Universidad de las Ciencias Informáticas
Facultad 5

Título: Sistema de adquisición de variables eléctricas mediantes dispositivos PDA.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Carlos Felipe Pérez

Jorge Denis Riverón

Tutor: Raúl Pérez-Alejo Neyra.

Ciudad de la Habana, Cuba

Junio, 2009

DATOS DE CONTACTO

Nombre y apellidos: Raúl Pérez-Alejo Neyra

Especialidad de graduación: Ing. en Ciencias Informáticas

Categoría docente:

Categoría Científica: Ingeniero

Años de experiencia en el tema: - años

Años de graduado: 2

Correo electrónico: rperez-alejo@uci.cu

DEDICATORIA

A mi mamá y a mi papá por todo su cariño, por haberme guiado siempre en todos mis estudios y por ser a quién le debo todos mis resultados, sin su guía nunca hubiera sido quien soy en este momento.

A mi hermana que me ha apoyado toda mi vida en todos los momentos y siempre ha estado ahí para mí cuando la he necesitado.

A todas esas personas que a lo largo de mi vida me han querido y apoyado incondicionalmente.

Carlos Felipe.

A mi mamá, porque a ella debo todo lo que soy; a mi papá, quien aunque no esté, me aconseja y me apoya siempre.

A mis amigos y compañeros, y a todas aquellas personas que de una forma u otra me han ayudado a lo largo del camino.

Jorge Denis.

AGRADECIMIENTOS

Carlos

A mi mamá por ser siempre mi luz y por estar siempre pendiente a todo lo que me sucede.

A mi papá por su incansable paciencia en todo para conmigo.

A ambos por estar siempre pendientes en todos mis estudios y mis resultados.

A mi hermana por todo su cariño y apoyo en todos los momentos.

A mi tutor Raúl por toda su ayuda y paciencia con nosotros, por haber hecho posible el desarrollo satisfactorio de la tesis.

A mi compañero de tesis Denis por cargar conmigo durante este tiempo y por todo lo que aprendí a su lado.

A todas mis amistades que de una forma u otra me brindaron su ayuda y que también son parte de mis resultados.

A la Revolución por darme la posibilidad de estudiar en esta maravillosa escuela.

Muchísimas gracias a todos ustedes.

Jorge

A toda mi familia, por su apoyo incondicional y por siempre creer en mí, ellos son el combustible de mi vida.

A nuestro tutor Raúl, que hizo lo posible y lo imposible por que este trabajo saliera adelante, este resultado es tan suyo como nuestro.

A mi compañero de tesis Felipe, por la confianza que depositó en mí y por toda la valentía y dedicación con que enfrentó las tareas que se nos asignaban.

A todas las personas del polo de automática que nos ayudaron y nos aconsejaron, ha sido un verdadero honor trabajar con ustedes.

A nuestros profesores, por toda la dedicación y esfuerzo que ponen cada día en su trabajo.

A esta maravillosa universidad, por darme la oportunidad de ser un profesional de la informática.

Muchísimas gracias…

RESUMEN

Cuba atraviesa por un momento difícil en cuanto al ahorro energético consecuencia de malas políticas como consecuencia del período especial, una pobre cultura de ahorro por parte de la población y de otros factores, como son el mal estado de las líneas, la mala distribución de la red del sistema energético, entre otras. Debido a esta situación energética en que se encuentra el país la Unión Nacional Eléctrica (UNE) adquirió un número de dispositivos que recopilan una mayor cantidad de información acerca de la red eléctrica que supervisan (metro-contadores) y otros que ayudan a los técnicos eléctricos a almacenar y portar la información obtenida de los primeros, estos son los Asistentes Digitales Personales (PDA). Pero todavía persiste un problema en la lectura de los parámetros eléctricos la cual la realiza el operador de forma manual, por lo que se quiere automatizar el proceso de forma íntegra para ganar en rapidez, fiabilidad y eficiencia, y además lograr un proceso libre de errores y sin pérdida de información. Como resultado de esta investigación se pretende crear una aplicación capaz de realizar la comunicación entre los metro-contadores multifuncionales CIRWATT y un dispositivo PDA, realice la lectura de los parámetros eléctricos y los almacene, y de esta forma ahorrar dinero al país, que no tendría que mandar a fabricar un software parecido a este a empresas extranjeras.

PALABRAS CLAVE

PDA, metro-contador, protocolos de comunicación, dispositivos móviles, PocketPC, variables eléctricas, ModBus RTU**.**

TABLA DE CONTENIDOS

TABLA DE CONTENIDOS

INTRODUCCIÓN

Hoy en día presenciamos una increíble evolución en el campo de las tecnologías de la información y las comunicaciones (TIC). Actualmente las TIC encuentran su aplicación en casi todas las esferas de la sociedad y ramas de la ciencia. En la industria, particularmente, vemos a las TIC jugar un papel fundamental en la automatización de procesos, actividad que influye directamente en la eficiencia de las empresas y que se ha convertido, por lo tanto, en una necesidad fundamental de la industria moderna. La supervisión energética, siendo una rama más de la industria, no se ve exenta de esta tendencia. Así, nuevos y más modernos metro-contadores han surgido, capaces de comunicarse con otros dispositivos digitales automáticamente, transmitir valores y recibir órdenes de los mismos.

Cuba ha atravesado por una crisis energética durante los últimos años, consecuencia de malas políticas, del periodo especial, además de una pobre cultura de ahorro por parte de la población. Para dar solución a esta crisis el estado cubano ha puesto en marcha un gran plan para el ahorro de energía eléctrica, que conocemos como Revolución Energética. Como parte de dicha estrategia la Unión Nacional Eléctrica (UNE) ha implementado un sistema de supervisión para los grandes consumidores del país específicamente. Dicho proceso contaba con deficiencias: La lectura de los metro-contadores se realizaba de forma manual, lo que puede conducir a la introducción de errores humanos. El almacenamiento y traslado de dicha información se realizaba en papel, por lo que su manejo era muy engorroso y daba lugar a imprecisiones y pérdida de información.

Para corregir esto la UNE adquirió una serie de dispositivos que automatizaron en gran medida dicho proceso, ellos fueron los metro-contadores multifuncionales, que recopilan una mayor cantidad de información acerca de la red que supervisan de forma automática, y dispositivos móviles o Asistentes Digitales Personales (PDA por sus siglas en inglés) de tipo industrial para ayudar al especialista a almacenar y portar la información que brindan los primeros. Pero todavía persiste el problema de la no automatización del proceso de captación de la información.

Es por todo ello que la Universidad de Ciencias Informáticas (UCI), como parte de su sistema educacional y de práctica profesional se ha dado a la tarea de construir una aplicación para uso nacional, que realice las actividades de recopilación y almacenamiento de la información del metro-contador multifuncional

CIRWATT (metro-contador que actualmente está en explotación en nuestro país) soportable sobre dispositivos portátiles o PDAs.

La anterior **Situación Problémica** nos lleva al planteamiento del siguiente **Problema Científico**:

¿Cómo implementar un sistema que permita acceder a los metro-contadores multifuncionales instalados por la UNE desde un dispositivo móvil para adquirir y almacenar los datos de las lecturas que registran?

Como resultado de esta investigación se pretende crear una aplicación capaz de realizar la comunicación entre el metro-contador y el dispositivo PDA y de esta forma ahorrar dinero al país, que no tendría que adquirir una aplicación similar implementada por terceros. Dicha aplicación estaría destinada al uso nacional solamente, y no tendría posibilidades de comercialización en el exterior, debido a las trabas existentes en el mercado mundial con respecto a Cuba, ya que somos un país bloqueado y no podemos adquirir las licencias de los programas utilizados para el desarrollo de la aplicación; pero como ya mencionamos, si representaría un aporte a la economía del país.

Constituye como **Objeto de estudio** de este tema las aplicaciones recolectoras de información siendo el **Campo de acción** Aplicaciones recolectoras de información en dispositivos móviles.

El **Objetivo General** de este trabajo es implementar una aplicación que se ejecutará en un dispositivo móvil o PDA con fines específicos para la adquisición de distintos tipos de variables eléctricas en metro-contadores para grandes consumidores de energía.

Para lograr el total cumplimiento del objetivo se trazan las siguientes **tareas investigativas**:

- Realizar el proceso de captura de requisitos de la aplicación a desarrollar.
- Realizar un estudio sobre los protocolos de comunicación de los metro-contadores.
- Realizar el proceso de Diseño de la Aplicación a desarrollar.
- Implementar las clases y funcionalidades que den solución al problema.
- Realizar pruebas de caja negra para validar las funcionalidades de la aplicación.
- Documentar los procesos desarrollados para dar solución a la problemática planteada.

Métodos teóricos

-Analítico – sintético: Se utilizó este método con el objetivo de extraer la esencia de la bibliografía relacionada con el objeto de estudio, definiendo los rasgos y características que lo identifican.

-Modelación: Este método, para un mayor entendimiento del trabajo, permitirá la visualización de propuestas, alternativas y estrategias, que se utilizará en el desarrollo del trabajo.

Métodos empíricos:

-Observación: Este método servirá para lograr un mayor realismo en el trabajo basándose en la captación visual del diseño, reglas y estructura del PDA.

-Entrevista: Permitirá la obtención del conocimiento necesario para saber sobre el tema y continuar con el trabajo.

CAPÍTULO I. FUNDAMENTACIÓN TEÓRICA

Este capítulo realiza una profunda investigación sobre los procesos de negocios asociados a las PDAs en el mundo y en Cuba. Analiza el desarrollo de las PDAs enfocado a la realización del sistema. También fundamenta la utilización de técnicas, herramientas y lenguajes que apoyan el desarrollo del sistema. La investigación realizada, contribuye en gran medida al desarrollo de la solución que se propone en el trabajo.

CAPÍTULO II. CARACTERÍSTICAS DEL SISTEMA

Este capítulo realiza una caracterización del sistema; identificándose los trabajadores involucrados y artefactos dentro de los procesos del negocio. Brinda visión general del funcionamiento de los procesos del negocio. Además define los requerimientos funcionales, define los casos de uso necesarios para la realización de dicha aplicación.

CAPÍTULO III. DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN

Este capítulo realiza la solución general de la aplicación. Analiza los casos de usos críticos, definiendo las clases del análisis para cada uno, así como la organización de los módulos mediante un diagrama de

paquetes. En el diseño esboza todas las clases para cada uno de los objetos determinados y propone un modelo de datos que sustente las clases desarrolladas.

CAPÍTULO IV. PRUEBAS DEL SISTEMA.

Este capítulo lleva a cabo el proceso de pruebas del software para la detección y corrección de los errores que pueda presentar la aplicación con el objetivo de darles solución y respuesta a los mismos. El objetivo específico de estas pruebas es encontrar cuantos más errores, mejor. Finalmente muestra las conclusiones del trabajo, las recomendaciones propuestas, la bibliografía, los anexos y el glosario de términos.

CAPÍTULO I. FUNDAMENTACIÓN TEÓRICA

INTRODUCCIÓN

No se puede comenzar a plantear una solución para el problema en cuestión sin que se haya ganado antes un claro entendimiento del dominio del mismo, la terminología básica utilizada, así como todo un conjunto de características y situaciones que afecten o influyan en la realización de la misma. Para dar cumplimiento a este propósito se ha dedicado este primer Capítulo. Se hace una necesidad conocer el estado del problema en el mundo: ¿qué hay hecho? ¿En qué se está trabajando? ¿Qué se puede hacer? Todos estos factores tienen un gran impacto en el desenvolvimiento de la investigación, ya que en el proceso de desarrollo se debe evitar a toda costa “volver a inventar la rueda”.

El proceso que se pretende automatizar consta de dos componentes físicos fundamentales. El primero sería el dispositivo que provee los datos (metro-contador multifuncional CIRWATT) y el segundo, el dispositivo que captaría y almacenaría dichos datos de manera automática (el PDA). Es por ello que antes de iniciar un diseño del sistema capaz de realizar dicho proceso, es de gran utilidad la realización de un estudio sobre estos componentes, para adquirir un mayor entendimiento del proceso y familiarizarse con las tecnologías involucradas en el mismo.
Primeramente, y para seguir un orden lógico en los eventos, trataremos sobre el equipo origen del flujo de datos, o sea, el metro-contador.

1.1 Dispositivo metro-contador

Circutor serie CIRWATT

CIRWATT es un contador trifásico digital multifuncional de cuatro fases, cuya principal característica es su gran versatilidad, ello le permite adecuarse a las necesidades de cada usuario.
Está diseñado especialmente para instalaciones donde los contadores electromecánicos no satisfacen las necesidades actuales, concretamente en aquellas en las que se precise un contador con sistema tarifario o donde la facturación de la energía eléctrica se haga mediante perfiles de carga.
El contador dispone de un puerto serie RS-232 ó RS-485 (según modelo), y un puerto óptico R1, para la comunicación local o remota, lectura y programación.

La presentación de datos se realiza mediante una pantalla de cristal líquido (LCD por sus siglas en inglés: *Liquid Crystal Display*) de una línea especialmente diseñada para el CIRWATT.
Las lecturas se almacenan en unidades kW. El tamaño de los registros es de 8 dígitos enteros (99.999.999). El Cirwatt cumple las normas existentes aplicables a contadores electrónicos, y dispone de un sistema autónomo de retención de datos que evita su pérdida frente a la ausencia de tensión de alimentación. [1]
Este circutor lee alrededor de 405 variables entre las ke estan: las variables de energia tanto activa como reactiva , tensiones y corrientes del primario, valores de frecuencia, potencia, maxima demanda actual, maxima demanda total, potencia contratada, exceso de potencia, entre otros parametros que lee este circuitor.
La familia **CIRWATT**, ofrece una completa gama de contadores de energía eléctrica adaptada al tipo de aplicación a la que va destinado.

CIRWATT D
Alta precisión, medida en 4 cuadrantes y la máxima flexibilidad en programación y comunicaciones, para los requisitos más exigentes. La mejor solución para grandes consumidores: generación, subestaciones y gran industria.

CIRWATT C
Compromiso entre prestaciones y coste, sin renunciar a la máxima calidad. Disponible en 2 ó 4 cuadrantes.
Ideal, tanto para aplicaciones industriales como para servicios.

CIRWATT A
Contadores de uso doméstico de última generación. Lo último en medida, grandes prestaciones y sencillez de lectura.

Ver Anexo (Tabla29: tipo de institución donde se usa cada uno de estos tipos de CIRWATT).

Como ya se mencionó estos dispositivos para la comunicación con otros dispositivos cuentan con varias interfaces físicas, de ellas la que se pretende utilizar es la RS-232. A continuación algunas especificaciones de la misma.

1.1.1 Interfaz de comunicación
RS-232.

El RS-232 (también conocido como Electronic Industries Alliance RS-232C) es una interfaz que designa una norma para el intercambio serie de datos binarios entre un DTE (Equipo terminal de datos) y un DCE (Data Communication Equipment, Equipo de Comunicación de datos).

La interfaz RS-232 está diseñada para distancias cortas, de unos 15 metros o menos, y para velocidades de comunicación bajas, de no más de 20 [Kb/s]. A pesar de ello, muchas veces se utiliza a mayores velocidades con un resultado aceptable. [2]

Además de esta interfaz, la comunicación entre dos dispositivos se establece utilizando un protocolo de comunicación, los cuales son reglas de comunicación que permiten el flujo de información entre distintos dispositivos que manejan o no lenguajes distintos, por ejemplo, dos computadores conectados en la misma red pero con protocolos diferentes no podrían comunicarse jamás, para ello, es necesario que ambas hablen el mismo idioma. Los protocolos de comunicación se dividen en varios niveles: **Nivel de transporte, Nivel de interred**, **Direcciones IP,** entre otros. [3]

Para el caso del metro-contador CIRWATT el protocolo establecido por sus fabricantes es el que especificaremos a continuación.

1.1.2 Protocolo de comunicación

ModBus

Desarrollado por Modicon para comunicación entre PLC's.
Debido a su simplicidad y especificación abierta, actualmente es ampliamente utilizado por diferentes fabricantes.
Entre los dispositivos que lo utilizan podemos mencionar: PLC, RTU, sensores y actuadores inteligentes.
El protocolo establece cómo los mensajes se intercambian y la detección de errores.

Características:

- Control de acceso al medio tipo Maestro/Esclavo.
- El protocolo especifica: formato de trama, secuencias y control de errores.
- Existen dos variantes en la forma de transmisión de los datos: ASCII y RTU, esta última transmite los datos en formato binario.
- Sólo especifica la capa de enlace del modelo ISO/OSI.
- A cada esclavo se le asigna una dirección fija y única en el rango de 1 a 247.
- La dirección 0 está reservada para mensajes de difusión sin respuesta.

Detección de errores.

A excepción de los mensajes de difusión, cuando un dispositivo maestro envía una solicitud a un esclavo, espera una respuesta normal. Uno de estos cuatro posibles eventos puede ocurrir a partir de una solicitud del maestro:

- Si el dispositivo esclavo recibe la solicitud sin errores de comunicación, y la puede procesar, retorna una respuesta normal.
- Si el esclavo no recibe la solicitud debido a un error de comunicación, no se emite ninguna respuesta. Queda bajo la responsabilidad del programa maestro procesar una condición de tiempo de espera máximo para la solicitud.
- Si el esclavo recibe la solicitud, pero detecta un error de comunicación (paridad, LRC o CRC), no retorna ninguna respuesta. Queda bajo la responsabilidad del programa maestro procesar una condición de tiempo de espera máximo para la solicitud.
- Si el esclavo recibe la solicitud sin error de comunicación, pero no la puede procesar (por ejemplo, si la solicitud es leer una bobina o registro inexistente), el esclavo enviará una respuesta de error, informando al maestro la naturaleza del mismo.

La respuesta de error tiene dos campos que la diferencian de una respuesta normal:

Campo de Código de Función: En una respuesta normal, el esclavo envía un eco del código de función de la solicitud original en el campo de código de función de la respuesta. Todos los códigos de función tienen el bit más significativo (BMS) en 0 (todos sus valores están por debajo de 80 hexadecimal). En una

respuesta de error el esclavo pone el BMS a 1. Esto hace al código de función en una respuesta de error exactamente 80 hexadecimal más alto que el valor para una respuesta normal.
Con el BMS del código de función alterado, el programa maestro puede reconocer la respuesta de error de la normal y puede examinar entonces el campo de datos de la misma en busca de la causa.

Campo de Datos: En una respuesta normal, el esclavo puede retornar datos o estadísticas en el campo de datos (cualquier información pedida en la solicitud). En una respuesta de error, el esclavo devuelve un código de excepción en el campo de datos. Esto define la condición del esclavo que causo la excepción. [4]

Código	Tipo de Error	Significado
01	Función ilegal	La función recibida no está permitida en el esclavo.
02	Dirección ilegal	La dirección está fuera del rango permitido.
03	Dato ilegal	El dato contiene un valor no válido.
04	Falla en el dispositivo	El controlador no responde o ha ocurrido un error.
05	Reconocimiento (ACK)	Se ha aceptado la función y se está procesando.
06	Ocupado	El mensaje ha sido recibido sin error, pero el dispositivo no puede procesarlo en este momento.
07	Reconocimiento Negativo (NAK)	La función solicitada no puede realizarse en este momento.

Tabla 1. Códigos de Error.

De las dos variantes del formato de la transmisión mencionadas, se utilizará la RTU, ya que es la que utiliza el dispositivo metro-contador en cuestión, además la interfaz física que se utilizará solo soporta la transmisión de datos binarios, por lo que la utilización de la variante ASCII esta fuera de contexto.
Ver Anexo (Tabla28: formato de la trama ModBus RTU)

Hasta aquí hemos cumplimentado la primera parte, que era investigar sobre el equipo transmisor de datos en el proceso, ahora quedaría analizar la otra parte involucrada en el mismo, o sea, el dispositivo receptor de los datos.

1.2 Asistente Personal Digital (PDA)

1.2.1 Breve Historia

En 1989, (con 19901246 pajillas) el Atari Portfolio, aunque técnicamente clasificado como palmtop (Ordenador de pequeño tamaño, algo mayor que un paquete de cigarrillos, que se puede llevar en la palma de la mano (palm) y que, además de otras funciones, permite la conexión con Internet y generalmente se maneja desde pantalla táctil.) fue una muestra temprana de algunos de los más modernos dispositivos electrónicos. Le siguieron otros dispositivos como los Psion Organiser, el Sharp Wizard o la Amstrad Penpad que fueron sentando la base de las funcionalidades de las PDAs.
La primera mención formal del término y concepto de PDA (*Personal Digital Assistant*) es del 7 de enero de 1992 por John Scullev al presentar el Apple Newton, en el Consumer Electronics Show (Muestra de electrónica de consumo) de Las Vegas (EE.UU.). Sin embargo fue un sonoro fracaso financiero para la compañía Apple, dejando de venderse en 1998. La tecnología estaba aún poco desarrollada y el reconocimiento de escritura en la versión original era bastante impreciso, entre otros problemas. Aún así, este aparato ya contaba con todas las características del PDA moderno: pantalla sensible al tacto, conexión a una computadora para sincronización, interfaz de usuario especialmente diseñada para el tipo de máquina, conectividad a redes vía módem y reconocimiento de escritura. [5]
En 1995 con la aparición de la empresa Palm comenzó una nueva etapa de crecimiento y desarrollo tecnológico para el mercado de estos dispositivos. Tal fue el éxito que las PDA son a veces llamadas *Palm* o *Palm Pilot*, lo cual constituye un caso de una marca registrada que se transforma en el nombre genérico del producto.
La irrupción de Microsoft Windows CE (2000) y Windows Mobile (2003) en el sector los dotó de mayores capacidades multimedia y conectividad, y sobre todo incorporó a un público ya acostumbrado al uso de sus programas y que se los encontraban en versión reducida.
La irrupción de los Smartphone o Comunicadores (híbridos entre PDA y teléfono móvil) trajeron por un lado nuevos competidores al mercado y por otro incorporaron al usuario avanzado de móviles al mercado. De paso supuso la vuelta de un sistema operativo que había abandonado el mercado de las PDAs y ordenadores de mano en favor de los móviles: el Symbian OS. Las PDAs de hoy en día traen multitud de comunicaciones inalámbricas (Bluetooth, WiFi, IrDA, GPS...) que los hace tremendamente atractivos hasta para cosas tan inverosímiles como su uso para domótica (se refiere a la automatización y control (encendido / apagado, apertura / cierre y regulación) de aparatos y sistemas de instalaciones eléctricas y

electrotécnicos (iluminación, climatización, persianas y toldos, puertas y ventanas motorizados, el riego, etc.) de forma centralizada y/o remota. El objetivo del uso de la domótica es el aumento del confort, el ahorro energético y la mejora de la seguridad personal y patrimonial en la vivienda. [6]

Los PDA han ampliado su uso a muchas esferas y en diferentes formas se hace uso de su capacidad de recolección de datos.

Se usa para la recolección de datos en trabajo de campo, reemplazando a los cuestionarios de papel. Las PDA se usan en las encuestas. En cuanto a la captura de datos, reduce a cero los datos faltantes en una entrevista, dado que el flujo de datos es monitoreado por el programa de captura.

Los PDA, cumplen con las características básicas necesarias para ser usadas durante la entrevista: memoria superior a los 60 MB, velocidad del procesador mayor a los 600 MHZ, entre otras.

Otro factor importante a tener en cuenta es la no impresión de documentos para la realización de la entrevista (cuestionarios).

Se utiliza el correo electrónico, como medio de transmisión de datos, esto significa el ahorro en el gasto de flete de envío de los cuestionarios.

Las aplicaciones que utilizan PDA como herramienta de recolección de datos en campo es una de las áreas de mayor desarrollo en los últimos tiempos.

1.2.2 Características Generales

Arquitectura de las PDAs

Las PDAs usan una arquitectura de microprocesador llamada ARM (Advanced RISC Machine), estos son procesadores integrados RISC de 32 bits (Reduced Instruction Set Computer diseñados por la empresa Acorn Computers y desarrollados por Advanced RISC Machines Ltd., una empresa derivada de la anterior) este tipo de procesador, ejecuta menos instrucciones y más simple, y consumen muy poca energía, lo cual lo hace ideal para dispositivos móviles, como las PDAs. [7]

Sistemas Operativos

Los PDA cuentan en la actualidad con sistemas operativos (SO) adaptados a la resolución de su pantalla cuyas características corresponden con las del dispositivo.

El sistema operativo contiene instrucciones pre-programadas que le dicen al procesador qué hacer. Los sistemas operativos usados por las PDA no son tan complejos como los usados por las PCs. Tienen menos instrucciones que requieren menos memoria. [7]
Existen diversos tipos de sistemas operativos para los PDA, normalmente diferentes para cada tipo de PDA y para cada fabricante, de la misma manera que existen equipos Mac y PC [8]. Los sistemas operativos principales son:

- **PalmOS**, comercializado por *Palm*.

El sistema operativo Palm fue desarrollado originalmente por Jeff Hawkins para el Pilot PDA de US Robotics. La versión 1.0 se vendía con los primeros Pilot 1000 y 5000 y la versión 2.0 se introducía con el Palm Pilot (**Pilot** fue el nombre dado a la primera generación de PDAs) Personal y Profesional.
Cuando salieron los Palm de la serie III se introdujo la versión 3.0 del Sistema operativo. Posteriormente, salieron las versiones 3.1, 3.3 y 3.5, que añadían apoyo para color, puertos de expansión múltiples, nuevos procesadores y otras prestaciones.
La versión 5 (Garnet) fue la primera versión que soportó los dispositivos ARM.
PalmSource, Inc. presentó Palm OS Cobalt (también denominado Palm OS 6) a los licenciatarios el 29 de diciembre de 2003. Esto completaría la migración a aparatos con ARM, y permitiría apoyar a las aplicaciones nativas ARM junto con apoyo multimedia mejorado. Actualmente NO existen equipos que usen el Palm OS 6 o Cobalt. No está muy claro el futuro de esta versión de Palm OS, derivado de la compra de PalmSource por la compañía japonesa ACCESS, LTD.
Aparentemente, en algún momento será posible tener nuevos equipos PDA con Palm OS cuyo núcleo (Kernel) sea un Linux completamente funcional. [6]

- **Windows Mobile** o **Pocket PC** (anteriormente *Windows CE*), comercializado por *Microsoft*.

Windows Mobile es un sistema operativo compacto, con una suite de aplicaciones básicas para dispositivos móviles basados en la API (Interfaz de programación de aplicaciones) Win32 de Microsoft. Los dispositivos que llevan Windows Mobile son Pocket PC (es un ordenador de bolsillo, también llamado PDA), Smartphone (*Teléfono inteligente* en español) es un dispositivo electrónico que funciona en un teléfono celular con características similares a las de un computador personal) y Media Center portátil (***PMC** por sus siglas en inglés*) es un disco duro con capacidad de reproducir archivos de audio y video.

Un PMC también puede ser usado para ver imágenes JPEG). Ha sido diseñado para ser similar a las versiones de escritorio de Windows. [9]

- **Windows CE** es el sistema operativo de Microsoft incrustado modular de tiempo real para dispositivos móviles de 32-bits inteligentes y conectados.

Para los fabricantes de equipo (OEMs) Windows CE 5.0 incluye una OAL (OEM Adaptation Layer, es decir, la interfaz entre Windows CE y el aparato físico) con "calidad de producción". Esto significa que se ha mejorado la manipulación de componentes mediante bibliotecas de código, estructuras de directorios para reutilizar código, archivos centralizados de configuración, y una arquitectura consistente a través de distintas familias de procesadores.

Para el usuario final, Windows CE 5.0 cuenta con las siguientes características y funcionalidad:

- ✓Soporte extendido de Bluetooth para audífonos y manos libres, soporte the PAN (Personal Aerea Network) y Bluetooth Secure Digital I/O (SDIO).
- ✓Capacidad de descubrimiento de dominios.
- ✓IPSec v4.
- ✓Soporte nativo para Wi-Fi.
- ✓Conexiones compartidas de Internet.
- ✓Captura de paquetes NDIS.
- ✓Remote Configuration Framework, que provee la estructura para crear interfaces web para aparatos sin pantalla (headless devices).
- ✓IPv6.
- ✓Herramienta de configuración para USB Flash
- ✓Controles de seguridad para padres de familia.
- ✓Proxy para web.
- ✓Peer-to-peer networking (PNRP).
- ✓Plug and play universal (UPnP).
- ✓Windows Media 9.
- ✓Múltiples idiomas de entrada.
- ✓Internet Explorer 6 para Windows CE.

[10]

- **Moblin SO**

Moblin será lo nuevo en sistema operativo, tanto para dispositivos móviles, MIDs y netbooks.

Moblin llegará para quedarse, ya que este nuevo proyecto es a código abierto y de distribución Linux con el soporte de la Linux Foundation enfocado al desarrollo de software orientado como dijimos a dispositivos móviles para conectividad a Internet (MIDs) y nuevas clases de dispositivos como netbooks.

Moblin cuenta con una interfaz propia más estilizada y simplificada, buscando aprovechar al máximo el espacio del Escritorio. De esta manera Moblin, por ejemplo su menú se encuentra oculto hasta que colocas el ratón en la parte superior de la pantalla, desde donde podrás acceder a mensajería instantánea, navegador web, conexión a Internet y toda la multimedia.

Hace poco ya contamos con la nueva versión de moblin, siendo esta la **Moblin 2.0** que integra funcionalidades sociales, ofreciendo acceso a sitios como Facebook, además de un navegador web basado en Mozilla y reproducción multimedia, todo ello desde un interfaz unificado y, porque no decirlo, muy lindo y mucho más manejable. [11]

- **Android SO**

Android, es un Sistema Operativo basado en Linux que cualquier móvil podrá utilizar, que será libre y fomentará el desarrollo de aplicaciones hechas por los usuarios/desarrolladores externos. Si nos fijamos, es exactamente la apuesta contraria que hizo Apple, con un SO para un solo terminal, no libre y con aplicaciones cerradas: por esto muchos ya le llaman el AntiPhone.

En realidad Google no actúa sola, ha promovido una alianza entre las empresas del género, tanto fabricantes como proveedores, llamada Open Handset Alliance y que servirá para promover este SO libre. Android apareció en el mercado a finales de 2008. [12]

- **Symbian SO**

Este es el sistema operativo para móviles más extendido entre smartphones, y por tanto el que más aplicaciones para su sistema tiene desarrolladas. Actualmente Symbian copa más del 65% del mercado de sistemas operativos.

Symbian dispone de varios interfaces, cosa que sería mejor unificar y simplificar. La última versión de Symbian es la 9.5, aunque aún no existen móviles con esta versión, siendo la 9.4 la última disponible en teléfonos como el próximo Nokia 5800 XpressMedia, con pantalla táctil y que es la apuesta de Nokia para competir con el iPhone. Además desde la versión 9.1 es totalmente necesaria que las aplicaciones sean firmadas para poder usar algunas funciones del teléfono, aunque ya existen formas de saltarse esa limitación.

Recientemente Nokia anunció que se hacía con el control total de la compañía Symbian, de la que contaba hasta el momento con el 48% de las acciones. A la vez que hacía pública su intención de liberar el sistema operativo como Software Libre en un intento de competir con futuros sistemas libres como Android de Google. [13]

La productora de este SO a empezado con la fabricación de un nuevo SO que se piensa que es el que va a reemplazarlo… este SO es Maemo.

- **Maemo SO**

Maemo Harmattan será, según la información desvelada por MobileCrunch, la primera versión del sistema operativo creado por Nokia para sus tabletas de Internet que gestionará un teléfono móvil. Su llegada al mercado, estimada entre finales de 2010 y el primer trimestre de 2011, también podría significar el principio del fin para Symbian S60 como sistema estrella de los smartphones de origen finlandés.

De cara a adaptar el sistema operativo basado en Linux a su nuevo hogar, – móviles con pantalla táctil – Nokia estaría trabajando no sólo en el replanteamiento estético que salta a la vista, sino también en modificar las entrañas de Maemo para mejorar su rendimiento, eficiencia y usabilidad.

La pantalla de inicio de este Maemo para móviles se caracterizaría por sus grandes dimensiones, pues solamente la zona con fondo azul sería visible sin necesidad de hacer scroll vertical, y por estar salpicada de pequeñas aplicaciones (widgets) a las que le gusta trabajar en equipo. [14]

Conexiones Inalámbricas

Las PDA tienen conexión inalámbrica de tres tipos Bluetooth, Wi-Fi, e IrDA.

Bluetooth: es un estándar de protocolo de radio y comunicación diseñado para consumir poca energía, con corto alcance, y basado en micro chips barato. Como el bluetooth usa un sistema de comunicación de radio, no necesita estar un dispositivo en frente del otro con que estén en el área de alcance uno de otro, es suficiente, incluso pueden estar en habitaciones diferentes si la señal es lo suficientemente fuerte. La configuración de estos dispositivos es muy simple, a diferencia del Wi-Fi, por lo tanto de más fácil acceso para la población que no está habituada a la tecnología. [7]

Wi-Fi: (Wireless Fidelity) es una "red de área local inalámbrica" (WLAN), permite conectarse a Internet en dispositivos móviles, siempre y cuando este cerca de un punto de acceso, o Hotspot, también permite la conexión entre dos dispositivos con Wi-Fi. Esta tecnología es más potente que el Bluetooth, y tiene más alcance, aunque consume mucha más batería, y necesita ser configurado, lo que representa una dificultad para la gente que no está habituada a relacionarse con la tecnología. [7]

IrDA (Infrared Data Assocation): este es otro protocolo para intercambio de datos, esta vez, a través de luz infrarroja, entre dos dispositivos, se utiliza en redes de área personal. Esta tecnología es de muy corto alcance (1metro aprox.), lo que permite ser usados en lugares que se requiere tener seguridad, consume muy poca energía, los sensores deben estar alineados para que se transfiera la información. Antes se utilizaba en PDAs y celulares, pero ahora se está dejando de usar para dar paso al Wi-Fi, y Bluetooth. [7]

Interacción

Cuentan con una tecnología llamada Touch Screen (pantalla táctil) para que los usuarios interactúen con sus aplicaciones, teniendo solo unos pocos botones reservados como atajos para los programas más utilizados. Las PDAs con esta tecnología suelen tener un Stylus (Puntero). La interacción con el dispositivo se crea al presionar la pantalla para activar botones o elegir menús o deslizar el Stylus para por

ejemplo subrayar menús. Hay muchos tipos de Touch Screen, aunque todos los tipos tienen algo en común, se manejan por coordenadas cartesianas, X e Y. El ingreso del texto está hecho generalmente de alguna de estas dos maneras:

- Usando un teclado virtual, donde el teclado esta mostrado en la pantalla y uno elije las letras apretándolas.
- Usando reconocimiento de palabras o letras, donde las letras son escritas por uno mismo en la PDA, y luego son "traducidas" a la letra programada en la PDA. [7]

Memoria

Las PDAs usan una memoria llamada memoria flash, estas son las sucesoras de las EEPROM, posibles de borrar y grabar electrónicamente. Las ventajas de estas es que permite que múltiples posiciones de memoria sean escritas o borradas en una misma operación de programación, frente a las EEPROM anteriores que sólo permite escribir o borrar una única celda cada vez. Por ello, flash permite funcionar a velocidades muy superiores cuando los sistemas emplean lectura y escritura en diferentes puntos de esta memoria al mismo tiempo. [7]

Características generales de memoria flash.

- No es volátil.
- Su volumen de almacenamiento varía de 512MB hasta 16 GB.
- Su velocidad de transferencia va desde 7 hasta 30 MB/s.
- Son más económicas en comparación con otras memorias de este volumen de almacenamiento, como las EEPROM.
- Tienen un diseño que le permite soportar golpes.
- Su funcionalidad es silenciosa ya que no posee sistemas mecánicos.
- Su pequeño tamaño es determinante a la hora de elegir un dispositivo portátil.
- Tienen una cantidad limitada de escritura, normalmente es 10.000 veces.
- Su información está guardada en forma encriptada.
- Resiste temperaturas de -25°C a 85°C.

[7]

1.2.3 Desarrollo de Aplicaciones para PDA.

Con la creciente popularidad de los dispositivos informáticos móviles, entre los que se encuentran los Asistentes Digitales Personales (PDA) y los teléfonos móviles, existe una demanda cada vez mayor de desarrolladores que puedan diseñar aplicaciones que puedan ejecutarse en varios dispositivos.
Pero debido a las peculiares características de estos dispositivos y los sistemas operativos que soportan, es necesario tener en cuenta una serie de consideraciones a la hora de diseñar e implementar aplicaciones informáticas para los mismos.

Interfaz:

Las PDAs se caracterizan por tener un tamaño muy reducido de pantalla, por lo que las dimensiones del interfaz de usuario no deberán exceder de 250 x 275 píxeles. Esta es una de las características más importantes a la hora de realizar interfaces de usuario para PDA.
La limitación en el tamaño de la pantalla hace que haya que colocar los componentes de las interfaces de la forma adecuada y prescindiendo de los componentes que no sean útiles, para proporcionar la usabilidad y la accesibilidad requeridas.
Así pues en las aplicaciones habrá que prescindir de las imágenes o el texto no significativo o reducirlas y recolocarlas en la posición adecuada.
En una PDA hay varias formas de realizar la escritura de caracteres, pero ninguna de ellas es tan rápida como la que se puede realizar con un teclado normal, así pues este debe ser otro detalle a tener en cuenta en el diseño de interfaces para PDA y es el hecho de que en la medida de lo posible no se necesite escribir mucho texto. Y si es posible es más cómodo puntear sobre la opción deseada en una lista desplegable que escribir el texto en la pantalla. [15]

Sincronización:

Ya que las PDA están diseñadas para complementarse con la PC, necesitan trabajar con la misma información en ambos lugares. Por ejemplo, si anotas algo en tu computadora, vas a necesitar transferirlo a tu PDA, y viceversa.
El software de sincronización de la PDA trabaja en conjunto con software que se instala en la PC. Los dispositivos Microsoft Pocket PC usan Active Sync y los dispositivos Palm OS usan HotSync.
Una de las mejores cosas de la sincronización es que siempre tienes un “back-up” de la información, por si la PDA se rompe, es robada, o se queda absolutamente sin energía.

El proceso de sincronización tiene dos pasos importantes:

- Los usuarios móviles envían los cambios realizados en las copias locales de los datos fuente.
- Los usuarios reciben los cambios realizados en los datos fuente que residen en el servidor corporativo desde la última sincronización.
  [7]

1.2.4 Uso de los dispositivos móviles

Los asistentes personales ofrecen versiones "livianas" de herramientas de ofimáticas como por ejemplo editores de texto, hojas de cálculo, calculadora y visores para una amplia variedad de formatos de archivo (archivos PDF, imágenes, etc.).

Además de estas funciones básicas, cada vez más PDA brindan herramientas multimedia de avanzada que permiten la reproducción de videos (en distintos formatos, que incluyen el formato Divx), música (en formato mp3) y animación Flash.

Los PDA también se utilizan cada vez con más frecuencia como sistemas de georreferencia o bien para el mapeo o navegación de carreteras, al conectarlos a un dispositivo de georreferencia (GPS**:** Sistema de Posicionamiento Global). [8]

Hoy en día se hacen más y más comunes los llamados Smartphone, otro uso más que se le ha dado a las PDA. Con ellos es posible además de todas las funciones mencionadas aquí, recibir mensajes de texto, llamadas y todos los servicios que ofrece la telefonía actual.

Otra funcionalidad que se le ha agregado ha sido la conexión remota a otros dispositivos, como computadoras, y la sincronización con aplicaciones desarrolladas para los mismos. Estos son algunos ejemplos de ese y otros tipos de aplicaciones construidas para la plataforma Pocket PC:

- Data On The Run: Base de Datos de fácil uso, que puede crear archivos en el dispositivo o sincronizarse con Microsoft Access.
- MobiForms For PPC / WinMobile: MobiForms es una herramienta Rapid Application Development (RAD) para bases de datos móviles en Palm OS, Windows Mobile, Symbian y PocketPC.
- Elecont Weather for Windows Pocket PC and Smartphone: Pronóstico del tiempo en su Smartphone o Pocket PC.

Además se han fabricado diversas PDA de tipo industrial que soportan diversas condiciones ambientales. Entre las principales características de dichas PDA que las hacen idóneas para trabajos en arduas condiciones se encuentran:

- Permite una recolección rápida y exacta de datos en arduas condiciones operativas.
- Sellado según las normas IP67, lo cual lo hace resistente al agua y el polvo.
- Temperatura de operación: -30ºC a 50ºC.
- Batería de larga duración.
- Mango (Gun Grip montable opcional) con batería de reserva incorporada.
- Teclado hermético y lumínico.
- Pantalla lumínica touch-screen.
- Lector de código de barras incorporado.
- Ideal para trabajo industrial bajo duras condiciones ambientales de trabajo.
- Slot de expansión PCMCIA Tipo II & CF Tipo I/II accesible al usuario. [16]

La PDA es un colector compacto, resistente y maleable, diseñado para la máxima durabilidad y tolerancia. Es resistente al agua y al polvo, y es un equipo verdaderamente industrial que puede soportar duras condiciones ambientales de trabajo (sumergible en agua hasta un metro de profundidad).
Fácil de usar

- La unidad está equipada con una amplia pantalla, touch-screen y 22 teclas. El teclado es hermético y posee teclas lumínicas altamente visibles, que permiten la manipulación con guantes en débiles condiciones de iluminación. Contiene lectores de alto alcance para lograr una recolección de datos rápida y exacta.
- La batería de larga duración 7.4V 1700mAH soporta períodos operatorios de hasta 12 horas, con opción a una segunda batería en la versión con mango montable (Gun Grip), para operar en períodos más extensos. [16]

Raymond Wolfert, Sales & Marketing Manager de Unitech comentaba “El PA982 es nuestra unidad más robusta y es perfecto para entornos tales como almacenamiento en frío, forestal, almacenamiento donde las temperaturas extremas pueden suponer un problema para el hardware. El sellado IP contra agua y polvo lo hacen perfecto para aplicaciones de la industria de la alimentación. Con el PA982 los usuarios

pueden asegurar que tienen un dispositivo robusto y fiable que mantendrá sus operaciones en funcionamiento en todo momento". [16]

Todos estos usos y características que presentan estos dispositivos no se podría lograr sin tener un conocimiento de la tecnología y herramientas que se usaran para la confección de esta aplicación.

1.3 Metodologías usadas en el desarrollo de la solución

1.3.1 Metodología de desarrollo OpenUP

OpenUP es un Proceso Unificado de Desarrollo de Software que aplica un acercamiento iterativo e incremental dentro de un ciclo de vida estructurado. OpenUP adopta una filosofía ágil y pragmática, que se enfoca en la naturaleza colaborativa del desarrollo de software. Es un proceso apático a herramientas, sin mucha ceremonia, que puede ser extendido para aplicarse a una amplia gama de proyectos.
Entre sus características encontramos:

- Iterativo e incremental.
- Organizado en micro-incrementos.
- Colaboración intensiva entre los miembros del equipo de desarrollo y los stakeholders.
- Óptimo para pequeños grupos de desarrollo.
- Metodología ágil.
- Ciclo de vida del proyecto dividido en Cuatro Fases: Inicio, Elaboración, Construcción y Transición.

Cuatro Principios Medulares capturan las intenciones generales tras OpenUP. Ellos crean las bases para interpretar los roles y productos de trabajo, y realizar las tareas:

- Balancear prioridades concernientes al proyecto, para maximizar el valor de los stakeholders.
- Colaborar para alinear los intereses y compartir el entendimiento.
- Centrarse en la arquitectura tempranamente para minimizar riesgos y organizar el proceso de desarrollo.
- Evolucionar para obtener retroalimentación y progresar continuamente.

Se dice además que OpenUP es Mínimo, Completo y Extensible:

- Mínimo: Solo es incluido el contenido fundamental. El proceso no está gobernado por la creación y entrega de artefactos.
- Completo: Puede ser manifestado como un proceso completo para construir un sistema. Cubre las disciplinas esenciales en un ciclo de vida de desarrollo de software.
- Extensible: Puede ser personalizado para ajustarse a las necesidades de las organizaciones.

[17]

1.4 Tecnologías utilizadas para la confección de la aplicación.

1.4.1 Lenguaje de programación C++.

Como lenguaje de programación se escoge C++ por ser un lenguaje ampliamente difundido, y con una biblioteca estándar que lo ha convertido en un lenguaje universal, además es conocido por los desarrolladores del proyecto y desarrolla el paradigma de programación orientado a objetos (POO), a pesar de su complejidad, es muy potente y veloz, ya que presenta funciones a bajo nivel, permitiendo que se acelere el procesamiento.

Una de las razones de programar en C++ es su increíble versatilidad. Con C++ pueden programarse desde los programas más simples, a los programas más complicados como incluso sistemas operativos. Además es portable, es decir, un programa con el código escrito en C++, se podrá compilar en cualquier sistema operativo o sistema informático sin necesidad de cambiar casi el código fuente. Este es por ejemplo uno de los grandes secretos de Linux, al estar el código escrito en este lenguaje(al menos en su concepción original), es más fácil portarlo a diferentes ordenadores como PC´s, Macintosh, incluso superordenadores. Además otras de las grandes ventajas de C++, es que es un lenguaje multi-nivel, es decir, puedes usarlo tanto para programar directamente el hardware (dependiendo del sistema operativo), como para crear aplicaciones tipo Windows, definidas todas por poseer una misma interfaz.

Es un hecho de que la demanda de programadores de C/C++ en un país vanguardista es muy elevado porque C/C++ es el lenguaje base para la programación de sistemas operativos, compiladores, intérpretes, servidores (core), juegos, herramientas de oficina, herramientas de sistema, drivers, etc. Sólo basta recordar que el compilador java, su máquina virtual, los intérpretes de php, perl, ruby, python, entre otros, están escritos en C/C++. [18]

1.4.2 Qt.

Qt es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario. La biblioteca la desarrolla la que fue su creadora, la compañía noruega Trolltech, actualmente renombrada a Qt Software, y que desde junio de 2008 es propiedad de Nokia. Qt es utilizada en KDE, un entorno de escritorio para sistemas como GNU/Linux o FreeBSD, entre otros. Utiliza el lenguaje de programación C++ de forma nativa y además existen bandines para C, Pitón (Pitt), Java (Tú Jambi), Perl (Perlita), Gambas (gb.qt), Ruby (QtRuby), PHP (PHP-Qt) y Mono (Qyoto) entre otros.

El API (interfaz de programación de aplicaciones) de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL, así como uso de XML y una multitud de otros para el manejo de ficheros, además de estructuras de datos tradicionales.

Breve historia e impacto en el software libre

Inicialmente Qt apareció como biblioteca desarrollada por Trolltech (en aquel momento «Quasar Technologies») en 1992 siguiendo un desarrollo basado en el código abierto, pero no libre. Se usó activamente en el desarrollo del escritorio KDE (entre 1996 y 1998), con un notable éxito y rápida expansión. Esto fomentó el uso de Qt en programas cerrados para el escritorio, situación vista por el proyecto GNU como amenaza para el software libre. Para contrarrestar la situación se plantearon dos ambiciosas iniciativas: por un lado el equipo de GNU en 1997 inició el desarrollo del entorno de escritorio GNOME con GTK para GNU/Linux. Por otro lado intentan hacer una biblioteca compatible con Qt pero totalmente libre, llamada Harmony.

En noviembre de 1998, anuncian el cambio de licencia de Qt que, a pesar de todo, no contaba con el beneplácito de la Free Software Foundation.

El 4 de septiembre de 2000, Trolltech comenzó a ofrecer la biblioteca Qt bajo la licencia GPL versión 2.1.

El 18 de enero de 2008, Trolltech anunció que también ofrecería Qt bajo la licencia GPL v3.

El 14 de enero de 2009, Nokia anunció que Qt v4.5 se licenciaría adicionalmente bajo la licencia LGPL 2.1, con el lema «Qt Everywhere».

Qt cuenta actualmente con un sistema de triple licencia: GPL v2/v3 para el desarrollo de software de código abierto (open source) y software libre, y otra de pago para el desarrollo de aplicaciones con cualquier licencia.

Actualmente se encuentra la versión 4 de la biblioteca, y además de las múltiples mejoras, ahora las bibliotecas Qt son también liberadas bajo licencia GPL para Windows y Mac.

Plataformas

Qt se encuentra disponible para las siguientes plataformas:

- X11 - Para X Windows System con licencia GNU. (Linux, Unix, BSD).
- Mac - Para Mac OS X bajo la licencia GNU.
- Windows - Para sistemas Windows con licencia GNU (las antiguas versiones, anteriores a la 4.X eran no libres para este sistema operativo).
- PDA - Para dispositivos empotrados, también con licencia GNU y generalmente distribuido junto con Qt Extended, un entorno completo para PDAs.

Actualmente también está disponible QSA (Qt Scripts for Applications), que, basándose en ECMAScript/JavaScript, permite introducir y crear scripts en las aplicaciones creadas con Qt.

Además existen 4 ediciones de Qt disponibles dentro de cada una de las plataformas anteriores, llamadas:

- Qt Console - edición para desarrolladores non-GUI.
- Qt Desktop Light - edición con nivel reducido de GUI, orientado a redes y bases de datos.
- Qt Desktop - edición completa.
- Qt Open Source Edition - edición "completa", con algunas excepciones como el control ActiveQt (ActiveX) para Windows, destinada a desarrolladores de software libre.

1.4.3 XML

Dentro de las tecnologías disponibles hoy en día, existe una tecnología que ofrece la posibilidad de realizar una separación efectiva entre los datos y la representación de los mismos. Esta tecnología es XML (del inglés ***Ex**tensible **M**arkup **L**anguage*) desarrollado por World Wide Web Consortium.

El lenguaje XML utiliza una sintaxis mínima, de forma que al generar los programas el propio programador sea quien defina cuáles son los "tags" adecuados según su conveniencia. Esto da una flexibilidad sin par, ya que se puede definir cualquier tipo de estructura de datos sin ningún tipo de cortapisa.

Ventajas del XML.

- Es extensible: Después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna.
- El analizador es un componente estándar, no es necesario crear un analizador específico para cada versión de lenguaje XML. Esto posibilita el empleo de cualquiera de los analizadores disponibles. De esta manera se evitan *bugs* y se acelera el desarrollo de aplicaciones.
- Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarla. Mejora la compatibilidad entre aplicaciones.

XML es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML pero su función principal es describir datos y no mostrarlos como es el caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones.
Las tecnologías XML son un conjunto de módulos que ofrecen servicios útiles a las demandas más frecuentes por parte de los usuarios. XML sirve para estructurar, almacenar e intercambiar información.

Entre las tecnologías XML disponibles se pueden destacar:

- XSL: Lenguaje Extensible de Hojas de Estilo, cuyo objetivo principal es mostrar cómo debería estar estructurado el contenido, cómo debería ser diseñado el contenido de origen y cómo debería ser paginado en un medio de presentación como puede ser una ventana de un navegador Web o un dispositivo móvil, o un conjunto de páginas de un catálogo, informe o libro.
- XPath: Lenguaje de Rutas XML, es un lenguaje para acceder a partes de un documento XML.
- XLink: Lenguaje de Enlace XML, es un lenguaje que permite insertar elementos en documentos XML para crear enlaces entre recursos XML.
- XPointer: Lenguaje de Direccionamiento XML, es un lenguaje que permite el acceso a la estructura interna de un documento XML, esto es, a sus elementos, atributos y contenido.

- XQL: Lenguaje de Consulta XML, es un lenguaje que facilita la extracción de datos desde documentos XML. Ofrece la posibilidad de realizar consultas flexibles para extraer datos de documentos XML en la Web. [19]

1.4.4 Lenguaje de modelado UML.

UML (Unified Modeling Language) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucran una gran cantidad de software. Está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Debido a que el mismo es un lenguaje, cuenta con reglas para combinar tales elementos y conformar diagramas para el análisis del sistema, además de permitir la modelación de sistemas con tecnología orientada a objetos. Entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables. [20]

Hoy en día el lenguaje de modelado visual está consolidado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo. Mediante UML es posible establecer la serie de requerimientos y estructuras necesarias para plasmar un sistema de software previo al proceso intensivo de escribir código. [20]

Características:

- Permite modelar sistemas utilizando técnicas Orientadas a Objetos (OO).
- Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Puede conectarse con los lenguajes de programación (Ingeniería directa e inversa).
- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, etc.).
- Cubre las cuestiones relacionadas con el tamaño propio de los sistemas complejos y críticos.
- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas.

- Existe un equilibrio entre expresividad y simplicidad, pues no es difícil de aprender ni de utilizar.
- UML es independiente del proceso, aunque para utilizarlo óptimamente se debería usar en un proceso que fuese dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental. [21]

1.5 Herramientas utilizadas en la confección de la solución

1.5.1 Visual Paradigm

Visual Paradigm es una herramienta CASE (Computer-Aided Software Engineering) que permite realizar ingeniería tanto directa como inversa. Es una herramienta colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto; genera la documentación del proyecto automáticamente en varios formatos como Web o pdf y permite el control de versiones. Cabe destacar igualmente su robustez, usabilidad y portabilidad. Visual Paradigm para UML es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

Características:

- Producto de calidad.
- Soporta aplicaciones web.
- Las imágenes y reportes generados, no son de muy buena calidad.
- Varios idiomas.
- Generación de código para Java y exportación como HTML.
- Fácil de instalar y actualizar.
- Compatibilidad entre ediciones.

1.5.2 Microsoft Visual Studio

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#,

ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros.

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión net 2002). Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles. [22]

Visual Studio 2005

Visual Studio 2005 se empezó a comercializar a través de Internet a partir del 4 de Octubre de 2005 y llegó a los comercios a finales del mes de Octubre en inglés. En castellano no salió hasta el 4 de Febrero de 2006. Microsoft eliminó .NET, pero eso no indica que se alejara de la plataforma .NET, de la cual se incluyó la versión 2.0.

La actualización más importante que recibieron los lenguajes de programación fue la inclusión de tipos genéricos, similares en muchos aspectos a las plantillas de C#. Con esto se consigue encontrar muchos más errores en la compilación en vez de en tiempo de ejecución, incitando a usar comprobaciones estrictas en áreas donde antes no era posible. C++ tiene una actualización similar con la adición de C++/CLI como sustituto de C# manejado.

Se incluye un diseñador de implantación, que permite que el diseño de la aplicación sea validado antes de su implantación. También se incluye un entorno para publicación web y pruebas de carga para comprobar el rendimiento de los programas bajo varias condiciones de carga.

Visual Studio 2005 también añade soporte de 64-bit. Aunque el entorno de desarrollo sigue siendo una aplicación de 32 bits Visual C++ 2005 soporta compilación para x86-64 (AMD64 e Intel 64) e IA-64 (Itanium). El SDK incluye compiladores de 64 bits así como versiones de 64 bits de las librerías. Se lanzó el Service Pack 1 para Visual Studio 2005 el 14 de Diciembre de 2006.

La versión interna de Visual Studio 2005 es la 8.0, mientras que el formato del archivo es la 9.0.

Microsoft Visual Studio 2005 es un software que dentro de sus características fundamentales se encuentran algunas como, una muy buena detección y corrección de errores, capacidad de crear software profesional con rapidez, reducir los costos de funcionamiento de tecnologías de la información y además se integra con una amplia gama de aplicaciones, sistemas y dispositivos. Contiene muchas librerías con códigos pre-compilados que ayudan en la escritura del código de la aplicación.

1.5.3 QtCreator

Qt Creator es un IDE creado por Trolltech para el desarrollo de aplicaciones con las bibliotecas Qt, requiriendo su versión 4.x. Este IDE se encuentra actualmente en su versión 1.0 lanzada en este año (2009), después del lanzamiento de Qt 4.5. La versión preliminar tiene una licencia especial, pero la licencia de la versión definitiva aún está en estudio, será compatible con GPL y tendrá una versión de código abierto y otra comercial.

Los sistemas operativos que soporta en forma oficial son:

Linux 2.6.x, para versiones de 32 y 64 bits con Qt 4.x instalado. Además hay una versión para Linux con gcc 3.3.
Mac OS X 10.4 o superior, requiriendo Qt 4.x
Windows XP y Vista, requiriendo el compilador MinGW y Qt 4.4.3 o superior.

CONCLUSIONES DEL CAPÍTULO

El capítulo anterior hizo un estudio sobre el objeto de la investigación tratada, y quedaron enunciados importantes conceptos que están estrechamente relacionados con el uso de las PDA. Expuso algunas de las características de las PDA y su uso en las diferentes ramas de la industria. Además, obtuvo los elementos necesarios para dar respuesta al problema científico de este trabajo.

CAPÍTULO II. CARACTERÍSTICAS DEL SISTEMA.

INTRODUCCIÓN

Este capítulo obtiene una visión más específica del sistema que va a desarrollar, partiendo fundamentalmente de las soluciones técnicas. De inicio obtiene una panorámica desde el punto de vista del dominio del sistema dando continuidad a crear la concepción práctica del producto a elaborar partiendo de las necesidades del cliente desde el punto de vista de que es lo que debe hacer el sistema.

2.1 Modelo de Negocio

El modelo de negocios es el estudio de la organización. Es utilizado por el analista para comprender el sector, ya sea industrial o de negocio, al que el sistema va a servir.

Durante el proceso de modelado del negocio, se examina la estructura de la organización y se observan los roles en la compañía y como estos se relacionan. Además, se deben examinar las entidades externas, cualquier individuo u otras compañías, y cómo interactúan con el negocio, y observar las implicaciones de esas interacciones. Con el modelado de los procesos se identificaron las necesidades de cada entidad en función de las actividades que realizan.

2.2 Modelo de Casos de Uso del Negocio

2.2.1 Determinación y justificación de los actores del negocio

Jefe de Empresa: El Jefe de Empresa es el que inicia todas las acciones que dan comienzo a los procesos de negocio analizados, y al mismo tiempo es el principal beneficiado con el resultado de dichos procesos de negocio.

2.2.2 Determinación y justificación de los trabajadores del negocio

Lector del Contador: Es el encargado de descargar todos los datos del metro-contador de la empresa al sistema utilizando para esto cuestionarios de papel. No se beneficia en ningún momento de las acciones ejecutadas en los procesos de negocio, sino que se limita a ejecutar dichas acciones.

2.2.3 Diagrama de Casos de Uso del Negocio

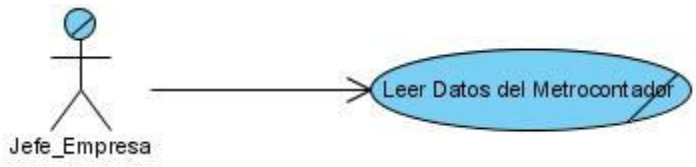


Fig.1: Diagrama de Caso de Uso del Negocio.

2.3 Requerimientos

2.3.1 Captura de requisitos

2.3.1.1 Requisitos Funcionales

Los requerimientos funcionales especifican acciones que el sistema debe ser capaz de realizar, sin tomar en consideración ningún tipo de restricción física. Es por ello que su definición debe ser clara y libre de ambigüedades.

A continuación se muestran los requerimientos funcionales definidos para la realización de la aplicación a desarrollar.

- **RF 1** Definir cuáles de las variables energéticas son relevantes para el cliente y el protocolo de comunicación a utilizar, en una *configuración de descarga*.
  La configuración por defecto incluiría las variables de día, pico y madrugada del período de máxima demanda, la máxima demanda total y la fecha de la máxima demanda total.
- **RF 2** Almacenar las diferentes configuraciones de descarga en la aplicación.
- **RF 3** Eliminar una o varias configuraciones de descarga existentes en la aplicación.

- **RF 4** Realizar la comunicación del PDA con el metro-contador mediante la interfaz serie RS232. Para el caso del metro-contador CIRWATT el sistema debe hacer uso del protocolo de comunicación ModBus RTU.
- **RF 5** Leer los valores de las variables energéticas contenidas en el metro-contador y definidas en una configuración de descarga.
- **RF 6** Almacenar los valores de las variables leídas del metro-contador en la aplicación.
- **RF 7** Eliminar uno o varios juegos de datos descargados existentes en la aplicación.
- **RF 8** Realizar la comunicación del PDA con una computadora personal mediante el puerto serie RS232.
- **RF 9** Almacenar los valores de las variables contenidos en el PDA hacia un fichero XML localmente.
- **RF 10** Visualizar los valores descargados en el sistema.

2.3.1.2 Requisitos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Normalmente están vinculados a requerimientos funcionales, es decir una vez que se conozca lo que el sistema debe hacer se determina cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser.

Los requerimientos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación, por lo que se consideran fundamentales en el éxito del producto.

A continuación se presentan los requerimientos no funcionales definidos para la realización de la aplicación a desarrollar.

- **Usabilidad**

  El sistema se podrá usar completamente desde el teclado sin necesidad del stylus.

  El tiempo necesario para la asimilación del sistema por los usuarios será menor a 15 días.

- **Fiabilidad**

  El sistema validará las entradas del usuario para no permitir valores fuera de los rangos probables.

- **Funcionamiento**

  El tiempo de respuesta del sistema para el almacenamiento de la lectura manual de datos será menor de 1 segundo.

  El tiempo de transferencia de datos entre el sistema y una PC será menor de un minuto.

- **Soportabilidad**

  El lenguaje de programación a utilizar es C++.

  El IDE de desarrollo para la creación del sistema será QtCreator.

  Como gestor de base de datos se utilizará SQLite.

Como sistema operativo se utilizará Windows CE 5.0

El sistema debe ser lo suficientemente pequeño y eficiente para que pueda portarse y ejecutarse en el dispositivo PDA.

Interfaces del Sistema

- **Interfaces de usuarios**

  La interfaz de usuario es una ventana sencilla, por lo que el usuario puede predecir que sucederá.

- **Interfaces de Software**

  El protocolo a usar es ModBus.

- **Interfaces de Hardware**

  La aplicación hace uso del puerto RS232 para comunicarse con los metros contadores.

Disposición y Exigencias de la navegación

La distribución de los elementos en la pantalla debe realizarse lo mas óptimo posible, ubicando solo los componentes necesarios y eliminando el resto.

Interfaces con Sistemas Externos o Dispositivos

El sistema hace uso de las interfaces del software RTE que implementa el driver de comunicación con el dispositivo, este define unas interfaces que hacen uso de la tecnología XML para la configuración, y SQL para el soporte de los datos.

Coacciones del sistema

El sistema debe implementarse en la plataforma de dispositivos portátiles de Microsoft Pocket PC, no obstante debe garantizar que se use un lenguaje multiplataforma como C++ para la posterior reutilización de cualquier componente. Además debe ser un sistema pensado para el poco consumo de recursos, debido a la poca capacidad de procesamiento de los dispositivos donde estará ubicado.

2.4 Modelo de Casos de Uso

2.4.1 Definición de los casos de uso

Caso de uso	Nombre
CU-1	Descargar Datos hacia el PDA
CU-2	Configurar Descarga en el PDA
CU-3	Exportar Datos
CU-4	Visualizar Descarga

Tabla 2: Definición de los casos de uso.

CU-1	Descargar Datos hacia el PDA.
Actor	Operador
Descripción	Descargar los datos del metro-contador hacia la PDA de forma automática.

Tabla 3: Caso de Uso “Descargar Datos hacia el PDA”.

CU-2	Configurar Descarga en el PDA.
Actor	Operador
Descripción	Configurar los datos que necesite descargar del metro-contador hacia el PDA.

Tabla 4: Caso de Uso “Configurar Descarga en el PDA”.

CU-3	Exportar Datos
Actor	Operador
Descripción	Exportar los datos descargados hacia un archivo XML.

Tabla 5: Caso de Uso “Exportar Datos”.

CU-4	Visualizar Descarga
Actor	Operador
Descripción	Se muestran los valores de las variables de una descarga.

Tabla 6: Caso de Uso “Visualizar Descarga”.

2.4.2 Determinación y justificación de los actores del sistema

Operador: Encargado de exportar datos hacia la computadora, descargar los datos del metro-contador hacia la PDA de forma manual o automática y configurar los datos en la PDA.

2.4.3 Diagrama de Casos de Uso del Sistema

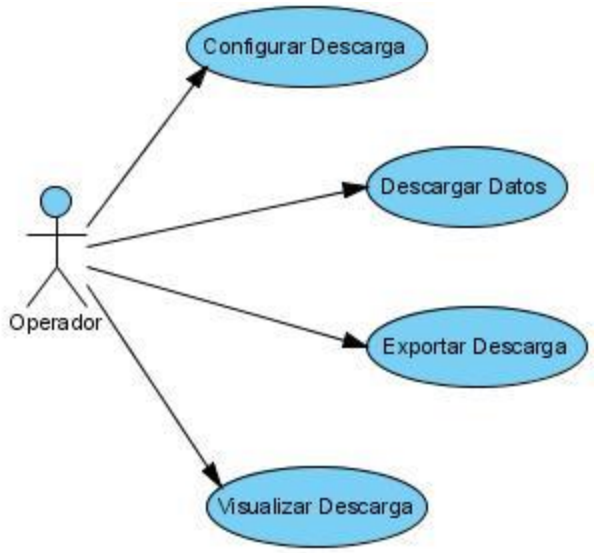


Fig.2: Diagrama de casos de uso del sistema.

2.5 Descripción de los Casos de Uso en formato expandido.

2.5.1 Descripción Textual del Caso de Uso: Descargar Datos hacia el PDA.

1 Breve Descripción

El caso de uso se inicia cuando el operador mediante el sistema descarga los datos del metro-contador hacia el PDA, finalizando así el caso de uso.

2 Breve Descripción de los Actores

- **Operador:** Encargado de exportar datos hacia la computadora, descargar los datos del metro-contador hacia la PDA de forma manual o automática y configurar los datos en la PDA.

3 Precondiciones

- Debe de existir la conexión entre el metro-contador y la PDA por el puerto serie RS232.

4 Flujo Básico de Eventos

4.1 Descargar Datos.

4.1.1 El operador tiene la posibilidad de seleccionar la operación que desea ejecutar ya sea Descargar Datos ó Eliminar una Descarga existente.

-Si desea Descargar Datos escoger la opción Descargar Datos.

-Si desea Eliminar Descarga escoger la opción Eliminar Descarga.

5 Escenarios Claves.

5.1 Escenario 1: Descargar Datos.

5.1.1 El operador selecciona la opción de Descargar Datos hacia la PDA. El sistema muestra un formulario para insertar los datos generales del metro-contador.

5.1.2 El operador inserta los datos generales del metro-contador: identificador, tipo de metro-contador.

5.1.3 El sistema pide que se escoja la configuración de descarga a utilizar.

5.1.4 El operador selecciona la configuración de descarga correspondiente.

5.1.5 El sistema verifica la configuración.

5.1.6 El sistema realiza la Descarga Automática de los datos.

5.1.7 El sistema almacena los datos en el PDA y se le muestra el mensaje al operador ¨Descarga Satisfactoria¨; finalizando así el caso de uso.

5.2 Escenario 2: Eliminar Descarga.

5.2.1El operador selecciona la opción de Eliminar Descargar.

5.2.2 El sistema muestra un formulario para seleccionar la(s) Descargas a eliminar.

5.2.3 El operador selecciona la(s) Descargas a Eliminar y presiona el botón “Eliminar y Salir”.

5.2.4 El sistema elimina las Descargas del PDA y se muestra el mensaje de “Descargas Eliminadas”.

6 Flujos Alternos

6.1 Escenario 1: Descargar Datos.

6.1.1 En caso que la configuración no sea compatible con el tipo de metro-contador se le muestra al operador el mensaje de error “Configuración Incompatible”.

6.1.2 En caso que no se puede realizar la descarga automática se le muestra mensaje al operador ¨Descarga Fallida¨.

7 Poscondiciones

7.1 Quedan almacenados en la PDA los datos del metro-contador.

7.2 Quedan eliminados la(s) descarga(s) de la PDA.

8 Referencias

RF4, RF5, RF6, RF7.

2.5.2 Descripción Textual del Caso de Uso: Configurar Descarga en el PDA.

1 Breve Descripción

- El caso de uso se inicia cuando el Operador mediante el sistema, configura los datos que necesite descargar del metro-contador hacia el PDA, finalizando el caso de uso.

2 Breve Descripción de los Actores

Operador: Encargado de exportar datos hacia la computadora, descargar los datos del metro-contador hacia la PDA de forma manual o automática y configurar los datos en la PDA.

3 Flujo Básico de Eventos

3.1 Configurar Descarga.

3.1.1 El operador tiene la posibilidad de seleccionar la operación que desea ejecutar ya sea Configurar Descarga ó Eliminar una Configuración existente.

-Si desea Configurar Descarga escoger la opción Configurar Descarga.

-Si desea Eliminar Configuración escoger la opción Eliminar Configuración.

4 Escenarios Claves.

4.1 Escenario 1: Configurar Descarga.

4.1.1 El operador selecciona la opción Configurar Descarga.

4.1.2 El sistema le brinda la posibilidad de configurar los datos a descargar (Crear cada una de las variables a descargar, insertando su nombre, dirección física y tipo de dato que representa) ó seleccionar la opción de Modificar una Configuración existente.

- Si se elige la opción de Configurar Descarga
  - El operador configura los datos a descargar y oprime el botón “Guardar y Salir”.
  - El sistema verifica que no existan campos en blanco.
  - El sistema almacena la configuración de datos y muestra el mensaje “Configuración Almacenada”.
- Si se elige la opción de Modificar Descarga.
  - El sistema brinda las diferentes Configuraciones de Descarga existentes.
  - El operador elige la Configuración a Modificar.
  - El sistema brinda los datos de dicha configuración.
  - El operador modifica los datos y oprime el botón “Guardar y Salir”.
  - El sistema verifica que no existan campos en blanco.
  - El sistema almacena la configuración de datos y muestra el mensaje “Configuración Almacenada”.

4.2 Escenario 2: Eliminar Configuración.

4.2.1 El operador selecciona la opción de Eliminar Configuración.

4.2.2 El sistema muestra un formulario para seleccionar la(s) Configuraciones a eliminar.

4.2.3 El operador selecciona la(s) Configuraciones a Eliminar y presiona el botón “Eliminar y Salir”.

4.2.4 El sistema elimina las Configuraciones y se muestra el mensaje de “Configuraciones Eliminadas”.

5 Flujos Alternos

5.1 Escenario1: Configurar Descarga.

5.1.1 En caso de existir campos en blanco, se le muestra al operador el mensaje “No pueden existir campos vacíos”.

5.1.2 En caso de existir identificador del metro-contador, se le muestra al operador el mensaje “Ese identificador ya está en uso”.

6 Poscondiciones

6.1 Se configuraron satisfactoriamente los datos.

6.2 Se eliminaron satisfactoriamente la(s) configuración(es).

6.3 Se modifico satisfactoriamente la configuración.

7 Referencias

RF1, RF2, RF3.

2.5.3 Descripción Textual del Caso de Uso: Exportar Datos.

1 Breve Descripción

- El caso de uso se inicia cuando el operador decide exportar los datos del PDA hacia la computadora, los datos son exportados, finalizando así el caso de uso.

2 Breve Descripción de los Actores

Operador: Encargado de exportar datos hacia la computadora, descargar los datos del metro-contador hacia la PDA de forma manual o automática y configurar los datos en la PDA.

3 Precondiciones

- El PDA debe de tener almacenado los datos descargados de algún metro-contador.

4 Flujo Básico de Eventos

4.1 Exportar Datos

4.1.1 El operador decide exportar los datos descargados en el PDA, pulsando la opción Exportar

4.1.2 El sistema muestra los datos almacenados en el PDA.

4.1.4 El operador decide cuales datos se van a exportar.

4.1.5 El sistema almacena los datos y muestra mensaje “Datos Exportados”.

5 Flujos Alternos

5.1. Ya existe el archivo exportado de los datos, el sistema muestra el mensaje “El archivo ya existe, desea sobrescribirlo?”, el usuario decidirá entonces cual de las opciones tomar:

- Si: se sobrescriben los datos.
- No: se preservan los originales y los nuevos no se exportan (persisten dentro del sistema).

6 Poscondiciones

- Los datos han sido exportados hacia la computadora satisfactoriamente.

7 Referencias

RF8, RF9.

2.5.4 Descripción Textual del Caso de Uso: Visualizar Descarga.

1 Breve Descripción

- El caso de uso se inicia cuando el operador decide visualizar los datos de una descarga determinada almacenada en el PDA, los mismos le son mostrados, finalizando así el caso de uso.

2 Breve Descripción de los Actores

Operador: Encargado de exportar datos hacia la computadora, descargar los datos del metro-contador hacia la PDA de forma manual o automática y configurar los datos en la PDA.

3 Precondiciones

- El PDA debe de tener almacenado los datos descargados de algún metro-contador.

4 Flujo Básico de Eventos

4.2 Exportar Datos

4.1.1 El operador decide visualizar los datos descargados en el PDA, pulsando la opción Visualizar.

4.1.3 El sistema muestra las descargas realizadas.

4.1.4 El operador decide cual descarga desea visualizar.

4.1.5 El sistema muestra los valores y atributos de cada una de las variables descargadas.

5 Poscondiciones

- Los datos son mostrados al operador.

6 Referencias

RF10.

CONCLUSIONES DEL CAPÍTULO

Este capítulo recopiló los requisitos funcionales y no funcionales, define y describe los casos de uso de acuerdo con lo que espera el cliente que realice el módulo a desarrollar en el primer ciclo de desarrollo. Ya se puede proceder con el diseño del mismo.

CAPÍTULO III. DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN.

INTRODUCCIÓN.

Este capítulo describe todos los aspectos relacionados con el módulo a construir mediante los diagramas de secuencia de la realización de los casos de uso que intervendrán en el primer ciclo de desarrollo del proyecto. Este paso constituye el refinamiento de las etapas anteriores.
Además, hace una descripción de cada una de las clases del módulo.

3.1 Patrones.

Una definición de patrón interesante es: un patrón no es más que la solución estándar de un problema, una forma de flexibilizar el código sin dejar de satisfacer ciertos criterios obligatorios, expresa como solucionar un problema que ocurre repetidas veces en algún ámbito determinado de desarrollo de software y brindan una buena solución probada con anterioridad. Ahorra el tiempo dedicado al diseño, la mayoría de las veces se logran construir soluciones reutilizables y extensibles.

Son soluciones basadas en la experiencia y que se ha demostrado que funcionan. [23]

CRUD

Este patrón engloba las operaciones de crear, actualizar, leer y eliminar tipos de información en un solo caso de uso formando una unidad conceptual. Este patrón debe aplicarse cuando todas las acciones estén enfocadas en un mismo valor o entidad del negocio. Describe los casos de uso que se encargan de administrar o gestionar la información, relacionándose con el usuario que maneja dicha información.

MVC

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

- **Modelo**: Esta es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos; por ejemplo, no

permitiendo comparar un número de unidades negativo, calculando si hoy es el cumpleaños del usuario o los totales, impuestos o importes en un carrito de la compra.

- **Vista**: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- **Controlador**: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

3.2 Estándares de codificación

Un estándar de codificación comprende los aspectos de la generación de código y repercute directamente en la legibilidad y la extensibilidad de cualquier proyecto de Software, haciendo que nuevos desarrolladores se acoplen rápidamente al proceso de desarrollo.

Las convenciones o estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código.

CONSTANTES

Los nombres de las constantes se escriben completamente con mayúsculas y en español, en caso de que el nombre este compuesto por más de una palabra, estas son separadas por el carácter “_”.

CLASES

Los nombres de las clases se escriben con letra inicial mayúscula y en español, en caso de tener un nombre conformado por más de una palabra estas se escribirán una a continuación de la otra comenzando siempre cada palabra con una letra mayúscula y el resto en minúscula. Si es una clase interfaz comenzará con el prefijo "Inter" seguido por el nombre de dicha clase. Si es una clase entidad el nombre se escribirá con minúscula y en español.

NOMBRES DE ENUMERADOS

Los enumerados comenzarán con la letra “E” y a continuación la palabra o palabras que conformen el nombre, cada una comenzando con letra mayúscula y el resto en minúsculas siempre en español.

Los valores de los enumerados cumplen con las mismas reglas que las constantes.

COMENTARIOS

Formato: Específica cómo va estar escrito el comentario, el cual deberá ser en lenguaje español.

Posición: La Ubicación del comentario deberá ser antes del bloque de código correspondiente. Y en caso de especificar una línea de código se pondría el comentario al final de la línea de código.

CONSTRUCTOR Y DESTRUCTOR

ClaseNombre (bool arg_bVarNombre, float& arg_fVarNombre);
~ClaseNombre ();

ORGANIZACIÓN VISUAL

- Declarar las variables del mismo tipo en la misma sentencia separadas por comas y en español.
  Ejemplo:
  int a, b;

- Añadir comentarios descriptivos junto a cada declaración de variables, si es necesario.

- Deben incluirse espacios en ambos lados de los operadores binarios.
  Ejemplo:
  y = 50 + 15 - x;

3.3 Diagramas de clases del diseño.

Un diagrama de clases representa las clases que serán utilizadas dentro del sistema y las relaciones que existen entre ellas. La definición de clase incluye definiciones para atributos y operaciones. Los diagramas de clases por definición son estáticos, representan qué partes interactúan entre sí.

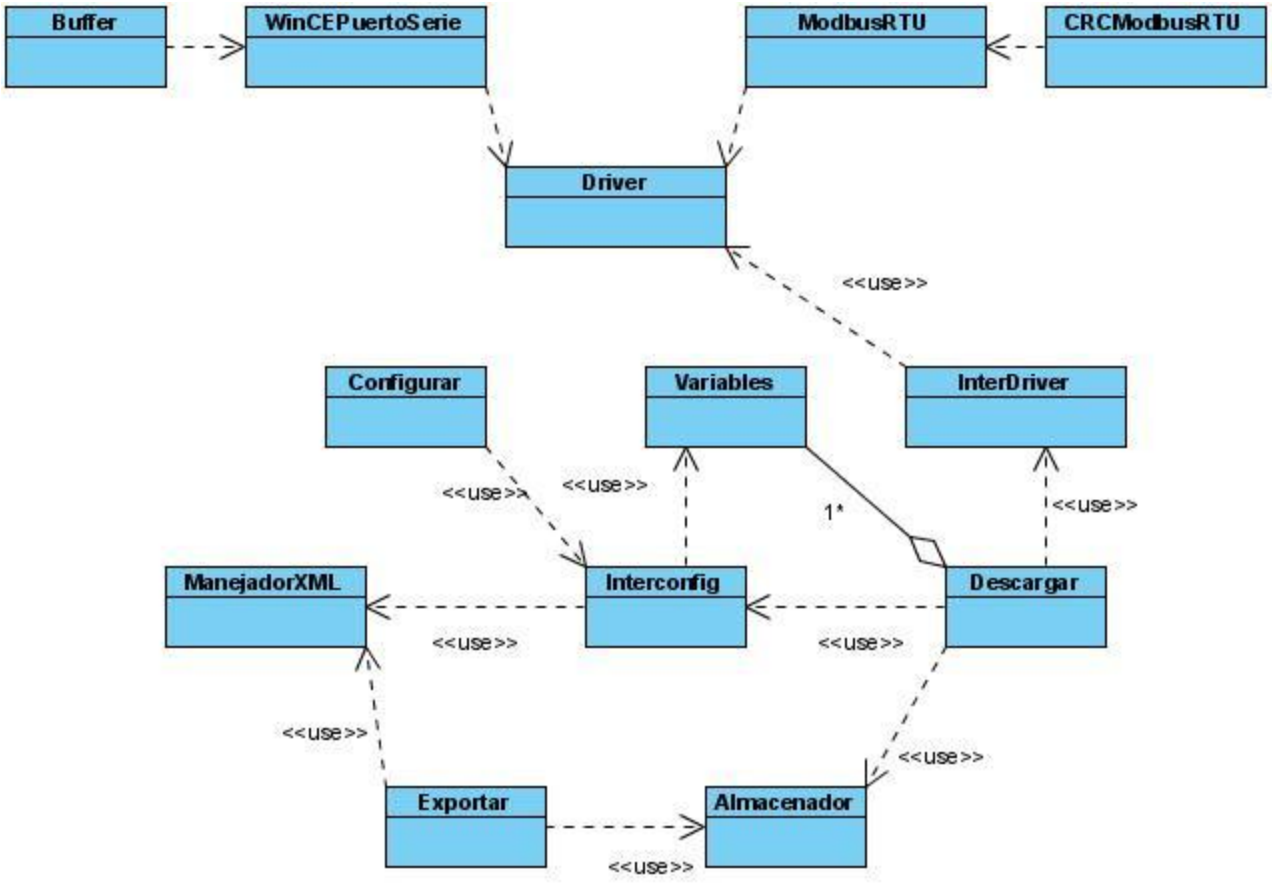
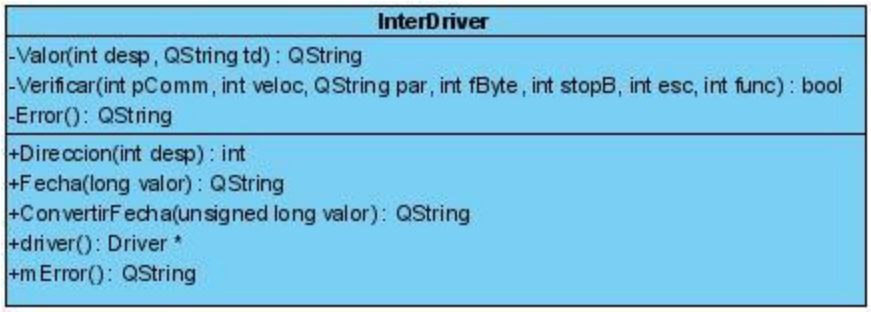
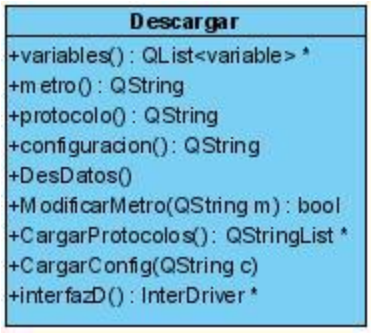
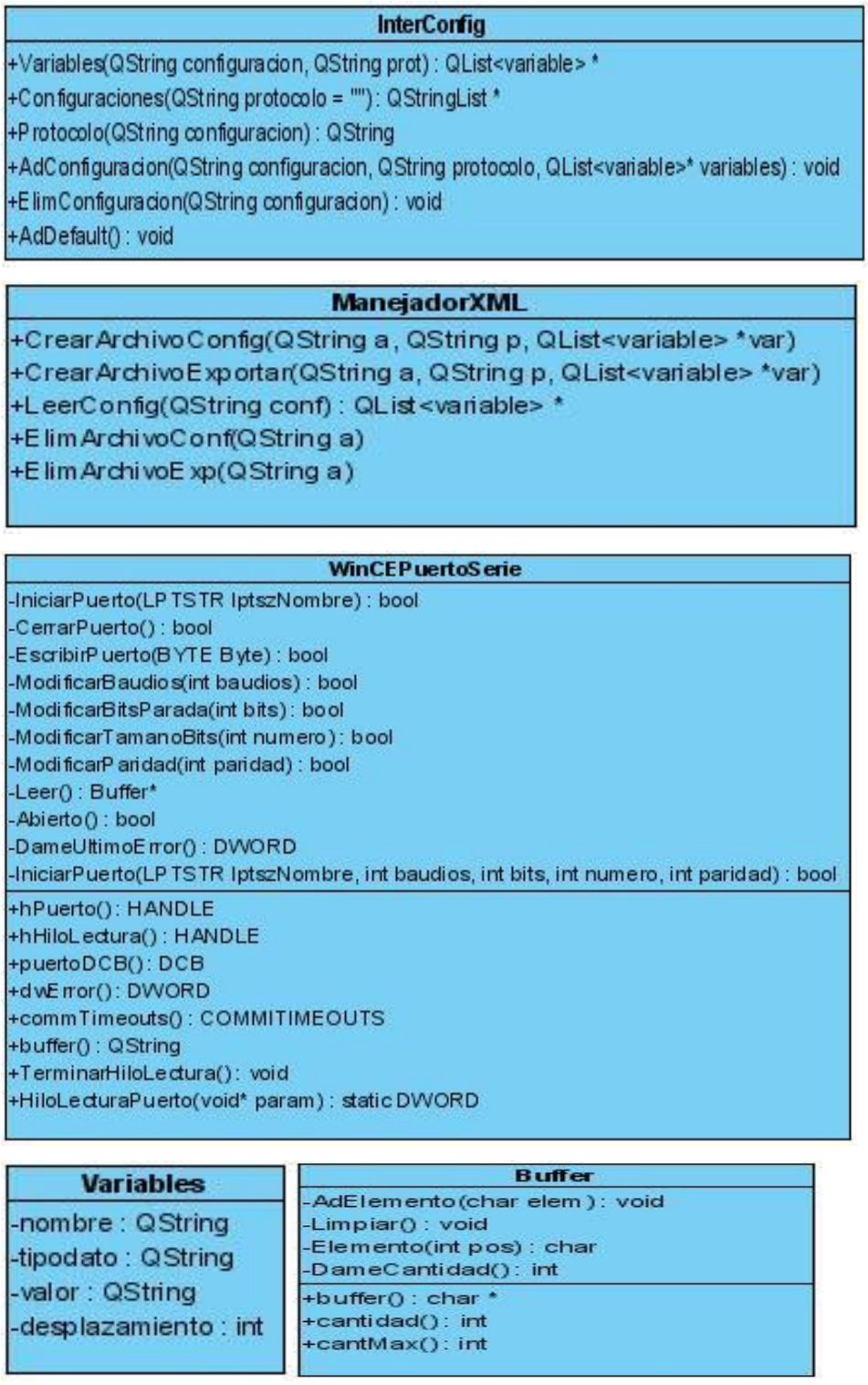


Fig.3: Diagrama de clases del Diseño.

Expansión de las clases del diseño






InterConfig
+Variables(QString configuracion, QString prot) : QList<variable> *
+Configuraciones(QString protocolo = "") : QStringList *
+Protocolo(QString configuracion) : QString
+AdConfiguracion(QString configuracion, QString protocolo, QList<variable>* variables) : void
+ElimConfiguracion(QString configuracion) : void
+AdDefault() : void
ManejadorXML
+CrearArchivoConfig(QString a, QString p, QList<variable> *var)
+CrearArchivoExportar(QString a, QString p, QList<variable> *var)
+LeerConfig(QString conf) : QList<variable> *
+ElimArchivoConf(QString a)
+ElimArchivoExp(QString a)
WinCEPuertoSerie
-IniciarPuerto(LPTSTR lptszNombre) : bool
-CerrarPuerto() : bool
-EscribirPuerto(BYTE Byte) : bool
-ModificarBaudios(int baudios) : bool
-ModificarBitsParada(int bits) : bool
-ModificarTamanoBits(int numero) : bool
-ModificarParidad(int paridad) : bool
-Leer() : Buffer*
-Abierto() : bool
-DameUltimoError() : DWORD
-IniciarPuerto(LPTSTR lptszNombre, int baudios, int bits, int numero, int paridad) : bool
+hPuerto() : HANDLE
+hHiloLectura() : HANDLE
+puertoDCB() : DCB
+dwError() : DWORD
+commTimeouts() : COMMITIMEOUTS
+buffer() : QString
+TerminarHiloLectura() : void
+HiloLecturaPuerto(void* param) : static DWORD
Variables
-nombre : QString
-tipodato : QString
-valor : QString
-desplazamiento : int
Buffer
-AdElemento(char elem) : void
-Limpiar() : void
-Elemento(int pos) : char
-DameCantidad() : int
+buffer() : char *
+cantidad() : int
+cantMax() : int

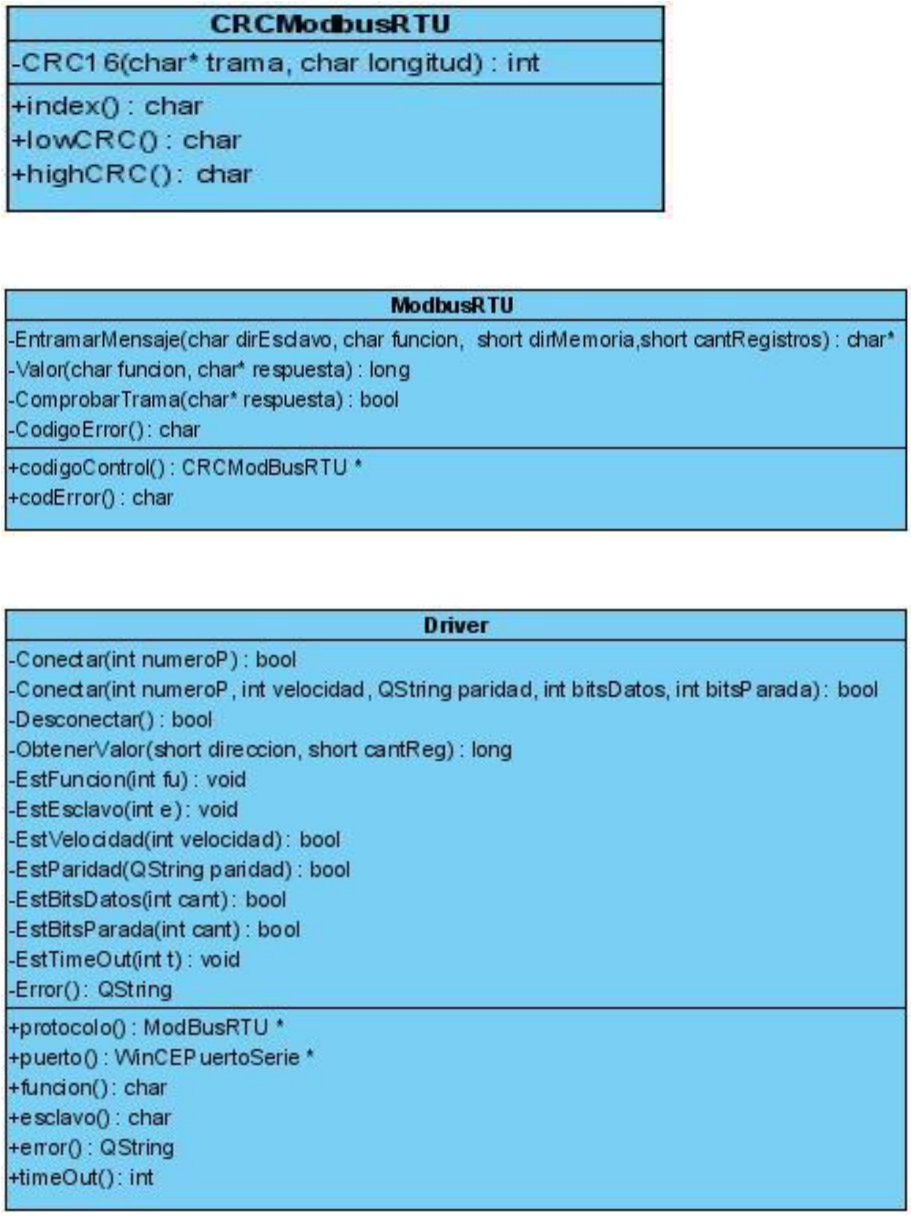


Fig.4: Expansión de las clases del diseño.

3.4 Descripción de las clases del diseño.

Después de haber visto los diagramas de clases es necesario especificar las características de ellas. En las tablas que aparecen a continuación se describen los atributos y métodos de las clases incluidas en el módulo, estas son: Almacenador, Exportar, ManejadorXML, Variables, Buffer, CRCModbusRTU, ModbusRTU, InterConfig, InterDriver, Descargar, Configurar, WinCEPuertoSerie, EmbedbeDriver. Principal, Visualizar.

Nombre: InterConfig		
Descripción: Clase que brinda una interfaz al subsistema de configuración para el resto de la aplicación.		
Tipo de Clase: Interfaz		
Atributo	**Tipo**	**Descripción**
Variables(QString configuración)	static QList<Variable>*	Devuelve una lista de variables de una configuración dada.
Configuraciones(QString protocolo = "")	static QStringList*	Lista de configuraciones que cumplen con un protocolo.
Protocolo(QString configuración)	static QString	Devuelve protocolo de una configuración dada.
AdConfiguracion(QString configuración, QString protocolo, QList<Variable>* variables)	static void	Adiciona una nueva Configuración.
ElimConfiguración(QString configuración)	static void	Elimina una Configuración dada.
AdDefault()	static void	Crea las configuraciones por defecto de cada protocolo.

Tabla 7: Clase InterConfig

Nombre: Principal		
Descripción: Clase que provee un punto común para todas las interfaces de los diferentes casos de uso		
Tipo de Clase: Controladora		
Atributo	**Tipo**	**Descripción**
on_pushButtonVis_clicked()	Void	Acción click del componente visual Visualizar.
on_pushButtonDelDesc_clicked()	Void	Acción click del componente visual Borrar Descarga.
on_pushButtonDelConfig_clicked()	Void	Acción click del componente visual Borrar Configuración.
on_pushButtonExport_clicked()	void	Acción click del componente visual Exportar.
on_pushButton_clicked()	Void	Acción click del componente visual Salir.
on_pushButtonDesc_clicked()	Void	Acción click del componente visual Descargar.
Cancelado()	Void	SLOT: se ejecuta cuando el usuario cancela una acción.
Realizado(QString mensaje)	Void	SLOT: se realiza cuando termina un caso de uso satisfactoriamente.
Acerca()	Void	Muestra información general del producto(nombre, versión)
SelecVacia(QString mensaje)	Void	SLOT: acción que se ejecuta cuando una lista de selección dada esta vacía.

Tabla 8: Clase Principal

Nombre: Variables		
Descripción: Clase que encapsula los atributos de una variable en la aplicación, para facilitar su procesamiento.		
Tipo de Clase: Entidad		
Atributo	**Tipo**	**Descripción**
Nombre	QString	Declara el nombre de una variable.
tipoDato	QString	Declara el tipoDato de una variable.
desplazamiento	QString	Declara el desplazamiento de una variable.
Valor	QString	Declara el valor de una variable.

Tabla 9: Clase Variables

Nombre: Exportar		
Descripción: Clase encargada de exportar los datos descargados contenidos en el sistema.		
Tipo de Clase: Clase Controladora		
Atributo	**Tipo**	**Descripción**
Mostrar()	void	Inicializa la interfaz visual del subsistema.
ExportarDatos(QString, QString)	void	Exporta los datos seleccionados por el operador.
Cancelado()	void	SIGNAL: Emitida en caso de que el operador cancele el flujo.
Exito(QString)	void	SIGNAL: Emitida en caso de que la operación concluya satisfactoriamente.
ListaVacia(QString)	void	SIGNAL: Emitida en caso de que no existan datos para exportar.

on_pushButtonDeselAll_clicked()	void	Acción click del componente visual desmarcar todos.
on_pushButtonDesel1_clicked()	void	Acción click del componente visual desmarcar uno.
on_pushButtonSelAll_clicked()	void	Acción click del componente visual seleccionar todos.
on_pushButtonSel1_clicked()	void	Acción click del componente visual seleccionar uno.
on_pushButtonEx_clicked()	void	Acción click del componente visual exportar.
on_pushButtonCancel_clicked()	void	Acción click del componente visual cancelar.

Tabla 10: Clase Exportar

Nombre: Configurar		
Descripción: Clase encargada de realizar o modificar una configuración de descarga.		
Tipo de Clase: Clase Controladora		
Atributo	**Tipo**	**Descripción**
listaVar	QList<Variable> *	Contendrá las variables de la configuración especificada
VerifVar()	bool	Verifica que la lista de variables no esté vacía.
ElimVar(QString)	void	Elimina una variable de la lista de variables.
VerifNombre(QString)	bool	Verifica que el nombre de cada variable sea único (no esté en la lista de variables).
VerifDesp(QString)	bool	Verifica que el desplazamiento de cada variable sea único (no esté en la lista de variables).

GuardarConf(QString id, QString prot)	void	Guarda la configuración realizada.
ModificarConf(QString id, QString prot)	void	Modifica la configuración seleccionada.
Cancelado()	void	SIGNAL: Emitida en caso de que el operador cancele el flujo.
Exito(QString)	void	SIGNAL: Emitida en caso de que la operación concluya satisfactoriamente
on_pushButtonElim_clicked()	void	Acción click del componente visual Eliminar una variable seleccionada del listWidgetVars.
on_pushButtonLimpiar_clicked()	void	Acción click del componente visual elimina todos las variables del listWidgetVars.
on_pushButtonAd_clicked()	void	Acción click del componente visual adiciona una variable al listWidgetVars
on_checkBoxMod_clicked()	void	Acción click del componente visual Modificar.
on_comboBoxConf_activated(QString)	void	Acción click del componente visual, activa el comboBox con las configuraciones existentes.
on_pushButtonGuardar_clicked()	void	Acción click del componente visual Guardar.
on_pushButtonCancel_clicked()	void	Acción click del componente visual Cancelar.

Tabla 11: Clase Configurar

Nombre: Descargar		
Descripción: Clase encargada de realizar la descarga de datos.		
Tipo de Clase: Controladora		
Atributo	**Tipo**	**Descripción**
variables	QList<Variable> *	Lista de variables a descargar.
metro	QString	Identificador del dispositivo.
protocolo	QString	Protocolo a utilizar en la descarga
configuración	QString	Configuración de descarga.
DesDatos()	void	Obtener los valores de las variables especificadas en la lista de variables.
ModificarMetro(QString m)	bool	Establecer el nombre del dispositivo.
CargarProtocolo()	QStringList*	Establecer el protocolo de comunicación.
CargarConfig(QString c)	void	Establecer la configuración de descarga a utilizar.
Cancelado()	void	SIGNAL: Emitida en caso de que el operador cancele el flujo.
Exito(QString)	void	SIGNAL: Emitida en caso de que la operación concluya satisfactoriamente
on_pushButtonSig_clicked()	void	Acción click del componente visual Siguiente.
on_pushButtonDes_clicked()	void	Acción click del componente visual Descargar.
on_pushButtonCan_clicked()	void	Acción click del componente visual Cancelar.

Tabla 12: Clase Descargar

Nombre: Almacenador		
Descripción: Clase encargada de almacenar los juegos de datos descargados en la aplicación.		
Tipo de Clase: Controladora		
Atributo	**Tipo**	**Descripción**
AdDispositivo(QString metro, QString prot)	static void	Añadir un dispositivo.
AdVariable(QString metro, QString var, int desp, QString tipo, QString valor="")	static void	Añadir una variable a un dispositivo.
ModificarValor(QString metro, int desp, QVariant valor)	static void	Modificar el valor de una variable dado un dispositivo determinado.
ListaDesp(QString metro)	static QList<int>*	Devuelve la lista de desplazamientos de todas las variables de un dispositivo.
ElimDispositivo(QString metro)	static void	Elimina un dispositivo.
ElimVariable(QString metro, int desp)	static void	Elimina una variable de un dispositivo determinado dado un desplazamiento.
ListaMetros(QString protocolo = "")	static QStringList*	Devuelve la lista de todos los dispositivos que hayan sido descargados con un protocolo determinado.
ResConsulta(QString con)	static QSqlQuery	Devuelve el resultado de una consulta a la base de datos.
ListaVars(QString metro)	static QList<Variable>*	Devuelve una Lista de variables descargadas de un dispositivo.
Protocolo(QString metro)	static QString	Devuelve el protocolo con que fue descargado un dispositivo.

Tabla 13: Clase Almacenador

Nombre: ManejadorXML		
Descripción: Clase encargada de crear, leer y modificar un fichero XML.		
Tipo de Clase: Controladora		
Atributo	**Tipo**	**Descripción**
CrearArchivoConfig(QString a, QString p, QList<Variable> *var)	static void	Crea un archivo XML de configuración.
CrearArchivoExportar(QString a, QString p, QList<Variable> *var)	static void	Crea un archivo XML con los datos exportados.
LeerConfig(QString conf)	static QList<Variable>*	Lee un archivo XML de configuración.
ElimArchivoConf(QString a)	static void	Elimina un archivo XML de configuración.
ElimArchivoExp(QString a)	static void	Elimina un archivo XML con datos exportados.
AdVar(QDomDocument &d, QDomElement &r, Variable)	static void	Adiciona una variable a un archive XML dado.

Tabla 14: Clase ManejadorXML

Nombre: Visualizar		
Descripción: Clase encargada de mostrarle al operador un juego de datos contenidos en la aplicación		
Tipo de Clase: Interfaz		
Atributo	**Tipo**	**Descripción**
Mostrar()	void	Inicializa la interfaz visual del subsistema.
Cancelado()	void	SIGNAL: Emitida en caso de que el operador cancele el flujo.
ListaVacia(QString)	void	
on_comboBox_activated(QString)	void	Acción click del componente visual, activa ComboBox con las

		identificadores de los metros existentes.
on_pushButtonCerrar_clicked()	void	Acción click del componente visual Cancelar.

Tabla 15: Clase Visualizar

Nombre: InterDriver		
Descripción: Clase que brinda una interfaz al subsistema de driver para el resto de la aplicación.		
Tipo de Clase: Interfaz		
Atributo	**Tipo**	**Descripción**
Valor(int desp)	QString	Obtiene el valor una variable.
Verificar(int pComm, int veloc, QString par, int fByte, int stopB, int esc, int func)	bool	Verifica que el dispositivo esté conectado y se pueda establecer la comunicación con un juego de parámetros determinado.
Error()	QString	Devuelve un error que haya ocurrido en el subsistema.

Tabla 16: Clase InterDriver

Nombre: WinCEPuertoSerie		
Descripción: Clase encargada de abrir, cerrar, leer y escribir del puerto serie.		
Tipo de Clase: Controladora		
Atributo	**Tipo**	**Descripción**
IniciarPuerto(LPTSTR lptszNombre)	bool	Inicia la conexión con el Puerto serie.
CerrarPuerto()	bool	Cierra conexión con Puerto serie
EscribirPuerto(BYTE Byte)	bool	Escribe un byte al puerto serie
ModificarBaudios(int baudios)	bool	Modifica la velocidad de

		transmisión del Puerto serie.
AlterarBitsParada(int bits)	bool	Modifica la cantidad de bit de parada de la comunicación.
AlterarTamanoBits(int número)	bool	Modifica el formato de los bytes a transmitir por el puerto.
AlterarParidad(int paridad)	bool	Modifica la paridad de la conexión serie.
Leer()	QString	Devuelve el valor del buffer de lectura del Puerto serie.
hPuerto	HANDLE	Manejador del Puerto serie.
hHiloLectura	HANDLE	Manejador del hilo de lectura.
puertoDCB	DCB	Bloque de control de datos del Puerto serie.
dwError	DWORD	Mensaje de error del Puerto.
commTimeouts	COMMTIMEOUTS	Tiempo de espera.
buffer	Buffer*	Contenedor de la información recibida por el puerto serie.
TerminarHiloLectura()	void	Finaliza el hilo de lectura.
HiloLecturaPuerto(void* param)	static DWORD	Hilo que escucha el Puerto serie en espera de recibir información y la almacena en el buffer.

Tabla 17: Clase WinCEPuertoSerie

Nombre: Buffer		
Descripción: Clase que contiene una cantidad de bytes de información.		
Tipo de Clase: Entidad		
Atributo	**Tipo**	**Descripción**
AdElemento(unsigned char elem)	void	Adiciona un elemento al buffer.
Limpiar()	void	Vacía el buffer.
Elemento(int pos)	unsigned char	Devuelve el elemento dado una

		posición.
DameCantidad()	int	Devuelve Cantidad
buffer	unsigned char*	Arreglo de bytes
cantidad	int	Cantidad de bytes que contiene el arreglo.
cantMax	int	Cantidad máxima de bytes que contiene.

Tabla 18: Clase Buffer

Nombre: CRCModbusRTU		
Descripción: Clase encargada de construir el código de control de errores del protocolo mediante el código de redundancia cíclica.		
Tipo de Clase: Controladora		
Atributo	**Tipo**	**Descripción**
CRC16(unsigned char* trama, int longitud)	int	Devuelve el valor del código de redundancia cíclica de 16bits de una trama.
index	unsigned char	Posición relativa de la tabla de CRC.
lowCRC	unsigned char	Valor de los 8bits bajos de CRC.
highCRC	unsigned char	Valor de los 8bits altos de CRC.

Tabla 19: Clase CRCModbusRTU

Nombre: ModbusRTU,		
Descripción: Clase encargada de entramar y desentramar los mensajes del protocolo ModBus.		
Tipo de Clase: Controladora		
Atributo	**Tipo**	**Descripción**
EntramarMensaje(unsigned char dirEsclavo, unsigned char función, short dirMemoria,	unsigned char*	Construir el mensaje ModBusRTU.

short cantRegistros)		
Valor(unsigned char función, unsigned char* respuesta)	unsigned long	Dado un mensaje de respuesta del protocolo, devuelve la cantidad de bytes que representa esa respuesta.
CodigoError()	unsigned char	Devuelve el último código de error que ocurrió.
codError	unsigned char	Contiene el último código de error.

Tabla 20: Clase ModbusRTU

Nombre: Driver		
Descripción: Clase encargada de establecer la comunicación entre el sistema y el dispositivo.		
Tipo de Clase: Controladora		
Métodos	**Valor de Retorno**	**Descripción**
Conectar(int numeroP)	bool	Realiza conexión.
Conectar(int numeroP, int velocidad, QString paridad, int bitsDatos, int bitsParada)	bool	Realiza conexión con parámetros predefinidos.
Desconectar()	bool	Desconecta.
ObtenerValor(unsigned short direccion, unsigned short cantReg)	bool	Retorna verdadero en caso de la comunicación se haya realizado satisfactoriamente y almacena el resultado en valor en caso contrario retorna falso.
EstFuncion(int fu)	void	Establece la función de comunicación.
EstEsclavo(int e)	void	Establece el identificador del dispositivo.
EstVelocidad(int velocidad)	bool	Establece la velocidad.
EstParidad(QString paridad)	bool	Establece la paridad.
EstBitsDatos(int cant)	bool	Establece los bits de datos

EstBitsParada(int cant)	bool	Establece los bits de parada.
EstTimeOut(int t)	void	Establece tiempo de espera.
Error()	QString	Devuelve el último error.
UltimoValor()	unsigned long	Devuelve último valor.
valor()	unsigned long	Devuelve último valor.
Atributos	**Tipo**	**Descripción**
función	unsigned char	Define que registros se van a leer del dispositivo.
esclavo	unsigned char	Identificador del dispositivo.
timeOut	int	Tiempo de espera en milisegundos.
reintentos	unsigned char	Cantidad de reintentos en caso de fallos.

Tabla 21: Clase EmbedbeDriver

3.5 Diagrama de secuencia.

En el diagrama de secuencia se muestra la interacción de los objetos que componen un sistema de forma temporal.

3.5.1 Diagrama de Secuencia CU "Configurar Descarga" (escenario: Configurar).

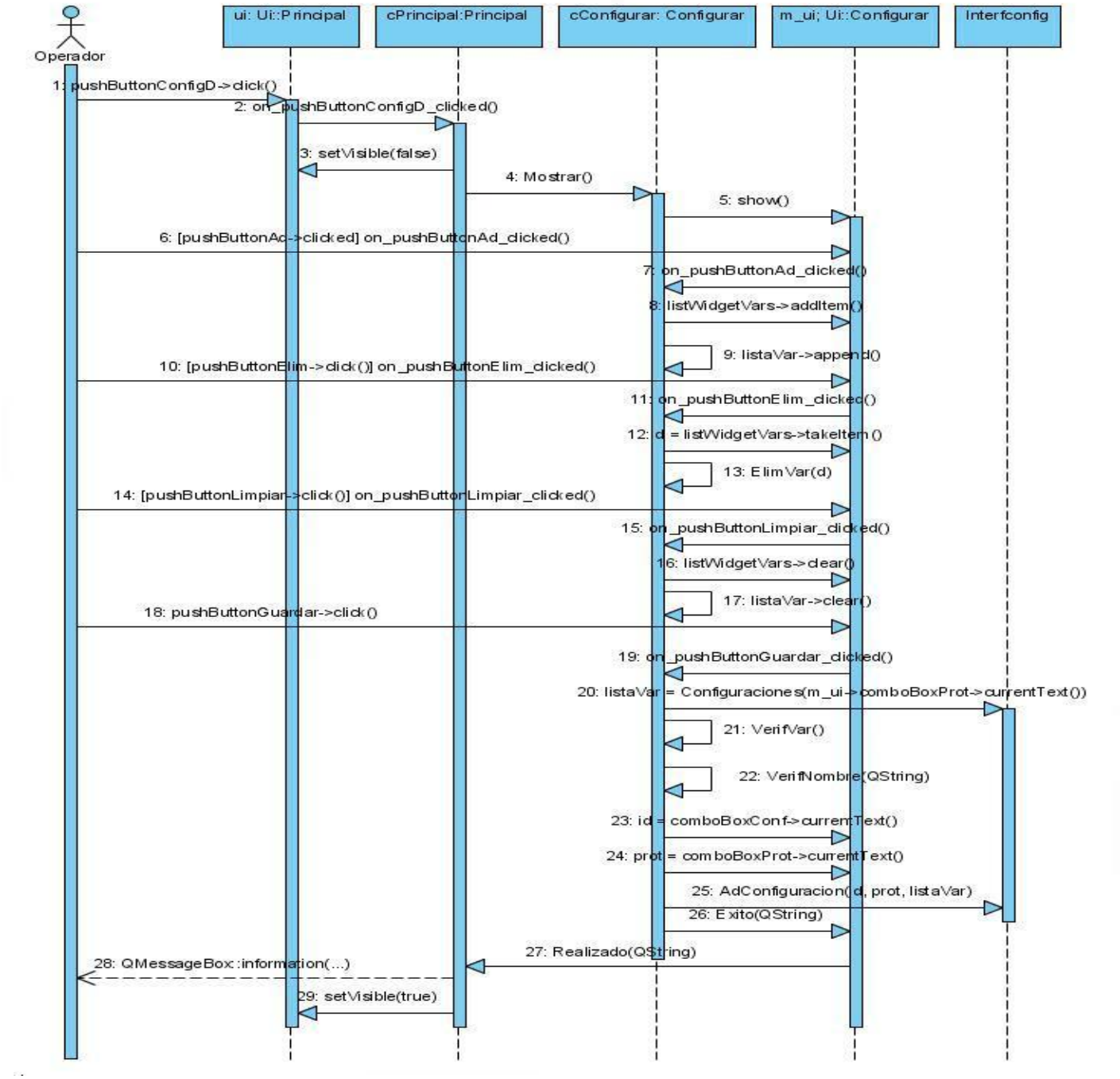


Fig.5: Diagrama de Secuencia CU " Configurar Descarga" (escenario: Configurar).

3.5.2 Diagrama de Secuencia CU " Configurar Descarga" (escenario: Modificar).

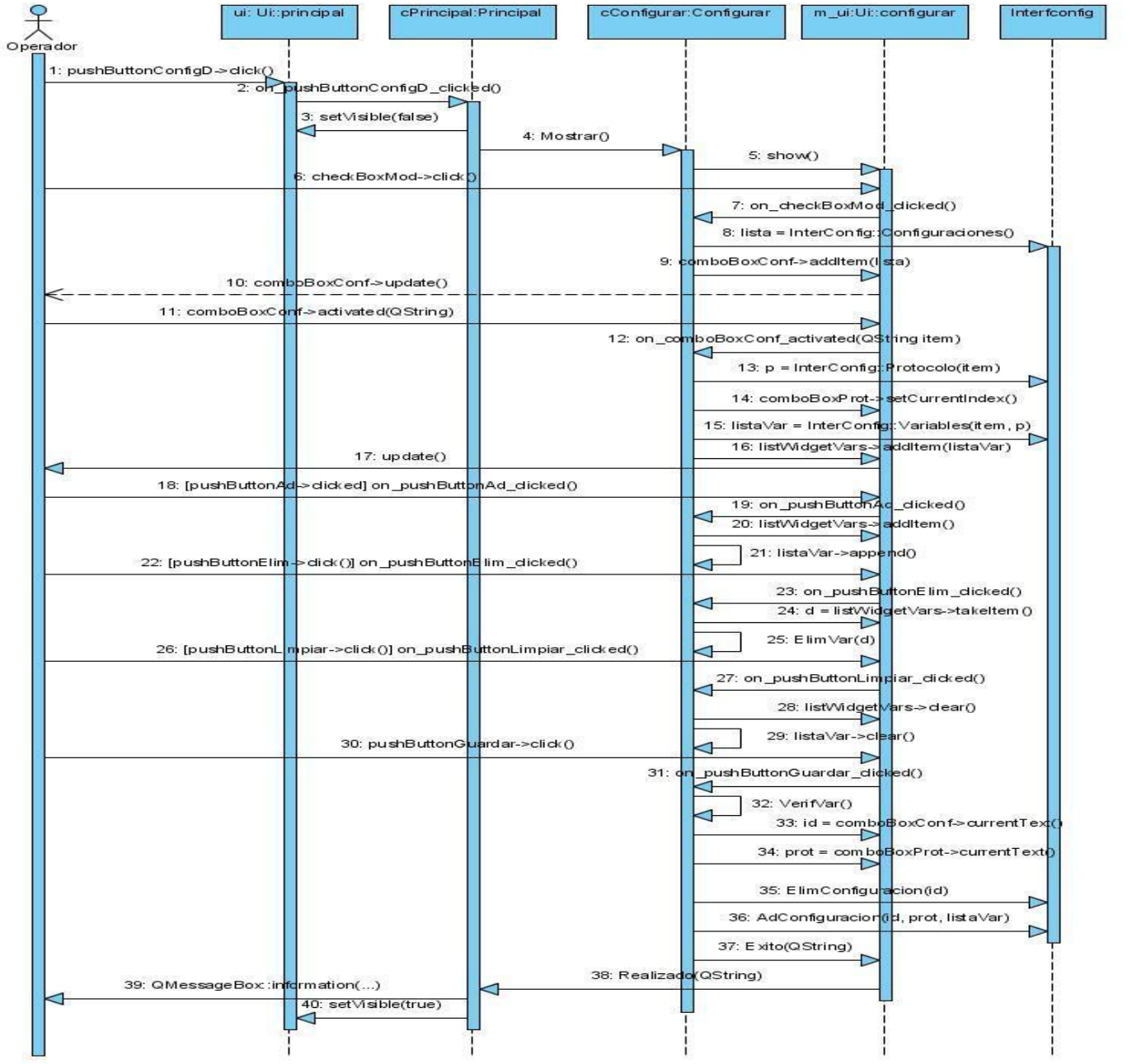


Fig.6: Diagrama de Secuencia CU " Configurar Descarga" (escenario: Modificar).

3.5.3 Diagrama de Secuencia CU " Configurar Descarga" (escenario: Eliminar).

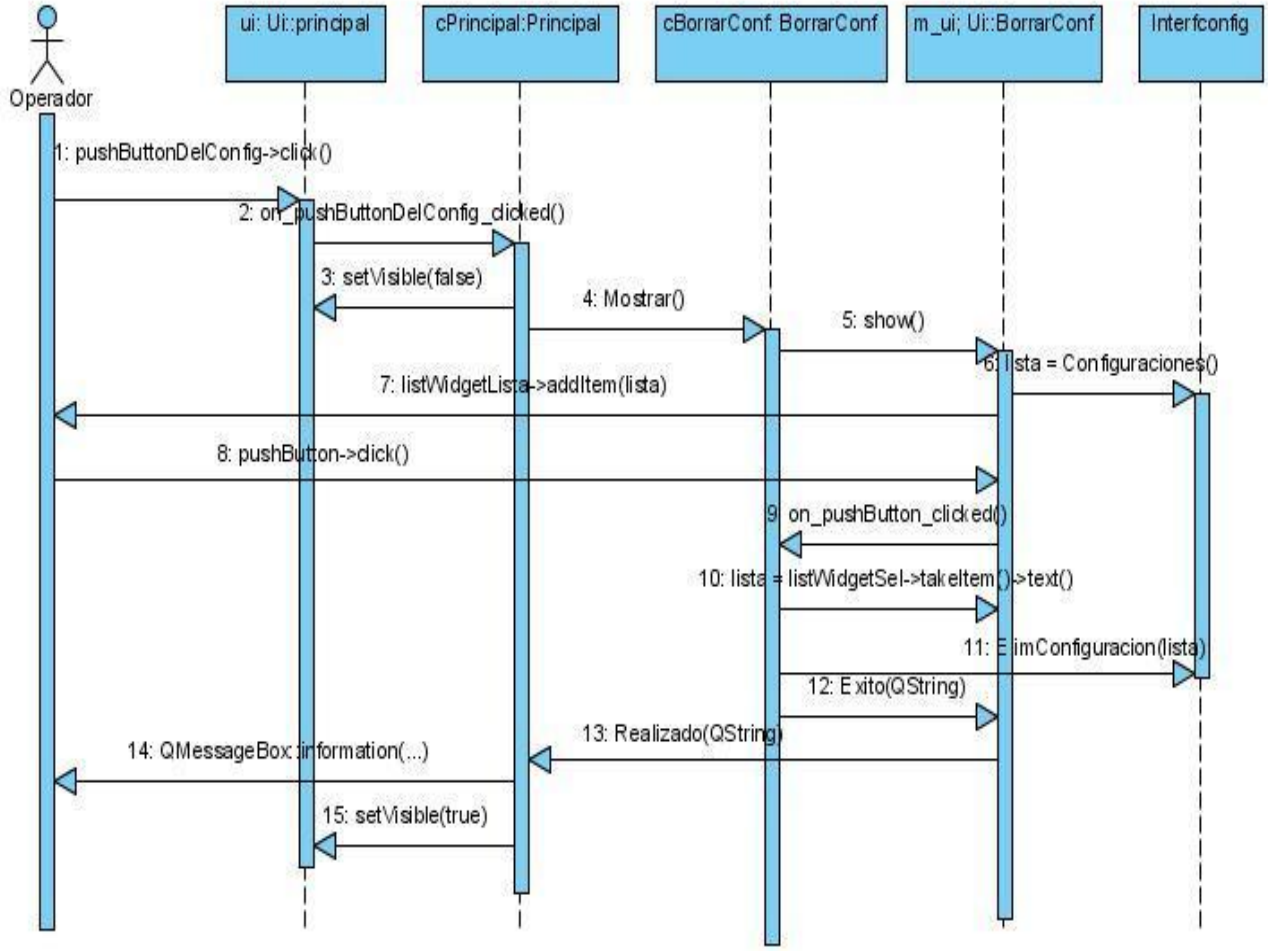


Fig.7: Diagrama de Secuencia CU " Configurar Descarga" (escenario: Eliminar).

3.5.4 Diagrama de Secuencia CU Exportar Datos.

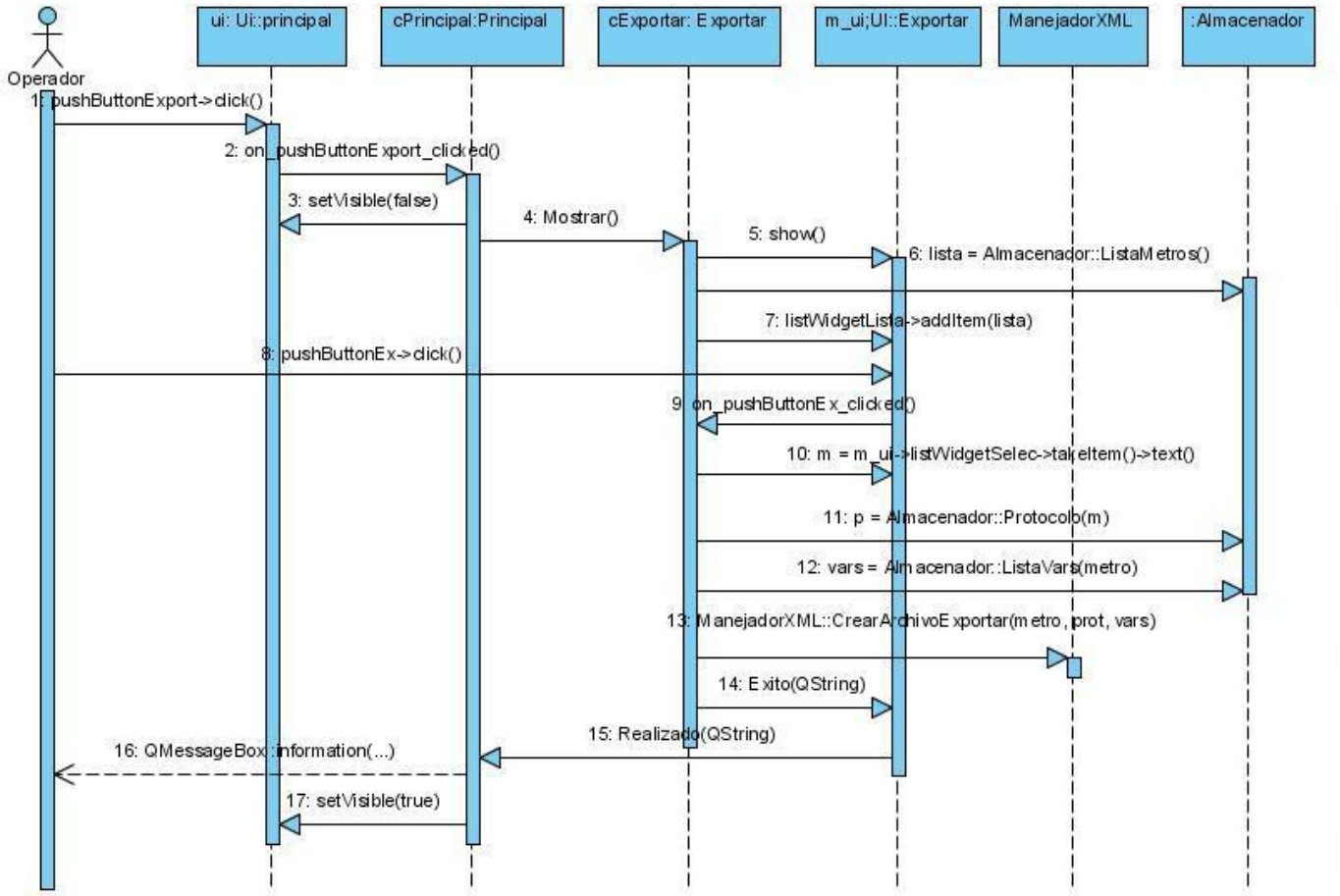


Fig.8: Diagrama de Secuencia CU Exportar Datos.

3.5.5 Diagrama Secuencia "Descargar Datos" (escenario Descargar).

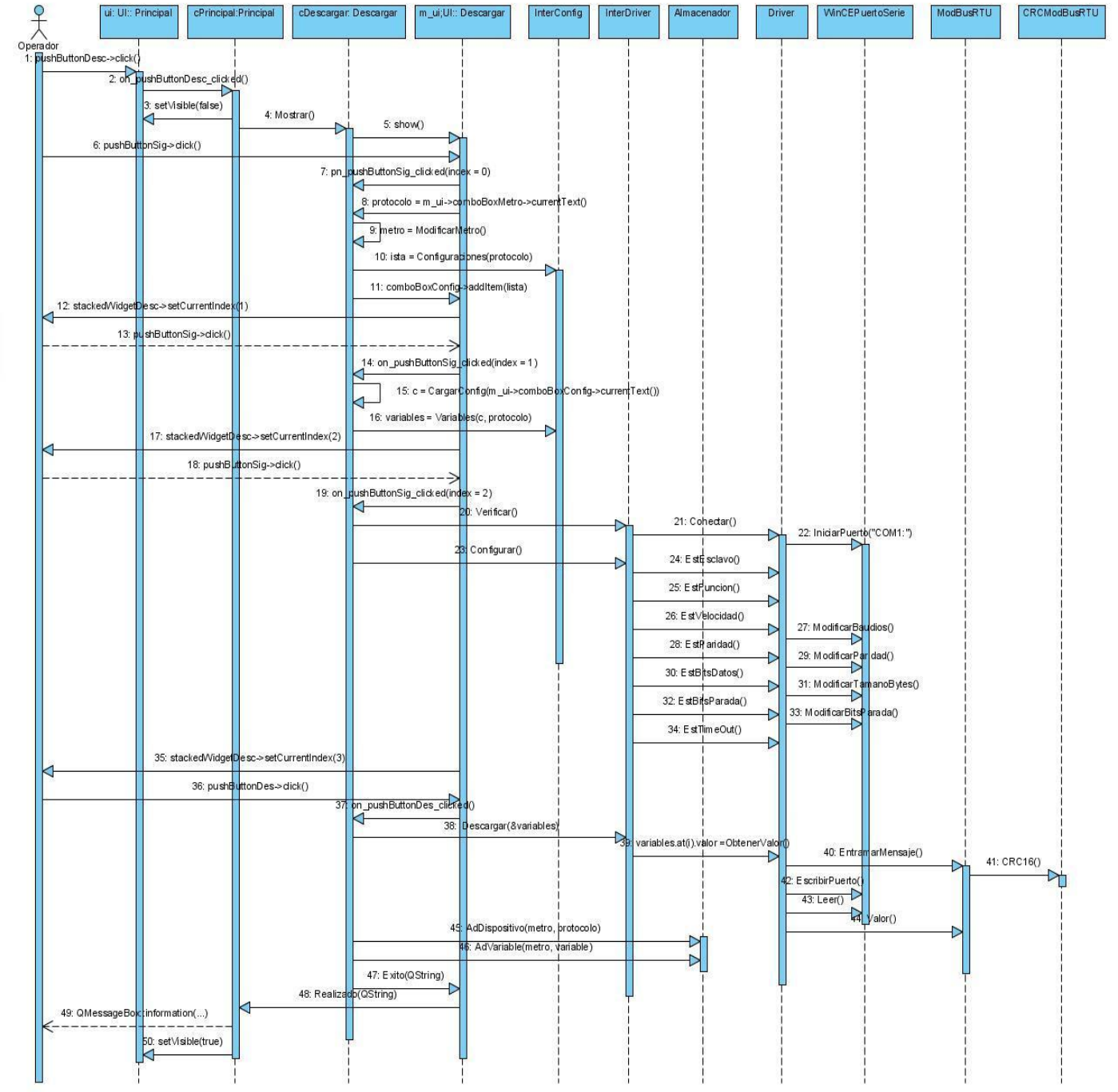


Fig.9: Diagrama de Secuencia “Descargar Datos” (escenario Descargar).

3.5.6 Diagrama Secuencia " Descargar Datos" (escenario Eliminar).

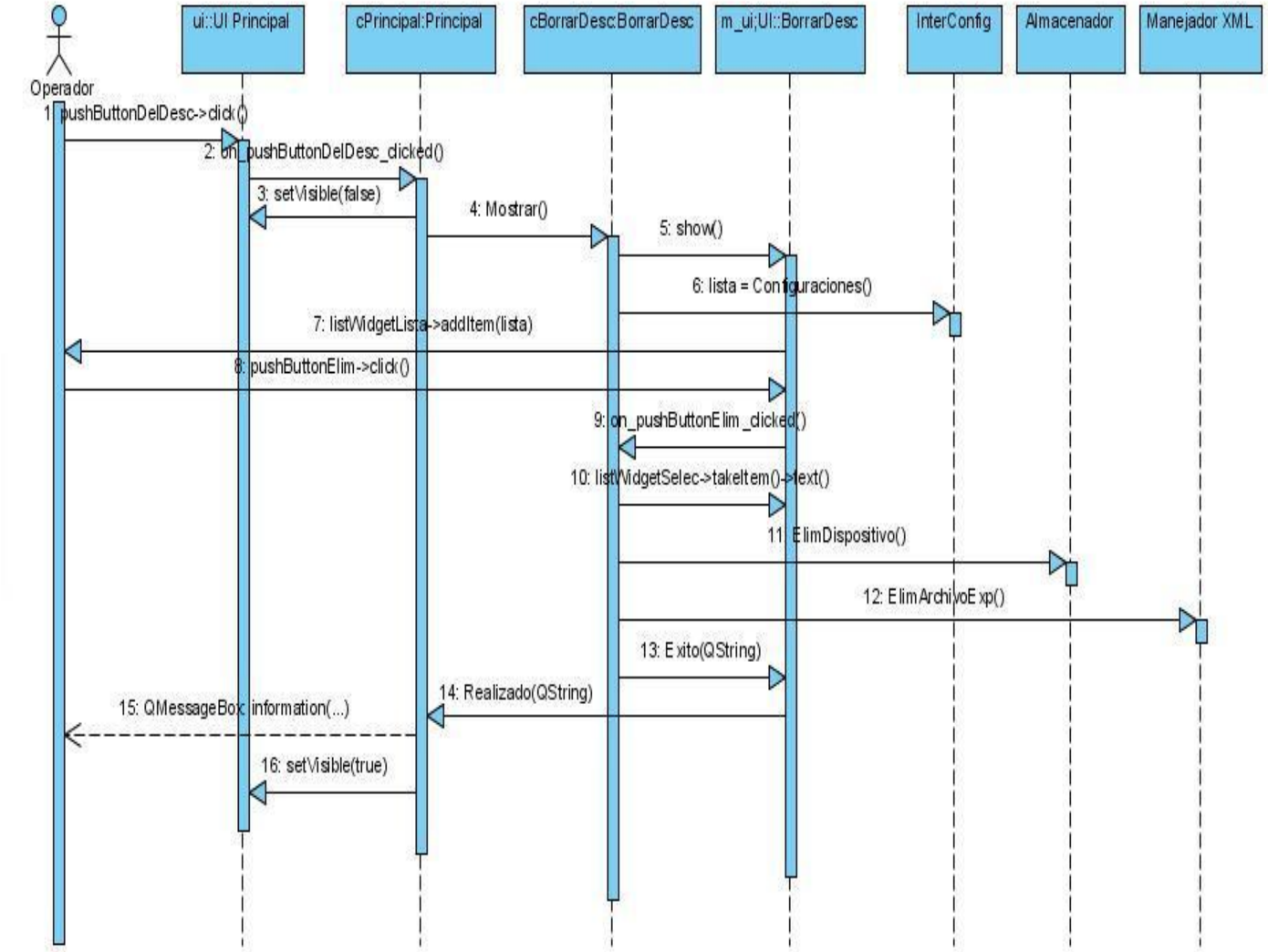


Fig.10: Diagrama Secuencia " Descargar Datos" (escenario Eliminar).

3.5.7 Diagrama Secuencia "Visualización".

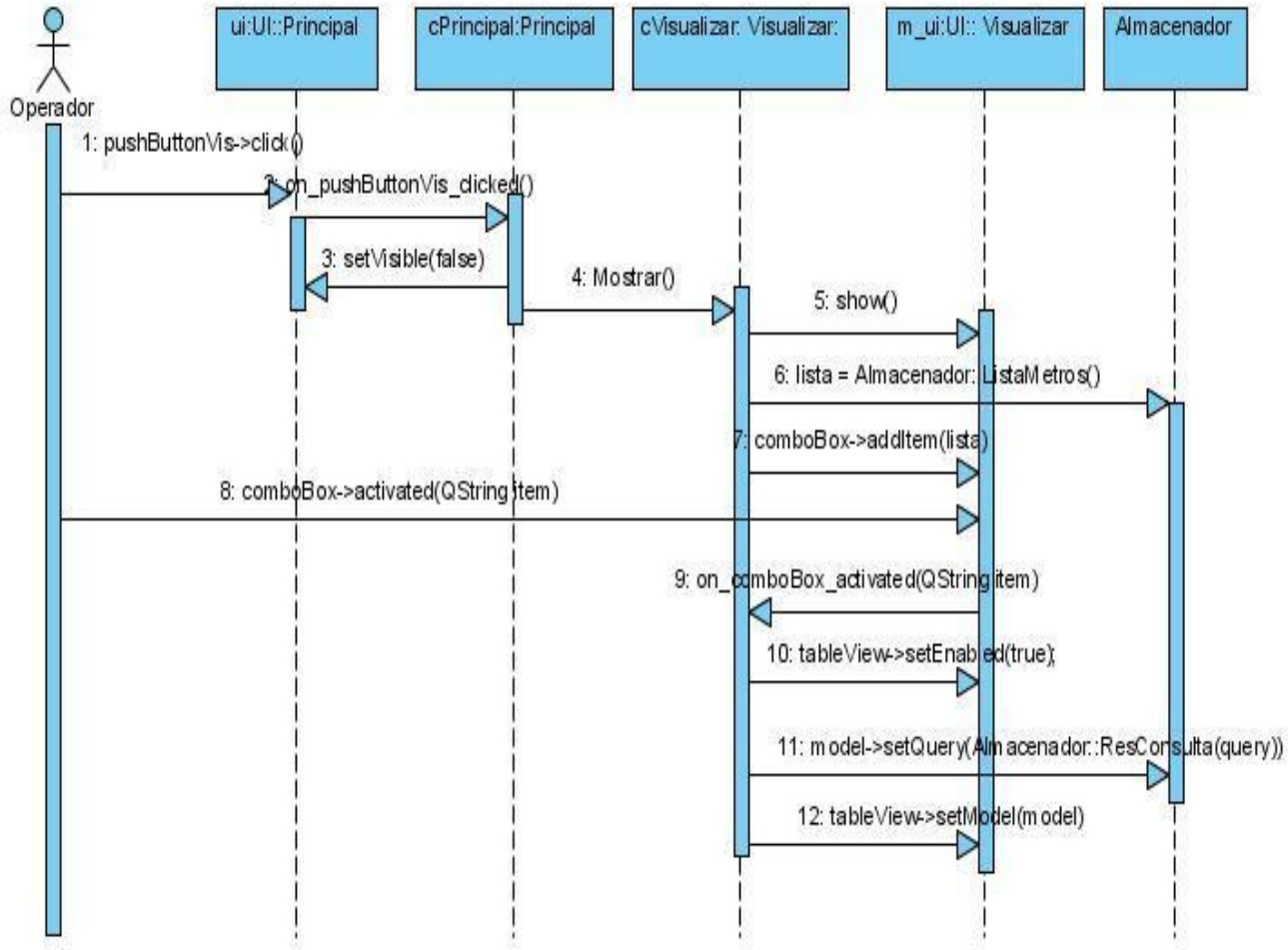


Fig.11: Diagrama Secuencia "Visualización".

3.6 Diagrama de despliegue.

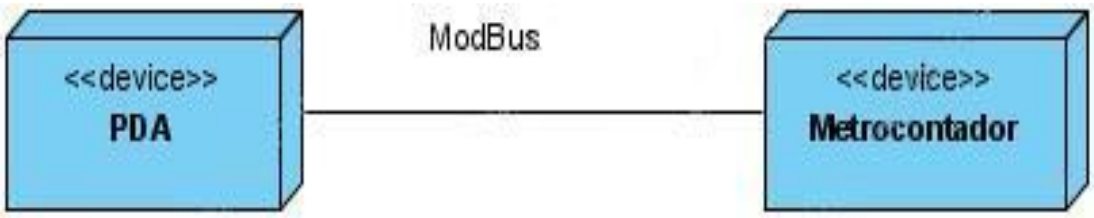


Fig.12: Diagrama de Despliegue.

3.7 Diagrama de componentes.

Este diagrama refleja la creación de componentes físicos, que se traducen en ficheros .h, ficheros .cpp y los .ui, correspondiente a la implementación en C++.

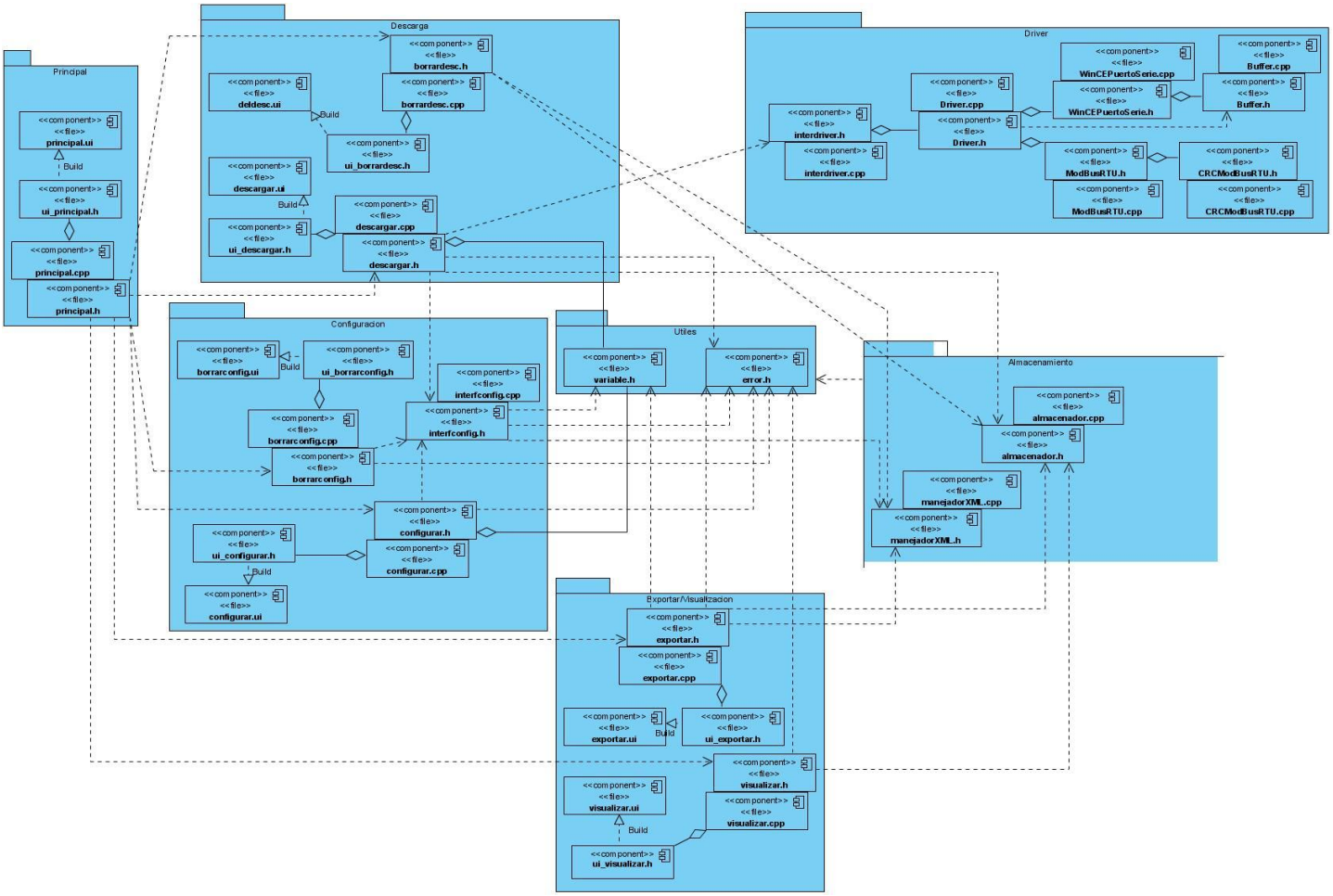


Fig.13: Diagrama de Componentes.

CONCLUSIONES DEL CAPÍTULO.

Este capítulo realizó la solución general de la aplicación. Analizó los casos de usos críticos, definiendo las clases del análisis para cada uno, así como la organización de los módulos mediante un diagrama de paquetes. En el diseño se esbozan todas las clases para cada uno de los objetos determinados y se propone un modelo de datos que sustente las clases desarrolladas. Se modelaron los diagramas de secuencia para cada uno de los diferentes escenarios de los casos de uso viendo como seria la relación entre el operador y el sistema.

CAPÍTULO IV. PRUEBAS DEL SISTEMA.

Introducción

Este capítulo lleva a cabo el proceso de pruebas del software para la detección y corrección de los errores que pueda presentar la aplicación con el objetivo de darles solución y respuesta a los mismos antes de dar por terminada la aplicación, la pruebas no excluyen la aparición de defectos en el software pero permiten desarrollar una serie de técnicas y métodos para la solución de los posibles problemas que puedan surgir.

Como parte que es de un proceso industrial, la fase de pruebas añade valor al producto que se maneja: todos los programas tienen errores y la fase de pruebas los descubre; ese es el valor que añade.

4.1 Plan de Pruebas

Para tener una buena aplicación es bueno tener un plan de pruebas. Los planes de prueba deben tener las siguientes características:

- **Precisos**: Las pruebas deben describir correctamente lo que verificarán.
- **Económicos**: Sólo ha de contener los pasos necesarios para su propósito.
- **Repetible**: Las pruebas deben ser consistentes y recoger todo lo necesario para que los resultados de cualquier ejecución sean los mismos.
- **Adecuado**: La prueba debe describir la situación en la que es aplicable.
- **Trazable**: La prueba debe estar relacionada con el requisito funcional que cubre con objeto de facilitar la identificación del fallo y permitir hacer un seguimiento de las correcciones más fácilmente.

4.1.1 Nivel de Prueba: Prueba de Sistema

Las pruebas de sistema verifican el correcto funcionamiento del sistema completo incluyendo casos de prueba que busquen los fallos del sistema. Son pruebas destructivas y persiguen demostrar la robustez del sistema aun en condiciones adversas. Verifican requisitos funcionales y no funcionales. [24]

4.1.2 Tipo de Prueba: Funcionalidad

4.1.3 Especificaciones de Software y Hardware

- **Hardware:**
  - ✓ PC cliente con 1.0 giga byte de memoria RAM, microprocesador Intel(R) Core(TM)2 Duo CPU, E4500 con velocidad de 2.20 Ghz, placa madre Intel Red alámbrica.

- **Software**
  - ✓ Sistema Operativo WindowsXP. Servipack2.
  - ✓ El IDE para desarrollo de la aplicación Visual Studio 2005.
  - ✓ El IDE de desarrollo para la interfaz visual del sistema será QtCreator.
  - ✓ Instalado ActiveSync 4.0 o superior.
  - ✓ Instalado Windows Mobile 5.0 Pocket Pc SDK.
  - ✓ Librería de Qt 4.5.0 para dispositivos embebidos con el plugin de SQLite instalado.

4.1.4 Método de Prueba: Caja Negra

Pruebas de caja Negra

La prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. O sea, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene. [25]

Objetivos

El objetivo de realizar este tipo de prueba al sistema es para revelar el incorrecto o incompleto funcionamiento de este, así como los errores de interfaz, rendimiento y errores de inicialización y terminación.

Alcance

El proceso de pruebas de caja negra se va a centrar principalmente en los requisitos funcionales del software para verificar el comportamiento de la unidad observable externamente y la calidad funcional.

Descripción

Se llevan a cabo sobre la interfaz del software, y es completamente indiferente el comportamiento interno y la estructura del programa. Los casos de prueba de la caja negra pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma adecuada.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

La prueba de la caja negra intenta encontrar errores de las siguientes categorías:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales del programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos.

4.2 Diseño de Casos de Prueba

El diseño de casos de prueba tiene como objetivo verificar que las funcionalidades de la aplicación cumplen su propósito con los resultados esperados.

4.2.1 Caso de Uso Configurar Descarga

Descripción General

Mediante este caso de uso el usuario podrá realizar diversas operaciones como crear, modificar y eliminar una descarga.

Pruebas a realizar:

- Crear Configuración de Descarga.
- Eliminar Configuración de Descarga.
- Modificar Configuración de Descarga.

CPR #1 Crear Configuración de Descarga.

Breve Descripción

Este caso de prueba le permite al operador crear una configuración de descarga.

Flujo Central

1. El operador selecciona la operación Configurar Descargas.
2. El sistema le brinda la posibilidad de configurar los datos: Identificador y Protocolo y Variables (insertando su nombre, dirección física y tipo de dato que representa) ó seleccionar la opción de Modificar una Configuración existente.
3. El operador configura los diferentes campos para crear la configuración y presiona el botón "Guardar y Salir".
4. El sistema almacena la configuración de datos y lo notifica al operador mostrándole un mensaje.

Iteraciones.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Teniendo abierta la ventana para configurar descarga, el operador introduce los datos		El sistema debe configurar la descarga y salir a la interfaz padre.	El sistema configura la descarga, muestra el mensaje "Configuración Realizada" y sale a la interfaz padre.	

necesarios para crear la configuración (id_metro, protocolo y variables) y oprime el botón **Guardar y Salir**.				
	Teniendo abierta la ventana para configurar descarga, el operador omite alguno de los datos necesarios para crear la configuración (id_metro, protocolo y variables) y oprime el botón **Guardar y Salir**.	El sistema debe mostrar un mensaje de alerta pidiendo al operador que teclee el dato omitido.	El sistema muestra un mensaje de alerta pidiendo al operador que teclee el dato omitido y muestra la ventana para realizar la operación.	
	Teniendo abierta la ventana para configurar descarga, el operador teclea un id_metro con el nombre de uno ya existente.	El sistema debe mostrar un mensaje alertando que ya existe un metro con ese nombre.	El sistema muestra el mensaje “Ese identificador ya está en uso”.	
Teniendo abierta la ventana para configurar descarga, el operador oprime el botón **Cancelar**.		El sistema debe retornar a la interfaz padre.	El sistema retorna a la interfaz padre.	

Tabla 22: Caso de Prueba #1

CPR #2 Modificar Configuración de Descarga.

Breve Descripción

Este caso de prueba le permite al operador modificar una configuración de descarga a partir de una que esta creada previamente.

Flujo Central

1. El operador selecciona la opción Configurar Descarga.
2. El sistema muestra una nueva ventana con las opciones para crear la Configuración de Descarga y la opción de Modificar Configuración.
3. El operador selecciona la pestaña de Modificar Descarga.
4. El sistema brinda las diferentes Configuraciones de Descarga existentes.
5. El operador elige la Configuración a Modificar.
6. El sistema brinda los datos de dicha configuración.
7. El operador modifica los datos y oprime el botón “Guardar y Salir”.
8. El sistema almacena la configuración de datos y lo notifica al operador mostrándole un mensaje.

Iteraciones.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Teniendo abierta la ventana para modificar una configuración, el operador selecciona la configuración a modificar y modifica los datos		El sistema debe modificar la configuración y salir a la interfaz padre.	El sistema modifica la configuración, muestra el mensaje “Configuración Modificada” y sale a la interfaz padre.	

necesarios para crear la configuración (variables) y oprime el botón **Guardar y Salir**.				
	Teniendo abierta la ventana para modificar una configuración, el operador selecciona la configuración a modificar y omite algunos de los datos necesarios para modificar la configuración (id_metro, protocolo y variables) y oprime el botón **Guardar y Salir**.	El sistema debe mostrar un mensaje de alerta pidiendo al operador que teclee el dato omitido.	El sistema muestra el mensaje de alerta pidiendo al operador que teclee el dato omitido.	
Teniendo abierta la ventana para		El sistema debe retornar a la interfaz padre.	El sistema retorna a la interfaz padre.	

modificar una configuración, el operador oprime el botón **Cancelar**.				

Tabla 23: Caso de Prueba #2

CPR 3 Eliminar Configuración de Descarga.

Breve Descripción

Este caso de prueba le permite al operador eliminar una configuración de descarga a partir de una que esta creada previamente.

Flujo Central

1. El operador tiene la posibilidad de seleccionar la operación que desea ejecutar ya sea Configurar Descarga ó Eliminar una Configuración existente.
2. El operador selecciona la opción de Eliminar Configuración.
3. El sistema muestra un formulario para seleccionar la(s) Configuraciones a eliminar.
4. El operador selecciona la(s) Configuraciones a Eliminar y presiona el botón **"Eliminar y Salir".**
5. El sistema elimina las Configuraciones y lo notifica al Operador mostrándole un mensaje.

Iteraciones.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Teniendo abierta la ventana para eliminar una configuración, el operador selecciona la		El sistema debe eliminar la configuración y salir a la interfaz padre.	El sistema elimina la configuración seleccionada y muestra el mensaje "Configuraciones Eliminadas".	

configuración a eliminar y oprime el botón **Eliminar y Salir.**				
Teniendo abierta la ventana para eliminar una configuración, el operador selecciona la configuración a eliminar y oprime el botón **Cancelar**.		El sistema debe retornar a la interfaz padre.	El sistema retorna a la interfaz padre.	

Tabla 24: Caso de Prueba #3

4.2.2 Caso de Uso: Descargar Datos.

Descripción General

Mediante este caso de uso el usuario podrá realizar diversas operaciones como crear, modificar y eliminar una descarga.

Pruebas a realizar:

- Realizar Descarga de Datos.
- Eliminar Descarga de Datos.

CPR 4 Realizar Descarga de Datos.

Breve Descripción

Este caso de prueba le permite al operador Descargar los datos del metro-contador hacia la PDA de forma automática.

Flujo Central

1. El operador selecciona la opción de Descargar Datos hacia la PDA.
2. El sistema muestra un formulario para insertar los datos generales del metro-contador.
3. El operador inserta los datos generales del metro-contador: identificador, tipo de metro-contador.
4. El sistema pide que se escoja la configuración de descarga a utilizar.
5. El operador selecciona la configuración de descarga correspondiente y presiona el botón "Descargar".
6. El sistema realiza la Descarga Automática de los datos.
7. El sistema almacena los datos en el PDA y lo notifica al Operador mostrándole un mensaje.

Condiciones de Ejecución:

Debe de existir la conexión entre el metro-contador y la PDA por el puerto serie RS232.

Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Teniendo abierta la interfaz para descargar datos, el operador introduce los datos necesarios para descargar los datos (Dispositivo, Protocolo), selecciona la configuración de descarga, la configuración de conexión, y oprime el botón **Descargar.**		El sistema debe descargar los datos y salir a la interfaz padre.		

	Teniendo abierta la interfaz para descargar datos, el operador introduce datos incorrectos en la configuración para la descarga de los datos y oprime el botón **Descargar.**	El sistema debe mostrar un mensaje notificando el error.	El sistema muestra el mensaje y no descarga los datos.	
	Teniendo abierta la interfaz para descargar datos, el operador omite el nombre del dispositivo y oprime el botón **Siguiente.**	El sistema debe mostrar un mensaje de alerta pidiendo al operador que teclee el dato omitido.	El sistema muestra un mensaje de alerta, pidiendo que se teclee el mismo.	
Teniendo abierta la interfaz para descargar datos, el operador oprime el botón **Cancelar**.		El sistema debe retornar a la interfaz padre.	El sistema retorna a la interfaz padre.	

Tabla 25: Caso de Prueba #4

CP 5 Eliminar Descarga de Datos.

Breve Descripción

Este caso de prueba le permite al operador Eliminar una Descargar de datos.

Flujo Central

1. El Operador selecciona la opción Eliminar Descargar.
2. El sistema muestra un formulario para seleccionar la(s) descargas a eliminar.
3. El Operador selecciona la(s) descargas a eliminar y presiona el botón “Eliminar y Salir”.
4. El sistema elimina las Descargas del PDA y lo notifica al Operador mostrándole un mensaje.

Condiciones de Ejecución:

Debe de existir la conexión entre el metrocontador y la PDA por el puerto serie RS232.

Iteraciones.

Clases Válidas	**Clases Inválidas**	**Resultado Esperado**	**Resultado de la Prueba**	**Observaciones**
Teniendo abierta la interfaz para borrar descargas, el operador selecciona la descarga a eliminar y oprime el botón **Eliminar y Salir**.		El sistema debe eliminar la descarga y salir a la interfaz padre.	El sistema elimina la descarga y sale a la interfaz padre.	
Teniendo abierta la interfaz para borrar descargas, el operador oprime el botón **Cancelar.**		El sistema debe retornar a la interfaz padre.	El sistema retorna a la interfaz padre.	

Tabla 26: Caso de Prueba #5

4.2.3 Caso de Uso: Exportar Datos.

Descripción General

Mediante este caso de uso el operador decide exportar los datos del PDA hacia la computadora.

Pruebas a realizar:

- Exportar Datos.

CPR 6 Exportar Datos.

Breve Descripción

Este caso de prueba le permite al operador exportar los datos del PDA hacia la computadora.

Flujo Central

1. El Operador decide exportar los datos descargados en el PDA, pulsando la opción “Exportar Datos”.
2. El sistema muestra una nueva ventana con las opciones de datos para Exportar.
3. Selecciona los datos a exportar y presiona el botón “Exportar y Salir”.
4. El sistema exporta los datos y lo notifica al Operador mostrándole un mensaje.

Condiciones de Ejecución:

El PDA debe de tener almacenado los datos descargados de algún metrocontador.

Iteraciones.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Teniendo abierta la interfaz para exportar datos, el operador selecciona lo(s) datos a exportar y oprime el botón **Exportar y Salir.**		El sistema debe exportar los datos y salir a la interfaz padre.	El sistema exporta los datos y sale a la interfaz padre.	
Teniendo abierta la interfaz para exportar datos, el operador oprime el botón **Cancelar.**		El sistema debe retornar a la interfaz padre.	El sistema retorna a la interfaz padre.	
	Teniendo abierta la interfaz para exportar datos, el operador selecciona datos que ya están	El sistema debe mostrar un mensaje notificando que el archivo ya.	El sistema muestra un mensaje notificando que el archivo ya existe y da la posibilidad de sobrescribirlo o no.	

	exportados.			

Tabla 27: Caso de Prueba #6

4.2.4 Caso de Uso: Visualizar Descarga.

Descripción General

Este caso de uso le permite al Operador visualizar los datos de una descarga almacenada en el PDA.

Pruebas a realizar:

- Visualizar Descarga.

CPR 7: Visualizar Descarga.

Breve Descripción
Este caso de prueba le permite al Operador visualizar los datos de una descarga almacenada en el PDA.

Flujo Central:

- El operador decide visualizar los datos descargados en el PDA, pulsando la opción Visualizar.
- El sistema muestra las descargas realizadas.
- El operador decide cual descarga desea visualizar.
- El sistema muestra los valores y atributos de cada una de las variables descargadas.

Condiciones de Ejecución:

El PDA debe de tener almacenado los datos descargados de algún metro-contador.

Iteraciones.

Clases Válidas	Clases Inválida	Resultado Esperado	Resultado de la Prueba	Observaciones
Teniendo abierta la ventana para visualizar descargas el operador escoge la descarga a visualizar.		El sistema debe mostrar los valores de la descarga seleccionada.	El sistema muestra los valores de la descarga seleccionada.	

Teniendo abierta la ventana para visualizar descargas el operador presiona el botón **Cerrar**.		El sistema debe retornar a la interfaz padre.	El sistema retorna a la interfaz padre.	

Tabla 28: Caso de Prueba #7

CONCLUSIONES DEL CAPÍTULO

Este capítulo intenta poner al sistema en condiciones óptimas realizándole diversas pruebas para optimizar la respuesta del mismo. El objetivo del capítulo es encontrar los errores o deficiencias que tenga la aplicación para darles solución. Estas pruebas fueron realizadas y avaladas por el grupo de calidad del polo de automática de la Facultad 5.

CONCLUSIONES GENERALES

Dando cumplimiento al objetivo de este proyecto, y a los requisitos del cliente, se obtuvo el módulo de Adquisición de variables eléctricas mediante dispositivos PDA realizado para la *UNE,* el cual brinda las funcionalidades básicas de descarga de datos, la exportación de datos y configuración de descargas. Con esto se optimiza el proceso de adquisición de variables eléctricas el cual resultaba trabajoso y engorroso. Otro resultado que se obtuvo con la realización de este trabajo fue la incursión en una serie de tecnologías que hacen posible la realización de aplicaciones multiplataforma para dispositivos móviles.

RECOMENDACIONES

Todo trabajo desarrollado está sujeto al cambio en pro de mejoras, por lo que se recomienda:

- Continuar con el estudio del sistema con el objetivo de añadir nuevas funcionalidades que contribuyan con el perfeccionamiento del servicio.
- Probar la aplicación en un dispositivo físico.
- Continuar con la elaboración de la capa de transporte del driver para que la aplicación llegue a soportar otros sistemas operativos que no sea solamente Windows CE.

BIBLIOGRAFÍA

[Online] http://static.scribd.com/docs/8wmxgmv38a49w.swf?INITIAL_VIEW=width.

ArqHys. 2004. ArqHys Tecnologia. [Online] 2004. http://tecnologia.arqhys.com/pda/index.html.

Cardenas, Alfonso Araujo. 2004. http://mx.geocities.com. *Protocolos de Comunicación.* [Online] junio 2004. http://mx.geocities.com/alfonsoaraujocardenas/protocolos.html.

CIRCUITOR. 2005. *CONTADOR TRIFÁSICO MULTIFUNCIONAL Serie CIRWATT.* 2005.

CIRCUTOR. Contadores Multifuncional de Energia Electrica. [Online] http://www.circutor.com/novedades/cirwattc/Q1_01_E.pdf.

Clikear.com. http://www.clikear.com. *Desarrollo Orientado a Objetos con UML.* [Online] http://www.clikear.com/manuales/uml/index.aspx.

Eclipse. 2009. OpenUP. [Online] 2009. http://epf.eclipse.org/wikis/openup/.

Fdez-Baillo, David Carrero. 2008. Carrero.es. *Sistemas operativos para dispositivos móviles.* [Online] Agosto 12, 2008. http://carrero.es/sistemas-operativos-para-dispositivos-moviles/2059.

Genbeta. http://www.genbeta.com. [Online] http://www.genbeta.com/web/android-un-sistema-operativo-libre-para-moviles-promovido-por-google.

Gracia, Joaquin. 2005. *Patrones de diseño.* [Online] 5 27, 2005. http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

Gutiérrez, Juan José Andrés. 2008. La Cofa. *Linux y la Electrónica de Consumo.* [Online] Junio 4, 2008. [Cited: Junio 19, 2009.] http://www.lacofa.es/index.php/tag/sistemas-embebidos.

HackvaNews. http://hackvan.blogspot.com/. [Online] http://hackvan.blogspot.com/2009/03/desarrollando-en-qt-y-c.html.

2007. http://usuarios.lycos.es. [Online] 2007. http://usuarios.lycos.es/absurdosoyyo/tutc.html.

iNextSystems. 2007. Tecnologías de NextSystems. [Online] iNextSystems, 2007. http://inextsystems.com/pag/tecnologias-vs.php.

Interfaces W&T. *Sistemas de bus RS485.* [Online] http://www.wut.de/e-6wwww-11-apes-000.php.

InterEmpresas. http://www.interempresas.net. *Contador trifásico - Circutor, S.A.* [Online] http://www.interempresas.net/Electricidad_Electronica/FeriaVirtual/ResenyaProducto.asp?R=14379.

Jacobson, Ivar. 2000. *El Proceso Unificado De Desarrollo De Software.* s.l. : Prentice Hall Hispanoamerica , 2000.

Jose, Luis. 2008. My Opera. [Online] 11 24, 2008. http://my.opera.com/LuisJosue/blog/.

Juan, Francisco Javier Martinez. *Interfaces en los dispositivos computadores.* España : s.n.

Kioskea. Kioskea.net. [Online] http://es.kioskea.net/contents/pc/pda.php3.

Mañas, José A. 1994. http://www.lab.dit.upm.es. [Online] 3 16, 1994. http://www.lab.dit.upm.es/~lprg/material/apuntes/pruebas/testing.htm.

Martin Fouler, Kendall Scott. 1999. *UML Gota a Gota.* Mexico : s.n., 1999.

Modicon, Inc. 1996. *Modicon Modbus Protocol Reference Guide.* Massachusetts EE.UU : s.n., 1996.

PDA Electronic. 2005. PDA Electronic. [Online] 2005. http://www.pdaelectronic.com/.

Perez, Mauricio I. 2004. Pocket Mexico. [Online] 8 16, 2004. http://www.pocketmexico.com/Reviews+index-req-showcontent-id-39.html.

Pinedo, María José. 2007. El PDA robusto para entornos extremos. [Online] UNITECH, 10 25, 2007. www.unitech-europe.com.

Scott., Martin Fouler. Kendall. 1999. *UML Gota a Gota.* Mexico : s.n., 1999.

Tapía, John. 2009. WordPress.com. *Jhon Tapia Weblog.* [Online] 2009. [Cited: 6 14, 2009.] http://johntapia.wordpress.com/.

Tecnologias. 2009. http://www.tecnologiait.com.ar. [Online] junio 2009. http://www.tecnologiait.com.ar/?tag=so-moblin.

W3C. 2005. [Online] 2005. http://www.w3c.es/divulgacion/guiasbreves/tecnologiasXML.

—. 2008. Guía Breve de Tecnologías XML. [Online] 1 9, 2008. http://www.w3c.es/divulgacion/guiasbreves/tecnologiasXML.

Wolfert, Raymond. 2009. [Online] Sales & Marketing Manager, 2009.

Xataka Móvil. http://www.xatakamovil.com/sistemas-operativos. *Maemo Harmattan podría ser el principio del fin de Symbian S60 .* [Online] http://www.xatakamovil.com/sistemas-operativos/maemo-harmattan-podria-ser-el-principio-del-fin-de-symbian-s60.

REFERENCIAS BIBLIOGRAFICAS.

[1]. InterEmpresas. http://www.interempresas.net. *Contador trifásico - Circutor, S.A.* [Online] http://www.interempresas.net/Electricidad_Electronica/FeriaVirtual/ResenyaProducto.asp?R=14379.

[2]. Tapía, John. 2009. WordPress.com. *Jhon Tapia Weblog.* [Online] 2009. [Cited: 6 14, 2009.]

[3]. Cardenas, Alfonso Araujo. 2004. http://mx.geocities.com. *Protocolos de Comunicación.* [Online] junio 2004. http://mx.geocities.com/alfonsoaraujocardenas/protocolos.html.

[4]. Modicon, Inc. 1996. *Modicon Modbus Protocol Reference Guide.* Massachusetts EE.UU : s.n., 1996. http://johntapia.wordpress.com/.

[5]. ArqHys. 2004. ArqHys Tecnologia. [Online] 2004. http://tecnologia.arqhys.com/pda/index.html.

[6]. PDA Electronic. 2005. PDA Electronic. [Online] 2005. http://www.pdaelectronic.com/.

[7]. [Online] http://static.scribd.com/docs/8wmxgmv38a49w.swf?INITIAL_VIEW=width.

[8]. Kioskea. Kioskea.net. [Online] http://es.kioskea.net/contents/pc/pda.php3.

[9]. Jose, Luis. 2008. My Opera. [Online] 11 24, 2008. http://my.opera.com/LuisJosue/blog/.

[10]. Perez, Mauricio I. 2004. Pocket Mexico. [Online] 8 16, 2004. http://www.pocketmexico.com/Reviews+index-req-showcontent-id-39.html.

[11].Tecnologias. 2009. http://www.tecnologiait.com.ar. [Online] junio 2009. http://www.tecnologiait.com.ar/?tag=so-moblin.

[12].Genbeta. http://www.genbeta.com. [Online] http://www.genbeta.com/web/android-un-sistema-operativo-libre-para-moviles-promovido-por-google.

[13].Fdez-Baillo, David Carrero. 2008. Carrero.es. *Sistemas operativos para dispositivos móviles.* [Online] Agosto 12, 2008. http://carrero.es/sistemas-operativos-para-dispositivos-moviles/2059.

[14].Xataka Móvil. http://www.xatakamovil.com/sistemas-operativos. *Maemo Harmattan podría ser el principio del fin de Symbian S60 .* [Online] http://www.xatakamovil.com/sistemas-operativos/maemo-harmattan-podria-ser-el-principio-del-fin-de-symbian-s60.

[15]. Juan, Francisco Javier Martinez. *Interfaces en los dispositivos computadores.* España : s.n.

[16]. Pinedo, María José. 2007. El PDA robusto para entornos extremos. [Online] UNITECH, 10 25, 2007. www.unitech-europe.com.

[17]. Eclipse. 2009. OpenUP. [Online] 2009. http://epf.eclipse.org/wikis/openup/.

[18]. 2007. http://usuarios.lycos.es. [Online] 2007. http://usuarios.lycos.es/absurdosoyyo/tutc.html.

[19]. —. 2008. Guía Breve de Tecnologías XML. [Online] 1 9, 2008. http://www.w3c.es/divulgacion/guiasbreves/tecnologiasXML.

[20]. Clikear.com. http://www.clikear.com. *Desarrollo Orientado a Objetos con UML.* [Online] http://www.clikear.com/manuales/uml/index.aspx

[21]. Martin Fouler, Kendall Scott. 1999. *UML Gota a Gota.* Mexico : s.n., 1999.

[22]. iNextSystems. 2007. Tecnologías de NextSystems. [Online] iNextSystems, 2007. http://inextsystems.com/pag/tecnologias-vs.php.

[23]. Gracia, Joaquin. 2005. *Patrones de diseño.* [Online] 5 27, 2005. http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

[24]. Jacobson, Ivar. 2000. *El Proceso Unificado De Desarrollo De Software.* s.l. : Prentice Hall Hispanoamerica , 2000.

[25]. Mañas, José A. 1994. http://www.lab.dit.upm.es. [Online] 3 16, 1994. http://www.lab.dit.upm.es/~lprg/material/apuntes/pruebas/testing.htm.

ANEXOS

Fig.14: Conector hembra RS-232.

Fig.15: Contador Trifásico Multifuncional serie CIRWATT

Modo RTU

Comienzo de Trama	Dirección	Función	Datos	Control de Errores	Fin de Trama
Tiempo de 3 bytes	1 bytes	1 bytes	N x 1 bytes	2 bytes	

Tabla 29: Formato Trama ModBus RTU

Fig.16: Visual de Configurar Descarga

Fig.17: Visual de Descargar Datos.

Fig.18: Visual de la forma principal.

Fig.19: Visual de Exportar Datos.

Aplicaciones de los tipos de circutor CIRWATT.

Gran Industria/ Subestaciones	Mediana Industria	Pequeña Industria/ Servicios	Uso Domestico
CIRWATT D	CIRWATT D		
	CIRWATT C	CIRWATT C	
		CIRWATT A	CIRWATT A

Tabla 30: Aplicaciones de los tipos de circutor CIRWATT

GLOSARIO

CRC: Comprobación de redundancia cíclica. CRC es un tipo de función que recibe un flujo de datos de cualquier longitud como entrada y devuelve un valor de longitud fija como salida. Pueden ser usadas como suma de verificación para detectar la alteración de datos durante su transmisión o almacenamiento. La CRC fue inventada y propuesta por W. Wesley Peterson en un artículo publicado en 1961.

Módulo: Es una parte de un programa, o más general un componente de un sistema que tiene una interfaz bien definida para interactuar con otros módulos.

OpenUP: Es un Proceso Unificado de Desarrollo de Software que aplica un acercamiento iterativo e incremental dentro de un ciclo de vida estructurado.

PDA: Asistente Digital Personal es un Ordenador de pequeño tamaño, algo mayor que un paquete de cigarrillos.

Qt: (Quasar Technologies) Es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario.

UML: Unified Modeling Language. Es una notación estándar para modelar objetos del mundo real como primer paso en el desarrollo de programas orientados a objetos. Es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema de software.

Visual Paradigm: Es una herramienta CASE (Computer-Aided Software Engineering) que permite realizar ingeniería tanto directa como inversa.

XML: XML (del inglés Extensible Markup Language) desarrollado por World Wide Web Consortium, es una versión simplificada del SGML (del inglés Standard Generalized Markup Language), es considerado un método para introducir datos estructurados en un fichero de texto.