberá encender el aire acondicionado de repuesto, si recibe la orden específica luego de haber enviado el mensaje de alerta sobre el estado del aire acondicionado.
- El sistema no actuará si recibe la orden específica de encender el aire acondicionado de repuesto, sin antes haber enviado el mensaje de alerta sobre el estado del aire acondicionado.
- El sistema almacenará los datos de temperatura y humedad recolectados por el sensor de temperatura y humedad en su base de datos.
- El sistema mostrará una aplicación web donde el administrador puede acceder mediante inicio de sesión a las funciones web del sistema.
- Si el administrador inició sesión correctamente, el sistema mostrará en la aplicación web una página donde se podrá consultar los datos de temperatura y humedad del ambiente y el estado de los aires, a través de gráficos.
- Si el administrador inició sesión correctamente, el sistema mostrará en la aplicación web, en la página de los gráficos, un gráfico con registros históricos, donde se podrá seleccionar una fecha deseada para conocer los datos recolectados en ese día.
- Si el administrador inicio sesión correctamente, el sistema mostrará en la aplicación web una página donde se podrá visualizar, ingresar o modificar los números de celular con los cuales se comunica.
- Si el administrador inicio sesión correctamente, el sistema mostrará en la aplicación web una página donde se podrá modificar los datos de inicio de sesión e la cuenta.
- El sistema no debe permitir acceder a las funciones de la aplicación web si se ingresan incorrectamente los datos de inicio de sesión.
- El sistema reiniciará todas sus funcionalidades, si el botón de reboot es presionado.
- El sistema reiniciará todas sus funcionalidades, si recibe la orden específica vía mensaje de texto SMS del administrador.

➢ **Requisitos No Funcionales**

- El sistema está pensado para utilizarse con un número de celular correspondiente al administrado del data center y dos números adicionales pertenecientes a personal auxiliar.
- El sistema está pensando para utilizarse en una sala data center, que puede tener equipos de aire acondicionado.
- El sistema debe ser operado únicamente por el personal a cargo del data center.
- La información que recolecta el sistema, debe ser clara y concisa a la hora de ser consultada por el personal del data center.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

4.3.4 Modelo Esencial

Este modelo representa gráficamente lo que el sistema debe hacer para satisfacer los requerimientos del usuario, mostrando lo mínimo posible acerca de cómo se implementa. Aquí se encuentran el **Modelo Ambiental,** y el **Modelo de Comportamiento**.

- **Modelo Ambiental**

Este modelo tiene como objetivo, marcas los límites entre nuestro sistema y los sistemas que lo rodean, es decir, delimitar las fronteras entre nuestro sistema y su ambiente.

- **Diagrama de Contexto**

Uno de los componentes que conforman al Modelo Ambiental es el Diagrama de Contexto. Este diagrama enfatiza algunas características importantes del sistema como son las personas, organizaciones y sistemas con lo que se comunica el sistema. También, los datos que el sistema produce y los datos que recibe del exterior y debe procesar de alguna forma.

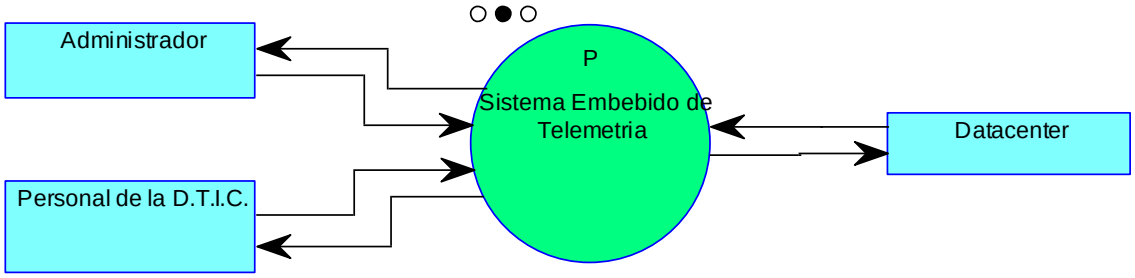


Figura 4-14 - Diagrama de Contexto del Sistema Embebido de Telemetría

- **Lista de Eventos**

A continuación se listan los eventos que ocurren en el exterior que desencadenan respuestas del sistema:

1) Al iniciar el sistema, este envía un mensaje de texto al administrador y al personal del data center con las primeras opciones y la IP del dispositivo.
2) El administrador o el personal del data center solicitan el menú de opciones al sistema por mensaje de texto SMS.
3) El administrador del data center solicita que el sistema empiece a monitorear el aire acondicionado.
4) El administrador del data center solicita que el sistema deje de monitorear el aire acondicionado.
5) El administrador o el personal del data center solicitan la temperatura y humedad de la sala por mensaje de texto SMS.
6) El administrador o el personal del data center solicitan el estado de los aires acondicionados por mensaje de texto SMS.
7) El administrador o el personal del data center acceden a la aplicación web.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

8) El administrador o el personal del data center ingresan o modifican los números de celular almacenados en el sistema y esté envía un mensaje SMS para confirmar los números.
9) El administrador o el personal del data center modifican los datos de inicio de sesión.
10) El equipo de aire acondicionado monitoreado en el data center deja de funcionar y el sistema envía un alerta SMS al personal del data center.
11) El administrador del data center envía la orden vía SMS de encender el aire acondicionado de repuesto.
12) El Sistema mide la temperatura y humedad del data center y la almacena en su base de datos.
13) El Sistema verifica el estado actual del funcionamiento del aire acondicionado monitoreado.
14) La temperatura en el data center sobrepasa el valor permitido y se envía una alerta SMS al personal del data center.
15) El administrador reinicia el sistema ante cualquier fallo del mismo enviando una orden especifica vía mensaje de texto SMS.
16) El administrador o el personal del data center solicitan la dirección de IP del dispositivo por mensaje de texto SMS.

❖ Modelo de Comportamiento

El diagrama de comportamiento describe el comportamiento final que el sistema debe cumplir para interactuar adecuadamente con el ambiente. Para esto, se realizan diagramas de flujo de datos y diccionario de datos.

❖ Diagramas de Flujo de datos

A continuación se utilizarán diagramas de flujo de datos (DFD) para explicar las respuestas del sistema ante los eventos listados anteriormente.

Evento 1: Al iniciar el sistema, este envía un mensaje de texto al administrador y al personal del data center con las primeras opciones y la IP del dispositivo.

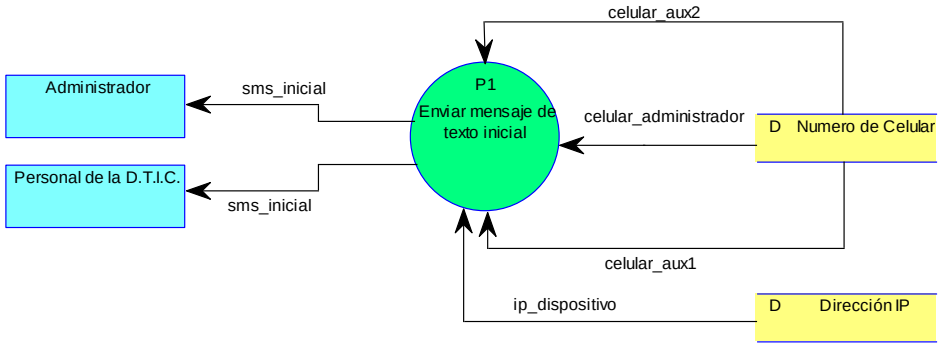


Figura 4-15 - DFD Evento 1

Evento 2: El administrador o el personal del data center solicita el menú de opciones al sistema por mensaje de texto SMS.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

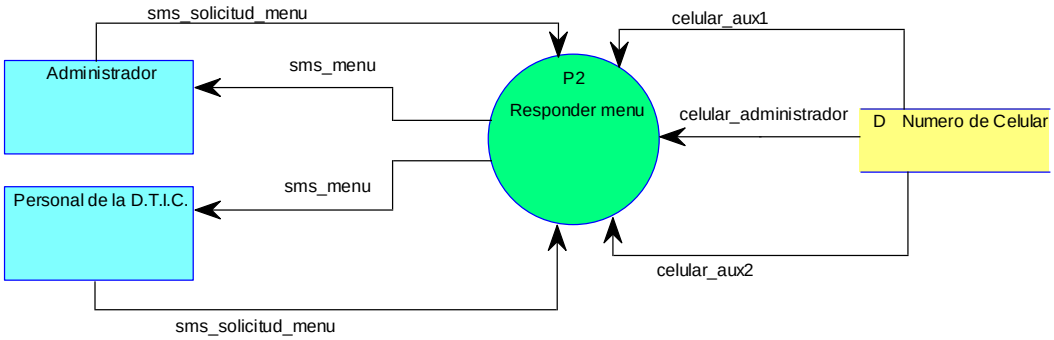


Figura 4-16 - DFD Evento 2

Evento 3: El administrador del data center solicita que el sistema empiece a monitorear el aire acondicionado.

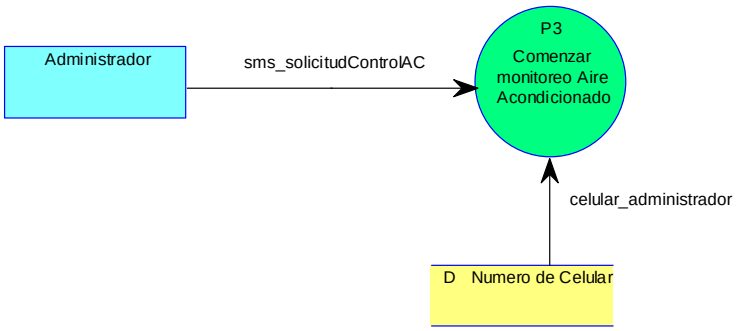


Figura 4-17 - DFD Evento 3

Evento 4: El administrador del data center solicita que el sistema deje de monitorear el aire acondicionado.

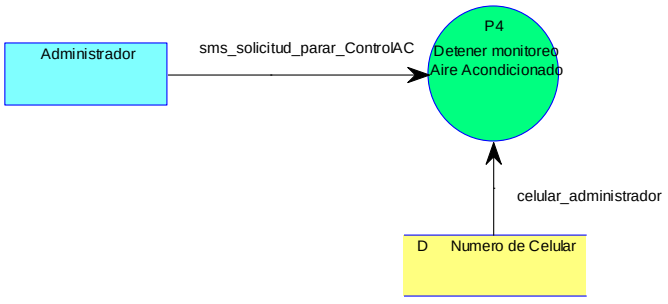


Figura 4-18 - DFD Evento 4

Evento 5: El administrador o el personal del data center solicitan la temperatura y humedad de la sala por mensaje de texto SMS.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

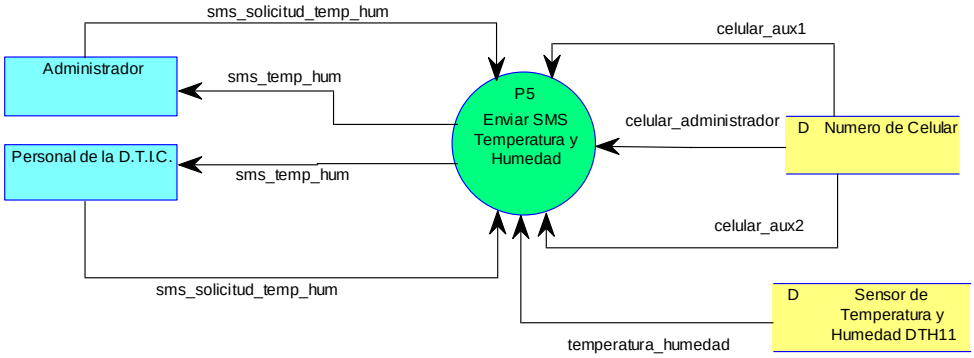


Figura 4-19 - DFD Evento 5

Evento 6: El administrador o el personal del data center solicitan el estado de los aires acondicionados por mensaje de texto SMS

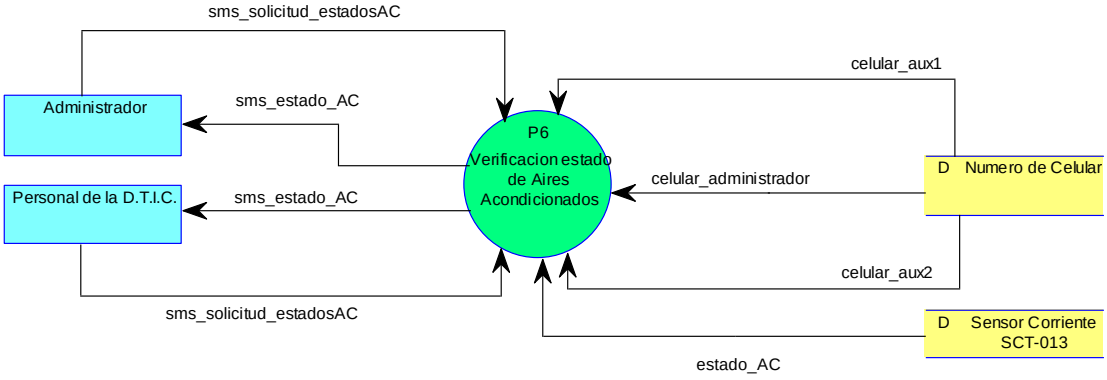


Figura 4-20 - DFD Evento 6

Evento 7: El administrador o el personal del data center acceden a la aplicación web.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

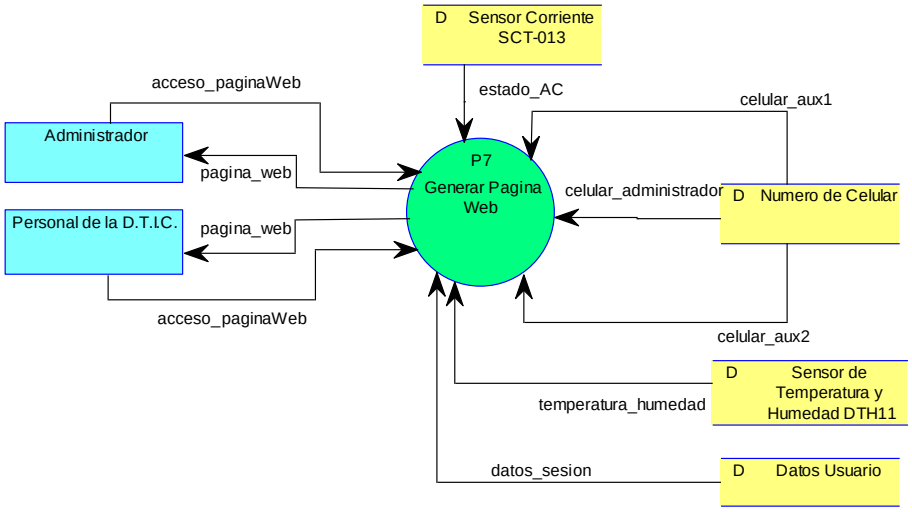


Figura 4-21 - DFD Evento 7

Evento 8: El administrador o el personal del data center ingresan o modifican los números de celular almacenados en el sistema y esté envía un mensaje SMS para confirmar los números.

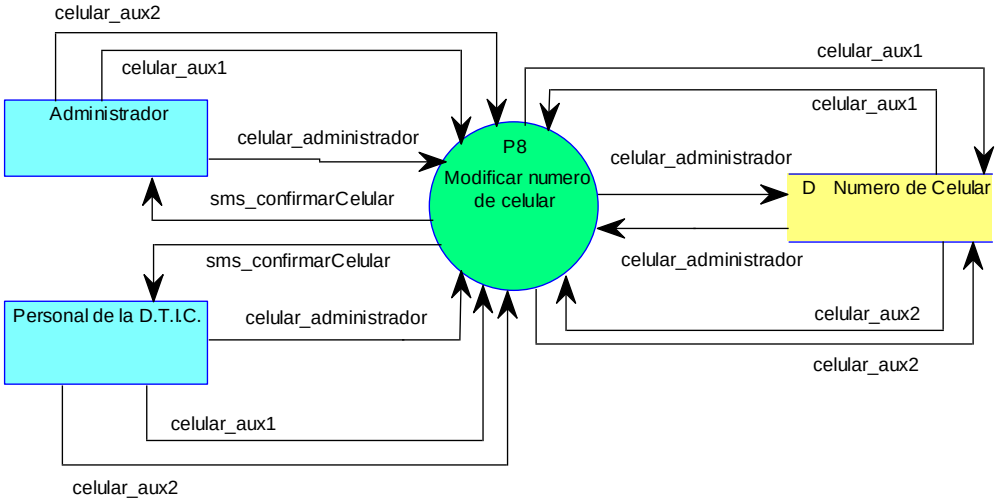


Figura 4-22 - DFD Evento 8

Evento 9: El administrador o el personal del data center modifican los datos de inicio de sesión.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

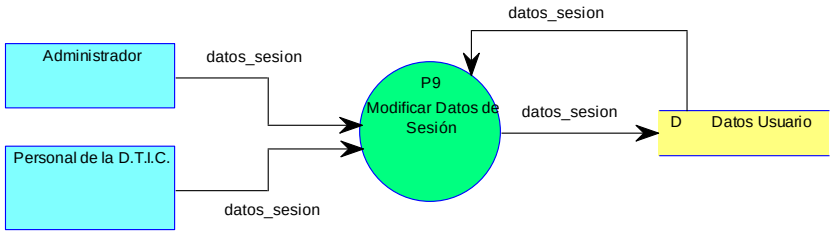


Figura 4-23 - DFD Evento 9

Evento 10: El equipo de aire acondicionado monitoreado en el data center deja de funcionar y el sistema envía un alerta SMS al personal del data center.

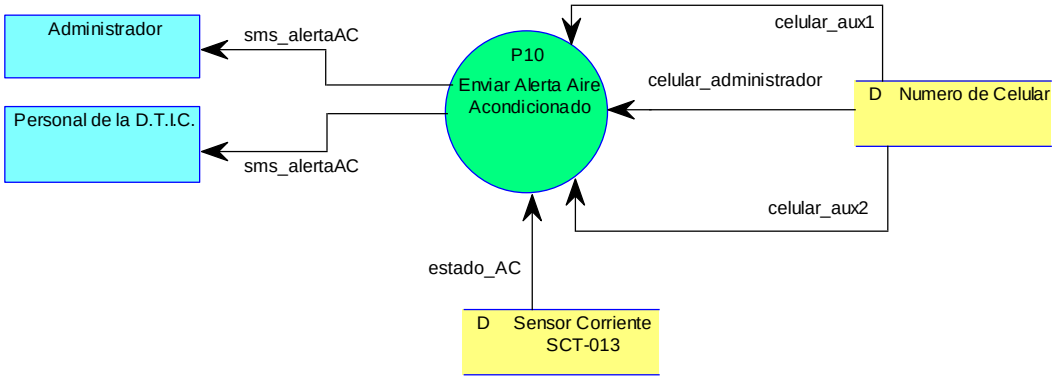


Figura 4-24 - DFD Evento 10

Evento 11: El administrador del data center envía la orden vía SMS de encender el aire acondicionado de repuesto.

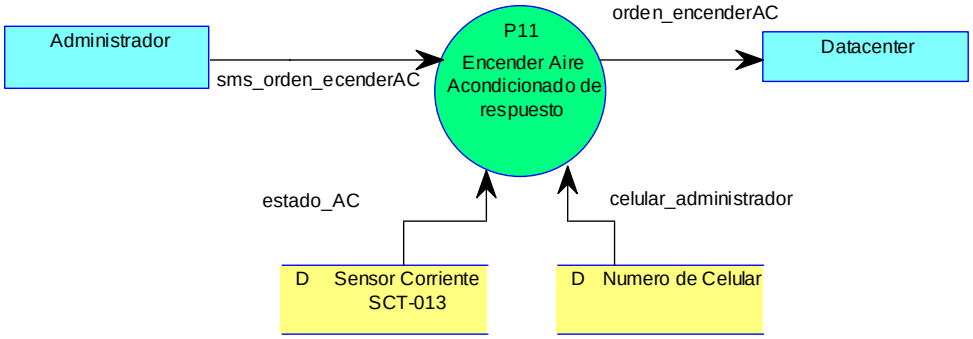


Figura 4-25 - DFD Evento 11

Evento 12: El Sistema mide la temperatura y humedad del data center y la almacena en su base de datos.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

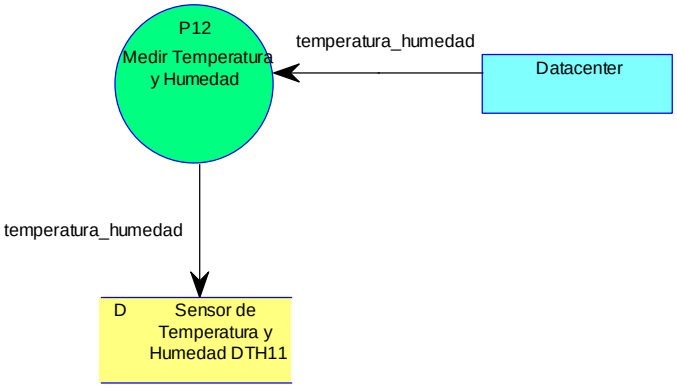


Figura 4-26 - DFD Evento 12

Evento 13: El Sistema verifica el estado actual del funcionamiento del aire acondicionado.

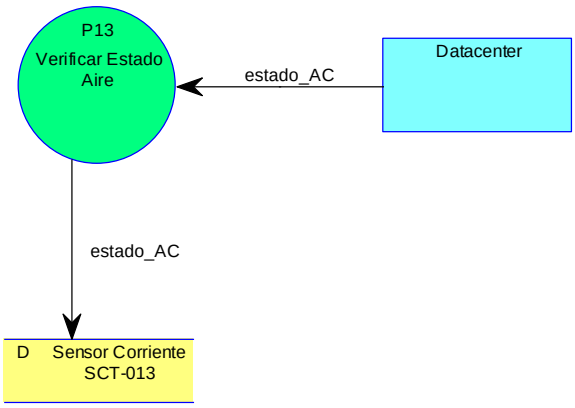


Figura 4-27 - DFD Evento 13

Evento 14: La temperatura en el data center sobrepasa el valor permitido y se envía una alerta SMS al personal del data center.

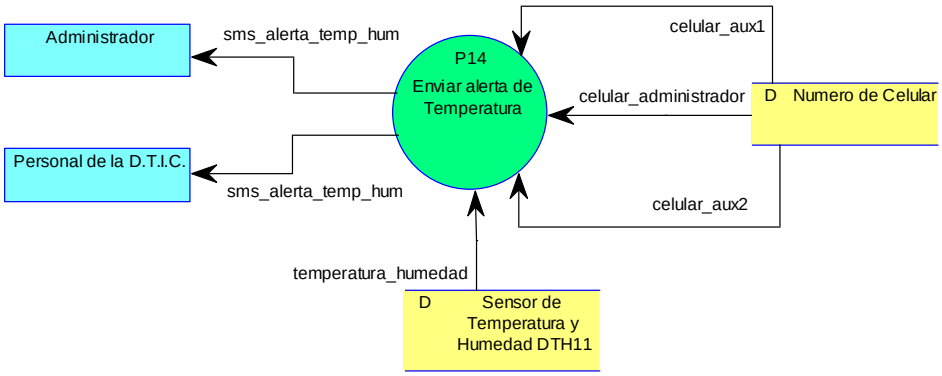


Figura 4-28 - DFD Evento 14

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Evento 15: El administrador reinicia el sistema vía mensaje de texto SMS.

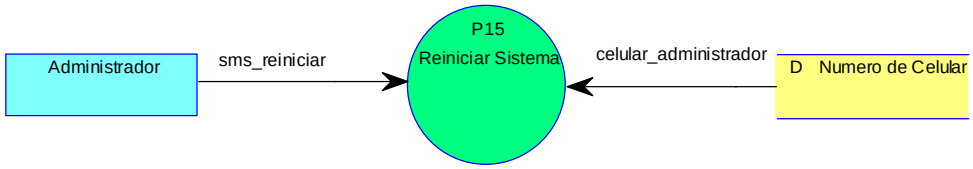


Figura 4-29 - DFD Evento 15

Evento 16: El administrador o el personal del data center solicitan la dirección de IP del dispositivo por mensaje de texto SMS.

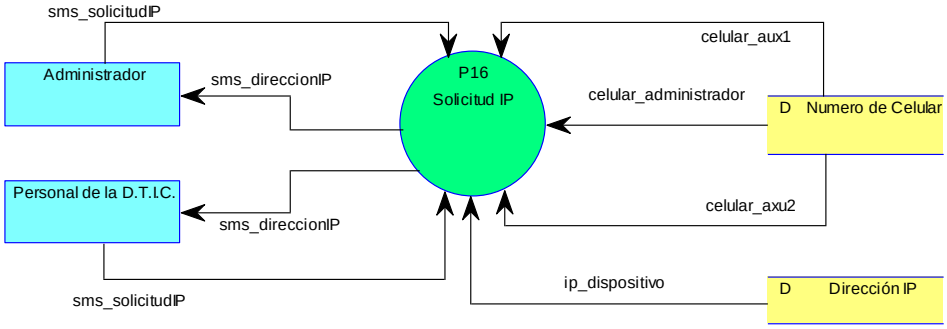


Figura 4-30 - DFD Evento 16

❖ Especificaciones de Procesos

La especificación de procesos, es una herramienta del modelado de sistemas, que nos permite definir qué sucede en los procesos o funciones de un sistema. El objetivo es definir qué debe hacerse para transformar entradas en salidas.

La forma más utilizada para realizar las especificaciones de procesos es el lenguaje estructurado. A continuación, se utilizará el lenguaje estructurado, para describir a cada uno de los procesos involucrados de forma tal que puedan ser comunicados de forma efectiva para todo aquel que desee entender el funcionamiento de los mismos.

Proceso 1: Enviar Mensaje de Texto inicial

Comenzar

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Leer **ip_dispositivo**

Enviar **sms_inicial**

Fin

Proceso 2: Responder Menú

Comenzar

Leer **sms_solicitud_menu**

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Si **celular** es igual **celular_administrador** o **celular_aux1** o **celular_aux2**

Si **contenido** es igual a “hi”

Enviar **sms_menu**

Fin Si

Fin Si

Finalizar

Proceso 3: Comenzar Monitoreo Aire Acondicionado

Comenzar

Leer **sms_solicitudControlAC**

Leer **celular_administrador**

Si **celular** es igual **celular_administrador**

Si **contenido** es igual a “ok”

Iniciar Monitoreo de Aire Acondicionado

Fin Si

Fin Si

Fin

Proceso 4: Detener Monitoreo Aire Acondicionado

Comenzar

Leer **sms_solicitud_parar_ControlAC**

Leer **celular_administrador**

Si **celular** es igual **celular_administrador**

Si **contenido** es igual a “de”

Detener Monitoreo de Aire Acondicionado

Fin Si

Fin Si

Fin

Proceso 5: Enviar SMS Temperatura y Humedad

Comenzar

Leer **sms_solicitud_temp_hum**

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Si **celular** es igual **celular_administrador** o **celular_aux1** o **celular_aux2**

Si **contenido** es igual “o1”

Leer **temperatura_humedad**

Enviar **sms_temp_hum**

Fin Si

Fin si

Finalizar

Proceso 6: Verificación estado de Aires Acondicionado

Comenzar

Leer **sms_solicitud_estadosAC**

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Si **celular** es igual a **celular_administrador** o **celular_aux1** o **celular_aux2**

Si **contenido** es igual “o2”

Si Monitoreo de Aire Acondicionado Iniciado

Leer **estado_AC**

Enviar **sms_estado_AC**

Fin Si

Si no

estado es igual “No iniciado”

Enviar **sms_estado_AC**

Fin Si no

Fin Si

Finalizar

Proceso 7: Generar Pagina Web

Comenzar

Leer **acceso_paginaweb**

Leer **datos_sesion**

Si **usuario** es igual a **usuario_registrado** y **contraseña** es igual a **contraseña_registrada**

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Leer **estado_AC**

Leer **temperatura_humedad**

Visualizar **pagina_web**

Fin Si

Finalizar

Proceso 8: Modificar número de celular

Comenzar

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Visualizar **celular_administrador**

Visualizar **celular_aux1**

Visualizar **celular_aux2**

Actualizar Almacén **Numero de Celular**

Enviar **sms_confirmarCelular**

Fin

Proceso 9: Modificar datos de sesión

Comenzar

Leer **datos_sesion**

Actualizar Almacén **Datos de Usuario**

Fin

Proceso 10: Enviar Alerta Aire Acondicionado

Comenzar

Leer **estado_AC**

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Si **estado** es igual “Apagado”

Enviar **sms_alertaAC**

Fin Si

Finalizar

Proceso 11: Encender Aire Acondicionado de repuesto

Comenzar

Leer **sms_orden_encenderAC**

Leer **celular_administrador**

Si **celular** es igual **celular_administrador**

Si **contenido** es igual a “si”

Leer **estado_AC**

Si estado es igual “Apagado”

Enviar **orden_encenderAC**

Fin Si

Fin Si

Fin Si

Finalizar

Proceso 12: Medir Temperatura y Humedad

Comenzar

Leer **temperatura_humedad**

Actualizar Almacén **Sensor de Temperatura y Humedad DHT11**

Finalizar

Proceso 13: Verificar estado Aire

Comenzar

Leer **estado_AC**

Actualizar Almacén **Sensor corriente SCT-013**

Finalizar

Proceso 14: Enviar Alerta de Temperatura

Comenzar

Leer **temperatura_humedad**

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Si **temperatura_humedad** mayor o igual a **temperatura_permitida**

Enviar **sms_alerta_temp_hum**

Fin Si

Finalizar

Proceso 15: Reiniciar sistema

Comenzar

Leer **sms_reiniciar**

Leer **celular_administrador**

Si **celular** es igual **celular_administrador**

Si **contenido** es igual a “re”

Reiniciar Sistema

Fin Si

Fin Si

Finalizar

Proceso 16: Solicitud IP

Comenzar

Leer **sms_solicitudIP**

Leer **celular_administrador**

Leer **celular_aux1**

Leer **celular_aux2**

Si **celular** es igual **celular_administrador** o **celular_aux1** o **celular_aux2**

Si **contenido** es igual “ip”

Leer **ip_dispositivo**

Enviar **sms_direccionIP**

Fin Si

Fin si

Fin

❖ Diccionario de Datos

El diccionario de datos contiene un listado organizado de todos los datos relacionados al sistema con una descripción precisa que permita tanto al analista como al usuario entender las entradas, salidas y otros componentes que forman parte del sistema.

Entidades Externas

Entidad Externa	Descripción
Administrador	Es la persona a cargo del data center, también la que proveerá el número de teléfono celular al cual el sistema responderá
Datacenter	Es el lugar donde estará alojado el sistema embebido de telemetría y el que debe monitorear, los sensores del sistema obtendrán datos de este lugar.
Personal de la D.T.I.C.	Relacionado a las personas que trabajan en el D.T.I.C. además del administrador, los auxiliares proveerán números de celular para interactuar con el sistema..

Tabla 4-2 - Entidades Externas

Flujos de Datos

Flujo de dato	Especificación
acceso_paginaweb	Solicitud de acceso a la aplicación web del sistema embebido, contiene la dirección de ip para acceder al sistema y los datos de sesión. **[ip_dispositivo+datos_sesion]**
alerta_AC	Tipo String. Mensaje generado por el sistema, es el alerta indicando que el aire acondicionado dejo de funcionar.
alerta_temp_hum	Tipo String. Mensaje generado por el sistema, es el alerta indicando que la temperatura alcanzó valores no deseados.
amperios	Tipo doublé. Unidad de intensidad de corriente eléctrica, valor obtenido por el sensor de corriente SCT-013.
celular	Tipo String. Número de celular contenido en el mensaje de texto entrante al sistema. **[+54XXXXXXXXXX](X es un número entre 0-9)**
celular_administrador	Tipo String. Número de celular propiedad del administrador del data center, se encuentra almacenado en el sistema. **[+54XXXXXXXXXX](X es un número entre 0-9)**
celular_aux1	Tipo String. Número de celular perteneciente al personal asignado como auxiliar 1 para manipular el sistema, el número, se encuentra almacenado en el sistema. **[+54XXXXXXXXXX](X es un número entre 0-9)**
celular_aux2	Tipo String. Número de celular perteneciente al personal asignado como auxiliar 2 para manipular el sistema, el número, se encuentra almacenado en el sistema.

	[+54XXXXXXXXXX](X es un número entre 0-9)
confirmacion	Tipo string, contenido del mensaje de confirmación, se utiliza para confirmar que el nuevo número de celular es correcto.
contenido	Tipo String. Contenido del mensaje de texto enviando por el administrador al sistema, el contenido debe ser cualquiera de estas opciones para obtener respuesta. **[“hi”|”o1”|”o2”|”re”|”si”|”de”|”ok”|”ip”]**
contraseña	Tipo string y password, contraseña enviada por el cliente para iniciar sesión, se encuentra cifrada.
contraseña_registrada	Tipo string y password, contraseña correspondiente a un usuario almacenado en el sistema, se encuentra cifrada.
datos_sesion	Contiene los datos enviados por el cliente de la aplicación web para iniciar sesión. **[usuario+contraseña]**
estado	Tipo String. Variable obtenida después que el sistema proceso los datos del sensor SCT-013 y determino si el aire acondicionado está encendió o no. **[“Encendido”|”Apagado”|”No Inicio monitoreo”]**
estado_AC	Variable utilizada para determinar si el aire acondicionado esta encendido o no: **[voltaje+amperios+estado]**
humedad	Tipo Float. Unidad en porcentaje, valor obtenido del data center por el sensor de temperatura y humedad DHT11
ip_dispositivo	Tipo String, dirección IP del dispositivo en la red de área local. **[X.X.X.X](X es un numero entre 0-255)**
menu	Tipo String, contenido de texto en el mensaje SMS que contiene el menú de opciones del sistema cuando es solicitado.
menu_inicial	Tipo String, contenido de texto en el mensaje SMS que envía el sistema al iniciar.
orden_encenderAC	Comando enviando al pin del relé donde se encuentra conectado el aire acondicionado de repuesto para encenderlo.
pagina_web	Archivo index.php que contiene la aplicación web del sistema.
sms_alerta_temp_hum	Tipo string. Mensaje de texto generado por el sistema para el administrador y el personal auxiliar, en su contenido se encuentra un mensaje de alerta indicando que la temperatura sobrepaso el valor permitido. **[(celular_administrador & celular_aux1 & celular_aux2) +alerta_temp_hum]**
sms_alertaAC	Tipo String. Mensaje de texto generado por el sistema para el administrador, en su contenido se encuentra un mensaje de alerta indicando que el aire acondicionado dejo de funcionar. **[(celular_administrador & celular_aux1 & celular_aux2) +alerta_AC]**
sms_confirmarCelular	Tipo String, contiene el mensaje de texto con la confirmación, cuando se modifica el número de celular registrado.

	[(celular_administrador & celular_aux1 & celular_aux2)+confirmacion]
sms_estado_AC	Tipo string. Mensaje de texto generado por el sistema para el administrador o el personal auxiliar, en su contenido se encuentra el estado del aire acondicionado. **[(celular_administrador|celular_aux1|celular_aux2)+estado]**
sms_inicial	Tipo String, Mensaje de texto generando por el sistema al iniciarse, en su contenido se encuentra el menú de opciones inicial **[(celular_administrador & celular_aux1 & celular_aux2)+menu_inicial+ip_dispositivo]**
sms_menu	Tipo string. Mensaje de texto generado por el sistema para el administrador, en su contenido se encuentra un menú indicando los mensajes y opciones que soporta el sistema. **[[(celular_administrador|celular_aux1|celular_aux2)+menu]**
sms_orden_encenderAC	Tipo String. Mensaje de texto enviado por el administrador, en su contenido se encuentra la orden de encender el aire acondicionado de repuesto. **[celular+contenido]**
sms_reiniciar	Tipo String. Mensaje de texto enviado por el administrador, en su contenido se encuentra la orden de reiniciar el sistema. **[celular+contenido]**
sms_solicitud_estadosAC	Tipo String. Mensaje de texto enviado por el administrador, en su contenido se encuentra la solicitud el estado del aire acondicionado. **[celular+contenido]**
sms_solicitud_menu	Tipo String. Mensaje de texto enviado por el administrador, en su contenido se encuentra la orden que solicita el menú del sistema. **[celular+contenido]**
sms_solicitud_parar_ControlAC	Tipo string. Mensaje de texto enviando por el administrador, en su contenido se encuentra la solitud de dejar de monitorear el aire acondicionado **[celular+contenido]**
sms_solicitud_temp_hum	Tipo String. Mensaje de texto enviado por el administrador, en su contenido se encuentra la solicitud de la temperatura y humedad de la sala **[celular+contenido]**
sms_solicitudControlAC	Tipo string. Mensaje de texto enviando por el administrador, en su contenido se encuentra la solitud de comenzar a monitorear el aire acondicionado **[celular+contenido]**
sms_solicitudIP	Tipo String. Mensaje de texto enviado por el administrador o los auxiliares, en su contenido se encuentra la solicitud de la dirección ip del sistema. **[celular+contenido]**
sms_temp_hum	Tipo String. Mensaje de texto generado por el sistema, en su contenido se encuentra la temperatura y humedad del data center.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

	[celular_administrador+temperatura_humedad]
temperatura	Tipo Float. Unidad en grados centígrados, valor obtenido del data center por el sensor de temperatura y humedad DHT11
temperatura_humedad	Tipo string. Variable compuesta por la temperatura y humedad. **[temperatura+humedad]**
temperatura_permitida	Tipo Float. Temperatura establecida como límite máximo tolerable en el data center.
usuario	Tipo string, usuario enviado por el cliente para iniciar sesión.
usuario_registrado	Tipo string, usuario almacenado en la base de datos del sistema.
voltaje	Tipo Double. Unidad de tensión eléctrica, valor obtenido del aire acondicionado por el sensor de corriente SCT-013

Tabla 4-3 - Flujo de Datos

Almacenes de datos

Almacén	Descripción
Número de Celular	Número de celular del administrador almacenado en el sistema
Sensor de corriente SCT-013	Datos obtenidos por el sensor de corriente SCT-013, incluidos voltaje y amperaje actuales del aire acondicionado.
Sensor de Temperatura y Humedad DHT11	Datos obtenidos por el sensor de temperatura y humedad DHT11, contiene la temperatura y humedad actual del data center.
Dirección IP	Dirección IP del dispositivo asignada por el router de la red de área local.
Datos usuario	Datos del usuario para iniciar sesión en la aplicación web.

Tabla 4-4 - Almacenes de datos

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

4.3.5 Diseño Estructural

El diagrama de estructuras muestra el modelo top-down de las estructuras de control del firmware mediante un árbol de invocación. El diagrama de estructuras permite modelar el firmware de forma jerárquica. Cada nivel representa una descomposición más detallada del módulo en el nivel superior (Figura 4-31).

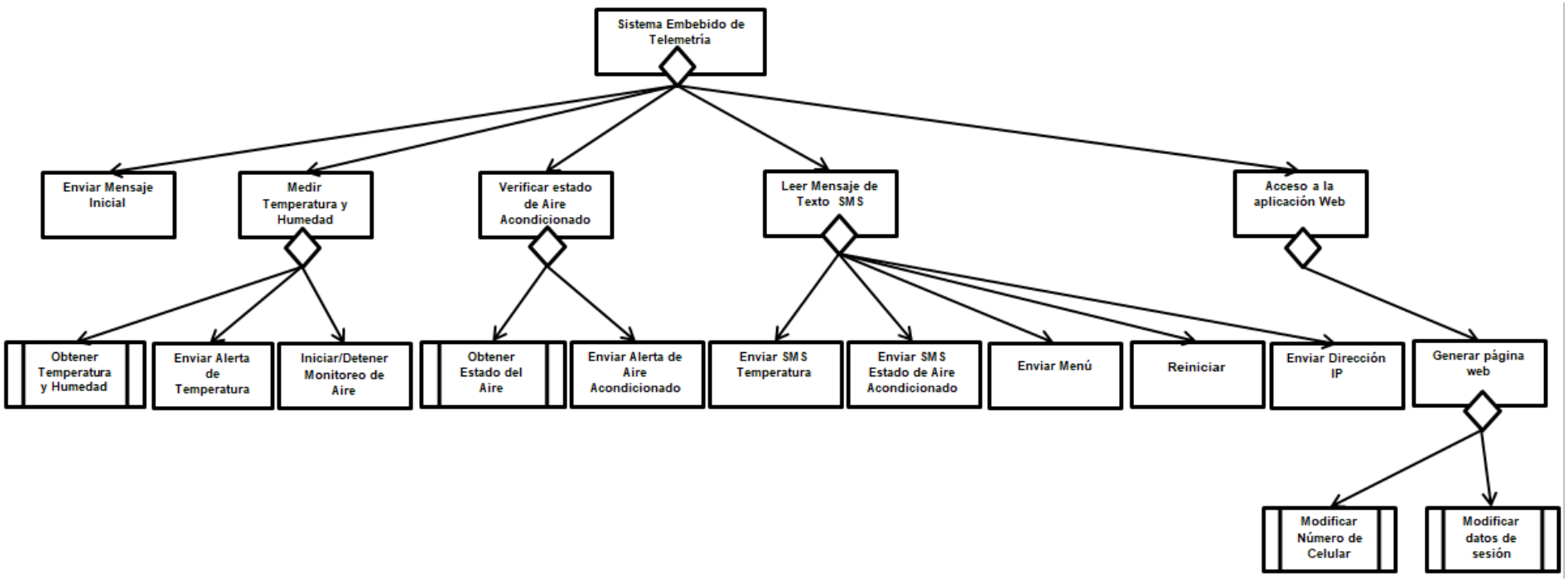


Figura 4-31 - Diagrama de Estructuras del Sistema Embebido de Telemetría

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

4.3.6 Modelo de Negocio

El modelo del negocio (Figura 4-32), especifica los procesos de negocio que soportará el sistema, este modelo ayuda a la comprensión del funcionamiento del sistema. Los procesos de negocio son expresados en términos de casos de uso y actores del negocio; los casos de uso se corresponden con los procesos del negocio y los actores con los usuarios.

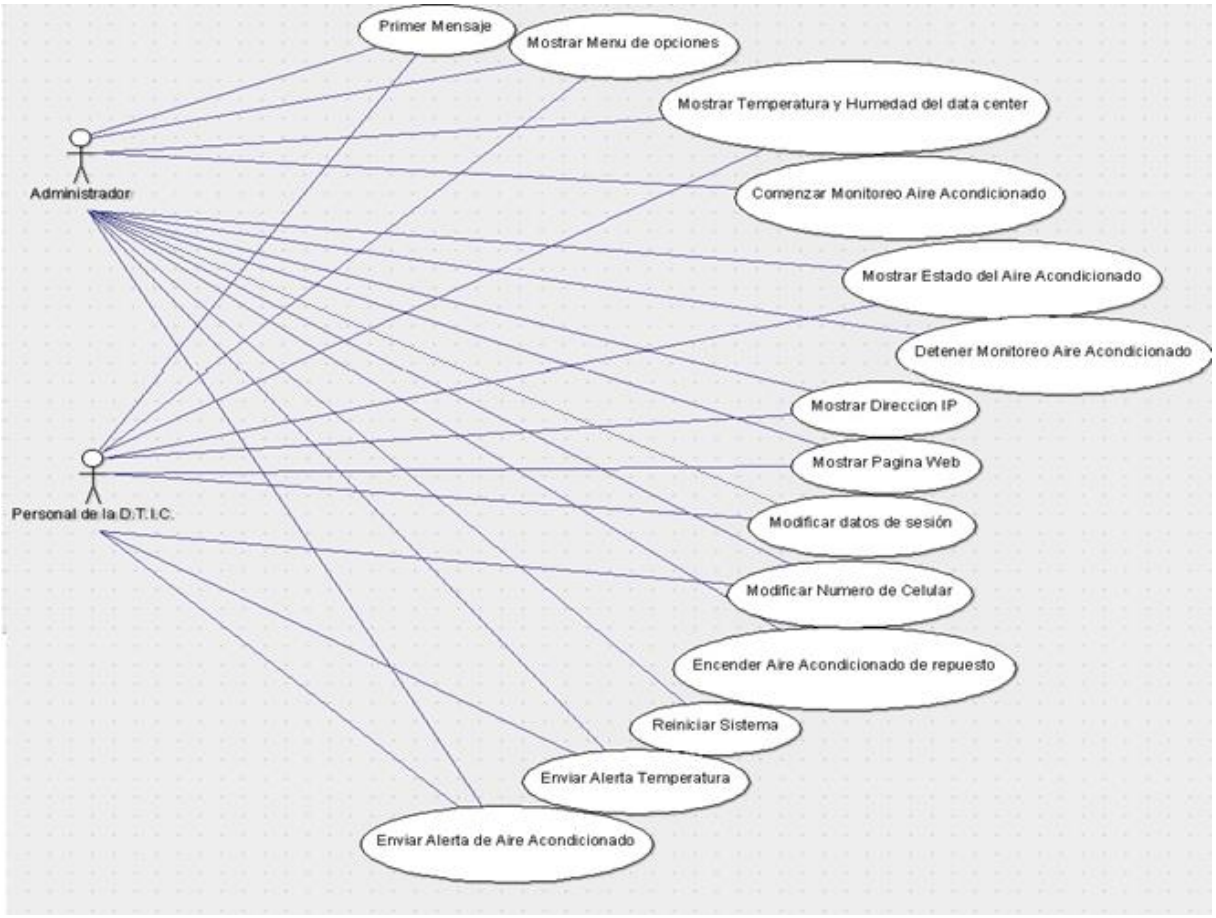


Figura 4-32 - Diagramas de Caso de Uso del Sistema Embebido de Telemetría

4.3.7 Especificación de Requisitos funcionales

A partir del modelo del negocio se derivan los casos de uso, cada caso de uso del negocio implica un diagrama de casos de uso.

Caso de Uso: Primer Mensaje

Iniciador	Sistema
Precondición	El número de celular del administrador y/o los auxiliares deben estar registrados como válidos en el sistema
Camino básico	
Actor	**Sistema**
	1. El Sistema inicia sus funciones
	2. El Sistema lee la dirección de IP que se le asigno en la red de área local.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

	3. El Sistema envía el mensaje de texto al administrador y los auxiliares con el primer menú de opciones y la dirección de IP.
Camino Alternativo 1	Si en el paso 2, alguno de los números registrados no es válido, el mensaje no se enviara para ese número.
Poscondición	-

Tabla 4-5 - Descripción de Caso de Uso Primer Mensaje

Caso de Uso: Mostrar Menú de opciones

Iniciador	Administrador, Personal de la D.T.I.C.
Precondición	Los números de celular deben estar registrados como válidos en el sistema
Camino básico	
Actor	**Sistema**
1. Solicita el Menú de Opciones por mensaje de texto.	
	2. Comprueba que el número de procedencia este registrado en el sistema.
	3. Revisa si en el contenido del mensaje se encuentra la palabra específica "**hi"**.
	4. Responde enviando un mensaje de texto al número de procedencia con el menú de opciones.
Camino Alternativo 1	Si en el paso 2 el número de procedencia en el mensaje no está registrado, el sistema ignora el mensaje.
Camino Alternativo 2	Si en el paso 3 no se encuentra la palabra, el sistema ignora el mensaje.
Poscondición	-

Tabla 4-6 - Descripción de Caso de Uso Mostrar Menú de Opciones

Caso de Uso: Mostrar Temperatura y Humedad del data center

Iniciador	Administrador, Personal de la D.T.I.C.
Precondición	Los números de celular deben estar registrados como válidos en el sistema
Camino básico	
Actor	**Sistema**
1. Solicita temperatura y humedad por mensaje de texto.	
	2. Comprueba que el número de procedencia este registrado en el sistema.
	3. Revisa si en el contenido del mensaje se encuentra la palabra **o1.**
	4. Realiza la medición de la Temperatura y Humedad del data center.
	5. Responde enviando un mensaje de

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

	texto al número de procedencia con la temperatura y humedad del data center.
Camino Alternativo 1	Si en el paso 2 el número de procedencia en el mensaje no está registrado, el sistema ignora el mensaje.
Camino Alternativo 2	Si en el paso 3 no se encuentra la palabra específica para solicitar la temperatura y humedad, el sistema ignora el mensaje.
Poscondición	-

Tabla 4-7 - Descripción de Caso de Uso Mostrar Temperatura y Humedad del data center

Caso de Uso: Mostrar Estado de Aire Acondicionado

Iniciador	Administrador, Personal de la D.T.I.C.
Precondición	Los números de celular deben estar registrados como válidos en el sistema
Camino básico	
Actor	**Sistema**
1. Solicita estado del Aire Acondicionado por mensaje de texto.	
	2. Comprueba que el número de procedencia este registrado en el sistema.
	3. Revisa si en el contenido del mensaje se encuentra la palabra **o2.**
	4. Utiliza el sensor de corriente y determina el estado del aire acondicionado.
	5. Responde enviando un mensaje de texto al número de procedencia con el estado del aire acondicionado.
Camino Alternativo 1	Si en el paso 2 el número de procedencia en el mensaje no está registrado, el sistema ignora el mensaje.
Camino Alternativo 2	Si en el paso 3 no se encuentra la palabra específica para solicitar el estado del aire acondicionado, el sistema ignora el mensaje.
Poscondición	-

Tabla 4-8 - Descripción de Caso de Uso Mostar Estado de Aire Acondicionado

Caso de uso: Comenzar Monitoreo de Aire Acondicionado

Iniciador	Administrador
Precondición	El número de celular debe estar registrado como válido en el sistema
Camino básico	
Actor	**Sistema**
1. Solicita comenzar a monitorear Aire Acondicionado por mensaje de texto.	
	2. Comprueba que el número de procedencia este registrado en el sistema y pertenezca al administrador.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

	3. Revisa si en el contenido del mensaje se encuentra la palabra **ok.**
	4. Comienza los procesos para monitorear el Aire Acondicionado.
Camino Alternativo 1	Si en el paso 2 el número de procedencia en el mensaje no está registrado, el sistema ignora el mensaje.
Camino Alternativo 2	Si en el paso 3 no se encuentra la palabra específica para solicitar el monitoreo de Aire acondicionado, el sistema ignora el mensaje.
Poscondición	El Sistema está monitoreando el Aire Acondicionado

Tabla 4-9 - Descripción de Caso de Uso Comenzar Monitoreo de Aire Acondicionado

Caso de uso: Detener Monitoreo de Aire Acondicionado

Iniciador	Administrador
Precondición	El número de celular debe estar registrado como válido en el sistema
Camino básico	
Actor	**Sistema**
1. Solicita detener el monitoreo del Aire Acondicionado por mensaje de texto.	
	2. Comprueba que el número de procedencia este registrado en el sistema y pertenezca al administrador.
	3. Revisa si en el contenido del mensaje se encuentra la palabra **de.**
	4. Detiene la supervisión del funcionamiento del Aire Acondicionado.
Camino Alternativo 2	Si en el paso 2 el número de procedencia en el mensaje no está registrado, el sistema ignora el mensaje.
Camino Alternativo 2	Si en el paso 3 no se encuentra la palabra específica para solicitar detener el monitoreo de Aire acondicionado, el sistema ignora el mensaje.
Poscondición	El Sistema no estará monitoreando el Aire Acondicionado

Tabla 4-10 - Descripción de Caso de Uso Detener Monitoreo de Aire Acondicionado

Caso de Uso: Mostrar Dirección IP

Iniciador	Administrador, Personal de la D.T.I.C.
Precondición	Los números de celular deben estar registrados como válidos en el sistema
Camino básico	
Actor	**Sistema**
1. Solicita la dirección IP del dispositivo.	
	2. Comprueba que el número de procedencia este registrado en el sistema.
	3. Revisa si en el contenido del mensaje se encuentra la palabra **ip.**
	4. Lee la dirección de IP que le asigno

	se le asigno en la red de área local.
	5. Responde enviando un mensaje de texto al número de procedencia con la dirección de IP.
Camino Alternativo 1	Si en el paso 2 el número de procedencia en el mensaje no está registrado, el sistema ignora el mensaje.
Camino Alternativo 2	Si en el paso 3 no se encuentra la palabra específica para solicitar el estado del aire acondicionado, el sistema ignora el mensaje.
Poscondición	-

Tabla 4-11 - Descripción de Caso de Uso Mostrar Dirección IP

Caso de Uso: Mostrar Pagina Web

Iniciador	Administrador, Personal D.T.I.C.
Precondición	El sistema debe tener asignada una dirección IP en la red LAN
Camino básico	
Actor	**Sistema**
1. Accede como cliente a la aplicación web.	
	2. Abre la conexión con el cliente**.**
	3. Envía los archivos de la aplicación web al navegador web al cliente.
	4. Solicita al cliente que inicie sesión
5. Introduce usuario y contraseña	
	6. Comprueba si el usuario y contraseña son correctos.
	7. Lee los datos recolectados hasta ese momento y los muestra en el grafico histórico
	8. Lee la temperatura y humedad de la sala y muestra estos datos en el gráfico de calibre.
	9. Lee el consumo del aire en ese momento y muestra los valores en pantalla
	10. Lee el estado de los aires y los muestra con indicadores de encendido (ON) y apagado (OFF), de acuerdo al estado de cada uno.
11. Introduce la fecha de la cual desea ver los datos en el grafico histórico	
	12. Lee los datos recolectados en la fecha ingresada y los muestra en el grafico histórico.
Camino Alternativo 1	Si en el paso 1 no se puede abrir la conexión como cliente, se informa de dicha situación
Camino Alternativo 2	Si en el paso 6 el usuario y contraseña no están registrados se informa de esta situación y no se permite el acceso.
Camino Alternativo 3	Si en el paso 7 no se pueden leer los datos, se informa de esta situación
Camino Alternativo 4	Si en el paso 8 no se pueden leer los datos, se informa de

	esta situación
Camino Alternativo 5	Si en el paso 9 no se pueden leer los datos, se informa de esta situación
Camino Alternativo 6	Si en el paso 10 no se pueden leer los datos, se informa de esta situación
Poscondición	-

Tabla 4-12 - Descripción de Caso de Uso Mostrar Aplicación web

Caso de Uso: Modificar datos de sesión

Iniciador	Administrador, Personal D.T.I.C.
Precondición	El sistema debe tener asignada una dirección IP en la red LAN
Camino básico	
Actor	**Sistema**
1. Accede como cliente a la aplicación web.	
	2. Abre la conexión con el cliente**.**
	3. Envía los archivos de la aplicación web al navegador web al cliente.
	4. Solicita al cliente que inicie sesión
5. Introduce usuario y contraseña	
	6. Comprueba si el usuario y contraseña son correctos.
7. Modifica el usuario o la contraseña para iniciar sesión.	
	8. Actualiza los datos de inicio de sesión en la base de datos.
Camino Alternativo 1	Si en el paso 1 no se puede abrir la conexión como cliente, se informa de dicha situación
Camino Alternativo 2	Si en el paso 6 el usuario y contraseña no están registrados se informa de esta situación y no se permite el acceso, se repite el paso 4.
Poscondición	Se Actualizaron los datos de inicio de sesión de la aplicación web del sistema.

Tabla 4-13 - Descripción de Caso de Uso Modificar datos de sesión

Caso de Uso: Modificar números de celular

Iniciador	Administrador, Personal D.T.I.C.
Precondición	El sistema debe tener asignada una dirección IP en la red LAN
Camino básico	
Actor	**Sistema**
1. Accede como cliente a la aplicación web.	
	2. Abre la conexión con el cliente**.**
	3. Envía los archivos de la aplicación web al navegador web al cliente.
	4. Solicita al cliente que inicie sesión
5. Introduce usuario y contraseña	
	6. Comprueba si el usuario y contraseña son correctos.
	7. Lee los números de celular

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

	almacenados y los muestra en pantalla
8. Introduce un número de celular con su categoría (Administrador, Auxiliar 1, Auxiliar 2).	
	9. Actualiza el número de celular almacenado por el nuevo número ingresado en su correspondiente categoría.
	10. Envía un mensaje de texto al nuevo número registrado.
Camino Alternativo 1	Si en el paso 1 no se puede abrir la conexión como cliente, se informa de dicha situación
Camino Alternativo 2	Si en el paso 6 el usuario y contraseña no están registrados se informa de esta situación y no se permite el acceso, se repite el paso 4.
Camino Alternativo 3	Si en el paso 7 no se pueden leer los datos, se informa de esta situación
Poscondición	Se Actualizo el número de celular almacenado en el sistema

Tabla 4-14 Descripción de Caso de Uso Modificar Número de Celular

Caso de Uso: Encender Aire de repuesto

Iniciador	Administrador
Precondición	1. El número de celular debe estar registrado como válido en el sistema 2. El sistema ya envió el alerta del aire acondicionado
Camino básico	
Actor	**Sistema**
1. Ordena encender el aire de repuesto por mensaje de texto.	
	2. Comprueba que el número de procedencia este registrado en el sistema y pertenezca al administrador.
	3. Revisa si en el contenido del mensaje se encuentra la palabra **si.**
	4. Envía el comando hacía los relés para cambiar su estado y encender el aire de repuesto.
Camino Alternativo 1	Si en el paso 2 el número no se encuentra registrado o no pertenece al administrador, el sistema ignora el mensaje.
Camino Alternativo 2	Si en el paso 3 no se encuentra la palabra específica para solicitar el encendido del aire de repuesto, el sistema ignora el mensaje.
Poscondición	El aire de repuesto, esta encendido

Tabla 4-15 - Descripción de Caso de Uso Encender Aire de Repuesto

Caso de Uso: Reiniciar Sistema

Iniciador	Administrador
Precondición	El número de celular debe estar registrado como válido en el

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

	sistema
Camino básico	
Actor	**Sistema**
1. Solicita el reinicio del sistema por mensaje de texto.	
	2. Comprueba que el número de procedencia este registrado en el sistema y pertenezca al administrador.
	3. Revisa si en el contenido del mensaje se encuentra la palabra **re.**
	4. Envía un mensaje de texto al administrador informando que se va a ejecutar el reinicio del sistema.
	5. Ejecuta el comando de reinicio.
Camino Alternativo 1	Si en el paso 2 el número no se encuentra registrado o no pertenece al administrador, el sistema ignora el mensaje.
Camino Alternativo 2	Si en el paso 3 no se encuentra la palabra específica para solicitar el reinicio del sistema, este ignora el mensaje.
Poscondición	-

Tabla 4-16 - Descripción de Caso de Uso Reiniciar Sistema

Caso de Uso: Enviar Alerta de Temperatura

Iniciador	Sistema
Precondición	El número de celular debe estar registrado como válido en el sistema
Camino básico	
Actor	**Sistema**
	1. Realiza la medición de temperatura y humedad del data center.
	2. Almacena la temperatura y humedad en la base de datos
	3. Compara si la temperatura actual es mayor que la temperatura permitida.
	4. Envía un mensaje de texto al administrador y los auxiliares con una alerta sobre el exceso de temperatura.
5. Recibe el mensaje de texto con el alerta sobre la temperatura	
Camino Alternativo 1	Si en el paso 3 la temperatura actual no es superior a la temperatura actual, el camino termina allí.
Poscondición	-

Tabla 4-17 - Descripción de Caso de Uso Enviar Alerta de Temperatura

Caso de Uso: Enviar Alerta de Aire Acondicionado

Iniciador	Sistema
Precondición	El número de celular debe estar registrado como válido en el sistema
Camino básico	

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Actor	Sistema
	1. Realiza comprobación del aire acondicionado.
	2. Almacena los datos de consumo del aire en la microSD.
	3. Utiliza el sensor de corriente y determina si aire acondicionado esta funcionado.
	4. Envía un mensaje de texto al administrador y los auxiliares con una alerta sobre el exceso el aire acondicionado
5. Recibe el mensaje de texto con el alerta sobre el aire acondicionado	
Camino Alternativo 1	Si en el paso 3 el aire acondicionado está funcionando, el camino termina allí.
Poscondición	-

Tabla 4-18 - Descripción de Caso de Uso Enviar Alerta de Aire Acondicionado

4.4 Aplicación Web

A Continuación se explicará la aplicación web y los archivos que la componen en el sistema embebido de Telemetría.

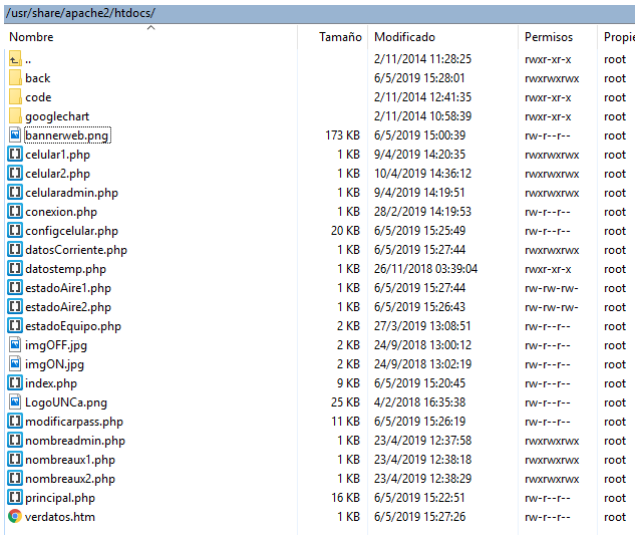
La aplicación web sirve para brindar otra manera de poder visualizar los datos recolectados por el sensor de temperatura DHT11, así también como el consumo en voltajes y amperios del aire acondicionado conectado al sensor de corriente SCT-013, todo esto a través de gráficos que se actualizan en tiempo real mientras el sistema captura la información del data center, además, posee dos indicadores sobre el funcionamiento de los aires acondicionado, indicando cuál de ellos es el que está activo en ese momento. También, permitirá administrar de manera sencilla los números de celular a los que responde el sistema para sus funciones, todo esto protegido con un sistema de cuentas de usuario que tendrán que iniciar sesión en el sistema.

4.4.1 Archivos web del Proyecto

Para que la aplicación web pueda funcionar en el sistema operativo embebido Linux Yocto de la placa Intel Galileo, previamente se instaló el servidor Web Apache y sus repositorios, luego, para que el servidor Web Apache, muestre una página web cuando se abre la conexión con un cliente, se deben colocar los archivos de la misma, en la carpeta donde Apache busca y carga automáticamente los archivos al cliente, esta ubicación en la tarjeta microSD es la siguiente (ver Figura 4-33):

- **/usr/share/apache2/htdocs**

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852



/usr/share/apache2/htdocs/

Nombre	Tamaño	Modificado	Permisos	Propie
..		2/11/2014 11:28:25	rwxr-xr-x	root
back		6/5/2019 15:28:01	rwxrwxrwx	root
code		2/11/2014 12:41:35	rwxr-xr-x	root
googlechart		2/11/2014 10:58:39	rwxr-xr-x	root
bannerweb.png	173 KB	6/5/2019 15:00:39	rw-r--r--	root
celular1.php	1 KB	9/4/2019 14:20:35	rwxrwxrwx	root
celular2.php	1 KB	10/4/2019 14:36:12	rwxrwxrwx	root
celularadmin.php	1 KB	9/4/2019 14:19:51	rwxrwxrwx	root
conexion.php	1 KB	28/2/2019 14:19:53	rw-r--r--	root
configcelular.php	20 KB	6/5/2019 15:25:49	rw-r--r--	root
datosCorriente.php	1 KB	6/5/2019 15:27:44	rwxrwxrwx	root
datostemp.php	1 KB	26/11/2018 03:39:04	rwxr-xr-x	root
estadoAire1.php	1 KB	6/5/2019 15:27:44	rw-rw-rw-	root
estadoAire2.php	1 KB	6/5/2019 15:26:43	rw-rw-rw-	root
estadoEquipo.php	2 KB	27/3/2019 13:08:51	rw-r--r--	root
imgOFF.jpg	2 KB	24/9/2018 13:00:12	rw-r--r--	root
imgON.jpg	2 KB	24/9/2018 13:02:19	rw-r--r--	root
index.php	9 KB	6/5/2019 15:20:45	rw-r--r--	root
LogoUNCa.png	25 KB	4/2/2018 16:35:38	rw-r--r--	root
modificarpass.php	11 KB	6/5/2019 15:26:19	rw-r--r--	root
nombreadmin.php	1 KB	23/4/2019 12:37:58	rwxrwxrwx	root
nombreaux1.php	1 KB	23/4/2019 12:38:18	rwxrwxrwx	root
nombreaux2.php	1 KB	23/4/2019 12:38:29	rwxrwxrwx	root
principal.php	16 KB	6/5/2019 15:22:51	rw-r--r--	root
verdatos.htm	1 KB	6/5/2019 15:27:26	rw-r--r--	root

Figura 4-33 - Archivos de la Pagina web en la microSD

Dentro de esta carpeta la aplicación web estará compuesta por diferentes archivos, a continuación se explicará los archivos index.php, principal.php, configcel.php, conexión.php, estadoEquipo.php, modificarpass.php y datosTemp.php, los cuales tienen una función importante en el funcionamiento de la aplicación web, el resto, o son imágenes que utilizan los archivos para la página, o contienen datos temporales de los sensores, que no eran necesario almacenarlos en la base de datos por su uso esporádico.

❖ Archivos PHP

Los archivos con esta extensión (.php), son aquellos que el servidor web reconoce que en su interior habrá líneas con ese lenguaje de programación, php también soporta el lenguaje HTML, así que es posible tener ambos en un solo archivo, pero es obligatorio la extensión php si deseamos que se ejecute la lógica del código php.

Dentro de estos archivos, se encuentran las etiquetas HTML, que configuran la interfaz visual de la página.

- **<html>:** Indica que el contenido es código HTML.
- **<head>:** Dentro de esta etiqueta se colocó la configuración visual de la página web, tales como el alto y ancho de los elementos, el color de fondo, también aquí se encuentran los elementos <script> ,donde se les asigna una identificación (id) para luego ser usados en contenedores dentro del <body>.
- **<body>:** Cuerpo principal de la página web, dentro se encuentran los elementos previamente configurados en <head>, también tiene tablas <table> para organizar los contenedores donde se ubican los distintos <script> que permiten mostrar los gráficos y funciones de la página.
- **<?php?>:** Envuelve código PHP, se utiliza para Programación Lógica en este lenguaje.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

En el **Anexo 3** se pueden apreciar fragmentos de los códigos fuente de cada archivo PHP.

- **Index.php**

El **index.php**, es el primer archivo que el servidor busca y envía por defecto a un cliente cuando este inicia una conexión, así que es el iniciador principal de la aplicación web, por lo tanto, es este archivo el que solicitará al cliente que inicie sesión ingresando un usuario y contraseña válidos, una vez iniciada la sesión el cliente será redirigido hacia el archivo **principal.php**.

Cuando la sesión finaliza, o el cliente intenta acceder a los demás archivos de la aplicación directamente, será redirigido al **index.php**.

- **Principal.php**

Este archivo es al cual es direccionado el usuario una vez que la sesión se inició correctamente, habilita el menú de opciones así como el acceso a las demás funciones del sistema. También, es en este archivo donde serán visibles los gráficos que estarán mostrando en tiempo real los datos obtenidos por los sensores, así también como el estado del aire acondicionado que se esté monitoreando.

- **Configcel.php**

Este archivo puede ser accedido desde el menú de opciones de la aplicación web y se encarga de brindarle al cliente, las herramientas para poder ingresar o modificar los números de celular del administrador y los auxiliares, estos números, son necesarios para las funciones de mensaje de texto del sistema, y será a estos números los cuales el sistema responderá y enviará los diversos mensajes de texto.

- **Modificarpass.php**

Este archivo puede ser accedido desde el menú de opciones de la aplicación web, permite al administrador cambiar los datos de inicio de sesión, usuario y contraseña.

- **Conexion.php**

Este archivo es solicitado cada vez que se desea abrir una conexión con la base de datos MySQL alojada en el sistema embebido Yocto Linux de la Intel Galileo. También devuelve cualquier tipo de error que se pudiera producir al realizar la conexión.

- **estadoEquipo.php**

Este archivo se encarga de realizar la verificación del estado de los aires acondicionados, se encuentra por separado del archivo **principal.php**, porque si se perciben cambios en el consumo o el estado del aire, los resultados de este archivo cambiaran, y para visualizar esos cambios, sería necesario que el cliente cargara todo el archivo de nuevo, para resolver

esto, **principal.php** posee un contenedor dinámico que se encarga de solicitar este archivo cada cierto tiempo, pudiendo visualizar en tiempo real los cambios en los valores mostrados por este archivo.

- **datostemp.php**

Este archivo realiza las consultas de los datos de temperatura, humedad, hora y fecha almacenados en la base de datos, que son utilizados por el grafico de datos históricos.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

CAPITULO 5

5 Fase de Pruebas del Sistema Embebido de telemetría

El Sistema embebido de Telemetría aún se encuentra en etapa de pruebas, a continuación se muestra los procedimientos de puesta en marcha del sistema, una vez el mismo este aprobado para su funcionamiento.

5.1 Preparación del Sistema

Estos son los pasos que se realizan, así también, los que se consideran óptimos para que el dispositivo funcione correctamente.

5.1.1 Ubicación del dispositivo

Para que el sistema pueda medir correctamente la temperatura y humedad, es recomendable colocar el dispositivo en una ubicación medianamente elevada del suelo, sin estar cerca del techo, debido a que las corrientes de aire caliente tienden a elevarse, y podría causar valores incorrectos, un punto medio seria lo adecuado para obtener valores más precisos. La Figura 5-1 muestra la ubicación recomendada del sistema una vez sea implementado, la ubicación de los aires acondicionado es a modo de ejemplo.

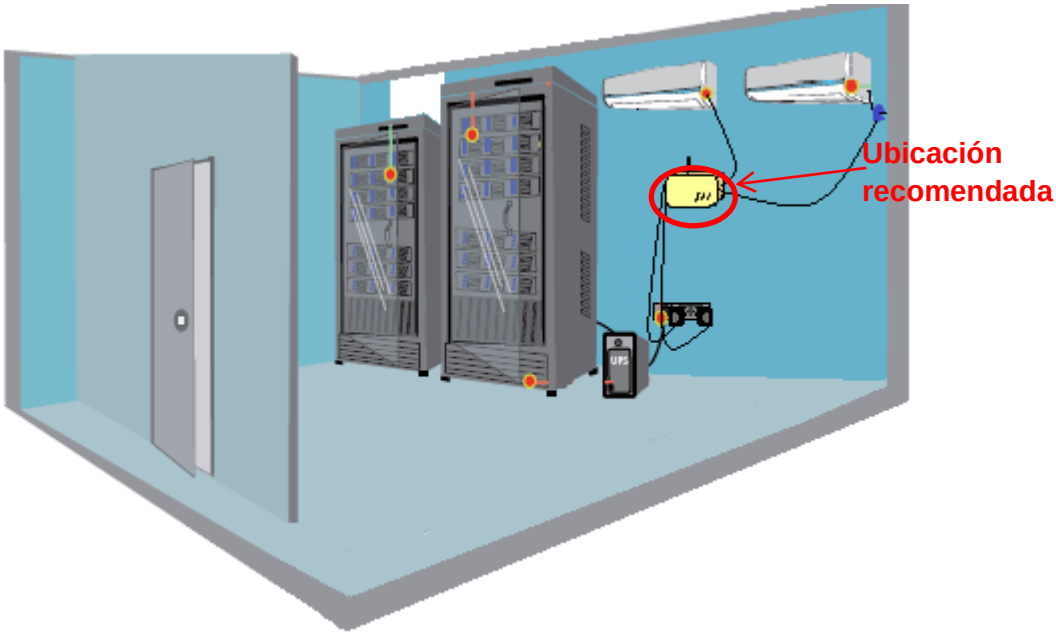


Figura 5-1 - Ubicación recomendada del dispositivo

5.1.2 Conexiones

A continuación se explicará donde se deben realizar las conexiones en el sistema y se explica la función de cada una de estas.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

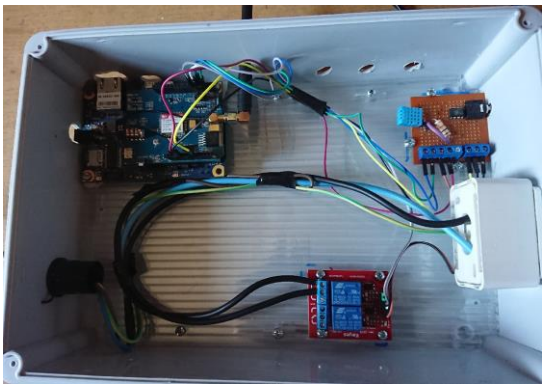


Figura 5-2 - Vista Interior del dispositivo

La placa Intel Galileo debe conectarse con su propia alimentación AC a la línea 220v, luego, se conecta también en su propia toma corrientes, el cable que le brindara energía al aire de repuesto cuando este sea activado indicado por “220v IN” como se muestra en la Figura 5-3.

Figura 5-3 - Alimentación del Sistema

Si se desea utilizar la función de monitoreo sobre un aire acondicionado, se debe hacer un corte en la cobertura del cable de alimentación del equipo, dejando expuestos los cables por separado, luego se coloca la pinza del sensor de Corriente SCT-013 con uno de los cables expuesto a través de ella como muestra la Figura 5-4. , luego se conecta la ficha Jack, en su lugar correspondiente indicado por la etiqueta “Sensor SCT 013” como se ve en la Figura 5-5. Este Aire Acondicionado, será considerado como el principal.

Figura 5-4 - Colocación sensor de corriente SCT-013

El Aire Acondicionado de repuesto debe conectarse al sistema utilizando el conector 220V OUT lateral, que se muestra en la Figura 5-5

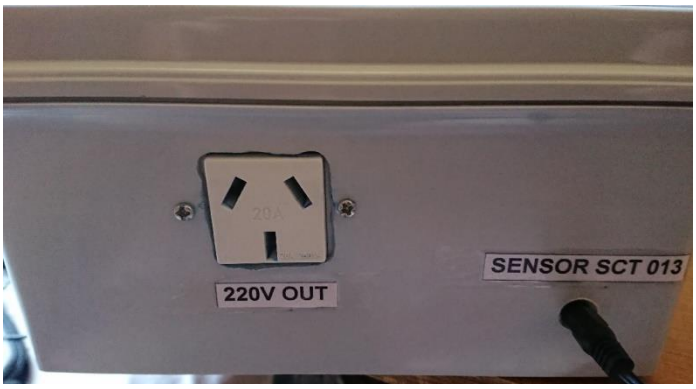


Figura 5-5 - Conexión del aire acondicionado de repuesto y sensor SCT013

Para la conexión en red de área local del sistema, se conecta un cable de tipo Ethernet ETH RJ45 en el puerto indicado por la etiqueta “Ethernet”, si se desea acceder a la placa a través de un puerto Mini-USB, se debe utilizar el puerto correspondiente a la etiqueta, ambos puertos pueden apreciarse en la Figura 5-6.

Figura 5-6 - Conexión Ethernet y Mini-USB

Si se desea reiniciar manualmente el sistema por cualquier motivo, se debe presionar el botón de **RESET**, ubicado en la tapa del gabinete, como muestra la figura 5-7.

Figura 5-7 - Vista superior y botón de RESET

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

5.2 Instrucciones de funcionamiento

5.2.1 Ingresar número de Celular

Para que el sistema pueda utilizar sus funciones de mensajería de texto SMS, se debe ingresar el número de celular de un administrador, quien será la única persona que tendrá acceso a todas las funciones y dos números de celular pertenecientes a personal auxiliar, quienes podrán consultar los datos del sistema y recibir alertas.

La Carga de estos números, puede realizarse a través de la aplicación web del sistema explicado más adelante en el artículo 5.4.2 de este capítulo.

5.2.2 Función de Mensajes de Texto, Enviar/Recibir

El sistema está diseñado para responder únicamente a los números de celular del administrador y los auxiliares, también, a un conjunto de mensajes de texto en específico, esto es así para evitar que personas ajenas a la D.T.I.C., manipulen el sistema, los mensajes y sus respuestas se listan a continuación:

- **Mensaje de Texto inicial**

Al encenderse, o después de un reinicio, el sistema enviaría un mensaje de texto informando que ya está en funcionamiento, incluirá la dirección de IP que posea en ese momento y también se solicitará si el administrador desea comenzar a monitorear el aire acondicionado conectado al sensor enviando la palabra **"ok"**, o solicitar el menú completo de opciones con la palabra **"hi"**.

Figura 5-8 - Mensaje de texto: Inicial y Monitoreo Aire

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

❖ Solicitar Menú de Opciones

Para solicitar el menú con las opciones del sistema, se debe enviar un mensaje de texto al número de celular del sistema con la palabra **"hi"**

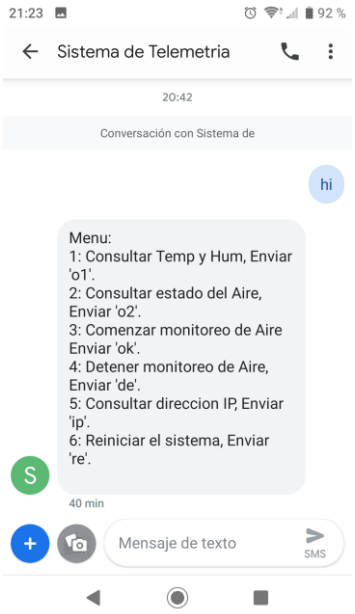


Figura 5-9 - Mensaje de Texto: Menú

❖ Solicitar Temperatura y Humedad

Para solicitar la temperatura y humedad actuales del data center, se debe enviar un mensaje de texto al número de celular del sistema con la palabra **"o1"**

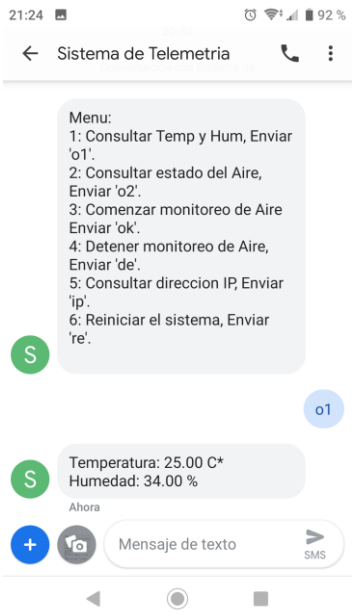


Figura 5-10 - Mensaje de Texto: Solicitar temperatura y humedad de la sala

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

❖ Solicitar estado del aire acondicionado

Para solicitar el estado del aire acondicionado monitoreado en ese momento, se debe enviar un mensaje de texto al número de celular del sistema con la palabra **"o2"**. Si el sistema no estaba controlando el aire acondicionado en ese momento, responderá con un mensaje informando esa situación y preguntará si se desea monitorear el aire acondicionado (Figura 5-11).

Figura 5-11 - Mensaje de Texto: Estado aire acondicionado

❖ Solicitar dirección de IP del sistema

Para consultar la dirección de IP actual del sistema, se debe enviar un mensaje de texto con la palabra **"ip"**.

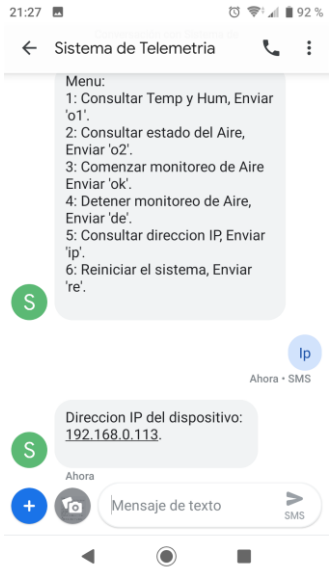


Figura 5-12 - Mensaje de Texto: Dirección IP del Sistema

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

❖ Solicitar detener monitoreo de aire acondicionado

Para solicitar que el sistema deje de monitorear el aire acondicionado conectado al sensor de corriente, el administrador debe enviar un mensaje de texto al número de celular del sistema con la palabra **"de"**.

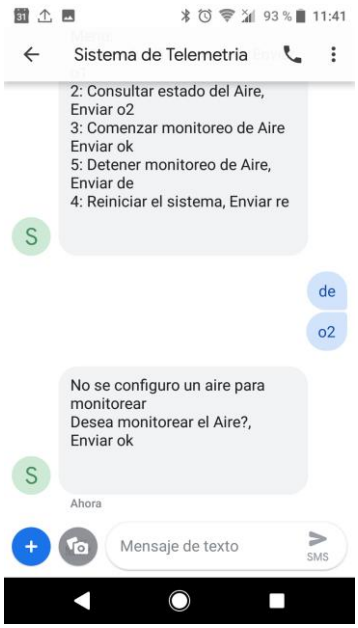


Figura 5-13 - Mensaje de Texto: Detener monitoreo de Aire Acondicionado

❖ Solicitar Reinicio del Sistema

Para solicitar el reinicio del sistema, se debe enviar un mensaje de texto al número de celular del sistema con la palabra **"re"** Solo el administrador podrá dar esta orden al sistema.

Figura 5-14 - Mensaje de Texto: Reiniciar sistema

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

❖ Alerta de Temperatura

Si el sistema detecta que la temperatura es demasiado alta, enviará automáticamente un mensaje de texto a los números registrados.

Figura 5-15 - Mensaje de Texto: Alerta Temperatura

❖ Alerta de Aire Acondicionado

Si el sistema detecta que el aire acondicionado al cual se está monitoreando deja de funcionar, enviará un mensaje de alerta a los números registrados de manera automática, el administrador podrá intentar encender el aire de repuesto respondiendo con la palabra **"si".**

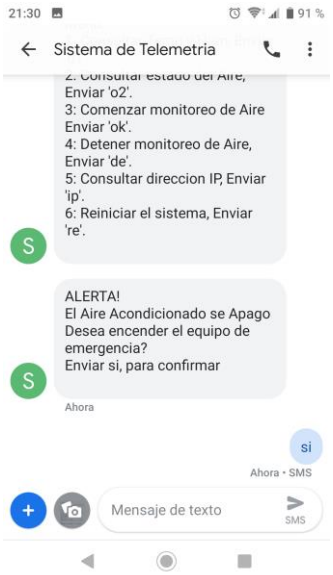


Figura 5-16 - Mensaje de Texto: Alerta de estado de aire acondicionado

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

5.2.3 Interfaz de Servicio Web

Para acceder a la aplicación web del sistema, solamente se debe introducir la dirección IP que le fue asignado al dispositivo por el router de la red de área local.

A continuación se mostrará la aplicación web del sistema embebido de telemetría indicando con referencias, las distintas partes de la misma.

- **Página Gráficos**

Figura 5-17 - Aplicación web: Gráficos – superior

1. **Banner:** Cabecera de la aplicación web, incluye el título del proyecto, los logos de la U.N.Ca. y la D.T.I.C.
2. **Botón Gráficos:** Permite acceder desde el side bar a la página web del sistema que contiene los gráficos.
3. **Botón Modificar Usuario/Clave:** Permite acceder desde el side bar, a la página web del sistema que contiene las herramientas para poder cambiar el usuario o la clave para iniciar sesión a la aplicación web del sistema.
4. **Botón Cambiar número de celular:** Permite acceder desde el side bar a la página web del sistema con las herramientas para ingresar o modificar los números de celular almacenados en el sistema.
5. **Botón Cerrar Sesión:** Cierra la sesión actual, cualquier nuevo acceso, requerirá que el cliente ingrese su usuario y contraseña nuevamente.
6. **Gráficos de reloj Calibre:** Muestran los datos de Temperatura y Humedad medidos por el sensor DHT11.
7. **Consumo y Estado:** Datos recolectados del consumo del aire acondicionado por el sensor SCT-013, también se muestran indicadores del estado del equipo principal conectado al sensor, y el secundario.

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Figura 5-18 - Aplicación web: Gráficos - inferior

8. **Campo de Fecha:** Permite ingresar una fecha específica para mostrar los datos recolectados ese día.
9. **Grafico Histórico:** Se utiliza un gráfico de línea para expresar los datos históricos recolectados por el sensor de temperatura y humedad, por defecto, al acceder en la página web, mostrará los resultados obtenidos hasta ese momento en el día de hoy, hasta que se ingrese una fecha utilizando el campo fecha. Tambien permite crear un archivo pdf o una imagen formato jpeg, png, svg del gráfico.
10. **Footer:** Pie de página, marca el final de la página web, también se utiliza para organizar los elementos.

❖ **Página Modificar Usuario/Clave**

Figura 5-19 – Aplicación Web: Modificar Usuario/Clave

1. **Formulario:** Se utiliza para brindarle al usuario campos para poder ingresar los datos que necesite modificar.
2. **Botón Aceptar:** Envía el formulario al sistema para actualizar los datos ingresados, si los nuevos datos se actualizan satisfactoriamente una ventana emergente le informará al usuario de esto, asi mismo, si hubo un error, se le informará al usuario.

❖ **Página Cambiar Número de Celular:**

Figura 5-20 - Aplicación web: Cambiar número de celular - superior

1. **Tabla números registrados:** En esta tabla se visualiza los números que están registrados actualmente en el sistema embebido.

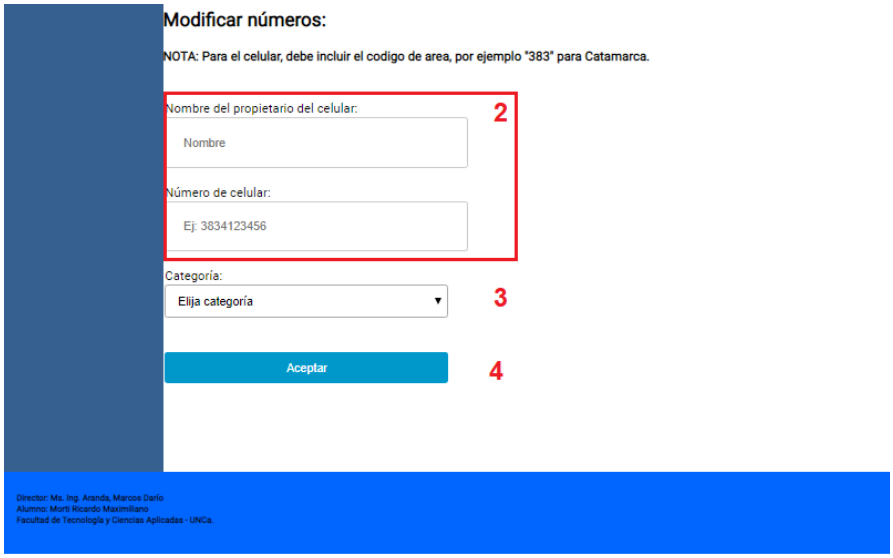


Figura 5-21 - Aplicación Web: Cambiar número de celular - inferior

2. **Formulario:** Campos donde el usuario puede ingresar el número de celular y el nombre del propietario de ese número.
3. **Botón lista Categoría:** Permite elegir la categoría del número de celular, si es del administrador, del auxiliar 1 o del auxiliar 2.
4. **Botón Aceptar:** Envía los datos ingresados por el usuario, si la categoría elegida previamente contenía datos, el sistema los reemplazará por los nuevos datos modificados, ya sean el nombre o el número.

5.3 Etapas de Pruebas

El sistema se encuentra en etapas de pruebas, para evaluar su desempeño, se lo colocó en un ambiente controlado (Figura 5-22), los aires acondicionados son remplazados por lámparas de escritorio, que a fines prácticos, son más sencillas de manipular, además de prevenir posibles daños a los equipos de aire acondicionado.

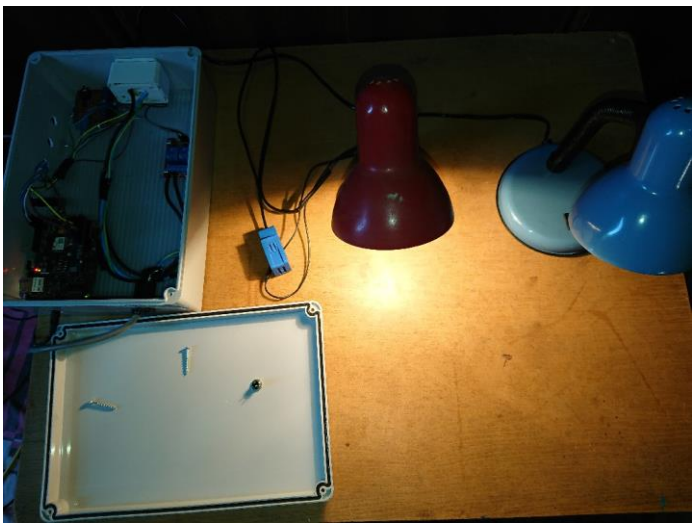


Figura 5-22 - Área de Pruebas

5.3.1 Pruebas del sensor de Temperatura y Humedad DHT11

Para el sensor de temperatura, se realizaron pruebas utilizando el calor emitido por las lámparas de escritorio (Figura 5-23) para evaluar el comportamiento del sensor ante los cambios abruptos de temperatura y asegurar que el sistema responda enviando las alertas necesarias.

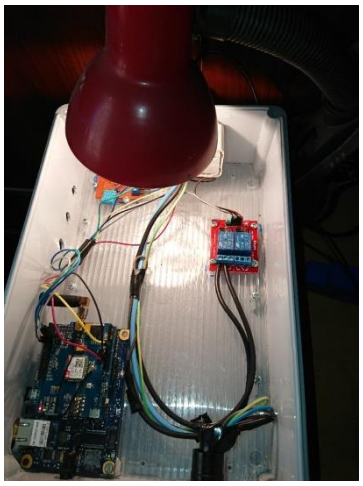


Figura 5-23 - Prueba de Temperatura

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

❖ **Resultados:**

El sensor DHT11 presentó un tiempo de respuesta aceptable ante los cambios de temperatura inducidos, el sistema pudo detectar estos cambios y reaccionó de la manera esperada enviando las alertas por mensajes de texto, tampoco se observaron anomalías en el funcionamiento de la aplicación web mientras muestra los datos del sensor.

5.3.2 Pruebas del sensor de corriente SCT-013-000

Para evaluar el sensor de corriente, se utilizaron las lámparas de escritorio como reemplazo de los equipos de aire acondicionado como se había mencionado (Figura 5-24), luego, se observó los valores de voltaje y amperaje que se obtuvieron del sensor SCT-013-000, así también como la reacción del sistema cuando la lámpara monitoreada por el sensor deja de funcionar.

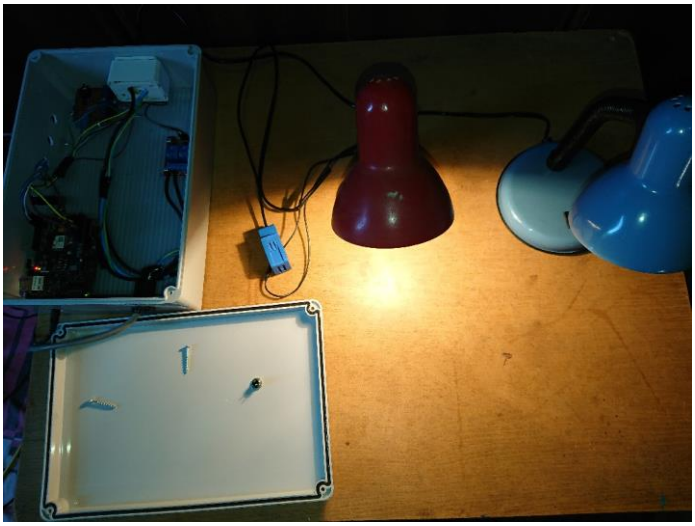


Figura 5-24 – Prueba de Sensor de Corriente SCT-013-000, Encendido

❖ **Resultados:**

Se utilizó una calculadora de vatios y amperios para establecer una tabla de referencia con los valores de consumo de algunos aparatos eléctricos, como muestra la tabla 5-1, luego utilizando las variables de calibración en el código del sistema, se configuró el sensor para acercarse lo más posible a estos valores.

Aparato	Potencia en vatios (W)	Voltaje en Voltios (V)	Corriente en Amperios (A)
Lámpara de Escritorio	70	220	0.31818181818
Aire Acondicionado Split de 2500	2500	220	11.818181818
Plancha	2000	220	9.0909090909

Tabla 5-1 - Valores de referencia potencia Eléctrica

En la Figura 5-25, se puede apreciar los datos obtenidos por el sensor mostrado en la aplicación Web, mientras monitoreaba la lámpara y su consumo.

Figura 5-25 - Prueba del sensor SCT- 013, pantalla Aplicación Web encendido

Los valores obtenidos, son aceptables ya que se acercan a los de una lámpara de escritorio como indica la tabla de referencia.

En el siguiente paso, se realizó la prueba de comportamiento cuando la lámpara conectada al sensor, se apaga (Figura 5-26), el sistema utilizará los valores del sensor para determinar el estado de la lámpara. La prueba resulto aceptable, el dispositivo pudo detectar estos cambios en el funcionamiento de la lámpara, para luego enviar los mensajes de alerta. En la figura 5-27, se puede apreciar la aplicación web después de encender la segunda lámpara.

Figura 5-26 - Prueba de Sensor de Corriente SCT-013-000, Apagado

Figura 5-27 - Prueba del sensor SCT-013, pantalla Aplicación Web apagado

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Se pudo observar que el tiempo de respuesta del sensor fue apropiado para detectar que la lámpara estaba encendida, pero el sensor presento un tiempo notable para reaccionar cuando la lámpara deja de funcionar, debido a que el transformador dentro del sensor necesita descargarse y lo hace progresivamente, como pudo observarse en la Figura 5-27, existe corriente residual que es detectada por el sensor. Este tiempo se usó a favor del funcionamiento del sistema, debido a que la red eléctrica puede presentar bajas en los picos de tensión, los que podrían provocar falsas alertas si son detectados de inmediato, la descarga progresiva del sensor, permite descartar estas medidas y aumentar la precisión para detectar alertas reales.

5.3.3 Pruebas del módulo GSM SIM800C GSM/GPRS

Para evaluar el funcionamiento del módulo, se utilizaron los comandos de mensajes de texto establecidos en el código, luego se observó los tiempos de respuesta del módulo, tanto para recibir como para enviar los mensajes de texto.

- **Resultados**

Se observó tiempos de respuesta de un promedio entre 5 a 10 segundos entre mensajes, tanto para enviar como para recibir, esto se debe, a que los mensajes de texto, deben ser procesados por la red celular, estos tiempos escapan al control del sistema, se concluyó configurar el código teniendo en cuenta estos tiempos, informando al usuario de esta posible demora en la respuesta del sistema.

CAPITULO 6

6 Resultados Alcanzados

De acuerdo al objetivo principal fijado en este proyecto, el cual fue crear un Sistema de Telemetría con un sistema embebido, que capture temperatura, humedad y consumo eléctrico, que permitiese la consulta de estos datos a distancia, y el control del data center con alertas ante situaciones no deseadas, el objetivo se cumplió parcialmente ya que no se pudo implementar el sistema de Telemetría en la D.T.I.C. U.N.Ca., debido a que los tiempos administrativos para la aprobación del uso y prueba del dispositivo, fueron mayores a las plazos de entrega del presente informe de Trabajo Final, se espera que los permisos gestionados sean otorgados sin mayor inconvenientes en un futuro cercano a la finalización del mencionado informe.

Si bien, en un principio la plataforma Intel Galileo, presentó ciertos inconvenientes para configurarse con los elementos del sistema utilizados, la gran capacidad y potencial de la plataforma abre el camino a futuras mejoras, tanto en hardware por su compatibilidad con el universo de módulos Arduino, como en software, gracias al sistema operativo embebido.

El desarrollo de este Trabajo Final, además de cumplir en gran parte con el objetivo principal, también permitió ampliar y profundizar los conocimientos sobre el manejo de Sistemas Embebido, así como, incorporar conocimientos de electrónica, reforzando la noción de como la informática es una disciplina muy abarcadora.

CAPITULO 7

7 Conclusiones

a) El relevamiento de datos expuso un gran potencial en la información que puede ser recolectada del estado de un data center y como es necesario un método para utilizarla y facilitar el acceso de la misma al personal del lugar para brindar comodidad y tranquilidad.

b) El ecosistema de módulos Arduino ofrece una gran cantidad de componentes como placas y sensores para realizar un sinfín de tareas, y a su vez, ofreciendo la misma variedad en costos, lo que facilita encontrar el hardware adecuado, esto abre paso a expandir las funciones del sistema con el surgimiento de nuevas necesidades en el data center.

c) La implementación de los componentes del proyecto a la plataforma Intel Galileo presentó ciertas dificultades ya que se debieron realizar configuraciones previas, tanto en algunos componentes como en la misma Intel Galileo, esto se debe a que la plataforma, al presentar una mayor potencia que la mayoría de las placas de la familia Arduino y un sistema operativo embebido, causa que la comunicación entre la placa y los componentes sea diferente, pero una vez se superó esta contrariedad, se pudo aprovechar el gran potencial que ofrece la plataforma Intel Galileo, dando la bienvenida a cualquier posible mejora en el futuro del proyecto, siendo las más destacables el módulo Wi-Fi que le brindará al sistema la capacidad de conectarse inalámbricamente, un sensor de humo y un sensor de puertas y ventanas.

d) El módulo SIM800C GSM/GPRS, tuvo que ser configurado en su hardware para poder conectarse a la plataforma Intel Galileo, debido a la diferencia entre ambos para la comunicación de datos. A diferencia de Arduino, que ofrece una librería para hacer uso de funciones de mensajería GSM/GPRS, en la Intel Galileo esta no es posible de utilizar, al principio, esto supuso el inconveniente de realizar los comandos manualmente, pero resulto ser favorable, ya que facilitó llevar un mejor control en la comunicación del módulo y la plataforma mejorando las tareas de depuración de errores, de esta forma, el módulo SIM800C GSM/GPRS, demostró ser adecuado para el proyecto.

e) La implementación del módulo de relé/relay, no presento problemas, esto brindó al sistema la posibilidad de controlar el encendido y apagado de cualquier aparato, sin embargo, para considerar los altos consumos y voltajes que utilizan los equipos de refrigeración, es necesario conectar un complemento en el circuito eléctrico que sea capaz de manejar estas cantidades de corriente, se concluyó que una contactora sería lo adecuado.

f) El IDE de Arduino es una herramienta que brinda de manera sencilla la posibilidad de crear código C++ para las placas de micro-controladores de Arduino, a su vez nos ofreció compatibilidad con la plataforma Galileo, siendo la instalación de los controladores la configuración más destacable que se tuvo que realizar sobre el IDE, este programa sirvió de manera satisfactoria al propósito de crear el código para el Sistema Embebido de Telemetría.

g) El Sistema se encuentra en su primera versión y aún en su etapa de pruebas previa a la implementación, las pruebas realizadas dieron resultados alentadores para avanzar en las siguientes etapas de desarrollo.

h) La ideología del internet de las cosas nos permitió considerar la integración de plataformas como la Intel Galileo, para todo tipo de situaciones cotidianas, como la seguridad y control de lugares u otros dispositivos que cuenten con valiosa información como un data center, por lo tanto, el uso de Sistemas Embebidos para gestionar la información que estos objetos pudieran generar así también como crear medios para poder combinar sus funcionalidades, nos permite concluir que estos sistemas serán una parte muy importante para el futuro desarrollo de la Informática.

i) Gracias al potencial que ofrecen los sistemas embebidos como base para utilizar las plataformas, es factible decir que, como bien afirma la ideología de IoT, en un futuro cercano, estaremos completamente rodeados de objetos conectados a internet, y serán los lenguajes de programación los que jugaran el papel más importante para crear aplicaciones y servicios que se usaran para formar parte del IoT, este proyecto permitió reforzar los conocimientos aprendidos de estos lenguajes.

8 Referencias

1. ¿Qué es el IoT (Internet de las Cosas)? [En línea] [Citado el: 17 de Enero de 2018] https://www.domodesk.com/221-a-fondo-que-es-iot-el-internet-de-las-cosas.html.

2. UK Space Agency [En línea][Citado el: 17 de Enero de 2018] https://www.gov.uk/government/organisations/uk-space-agency

3. Radio Comunicaciones: Radio & Engineering Company SL [En línea][Citado el: 19 de Enero de 2018] http://www.radiocomunicaciones.net/radio/telemetria/

4. ¿Qué es MySQL? [En línea][Citado el: 20 de Enero de 2018] http://www.esepestudio.com/noticias/que-es-mysql.

5. **BARR, Michael y MASSA, Anthony.** *Programming Embedded Systems with C and GNU Development Tools.*

6. **SHIBU, V K.** *Introduction to Embedded Systems.* s.l.: MacGraw Hill Education Private Limited, 2009.

7. **Navarro, Mario Alberto**. Integración de sistemas embebidos y web services para la automatización de la carga de signos vitales en el SIGeSa. Facultad de Tecnologías y Ciencias Aplicadas, Universidad Nacional de Catamarca., 2017

8. **Noergaard, Tammy**. Embedded Systems Arquitecture - A Compresive Guide for Engineers and Programmers. s.l.: Elsevier Inc., 2005.

9. Galileo Feature Sheet [En línea][Citado el: 19 de Septiembre de 2018] https://www.ibm.com/developerworks/ssa/webservices/newto/service.html

10. Arduino Reference [En línea][Citado el: 19 de Septiembre de 2018] https://www.arduino.cc/reference/en/#functions

11. DHT11 Humidity & Temperature Sensor – Mouser Electronics [En línea][Citado el 11 de Noviembre de 2018] https://www.mouser.com/ds/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf

12. SCT-013 Datasheet – MCI Electronics[En línea][Citado el 11 de Noviembre de 2018] https://www.mcielectronics.cl/website_MCI/static/documents/Datasheet_SCT013.pdf

13. Open Hacks | Open Source Hardware | Productos – SIM800C GPRS/GSM - Datasheet[En línea][Citado el 13 de Noviembre de 2018] https://www.openhacks.com/uploadsproductos/sim800c_hardware_design_v1.02.pdf

14. Open Hacks | Open Source Hardware | Productos – SIM800C GPRS/GSM – At_command_manual[En línea][Citado el 13 de Noviembre de 2018] https://www.openhacks.com/uploadsproductos/sim800_series_at_command_manual_v1_wf.pdf

15. **Pankaj, Jalote.** *A Concise Introduction to Software Engineering.* London: Springer, 2008.

16. **Cataldi,Z., Lage, F.,Pessacq, R.** y **García Martínez, R**. *Ingeniería de* software *educativo*.[En Línea][Citado el 15 de Septiembre de 2018] http://web.archive.org/web/20131229044335/http://www.iidia.com.ar/rgm/comunicaciones/c-icie99-ingenieriasoftwareeducativo.pdf

17. **Jacobson, Ivar, Booch, Grady** y **Rumbaugh, James**. El Proceso Unificado de Desarrollo de Software. s.l. : Addison Wesley, 2006

18. **A.U.S. Gustavo Torossi.** El Proceso Unificado del desarrollo del Software.[En Línea][Citado el 14 de noviembre de 2018] http://dsc.itmorelia.edu.mx/~jcolivares/courses/pm10a/rup.pdf

19. Programación en C++/Introducción[En Línea][Citado el 20 de Noviembre de 2018] https://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C%2B%2B/Introducci%C3%B3n

20. C++ [En Línea][Citado el 20 de Noviembre de 2018] https://es.wikipedia.org/wiki/C%2B%2B#cite_ref-stroustruptcpppl_1-0

21. ¿Qué es y para qué sirve HTML? El Lenguaje más Importante para crear páginas webs.[En línea][Citado el 20 de Noviembre de 2018] https://www.aprenderaprogramar.es/index.php?option=com_content&view=article&id=435:iqu e-es-y-para-que-sirve-html-el-lenguaje-mas-importante-para-crear-paginas-webs-html-tags-cu00704b&catid=69:tutorial-basico-programador-web-html-desde-cero&Itemid=192

22. **Morteo, Bocalandro, Francisco, Nicolás**. Un enfoque práctico de SQL. Ediciones Cooperativas, 2004.

23. **Myke Chapple.** Introduction of SQL fundamentals.[En línea][Citado el 22 de Noviembre de 2018] https://www.lifewire.com/sql-fundamentals-1019780

24. Que es el lenguaje HTML[En Línea][Citado el 27 de Noviembre de 2018] http://www.ri5.com.ar/ayuda03.php

25. On SGML and HTML[En línea][Citado el 27 de Noviembre de 2018] https://www.w3.org/TR/html401/intro/sgmltut.html#h-3.2.2

26. ¿Qué es PHP?[En línea][Citado el 27 de Noviembre de 2018] http://php.net/manual/es/intro-whatis.php

27. SCT-013 Mide el consumo eléctrico en tu casa con Arduino [En linea][Citado el 27 de Noviembre de 2018] https://programarfacil.com/blog/arduino-blog/sct-013-consumo-electrico-arduino/

9 Bibliografía

1. **Newcomer, Eric**. Understanding Web Services- XML, WSDL, SOAP and UDDI. s.l.: Independent Technology Guides.

2. **Totty, David Gourley & Brian.** HTTP - The Definitive Guide. s.l.: O'Reilly Media, Inc, 2002.

3. **BARR, Michael y MASSA, Anthony**. Programming Embedded Systems with C and GNU Development Tools.

4. **SHIBU, V K**. Introduction to Embedded Systems. s.l.: MacGraw Hill Education Private Limited, 2009.

5. **Noergaard, Tammy.** Embedded Systems Arquitecture - A Compresive Guide for Engineers and Programmers. s.l.: Elsevier Inc, 2005.
6. **Pankaj, Jalote. A** Concise Introduction to Software Engineering. London: Springer, 2008.

7. **Jacobson, Ivar, Booch**, **Grady y Rumbaugh, James**. El Proceso Unificado de Desarrollo de Software. s.l.: Addison Wesley.

8. **S.Tanenbaum, Andrew**. Sistemas Distribuidos Principios y Paradigmas. Mexico: PEARSON EDUCACION, 2008.

Anexos

Anexo 1: Comandos AT

A Continuación, se listaran los comandos AT utilizados para el sistema embebido de telemetría, su descripción y respuestas:

Comando: AT+CMGF	
Descripción:	Establece el modulo en formato SMS, para él envió y recibimiento de mensajes de texto en código ASCII
Comando de Prueba: **AT+CMGF=?**	Respuesta: **+CMGF:** (lista de los modos compatibles) **OK**
Comando de Lectura: **AT+CMGF?**	Respuesta: **+CMGF:<mode>** **OK**
Comando de Escritura: **AT+CMGF=<mode>**	Respuesta: (TA, establece el parámetro de entrada y salida a usar en los mensajes) **OK**
	Parámetros: **<mode>** 0 modo PDU (Hexadecimal) 1 modo Texto
Comando: AT+CMGD	
Descripción:	Borra mensaje/s de texto almacenados en la tarjeta SIM
Comando de Prueba: **AT+CMGD=?**	Respuesta: **+CMGD:** (lista <index> compatibles), (lista de <deflag>s compatible) **OK**
Comando de Escritura: **AT+CMGD=<index>[,<deflag>]**	Respuesta: TA, borrará los mensajes del lugar de almacenamiento de preferencia **<mem1>** en posición **<index>** **OK** **ERROR** Si se produce un error relacionado al funcionamiento del ME **+CMS ERROR:<err>** (código de error)
	Parámetros: **<index>** Tipo entero; valor en el rango de posiciones soportado por la memoria asociada

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

	<deflag> **0** Borra el mensaje especificado en <index> **1** Borra todos los mensajes del lugar de almacenamiento de preferencia, pero ignora los mensajes sin leer y los almacenados de origen móvil (ya sean enviados o no). **2** Borra todos los mensajes del lugar de almacenamiento de preferencia y los mensajes desde móvil enviados, dejando los mensajes sin leer y los de móvil sin enviar. **3** Borra todos los mensajes del lugar de almacenamiento de preferencia, los enviados y sin enviar desde móvil, ignorando los mensajes sin leer. **4** Borra todos los mensajes del lugar de almacenamiento de preferencia, incluido los mensajes sin leer.
Comando: AT+CMGR	
Descripción:	Lee mensajes de texto.
Comando de Prueba: **AT+CMGR=?**	Respuesta: **OK**
Comando de Escritura: **AT+CMGR=<index>[,<mode>]**	Parámetros: **<index>** Tipo entero; valor en el rango de posiciones soportado por la memoria asociada **<mode>** 0 Normal 1 Sin cambios en el estado del SMS almacenado
	Respuesta: El TA devuelve el mensaje SMS con el valor de la posición **<index>** en el almacenamiento de preferencia **<mem1>** al TE. Si el estado del mensaje es 'received unread' (recibido

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

<table>
<tr><td></td><td colspan="2">sin leer), el estado en el almacenamiento se cambia a ‘received read’ (recibido leído).
1) Si el modo texto (+CMGF=1) y el comando es exitoso para:
Mensaje-entregado:
+CMGR:
<stat>,<oa>[,<alpha>],<scts>[,<tooa>,<fo>,<pid>,<dcs>
,<sca>,<tosca>,<length>]<CR><LF><data>
Mensaje-enviado:
+CMGR:
<stat>,<da>[,<alpha>][,<toda>,<fo>,<pid>,<dcs>[,<vp>]
,<sca>,<tosca>,<length>]<CR><LF><data>
Reporte de estado del mensaje:
+CMGR:
<stat>,<fo>,<mr>[,<ra>][,<tora>],<scts>,<dt>,<st>
Comandos del mensaje:
+CMGR:
<stat>,<fo>,<ct>[,<pid>[,<mn>][,<da>][,<toda>]
Almacenamiento CBM:
+CMGR:
<stat>,<sn>,<mid>,<dcs>,<page>,<pages><CR><LF><data>
2) Si el modo PDU (+CMFG=0) y el comando es exitoso para:
+CMGR: <stat>[,<alpha>],<length><CR><LF><pdu>

OK

3) Si existe error relacionado al funcionamiento del ME:
+CMS ERROR: <err></td></tr>
<tr><td></td><td colspan="2">Parámetros:</td></tr>
<tr><td></td><td><alpha></td><td>Tipo cadena de caracteres ‘string’ (el string debe ser incluido entre comillas), representación alfanumérica de <da> o <oa> correspondiente a la entrada encontrada en el directorio telefónico de MT; la implementación de esta caracteriza es especifica del fabricante.</td></tr>
<tr><td></td><td><da></td><td>Campo con formato cadena de caracteres, con la especificación GSM 03.40 TP-Dirección Destino-Valor Direccion; los números BCD (o los caracteres alfabéticos por defecto de GSM) son convertidos a caracteres del TE seleccionado actualmente (especificado por +CSCS en 3GPP TS 27.00); el tipo</td></tr>
</table>

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

		de dirección indicado en **<toda>**
	<data>	En el caso de un SMS: GSM 03.40 TP-User-Data en modo texto, responde; formato: -Si **<dcs>** indica que el alfabeto por defecto GSM 03.38 está en uso y **<fo>** indica que GSM 03.40 TPUser-Data-Header-Indication no está establecido: -Si el carácter SE TE establece distinto a "HEX" (refiriéndose al Comando Select TE Character Set +CSCS in 3GPP TS 27.007):ME/TA convierte el alfabeto GSM en un conjunto de caracteres TE de acuerdo a las reglas del Anexo A -Si conjunto de caracteres TE está establecido en "HEX": ME/TA convierte cada carácter de 7-bit del alfabeto GS; en dos caracteres de tipo numero hexadecimal long IRA(ej. El Carácter 'P' (GSM 23) se presenta como 17 (IRA 49 y 55)) -si **<dcs>** indica que el esquema de codificación de la información es de 8bit o UCS2, o **<fo>** indica que está establecido en GSM 30.40 TP-Use-Data-Header-Indication: ME/TA convierte cada octeto de 8-bit en dos caracteres de tipo numero hexadecimal long IRA (ej. El octeto con el valor entero 42, se presenta en TE como dos caracteres 2ª (IRA 50 y 65)) en el caso de CBS: GSM 03.41, Content of Message en modo texto; formato: -si **<dcs>** indica que GSM 03.38 es el alfabeto en uso por defecto: -si el conjunto de caracteres TE está establecido de otra forma que no sea "HEX" (refiriéndose a Command +CSCS in #GPP TS 27.0007): ME/TA convierte el alfabeto GSM en un conjunto de caracteres TE de acuerdo a las reglas del Anexo A. -Si el conjunto de caracteres está establecido en "HEX": ME/TA convierte cada carácter de 7-bit del alfabeto GSM en dos caracteres de tipo numero hexadecimal long IRA -Si **<dcs>** indica que el esquema de codificación de la información en uso

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

		es 8-bit o UCS2: ME/TA convierte cada octeto de 8-bit en dos caracteres de tipo numero hexadecimal long IRA
	<dcs>	Dependiendo del comando o código del resultado: GSM 03.38 SMS Esquema de Codificación de la Información (por defecto 0), o Cell Broadcast Data Coding Scheme en formato entero.
	<fo>	Dependienta del comando o código del resultado: el primer octeto de GSM 03.40 SMS-DELIVER, SMS-SUBMIT(por defecto 17), SMS-STATUS-REPORT, o SMS-COMMAND(por defecto 2) en formato entero)
	<lenght>	Valor entero que se indica en el modo de texto (+CMGF=1) La longitud del cuerpo del mensaje **<data>** (o **<cdata>**) en caracteres; o en modo PDU (+CMFG=0), la longitud de la unidad TP en octetos (ej. La capa RP y la dirección SMSC no están incluidas en la longitud)
	<mid>	GSM 03.41 CBM, identificador del mensaje en formato entero
	<oa>	GSM 03.40 TP-Originating-Address-Addres-Value, campo en formato string; números BCD(o alfabeto de caracteres GSM por defecto) son caracteres convertidos al conjunto de caracteres establecido por TE (especificado por +CSCS en 3GPP TS 27.007); tipo de dirección proporcionado por **<tooa>**
	<pdu>	En el caso de la dirección SMS: GMS 04.11 SC seguida de GSM 03.40 TPDU en formato hexadecimal: ME/TA convierte cada octeto de información TP en dos caracteres de tipo numero hexadecimal long IRA (ej. El octeto con el valor entero 42 se presenta a TE como dos caracteres 2ª (IRA 50 y 65)). En el caso de CBS: GSM 03.41 TPDU en formato hexadecimal.
	<pid>	GSM 03.40 TP-Protocol-Identifier en formato entero (por defecto 0)
	<sca>	GSM 04.11 RP SC Adress Adress, campo con valor en formato string;

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

		números BCD (o alfabeto de caracteres GSM por defecto) son convertidos a caracteres del conjunto de caracteres establecidos por TE(especificado por +CSCS en 3GPP TS 27.007); tipo de dirección proporcionado por **<tosca>**
	<scts>	GSM 03.40 TP-Service-Centre-Time-Stamp en formato time-string (referencia **<dt>**)
	<stat>	0 “REC UNREAD” Mensajes recibidos sin leer 1 “REC READ” Mensajes leídos recibidos 2 “STO UNSENT” Mensajes sin leer almacenados 3 “STO SENT” Mensajes leídos almacenados 4 “ALL” Todos los mensajes
	<toda>	GSM 04.11 TP-Destination-Address Type-of-Address octeto en formato entero (cuando el primer carácter **<da>** es +(IRA 43) por defecto es 145, de otra forma por defecto es 129)
	<tooa>	GSM 04.11 TP-Originating-Address Type-of-Address octeto en formato entero (referencia a **<toda>** por defecto)
	<tosca>	GSM 04.11 RP SC Address Type-of-Address octeto en formato entero(referencia a **<toda>** por defecto)
	<vp>	Dependiendo de SMS-SUBMIT **<fo>** establecido: GSM 03.40 TP-Validity-Period podrá ser tanto en formato entero (por defecto 167) o en formato time-string(referencia**<dt>**)
Comando: AT+CMGS		
Descripción:	Envía un mensaje	
Comando de Prueba: **AT+CMGS=?**	Respuesta **OK**	
Comando de Escritura: 1) Si está en modo texto (+CMGS=1): **+CMGS=<da>**[,**<toda>**]**<CR>T**	Parámetros:	
	<da>	GSM 03.40 TP-Destination-Address Address-Value campo en formato string(string, debe

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

<table>
<tr>
<td rowspan="4">exto introducido<ctrl-Z/ESC>
ESC cancela sin enviar.

2) Si está en modo PDU (+CMGS=0): +CMGS=<length><CR>Se introduce PDU<ctrl-Z/ESC></td>
<td></td>
<td>ser incluido entre comillas); números BCD (o caracteres del alfabeto GSM por defecto) son convertidos en el conjunto de caracteres establecidos por TE (especificado por +CSCS en 3GPP TS 27.007); el tipo de dirección es proporcionado por <toda></td>
</tr>
<tr>
<td><toda></td>
<td>GSM 04.11 TP-Destination-Address Type-of-Address octeto en formato entero(cuando el primer carácter de <da> es +(IRA43) por defecto es 145), de otra forma, por defecto es 129)</td>
</tr>
<tr>
<td><length></td>
<td>Valor de tipo entero (no excede los 160 bytes) indicando en modo texto (+CMGF=1) la longitud del cuerpo del mensaje <data> (o <cdata>) en caracteres; o en modo PDU (+CMGF=0), la longitud de la unidad de información TP actual en octetos (ej. Los octetos de la capa RP en la dirección SMSC no se consideran en la longitud)</td>
</tr>
<tr>
<td colspan="2">Respuesta:

TA envía el mensaje de TE a la red (SMS-SUBMIT).
El Valor de referencia del mensaje <mr>, es devuelto al TE si el mensaje se envió con éxito.
Opcionalmente (cuando el valor de +CSMS <service> es 1 y la red soporta esta función) se devuelve <scts>. Los valores pueden ser utilizados para identificar al mensaje hasta el resultado del código de estado de entrega

1) Si está en modo texto(+CMGF=1) y el envío es exitoso:
+CMGS: <mr>

OK
2) Si está en modo PDU(+CMGF=0) y el envío es exitoso
+CMGS: <mr>

OK

3) Si hay un error relacionado al funcionamiento de ME:</td>
</tr>
</table>

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

	+CMS ERROR: <err>	
	Parámetro: **<mr>**	GSM 03.40 TP-Message-Reference en formato entero

Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Anexo 2: Código Fuente Firmware

Se adjunta un fragmento del código fuente del firmware del Sistema Embebido, se muestran la declaración de las librerías utilizadas, la definición de variables globales y la función setup().

```
//Librerias
#include <Ethernet.h>
#include "DHT.h"              //Sensor de Temperatura y Humedad DHT
#include <iostream>           //Librerias para el manejo y conversion de strings
#include <fstream>
#include <sstream>
#include <string>
#include "EmonLib.h"          //Sensor de corriente SCT013
#include <SPI.h>

//Variales sensor temperatura DHT
#define DHTIN 2             //PIN de conexión del sensor DHT IN
#define DHTOUT 3            //PIN de conexión del sensor DHT OUT
#define DHTTYPE DHT11       //Definición del tipo de sensor

DHT dht(DHTIN, DHTOUT, DHTTYPE); //Definición de la variable del sensor de temperatura

//Variables de los modulos relé/relay
#define relay1 4
#define relay2 5


//Variables para los Archivos en la microSD
using namespace std;
string archivoDHT = "/usr/share/apache2/htdocs/verdatos.htm";          //Directorio donde se almacena los datos del sensor de Temperatura
string archivoSCTR1 = "/usr/share/apache2/htdocs/estadoAire1.php";     //Directorio donde se almacena los datos del sensor de corriente
string archivoSCTR2 = "/usr/share/apache2/htdocs/estadoAire2.php";     //Directorio donde se almacena los datos del sensor de corriente
string archivoSCTdatos = "/usr/share/apache2/htdocs/datosCorriente.php";//Directorio donde se almacena los datos del sensor de corriente
char valoresDHT[160];
char valoresSCT[160];
char comandoMYSQL[160];
char tem[15];
char hum[15];
char watt[15];
char amp[15];
```

```
//Variables Multitareas
unsigned long intervaloAnteriorTEMP = 0; //Reloj del intervalo
unsigned long intervaloAnteriorAIRE = 0;
unsigned long intervaloAnteriorCELULAR = 0;
//unsigned long intervaloTEMP = 1200000; //intervalo a usar en implementacion
//unsigned long intervaloAIRE = 300000;  //intervalo a usar en implementacion
unsigned long intervaloTEMP = 120000;    //intervalo para pruebas cortas
unsigned long intervaloAIRE = 30000;     //intervalo para pruebas cortas
unsigned long intervaloCELULAR = 10000;
int contadorApagado = 0;
int contadorEncendido = 0;


//Variables del modulo GSM
#define OnGSM 9                       //PIN de encendido del modulo GSM
//char nrocel[] = "+5493834330386";   //Numero de Celular para pruebas
byte pos = 0;                         //Posicion en el buffer
boolean validSender = false;          //Verificar si el remitente es valido
boolean confirmarRecibido = false;    //confirma que se recibieron las alertas
boolean controlAires = false;
char buffer[80];                      //Buffer que lee la cadena
char validCelAdmin[160];
char validCelAux1[160];
char validCelAux2[160];
String celularActualAdmin;
String celularNuevoAdmin;
String celularActualAux1;
String celularNuevoAux1;
String celularActualAux2;
String celularNuevoAux2;
String nroadmin;                        //Numero de Celular del administrador para recepción/envío de mensajes
String nroaux1;                         //Numero de Celular del auxiliar 1 para recepción/envío de mensajes
String nroaux2;                         //Numero de Celular del auxiliar 2 para recepción/envío de mensajes
String nrocel;
String nrocelarray[3];
String nroceladmin;
String nrocelaux1;
String nrocelaux2;
int categoria;                          //Categoria del numero de celular: 1 administrador, 2 auxiliar 1, 3 auxiliar 2.
int lastReceivedSMSId = 0;
IPAddress ip;
```


Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

```
//Variables estados del switch para leer SMS
enum _parseState {
  PS_DETECT_MSG_TYPE,                 //Estado basico, detecta mensajes

  PS_IGNORING_COMMAND_ECHO,           //ignora comandos que empiezan con "AT+..."

  PS_READ_CMTI_STORAGE_TYPE,          //detecta el tipo y el ID del mensaje
  PS_READ_CMTI_ID,                    //lee la coma para detectar el ID del mensaje

  PS_READ_CMGR_STATUS,
  PS_READ_CMGR_NUMBER,
  PS_READ_CMGR_SOMETHING,
  PS_READ_CMGR_DATE,
  PS_READ_CMGR_CONTENT
};

byte state = PS_DETECT_MSG_TYPE;      //Estado Inicial

//Variables del sensor de corriente SCT 013

EnergyMonitor energyMonitor;
float voltajeRed = 220.0;
boolean aireSecundario = false;


void resetBuffer() {
  memset(buffer, 0, sizeof(buffer));
  pos = 0;
}

```

```
//Configura las variables y funciones de primera instancia que seran usadas en el Loop
void setup()
{
  Serial1.begin(9600);
  Serial.begin(9600);

  //Configura la IP de la placa
  //system("ifconfig eth0 192.168.0.113 netmask 255.255.255.0");


  //Apaga el modulo GSM para inicie con el sistema en caso de reinicio fallido, espera 20 seg al modulo.
  Serial1.println("AT+CPOWD=0");
  delay(20000);

  // PIN utilizado por el sensor
  pinMode(DHTIN, OUTPUT);
  digitalWrite(DHTIN, HIGH);

  //Calibracion del sensor de Corriente (PIN de conexion del sensor, calibracion)
  energyMonitor.current(0, 0.34);

  //PIN utilizado por el modulo relé/relay
  pinMode(relay1, OUTPUT);
  pinMode(relay2, OUTPUT);

  //Variable que enciende el modulo GSM
  pinMode(OnGSM, OUTPUT);
  digitalWrite(OnGSM, HIGH);
  celularActualAdmin = leerCelular(1);
  celularActualAux1 = leerCelular(2);
  celularActualAux2 = leerCelular(3);

  delay(15000);                   //Espero 15 segundos a que encienda y se conecte el Modulo GSM
  Serial1.println("AT+CMGF=1");   //Coloca el Modulo en modo SMS, necesario para leer en ASCII los SMS recibidos

  for (int i = 1; i <= 15; i++) {
    Serial1.print("AT+CMGD=");
    Serial1.println(i);
    delay(200);
```

```
    // No es realmente necesario, pero puede prevenir que el Serial pierda algun dato.
    while (Serial1.available())
      Serial.write(Serial1.read());
  }
  primerMenu();
}

```


Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

Anexo 3: Código Fuente archivos .php de la Aplicación Web.

A continuación, se adjunta fragmentos del código de los distintos archivos PHP de la Aplicación Web.

❖ Index.php

```php
<?php

session_start();

require ("conexion.php");

if( (!empty($_POST['user']) && empty($_POST['pass'])) || (empty($_POST['user']) && !empty($_POST['pass'])) ){
    $mensaje="Usuario o Clave incorrectos";
    $error = true;
}
else{
    if(!empty($_POST['user']) && !empty($_POST['pass'])){


            $user=$_POST['user'];
            $pass=$_POST['pass'];

            $sql = "SELECT usuario,password FROM usuarios WHERE usuario='$user'";
            $records = mysqli_query($conexion,$sql);

            $resultados = mysqli_fetch_array($records);

            $mensaje = '';
            $error = false;

            if(count($resultados) >0 && password_verify($_POST['pass'], $resultados['password'])) {

                $_SESSION['usuario_id'] = $resultados['usuario'];
                header('Location: /principal.php');

            }
            else{
                $mensaje="Usuario o Clave incorrectos";
                $error = true;
            }


    }
}
?>
```


Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

❖ Principal.php

```
</head>

<body>

<div class="container">
  <div class="header"><img src="LogoUNCa.png" name="logo" border= 1 width="178" height="150" id="Insert_logo" style="background-color: #366090;
  display:inline-block;" /><img src="bannerweb.png" border=1 width="818" height="150"><!-- end .header --></div>

  <?php if(empty($datosuser)): ?>

 <div class="sidebar1">
  <ul class="nav">
      <li><a href="index.php">Iniciar Sesi&#243n</a></li>
    </ul>
    <p> </p>
    <!-- end .sidebar1 --></div>

<?php else: ?>
  <div class="sidebar1">
  <ul class="nav">
      <li><a href="principal.php">Gr&#225ficos</a></li>
      <li><a href="modificarpass.php">Modificar Usuario/Clave</a></li>
      <li><a href="configcelular.php">Cambiar n&#250mero de Celular</a></li>
      <li><a href="logout.php">Cerrar Sesi&#243n</a></li>
    </ul>
    <p> </p>
    <!-- end .sidebar1 --></div>
  <div id="content">
  <table width="760" align="center">
  <tr>
                <h2>Estado del Data Center:</h2>
  </tr>
      <tr>
      <td><div id="MedidorTemp"></div></td><td><div id="MedidorHum"></div></td><td><div id="tabla"></div></td>
  </tr>
<tr>
</tr>
  </table>
  <table width="780" align="left">
<tr>
        <h2>Datos Hist&#243ricos:</h2>
</tr>
<tr>
```

❖ Configcel.php

```
<body>

<div class="container">
  <div class="header"><img src="LogoUNCa.png" name="logo" border= 1 width="178" height="150" id="Insert_logo" style="background-color: #366090;
  display:inline-block;" /><img src="bannerweb.png" border=1 width="818" height="150"><!-- end .header --></div>

  <div class="sidebar1">
    <ul class="nav">
      <li><a href="principal.php">Gr&#225ficos</a></li>
      <li><a href="modificarpass.php">Modificar Usuario/Clave</a></li>
      <li><a href="configcelular.php">Cambiar n&#250mero de Celular</a></li>
      <li><a href="logout.php">Cerrar Sesi&#243n</a></li>
    </ul>
    <p> </p>
    <!-- end .sidebar1 --></div>
    <div id="content">
        <center><h2>N&#250meros de celular registrados:</h2><br></center>
      <?php

        if (isset($_POST['token'])){
         $error='';
         $mensaje = "";
         $exito = false;
         if($_POST['categoria']!= 0){
            if($_POST['categoria']== 1){

                $celular = $_POST['celular'];
                $nombre = $_POST['nombre'];

                if(!empty($_POST['celular'])){
                $abrircel = fopen("celularadmin.php",'r+');
                $contenidocel = fread($abrircel, filesize("celularadmin.php"));
                fclose($abrircel);

                $contenidocel = explode('+54',$contenidocel);
                $contenidocel[1] = $celular;
                $contenidocel = implode("+54",$contenidocel);

                $abrircel = fopen("celularadmin.php", 'w');
                fwrite($abrircel,$contenidocel);
                fclose($abrircel);
                $error = 0;
```


Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

❖ Modificarpass.php

```
<div class="container">
    <div class="header"><img src="LogoUNCa.png" name="logo" border= 1 width="178" height="150" id="Insert_logo" style="background-color: #366090;
    display:inline-block;" /><img src="bannerweb.png" border=1 width="818" height="150"><!-- end .header --></div>
    <div class="sidebar1">

<ul class="nav">

        <li><a href="principal.php">Gr&#225ficos</a></li>
        <li><a href="modificarpass.php">Modificar Usuario/Clave</a></li>
        <li><a href="configcelular.php">Cambiar n&#250mero de Celular</a></li>
        <li><a href="logout.php">Cerrar Sesi&#243n</a></li>

    </ul>
    <p> </p>
    <!-- end .sidebar1 --></div>
    <div id="content">

    <?php
    if(!empty($mensaje)){
        if($error){

            echo "<script type='text/javascript'>alert('$mensaje')</script>";
        }


        else{
            echo "<script type='text/javascript'>alert('$mensaje')</script>";
        }
    }
    ?>

    <center><h1>Modificar datos de Sesi&#243n</h1>
        <p style="color: #7b7b7b">NOTA: Si solo quiere modificar el nombre de usuario, deje en blanco el campo contraseña.</p>


    <form action="modificarpass.php" method="post">
        <input type="text" name="Usuario" placeholder="Ingrese el nuevo nombre de Usario">
        <input type="password" name="Pass" placeholder="Ingrese su nueva Clave">
        <input type="password" name="ConfirmaPass" placeholder="Confirme su Clave">
        <input type="submit" value="Aceptar">

```

❖ Conexión.php

```
<?php
    $conexion=mysqli_connect("127.0.0.1","root","root","temperatura_humedad") or die ("No se Pudo Conectar<br>Mensaje de Error:
    ".mysqli_connect_error());
?>
```


Alumno: Morti, Ricardo Maximiliano
M.U.: 0852

❖ estadoEquipo.php

```php
<?php

        $cont1 = file_get_contents("estadoAire1.php");
        $cont2 = file_get_contents("estadoAire2.php");
        $datos = file_get_contents("datosCorriente.php");


        echo "<h2>Consumo: <br></h2>";
        echo "<h2>";
        echo $datos;
        echo "</h2>";
?>
<html>
<body>
<table border="0" cellpadding="10">
    <tr>
        <td>
            <?php
            if($cont1 == 1 || $cont1 == 0){
              if($cont1 == 1){
                 echo "<b>Estado de Equipo 1</b><br>";
                 echo "<img src=\"imgON.jpg\" width=\"93\" height=\"60\" alt=\"imgON\"><br>";
              }
              if($cont1 == 0){
                 echo "<b>Estado de Equipo 1</b><br>";
                 echo "<img src=\"imgOFF.jpg\" width=\"93\" height=\"60\" alt=\"imgOFF\"><br>";
              }
           }
           else{
             echo "<h2>No se puede mostrar el Estado del equipo 1</h2>";
           }
            ?>
        </td>
        <td>
            <?php
             if($cont2 == 1 || $cont2 == 0){
               if($cont2 == 1){
                 echo "<b>Estado de Equipo 2</b><br>";
                 echo "<img src=\"imgON.jpg\" width=\"93\" height=\"60\" alt=\"imgON\">";
               }
             if($cont2 == 0){
                 echo "<b>Estado de Equipo 2</b><br>";
                 echo "<img src=\"imgOFF.jpg\" width=\"93\" height=\"60\" alt=\"imgOFF\">";
```

❖ datostemp.php

```php
<?php

require_once("conexion.php");

    $statement=$pdo->prepare("SELECT temperatura,humedad FROM temporal ORDER BY id DESC LIMIT 1");
            $statement->execute();
            $results=$statement->fetchAll(PDO::FETCH_ASSOC);
            $json=json_encode($results);
            echo $json;
?>
```


Alumno: Morti, Ricardo Maximiliano
M.U.: 0852