UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 5

Simulador de Variables Analógicas y Digitales para la Interfaz Hombre Máquina (HMI) del SCADA “Guardián del Alba”.

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

Autor: Darlin Mercedes Blanco García

Tutores: Ing. Raudi Agdel Bacallao Sánchez

Ing. Adriel Linares Ramos

Co-tutor: Msc Enrique Carreras Paz

La Habana, Mayo 2013

"El hombre debe transformarse al mismo tiempo que la producción progresa; no realizaríamos una tarea adecuada si fuéramos tan sólo productores de artículos, de materias primas y no fuéramos al mismo tiempo productores de hombres…"

Ché Guevara.

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco 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 _____.

_____

Darlin Mercedes Blanco García

Autor.

_____ _____

Ing. Raudi Agdel Bacallo Sánchez

Tutor.

Ing. Adriel Linares Ramos

Tutor.

DATOS DE CONTACTO

Tutor: Ing. Raudi Agdel Bacallao Sánchez
Edad: 28 años.
Institución: Universidad de las Ciencias Informáticas (UCI).
Cargo: Especialista
Título: Ingeniería en Ciencias Informáticas.
Categoría Docente:
E-mail: rbacallao@uci.cu

Tutor: Ing. Adriel Linares Ramos.
Edad: 25 años.
Institución: Universidad de las Ciencias Informáticas (UCI).
Cargo: Especialista
Título: Ingeniería en Ciencias Informáticas
Categoría Docente:
E-mail: alramos@uci.cu

DEDICATORIA

Dedico este trabajo de diploma a las personas que más amo en esta vida, a las que me dieron todo su amor y cariño durante esta etapa tan dura y difícil que fue mi carrera y que siempre me dieron todo ese apoyo que siempre se necesita: a mis padres Nelly y Juanito, a mi tatica Betty, a mis abuelos Juan ,Edelmira, y Jesús Orlando, a mi novio Reynier, a mi cuñado Ariel ,a todos mis amigos y compañeros de estudio y a una cosita que adoro con todo mi corazón: mi perrita Cinty Maria…

AGRADECIMIENTOS

Quisiera agradecer a todas las personas que de una forma u otra contribuyeron para que pudiera realizar mi sueño de hacerme ingeniera, a todas las que con paciencia y tolerancia me ayudaron en momentos tensos y decisivos en mi carrera, a todos llegue mi más sincero agradecimiento.

A mis padres por haber confiado en mi y haberme dado el todo el apoyo, tanto en malos como en buenos momentos, por haberse sacrificado tanto a la par mia, pues esta carrera en parte también fue de ellos, no tengo como agradecer todo lo que hacen por mi desde que llegué a este mundo, gracias por estar a mi lado en cada momento de mi vida, gracias por existir.

A mi hermana Betty que para mí es más que una hermana, es mi otra madre, es mi ejemplo, mi guía, mi luz que ilumina cada paso que doy en la vida, gracias por acompañarme siempre en tus pensamientos y en tu corazón, sin ti no hubiese podido lograr muchas cosas ni hubiese tenido la fuerza para seguir adelante y lograr esta meta.

A mis amigo Adrian quien jugó un papel muy importante en mi desempeño profesional y siempre me brindó su mano amiga cuando lo necesité, transmitiéndome siempre mucha seguridad y optimismo, gracias por todo ese apoyo incondicional. A mi amiga Gisela quien conocí el primer día que llegué a esta universidad le agradezco todos sus consejos que siempre me ayudaron a tomar decisiones importantes, gracias por tu amistad. A Liane por ser mi amiga, mi confidente, mi consuelo en momentos muy duros, gracias por entenderme y apoyarme siempre que lo necesito, A Robe por haberme acogido como su ahijada y haberme brindado todos sus conocimientos y experiencia, a mi amiga Betty que siempre supo guiarme , gracias por esas palabras tan sinceras y llenas de razón.

A mis compañeros de estudio que han estado conmigo durante estos cinco años y que hemos compartido momentos de felicidad y tristeza siempre juntos. A Isa por ser tan optimista y transmitirme mucha alegría . A Yadis por ser tan simpática y alegre y en ella poder encontrar una buena amistad. A Ari quien hizo crecer en mí una gran estima por ella, gracias por considerarme tu amiga. A Yalily y Edel que siempre me ayudaron en todo

momento y se convirtieron en mis buenos amigos, gracias por todo. A mis amigas del alma Amanda, Yenis, Yohanne, Leydis, Greter, Radel , Sandy, Wilder, Andiel, Geyrelis, Richar, Mursuli a todos mis amigos incluyendo a Mayra e ILeana mis amigas de la infancia, y compañeros que siempre han estado a mi lado, gracias por ser parte de mi vida.

A mi novio Reinier por su dedicación, paciencia y cariño incondicional, por darme todo su apoyo y estar conmigo en todo momento, haciendo suyos mis malos y buenos momentos, por considerarme una de las personas más importantes en su vida.

A mi profesores Alejandro, Yaself; Gelson, Luis Eduardo, Saylin, Yorji, Jessie, Coca a todos los que me brindaron sabias enseñanzas que nunca olvidaré y guiaron mi trayectoria por esta universidad que hoy me permite graduarme como una gran profesional.

A mis tutores Raudi y Adriel por su gran ayuda durante el desarrollo de este trabajo y por la confianza depositada en mí, a ellos mi más sincero agradecimiento pues no hubiese podido llegar hasta aquí sin su colaboración, gracias por sus consejos, por sus regaños, por sus sonrisas. Raudi aunque la distancia estuvo presente nunca me sentí sola pues contaba con tu apoyo incondicional, siempre estuviste presente en todo momento, Adriel fuiste mi gran motor impulsor y tu ayuda fue determinante en la última etapa de mi carrera, gracias por tu comprensión y tolerancia.

RESUMEN

Los sistemas de supervisión, control y adquisición de datos (*SCADA*) están basados en computadores que permiten supervisar y controlar una instalación, proceso o sistema de características variadas. Los datos supervisados se obtienen a partir de dispositivos que son instalados en las industrias los cuales poseen la capacidad de supervisar y/o controlar los diferentes estados en los que se encuentra un proceso determinado.

En el Centro de Informática Industrial de la Universidad de las Ciencias Informáticas se desarrolla en convenio con la Empresa de Petróleos de Venezuela S.A. (PDVSA), el SCADA Guardián del Alba (GALBA). A este proyecto lo componen varios módulos tales como: Comunicación, Procesamiento, Interfaz Hombre-Máquina (HMI) y Aplicaciones. El módulo HMI es quien permite el contacto directo con el sistema a través de sinópticos gráficos almacenados en el ordenador de proceso y generados desde el editor incorporado en el GALBA, integrando los datos generados por los demás componentes. De esta manera el HMI posibilita un mayor acercamiento al proceso supervisado, pero en ocasiones en dicho proceso, no se cuenta con los datos directos de los dispositivos de campos para comprobar si los requerimientos o funcionalidades fueron bien implementados.

En el ambiente de prueba del HMI se hace necesario el uso de valores reales o al menos lo más exactos posible. Para dar solución a dicho problema se realiza el siguiente trabajo que tiene como objetivo desarrollar una herramienta que provea a la Interfaz Hombre-Máquina del GALBA de un mecanismo que simule el comportamiento de variables analógicas y digitales permitiendo el análisis de los procesos de supervisión.

***Palabras clave**:* SCADA, módulos, variables, procesos, dispositivos.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLA

INTRODUCCIÓN

En Cuba se han desarrollado numerosas aplicaciones informáticas para contribuir a desarrollar la economía, así como la informatización de empresas tanto del país como del exterior. Una de las instituciones que ha dado grandes pasos en este sentido es la Universidad de las Ciencias Informáticas (UCI). En esta universidad se encuentra el Centro de Informática Industrial (CEDIN) que en convenio con la empresa Petróleos de Venezuela SA (PDVSA) desarrolla el proyecto productivo SCADA Guardián del ALBA (GALBA).

Los sistemas de supervisión, control y adquisición de datos, constituyen el núcleo de la mayoría de estos sistemas automatizados, por tal motivo se erigen como altamente estratégicos para cualquier industria. Un SCADA es una aplicación software especialmente diseñada para funcionar sobre ordenadores en el control de producción, proporcionando comunicación con los dispositivos de campo y controlando el proceso que se está realizando desde la pantalla del ordenador [1]. Además, provee a diversos usuarios información generada en el proceso productivo, control de calidad, supervisión y mantenimiento. En el proceso de desarrollo se cuenta con las siguientes limitantes:

1. En tiempo de desarrollo es difícil capturar los datos físicos de dispositivos puesto que las áreas de desarrollo y supervisión están distantes entre sí geográficamente, teniendo que utilizarse simuladores de datos.
2. Los simuladores que se utilizan no generan el conjunto de datos con las potencialidades que se necesitan como permisos de lectura, escritura, además de que se necesita una directa conexión con el módulo de HMI del SCADA GALBA, dificultando entonces el proceso de configuración de datos.
3. Resulta costoso y complejo generar una muestra de escenarios representativos que modele una relación completa de todos los estados posibles, por lo que no son suficientes los parámetros brindados por lo simuladores utilizados.
4. Los datos de simulación utilizados no son lo suficientemente flexibles como para procesar de forma automática los datos en estampas de tiempos, la accesibilidad, el rango numérico y calidad de los mismos, entre otros parámetros.

De todo lo expuesto anteriormente se presenta como **problema a resolver**:
¿Cómo garantizar la simulación numérica de valores en el proceso de supervisión y control en el SCADA a través de la Interfaz Hombre Máquina sin los dispositivos de campos?

A partir del problema de investigación planteado se define como **objeto de estudio** las herramientas de simulación numérica y como **campo de acción** las herramientas de simulación numérica en los HMI. El **objetivo general** consiste en: *Desarrollar una herramienta para simular los datos provenientes de los dispositivos de campo, haciendo énfasis en los permisos de lectura, escritura y la conexión directa con el módulo de HMI.*

Para dar cumplimiento al objetivo planteado, se elaboraron una serie de tareas investigativas que se muestran a continuación:

- Revisar bibliografía en lo referente al estado del arte del tema, así como las herramientas, estándares y métodos para implementar este tipo de aplicación para sistemas SCADA.

- Análisis de la estructura de los módulos de la Interfaz Hombre-Máquina (HMI) del SCADA GALBA para comprender su funcionamiento actual.

- Descripción del prototipo funcional de la herramienta de simulación para variable analógica y digital.

- Desarrollo de la herramienta de simulación de datos para el HMI del SCADA GALBA.

- Valoración de los resultados a partir de las pruebas realizadas.

Métodos Empíricos:

Consulta de fuentes de información: Este método se utilizó para la consulta de las fuentes bibliográficas durante la investigación, lo cual fue muy importante para la elaboración del marco teórico. A través del mismo se pudo abarcar y proveer gran parte de toda la información referente al tema de los simuladores y su utilización en los sistemas SCADA.

Observación: Este método se utilizó para apreciar el comportamiento de la herramienta elaborada de acuerdo a los algoritmos implementados en la misma, dando una mejor percepción de los resultados alcanzados y permitiendo de esta manera erradicar los errores cometidos durante la implementación de la aplicación.

Pruebas: Método utilizado para validar los resultados obtenidos con la solución propuesta a partir de la investigación realizada. A través del mismo se pudo obtener una mejor percepción de lo que se estaba elaborando, haciendo una comparación entre los resultados esperados y la respuesta del sistema al realizar las pruebas.

Métodos teóricos:

Analítico-Sintético: Este es un método teórico que se utilizó para extraer y analizar la información acerca de los sistemas de simulación, ya que se hacía necesaria una amplia investigación sobre dichos sistemas para lograr un mejor entendimiento de sus funcionalidades, un mayor acercamiento al estudio de estas herramientas y su gran utilidad para obtener resultados aproximados.

Para darle una mejor organización al trabajo, este se ha estructurado en tres capítulos. A continuación se muestra una breve síntesis de cada uno de ellos:

Capítulo 1. Fundamentación Teórica

Se reflejan las principales características y conceptos asociados a los simuladores. Se realiza una breve descripción de los sistemas SCADA, sus distintos módulos y ejemplos de simuladores utilizados por estos. Se hará énfasis en su utilidad en la esfera tecnológica y cómo constituye una herramienta importante para aproximar resultados. Se describen las principales tecnologías que se emplean en la construcción de software para la simulación.

Capítulo 2. Análisis y diseño de la herramienta

Se realiza un análisis detallado especificando los requisitos funcionales como historias de usuario (HU), diagramas, tarjetas CRC donde se describen las principales clases que componen la aplicación de acuerdo a la metodología XP que fue la definida para la realización

de este proyecto. De esta forma se detalla la información del análisis y diseño de la herramienta en cuestión.

Capítulo 3. Implementación y prueba de la herramienta

En este capítulo se detallan las tres iteraciones realizadas durante la etapa de implementación, definiendo las funcionalidades como tareas de ingeniería de acuerdo a lo planteado por la metodología escogida. Además se especifican los resultados obtenidos mediante las pruebas de aceptación realizadas para obtener la valoración de dichas pruebas de manera cualitativa.

CAPÍTULO 1. Fundamentación Teórica

1.1 Introducción

En el presente capítulo se introducen las principales características y conceptos asociados a los simuladores. Para ello se abordan definiciones, importancia, ventajas y desventajas de estos sistemas. Se realiza una breve descripción de los sistemas SCADA, de sus distintos módulos y ejemplos de simuladores utilizados por estos. Se describen las principales tecnologías que se emplean en la construcción de software para la simulación.

1.2 Sistemas SCADA

Los sistemas SCADA utilizan la computadora y tecnologías de comunicación para automatizar el monitoreo y control de procesos industriales. Estos sistemas son partes integrales de la mayoría de los ambientes industriales complejos o distantes geográficamente (ver Figura 1), ya que pueden recoger la información de una gran cantidad de fuentes rápidamente y la presentan a un operador de forma amigable. Los sistemas SCADA mejoran el proceso de monitoreo y control proporcionando la información oportuna para poder tomar decisiones operacionales apropiadas [1].

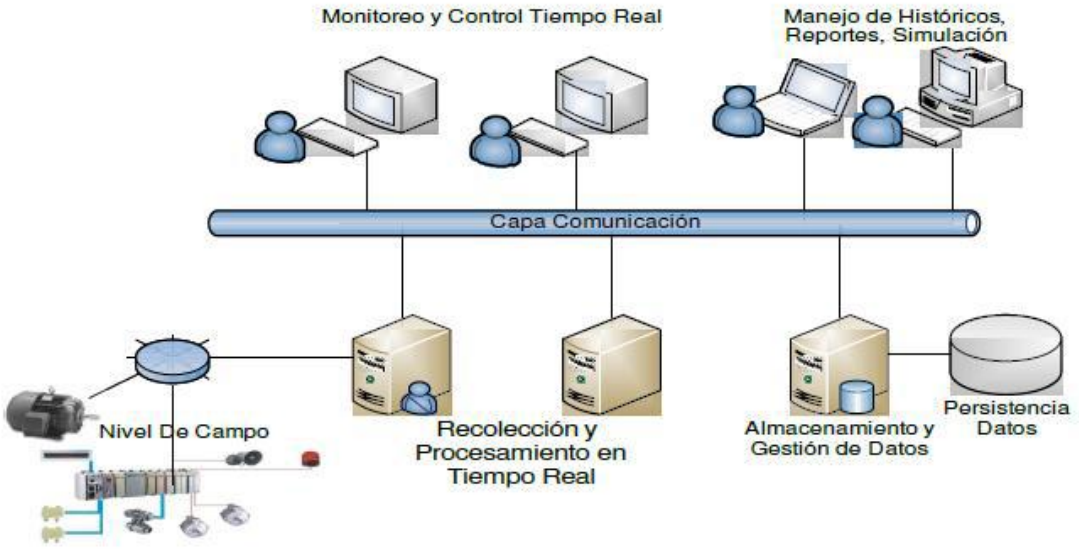

Figura 1. Sistema SCADA

Los primeros sistemas automatizados SCADA fueron altamente modificados con programas de aplicación específicos para atender a requisitos de algún proyecto particular. Varios ingenieros asistieron al diseño de estos sistemas y realizaron un análisis de las características de sus industrias para luego integrar estos sistemas al proceso productivo de acuerdo a los intereses de cada institución. Los proveedores de sistemas SCADA, deseando reutilizar su trabajo previo sobre nuevos proyectos, perpetuaron esta imagen de industria específica por su propia visión de los ambientes de control con los cuales tenían experiencia. Solamente cuando nuevos proyectos necesitaron funciones y aplicaciones adicionales, entonces los desarrolladores tuvieron la oportunidad de adquirir experiencias en otras industrias. [1]

En la actualidad, estos proveedores están diseñando sistemas que son pensados para resolver las necesidades de muchas industrias, con módulos de software industriales específicos disponibles para proporcionar las capacidades requeridas comúnmente. Usualmente se encuentran software de este tipo comercialmente disponible adaptado para procesamiento de papel y celulosa, industrias de aceite y gas, hidroeléctricas, gerencia y provisión de agua, así como el control de fluidos.

1.2.1 Componentes del sistema SCADA GALBA

Los módulos o bloques de software que permiten las actividades de adquisición, supervisión y control son los siguientes:

- **Configuración**: Este componente permite al operador configurar los procesos, definir y gestionar las variables, editar los controladores, los comandos, las alarmas y variadas opciones adicionales. Dicho componente permite a los mantenedores definir el ambiente de trabajo adaptándolo mejor a la aplicación en particular que se desea desarrollar.

- **Interfaz Hombre-Máquina**: Este proporciona al operador las funciones de control y supervisión sobre la planta, permitiéndole de esta forma un contacto directo con el sistema. Representa los procesos que ocurren en el campo en tiempo real, los componentes implicados, los sensores, las estaciones remotas, y el sistema de comunicación dándole al operador total control. Estos procesos son representados mediante sinópticos gráficos almacenados en el ordenador de proceso y generados desde el editor incorporado e importados desde otra aplicación durante la configuración del paquete [2].

- **Procesamiento**: Presenta como principales características la adquisición de datos del nivel de recolección en tiempo real, la conversión de unidades, el procesamiento de variables calculadas, la detección y el manejo de alarmas, el control de calidad de los datos recolectados, la publicación a los clientes de la actualización de los puntos, y la sucesión de las alarmas y los eventos. Ejecuta comandos sobre toda la información almacenada a nivel de campo.

- **Base de datos histórica:** Este se encargada del almacenamiento y procesamiento ordenado de los datos, de forma que otra aplicación o dispositivo pueda tener acceso a ellos.

- **Comunicación o Middleware:** Este módulo tiene como finalidad proporcionar la capa de comunicación de alto nivel, tanto sincrónica, como asincrónica, para la comunicación de todos los módulos que conforman el sistema [2].

1.3 Introducción a los simuladores

Los simuladores son objetos de aprendizaje que mediante un programa de software, intentan modelar parte de una réplica de los fenómenos de la realidad (Ver figura 2) y su propósito es que el usuario construya conocimiento a partir del trabajo exploratorio y la inferencia por descubrimiento. Estos se desarrollan en un entorno interactivo, que permite al usuario modificar parámetros y ver cómo reacciona el sistema ante el cambio producido. Tienen sus orígenes en la teoría de muestreo estadístico y análisis de sistemas físicos probabilísticos complejos. El aspecto común de ambos es el uso de números y muestras aleatorias para aproximar soluciones [3].

Antes de realizar un estudio de los simuladores es importante señalar dos conceptos fundamentales referentes a los tipos de variables que pueden soportar los mismos:

- **Variable analógica:**

  Una variable analógica es aquella que toma infinitos valores entre dos puntos cualesquiera de la misma. Un ejemplo de variable analógica es la temperatura. Si se analiza su variación durante un día, se observa que no puede pasar de un valor a otro dando un salto. Esto quiere decir que si la temperatura se incrementa de 11 grados a 17 grados, toma infinitos valores intermedios [4].

➢ **Variable digital:**

Una variable digital es aquella que solamente puede tomar un determinado número de valores. Un ejemplo de variable digital puede ser una alarma lanzada en un sistema informático para indicar alguna anomalía. Dentro de los tipos de variables digitales se encuentran las variables binarias las cuales toman solamente un estado de dos posibles. Algunos ejemplos son: una lámpara encendida o apagada, una llave cerrada o abierta, el abrir y cerrar una válvula, una compuerta, un circuito con tensión o sin tensión, es decir elementos que comúnmente son asociados a pulsos en el ámbito de la electricidad. La importancia de las variables binarias radica en el hecho de que los circuitos internos de las computadoras trabajan con ellas [4].

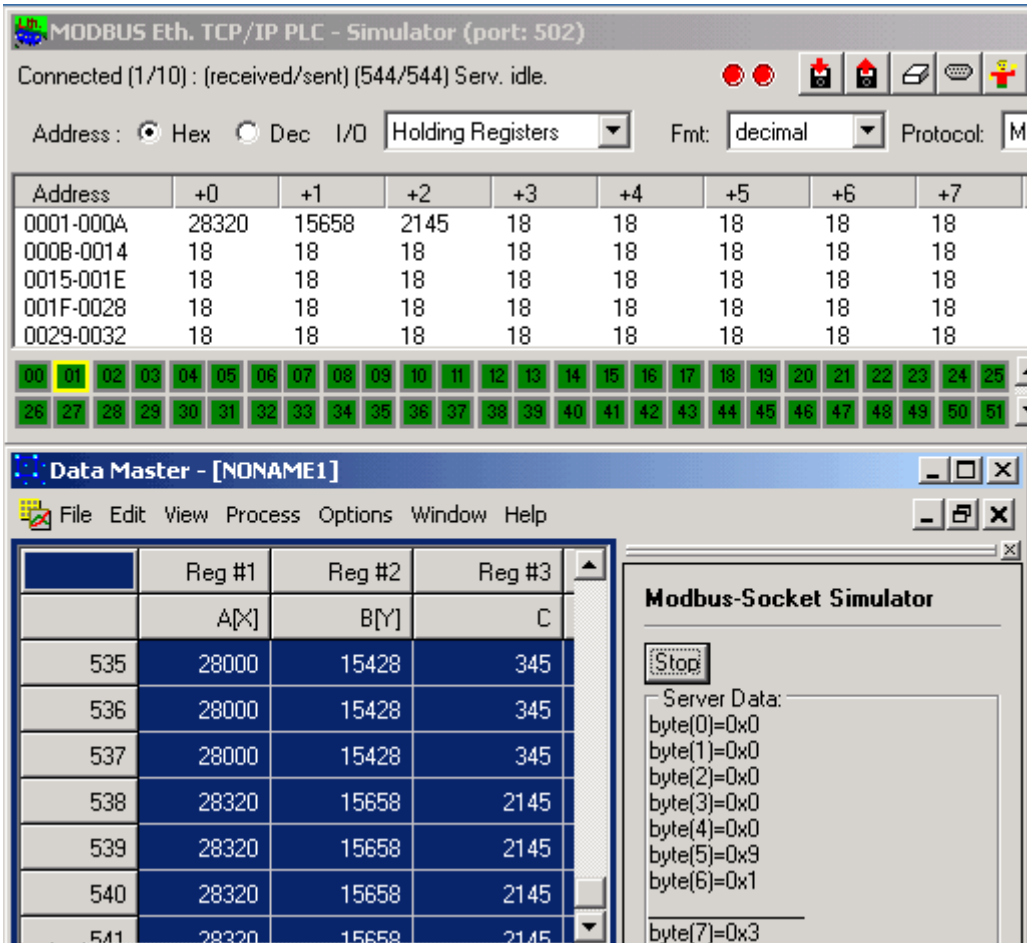

Figura 2 Simulador de puntos para un SCADA.

1.3.1 Clasificación de los simuladores

Existen diversos criterios según los cuales los simuladores pueden ser clasificados. Sobre la base de sus características de diseño y construcción, el simulador puede ser:

- **Genérico:** Estos no representan un determinado equipo (marca y modelo).

- **Semi-real:** En este caso el equipo representa un determinado modelo de la vida real, pero se utiliza una réplica que ha sido diseñada ajustándose al mismo que se utiliza en la realidad.

- **Real**: Se utiliza el mismo equipo que se usa en la vida real, simulando la operación del mismo [4]. La decisión del tipo de simulador a utilizar en un determinado programa, depende de las características del entrenamiento requerido y en todos los casos resulta de máxima importancia la decisión estratégica de un profesor o instructor experto en la materia. Es la forma de incorporar el equipo adecuado en el momento adecuado.

1.3.2 Metodología del proceso de simulación

Planificar un proceso de simulación requiere de los siguientes pasos:

1. **Formulación del problema**

   En este se definen las cuestiones para las que se buscan las respuestas, las variables implicadas y las medidas de ejecución que se van a usar. Esta fase es muy importante para poder alcanzar un modelo válido [3].

2. **Colección de datos y análisis**

   Durante este paso se recoge el mayor volumen de datos, se reduce y se analiza. Los métodos de recogida de datos son tan variados como los problemas a los que éstos se pueden aplicar. Si se clasifican por su sencillez, se puede ir desde las aproximaciones manuales hasta las técnicas más sofisticadas de alta tecnología.

3. **Desarrollo del modelo**

   Incluye la construcción y depuración del modelo del sistema real, incluyendo la selección de un lenguaje de programación, codificación del modelo.

4. Verificación y validación del modelo

La verificación del modelo seleccionado consiste en ver cuál es la consistencia interna del mismo, mientras que la validación asegura la existencia de la correspondencia entre el sistema real y el modelo. Un buen método para la validación es hacer un test para ver cómo el modelo predice el comportamiento del sistema ante determinadas entradas [3].

La verificación y validación del modelo se realiza en todos los niveles de modelización: modelo conceptual, modelo lógico y un modelo de ordenador. La verificación se centra en la consistencia interna del modelo, mientras que la validación se interesa por la correspondencia entre el modelo y la realidad.

5. Experimentación y análisis de las salidas

Se han de diseñar los experimentos que se van a llevar a cabo sobre el modelo y luego analizar las salidas obtenidas, de forma que se pueda responder a las cuestiones que se plantearon.

6. Implantación de los resultados de la simulación

Este paso final es uno de los más importantes y el que más se descuida de todo el proceso. Parece obvio que los beneficios de un largo y costoso análisis no se realizarán sin una implementación apropiada y una aceptación por parte de los usuarios [3].

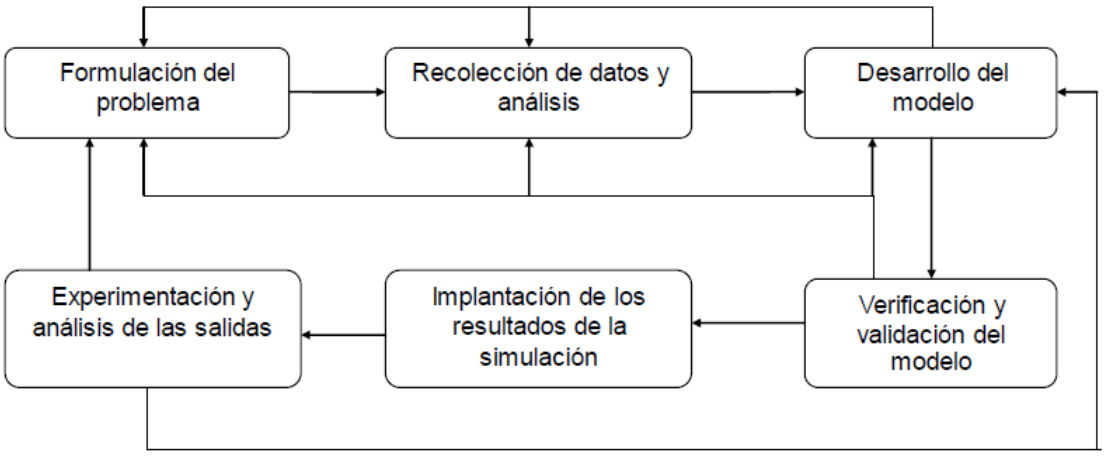

Figura 3 Pasos en la simulación

1.3.3 Modelos de simulación y sus clasificaciones

Un modelo de simulación es una representación simplificada de la realidad diseñada para representar, conocer o predecir propiedades del objeto real. Estos se construyen con una finalidad: estudiar el objeto real con más facilidad y deducir propiedades difíciles de observar en la realidad. Los modelos de simulación pueden representar objetos o procesos (simulación) [5].

Existen múltiples tipos de modelos de simulación [5], para representar la realidad entre los cuales se definen:

- **Dinámicos:** Estos se utilizan para representar sistemas cuyo estado varía con el tiempo.
- **Estáticos:** Se utilizan para representar sistemas cuyo estado es invariable a través del tiempo.
- **Matemáticos:** Representan la realidad en forma abstracta de diversas maneras.
- **Físicos:** Son aquellos modelos en que la realidad es representada por algo tangible, construido en escala o que por lo menos se comporta en forma análoga a esa realidad (maquetas, prototipos o modelos analógicos).
- **Analíticos:** La realidad se representa por fórmulas matemáticas. Estudiar el sistema consiste en operar con esas fórmulas matemáticas (resolución de ecuaciones).
- **Numéricos:** Se tiene el comportamiento numérico de las variables que intervienen en el proceso. No se obtiene ninguna solución analítica.
- **Continuos:** Representan sistemas cuyos cambios de estado son graduales. Las variables intervinientes son continuas.
- **Discretos:** Representan sistemas cuyos cambios de estado son de saltos. Las variables varían en forma discontinua.
- **Determinísticos:** Son modelos cuya solución para determinadas condiciones es única y siempre la misma.

- **Estocásticos:** Representan sistemas donde los hechos suceden al azar, lo cual no es repetitivo. No se puede asegurar cuáles acciones ocurren en un determinado instante. Se conoce la probabilidad de ocurrencia y su distribución probabilística.

1.3.4 Simulación por computadora

Es un intento de modelar situaciones de la vida real por medio de un programa de computadora, lo que requiere ser estudiado para ver cómo es que trabaja el sistema, ya sea por cambio de variables o quizás predicciones hechas acerca del comportamiento del mismo.
La simulación por computadora se ha convertido en una parte útil del modelado de muchos sistemas naturales como física, química, biología y en sistemas humanos como la economía y las ciencias_sociales (sociología computacional). Esta simulación es frecuentemente usada como un accesorio o para sustitución de sistemas de modelado para los cuales las soluciones analíticas de forma cerrada simple no son posibles. Ahí se encuentran muchos tipos diferentes de simulación por computadora, la característica común que todos ellos comparten es el intento por generar una muestra de escenarios representativos para un modelo en que una enumeración completa de todos los estados posibles serían prohibitivos o imposibles [4]. Es importante señalar como uno de los tipos de simulación más relevantes el método de Monte Carlo, el cual es un método no determinístico o estadístico numérico, usado para aproximar expresiones matemáticas complejas y costosas de evaluar con exactitud. El método se llamó así en referencia al Casino de Monte Carlo (Principado de Mónaco) por ser “la capital del juego de azar”, al ser la ruleta un generador simple de números aleatorios. El nombre y el desarrollo sistemático de los métodos de Monte Carlo datan aproximadamente de 1944 y se mejoraron enormemente con el desarrollo de la computadora[4].

El uso de los métodos de Monte Carlo como herramienta de investigación, proviene del trabajo realizado en el desarrollo de la bomba atómica durante la Segunda Guerra Mundial en el Laboratorio Nacional de Los Álamos en EE. UU. Este trabajo conllevaba la simulación de problemas probabilísticos de hidrodinámica concernientes a la difusión de neutrones en el material de fisión. Esta difusión posee un comportamiento eminentemente aleatorio.

El método de Monte Carlo proporciona soluciones aproximadas a una gran variedad de problemas matemáticos posibilitando la realización de experimentos con muestreos de

números pseudoaleatorios en una computadora. El método es aplicable a cualquier tipo de problema, ya sea estocástico o determinista[4].

1.4 Ventajas y desventajas del uso de la simulación

1.4.1 Ventajas

Aunque la técnica de simulación generalmente se ve como un método de último recurso, recientes estudios realizados y la gran disponibilidad de software que actualmente existe en el mercado, han hecho que esta técnica sea una de las herramientas más usadas en análisis de sistemas. Se ha demostrado a través de profundas investigaciones que un estudio de este tipo es muy recomendable ya que presenta las siguientes ventajas [7]:

- A través de un estudio de simulación, se puede estudiar el efecto de cambios internos y externos de un sistema, al hacer alteraciones en el modelo del mismo y observando los efectos de esas alteraciones en su comportamiento.

- Una observación detallada del sistema que se está simulando puede conducir a un mejor entendimiento del mismo y por consiguiente a sugerir estrategias que mejoren la operación y eficiencia de este.

- Esta técnica puede ser utilizada para experimentar con nuevas situaciones, sobre las cuales se tiene poca o ninguna información. A través de esta experimentación se puede anticipar mejor a posibles resultados no previstos.

- Cuando nuevos elementos son introducidos en una aplicación, la simulación puede ser usada para anticipar cuellos de botella o algún otro problema que puede surgir durante el desarrollo de dicho sistema.

- Los sistemas, los cuales son sujetos de investigación de su comportamiento, no necesitan existir actualmente para ser sujetos de experimentación basados en la simulación, solo necesitan existir en la mente del diseñador.

- El tiempo puede ser compensado en los modelos de simulación. El equivalente de días, semanas y meses de un sistema real en operación frecuente puede ser simulado en solo segundos, minutos u horas en una computadora. Esto significa que un largo número de alternativas de solución pueden ser simuladas y los resultados pueden estar disponibles de forma breve y ser suficientes para influir en la elección de un diseño para un sistema.

- Cada variable puede sostenerse constante excepto algunas cuya influencia está siendo estudiada. Como resultado del posible efecto de descontrol de las variables en el comportamiento del sistema necesitan no ser tomados en cuenta. Como frecuentemente debe ser hecho cuando el experimento está desarrollado sobre un sistema real.

- Es posible reproducir eventos aleatorios idénticos mediante una secuencia de números aleatorios. Esto hace posible usar las técnicas de reproducción de varianza para mejorar la precisión con la cual las características del sistema pueden ser estimadas para dar un valor que refleje el esfuerzo de la simulación.

1.4.2 Desventajas

- Falta de precisión en el comportamiento del circuito. Por ejemplo, las resistencias de carbón: Las resistencias reales tienen tolerancia. Las resistencias virtuales son exactas.

- Comportamiento erróneo de los modelos. En circuitos digitales, las entradas TTL al aire son "unos" lógicos, en el simulador, son "ceros" lógicos.

- Educativamente hablando, no se logra la misma experiencia que en un sistema real. Si el profesor no está de acuerdo con la filosofía de este tipo de material, y cree que sus estudiantes no serán capaces de lograr lo compuesto, no se sacará provecho de este tipo de material.

1.5 Simuladores utilizados en los sistemas SCADA

En el mundo existen diferentes simuladores para la generación de valores numéricos con el fin de utilizarlos en pruebas para diferentes procesos en los sistemas SCADA [8], entre ellos hacemos referencia a dos de los más utilizados en el proyecto SCADA GALBA.

➢ ***Honeynet Project***: Su objetivo es determinar la viabilidad de la construcción de un marco basado en software para simular una variedad de redes industriales tales como SCADA, PLC (Controlador Lógico Programable) y arquitecturas. Permiten simular una variedad de redes industriales tales como arquitecturas de SCADA, de DCS (Sistema de Control Distribuido) y de PLC. Este *honeypot(*software o conjunto de computadores cuya intención es atraer a atacantes, simulando ser sistemas vulnerables o débiles a los ataques*)* no solo puede servir de estudio de ataques que puede sufrir una red industrial, aplicando las cualidades de *Honeyd (tipo de honeypot)* es posible utilizarlo como técnica de camuflaje ante ataques de *Fingerprinting*, ya que entre las opciones de *Honeyd* es posible configurar que solo responda a un rango de IP determinado o que responda en una franja horaria concreta[8].

➢ **MODSIM** (*Modular simulator for ore dressing plants*): Es un software de simulación de plantas de procesamiento de minerales. Fue desarrollado debido a los exitosos resultados logrados por la creación de un simulador de plantas de flotación durante un programa de investigación del *Mintek's Chemical Engineering Research Group* del Departamento de Ingeniería Química de la Universidad de Natal, Durban, Sudáfrica. La aplicación extraordinaria del simulador a la industria (*King, Pugh* y *Langely* 1973), [8] activó el desarrollo de un simulador de circuitos arbitrarios y complejos de diversas operaciones asociadas al procesamiento de minerales. Ver figura 4.

➢ **MPSS / ESN** : Es un simulador de sistemas de alimentación modular con sistema de control SCADA y Sistema SCADA-NET. Está especialmente diseñado para ser manejado por usuarios a nivel técnico y profesional o el nivel inicial en la educación superior. El simulador incluirá las partes principales de un sistema de energía como: generación, transformación, transporte, distribución y consumo. Incluye varios elementos que juegan un rol muy importante en el control de sistemas de energía y la protección de los mismos, entre ellos se encuentran: el regulador automático de voltaje, el control automático de frecuencia, sincronización manual y automática, entre otros [8].

1.6 Tendencias y Tecnologías

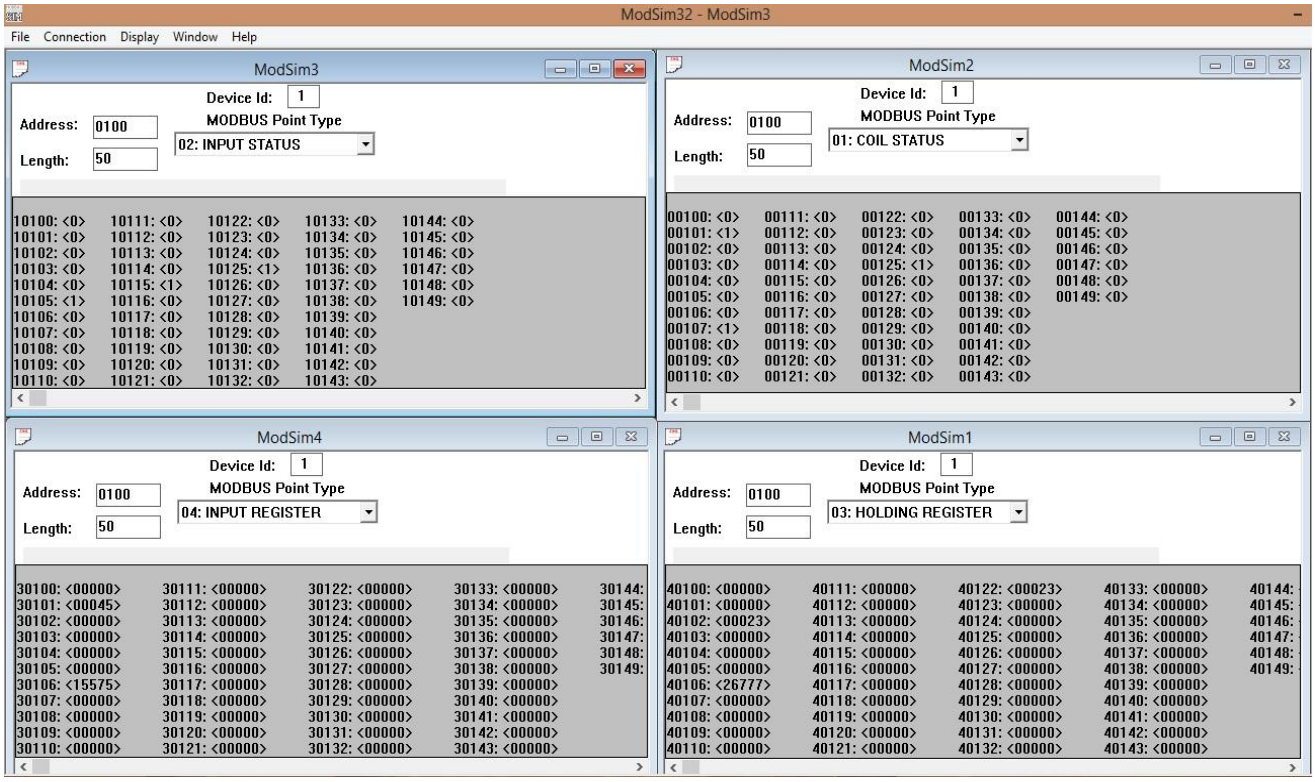

Figura 4 Tipos e simulaciones generadas por el MODSIM

1.6.1 QT Framework

Qt es una biblioteca multiplataforma ampliamente usada para desarrollar aplicaciones con interfaz gráfica de usuario, así como también para el desarrollo de programas sin interfaz gráfica, como herramientas para la línea de comandos y consolas para servidores. Es desarrollada como un software libre y de código abierto a través de Qt Project, donde participa tanto la comunidad, como desarrolladores de Nokia, Digia y otras empresas. Qt es utilizada en KDE (equipo internacional que coopera en el desarrollo y distribución de software libre y de código abierto para computadoras de escritorio y portátiles), entorno de escritorio para sistemas como GNU/Linux o FreeBSD (sistema operativo multiusuario, capaz de efectuar multitarea con apropiación y multiproceso en plataformas compatibles con múltiples procesadores), entre otros. Utiliza el lenguaje de programación C++ de forma nativa, adicionalmente puede ser

utilizado en varios otros lenguajes de programación a través de *bindings (*adaptación de una biblioteca para ser usada en un lenguaje de programación distinto de aquél en el que ha sido escrita.*)*. Funciona en todas las principales plataformas, y tiene un amplio apoyo. El API de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL, así como uso de XML, gestión de hilos, soporte de red, una API multiplataforma unificada para la manipulación de archivos y una multitud de otros para el manejo de ficheros, además de estructuras de datos tradicionales. Entre sus principales componentes se encuentran:

- **Las bibliotecas Qt** (clases implementadas en C++).
- **Qt Designer**: Para diseñar formularios de forma visual.
- **Qt Assistant**: Acceso rápido a la documentación.
- **Qt Linguist**: Traducción rápida de programas.
- **Qmake**: Simplifica el proceso de construcción de proyectos en las diferentes plataformas soportadas.

1.6.2 Colección de bibliotecas Boost

Boost es un conjunto de bibliotecas de software libre y revisión por pares(método usado para validar trabajos escritos y solicitudes de financiación con el fin de evaluar su calidad, originalidad, factibilidad, rigor científico, entre otros) preparadas para extender las capacidades del lenguaje de programación C++. Su licencia, de tipo BSD(Berkeley Software Distribution), permite que sea utilizada en cualquier tipo de proyectos, ya sean comerciales o no. Su diseño e implementación permiten que sea utilizada en un amplio espectro de aplicaciones y plataformas. Abarca desde bibliotecas de propósito general hasta abstracciones del sistema operativo. Con el objetivo de alcanzar el mayor rendimiento y flexibilidad se hace un uso intensivo de plantillas. Boost ha representado una fuente de trabajo e investigación en programación genérica y metaprogramación en C++.

Actualmente Boost está formada por más de 80 bibliotecas individuales, incluidas las bibliotecas de álgebra lineal, la generación de números pseudoaleatorios, multihilos,

procesamiento de imágenes, expresiones regulares, pruebas unitarias, entre otros. La mayoría de las bibliotecas Boost están basadas en cabeceras, funciones en línea y plantillas, por lo que no tienen que ser construidas antes de su uso. Entre las principales funcionalidades que permite desarrollar se encuentran:

- Cadenas y procesamiento de texto
- Contenedores
- Iteradores
- Algoritmos
- Programación genérica
- Meta programación con plantillas
- Meta programación de procesos
- Programación concurrente
- Análisis matemático y numérico
- Corrección y prueba
- Estructura de datos
- Entrada/salida
- Soporte para metalenguaje
- Serialización

Esta biblioteca es una especie de antesala del estándar C++, ya que algunos de sus componentes son analizados y evaluados por los miembros del comité como candidatos a ser incluidos en futuras revisiones del estándar C++.

1.6.3 Software Libre. GNU/Linux

El Software libre brinda libertad a los usuarios sobre el producto adquirido y por tanto, el mismo puede ser usado, copiado, estudiado, modificado y redistribuido libremente. Brindando de esta forma libertades a los usuarios sobre el software: la libertad de usar el programa, con cualquier propósito; de estudiar el funcionamiento del programa, y adaptarlo a las necesidades; de distribuir copias, con lo que puede ayudar a otros; de mejorar el programa y hacer públicas las mejoras, de modo que toda la comunidad se beneficie, así mismo ofreciendo el acceso pleno al código fuente sin ningún requisito auxiliar.

Una distribución de GNU/Linux es una variante de ese Sistema Operativo (SO) que incorpora determinados paquetes de software para satisfacer las necesidades de un grupo específico de usuarios, dando así origen a ediciones hogareñas, empresariales y para servidores. Pueden ser exclusivamente de software libre, o también incorporar aplicaciones o controladores propietarios. Una gran parte de las herramientas básicas que completan el SO, vienen del proyecto GNU; de ahí el nombre: GNU/Linux. La distribución de GNU/Linux seleccionada es Ubuntu v10.04. Esta es una distribución basada en Debian que proporciona un SO actualizado y estable para el usuario promedio, enfocándose en la facilidad de instalación y uso del sistema. Se compone de múltiples paquetes de software en su mayoría distribuidos bajo una licencia libre o de código abierto.

1.6.5 Lenguaje de programación C++

Fue diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue extender al lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido. Posteriormente se añadieron facilidades para la programación genérica, que se sumó a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje multi-paradigma [11].

Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales. Está categorizado por muchos como el lenguaje más potente ya que permite trabajar tanto a alto como a bajo nivel.

1.6.6 Entorno de desarrollo Integrado

Los entornos de desarrollo integrados (IDE) actualmente están considerados como un conjunto de herramientas indispensables para un desarrollador de software y a la vez, pueden ser utilizados para uno o varios lenguajes de programación. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI. Pueden ser aplicaciones por si solas o pueden ser parte de aplicaciones existentes. Proveen un marco de

trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Java, C#, Visual Basic, Object Pascal, etc. Otra característica importante es que pueden funcionar como un sistema en tiempo de ejecución en algunos lenguajes de programación. En la actualidad existe una gran variedad de los mismos, entre los más usados por los desarrolladores de GNU/Linux tenemos [9]:

- **Eclipse:** Es uno de los entornos de desarrollo integrados de código abierto y extensible más utilizados en los últimos tiempos. Este incluye varias características únicas, como la refactorización de código, la actualización/instalación automática de código (mediante Update Manager), una lista de tareas, soporte para unidades de test con JUnit, e integración con la herramienta de construcción de Jakarta: Ant. A pesar del gran número de características estándar, Eclipse se diferencia de los IDE tradicionales en varias cosas fundamentales. Quizás lo más interesante de Eclipse es ser completamente neutral a la plataforma - y al lenguaje. Este entorno emplea módulos (en inglés *plugin*) para proporcionar toda su funcionalidad. Este mecanismo de módulos es una plataforma ligera para componentes de software. Por otra parte permite a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Phyton y trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y sistema de gestión de base de datos. La arquitectura plugins permite escribir cualquier extensión deseada en el ambiente, como la gestión de la configuración. Se provee soporte para Java y el sistema de control de versiones en el SDK (Kit de Desarrollo de Software) de Eclipse.

- **Qt Creator:** Entorno de Desarrollo creado por Trolltech, multiplataforma, diseñado para hacer que el desarrollo en C++ de la aplicación Qt sea más rápido y fácil. Qt Creator no quiere ser un reemplazo de Eclipse ni Visual Studio, sino un IDE ligero pensado especialmente para el desarrollo en múltiples plataformas: Windows XP y Vista, Linux (desde la versión 2.6) y Mac OSX (desde 10.4 en adelante). Posee como principales características un avanzado editor de código C++, además soporta los lenguajes: C#/.NET Languages (Mono), Python: PyQt , Ada, Pascal, Perl, PHP y Ruby, posee también una guía integrada y diseñador de formularios, herramientas para proyectos y administración, ayuda sensible al contexto integrada, depurador visual, resaltado y auto-completado de código y soporte para refactorización de código.

1.7 Metodologías de desarrollo de software

Una metodología es aquella guía que se sigue a fin de realizar las acciones propias de una investigación. En términos más sencillos se trata de la guía que nos va indicando qué hacer y cómo actuar cuando se quiere obtener algún tipo de investigación. Es posible definir una metodología como aquel enfoque que permite observar un problema de una forma total, sistemática y disciplinada [12]. El ciclo de vida del software es complejo, de ahí la necesidad de utilizar metodologías diferentes de acuerdo con las características del proyecto en desarrollo.

Dentro de estas metodologías una de las más usadas es XP (Programación Extrema), centrada en satisfacer al cliente y potenciar al máximo el trabajo en equipo. Es una metodología ágil diseñada para entornos dinámicos, pensada para equipos pequeños (hasta 10 programadores), orientada fuertemente hacia la codificación, énfasis en la comunicación informal, verbal etc. Se deriva en seis fases que se mencionan a continuación:

1. Exploración.
2. Planificación de la Entrega (Release).
3. Iteraciones.
4. Producción.
5. Mantenimiento.
6. Muerte del Proyecto.

1.7.1 Metodología XP

Esta es una de las metodologías de desarrollo de software más utilizada en la actualidad, para proyectos de corto plazo y corto equipo. Consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final como uno de los requisitos para llegar al éxito del proyecto. En esta metodología, el cliente se convierte en un miembro más del equipo de trabajo y es el encargado de decidir que se implementa, puede añadir, cambiar o quitar requerimientos en cualquier momento para lo cual debe estar enterado constantemente del estado real y el progreso del proyecto obteniendo lo máximo de cada semana de trabajo. XP trabaja cuatro fases principales: Planificación, Diseño, Desarrollo y Pruebas. Estas fases se dividen, a su vez, en subfases de desarrollo que poseen una serie de pasos que permiten

realizar un adecuado desarrollo del proyecto, dentro de ellas la de planificación del proyecto, la cual se describe a continuación:

Planificación del proyecto

Historias de usuario: El primer paso de cualquier proyecto que siga la metodología XP es definir las historias de usuario (HU) con el cliente. Las HU tienen la misma finalidad que los casos de uso pero con algunas diferencias: constan de 3 ó 4 líneas escritas por el cliente en un lenguaje no técnico sin hacer mucho hincapié en los detalles; no se debe hablar ni de posibles algoritmos para su implementación ni de diseños de base de datos adecuados. Son usadas para estimar tiempos de desarrollo de la parte de la aplicación que describen. Se utilizan en la fase de pruebas, para verificar si el programa cumple con lo que especifica la HU, cuando llega la hora de implementarla, el cliente y los desarrolladores se reúnen para concretar y detallar lo que tiene que hacer dicha HU. El tiempo de desarrollo ideal es entre 1 y 3 semanas. Presenta los siguientes roles[11]:

- Administrador (programador): Su misión es administrador el ambiente de programación. Realiza las actividades de configurar el ambiente de programación.

- Programador: Su misión es ser responsable de implementar el código para dar soporte a los requisitos del cliente. Realiza las actividades de definir el estimado de la tarea, estimado del requisito del cliente, entre otros, generando los artefactos de XP construcción, XP pruebas unitarias.

- Probador: Su misión es ayudar al cliente a definir y escribir pruebas de aceptación para los requisitos. Realiza las actividades de correr las pruebas del cliente, automatizar las pruebas del cliente y configurar el ambiente del probador [11].

- Cliente: Su misión es ser responsable de definir cuál es el producto correcto a construir, determinar las características del mismo y asegurarse de que el producto realmente satisface sus requerimientos. Realiza las actividades de ajustar las iteraciones de las actividades, definir la iteración, definir la visión del documento, Escribir los requisitos, entre otros, generando los artefactos de requisitos, prueba del cliente, plan de iteraciones, entre otros.

- Encargado de Seguimiento: Su misión es medir y comunicar el progreso del proyecto. Realiza las actividades de seguimiento del progreso de la iteración y seguimiento del progreso de entrega del producto.

- Entrenador (coaching): Su misión es ayudar a mantener la disciplina y aprendizaje del equipo. realiza las actividades de explicar proceso, mejorar las habilidades del equipo, resolver conflictos, entre otros.

Esta metodología define un proceso iterativo e incremental, utiliza la programación en pareja, fomenta la reutilización de código y genera muy pocos artefactos, lo que acelera el proceso de terminación del software. Para proyectos de pequeña envergadura estas características resultan ventajas importantes. Por todo lo anterior se decidió escoger esta metodología para el desarrollo de este software.

1.8 Conclusiones parciales

En este capítulo se realizó un esbozo de las principales características de los simuladores, su importancia en varios ámbitos de la informática y las telecomunicaciones así como el rol que juega en la esfera de la industria y su aplicación en los sistemas SCADA de los que se mencionan y explican los módulos que lo componen. De los diferentes ejemplos de simuladores utilizados en los sistemas SCADA se toma como referencia el MODSIM y el MODBUS, los cuales presentan propiedades que servirán de referencia para la realización de este proyecto, pues se realizara un sistema con similares comportamientos.

Se realizó un análisis de las tecnologías a utilizar en el desarrollo del sistema, así como algunos conceptos y tendencias que este debe adoptar para sí. Se escoge como metodología a utilizar la ágil XP, como lenguaje de programación C++ y como IDE de desarrollo Eclipse, todo se desarrolla en el entorno de software libre Linux. Es importante mencionar también la utilización del método socket para la comunicación entre procesos. A partir de estos puntos se comenzará el desarrollo de la propuesta de sistema.

CAPÍTULO 2. Análisis y Diseño de la herramienta

El presente capítulo tiene como objetivo reflejar las actividades realizadas en los procesos de análisis y diseño del mecanismo de simulación. En el mismo se exponen los artefactos más importantes que describen el flujo normal de eventos que ocurren en el sistema tales como especificaciones de requisitos que rigieron el desarrollo de la solución, historias de usuarios, diagramas y las principales clases que componen el mecanismo, detallando la información del análisis y del diseño de la solución en cuestión.

2.1 Metodología XP

2.1.1 Actores del sistema

Se define como persona relacionada con el sistema, a aquella que interactúa de una forma u otra con el mecanismo propuesto, dígase vinculada al proceso de desarrollo, así como a las que interactúan con la herramienta de gestión de script.

Actores	**Descripción**
Desarrollador	Es la persona encargada de realizar la implementación de la herramienta de programación visual y desarrollar todas las especificaciones que requiera el cliente.
Usuario	Programador: Es la persona encargada de gestionar e implementar las funcionalidades del sistema.

Tabla 1 Actores del Sistema

2.1.2 Fase de exploración

La metodología XP, comienza en su ciclo de vida con la fase de exploración, proponiendo definir durante esta etapa el alcance general del proyecto; además, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo se realiza una familiarización con las herramientas, tecnologías y prácticas que se emplearán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. Las estimaciones realizadas en esta fase son primarias, ya que están basadas en datos de alto nivel y podrían variar cuando se analicen con mayor detalle en cada iteración. Para esta fase se sugiere una extensión de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

2.1.3 Historias de usuarios

Las historias de usuario son la técnica utilizada en XP para especificar los requisitos del sistema a desarrollar. El cliente describe y prioriza sus necesidades mediante estas descripciones cortas y escritas sin terminología técnica. Se realiza una por cada funcionalidad del sistema, se emplean para hacer estimaciones de tiempo y para el plan de lanzamientos. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Las historias de usuarios deben ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a las tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia de usuario. Como resultado del trabajo realizado durante la fase de exploración se identificaron las siguientes historias de usuario:

Historia de usuario	
Número: 1	**Nombre:** Crear, Modificar y Eliminar variable analógica y digital.
Usuario: Desarrollador	
Modificación de Historia Número:	**Iteración asignada: 1**
Prioridad del Negocio: Alta (Alta/Media/Baja)	**Puntos estimados:** 2
Riesgo en Desarrollo: (Alta/Media/Baja)	**Puntos reales:** 2
Descripción: Se debe dar la opción al usuario de crear una nueva variable, donde deberá definir qué tipo de variable va a simular, las cuales serán de dos tipos analógicas y digitales, las primeras serán valores enteros y las segundas tomarán uno de dos estados posibles: 0 ó 1. En cada caso existirán propiedades diferentes para cada tipo de variable. En el caso que se seleccione de tipo analógico el usuario deberá especificar la forma en que se realizará la simulación ya sea incremental, decremental o random. También se deberá definir el rango en que oscilarán las variables y el intervalo en segundos entre cada simulación. Para las variables de tipo digital sólo deberá definir si el método de simulación será random o toggle (aleatorio),	

así como también el intervalo en segundos para cada simulación. Todos los atributos de estas variables serán configurables y también podrán ser eliminadas las variables del sumario de puntos donde aparecerán todas las que han sido creadas.
Observaciones:

Tabla 2 Crear, Modificar y Eliminar variable analógica y digital

Historia de usuario	
Número: 1	**Nombre:** Iniciar proceso de simulación.
Usuario: Desarrollador	
Modificación de Historia Número:	**Iteración asignada: 1**
Prioridad del Negocio: Alta (Alta/Media/Baja)	**Puntos estimados:** 2
Riesgo en Desarrollo: (Alta/Media/Baja)	**Puntos reales:** 2
Descripción: Para iniciar el proceso se deberán habilitar las variables creadas que se muestran en el sumario y luego realizar la conexión mediante el menú “Connection” de la barra de menús con el módulo de Comunicación del SCADA. También se podrá inicializar la simulación al habilitar algunas de las variables luego de establecida la conexión, en este momento se comenzará a generarse los valores automáticamente.	
Observaciones:	

Tabla 3 Iniciar proceso de simulación

Historia de usuario	
Número: 1	**Nombre:** Visualizar variables configuradas en tiempo real.
Usuario: Desarrollador	
Modificación de Historia Número:	**Iteración asignada: 1**
Prioridad del Negocio: Alta (Alta/Media/Baja)	**Puntos estimados:** 2
Riesgo en Desarrollo:	**Puntos reales:** 2

(Alta/Media/Baja)	
Descripción: Permite la visualización de las variables ya configuradas, mostrando los valores generados por el sistema.	
Observaciones:	

Tabla 4 Visualizar variables configuradas en tiempo real

2.1.5 Diseño de Casos de Prueba

A diferencia de las metodologías tradicionales, donde la fase de pruebas, incluyendo la definición de las mismas, generalmente se realiza al final del proyecto, o sobre el final del desarrollo de cada módulo; la metodología XP propone un modelo inverso, en el que, lo primero que se describe son las pruebas que el sistema debe pasar [1].

Pruebas de aceptación

Las pruebas de aceptación son creadas en base a las historias de usuarios definidas por el cliente. Dichas pruebas son consideradas como pruebas de caja negra y los clientes son los responsables de verificar que el resultado de estas pruebas sean los correctos. Una HU puede tener más de una prueba de aceptación, tantas como sean necesarias para garantizar un correcto funcionamiento.

Para la realización de las pruebas a la herramienta se diseñaron 10 casos de pruebas donde se ejecutan paso a paso cada una de las posibles entradas al sistema y se evalúa el resultado obtenido de las mismas. Las tablas con los diseños de casos de prueba se encuentran en el próximo capítulo donde además se incluyen los resultados de la ejecución de las mismas.

2.1.6 Fase de planificación

La planificación es una fase corta donde el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación y el resultado es un Plan de Entregas.

Esta fase dura unos pocos días. Las estimaciones de esfuerzo asociadas a la implementación de las historias la establecen los programadores utilizando como medida el punto, lo que equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos.

La planificación se puede realizar basándose en el tiempo o el alcance. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuantos puntos se

pueden completar. Al planificar según el alcance, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación.

2.1.7 Estimación de esfuerzos por historia de usuario

Para realizar un buen desarrollo del sistema propuesto, se realizó una estimación para cada una de las historias de usuario identificadas, llegando a los resultados que se muestran a continuación:

Historias de usuario	Puntos de estimación
Crear, Modificar y Eliminar variable analógica y digital.	3 Semanas
Iniciar proceso de simulación.	3 Semanas
Visualizar variables configuradas en tiempo real.	3 Semanas

Tabla 5 Estimación de esfuerzos

2.1.8 Plan de iteraciones

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de entrega está compuesto por iteraciones de no más de 3 semanas de duración. En la primera iteración se intenta establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto; esto se logra estableciendo las historias que fuercen la creación de la arquitectura antes mencionada, sin embargo, esto no siempre es posible ya que es el cliente quien decide que historias se implementarán en cada iteración. Al final de la última iteración el sistema estará listo para entrar en producción.

Una vez definidas las HU y estimado el esfuerzo propuesto para la realización de cada una de ellas, se decide realizar el sistema en 3 iteraciones, las cuales se describen a continuación de manera más detallada:

Iteración 1

Esta iteración tiene como objetivo darle cumplimiento a las HU que se consideraron con una mayor importancia para el desarrollo de la herramienta. Al concluir dicha iteración se contará con todas las funcionalidades descritas en la HU 1, la cual alude a la creación, modificación y eliminación de variables analógicas y digitales y la asociación de este proceso a los recursos y controles.

Iteración 2

Esta iteración tiene como objetivo darle cumplimiento a la HU 2, la cual responde a la inicialización del proceso de simulación proporcionando un fácil manejo de las opciones implementadas para llevar a cabo el proceso de simulación.

Iteración 3

Esta iteración tiene como finalidad darle cumplimiento a la HU3, la que se encarga de la visualización de las variables configuradas en tiempo real. Una vez desarrollada esta visualización se podrán apreciar los valores que se simulan cada un intervalo de tiempo, parámetro que se define independientemente para cada variable.

2.1.9 Plan de duración de las iteraciones

Como parte del ciclo de vida de un proyecto guiado por la metodología de desarrollo XP, se crea el plan de duración de cada una de las iteraciones que se llevarán a cabo durante el desarrollo del proyecto. Este plan tiene como objetivo fundamental mostrar la duración de cada iteración, así como el orden en que serán implementadas las HU en cada una de las mismas según la prioridad asignada por el cliente.

Iteración	Historias de usuario	Duración total de las iteraciones
Iteración 1	Crear, Modificar y Eliminar variable analógica y digital.	3 Semanas
Iteración 2	Iniciar proceso de simulación.	3 Semanas
Iteración 3	Visualizar variables configuradas en tiempo real.	3 Semanas

Tabla 6 Historias de usuario

2.2 Diseño de la solución propuesta

La metodología XP, no requiere la descripción del sistema mediante diagramas de clase utilizando la notación UML, sino que en su lugar se guía por técnicas como las tarjetas CRC (Contenido, Responsabilidad y Colaboración). Esto no implica que no se utilicen dichos diagramas para obtener una mejor visión y comunicación entre el equipo de trabajo, siempre y cuando no posea una alta complejidad y defina información importante.

2.2.1 Tarjetas CRC

Las características más sobresalientes de las tarjetas CRC son su simpleza y ductilidad. Una tarjeta CRC no es más que una ficha de papel o cartón que representa a una entidad del sistema [2].

Las tarjetas CRC se utilizan para estructurar las clases y a su vez definir las responsabilidades sobre las mismas, así como la simulación de escenarios en el sistema. A continuación se definen las tarjetas CRC del sistema:

VariablesIO	
Súper clase: IO::VariablesIO	
Sub Clase(s): -	
Responsabilidades Clase encargada de realizar la conexión con el módulo de Comunicación del SCADA GALBA y recibir los datos que se están simulando.	**Colaboraciones:**

Tabla 7 Tarjeta CRC VariablesIO

Connection	
Súper clase: IO::Connection	
Sub Clase(s):-	
Responsabilidades: Clase que representa una conexión, utilizada por la clase VariablesIO para realizar la conexión con el módulo de Comunicación del SCADA GALBA.	**Colaboraciones:**

Tabla 8 Tarjeta CRC Connection

DataPoint	
Súper clase:-	
Sub Clase(s):-	
Responsabilidades: Clase que representa de un punto interno en el simulador que luego al pasar por la biblioteca Simulator se convertirá a un tipo de dato Variable.	**Colaboraciones:**

Tabla 9 Tarjeta CRC DataPoint

Simulator	
Súper clase: SimulatorData	
Sub Clase(s):-	
Responsabilidades: Clase que se encarga de simular los datos y publicarlos a través de una interfaz.	**Colaboraciones:**

Tabla 10 Tarjeta CRC Simulator

MainView	
Súper clase: -	
Sub Clase(s):-	
Responsabilidades: Clase encargada de la configuración de las variables para su posterior visualización a través de la interfaz visual correspondiente.	**Colaboraciones:**

Tabla 11 Tarjeta CRC MainView

PointConfig	
Súper clase: QDialog	
Sub Clase(s): -	
Responsabilidades: Representación de una punto analógico interno en el simulador.	**Colaboraciones:**

Tabla 12 Tarjeta CRC PointConfig

DigitalConfig	
Súper clase: QDialog	
Sub Clase(s): -	
Responsabilidades: Representación de una punto digital interno en el simulador.	**Colaboraciones:**

Tabla 13 Tarjeta CRC DigitalConfig

2.2.2 Diagrama de Clases

La metodología XP plantea que para un mejor entendimiento de las tareas, flujos y métodos de desarrollo de las funcionalidades se pueden crear diagramas, siempre y cuando su creación no implique un mayor esfuerzo que la implementación del mismo. Siguiendo este principio se elaboró el diagrama de clases, que muestra las dependencias lógicas entre las clases que conforman el software. Estos

diagramas prevalecen en el campo de la arquitectura de software pero pueden ser utilizados para modelar y documentar cualquier arquitectura de sistema [3].

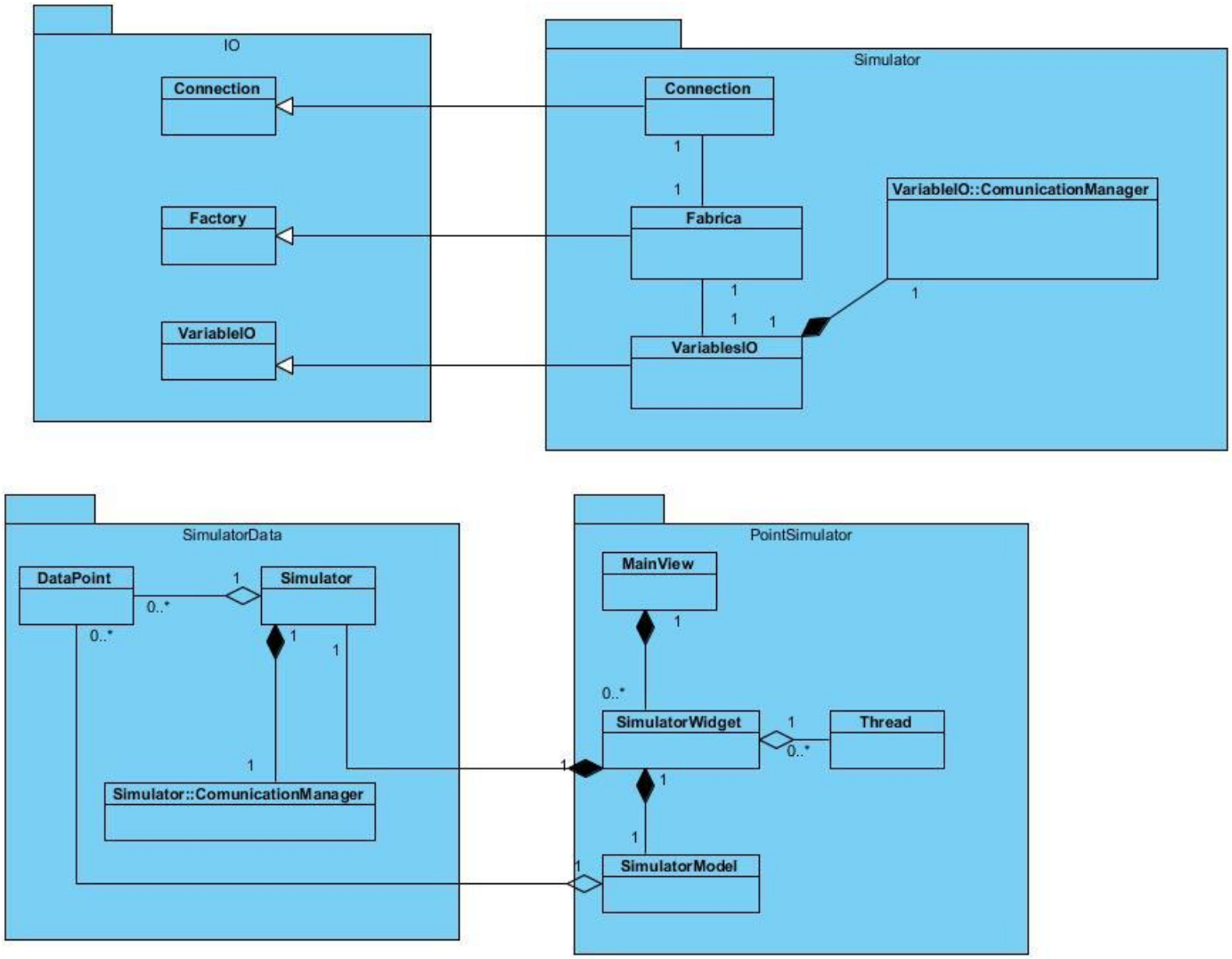

Figura 5 Diagrama de Clases

2.2.3 Estándar de codificación.

Los estándares de codificación, también conocidos como estilos de programación o convenciones de código, no son más que convenios para escribir código fuente en ciertos lenguajes de programación. Permiten que el código en consecuencia sea mantenible y que todos los participantes lo puedan entender en un menor tiempo. [4]

Para la implementación del mecanismo en cuestión es necesario utilizar el Estándar de codificación de C++ para el proyecto SCADA GALBA. Algunas de las pautas que define el estándar utilizado define:

- En los archivos cabecera debe incluir el copyright y la licencia, o una referencia de la misma, al

estilo GNU GPL.

- Se adopta el estilo de bloques de documentación de JavaDoc, el cual consiste de un bloque de comentario de estilo C.
- Para hacer una descripción breve se adopta el uso del comando @brief.
- Es importante especificar el nombre del autor y la fecha de creación de cualquier estructura en un código, para ello se utilizan los comandos @autor y @date.
- Para hacer referencia a otras clases utilizar el comando @see.
- El código será escrito en inglés y la documentación en español.
- Las variables y funciones comienzan con letra minúscula. Cada palabra consecutiva en el nombre comienza con letra mayúscula.

2.3 Conclusiones parciales

En este capítulo se inicia el desarrollo de la propuesta de solución que se desea implementar. Se trata todo lo referente a las dos primeras fases de la metodología de desarrollo de software a utilizar, fase de exploración y planificación del sistema, donde se documentaron todos los artefactos generados en el transcurso de las mismas. Quedó plasmado que el desarrollo de la aplicación se realizará en 3 iteraciones y programado en pareja, como lo propone la metodología utilizada.

CAPÍTULO 3. Implementación y Prueba

En el presente capítulo se detallarán las tres iteraciones llevadas a cabo durante la etapa de construcción del sistema, definiendo las tareas generadas a partir del desarrollo de las historias de usuario durante las iteraciones planificadas. Además quedarán especificados los resultados obtenidos de la ejecución de las pruebas de aceptación previamente diseñadas para probar las funcionalidades descritas y la valoración de dichas pruebas de manera cualitativa.

3.1 Desarrollo de las iteraciones

En la fase de Planificación se detallaron las historias de usuarios correspondientes a cada una de las iteraciones para desarrollar el sistema, teniendo presente las necesidades requeridas por el cliente. Durante el transcurso de cada iteración se realizó una revisión del plan de iteraciones. Como parte de este plan se desglosaron las historias definidas en tareas de programación o ingeniería, asignándole a un equipo de desarrollo o a una persona la responsabilidad de su implementación. Estas tareas son para el uso estricto del programador, por lo que no fue necesario escribirlas en un lenguaje no técnico para hacerla entendible al cliente.

A continuación se especifica con mayor detalle las tareas de desarrollo que se realizaron en cada una de las iteraciones:

Iteración 1

Esta iteración tuvo como objetivo dar cumplimiento a la HU que se consideró con una mayor importancia para el desarrollo de la aplicación. Al concluir dicha iteración se contó con todas las funcionalidades descritas en las HU1, la cual se refiere a la creación, modificación y eliminación de variables analógicas y digitales.

Historias de usuario	Tiempo de implementación (semanas)	
	Estimación	Real
Crear, modificar y eliminar variable analógica y digital.	3	3
Total	3	3

Tabla 14 Tiempo de estimación para la primera iteración

Tarea de Ingeniería

No. de la tarea: 1	**No. de la HU:** 1
Nombre de la tarea: Crear variable analógica y digital	
Tipo de tarea: Implementación.	**Puntos estimados:** 1
Fecha inicio: 10/11/2012	**Fecha fin:** 15/11/2012
Programador responsable: Darlin Mercedes Blanco García.	
Descripción: Funcionalidad encargada de crear variables analógicas y digitales.	

Tabla 15 Tarea de ingeniería "Crear variable analógica y digital"

El proceso de simulación inicia con la creación de un nuevo proyecto, opción que se brinda en el menú principal de la aplicación y en el acceso directo de la barra de herramientas. Posteriormente se escoge el tipo de simulación (analógico o digital) que se va a realizar, según las opciones que se muestran en la parte superior de la ventana, ver figura 6. Estos procesos también otorgaran permisos de lectura o escritura para cada simulación según los intereses del cliente. Luego se escoge la cantidad de puntos o variables a simular y en este momento es donde se crean tantas variables como el número escogido teniendo en cuenta el tipo de simulación.

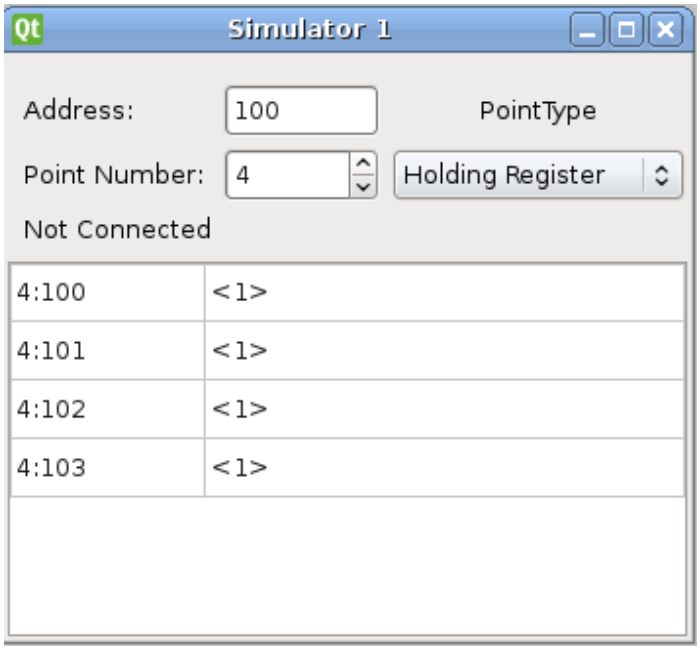

Figura 6 Interfaz Visual "Creación de variable analógica"

Tarea de Ingeniería

No. de la tarea: 2	**No. de la HU:** 1
Nombre de la tarea: Modificar variable analógica y digital.	
Tipo de tarea: Implementación.	**Puntos estimados:** 0.16
Fecha inicio: 16/11/2012	**Fecha fin:** 16/11/2012
Programador responsable: Darlin Mercedes Blanco García.	
Descripción: Funcionalidad encargada de la modificación de las variables creadas en el sumario de puntos.	

Tabla 16 Tarea de ingeniería "Modificar variable analógica y digital"

Esta funcionalidad tendrá lugar cuando el usuario seleccione un punto del sumario que se muestra en el área central de la ventana correspondiente al proyecto donde se trabaja. Cada una de estas variables podrán ser modificadas haciendo doble click sobre el punto seleccionado y modificar sus valores según el tipo de variable como se observa en las figuras 8 y 9. Para las variables de tipo analógico se configura el tipo de simulación, el intervalo de tiempo en segundos, el valor que se tendrá en cuenta para cuando el tipo de simulación sea incremental o decremental y el rango en el cual estará acotada la simulación. Después de dar en el botón aceptar la simulación se adaptará según los datos configurados para la variable seleccionada.

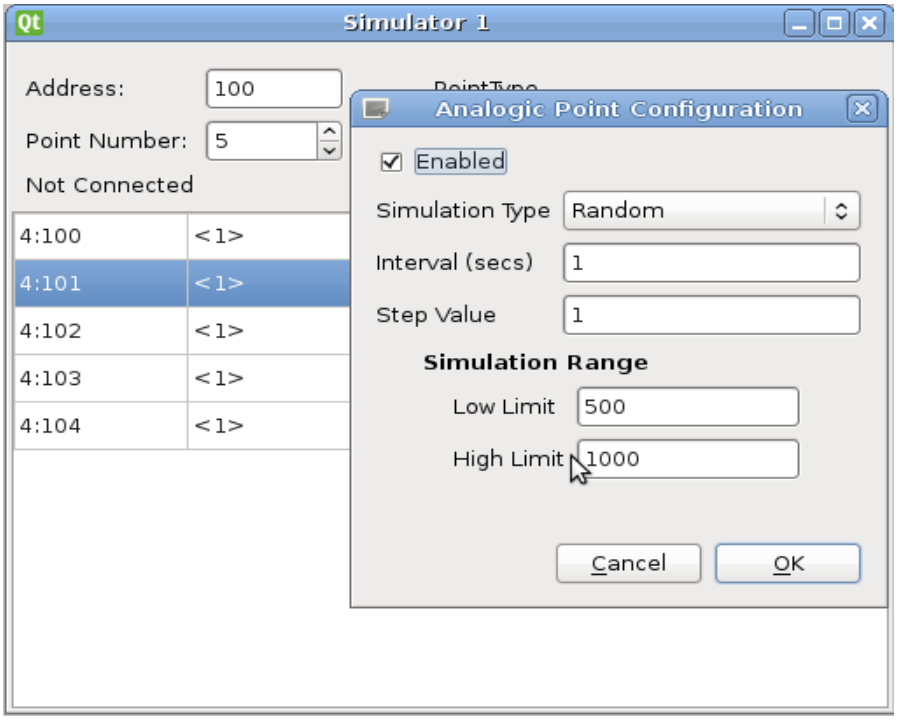

Figura 7 Interfaz Visual "Modificar variable analógica"

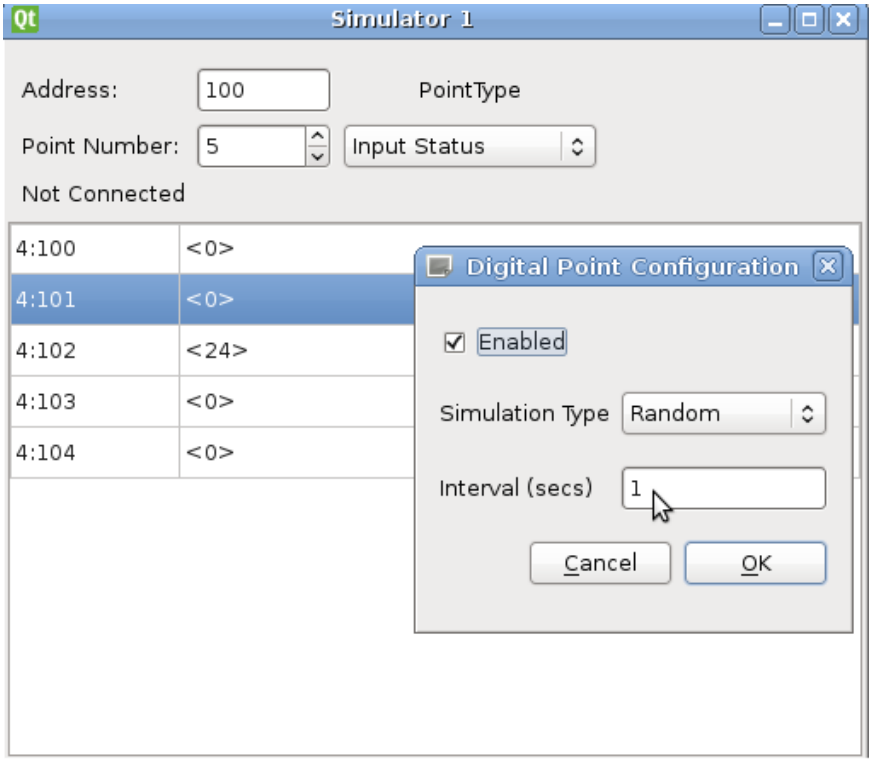

Figura 8 Interfaz Visual "Modificar variable digital"

Tarea de Ingeniería	
No. de la tarea: 3	**No. de la HU:** 1
Nombre de la tarea: Eliminar variable analógica y digital.	
Tipo de tarea: Implementación.	**Puntos estimados:** 0.16
Fecha inicio: 16/11/2012	**Fecha fin:** 16/11/2012
Programador responsable: Darlin Mercedes Blanco García.	
Descripción: Funcionalidad encargada de la eliminación de las variables analógicas y digitales.	

Tabla 17 Tarea de ingeniería "Eliminar variable analógica y digital"

La eliminación de variables se realiza del mismo modo en que se adicionan, pero decrementando la cantidad de variables que se visualizan en el sumario de puntos.

Iteración 2

Esta iteración tuvo como objetivo darle cumplimiento a la HU2 que responde al proceso de inicialización de la simulación. En dicha iteración es donde se realizan todas las funcionalidades necesarias para realizar la conexión a la capa de comunicaciones del SCADA.

Historias de usuario	Tiempo de implementación (semanas)	
	Estimación	Real
Iniciar proceso de simulación.	3	3
Total	3	3

Tabla 18 Tiempo de estimación para la segunda iteración

Tarea de Ingeniería	
No. de la tarea: 3	**No. de la HU:** 2
Nombre de la tarea: Iniciar proceso de simulación.	
Tipo de tarea: Implementación.	**Puntos estimados:** 0.16
Fecha inicio: 16/11/2012	**Fecha fin:** 16/11/2012
Programador responsable: Darlin Mercedes Blanco García.	
Descripción: Funcionalidad encargada de iniciar el proceso de simulación.	

Tabla 19 Tarea de ingeniería "Iniciar proceso de simulación"

Esta funcionalidad se inicia cuando se habilitan las variables que se muestran en el sumario y luego se realiza la conexión mediante el menú “Connection” de la barra de menús. También se puede inicializar la simulación al habilitar algunas de las variables luego de establecida la conexión, en este momento comienzan a generarse los valores automáticamente. Para lograr este funcionamiento se utiliza la biblioteca SimulatorData para iniciar o detener la conexión con la capa de comunicación del SCADA.

Iteración 3

El objetivo de esta iteración es darle cumplimiento a la HU3 que describe el proceso de visualización de las variables. Se desarrollaron todos los componentes visuales que permiten mostrar de forma dinámica los distintos valores que van tomando las variables cada un intervalo de tiempo, así como las interfaces visuales donde se configuran los parámetros necesarios para la simulación.

Historias de usuario	Tiempo de implementación (semanas)	
	Estimación	Real
Visualizar variables configuradas en tiempo real.	3	3
Total	3	3

Tabla 20 Tiempo de estimación para la tercera iteración

Tarea de Ingeniería	
No. de la tarea: 3	**No. de la HU:** 3
Nombre de la tarea: : Visualizar variables configuradas en tiempo real	
Tipo de tarea: Implementación.	**Puntos estimados:** 0.16
Fecha inicio: 16/11/2012	**Fecha fin:** 16/11/2012
Programador responsable: Darlin Mercedes Blanco García.	
Descripción: Funcionalidad encargada de iniciar el proceso de simulación.	
Fecha inicio: 16/11/2012	**Fecha fin:** 16/11/2012
Programador responsable: Darlin Mercedes Blanco García.	
Descripción: Funcionalidad encargada de la visualización de las variables ya configuradas, mostrando los valores generados por el sistema.	

Tabla 21 Tarea de ingeniería "Visualizar variables configuradas en tiempo real"

El sumario de puntos se muestra utilizando una tabla y el funcionamiento del mismo está controlado por un modelo que se crea utilizando el framework Qt y se le asocia a dicha tabla. Este modelo contiene el listado de todos los puntos o variables que se han creado. Mediante los componentes explicados anteriormente se realiza la funcionalidad de visualizar los valores que se van simulando. Este proceso se realiza automáticamente según van cambiando los valores de los puntos que se encuentran creados y almacenados en el modelo, debido a que el framework Qt provee la funcionalidad de actualizar dinámicamente los datos que se muestran en la tabla a través de los valores que contiene el modelo. Ver figura 11.

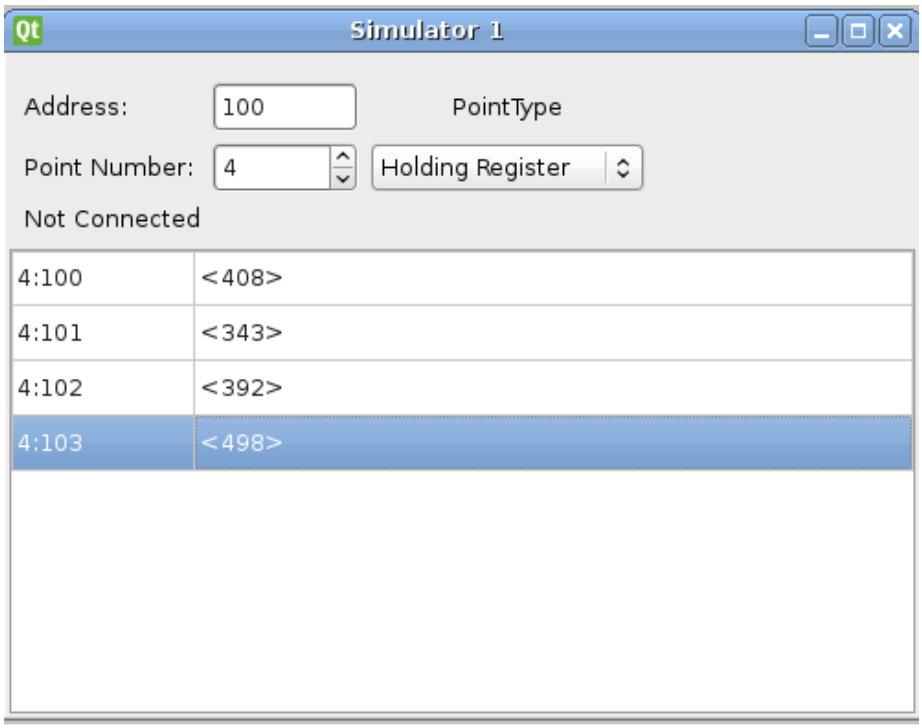

Figura 9 Interfaz Visual "Visualizar variables configuradas en tiempo real"

3.2 Pruebas

Como define la metodología XP, uno de sus pilares es el proceso de pruebas. XP anima a probar tanto como sea posible. Esto permitió aumentar la calidad del sistema reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permitió aumentar la seguridad al evitar efectos colaterales no deseados realizando modificaciones y refactorizaciones. Esta metodología divide las pruebas del sistema en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores y pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de cada iteración se consiguió la funcionalidad requerida por el cliente [7].

3.2.1 Pruebas de aceptación.

Las pruebas de aceptación tienen mayor importancia que las pruebas unitarias, dado que significan la satisfacción del cliente con el producto desarrollado, el final de una iteración y el comienzo de la siguiente, por consecuente el cliente es la persona indicada para diseñar las pruebas a ejecutar [7]. La ejecución de las pruebas previamente diseñadas, permitió la evaluación de las funcionalidades de la aplicación desarrollada antes de implantarlo en su entorno real de explotación. Los resultados obtenidos se muestran a continuación:

Caso de Prueba de Aceptación	
Número: 1	**Historia de usuario:** 1
Nombre: Crear Variable analógica o digital.	
Descripción: Se comprueba la creación de las variables a través de la interfaz visual, tanto de tipo analógico como digital. En este caso se verifica que en el sumario de puntos se creen la misma cantidad de variables que las especificadas por el usuario.	
Condiciones de ejecución: Se debe seleccionar el tipo de simulación que se va a realizar.	
Entradas/Pasos de ejecución: Luego de creada la interfaz correspondiente a un proyecto, se escoge en la parte superior de la misma la cantidad de variables que se deseen adicionar. Se agregan al sumario de puntos tantas filas como variables seleccionadas, donde cada fila corresponde a una variable del listado que se simulará posteriormente.	
Resultado esperado: Creación de las variables en correspondencia con el número de puntos especificados.	
Evaluación de la prueba: Satisfactorio	

Tabla 22 Caso de Prueba "Crear variable analógica o digital"

Caso de Prueba de Aceptación	
Número: 2	**Historia de usuario:** 1
Nombre: Modificar variable analógica o digital.	
Descripción: Se comprueba que se realicen los cambios efectuados en el punto seleccionado, y en caso de que se esté simulando, que los valores generados se adapten a la nueva configuración.	
Condiciones de ejecución: Existir una variable seleccionada en el sumario de puntos.	
Entradas/Pasos de ejecución: Se agregan nuevas variables y se inicia la conexión con la capa de comunicaciones. Se realiza doble click sobre el punto seleccionado para mostrar la interfaz donde s	

modifican los parámetros del mismo. Se cambian los valores correspondientes intervalo que se simulará y se habilita dicho punto para que comience su simulación. Se aceptan los cambios realizados y se observa que la simulación genera los valores d acuerdo con la nueva configuración.
Resultado esperado: Modificación de las variables según los parámetros especificados, mostrándose así en el sumario de puntos.
Evaluación de la prueba: Satisfactorio

Tabla 23 Caso de prueba "Modificar variable analógica o digital"

Caso de Prueba de Aceptación	
Número: 3	**Historia de usuario:** 1
Nombre: Eliminar variable analógica o digital.	
Descripción: Se verifica que al disminuir el número de puntos existentes se elimine esa misma cantidad del sumario de puntos.	
Condiciones de ejecución: Existir al menos una variable en el sumario de puntos.	
Entradas/Pasos de ejecución: Luego de creada la interfaz correspondiente a u proyecto se disminuye la cantidad de puntos que fueron agregados. Se observa la disminución de las variables en el sumario de puntos teniendo en cuen la nueva cantidad especificada.	
Resultado esperado: Eliminación de las variables que se muestran en el sumario de puntos.	
Evaluación de la prueba: Satisfactorio	

Tabla 24 Caso de prueba "Eliminar variable analógica o digital"

Caso de Prueba de Aceptación	
Número: 4	**Historia de usuario:** 2
Nombre: Iniciar proceso de simulación.	
Descripción: Se verifica que en el sumario de puntos se realice la simulación de las variables que se encuentran habilitadas luego de realizar la conexión con la capa de Comunicaciones del SCADA. Comprobar que luego de la conexión las variables que se habiliten inicien su simulación automáticamente.	

Condiciones de ejecución: Existir variables habilitadas en el sumario de puntos antes de la conexión. Luego de la conexión debe existir al menos una variable en el sumario.
Entradas/Pasos de ejecución: Se agregan variables al sumario, se habilitan algunas de ellas y se configuran los parámetros según el tipo de simulación a realizar. Se selecciona el botón de conectar de la barra de herramientas para comenzar el proceso de simulación. Seguidamente se observa que las variables habilitadas inician su simulación. Se habilitan otras de las variables que no se encuentran simulándose en el sumario de puntos. Se inicia la simulación de las variables que fueron habilitadas.
Resultado esperado: Simulación de las variables que se encuentran habilitadas en el sumario de puntos.
Evaluación de la prueba: Satisfactorio

Tabla 25 Caso de prueba "Iniciar proceso de simulación"

Caso de Prueba de Aceptación	
Número: 5	**Historia de usuario:**3
Nombre: Visualizar variables configuradas en tiempo real.	
Descripción: Luego de establecida la conexión se comprueba la visualización de las variables que han sido habilitadas en el sumario y se verifica la variación de sus valores de acuerdo al tipo de configuración establecida.	
Condiciones de ejecución: Deben existir variables habilitadas en el sumario de puntos y haberse realizado la conexión con la capa de Comunicaciones del SCADA.	
Entradas/Pasos de ejecución: A medida que se van simulando los valores para cada una de las variables habilitadas, se cambia el valor que se encuentra almacenado en el modelo que controla la visualización de los datos en el sumario. Se observa el cambio en los valores de los puntos que se están simulando.	
Resultado esperado: Mostrar los valores simulados en el sumario de puntos de acuerdo a la configuración establecida para cada variable.	

Evaluación de la prueba: Satisfactorio

Tabla 26 Caso de prueba "Visualizar variables configuradas en tiempo real"

Se realizaron las pruebas de aceptación que demostraron un buen funcionamiento del sistema, logrando simular gran cantidad de variables al mismo tiempo y en distintos proyectos. Se comprobó que luego de establecida una conexión con la capa de comunicaciones del SCADA se pueden agregar otras variables al proceso de simulación. Luego de comprobadas cada una de las funcionalidades se procedió a corroborar que el módulo HMI pudiera utilizar las simulaciones creadas en el sistema. Para ello se creó un nuevo proyecto en la aplicación, se configuraron varias variables con las configuraciones necesitadas y se inició el proceso de simulación al realizar la conexión con la capa de comunicaciones. Realizadas estas acciones se pudo observar la actualización de los valores mostrados en el sumario de puntos del entorno de trabajo del HMI en tiempo real, demostrando así que la herramienta desarrollada da cumplimiento a la problemática planteada en este trabajo.

3.3 Conclusiones parciales

En este capítulo se ha realizado la descripción de la mayor parte de los elementos implementados, se han seleccionado las funcionalidades más significativas y se ha descrito detalladamente las principales clases que intervienen en su desarrollo. Las pruebas y validaciones realizadas permitieron evaluar la calidad y fiabilidad de la herramienta propuesta, demostrando la factibilidad de su uso en el desarrollo de los algoritmos que serán ejecutados por las tareas en el Módulo de HMI.

CONCLUSIONES GENERALES

Una vez finalizado el trabajo de diploma se pudo llegar a las siguientes conclusiones:

- El análisis de las estructuras de los módulos del GALBA, especialmente el HMI, permitió el desarrollo de una herramienta para simular el comportamiento real de los procesos de supervisión y control, específicamente para los puntos analógicos y digitales.

- El sistema desarrollado permite que varios ordenadores de un entorno de trabajo puedan utilizar una misma configuración compartida por uno de ellos, utilizando solamente el módulo de comunicaciones del GALBA.

- Las pruebas y validaciones realizadas permitieron evaluar la calidad y fiabilidad de la herramienta propuesta, demostrando la factibilidad de su uso en el desarrollo de las funcionalidades que serán ejecutadas posteriormente en el Módulo de HMI.

RECOMENDACIONES

En versiones posteriores de la herramienta desarrollada pueden aplicarse mejoras considerables para su funcionamiento en conjunto con el SCADA, que no fueron desarrolladas en su totalidad en este trabajo. A continuación se muestran algunas recomendaciones que pueden tomarse en cuenta para el desarrollo de nuevas funcionalidades:

- Utilizar técnicas de simulación más avanzadas con el objetivo de lograr que los procesos simulados tengan mayor similitud con el funcionamiento del Módulo de HMI presente en el SCADA GALBA.

- Adicionar nuevos datos al proceso de simulación, como alarmas, estados y otros comportamientos que sean necesitados por el módulo HMI del proyecto GALBA.

- Agregar la funcionalidad donde el sistema a medida que genere datos permita visualizar procesos donde se utilicen los mismos.

BIBLIOGRAFÍA

Referencias Bibliográficas

1. Dagoberto Montero, David B. Barrantes, Jorge M. Quiróz. Scribd. [En línea] 2007. [Citado el: 1 de octubre de 2010.] http://www.scribd.com/doc/13473499/Introduccion-a-Los-Sistemas-SCADA.

2. Autómatas Industriales. [En línea] 2000. [Citado el: 3 de octubre de 2010.] http://www.automatas.org/redes/scadas.htm.

3. UCLA. [En línea] Universidad Centroccidental Lisandro Alvarado, 1962. [Citado el: 2 de octubre de 2010.] http://www.ucla.edu.ve/dac/Departamentos/coordinaciones/informaticai/documentos/PROCESAMIENTO%20DE%20DATOS.htm.

4. UAG. [En línea] Universidad Autóctona de Guadalajara, 2007. [Citado el: 2 de noviembre de 2010.] http://genesis.uag.mx/edmedia/material/comuelectro/uni1_2_4.cfm.

5. Escolme Virtual. [En línea] 2010. [Citado el: 25 de noviembre de 2010.] http://escolmevirtual.com/portal/index2.php?option=com_content&do_pdf=1&id=39.

6. Rattner, CU Eduardo. Centro de Capitanes de Ultramar y Oficiales de la Marina Mercante. [En línea] Escuela Nacional de Náutica. [Citado el: 27 de noviembre de 2010.] http://www.capitanes.org.ar/fundacion/nota2-simuladorses.htm.

7. Venkat Pothamsetty, Matthew Franz. Source Forge. [En línea] [Citado el: 3 de diciembre de 2010.] http://scadahoneynet.sourceforge.net.

8. Scribd. [En línea] UNIVERSIDAD DE CHILE - Facultad de Ciencias Físicas y Matemáticas. [Citado el: 03 de diciembre de 2010.] http://www.scribd.com/doc/37508751/Manual-Modsim.

9. GNU Operating System. [En línea] 1996. [Citado el: 15 de diciembre de 2010.] http://www.gnu.org/home.es.html.

10. Varga, José Luís García Sarasa y Gloria Pérez de la. E-Prints Complutense. [En línea] UNIVERSIDAD COMPLUTENSE DE MADRID, 2006. [Citado el: 16 de diciembre de 2010.] http://eprints.ucm.es/9064/1/TC2007-32.pdf.

11. Stroustrup's, Bjarne. The C++ Programming Language. 1997. ISBN 2-7440-0609-2.

12. Mis Respuestas. [En línea] 2055. [Citado el: 17 de diciembre de 2010.] http://www.misrespuestas.com/que-es-una-metodologia.html.

13. Buenas Tareas. [En línea] [Citado el: 20 de diciembre de 2010.] http://www.buenastareas.com/ensayos/Metodolog%C3%ADa-Rup/473434.html.

14. Roberth G. Figueroa, Camilo J. Solís y Armando A. Cabrera3. Entorno Virtual de Aprendizaje. [En línea] [Citado el: 19 de diciembre de 2010.] http://eva.uci.cu/mod/resource/view.php?id=37186&subdir=/Metodologias.

15. ITESCAM. [En línea] Instituto Tecnológico Superior de Calkini en el estado de Campeche. [Citado el: 10 de mayo de 2011.] https://www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r57039.DOC.