

# Universidad de las Ciencias Informáticas

## Facultad 5



### **Título: Sistema de adquisición de datos para mediciones Multiconto.**

**Trabajo de diploma para optar por el título de  
Ingeniero en Ciencias Informáticas.**

**Autor:** Leyner Rodríguez Borrego.

**Tutor:** Msc. Maikel Díaz Castro.

**Cotutor:** Ing. Ignais La Paz Trujillo.

*Ciudad de La Habana, 2010.*

*Año del 51 Aniversario de la Revolución*

## DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los \_\_\_\_ días del mes de \_\_\_\_\_ del año \_\_\_\_\_.

\_\_\_\_\_  
Autor

\_\_\_\_\_  
Tutor.

## **DATOS DE CONTACTO**

### **Maikel Díaz Castro. (maikel@instec.cu)**

Graduado como Licenciado en Física Nuclear en el Instituto Superior de Ciencias y Tecnologías Nucleares, ISCTN (actualmente Instituto Superior de Tecnologías y Ciencias Aplicadas, InSTEC) en el año 2003. Obtiene el título de Máster en Ciencias en Física Nuclear en el InSTEC en el año 2005. Durante sus estudios y su labor profesional ha realizado trabajos relacionados con la instrumentación nuclear y los sistemas de adquisición de datos relacionados con la Física Nuclear Experimental. Actualmente trabaja como profesor instructor en el InSTEC y forma parte del grupo de desarrollo de hardware del mismo.

### **Ignais La Paz Trujillo. (ilapaz@uci.cu)**

Graduado como Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en el año 2007. Durante su labor profesional a ocupado el rol de arquitecto de software en proyectos perteneciente al Centro de Informática Industrial (CEDIN) de la UCI y actualmente desempeña el rol desarrollador de software.

## AGRADECIMIENTOS

*A mi mamá por creer y depositar tanta confianza en mí, gracias por tus consejos y por estar siempre atenta.*

*A mi papá por ser mi ejemplo a seguir desde siempre.*

*A mi hermana por quererme tanto y apoyarme.*

*A toda mi familia por brindarme su apoyo en todo momento en especial a mis tías Hilda, Lidia y Mirian.*

*A mi tutor Maikel Díaz Castro por ser más que un tutor un excelente amigo y compañero, sin toda su ayuda y dedicación no hubiese sido posible la realización de este trabajo.*

*A mi cotutor Ignais por la ayuda prestada.*

*A mis amigos del barrio en especial a Aliannys y Jorge, por estar presente en las buenas y las malas.*

*A mis compañeros de los grupos en los que he estado por todo su apoyo y amistad en especial a Maydelin, Yailin, Yosleidys, Yunetsy, Julito, Hugo, Adrian (El chino) y Jose.*

*A todas las personas que de una forma u otra han hecho posible la realización de este trabajo.*

## DEDICATORIA

*A mis padres, lo más grande que tengo en la vida, gracias a ellos soy lo que soy.*

*A mi hermana que es mi vida y se que también soy la de ella.*

*A mis tres abuelas Mimí, Felipa y Delia.*

## RESUMEN

El presente trabajo de diploma está centrado en el desarrollo de un sistema de adquisición de datos para la colección de pulsos provenientes de un detector de radiaciones, basándose en el método multiconteo. Se pueden encontrar algunos aspectos relacionados con los analizadores multicanales y otros sistemas similares. Entre las principales tecnologías y herramientas que se emplearon en el desarrollo del sistema se encuentran OpenUp, Visual Paradigm, QT, Eclipse, Micro C, Proteus y como lenguajes de programación C, C++ y ensamblador. Se diseñó el firmware y el software para la adquisición y visualización de los datos, obteniéndose como resultado final, un sistema que permite realizar algunas de las prácticas docentes referentes al estudio de las propiedades nucleares en los laboratorios de física nuclear del Instituto Superior de Tecnologías y Ciencias Aplicadas (InSTEC).

**Palabra claves:** adquisición, multiconteo, firmware, analizador multicanal.

# INDICE

<b>INTRODUCCIÓN.....</b>	<b>1</b>
<b>CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA.....</b>	<b>4</b>
1.1 INTRODUCCIÓN.....	4
1.2 FUNDAMENTACIÓN TEÓRICA .....	4
1.2.1 <i>Analizador Multicanal</i> .....	4
1.3 ESTADO DEL ARTE.....	5
1.3.1 <i>Sistemas existentes para mediciones multiconto.</i> .....	5
1.3.3 <i>Propuesta de solución técnica</i> .....	7
1.4 METODOLOGÍA, LENGUAJE Y HERRAMIENTAS A UTILIZAR EN LA SOLUCIÓN DEL PROBLEMA.....	8
1.4.1 <i>Lenguaje de Modelado</i> .....	8
1.4.2 <i>Metodologías para el desarrollo de software</i> .....	9
1.4.2.1 <i>Proceso Unificado de Desarrollo</i> .....	10
1.4.2.2 <i>Extreme Programing (XP)</i> .....	12
1.4.2.3 <i>Proceso de desarrollo OpenUP</i> .....	14
1.4.2.4 <i>Microsoft Solution Framework (MSF)</i> .....	14
1.4.2.5 <i>Selección de la metodología de desarrollo de software a utilizar</i> .....	15
1.5 HERRAMIENTAS CASE.....	15
1.5.1 <i>Rational Rose</i> .....	16
1.5.2 <i>Visual Paradigm</i> .....	17
1.5.3 <i>Selección de la herramienta de modelado a utilizar</i> .....	17
1.6 LENGUAJES.....	17
1.6.1 <i>Ensamblador</i> .....	18
1.6.2 <i>Ensamblador para PIC</i> .....	18
1.6.3 <i>C</i> .....	19
1.6.4 <i>C++</i> .....	19
1.7 HERRAMIENTAS.....	20
1.7.1 <i>QT</i> .....	20
1.7.2 <i>Eclipse</i> .....	20
1.7.3 <i>Micro C PRO for PIC</i> .....	21
1.7.4 <i>Proteus</i> .....	21
1.8 CONCLUSIONES DEL CAPÍTULO.....	21
<b>CAPÍTULO 2 CARACTERÍSTICAS DEL SISTEMA.....</b>	<b>22</b>
2.1 INTRODUCCIÓN.....	22
2.2 MODELO DEL DOMINIO.....	22
2.3 ESPECIFICACIÓN DE LOS REQUERIMIENTOS DEL SISTEMA.....	23
2.3.1 <i>Requerimientos funcionales</i> .....	23
2.3.2 <i>Requerimientos no funcionales</i> .....	24

2.4 ACTORES DEL SISTEMA.....	26
2.5 DIAGRAMA DE CASOS DE USO DEL SISTEMA.....	26
2.6 DESCRIPCIÓN DE LOS CASOS DE USO DEL SOFTWARE.....	27
2.7 CONCLUSIONES DEL CAPÍTULO.....	34
<b>CAPÍTULO 3 CARACTERÍSTICAS DEL HARDWARE.....</b>	<b>35</b>
3.1 DESCRIPCIÓN DEL HARDWARE.....	35
3.2.1 Componentes.....	35
3.3 DESCRIPCIÓN DEL FIRMWARE.....	37
3.3.1 Diseño e implementación del firmware.....	38
3.3.2 Servicio de interrupciones.....	39
3.4 CONCLUSIONES DEL CAPÍTULO.....	41
<b>CAPÍTULO 4 ANÁLISIS, DISEÑO E IMPLEMENTACION DEL SOFTWARE.....</b>	<b>42</b>
4.1 INTRODUCCIÓN.....	42
4.2 ANÁLISIS.....	42
4.2.1 Diagrama de clases del análisis.....	42
4.2.2 Diagrama de iteración.....	45
4.3 DISEÑO.....	48
4.3.1 Arquitectura y patrones utilizados.....	48
4.3.2 Diagrama de paquete.....	50
4.3.3 Diagrama de clase del diseño.....	51
4.3.4 Descripción de las clases del diseño.....	53
4.4 IMPLEMENTACIÓN.....	63
4.4.1 Diagrama de despliegue.....	63
4.4.2 Diagrama de componente.....	63
4.5 CONCLUSIONES DEL CAPÍTULO.....	64
<b>CAPÍTULO 5 PRUEBAS.....</b>	<b>65</b>
5.1 INTRODUCCIÓN.....	65
5.3 PRUEBAS PARA EL SOFTWARE.....	65
5.4 PRUEBAS GENERALES REALIZADAS AL SISTEMA.....	69
5.5 CONCLUSIONES DEL CAPÍTULO.....	71
<b>CONCLUSIONES.....</b>	<b>72</b>
<b>REFERENCIAS BIBLIOGRÁFICAS.....</b>	<b>74</b>
<b>BIBLIOGRAFÍA.....</b>	<b>75</b>
<b>GLOSARIO DE TÉRMINO.....</b>	<b>78</b>
<b>ANEXOS.....</b>	<b>80</b>



## INDICE DE FIGURAS

<b>Figura 1:</b> Solución propuesta. ....	7
<b>Figura 2:</b> Proceso Unificado de Software. ....	11
<b>Figura 3:</b> Metodología XP. ....	13
<b>Figura 4:</b> Modelo del dominio. ....	23
<b>Figura 5:</b> Diagrama de casos de uso del software. ....	26
<b>Figura 6:</b> Diagrama de bloques del hardware. ....	35
<b>Figura 7:</b> Diagrama de flujo del firmware. ....	37
<b>Figura 8:</b> Diagrama de flujo general del servicio de atención a interrupciones. ....	39
<b>Figura 9:</b> Diagrama de flujo general del servicio de atención de la interrupción externa. ....	40
<b>Figura 10:</b> Diagrama de clases del análisis del CU Establecer comunicación con el dispositivo MC-mcSystem. ....	43
<b>Figura 11:</b> Diagrama de clases del análisis del CU Configurar adquisición del dispositivo MC–mcSystem. ....	43
<b>Figura 12:</b> Diagrama de clases del análisis del CU adquirir datos de dispositivo MC –mcSystem. ....	43
<b>Figura 13:</b> Diagrama de clases del análisis del CU Visualizar datos adquiridos. ....	43
<b>Figura 14:</b> Diagrama de clases del análisis del CU Guardar datos adquiridos. ....	44
<b>Figura 15:</b> Diagrama de clases del análisis del CU cargar datos de dispositivo extraíble. ....	44
<b>Figura 16:</b> Diagrama de secuencia del CU Establecer comunicación con el dispositivo MC-mcSystem. ....	45
<b>Figura 17:</b> Diagrama de secuencia del CU Configurar la adquisición del dispositivo MC –mcSystem. ....	46
<b>Figura 18:</b> Diagrama de secuencia del CU Adquirir datos de dispositivo MC-mcSystem. ....	46
<b>Figura 19:</b> Diagrama de secuencia del CU Visualizar datos adquiridos. ....	47
<b>Figura 20:</b> Diagrama de secuencia del CU Guardar datos adquiridos. ....	47
<b>Figura 21:</b> Diagrama de secuencia del CU Cargar datos de dispositivo extraíble. ....	48
<b>Figura 22:</b> Diagrama de paquete. ....	50
<b>Figura 23:</b> Diagrama de clases del diseño subsistema Graph. ....	51
<b>Figura 24:</b> Diagrama de clases del diseño subsistema Driver. ....	52

<b>Figura 25:</b> Diagrama de clases del diseño subsistema mcSystem. ....	53
<b>Figura 26:</b> Diagrama de despliegue del software. ....	63
<b>Figura 27:</b> Diagrama de componentes del sistema. ....	63
<b>Figura 28:</b> Montaje experimental para la medición del tiempo de decaimiento del In. ....	69
<b>Figura 29:</b> Curva de decaimiento para el sistema Aptec. ....	70
<b>Figura 30:</b> Curva de decaimiento para el sistema MC-mcSystem. ....	70
<b>Figura 31:</b> Interfaz del software para establecer la comunicación con el dispositivo MC-mcSystem. ....	80
<b>Figura 32:</b> Interfaz del software para la configuración de la adquisición desde el dispositivo MC-mcSystem. ..	81
<b>Figura 33:</b> Interfaz del software para la visualización de los datos adquiridos del dispositivo MC-mcSystem. ....	82

## INDICE DE TABLAS

<b>Tabla 1:</b> Actor del sistema. ....	26
<b>Tabla 2:</b> Caso de Uso Establecer comunicación con el dispositivo. ....	27
<b>Tabla 3:</b> Caso de Uso Configurar la adquisición del dispositivo MC –mcSystem. ....	28
<b>Tabla 4:</b> Caso de Uso Adquirir datos de dispositivo MC-mcSystem. ....	29
<b>Tabla 5:</b> Caso de Uso Visualizar datos adquiridos. ....	31
<b>Tabla 6:</b> Caso de Uso Guardar datos adquiridos. ....	32
<b>Tabla 7:</b> Caso de Uso Cargar datos de dispositivo extraíble. ....	32
<b>Tabla 8:</b> Descripción de la clase MCAGraph. ....	54
<b>Tabla 9:</b> Descripción de la clase MCAGraphBase. ....	55
<b>Tabla 10:</b> Descripción de la clase MSAdq. ....	57
<b>Tabla 11:</b> Descripción de la clase ADQThread. ....	58
<b>Tabla 12:</b> Descripción de la clase mcSystem. ....	59
<b>Tabla 13:</b> Descripción de la clase Preferences. ....	61
<b>Tabla 14:</b> Caso de prueba 1 Establecer Comunicación. ....	65

<b>Tabla 15:</b> Caso de prueba 2 Establecer Comunicación. ....	66
<b>Tabla 16:</b> Caso de prueba 3 Establecer Comunicación. ....	67
<b>Tabla 17:</b> Caso de prueba Adquirir datos. ....	68
<b>Tabla 18:</b> Resultados experimentales del tiempo de semidesintegración. ....	71



## INTRODUCCIÓN

La Universidad de las Ciencias Informáticas (UCI) es una universidad productiva, cuya misión es producir software y servicios informáticos a partir de la vinculación estudio – trabajo como modelo de formación. La producción de software y servicios informáticos se basa en la integración de los procesos de formación, investigación y producción en torno a una temática para convertirla en una rama productiva. Este espacio de integración temática es denominado Centro Productivo y se promueve la colaboración nacional e internacional, el fomento de líneas de investigación y desarrollo, y la ejecución de proyectos en el marco de acuerdos de trabajo. Esta integración garantiza la innovación continua que genera y aporta valor a los productos y servicios, promueve la gestión del conocimiento garantizando un mayor rendimiento, logra una mejor utilización y aprovechamiento de los recursos humanos y materiales, generando alta especialización y colaboración. [1]

La UCI desarrolla programas de informatización de la sociedad cubana a través de la relación con entidades y universidades nacionales, entre las que se encuentra el Instituto Superior de Tecnologías y Ciencias Aplicadas, que es un centro de la educación superior donde se realizan estudios de pregrado en especialidades nucleares. En el InSTEC, además, se imparten cursos de postgrados, maestrías, doctorados y se realizan investigaciones en diferentes campos de la actividad nuclear y la gestión ambiental. El instituto está dividido en tres facultades donde se estudian las especialidades: Radioquímica, Meteorología y Física Nuclear.

La Facultad de Ciencias y Tecnología Nucleares (FCTN), es la encargada de formar licenciados en la especialidad de Física Nuclear, dicha facultad cuenta con varios laboratorios docentes, los que deben estar equipados con los medios materiales necesarios que les permitan prestar un servicio de apoyo tecnológico a las asignaturas que así lo requieran.

Multiconteo es un método que se utiliza en el estudio de propiedades nucleares, principalmente en la medición del tiempo de desintegración de isótopos de vida corta y en la lectura de dosímetros termoluminiscentes. Este método consiste en contar la ocurrencia de un evento determinado durante un período de tiempo y repetir el proceso varias veces con el objetivo de analizar la relación entre las dos magnitudes.

Actualmente los laboratorios docentes no cuentan con una herramienta que permita realizar esta medición, ya que la que existe en el instituto esta obsoleta desde hace varios años y ha presentado problemas técnicos en varias ocasiones, lo que ha traído consigo que los estudiantes no puedan realizar varias de las prácticas de laboratorios concebidas por el plan de estudio, lo que trae como consecuencia que se vea afectado el proceso de enseñanza docente.

**Problema Científico:** ¿Cómo mejorar el proceso de medición de equipamiento, para mediciones multiconto de los laboratorios de Física Nuclear del InSTEC?

**Objeto de Estudio:** El proceso de medición de propiedades nucleares.

**Campo de Acción:** El proceso de medición de propiedades nucleares utilizando el método multiconto.

Acorde con el problema científico planteado, la **idea a defender** es la siguiente: el desarrollo del sistema de adquisición de datos para mediciones multiconto permitirá realizar mediciones nucleares en los laboratorios docentes de la Facultad de Ciencias y Tecnología Nucleares del InSTEC.

Para lograr definir una línea base en la investigación se plantea como **Objetivo General:** Diseñar y construir un sistema de adquisición de datos para mediciones multiconto.

Para dar cumplimiento al objetivo se han propuesto las siguientes **tareas de Investigación:**

- ❖ Definición de las tecnologías y herramientas existentes para el desarrollo de aplicaciones de escritorio.
- ❖ Realización de un diagnóstico de las tendencias actuales de los sistemas de mediciones nucleares.
- ❖ Evaluación y comparación de los sistemas de adquisición de datos con microprocesadores de la familia PIC.
- ❖ Implementación del firmware para el control de la adquisición y comunicación con la computadora.
- ❖ Implementación del software para la realización de mediciones multiconto.
- ❖ Realización de pruebas al sistema.

Para el cumplimiento de los objetivos se combinan diferentes métodos y técnicas en la búsqueda y procesamiento de la información, los fundamentales son:

### **A nivel teórico**

Se utilizó el método **analítico – sintético** a partir de un análisis detallado de los sistemas existentes, así como documentos y teorías que se relacionan con el proceso de mediciones multiconto, sintetizando los elementos más importantes y de mayor utilidad para el desarrollo del trabajo, de modo tal que dichos elementos contribuyan a proponer una solución acertada.

Se recurrió al uso del método **análisis histórico-lógico** para conocer los antecedentes y tendencias de los sistemas para mediciones multiconto existentes, así como la evolución de las herramientas y metodologías informáticas que sirvan de ayuda para el desarrollo e implementación del sistema.

## **CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA**

### **1.1 Introducción**

En este capítulo se realiza una investigación relacionada con los sistemas de adquisición de datos, incluyendo el estado de arte a nivel mundial, en Cuba y las tendencias actuales. También se exponen las principales características de las tecnologías y herramientas seleccionadas para la solución del problema planteado, así como la justificación de la selección de las mismas.

### **1.2 Fundamentación teórica**

#### **1.2.1 Analizador Multicanal**

El Analizador Multicanal (MCA) es un instrumento de laboratorio fundamental en la Física Nuclear que permite obtener distribuciones de señales a partir de la medición de los pulsos a su entrada. Éste opera en dos modos diferentes, el modo de análisis de alturas de pulsos (PHA por sus siglas en inglés) y el modo de multiconteo (MCS). En el modo PHA, los pulsos de entrada son almacenados en localizaciones (canales) de acuerdo a su amplitud, mientras que en el modo MCS estos son almacenados de acuerdo al tiempo en que llegan. En los MCA modernos el control es realizado por microprocesadores que adicionan nuevas funcionalidades y aumentan la capacidad de procesamiento de datos. Mientras que el modo PHA es utilizado fundamentalmente para investigar espectros nucleares de rayos gamma, el modo MCS es utilizado en el estudio de distribuciones estadísticas. En este modo, todos los pulsos entrantes son contados durante un tiempo, denominado *tiempo de residencia*, y son almacenados en una localización de memoria denominada *canal*. Posteriormente los conteos son movidos a la siguiente localización, por el mismo período de tiempo, repitiendo el proceso un número determinado de veces. El resultado final es una distribución temporal de los eventos medidos. El tiempo de residencia es usualmente seleccionado mediante el panel de control del dispositivo. [2]



## **1.3 Estado del Arte**

### **1.3.1 Sistemas existentes para mediciones multiconteo.**

Existen pocos sistemas en el mundo que implementan únicamente el método multiconteo. De manera general conforma uno de los modos de operación de los MCA comerciales, aunque no está presente en todos estos sistemas. La mayoría de los sistemas de este tipo se diseñan para una aplicación específica y presentan el inconveniente de ser poco flexibles para otras aplicaciones. En el caso opuesto existen otros que se diseñan lo suficientemente flexibles para abarcar un gran número de aplicaciones pero son mucho más caros y difíciles de adquirir. Este es el caso de los analizadores multicanales que tienen este método de adquisición como un modo de operación más.

En el caso de los software, son solamente diseñados para un único tipo de equipamiento teniendo en cuenta el hardware y la gran mayoría no son de código abierto. Readaptarlos a otro sistema es imposible. Además, no son multiplataformas, y en su mayoría son diseñados para ejecutarse principalmente sobre sistemas operativos Windows.

Los sistemas desarrollados en nuestro país ya están obsoletos, y tienen la desventaja de ser diseñados para usar como interfaz de comunicación con la computadora el puerto ISA no presente en las computadoras actuales.

#### **1.3.1.1 Sistema de Adquisición de datos MCA8000A**

Es un completo analizador multicanal de baja potencia para el uso de una amplia variedad de sistemas de detección. Este sistema está compuesto por el hardware MCSA8000A que se conecta a cualquier ordenador mediante una interfaz serie RS-232 o por puerto USB, aunque este puede trabajar de forma independiente almacenando los valores en memoria.

El sistema tiene un software llamado ADMCA que permite la configuración del MCA8000A, la adquisición de datos y su visualización. El mismo puede ser utilizado para la caracterización de los tubos de rayos-X, el monitoreo a largo plazo de radiación de ambiente, así como el estudio del detector de estabilidad como una función de tiempo. [3]

### **1.3.1.2 Módulo de Adquisición de datos CMCA-550**

Es un sistema de adquisición de datos utilizado para la espectroscopia Mössbauer. Tiene dos modos de funcionamiento: modo multiconteo y modo análisis de alturas de pulsos.

El CMCA-550 se comunica con el ordenador mediante una interfaz serie RS-232 o por puerto USB. Este módulo permite mediante un software la visualización de espectros Mössbauer, así como el análisis de la altura del pulso. [4]

### **1.3.1.3 MCA: Analizador Multicanal**

Este sistema utiliza una tarjeta de adquisición de datos y procesamiento de señales digitales D32kitty y un software para controlar el sistema multicanal MD 1.0, además de contar con otra tarjeta para crear una interfaz entre la D32kitty y el conversor análogo digital. La tarjeta diseñada para dicha interfaz incluye además un dispositivo para el control del tiempo real que permite realizar adquisiciones en modo de multiconteo.

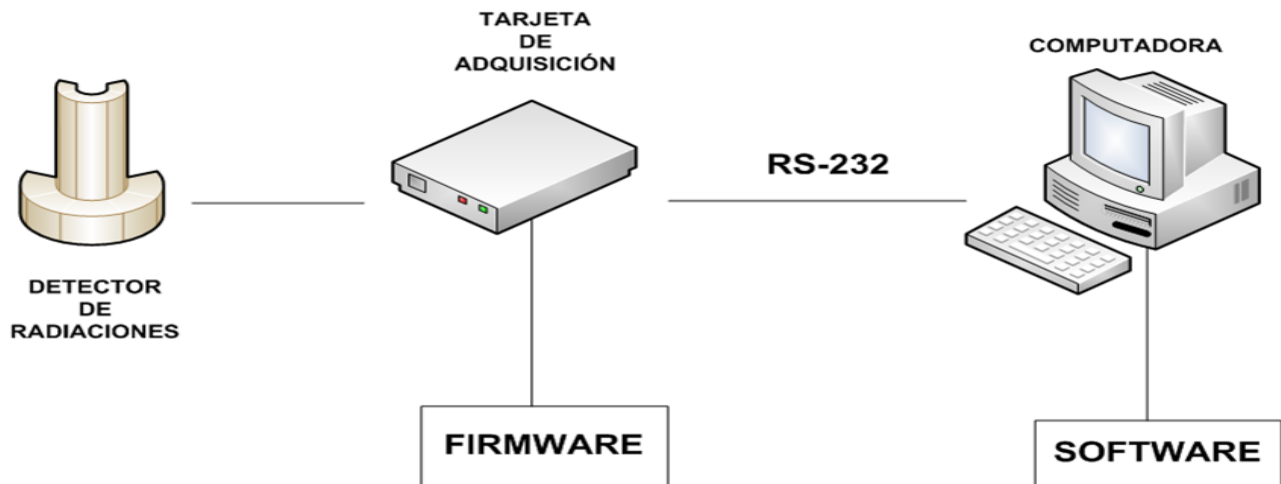
El programa permite configurar el sistema para realizar la adquisición en los modos de altura de pulso y multiconteo, situar el nivel de discriminación y tiempo de residencia, iniciar, detener y definir la forma de detención de la adquisición, hacer análisis simples de espectros, calibrar en energía y almacenar los espectros obtenidos. [5]

### **1.3.1.4 Sistema de adquisición de datos para aplicaciones donde sea necesario el multiconteo de eventos.**

Es un sistema de adquisición de datos de bajo costo formado por una tarjeta de adquisición que permite cubrir las necesidades de experimentos y mediciones donde sea necesario almacenar el resultado de los conteos de eventos, correlacionados con el valor de alguna magnitud física, en intervalos de tiempo definidos o en multiconteo, incluye además, un programa que se encarga de controlar la tarjeta de adquisición, realizar la adquisición, análisis, procesamiento, visualización y almacenamiento de los datos generados. Este sistema es usado principalmente para el análisis y procesamiento de los datos provenientes de la lectura de dosímetros termoluminiscentes empleados en la dosimetría personal. [6]

### 1.3.3 Propuesta de solución técnica

A continuación se muestra una propuesta de diseño del sistema para la adquisición y visualización de los datos a construir, así como una breve explicación de los elementos que los conforman.



*Figura 1: Solución propuesta.*

- **Detector de Radiaciones:** Dispositivo utilizado para medir la radiación emitida por los núcleos al desintegrarse.
- **Tarjeta de Adquisición:** Hardware diseñado para la adquisición de datos.
- **Firmware:** Programa para el control de la tarjeta de adquisición que permitirá recolectar los pulsos provenientes del detector y el envío de los datos a la computadora, basándose en el método multiconto.
- **Computadora:** Recibirá los datos enviados desde la tarjeta de adquisición por el puerto serie RS-232.
- **Software:** Visualizará los datos enviados desde la tarjeta de adquisición a la computadora.

## **1.4 Metodología, lenguaje y herramientas a utilizar en la solución del problema.**

### **1.4.1 Lenguaje de Modelado.**

El lenguaje de modelado unificado (UML), es el lenguaje de modelado de sistemas de software más utilizado en la actualidad; es el estándar internacional aprobado por la *Object Management Group* (OMG), consorcio creado en 1989 responsable de la creación, desarrollo y revisión de especificaciones para la industria del software.

UML es un grupo de especificaciones de notación orientadas a objetos compuesto por distintos diagramas que representan las diferentes etapas del desarrollo de un proyecto de software. Permite la especificación, visualización, construcción y documentación de elementos de la ingeniería del software.

Algunas de las características que propician que con el uso de UML se pueda desarrollar un modelado eficiente son:

- La simplicidad de la comunicación entre desarrolladores de software.
- La facilidad de entendimiento y aprendizaje de sus principios principales.
- Permite y viabiliza la comunicación entre trabajadores del proyecto y los usuarios.
- La estandarización de los elementos del diseño de sistemas.

UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de lo que se quiere representar.

Los diagramas de estructura resaltan los elementos que deben existir en el sistema modelado:

- Diagrama de clases
- Diagrama de componentes
- Diagrama de objetos
- Diagrama de estructura compuesta (UML 2.0)
- Diagrama de despliegue
- Diagrama de paquetes

- Diagrama de casos de uso

Los diagramas de comportamiento destacan lo que debe suceder en el sistema modelado:

- Diagrama de actividades
- Diagrama de estados

Los diagramas de interacción son un subtipo de diagramas de comportamiento, que enfatiza sobre el flujo de control y de datos entre los elementos del sistema modelado:

- Diagrama de secuencia
- Diagrama de comunicación, que es una versión simplificada del diagrama de colaboración (UML 1.x)
- Diagrama de tiempos (UML 2.0)
- Diagrama global de interacciones o diagrama de vista de interacción (UML 2.0) [7]

#### **1.4.2 Metodologías para el desarrollo de software.**

En la actualidad en la industria de software existe una tendencia al crecimiento del volumen y complejidad de los productos, los proyectos se terminan excesivamente tardes, se exige mayor calidad y productividad en menos tiempo y hay insuficiente personal calificado. Como resultado de estos problemas ha surgido una alternativa desde hace mucho: la metodología de desarrollo de software.

Se entiende por metodología de desarrollo, una colección de documentación formal referente a los procesos, las políticas y los procedimientos que intervienen en el desarrollo del software. La finalidad de una metodología de desarrollo es garantizar la calidad y la eficiencia en el proceso de generación de software.

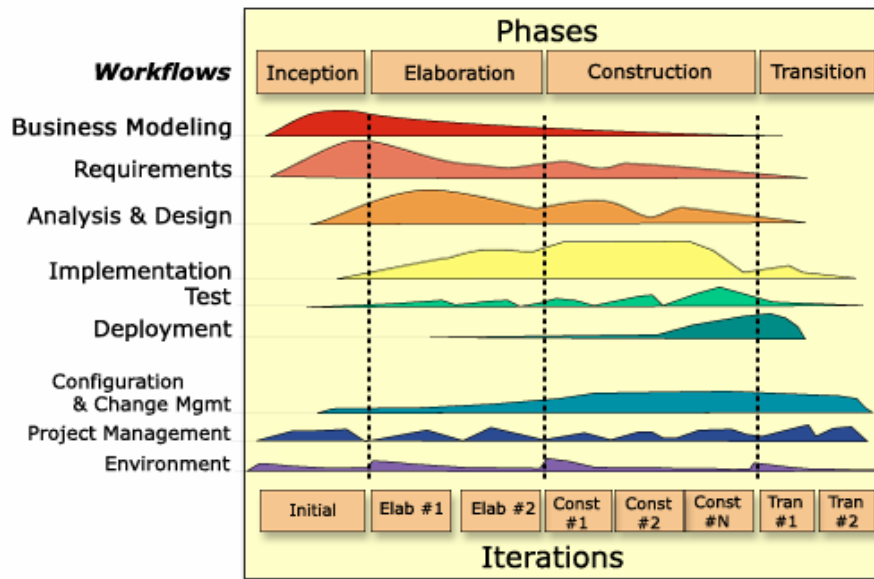
Existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo, elegir la mejor para un equipo en un determinado proyecto podría marcar la diferencia a la hora de obtener un software eficiente, fiable y con la calidad requerida.

### 1.4.2.1 Proceso Unificado de Desarrollo.

RUP por sus siglas en inglés *Rational Unified Process* traducido al español Proceso Unificado de Desarrollo es una metodología robusta que se distingue por ser: dirigida por casos de uso, centrada en la arquitectura, iterativo e incremental, esto es lo que la hace un proceso unificado. RUP es el resultado de varios años de desarrollo y uso práctico en el que se han unificado técnicas de desarrollo, a través del UML, y trabajo de muchas metodologías utilizadas por los clientes. La versión que se ha estandarizado surgió en 1998 y se conoció en sus inicios como Proceso Unificado de Rational 5.0; de ahí las siglas con las que se identifica este proceso de desarrollo. Como RUP es un proceso, en su modelación define como sus principales elementos:

- **Trabajadores (“Quién”):** Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.
- **Actividades (“cómo”):** Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.
- **Artefactos (“Qué”):** Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.
- **Flujo de actividades (“Cuándo”):** Secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.[8]

RUP agrupa las actividades en grupos lógicos, definiendo seis flujos de trabajo principales y tres de apoyo. Además divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en las distintas actividades.



*Figura 2: Proceso Unificado de Software.*

### Principales características:

- Proceso dirigido por casos de uso:** Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso.
- Proceso centrado en la arquitectura:** La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. RUP se desarrolla mediante iteraciones, comenzando por los casos de uso relevantes desde el punto de vista de la arquitectura.
- Proceso iterativo e incremental:** El equilibrio correcto entre los casos de uso y la arquitectura es algo muy parecido al equilibrio de la forma y la función en el desarrollo del

producto, lo cual se consigue con el tiempo. Para esto, la estrategia que propone RUP es un proceso iterativo e incremental donde el trabajo se divide en partes más pequeñas o miniproyectos, permitiendo que el equilibrio entre casos de uso y arquitectura se vaya logrando durante cada miniproyecto. Se puede ver como una iteración (un recorrido más o menos completo a lo largo de todos los flujos de trabajo fundamentales) del cual se obtiene un incremento que produce un crecimiento en el producto. El proceso iterativo e incremental consta de una secuencia de iteraciones. Cada iteración aborda una parte de la funcionalidad total, pasando por todos los flujos de trabajo relevantes y refinando la arquitectura. [8]

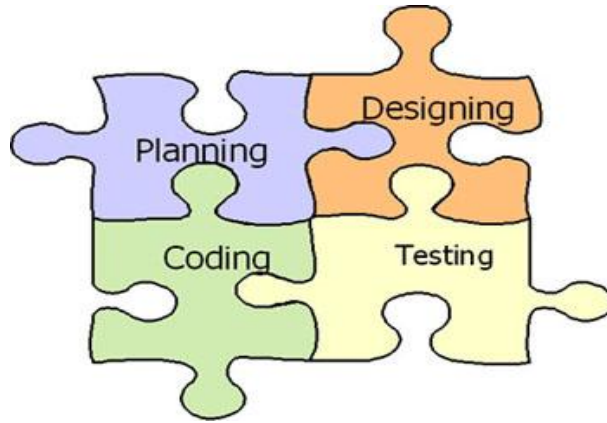
#### **1.4.2.2 Extreme Programing (XP)**

Es una de las metodologías de desarrollo de software más exitosas en la actualidad. Esta metodología ágil es utilizada para proyectos de corto plazo con equipos pequeños. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. Lo fundamental en esta metodología es:

- La comunicación entre los usuarios y los desarrolladores
- La simplicidad al desarrollar y codificar los módulos del sistema
- La retroalimentación concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales.



## Partes de XP:



*Figura 3: Metodología XP.*

## Principales características:

- **Pruebas Unitarias:** se basa en las pruebas realizadas a los principales procesos, de tal manera que si se adelanta en algo hacia el futuro, se puedan hacer pruebas de las fallas que pudieran ocurrir.
- **Refabricación:** se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- **Programación en pares:** una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. [9]

## ¿Qué es lo que propone XP?

- Empezar en pequeño y añadir funcionalidades con retroalimentación continua.
- Manejar el cambio como parte sustantiva del proceso.
- El costo del cambio no depende de la fase o etapa.
- No introducir funcionalidades antes que sean necesarias.

- Convertir al cliente o usuario en un miembro del equipo.

### 1.4.2.3 Proceso de desarrollo OpenUP

Es una metodología de desarrollo de software ágil basada en RUP. Es un proceso modelo y extensible, dirigido a gestión y desarrollo de proyectos de software basados en desarrollo iterativo, ágil e incremental. Permite un abordaje ágil al proceso de desarrollo de software, con sólo proveer un conjunto simplificado de contenidos, fundamentalmente relacionados con orientación, productos de trabajo, roles, y tareas.

Está caracterizado por cuatro principios básicos interrelacionados:

- Colaboración para unificar intereses y compartir conocimientos.
- Equilibrio de prioridades competentes a maximizar el valor de los involucrados con el resultado del proyecto.
- Enfoque en la articulación de la arquitectura.
- Desarrollo continuo para obtener realimentación y realizar las mejoras respectivas.

OpenUP se centra en articular la arquitectura para facilitar la colaboración técnica, reducir el riesgo y minimizar el esfuerzo de desarrollo. Procura un equilibrio entre las necesidades de los involucrados con los resultados del proyecto y los costos técnicos, con el fin de maximizar el valor de los involucrados y las guías del proceso de desarrollo. [10]

### 1.4.2.4 Microsoft Solution Framework (MSF)

Es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. MSF se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas.

#### Principales características:

- **Adaptable:** Es parecido a un compás, usado en cualquier parte como un mapa, del cual su uso es limitado a un específico lugar.

- **Escalable:** Puede organizar equipos tan pequeños entre tres o cuatro personas, así como también, proyectos que requieren cincuenta personas a más.
- **Flexible:** Es utilizada en el ambiente de desarrollo de cualquier cliente.
- **Tecnología Agnóstica:** Puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.

MSF se compone de varios modelos encargados de planificar las diferentes partes implicadas en el desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el Modelo de Aplicación.[8]

#### 1.4.2.5 Selección de la metodología de desarrollo de software a utilizar.

Una vez realizado el estudio de las diferentes metodologías de desarrollo, se decide utilizar la metodología OpenUp debido a las siguientes razones:

- Ya se han dado pasos de avances con el uso de esta metodología en la universidad, alcanzando una mayor experiencia en su utilización.
- Es la metodología usada en el Centro de Informática Industrial (CEDIN) para proyectos de poca envergadura, adecuándose así a las características de la investigación que se realiza.

### 1.5 Herramientas CASE.

Conocida como *Computer-Aided Systems Engineering*, es decir, (Ingeniería de Software Asistida por Ordenador), puede ser definida como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software (investigación preliminar, análisis, diseño, implementación e instalación).

CASE es también definido como el conjunto de métodos, utilidades y técnicas que facilitan el mejoramiento del ciclo de vida de desarrollo de un sistema de información, completamente o en alguna de sus fases.

#### Objetivos

- Mejorar la productividad en el desarrollo y mantenimiento del software.

- Aumentar la calidad del software.
- Mejorar el tiempo y coste de desarrollo y mantenimiento de los sistemas informáticos.
- Mejorar la planificación de un proyecto.
- Aumentar la biblioteca de conocimiento informático de una empresa ayudando a la búsqueda de soluciones para los requisitos.
- Automatizar, desarrollo del software, documentación, generación de código, pruebas de errores y gestión del proyecto.
- Ayuda a la reutilización del software, portabilidad y estandarización de la documentación
- Gestión global en todas las fases de desarrollo de software con una misma herramienta.
- Facilitar el uso de las distintas metodologías propias de la ingeniería del software.

Dentro de las principales aplicaciones CASE existentes para el modelado visual se encuentran el Rational Rose y Visual Paradigm.

### **1.5.1 Rational Rose**

Es una herramienta para el modelado visual mediante UML de sistemas de software. Rational Rose tiene una interfaz muy amigable conformada principalmente por un navegador que le permite al usuario navegar rápidamente por las distintas vistas del modelo. Esta herramienta con plataforma independiente ayuda a mejorar la comunicación en el trabajo en equipos y permite visualizar el sistema completo utilizando un lenguaje común UML. Los sistemas operativos que los soportan son Microsoft Windows y permite la integración con Borland JBuilder versiones 7.0 a 10.0, Microsoft Visual Studio, Sun Forte for Java Community y Enterprise Editions 3.0 entre otros.

#### **¿Qué ofrece?**

- Crear y visualizar diagramas UML.
- Especificar, analizar, y diseñar el sistema antes de codificarlo.
- Mantiene la consistencia de los modelos del sistema software.
- Chequeo de la sintaxis UML.
- Generar documentación automáticamente.
- Generar código Ada, ANSI C ++, C++, CORBA, Java y Visual Basic a partir de los Modelos.
- Ingeniería inversa (crear modelo a partir de código).

## 1.5.2 Visual Paradigm

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML; unas de las características más importantes de esta aplicación es que es multiplataforma con soporte para las plataformas GNU/Linux, Windows, Mac OS X entre otros. Visual Paradigm se integra con Eclipse, JBuilder, NetBeans IDE/Sun ONE, IntelliJ IDEA, Oracle JDeveloper, Microsoft Visual Studio entre otros.

### ¿Qué ofrece?

- Crear y visualizar diagramas UML.
- Generar código C++, CORBA, Java, esquemas XML a partir de los diagramas.
- Generar documentación.
- Realizar ingeniería inversa - código a modelo, código a diagrama.
- Generar bases de datos.
- Modelado colaborativo con CVS (*Concurrent Versions System*) y Subversión).

## 1.5.3 Selección de la herramienta de modelado a utilizar.

Aunque las dos herramientas antes estudiadas soportan el ciclo completo del desarrollo de software y cuentan con una interfaz amigable para los usuarios así como abundante documentación, se decide tomar como herramienta para desarrollar el modelado del sistema Visual Paradigm debido a que es multiplataforma con soporte para el sistema operativo GNU/Linux, además la UCI cuenta con la licencia que autoriza su uso.

## 1.6 Lenguajes.

### ¿Qué es un lenguaje de máquina?

Se denomina lenguaje máquina a la serie de datos que la parte física de la computadora o hardware, es capaz de interpretar.

### **1.6.1 Ensamblador**

Es un lenguaje simbólico de bajo nivel utilizado para escribir programas informáticos, y constituye la representación de la forma más directa del código máquina para una computadora o hardware más legible para un programador. Proporciona un conjunto de pseudo-operaciones también conocidas como directivas del ensamblador que sirven para definir datos, rutinas y todo tipo de información para que el programa ejecutable sea creado de determinada forma y en determinado lugar.

Ventajas:

- Un programa escrito en el lenguaje ensamblador requiere considerablemente menos memoria y tiempo de ejecución que un programa escrito en los conocidos lenguajes de alto nivel, como Pascal y C.
- El código en ensamblador se puede mezclar con códigos de lenguajes de alto nivel.
- El conocimiento del lenguaje ensamblador permite una comprensión de la arquitectura de la máquina o hardware que ningún lenguaje de alto nivel puede ofrecer.
- Los programas residentes y rutinas de servicio de interrupción casi siempre son desarrollados en el lenguaje ensamblador.

Desventajas:

- Requiere de un mayor tiempo de programación.
- Puede afectar recursos del sistema inesperadamente.
- Falta de portabilidad.

### **1.6.2 Ensamblador para PIC**

Es un lenguaje donde cada instrucción de un PIC está formada por una palabra de 14 bits y a su vez está dividida en un tipo de código denominado OPCODE, que especifica el tipo de instrucción, y uno o más operandos que además especifican la operación de la instrucción.

### 1.6.3 C

El lenguaje C fue creado en 1972 por Dennis Ritchie. Es un lenguaje muy utilizado para la programación de sistemas operativos, intérpretes, editores de texto, compiladores, ensambladores, controladores de redes, programas de comunicaciones, bases de datos entre otras cosas. El surgimiento de varios compiladores para su uso obligó a la creación del estándar ANSI (*American National Standards Institute*) que define una serie de características que deben tener los compiladores de C para garantizar la compatibilidad entre ellos.

#### Características del lenguaje:

- Compacto: sólo tiene 32 palabras reservadas.
- Estructurado: se basa en el empleo de funciones, que son bloques de código independientes.
- Portable: los programas ejecutables obtenidos funcionan en cualquier máquina (utilizando las bibliotecas y funciones estándar). El código fuente tampoco dependerá del compilador o del sistema operativo utilizado.
- Flexible: tiene pocas restricciones.
- Lenguaje de tipo medio: dispone de potentes sentencias propias de los lenguajes de alto nivel y características de lenguajes de bajo nivel como el ensamblador (permite el control de la CPU a nivel de registros, e incluso a nivel de bits), con plena disponibilidad de la memoria y puertos conectados a la CPU.
- Extensión: el C es un lenguaje muy extendido. Los compiladores son relativamente sencillos y reducidos, y cuando surge una nueva máquina o sistema operativo, es uno de los primeros programas que se diseñan.

### 1.6.4 C++

C++ es un lenguaje de programación creado por Bjarne Stroustrup en los laboratorios de At&T en 1983. Stroustrup tomó como base el lenguaje de programación más popular en aquella época el cual era C.

Bjarne Stroustrup, creó lo que se conoce como C++. Necesitaba ciertas facilidades de programación, incluidas en otros lenguajes pero que C no soportaba, al menos directamente, como son las llamadas

clases y objetos, conceptos muy en moda en la programación actual. Para ello rediseñó el C, ampliando sus posibilidades pero manteniendo su mayor cualidad, la de permitir al programador en todo momento tener controlado lo que está haciendo, consiguiendo así una mayor rapidez que no se conseguiría en otros lenguajes. El mismo es un lenguaje orientado a objetos derivado del C. Surgió con el objetivo de añadirle cualidades y características que carecía su ancestro. Este lenguaje es muy ligado al hardware, manteniendo una considerable potencia para programación a bajo nivel, le añade elementos que le permiten también un estilo de programación con alto nivel de abstracción.

## **1.7 Herramientas.**

### **1.7.1 QT**

Es una biblioteca multiplataforma para desarrollar programas con interfaces gráficas de usuarios, aplicaciones de consola y servidores. Esta biblioteca basada en C++, se puede integrar a varios IDEs como son Microsoft Visual Studio y Eclipse entre otros. Disponible para sistemas operativos GNU/Linux, Mac OS X, Microsoft Windows entre otros.

### **1.7.2 Eclipse**

Eclipse es principalmente una plataforma de programación, usada para crear entornos integrados de desarrollo (IDE). Fue desarrollada originalmente por IBM, actualmente sigue siendo desarrollada por la Fundación Eclipse, una organización independiente que fomenta una comunidad de código abierto.

Eclipse es una plataforma IDE que basa su funcionamiento en módulos (en inglés plugin), para proporcionar todas sus funcionalidades, lo que lo hace muy versátil y portable a diferencia de otros IDEs. Este mecanismo de módulos es una plataforma ligera para componentes de software, lo que permite a Eclipse extenderse a varios lenguajes de programación como son Java, C/C++, PHP, Cobol, Python entre otros. Además permite utilizar lenguajes de procesamiento de texto como LaTeX, aplicaciones de red como Telnet, Sistemas de Gestión de Bases de Datos (DBMS). Para la gestión de la configuración y el control de versiones tiene soporte para CVS y Subversión, incluye plug-ins para realizar pruebas de unidad.



### 1.7.3 Micro C PRO for PIC

Es un compilador de C para microcontroladores de la familia PIC (Controlador de Interrupciones Programable) lanzado por **mikroElektronika**. El IDE se caracteriza por el diseño basado en proyectos y soporta un amplio rango de microcontroladores PIC. MikroC PRO ofrece una serie de bibliotecas que simplifican la inicialización y el uso de MCU PIC y sus módulos. Tiene una extensa documentación que permite iniciarse rápidamente en la programación de PIC así como varios ejemplos que les sirven de guía a usuarios que se inician en la programación de microcontroladores.

### 1.7.4 Proteus

Es un entorno integrado diseñado para la realización completa de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño, simulación, depuración y construcción. La suite se compone de cuatro elementos, perfectamente integrados entre sí.

Está compuesta por tres módulos o herramientas:

- **ISIS** (*Intelligent Schematic Input System*): Herramienta para la elaboración avanzada de esquemas electrónicos, que incorpora una librería de más de 6.000 modelos de dispositivos digitales y analógicos.
- **VSM** (*Virtual System Modelling*): Herramienta que permite incluir en la simulación de circuitos el comportamiento completo de los microcontroladores más conocidos del mercado.
- **ARES** (*Advanced Routing Modelling*): Herramienta para la elaboración de placas de circuito impreso con posicionado automático de elementos y generación automática de pistas, que permite el uso de hasta 16 capas. Con ARES el trabajo duro de la realización de placas electrónicas recae sobre la computadora en lugar de sobre el diseñador.

## 1.8 Conclusiones del capítulo.

En este capítulo se realizó un estudio de los sistemas de adquisición de datos, así como las tecnologías, metodologías y herramientas para mediciones multiconto, definiéndose aquellas que por sus características contribuyen al desarrollo del sistema.

## **CAPÍTULO 2 CARACTERÍSTICAS DEL SISTEMA**

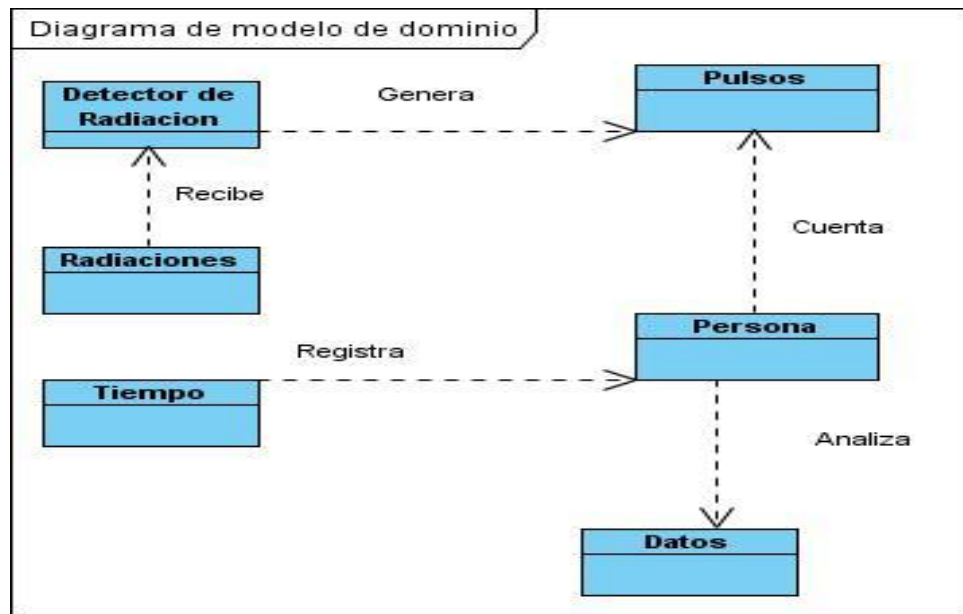
### **2.1 Introducción.**

Para la realización de un sistema que resuelva los problemas planteados en la situación problemática es necesario llegar a un entendimiento con el usuario con el objetivo de comprender el contexto y las condiciones o capacidades que debe tener dicho sistema. En este capítulo se comienza a tener una visión más clara del sistema que será desarrollado, recoge los requerimientos funcionales y no funcionales que el mismo debe cumplir, el modelo del dominio que aporta una idea inicial de cómo se realizan los procesos que requieren automatización.

El sistema a desarrollar está formado por tres entidades fundamentales: el hardware, el firmware y el software. El hardware, al que se hace referencia también como *dispositivo MC-mcSystem* o *tarjeta de adquisición*, es el encargado de la interconexión física entre la instrumentación nuclear y la computadora, el firmware, interfaz entre el hardware y el software, es el encargado del funcionamiento a bajo nivel del sistema y el software es la interfaz del sistema para el usuario.

### **2.2 Modelo del dominio.**

Para lograr un mejor entendimiento de los procesos que requieren automatización se realizó un modelo del dominio, el cual recoge y describe los conceptos más importantes dentro del contexto del sistema así como las relaciones entre ellos.



**Figura 4:** Modelo del dominio.

**Detector de radiaciones:** Dispositivo utilizado para detectar las radiaciones emitidas por sustancias en estado de desintegración.

**Pulsos:** Señales emitidas por el detector de radiaciones.

**Radiaciones:** Energía emitida por el núcleo al desintegrarse por medio de partículas o radiación electromagnética.

**Tiempo:** Intervalo de tiempo durante el cual se cuentan los pulsos emitidos por el detector de radiaciones.

**Persona:** Encargada de realizar los proceso de registro de los pulsos, tiempo y el análisis de los datos.

## 2.3 Especificación de los requerimientos del sistema.

### 2.3.1 Requerimientos funcionales.

Los requerimientos funcionales no son más que aquellas funcionalidades que el sistema debe tener, o sea, es todo lo que debe hacer para satisfacer las necesidades impuestas por un cliente o usuario final.

A continuación se muestran los requerimientos funcionales<sup>1</sup> que el sistema a desarrollar debe cumplir.

**RF1: Contar pulsos digitales provenientes del detector.** El sistema debe permitir al usuario contar la mayor cantidad posible de conteos por cada canal.

**RF2: Controlar intervalos temporales variables con gran exactitud.** El sistema debe contar con un reloj de tiempo real que permita programar intervalos temporales con gran exactitud y ser variables desde los milisegundos hasta los días.

**RF3: Establecer comunicación entre el hardware la computadora.** El sistema debe permitir al usuario configurar y establecer la comunicación entre el hardware y la computadora.

**RF4: Configurar la adquisición.** El sistema debe ser capaz de permitir al usuario configurar el tiempo de residencia y el número de intervalos que desea adquirir del dispositivo.

**RF5: Controlar la adquisición.** El sistema debe controlar los instantes de inicio y parada de la adquisición según la lógica del método multiconto.

**RF6: Visualizar datos adquiridos.** El sistema debe permitir visualizar los datos adquiridos en una gráfica con escala lineal, logarítmica o cuadrática.

**RF7: Guardar datos adquiridos.** El sistema debe brindar al usuario la posibilidad de guardar los datos adquiridos en un dispositivo de almacenamiento.

**RF8: Cargar datos de dispositivo extraíble.** El sistema debe permitir cargar los guardados en un dispositivo de almacenamiento para su visualización.

### 2.3.2 Requerimientos no funcionales.

Los requerimientos no funcionales no son más que aquellas cualidades o propiedades que el sistema debe tener. A continuación se muestran los requerimientos no funcionales para el sistema a desarrollar.

---

<sup>1</sup> Los requerimientos funcionales uno y dos están relacionados el hardware por tal razón no están referenciados en ningún caso de uso.

## **Usabilidad**

El sistema está destinado a usuarios especializados o con conocimientos básicos en elementos de física nuclear, debe ser fácil de usar incluso para aquellos que no tengan altos conocimientos informáticos, además debe contar con una ayuda para facilitar su uso.

## **Fiabilidad**

El sistema debe detectar si hay fallos en la conexión con el dispositivo e informar del error inmediatamente, además de garantizar la integridad de los datos adquiridos.

## **Funcionamiento**

El sistema debe tener un tiempo de reacción lo suficientemente rápido que permita adquirir en la mayor velocidad propuesta en el diseño. El objetivo es obtener un sistema que permita medir para tiempos de residencias tan rápidos como 1ms, con un margen de error mínimo. Además debe permitir contar más de un millón de pulsos por canal.

## **Soporte**

El sistema debe estar bien documentado, de forma tal que el tiempo de mantenimiento sea mínimo en caso de ser necesario. Debe contar con una ayuda que facilite a los usuarios el uso del mismo.

## **Exigencias de Personalización**

El sistema debe permitir al usuario personalizar el contenido a visualizar en pantalla, colores de la gráfica, estado de la comunicación y adquisición del dispositivo y mostrar el contenido de acuerdo a los cambios realizados por el mismo.

## **Interfaces**

El sistema mantendrá comunicación entre el dispositivo y la computadora mediante la interfaz serie RS-232.

## 2.4 Actores del sistema.

Tabla 1: Actor del sistema.

Actor	Descripción
Usuario	Persona que interactúa con el software, es el encargado de configurar y realizar la adquisición y todos los procesos relacionados con esta.

## 2.5 Diagrama de casos de uso del sistema.

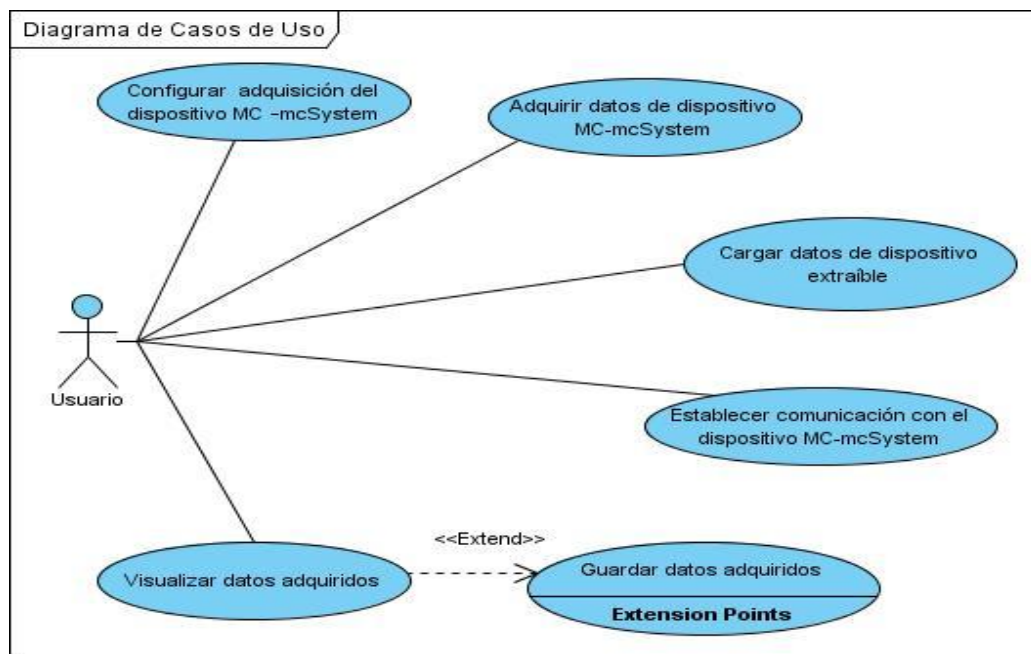


Figura 5: Diagrama de casos de uso del software.

## 2.6 Descripción de los casos de uso del software.

**Tabla 2:** Caso de Uso Establecer comunicación con el dispositivo.

<b>Caso de Uso:</b>	Establecer comunicación con el dispositivo	
<b>Actores:</b>	Usuario	
<b>Resumen:</b>	El caso de uso inicia cuando el usuario necesita configurar la comunicación del sistema con el dispositivo.	
<b>Precondiciones:</b>		
<b>Referencias</b>	RF3	
<b>Prioridad</b>	Crítico	
<b>Flujo Normal de Eventos</b>		
<b>Acción del Actor</b>	<b>Respuesta del sistema</b>	
1. El usuario selecciona la opción preferencias en el menú ver.	1.1 El sistema muestra una ventana con varias pestañas.	
2. El usuario pulsa la pestaña "General".	2.1 El sistema muestra los datos correspondientes a la pestaña "General".	
3. El usuario llena los datos correspondientes (Puerto).	3.1 El sistema verifica que no se esté realizando una adquisición.	
	4. El sistema establece la comunicación con el dispositivo.	
<b>Flujos Alternos</b>		
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>	

	3.1 En caso que se esté realizando una adquisición el sistema muestra un mensaje indicando que se está realizando una adquisición, concluyendo así el caso de uso.
	4. En caso de que no se pueda establecer comunicación con el dispositivo muestra un mensaje indicando que hubo un error en la comunicación.
<b>Poscondiciones</b>	Establecida la comunicación del sistema con el dispositivo MC-mcSystem.

**Tabla 3:** Caso de Uso Configurar la adquisición del dispositivo MC –mcSystem.

<b>Caso de Uso:</b>	Configurar la adquisición del dispositivo MC –mcSystem
<b>Actores:</b>	Usuario
<b>Resumen:</b>	El caso de uso se inicia cuando el usuario desea configurar el modo en que se adquieren los datos del dispositivo MC –mcSystem.
<b>Precondiciones:</b>	Debe estar establecida la comunicación con el dispositivo MC-mcSystem
<b>Referencias</b>	RF4
<b>Prioridad</b>	Crítico
<b>Flujo Normal de Eventos</b>	



<b>Acción del Actor</b>		<b>Respuesta del Sistema</b>
1. El usuario selecciona la opción preferencias en el menú ver.		1.1 El sistema muestra una ventana con varias pestañas.
2. El usuario pulsa la pestaña “Adquisición”.		2.1 El sistema muestra los datos correspondientes a la pestaña “Adquisición”.
3. El usuario llena los datos correspondientes (tiempo de residencia, cantidad de conteos).		3.1 El sistema verifica si se está realizando una adquisición.
		4. El sistema configura el modo de adquisición del dispositivo MC – mcSystem.
<b>Flujos Alternos</b>		
<b>Acción del Actor</b>		<b>Respuesta del Sistema</b>
		3.1 En caso que se esté realizando una adquisición el sistema muestra un mensaje indicando que se está realizando una adquisición, concluyendo así el caso de uso.
<b>Poscondiciones</b>	Configurado la adquisición del dispositivo MC-mcSystem.	

**Tabla 4:** Caso de Uso Adquirir datos de dispositivo MC-mcSystem.

<b>Caso de Uso:</b>	Adquirir datos de dispositivo MC-mcSystem
<b>Actores:</b>	Usuario

<b>Resumen:</b>	El caso de uso inicia cuando el usuario desea adquirir datos del dispositivo MC-mcSystem.	
<b>Precondiciones:</b>	Debe estar configurado el modo de adquisición del dispositivo MC-mcSystem.	
<b>Referencias</b>	RF5	
<b>Prioridad</b>	Crítico	
<b>Flujo Normal de Eventos</b>		
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>	
1. El usuario selecciona la opción “Iniciar adquisición” del menú “Adquisición”	1.1 El sistema verifica que no se esté realizando una adquisición o existan datos visualizados sin salvar.	
	1.2 El sistema comienza adquirir datos del dispositivo.	
<b>Flujos Alternos</b>		
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>	
	1.2 En caso que se esté realizando una adquisición o existan datos visualizados sin salvar, el sistema muestra un mensaje indicando que no se puede realizar la adquisición, concluyendo así el caso de uso.	
<b>Poscondiciones</b>	El sistema se encuentra adquiriendo datos del dispositivo MC-mcSystem.	

**Tabla 5:** Caso de Uso Visualizar datos adquiridos.

<b>Caso de Uso:</b>	Visualizar datos adquiridos	
<b>Actores:</b>	Usuario	
<b>Resumen:</b>	El caso de uso inicia cuando el usuario desea visualizar o actualizar la visualización de los datos adquiridos.	
<b>Precondiciones:</b>	Deben haberse adquirido datos anteriormente.	
<b>Referencias</b>	RF6	
<b>Prioridad</b>	Crítico	
<b>Flujo Normal de Eventos</b>		
<b>Acción del Actor</b>		<b>Respuesta del Sistema</b>
1.El usuario selecciona la opción “Actualizar gráfica” del menú “Adquisición”		1.1 Muestra una gráfica con los datos adquiridos.
		2. El sistema brinda las siguientes opciones: -Seleccionar escala de visualización.
<b>Sección “Seleccionar escala de visualización”</b>		
<b>Acción del Actor</b>		<b>Respuesta del Sistema</b>
1. El usuario selecciona la escala con la que desea visualizar los datos (lineales, logarítmicos, cuadráticos).		1.1 El sistema visualiza los datos de acuerdo a la escala seleccionada.
<b>Poscondiciones</b>	El sistema visualiza los datos adquiridos.	

**Tabla 6:** Caso de Uso Guardar datos adquiridos.

<b>Caso de Uso:</b>	Guardar datos adquiridos	
<b>Actores:</b>	Usuario	
<b>Resumen:</b>	El caso de uso inicia cuando el usuario desea guardar los datos adquiridos para analizarlos en otro momento.	
<b>Precondiciones:</b>	Deben existir datos adquiridos	
<b>Referencias</b>	RF7	
<b>Prioridad</b>	Secundario	
<b>Flujo Normal de Eventos</b>		
	<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>
	1. El usuario selecciona la opción “Guardar” del menú “Archivo”	1.1 El sistema muestra una ventana dándole la posibilidad de escoger la ruta donde va a guardar los datos adquiridos.
	2. El usuario escoge la ruta y el nombre del fichero donde desea guardar los datos adquiridos y pulsa la en el botón “Guardar”	2.1 El sistema guarda los datos dentro del fichero seleccionado por el usuario.
<b>Poscondiciones</b>		

**Tabla 7:** Caso de Uso Cargar datos de dispositivo extraíble.

<b>Caso de Uso:</b>	Cargar datos de dispositivo extraíble
<b>Actores:</b>	Usuario

<b>Resumen:</b>	El caso de uso inicia cuando el usuario desea cargar los datos de una adquisición guardados de un dispositivo extraíble.	
<b>Precondiciones:</b>		
<b>Referencias</b>	RF8	
<b>Prioridad</b>	Secundario	
<b>Flujo Normal de Eventos</b>		
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>	
1. El usuario selecciona la opción "Abrir" del menú "Archivo"	1.1 El sistema verifica que no se esté realizando una adquisición.	
	2. El sistema muestra una ventana dándole la posibilidad al usuario de escoger la ubicación donde se encuentran los datos.	
3. El usuario busca la dirección donde se encuentran ubicados los datos y pulsa el botón "Abrir".	3.1 El sistema carga los datos situados en la dirección seleccionada por el usuario.	
<b>Flujos Alternos</b>		
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>	
	1.1 En caso que se esté realizando una adquisición el sistema muestra un mensaje indicando que se está realizando una adquisición, concluyendo así el caso de uso.	

**Poscondiciones**

El sistema carga los datos existentes en el archivo seleccionado.

**2.7 Conclusiones del capítulo.**

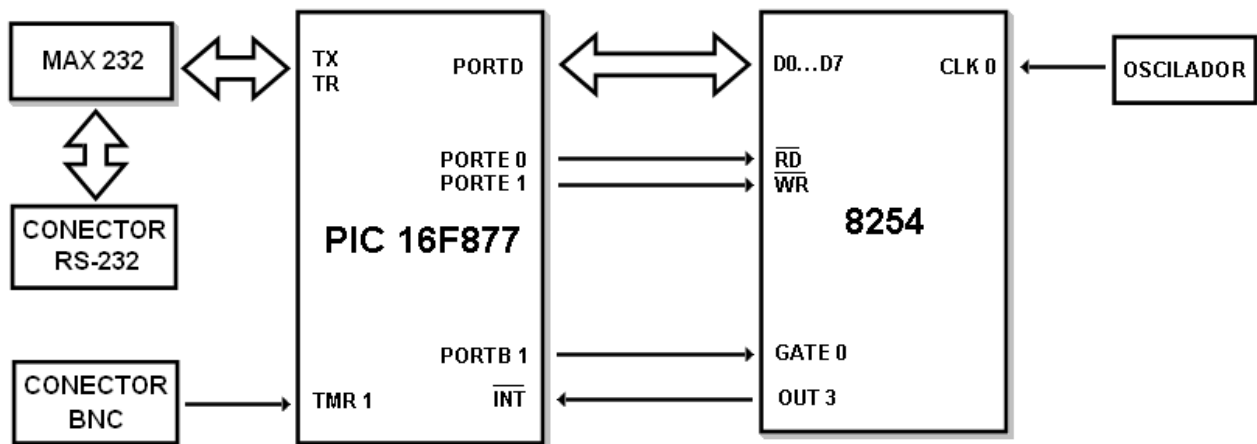
En este capítulo se definieron las funcionalidades que debe tener el sistema para dar solución a los objetivos propuestos, así como una descripción de los casos de uso, que sirven de base a los capítulos posteriores.

## CAPÍTULO 3 CARACTERÍSTICAS DEL HARDWARE.

En este capítulo se hará referencia al hardware y se describirá el diseño e implementación del firmware. Además se abordarán algunos conceptos relacionados con el desarrollo del mismo.

### 3.1 Descripción del hardware.

El hardware del sistema está conformado por la tarjeta de adquisición MC-mcSystem desarrollada en el InSTEC. El diagrama de bloques se muestra en la figura 6.



*Figura 6: Diagrama de bloques del hardware.*

#### 3.2.1 Componentes.

- Temporizador de Intervalos Programable PIT 8254.
- Microcontrolador PIC16F877.
- MAX 232.
- Conector RS-232.
- Conector BNC.
- Oscilador de 4MHz.

## **Microcontrolador PIC16F877**

El PIC16F877 pertenece a la familia de microcontroladores PIC (Controlador de Interfaz Periférico) fabricados por la empresa **Microchip Technology Inc.** Este microcontrolador posee varias características que hacen que sea un dispositivo muy eficiente y práctico. Entre las principales características o funcionalidades que brinda está el soporte a la comunicación serial por la interfaz RS232, muy importante ya que es la interfaz que se utilizará en el sistema. Posee además temporizadores programables que pueden ser utilizados si se quiere medir períodos de tiempos, o simplemente contar eventos. También posee conversores análogos-digital en caso que se requiera medir señales analógicas, cuenta con un set de instrucciones reducido tipo RISC, además posee cuatro puertos que pueden configurarse como entrada o salida digital que le permiten controlar varios periféricos a la vez.

## **Temporizador de Intervalos Programable PIT 8254**

El Temporizador de Intervalos Programable 8254 conocido como PIT 8254 es un chip temporizador que puede ser empleado como reloj de tiempo real, contador de sucesos, generador de ritmo programable, generador de onda cuadrada, etc. Este temporizador tiene tres contadores o timer de 16 bit que pueden ser programados de forma independiente en seis modos de operación diferentes. Unos de los modos de operación para lo que se puede programar los contadores del 8254 es el Modo 2: Temporizador de intervalos periódicos. En este modo el contador funciona como un reloj de tiempo real, el cual después de haber sido inicializado con un valor determinado, irá decrementando este con cada pulso a la entrada hasta llegar a cero.

En el diseño de la tarjeta MC-mcSystem, un generador de pulsos construido con un oscilador a una frecuencia de 4MHz está conectado a la entrada del primer contador. Los tres contadores son conectados en cascada y programados en Modo 2. La salida del tercer contador es conectada a la entrada de interrupción externa del PIC. De esta forma, el dispositivo permite generar con exactitud intervalos de tiempo desde los microsegundos hasta varios días.



### 3.3 Descripción del Firmware

Un firmware no es más que un programa que se inserta en la memoria ROM de un dispositivo de hardware. Funcionalmente es una interfaz entre las órdenes externas que recibe un dispositivo y su electrónica. En el caso de nuestro sistema, el firmware es el programa que se ejecutará en el PIC16F877.

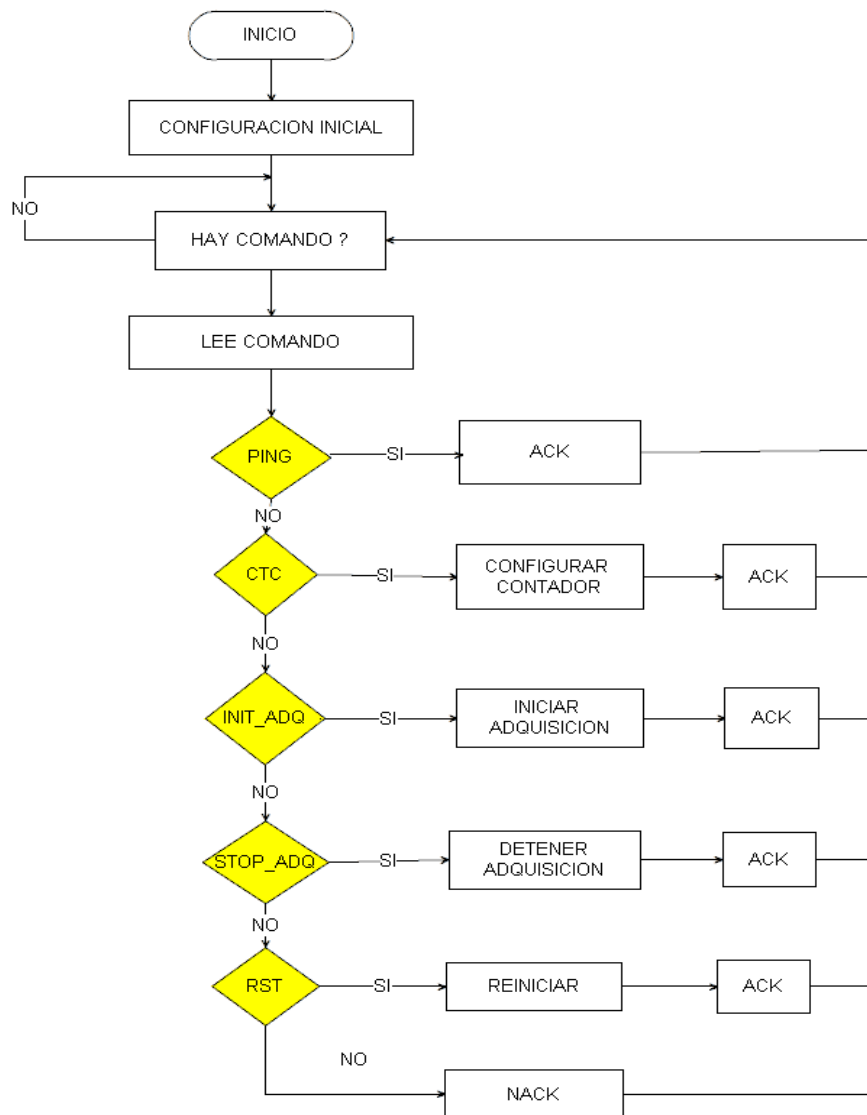


Figura 7: Diagrama de flujo del firmware.

### 3.3.1 Diseño e implementación del firmware.

El firmware está diseñado siguiendo el modelo de programación orientada a eventos, donde este queda a la espera de la ocurrencia de eventos (comandos recibidos, interrupciones de periféricos) y ejecuta la acción (subrutina) correspondiente. La figura 7 se muestra el diagrama de flujo de la subrutina principal del firmware implementado:

El programa comienza con la inicialización del sistema. Aquí se configuran los puertos que van a hacer utilizados por el microcontrolador para la comunicación con los periféricos, la comunicación por el puerto serie y los vectores de interrupción. Después de la inicialización, se pasa a un ciclo infinito en espera de eventos. Un evento no es más que la llegada de un comando enviado por el software.

#### Lista de comandos y subrutina asociada.

**PING:** *Comando de control de comunicación.* Utilizado por el software para poder chequear la comunicación con el sistema.

**CTC:** *Comando de programación del PIT.* La subrutina asociada a este comando espera del puerto serie los seis valores necesarios para la configuración del PIT 8254, estos valores son guardados en variables temporales para posteriormente ser pasados al PIT.

**INIT\_ADQ:** *Comando de inicio de adquisición.* Los valores de configuración guardados en las variables temporales son pasados al PIT 8254. Son habilitadas las entradas de conteo del TIMER1 y del 8254, así como las interrupciones externas e internas que posibilitan capturar las señales o pulsos enviados por el PIT 8254 y el detector de radiaciones.

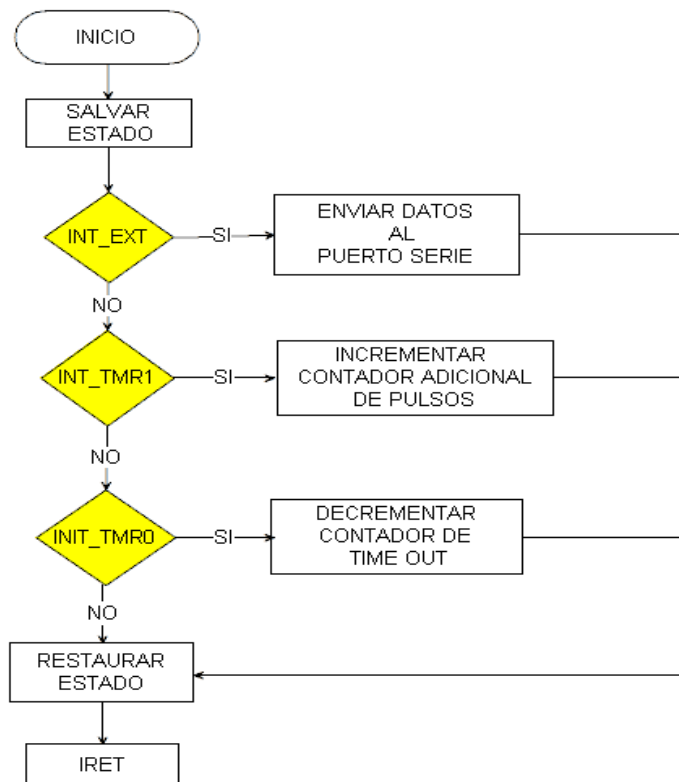
**STOP\_ADQ:** *Comando de fin de adquisición:* Son detenidos el PIT 8254 y el TIMER1 y deshabilitadas las interrupciones externas e internas.

**RST:** *Comando reiniciar.* Utilizado para restaurar el estado de los valores a su configuración inicial.

### 3.3.2 Servicio de interrupciones.

Una interrupción es una señal recibida por el procesador que indica que debe interrumpir el curso de ejecución actual y pasar a ejecutar el código específico para tratar esta situación. En el modelo propuesto, una interrupción también es un evento. El PIC16F877 cuenta con un total de 14 interrupciones que le permiten controlar varios periféricos y enviar información de este al procesador. A continuación se enuncian las principales interrupciones utilizadas en el desarrollo del firmware y una breve descripción de las mismas.

#### Diagrama de flujo general del servicio de atención a interrupciones.

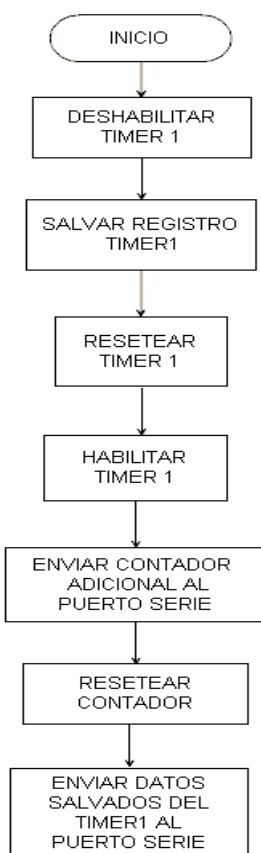


**Figura 8:** Diagrama de flujo general del servicio de atención a interrupciones.

**Interrupción Timer1 (T1IE):** Esta interrupción está relacionada con el desbordamiento del registro del TIMER1 y es utilizada en el proceso de almacenar en memoria la cantidad de pulsos enviados por el detector de radiaciones. Controlar esta interrupción permite aumentar la capacidad de conteo del

sistema mediante una variable auxiliar, ya que el TIMER1, de 16 bits, sólo nos permite contar un número limitado de eventos en el detector. Con el uso de esta variable se puede incrementar la cantidad de conteos desde 65536 hasta más de 10 000 000.

**Interrupción Externa (INTF):** Esta interrupción es generada por el PIT 8254, la misma se activa cada vez que llegue a cero el valor del tiempo programado en los contadores del PIT 8254. En la subrutina de atención de esta interrupción se procede a enviar el número de pulsos adquiridos del detector de radiaciones al puerto serie de la computadora.



**Figura 9:** Diagrama de flujo general del servicio de atención de la interrupción externa.

Las subrutinas de atención a interrupción son críticas en tiempo, en el desarrollo del firmware se utilizó la interrupción externa para controlar el tiempo que demora el PIC en realizar un cambio de canal. Para hacer el error del cambio de canal lo menor posible debido a esta demora, se implementó la subrutina de atención a interrupción en lenguaje ensamblador, a diferencia del resto del firmware que fue programado en C.

### **3.4 Conclusiones del capítulo.**

En este capítulo se realizó una descripción del hardware, se explicaron de forma breve pero concisa los elementos que lo conforman y algunos conceptos necesarios para un mejor entendimiento haciendo mayor énfasis en el desarrollo del firmware.

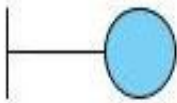

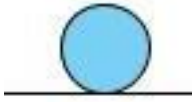
## CAPÍTULO 4 ANÁLISIS, DISEÑO E IMPLEMENTACION DEL SOFTWARE

### 4.1 Introducción.

Este capítulo recoge algunos aspectos relacionados con el desarrollo del software para la visualización de los datos adquiridos por la tarjeta de adquisición MC-mcSystem, abordándose aspectos como la arquitectura y patrones de diseño utilizados. Se exponen los diagramas de clases del análisis y diseño, así como la descripción de las principales clases del diseño. También se abordan aspectos del modelo de implementación, como el diagrama de componentes y despliegue.

### 4.2 Análisis.

#### 4.2.1 Diagrama de clases del análisis.

TIPO DE CLASE	ESTEREOTIPO	DESCRIPCION
Interfaz		Modelan la interacción entre el software y sus actores.
Controladora		Coordinan la realización de uno o unos pocos casos de uso coordinando las actividades de los objetos que implementan la funcionalidad del caso de uso.
Entidad		Modelan información que posee larga vida y que es a menudo persistente.

### Caso de Uso Establecer comunicación con el dispositivo MC-mcSystem.

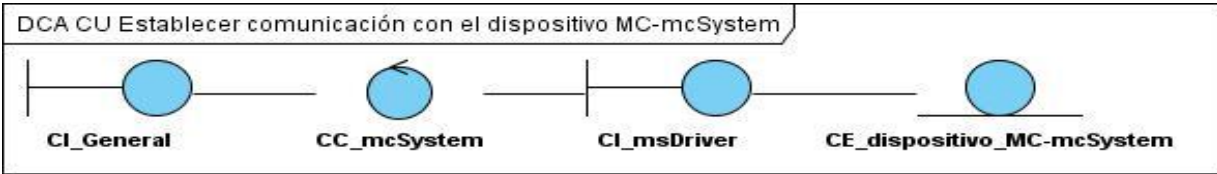


Figura 10: Diagrama de clases del análisis del CU Establecer comunicación con el dispositivo MC-mcSystem.

### Caso de Uso Configurar adquisición del dispositivo MC -mcSystem.

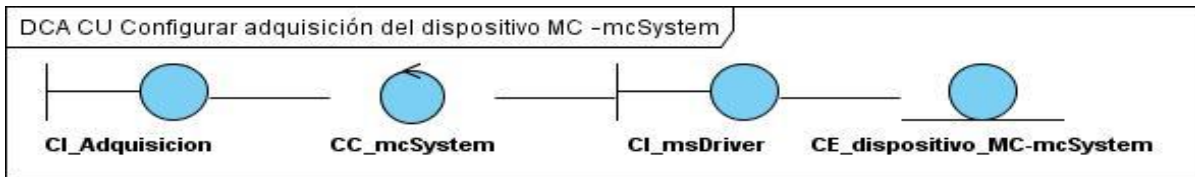


Figura 11: Diagrama de clases del análisis del CU Configurar adquisición del dispositivo MC-mcSystem.

### Caso de Uso Adquirir datos de dispositivo MC -mcSystem.

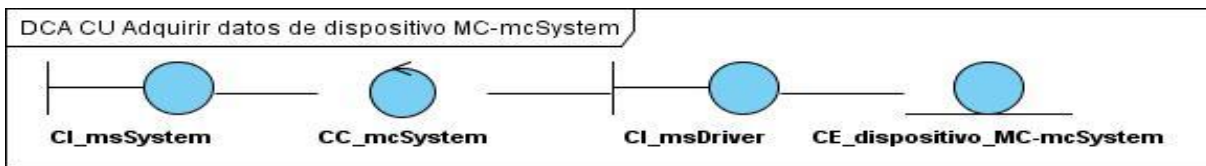


Figura 12: Diagrama de clases del análisis del CU adquirir datos de dispositivo MC -mcSystem.

### Caso de Uso Visualizar datos adquiridos.

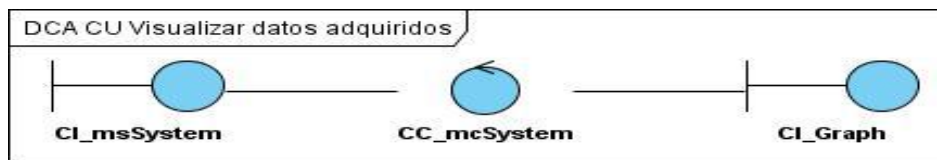
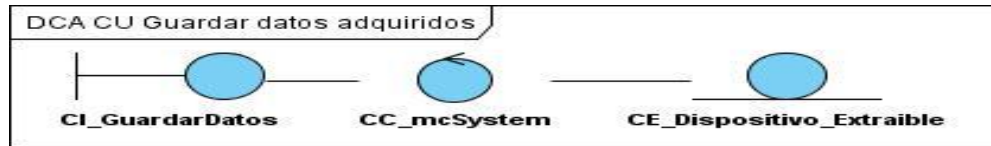


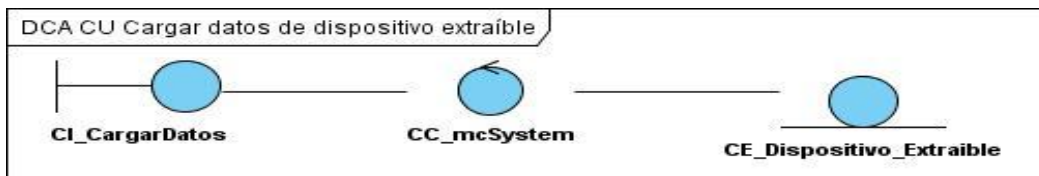
Figura 13: Diagrama de clases del análisis del CU Visualizar datos adquiridos.

### Caso de Uso Guardar datos adquiridos.



*Figura 14: Diagrama de clases del análisis del CU Guardar datos adquiridos.*

### Caso de Uso Cargar datos de dispositivo extraíble.

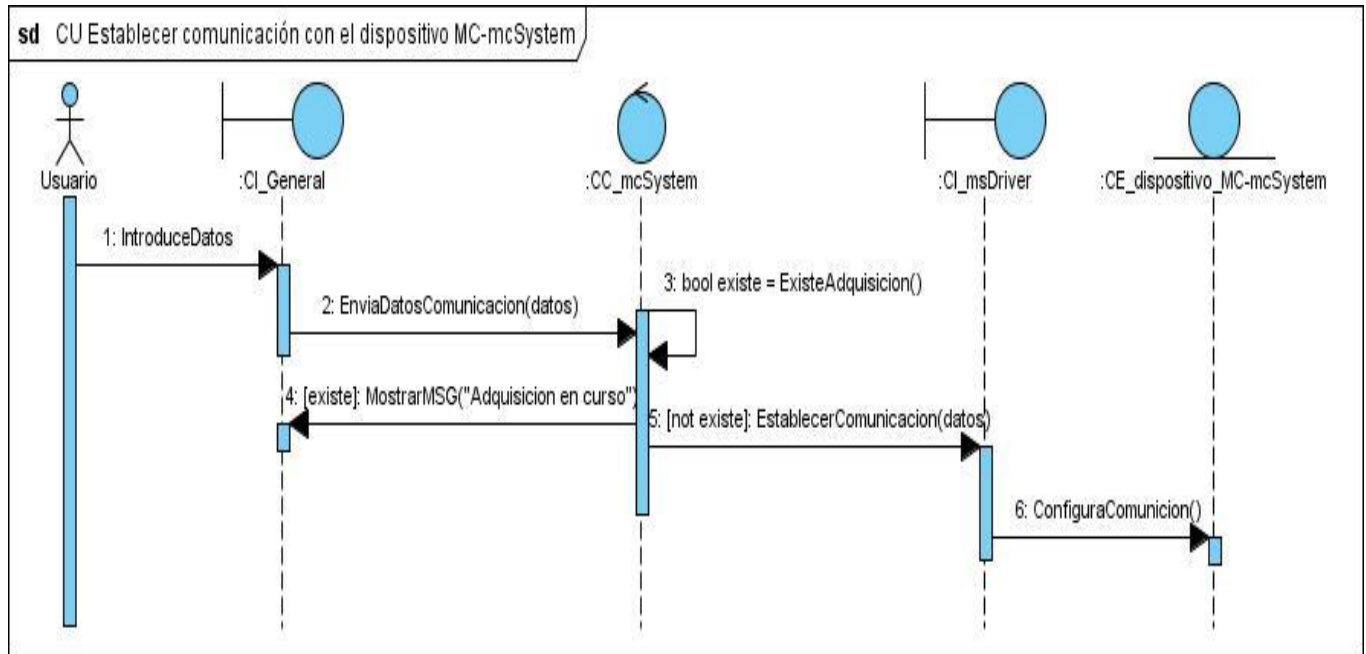


*Figura 15: Diagrama de clases del análisis del CU cargar datos de dispositivo extraíble.*



## 4.2.2 Diagrama de iteración.

### Caso de Uso Establecer comunicación con el dispositivo MC-mcSystem.



**Figura 16:** Diagrama de secuencia del CU Establecer comunicación con el dispositivo MC-mcSystem.

### Caso de Uso Configurar la adquisición del dispositivo MC –mcSystem.

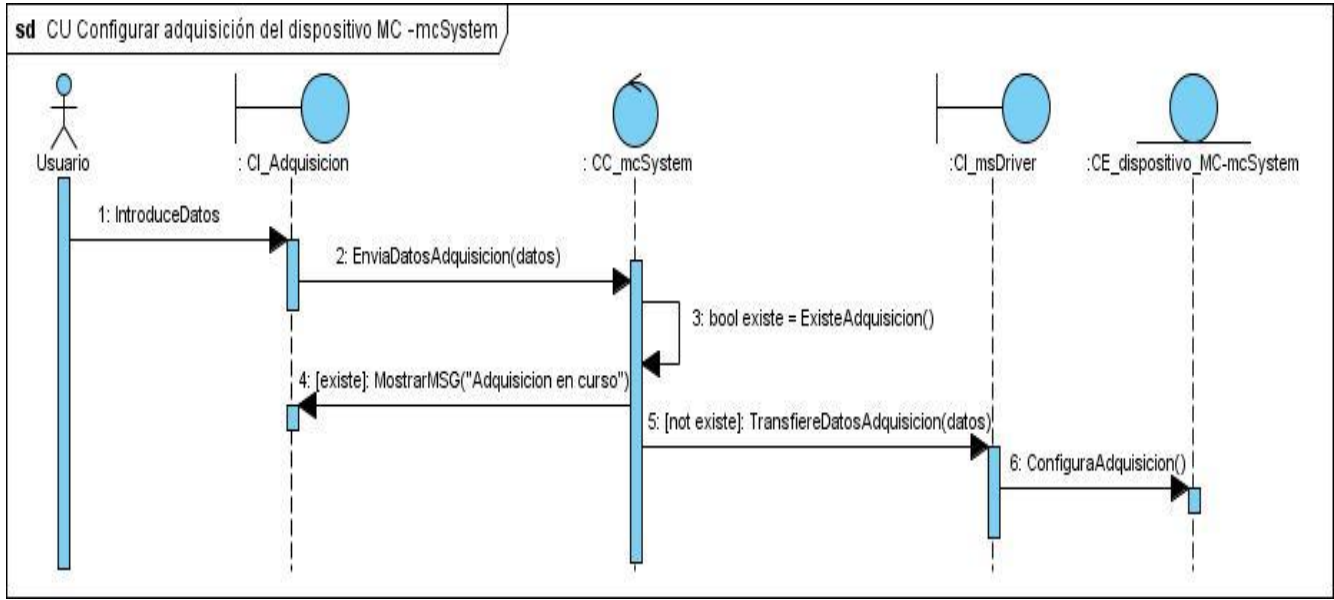


Figura 17: Diagrama de secuencia del CU Configurar la adquisición del dispositivo MC –mcSystem.

### Caso de Uso Adquirir datos de dispositivo MC-mcSystem.

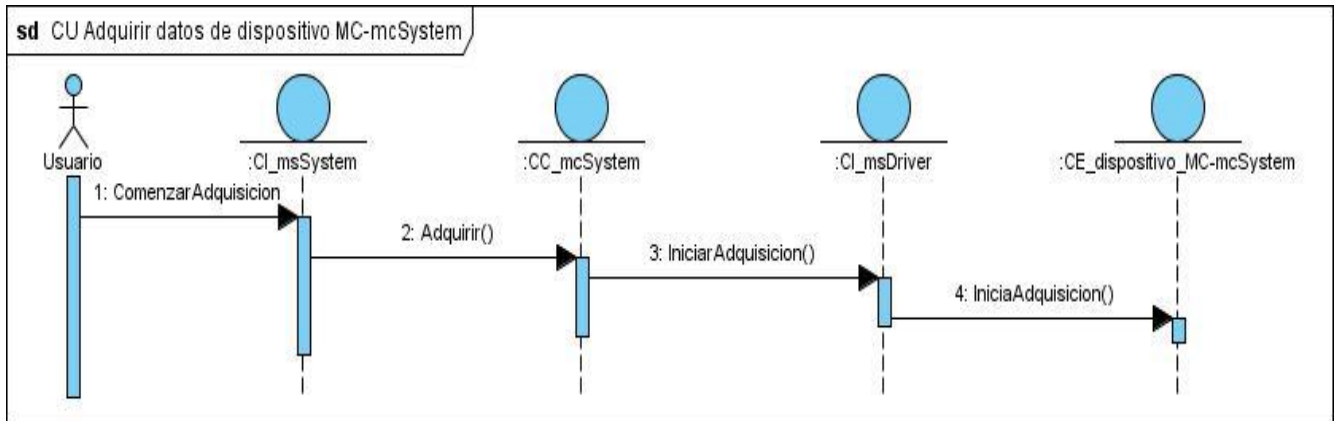


Figura 18: Diagrama de secuencia del CU Adquirir datos de dispositivo MC-mcSystem.

### Caso de Uso Visualizar datos adquiridos.

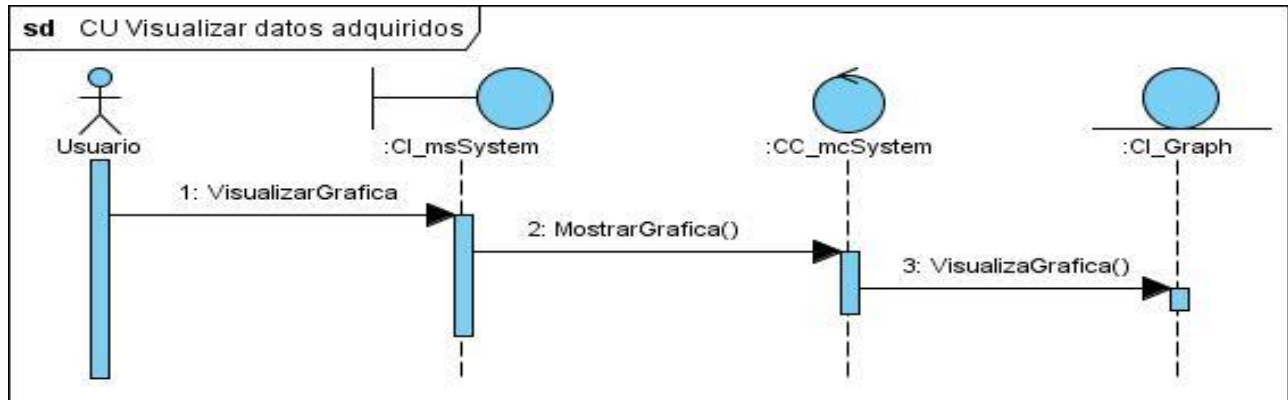


Figura 19: Diagrama de secuencia del CU Visualizar datos adquiridos.

### Caso de Uso Guardar datos adquiridos.

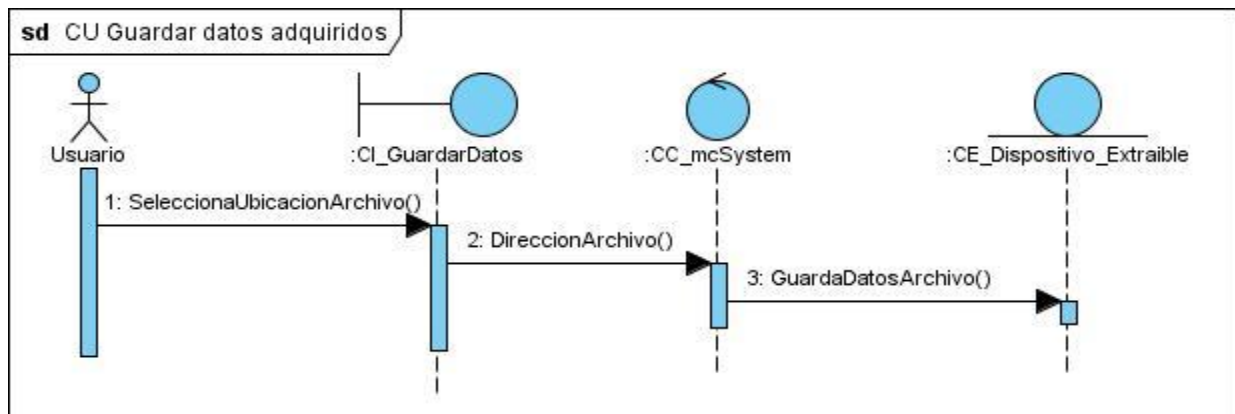


Figura 20: Diagrama de secuencia del CU Guardar datos adquiridos.

## Caso de Uso Cargar datos de dispositivo extraíble.

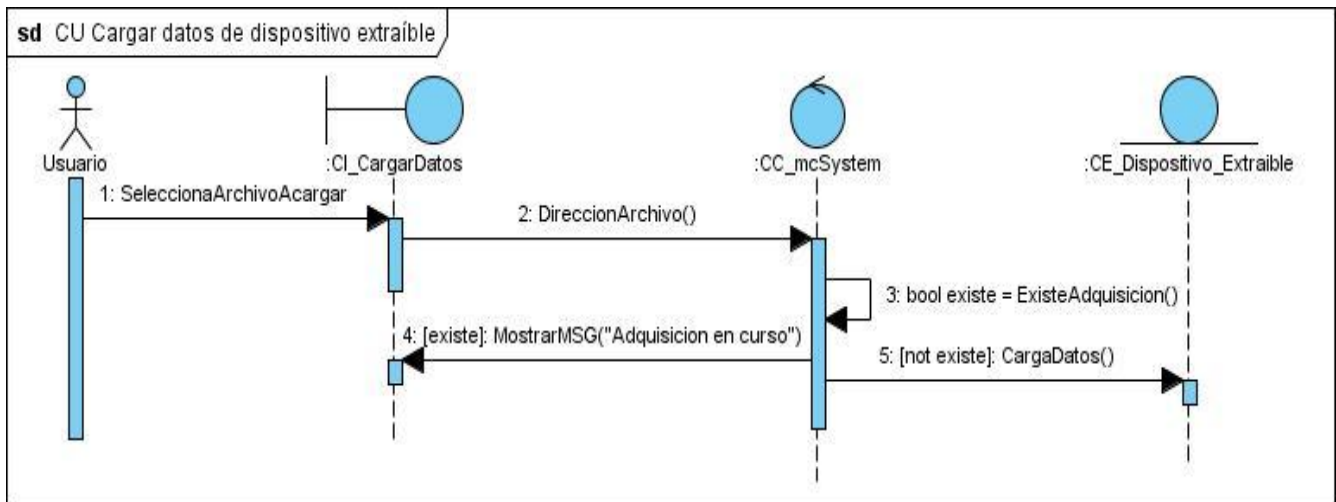


Figura 21: Diagrama de secuencia del CU Cargar datos de dispositivo extraíble.

### 4.3 Diseño.

#### 4.3.1 Arquitectura y patrones utilizados.

##### Arquitectura.

Una arquitectura es un conjunto de decisiones sobre la organización de un sistema para contribuir a una mejor comprensión, organización y reutilización del mismo.

En el diseño del software se hizo uso de la *arquitectura basada en componentes*. Esta arquitectura se enfoca en la descomposición del diseño en componentes funcionales o lógicos que expongan interfaces de comunicación bien definidas. Esto provee un nivel de abstracción mayor que los principios de orientación por objetos y no se enfoca en asuntos específicos de los objetos como los protocolos de comunicación y la forma como se comparte el estado. [11]

Los principales beneficios que aporta el uso de este estilo de arquitectura es que al estar el software dividido en pequeños componentes con una dependencia mínima entre ellos, se garantiza la reusabilidad y extensibilidad de estos, facilitando a su vez el desarrollo, instalación y mantenimiento del software.

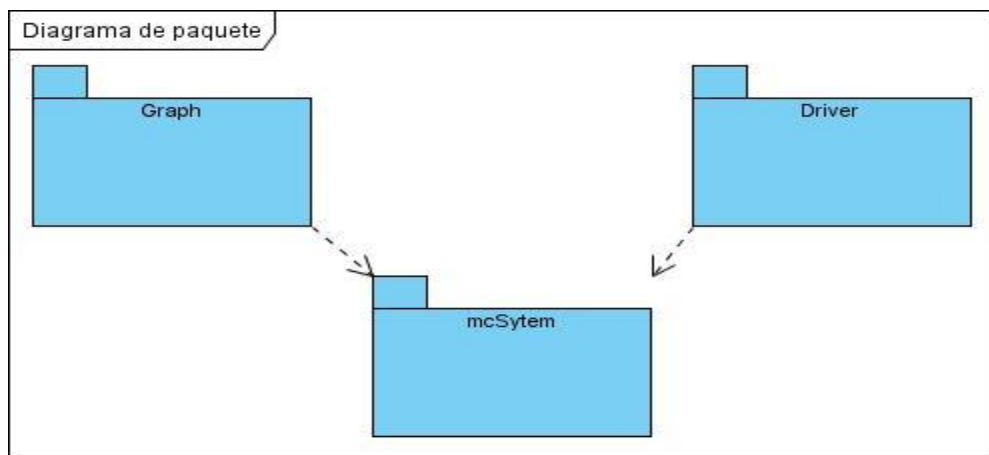
## Patrones de diseño.

Un patrón en la ingeniería de software es una solución a un problema en el desarrollo de software. Dicha solución ha sido probada y documentada y puede ser aplicada a otros problemas. A continuación se muestran los patrones utilizados en el diseño del sistema.

### Patrones GRASP

- **Experto:** Propone como solución asignar una responsabilidad al experto en información o sea la clase que posee la información necesaria para cumplir con la responsabilidad.
- **Creador:** Propone como solución la asignación a la clase A de crear una instancia de la clase B, en diseño de clase dicha responsabilidad la tiene la clase mcSystem ya que es la que usa las clases MCAGraph y MSAdq.
- **Controlador:** Propone como solución asignar a una clase la responsabilidad de administrar los mensajes de eventos del sistema. Esta responsabilidad la tiene la clase mcSystem.
- **Bajo acoplamiento:** Propone como solución asignar las responsabilidades de modo que se mantenga bajo acoplamiento. En el diseño se tuvo en cuenta este patrón para disminuir el número de dependencia fuerte entre las clases.
- **Alta cohesión:** Propone como solución asignar las responsabilidades de modo que se mantenga una alta cohesión. En el diseño de las clases se trató de asignarle el menor número de operaciones a una clase, logrando que fuese más fácil el mantenimiento y mejoramiento de la misma.

### 4.3.2 Diagrama de paquete.



**Figura 22:** Diagrama de paquete.

### 4.3.3 Diagrama de clase del diseño.

#### 4.3.3.1 Diagrama de clases subsistema Graph.

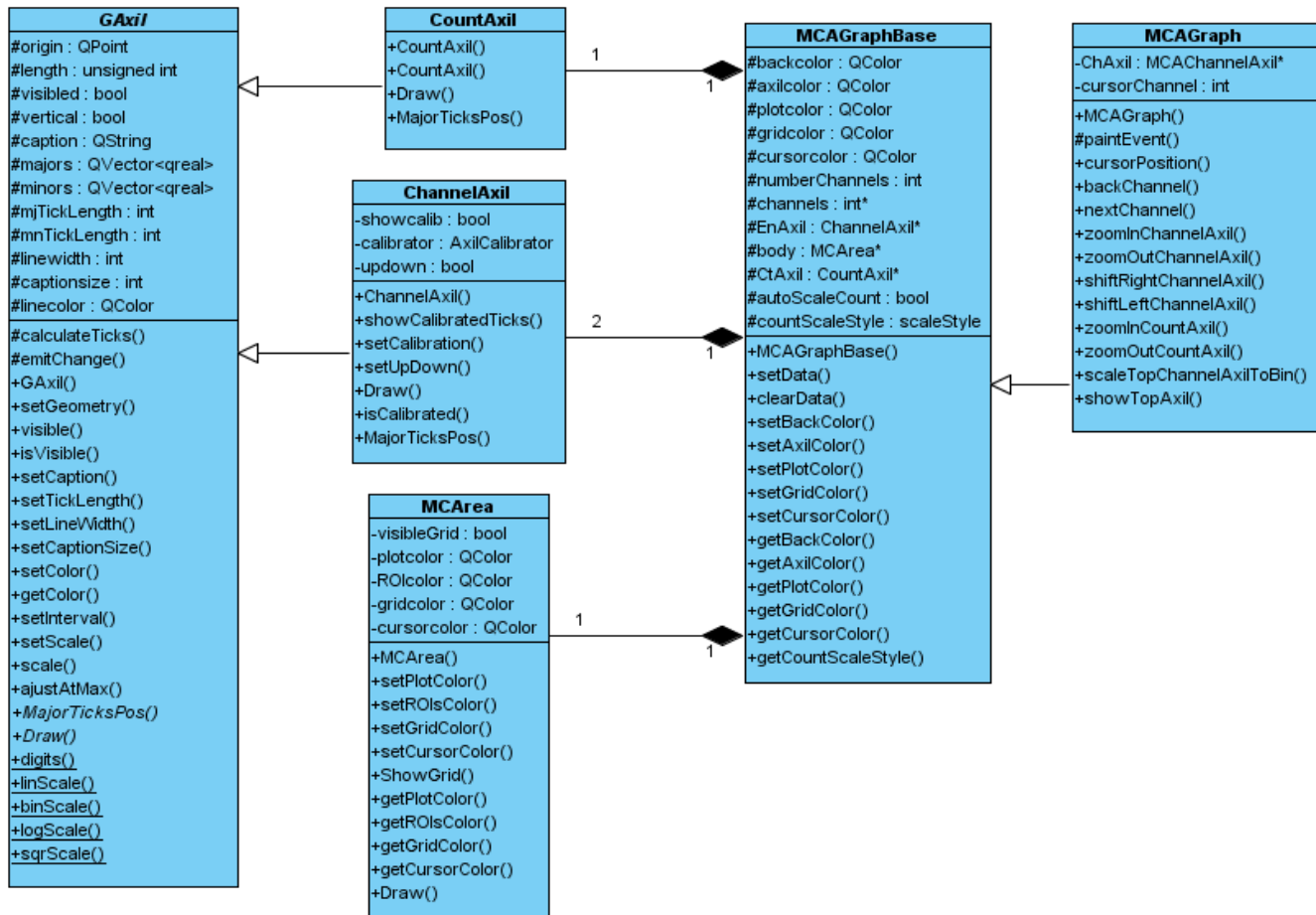


Figura 23: Diagrama de clases del diseño subsistema Graph.

### 4.3.3.2 Diagrama de clases subsistema Driver.

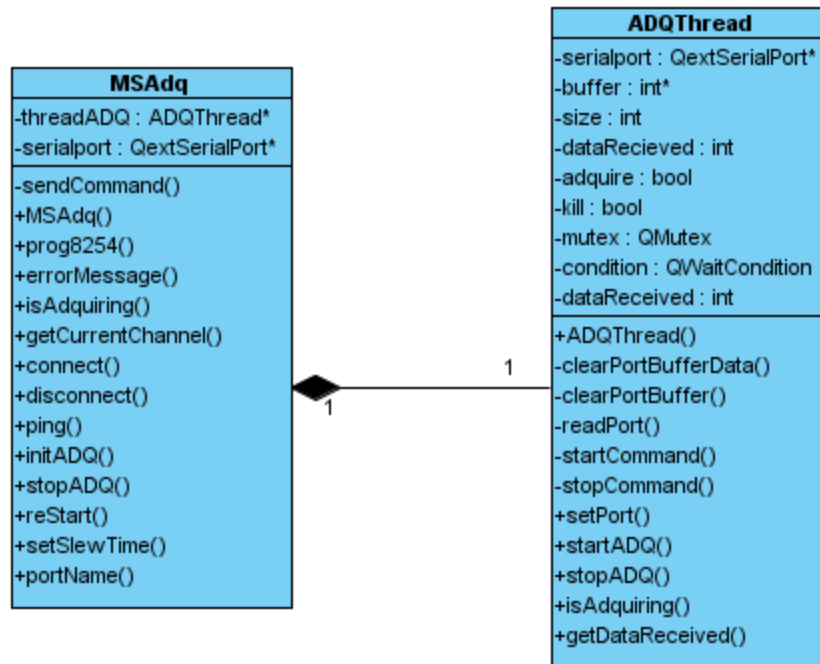


Figura 24: Diagrama de clases del diseño subsistema Driver.



### 4.3.3.3 Diagrama de clases subsistema mcSystem.

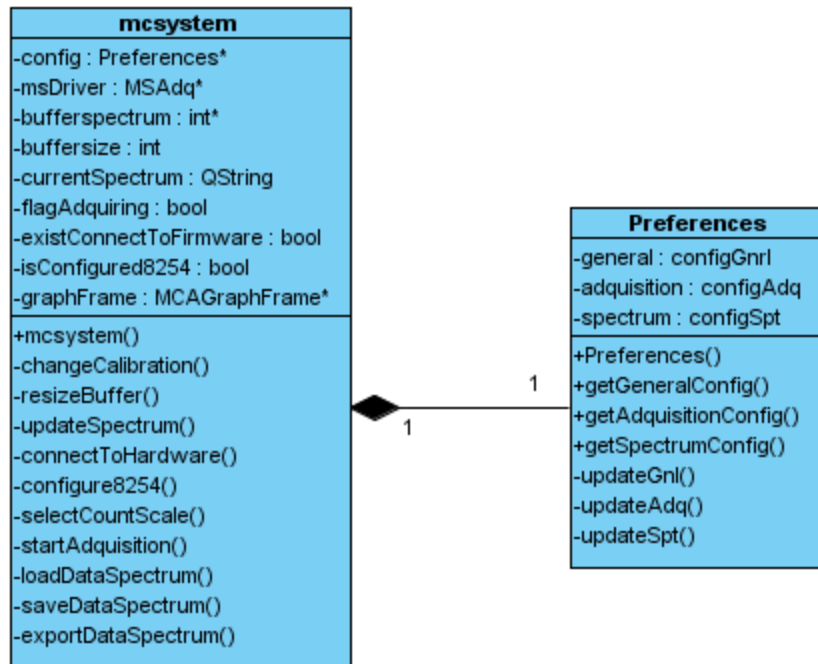


Figura 25: Diagrama de clases del diseño subsistema mcSystem.

### 4.3.4 Descripción de las clases del diseño.

A continuación se muestra una descripción de las principales clases modeladas en el diseño.

**Tabla 8:** Descripción de la clase *MCAGraph*.

<b>Nombre:</b>	MCAGraph
<b>Tipo de Clase:</b>	Control
<b>Atributos:</b>	<b>Tipo de datos:</b>
chAxil	ChannelAxil*
cursorChannel	int
<b>Responsabilidad</b>	paintEvent(event : QPaintEvent *):void
<b>Descripción:</b>	Visualiza todos los componentes que conforman la gráfica.
<b>Responsabilidad</b>	cursorPosition():int
<b>Descripción:</b>	Devuelve la posición del cursor.
<b>Responsabilidad</b>	backChannel():void
<b>Descripción:</b>	Mueve el cursor un canal hacia atrás.
<b>Responsabilidad</b>	nextChannel():void
<b>Descripción:</b>	Mueve el cursor un canal hacia delante.
<b>Responsabilidad</b>	zoomInChannelAxil():void
<b>Descripción:</b>	Disminuye el intervalo de la escala de los canales.
<b>Responsabilidad</b>	zoomOutChannelAxil():void
<b>Descripción:</b>	Aumenta el intervalo de la escala de los canales.
<b>Responsabilidad</b>	zoomInCountAxil():void
<b>Descripción:</b>	Disminuye el intervalo de la escala de los conteos.

<b>Responsabilidad</b>	zoomOutCounAxil():void
<b>Descripción:</b>	Aumenta el intervalo de la escala de los conteos.
<b>Responsabilidad</b>	scaleTopChannelAxilToBin(isbin : bool, repaint : bool) :void
<b>Descripción:</b>	Cambia el tipo de escala de lineal a binaria.
<b>Responsabilidad</b>	showTopAxil(show : bool):void
<b>Descripción:</b>	Muestra y oculta el eje superior.

*Tabla 9: Descripción de la clase MCAGraphBase.*

<b>Nombre:</b>	MCAGraphBase
<b>Tipo de Clase:</b>	Control
<b>Atributos:</b>	<b>Tipo de datos:</b>
backcolor	QColor
axilcolor	QColor
plotcolor	QColor
gridcolor	QColor
cursorcolor	QColor
body	MC Area*
CtAxil	CountAxil*
countScaleStyle	scaleStyle
<b>Responsabilidad:</b>	setData(data: int *, size : int)
<b>Descripción:</b>	Cambia los datos que son visualizados.

<b>Responsabilidad:</b>	clearData():void
<b>Descripción:</b>	Borra los datos visualizados.
<b>Responsabilidad:</b>	setBackColor(color : QColor) :void
<b>Descripción:</b>	Cambia el color del fondo.
<b>Responsabilidad:</b>	setAxilColor(color: QColor) :void
<b>Descripción:</b>	Cambia el color de los ejes.
<b>Responsabilidad:</b>	setPlotColor(color : QColor) :void
<b>Descripción:</b>	Cambia el color del plot.
<b>Responsabilidad:</b>	setGridColor(color : QColor) :void
<b>Descripción:</b>	Cambia el color del grid.
<b>Responsabilidad:</b>	setCursorColor(color: QColor) :void
<b>Descripción:</b>	Cambia el color del cursor.
<b>Responsabilidad:</b>	getBackColor():QColor
<b>Descripción:</b>	Devuelve el color del fondo.
<b>Responsabilidad:</b>	getAxilColor():QColor
<b>Descripción:</b>	Devuelve el color de los ejes.
<b>Responsabilidad:</b>	getPlotColor():QColor
<b>Descripción:</b>	Devuelve el color del plot.
<b>Responsabilidad:</b>	getGridColor():QColor
<b>Descripción:</b>	Devuelve el color del grid.

<b>Responsabilidad:</b>	getCursorColor():QColor
<b>Descripción:</b>	Devuelve el color del cursor.
<b>Responsabilidad:</b>	getCountScaleStyle():QColor
<b>Descripción:</b>	Devuelve el estilo del eje de los conteos.

*Tabla 10: Descripción de la clase MSAdq.*

<b>Nombre:</b>	MSAdq
<b>Tipo de Clase:</b>	Entidad
<b>Atributos:</b>	<b>Tipo de datos:</b>
threadADQ	ADQThread*
serialport	QextSerialPort*
<b>Responsabilidad</b>	sendCommand(MSCOMM command):void
<b>Descripción:</b>	Envía un comando al dispositivo.
<b>Responsabilidad:</b>	Prog8254(char lsb0, char msb0, char lsb1, char msb1, char lsb2, char msb2):bool
<b>Descripción:</b>	Permite enviar los valores de los contadores del temporizador al dispositivo.
<b>Responsabilidad</b>	errorMessage(MSANSW):void
<b>Descripción:</b>	Devuelve el mensaje de error correspondiente al comando de error pasado por parámetro.
<b>Responsabilidad</b>	isAcquiring():bool
<b>Descripción:</b>	Devuelve verdadero si se están adquiriendo datos del

	dispositivo.
--	--------------

**Tabla 11:** Descripción de la clase ADQThread.

<b>Nombre:</b>	ADQThread
<b>Tipo de Clase:</b>	Entidad
<b>Atributos:</b>	<b>Tipo de datos:</b>
serialport	QextSerialPort*
buffer	Int*
size	int
dataReceived	int
adquiere	bool
Kill	bool
mutex	QMutex
condition	QWaitCondition
<b>Responsabilidad:</b>	clearPortBuffer():void
<b>Descripción:</b>	Borra los datos excedentes del buffer del puerto serie después de detener la adquisición.
<b>Responsabilidad:</b>	clearPortBufferData():void
<b>Descripción:</b>	Borrar todos los datos del buffer del puerto serie
<b>Responsabilidad:</b>	readPort():void
<b>Descripción:</b>	Lee tres bytes de datos del puerto serie.

<b>Responsabilidad:</b>	startCommand():MSANSW
<b>Descripción:</b>	Envía comando para iniciar la adquisición.
<b>Responsabilidad:</b>	stopCommand():MSANSW
<b>Descripción:</b>	Envía comando para detener la adquisición.
<b>Responsabilidad:</b>	startADQ(buffer : int *, size : int) :void
<b>Descripción:</b>	Inicializa las variables donde serán guardados todos los datos adquiridos por el puerto serie.
<b>Responsabilidad:</b>	getDataReceived():int
<b>Descripción:</b>	Devuelve la cantidad de datos recibidos.

*Tabla 12: Descripción de la clase mcSystem.*

<b>Nombre:</b>	mcSystem
<b>Tipo de Clase:</b>	Control
<b>Atributos:</b>	<b>Tipo de datos:</b>
config	Preferences*
msDriver	MSAdq*
graphFrame	MCAGraphFrame*
bufferSpectrum	int*
bufferSize	int
flagAcquiring	bool
existConnectToFirmware	bool

isConfigured8254	bool
<b>Responsabilidad</b>	changeCalibration(time:int, baseTemp: BaseTime):void
<b>Descripción:</b>	Cambia la calibración de la gráfica.
<b>Responsabilidad</b>	updateSpectrum ():void
<b>Descripción:</b>	Actualiza los datos que se visualizan en la gráfica.
<b>Responsabilidad</b>	resizeBuffer(size: int):void
<b>Descripción:</b>	Cambia el tamaño del buffer utilizado para la adquisición y visualización de los datos.
<b>Responsabilidad</b>	connectToHardware(Port : QString):bool
<b>Descripción:</b>	Establece la conexión con el dispositivo MC-mcSystem por el puerto especificado.
<b>Responsabilidad</b>	configure8254(time : int, baseTemp : BaseTime):bool
<b>Descripción:</b>	Configura el tiempo de residencia durante el cual se van a contar los pulsos.
<b>Responsabilidad</b>	selectCountScale(scaleStyle : scale):void
<b>Descripción:</b>	Cambia el tipo de escala del eje de los conteos de la gráfica.
<b>Responsabilidad</b>	startAdquisition():void
<b>Descripción:</b>	Inicia la adquisición de datos del dispositivo.
<b>Responsabilidad</b>	loadDataSpectrum(fileName : QString):void
<b>Descripción:</b>	Carga los datos de un archivo con extensión *.mcf
<b>Responsabilidad</b>	saveDataSpectrum(fileName : QString):bool



<b>Descripción:</b>	Guarda los datos adquiridos en un archivo con extensión *.mcf en la ruta especificada.
---------------------	--

**Tabla 13:** Descripción de la clase Preferences.

<b>Nombre:</b>	Preferences
<b>Tipo de Clase:</b>	Interfaz
<b>Atributos:</b>	<b>Tipo de datos:</b>
general	configGnral
adquisición	configAdq
spectrum	configSptr
<b>Responsabilidad:</b>	getGeneralConfig():configGnral
<b>Descripción:</b>	Devuelve el objeto que contiene los valores de configuración de la comunicación con el dispositivo.
<b>Responsabilidad:</b>	getAdquisitionConfig():configAdq
<b>Descripción:</b>	Devuelve el objeto que contiene los valores de configuración de la adquisición del dispositivo.
<b>Responsabilidad:</b>	getSpectrumConfig():configSptr
<b>Descripción:</b>	Devuelve el objeto que contiene los valores de configuración de de los colores de la gráfica.
<b>Responsabilidad:</b>	updateGnl():void
<b>Descripción:</b>	Actualiza los valores del objeto que guarda la configuración de la comunicación.

<b>Responsabilidad:</b>	updateAdq():void
<b>Descripción:</b>	Actualiza los valores del objeto que guarda la configuración de la adquisición.
<b>Responsabilidad:</b>	updateSpt():void
<b>Descripción:</b>	Actualiza los valores del objeto que guarda la configuración de la gráfica.

## 4.4 Implementación.

### 4.4.1 Diagrama de despliegue.

A continuación se muestra como está desplegado físicamente el software y su interacción con otros dispositivos.

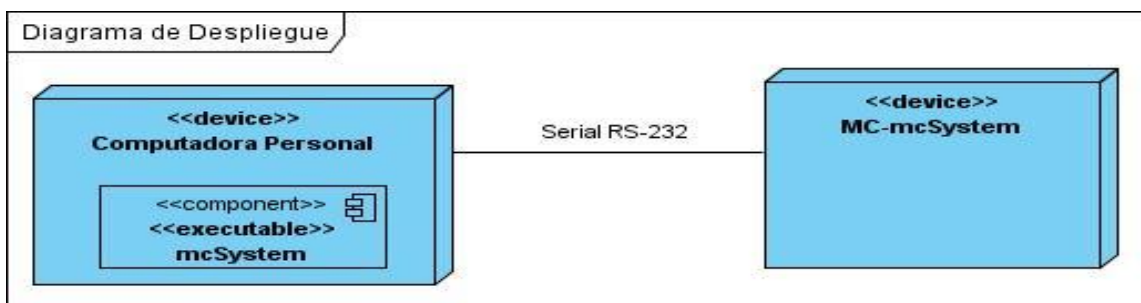


Figura 26: Diagrama de despliegue del software.

### 4.4.2 Diagrama de componente.

En el diagrama de la figura 27, se muestra la interacción entre el software y algunas bibliotecas de QT que son necesarias para el correcto funcionamiento del mismo.

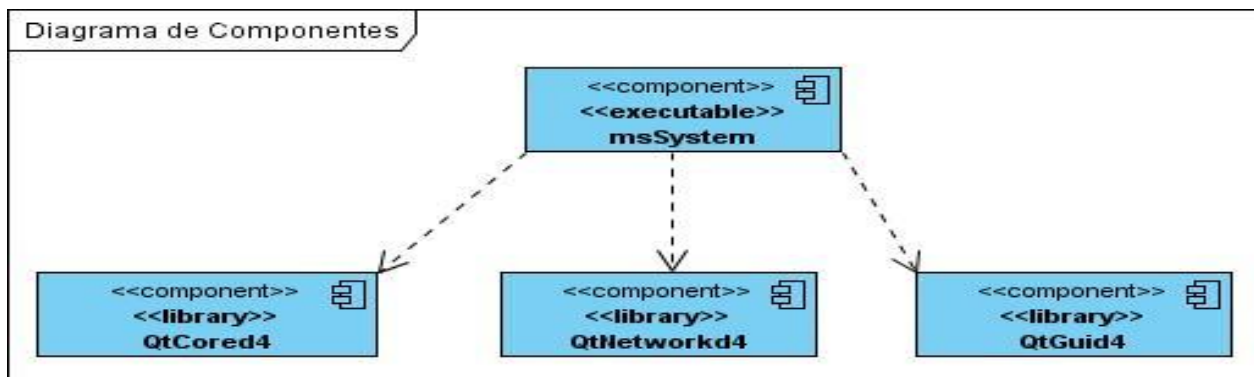


Figura 27: Diagrama de componentes del sistema.

#### **4.5 Conclusiones del capítulo.**

En este capítulo se abordó lo referente al análisis, diseño e implementación del software para la visualización de los datos adquiridos por la tarjeta de adquisición MC-mcSystem.

## CAPÍTULO 5 PRUEBAS

### 5.1 Introducción.

El presente capítulo aborda lo relacionado con las pruebas de aceptación realizadas tanto al software como al sistema en su totalidad.

### 5.3 Pruebas para el software.

Una prueba de software es un proceso o actividad a la cual un sistema es sometido bajo unas condiciones o requerimientos especificados. El objetivo de una prueba es identificar posibles fallos de implementación, usabilidad o calidad en un producto. Uno de los métodos de pruebas más utilizados es la conocida como prueba de la caja negra. La cual se refiere a las pruebas que se llevan a cabo sobre la interfaz del software y pretende demostrar que las funciones del software son operativas.

A continuación se muestran un conjunto de pruebas realizadas al sistema basadas en este método.

**Tabla 14:** Caso de prueba 1 Establecer Comunicación.

<b>Nombre de la prueba</b>	Establecer Comunicación			
<b>Caso de Uso de Prueba:</b>	Establecer comunicación con el dispositivo MC-mcSystem			
<b>Descripción de la prueba:</b>	Comprueba que el sistema establezca la comunicación correctamente con el dispositivo, habiendo llenado el usuario los datos correctamente.			
<b>Pre-condiciones</b>	El dispositivo MC-mcSystem debe estar conectado a la computadora			
<b>Post-condiciones</b>	El sistema establece la comunicación con el dispositivo mcSystem.			
<b>Notas:</b>				
<b>Resultado (Paso/ Falta/ Advertencia/ Incompleto)</b>				
	<b>Paso de la Prueba</b>	<b>Resultados esperados de la</b>	<b>P</b>	<b>F</b>

		Prueba		
1.	El usuario introduce en la sesión "General" de la ventana preferencia el nombre del puerto por donde se establecerá la comunicación con el dispositivo correctamente y pulsa el botón aplicar.	El sistema debe establecer la comunicación con el dispositivo.	X	

**Tabla 15:** Caso de prueba 2 Establecer Comunicación.

<b>Nombre de la prueba</b>	Establecer Comunicación 2			
<b>Caso de Uso de Prueba:</b>	Establecer comunicación con el dispositivo MC-mcSystem			
<b>Descripción de la prueba:</b>	Comprueba que el sistema muestre un mensaje de error al usuario indicándole que no se puede establecer la conexión cuando el dispositivo mcSystem cuando este último no este conectado al sistema.			
<b>Pre-condiciones</b>	El dispositivo MC-mcSystem no debe estar conectado a la computadora			
<b>Post-condiciones</b>	El sistema no establece la comunicación con el dispositivo mcSystem.			
<b>Notas:</b>				
<b>Resultado (Paso/ Falta/ Advertencia/ Incompleto)</b>				
	Paso de la Prueba	Resultados esperados de la Prueba	P	F
1.	El usuario introduce en la sesión "General" de la ventana preferencia el nombre del puerto por donde se establecerá la comunicación con el	El sistema debe mostrar un mensaje indicando que no se pudo establecer la comunicación	X	

dispositivo y pulsa el botón aplicar.	con el dispositivo.		
---------------------------------------	---------------------	--	--

**Tabla 16:** Caso de prueba 3 Establecer Comunicación.

<b>Nombre de la prueba</b>	Establecer Comunicación 3			
<b>Caso de Uso de Prueba:</b>	Establecer comunicación con el dispositivo MC-mcSystem			
<b>Descripción de la prueba:</b>	Comprueba que el sistema muestre un error indicándole que no se pudo establecer la conexión con el dispositivo cuando el mismo intenta establecer la conexión con el dispositivo después de haber llenado los datos para establecer la comunicación incorrectamente.			
<b>Pre-condiciones</b>	El dispositivo MC-mcSystem debe estar conectado a la computadora			
<b>Post-condiciones</b>	El sistema establece la comunicación con el dispositivo mcSystem.			
<b>Notas:</b>				
<b>Resultado (Paso/ Falta/ Advertencia/ Incompleto)</b>				
	<b>Paso de la Prueba</b>	<b>Resultados esperados de la Prueba</b>	<b>P</b>	<b>F</b>
1.	El usuario introduce en la sesión "General" de la ventana preferencia un nombre del puerto que no es el que el sistema operativo tiene definido para establecer la comunicación por el puerto serial y pulsa el botón aplicar para establecer la comunicación con el dispositivo.	El sistema debe mostrar un mensaje indicando que no se pudo establecer la comunicación con el dispositivo.	X	

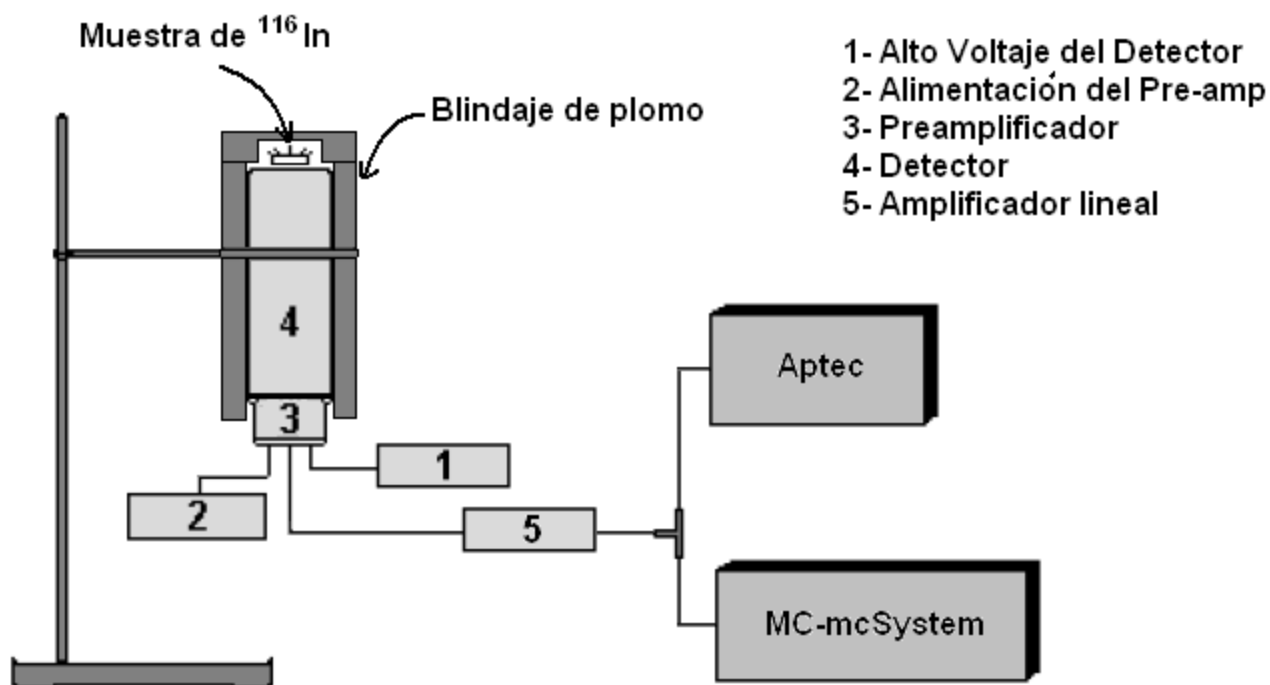
**Tabla 17:** Caso de prueba Adquirir datos.

<b>Nombre de la prueba</b>	Adquirir datos			
<b>Caso de Uso de Prueba:</b>	Adquirir datos de dispositivo MC-mcSystem			
<b>Descripción de la prueba:</b>	Comprueba que cuando el usuario intente iniciar la adquisición sin haber configurado la misma previamente el sistema muestre un mensaje indicando que no puede realizar la acción.			
<b>Pre-condiciones</b>	El dispositivo MC-mcSystem debe estar conectado a la computadora y haberse establecido la comunicación con el mismo.			
<b>Post-condiciones</b>	El sistema no debe iniciar la adquisición de datos del dispositivo MC –mcSystem			
<b>Notas:</b>				
<b>Resultado (Paso/ Falta/ Advertencia/ Incompleto)</b>				
	<b>Paso de la Prueba</b>	<b>Resultados esperados de la Prueba</b>	<b>P</b>	<b>F</b>
1.	El usuario pulsa en el botón iniciar adquisición del menú “adquisición” sin haber configurado la adquisición.	El sistema debe mostrar un mensaje indicando que no se ha configurado la adquisición.	<b>X</b>	



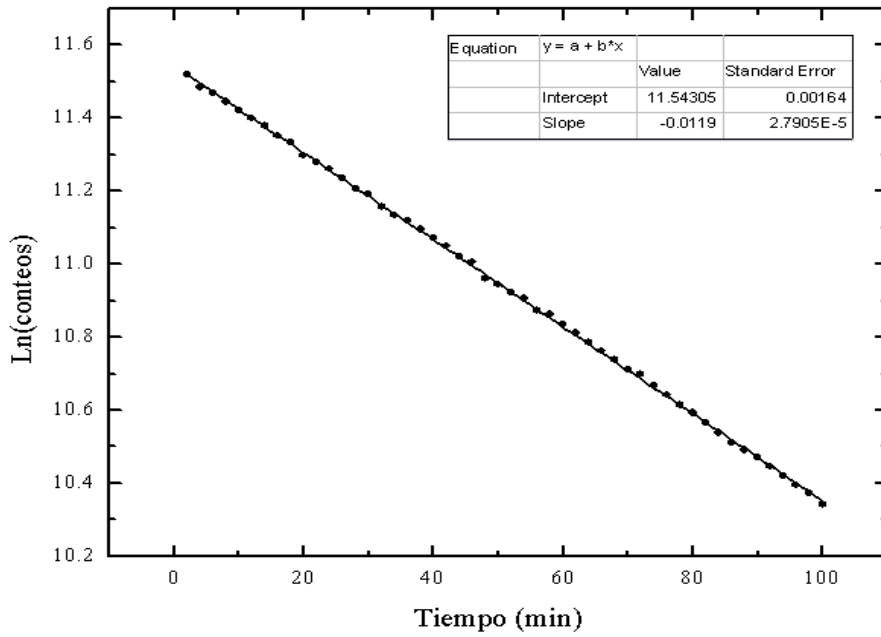
#### 5.4 Pruebas generales realizadas al sistema.

Como prueba final se decidió poner el sistema bajo las condiciones reales donde será usado y compararlo a su vez con un sistema profesional existente. Para ello se preparó el siguiente montaje experimental para la realización de la práctica de medición del tiempo de decaimiento del Indio ( $^{116}\text{In}$ ), con la particularidad que los pulsos provenientes del detector son suministrados simultáneamente al sistema MC-mcSystem y al Aptec.

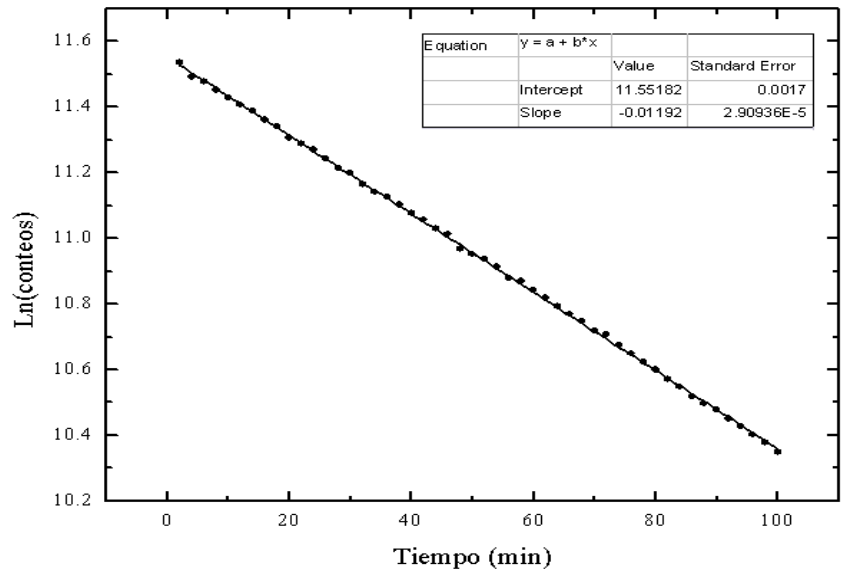


**Figura 28:** Montaje experimental para la medición del tiempo de decaimiento del In.

Los resultados para las curvas de decaimiento en ambos sistemas, utilizando un tiempo de residencia por canal de dos minutos, están representados en las siguientes gráficas.



**Figura 29:** Curva de decaimiento para el sistema Aptec.



**Figura 30:** Curva de decaimiento para el sistema MC-mcSystem.

El valor real para el período de semidesintegración del  $^{116}\text{In}$  es de 54 minutos, los valores obtenidos en el experimento se muestran en la tabla:

**Tabla 18:** Resultados experimentales del tiempo de semidesintegración.

	$T_{1/2}$	Error %
<b>Aptec</b>	58.15	7.7
<b>MC-mcSystem</b>	58.24	7.9

Aunque existe un error aproximado del 8% respecto al valor real, debido a diversos factores experimentales independientes a ambos sistemas, el error relativo entre ambos es menor del 0.2%.

### 5.5 Conclusiones del capítulo.

Con el desarrollo de este capítulo se expuso el conjunto de pruebas de caja negra realizadas al software. Se realizó una comparación del sistema desarrollado con un sistema profesional bajo las mismas condiciones experimentales y los resultados obtenidos muestran un correcto funcionamiento del sistema realizado.

## CONCLUSIONES

A través de la realización de este trabajo investigativo se arribaron a las siguientes conclusiones:

- Se desarrolló el firmware para la adquisición de los pulsos provenientes de un detector de radiaciones y el software para el control de la adquisición y visualización de los datos adquiridos.
- Se obtuvo como resultado un sistema sencillo que permite realizar algunas de las prácticas de laboratorio referentes al estudio de las propiedades nucleares realizadas por los estudiantes de física nuclear del InSTEC.

## RECOMENDACIONES

- Realizar un diseño del hardware con un PIC más moderno que permita la comunicación con la computadora a través del puerto USB.
- Incluir en el software nuevas funcionalidades que permitan el análisis de los datos obtenidos.

## REFERENCIAS BIBLIOGRÁFICAS

1. La Infraestructura Productiva. [En línea] [Citado el: octubre 20, 2010.] <http://www.uci.cu/?q=node/72>.
2. The Multichannel Analyzer. [En línea] [Citado el: diciembre 16, 2010.] <http://www.ortec-online.com/electronics/mca/mca-datasheets.htm> .
3. Multichannel Analyzer (MCA). [En línea] [Citado el: febrero 12, 2010.] <http://www.amptek.com/mca8000a.html> .
4. CMCA-550 Data Acquisition Module. [En línea] [Citado el: febrero 12, 2010.] [http://www.wissel-gmbh.de/index.php?option=com\\_content&task=view&id=45&Itemid=70](http://www.wissel-gmbh.de/index.php?option=com_content&task=view&id=45&Itemid=70) .
5. Desarrollo de un analizador multicanal. [En línea] julio 30, 2003. [Citado el: diciembre 17, 2009][http://www.cubaenergia.cu/index.php?option=com\\_docman&task=doc\\_download&gid=91&Itemid=1](http://www.cubaenergia.cu/index.php?option=com_docman&task=doc_download&gid=91&Itemid=1) .
6. Sistema de adquisición de datos para aplicaciones donde sea necesario el multiconteo de eventos [En línea] [Citado el: diciembre 18, 2009.] <http://www.redeweb.com/txt/articulos/56580121.pdf> .
7. Introducción a la Ingeniería de Software. [En línea] septiembre 28, 2009. [Citado el: diciembre 18, 2010.][http://eva.uci.cu/file.php/102/Curso\\_2009-2010/Semana\\_1/Conferencia\\_1/Materiales\\_Basicos/C1\\_guia\\_estudiante.doc](http://eva.uci.cu/file.php/102/Curso_2009-2010/Semana_1/Conferencia_1/Materiales_Basicos/C1_guia_estudiante.doc) .
8. Introducción a la Ingeniería de Software. [En línea] mayo 4, 2009. [Citado el: diciembre 18, 2009.] [http://eva.uci.cu/file.php/58/Bibliografia/Ingenieria\\_de\\_SW/Conferencias/Conferencias\\_IS1\\_2007-2008/Conferencias.rar](http://eva.uci.cu/file.php/58/Bibliografia/Ingenieria_de_SW/Conferencias/Conferencias_IS1_2007-2008/Conferencias.rar) .
9. **Sánchez, Maria Mendosa.** Metodologías de desarrollo de software. [en Línea] julio 7, 2004.[Citado el: diciembre 18, 2009.] [http://www.informatizate.net/articulos/metodologias\\_de\\_desarrollo\\_de\\_software\\_07062004.html](http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html) .
10. Proceso de Desarrollo OpenUP. [En línea] septiembre 2, 2008. [Citado el: diciembre 17, 2009.] <http://cbasqa.wordpress.com/2008/09/02/proceso-de-desarrollo-openup/> .

11. 2da. s.l.: Microsoft Corporation, 2009. 2. **Microsoft Corporation.** Microsoft Application Architecture Guide.

## **BIBLIOGRAFÍA**

1. CMCA-550 Data Acquisition Module. [En línea] [Citado el: febrero 12, 2010.] [http://www.wissel-gmbh.de/index.php?option=com\\_content&task=view&id=45&Itemid=70](http://www.wissel-gmbh.de/index.php?option=com_content&task=view&id=45&Itemid=70).
2. Multichannel Analyzer (MCA). [En línea] febrero 12, 2010. <http://www.amptek.com/mca8000a.html>.
3. Microcontroladores PIC. [En línea] [Citado el: febrero 6, 2010.] [http://www.unicrom.com/Tut\\_PICs1.asp](http://www.unicrom.com/Tut_PICs1.asp).
4. Microcontroladores PIC16CXX/FXX de Microchip. [En línea] [Citado el: febrero 7, 2010.] <http://r-luis.xbot.es/pic1/pic01.html>.
5. Principales herramientas CASE del mercado y su uso. [En línea] [Citado el: diciembre 14, 2009.] [http://www.um.es/docencia/barzana/IAGP/Enlaces/CASE\\_principales.html](http://www.um.es/docencia/barzana/IAGP/Enlaces/CASE_principales.html).
6. El lenguaje C++. [En línea] [Citado el: diciembre 15, 2009.] [http://www.zator.com/Cpp/E1\\_2.htm](http://www.zator.com/Cpp/E1_2.htm).
7. Introducción al lenguaje C. [En línea] [Citado el: diciembre 15, 2009.] <http://usuarios.multimania.es/cerebrum/cursoc/1-intro.html>.
8. **Buñuel, José Luis Asín.** Lenguaje C. [En línea] 2005. [Citado el: 16 de diciembre de 2009.] <http://jlab.tripod.com/LenguajeC.pdf>.
9. Introducción al lenguaje ensamblador. [En línea] [Citado el: diciembre 16, 209.] <http://eva.uci.cu/mod/resource/view.php?inpopup=true&id=18929>.
10. Lenguaje Ensamblador. [En línea] [Citado el: diciembre 17, 2009.] <http://homepage.mac.com/eravila/asmix861.html#l>.
11. Tutorial de lenguaje ensamblador. [En línea] [Citado el: diciembre 16, 2009.] <http://sistemas.itlp.edu.mx/tutoriales/ensamblador/index.htm>.

12. El leguaje ensamblador. [En línea] [Citado el: diciembre 22, 2009.] <http://www.ufps.edu.co/materias/ucontrol/htdocs/conte/usuarios.lycos.es/sfriswolker/pic/seis.htm>.
13. **Menéndez-Barzanallana, Rafael Asensio**. Principales herramientas CASE del mercado y su uso. [En línea] [Citado el: diciembre 17, 2009.] [http://www.um.es/docencia/barzana/IAGP/Enlaces/CASE\\_principales.html](http://www.um.es/docencia/barzana/IAGP/Enlaces/CASE_principales.html).
14. Introducción a la Ingeniería de Software. [En línea] septiembre 28, 2009. [Citado el: diciembre 18, 2009.] [http://eva.uci.cu/file.php/102/Curso\\_2009-2010/Semana\\_1/Conferencia\\_1/Materiales\\_Basicos/C1\\_guia\\_estudiante.doc](http://eva.uci.cu/file.php/102/Curso_2009-2010/Semana_1/Conferencia_1/Materiales_Basicos/C1_guia_estudiante.doc).
15. La interfaz RS-232. [En línea] [Citado el: enero 12, 2010.] <http://neo.lcc.uma.es/evirtual/cdd/tutorial/fisico/inter232.html>.
16. Nuevo compilador de C para PIC® - mikroC PRO para PIC® 2009. [En línea] 2009. [Citado el: diciembre 17, 2009.] <http://www.elektor.es/noticias/nuevo-compilador-de-c-para-pic-mikroc-pro-para.924745.lynx>.
17. MicroC PRO for PIC. [En línea] abril 2009. [Citado el: diciembre 16, 2009.] [http://www.mikroe.com/pdf/mikroc\\_pic\\_pro/mikroc\\_pic\\_pro\\_manual\\_v100.pdf](http://www.mikroe.com/pdf/mikroc_pic_pro/mikroc_pic_pro_manual_v100.pdf).
18. Qué es Eclipse. [En línea] [Citado el: diciembre 18, 2009.] <http://plataformaclipse.com/faq>.
19. QT. [En línea] [Citado el: diciembre 16, 2009.] <http://qt.nokia.com/products>.
20. **Salcedo, José Luís Villarroel**. Sistema Empotrado. [En línea] [Citado el: febrero 18, 2010.] <http://webdiis.unizar.es/~joseluis/SE.pdf>.
21. **Balcázar, Gisela Galicia**. Curso básico de microcontroladores PIC. [En línea] [Citado el: febrero 18, 2010.] <http://betosoria.googlepages.com/pics.pdf>.
22. **Figuroa, Pablo**. Introducción a Patrones. [En línea] 1997. [http://agamenon.uniandes.edu.co/~pfiguroa/soo/Magister\\_Patrones/intropatrones.html](http://agamenon.uniandes.edu.co/~pfiguroa/soo/Magister_Patrones/intropatrones.html).
23. **Carletti, Eduardo J**. Comunicación - MAX232 - Conversor TTL-RS232. [En línea] [Citado el: febrero 18, 2010.] [http://axxon.com.ar/rob/Comunicacion\\_max232.htm](http://axxon.com.ar/rob/Comunicacion_max232.htm).



24. Arquitectura de software. [En línea] [Citado el: febrero 16, 2010.]  
<http://www.sei.cmu.edu/architecture/definitions.html>.

## GLOSARIO DE TÉRMINO

**CPU:** Unidad central de procesamiento.

**CU:** Caso de Uso.

**Desintegración Nuclear:** Transformación producida en un núcleo atómico por pérdida de alguna partícula con absorción o pérdida de energía.

**Espectroscopia Mössbauer:** Es una técnica espectroscópica basada en el efecto Mössbauer que no es más que la absorción nuclear resonante de la radiación emitida por un núcleo.

**IBM:** *International Business Machines*. Empresa multinacional que comercializa programas, servicios y herramientas relacionadas con la informática.

**IDE:** Entorno de desarrollo integrado (normalmente conocida por su acrónimo, *integrated development environment*), es un programa informático compuesto por un conjunto de herramientas de programación.

**ISA:** Standard de Arquitectura Industrial (normalmente conocida por su acrónimo, *Industry Standard Architecture*) es una arquitectura de bus diseñada para conectar tarjetas de ampliación a la tarjeta madre de la computadora.

**Isótopo:** Cuerpo que, en el sistema periódico, tiene el mismo número atómico que otro y por tanto pertenece al mismo elemento químico. Isótopo radiactivo es aquel cuyos núcleos sufren transformaciones emitiendo rayos tipo alfa, beta o gamma.

**Microcontrolador:** Es un circuito integrado o chip programable, capaz de ejecutar las órdenes grabadas en su memoria.

**Núcleo Atómico:** Parte central del átomo.

**PIC:** Controlador de interfaz periférico.

**PIT:** Temporizador de intervalos programable.

**RAM:** Memoria de acceso aleatorio (normalmente conocida por su acrónimo, *Random-Access Memory*) es la memoria desde donde el procesador recibe las instrucciones y guarda los resultados.

**Radiación:** Energía ondulatoria o partículas materiales que se propagan a través del espacio.

**RISC:** Computadora con Conjunto de Instrucciones Reducidas, es un modelo de arquitectura para microprocesadores.

**ROM:** Memoria de sólo lectura (normalmente conocida por su acrónimo, *Read Only Memory*) es una clase de medio de almacenamiento utilizado en los ordenadores y otros dispositivos electrónicos.

**RS-232:** Estándar recomendado 232 es una interfaz que designa una norma para el intercambio serie de datos binarios entre un DTE (Equipo terminal de datos) y un DCE (Equipo de Comunicación de datos), aunque existen otras en las que también se utiliza la interfaz RS-232.

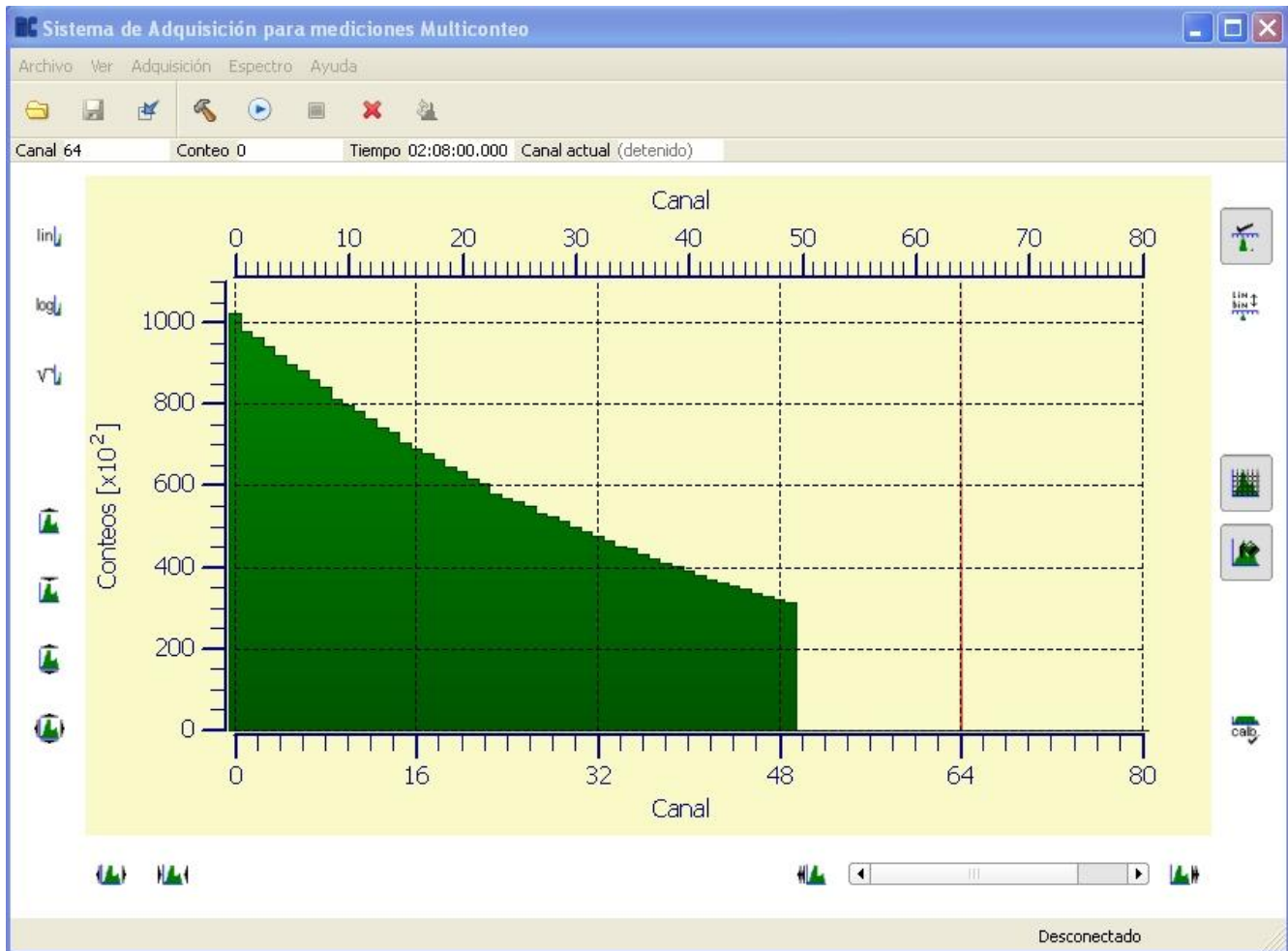
## ANEXOS



**Figura 31:** Interfaz del software para establecer la comunicación con el dispositivo MC-mcSystem.



**Figura 32:** Interfaz del software para la configuración de la adquisición desde el dispositivo MC-mcSystem.



**Figura 33:** Interfaz del software para la visualización de los datos adquiridos del dispositivo MC-mcSystem.