Facultad 5.

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

Título: Mecanismo de configuración en caliente para la interfaz hombre máquina del sistema SCADA UX.

Autor: Henry Marcelo Cabrera Robles.

Tutor: Ing. Yunier Alexander Pimienta Fernández.

Co-tutor: Ing. Yosell Luis Sehara Driggs.

Ciudad de la Habana, Julio de 2013.

"Año 55 de la revolución"

Declaración de autoría

Por este medio declaro ser autor de este trabajo y reconozco a la Universidad de las Ciencias Informáticas (UCI) los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ___ días del mes de _____ del año _____.

_____	_____
Firma del Autor	Firma del Tutor
Henry Marcelo Cabrera Robles	Ing. Yunier A. Pimienta Fernández

Agradecimientos

Quisiera agradecer a todas las personas que de una forma u otra han contribuido a mi formación personal y profesional, en muchos casos, las palabras no son suficientes para expresar las dimensiones de mi agradecimiento.

A mi familia

Por estar siempre a mi lado y brindarme el apoyo necesario en cada momento. Soy dichoso de tenerlos a todos ustedes. Para ustedes todo mi agradecimiento por tanto amor y dedicación. Gracias

Para Tahumara, que siempre me soportó, incluso en los momentos más difíciles, gracias de todo corazón.

A mis tutores

Por haber confiado en mí para la realización de este trabajo y guiarme en el desarrollo del mismo. Gracias

A mis compañeros

Ustedes han sido durante estos cinco años mi familia más cercana en esta maravillosa escuela, gran parte de lo que soy se lo debo a ustedes.

Gracias

Dedicatoria

A mi mamá, a mi papá, a mi padrastro y a mi hermana

La familia más cercana que tengo, que reconocieron mis triunfos, que lloraron conmigo en los fracasos y me cuidaron en todo momento. A los que cambiarían todo por ofrecerme un minuto de felicidad.

Resumen

El desarrollo de aplicaciones de software en Cuba ha sido de los sectores que más se ha desarrollado en los últimos años, la Universidad de las Ciencias Informáticas (UCI) es uno de sus pilares fundamentales representando un ejemplo en la investigación de tecnologías novedosas y la creación de soluciones informáticas que permite el avance de nuestro país. En la UCI existe un centro de desarrollo de software en el campo de la automatización de procesos industriales denominado Centro de Informática Industrial (CEDIN), esta entidad es una de las más importantes con las que cuenta hoy la UCI, debido a que es de las que más divisa le aporta a la Universidad y al país.

La interfaz hombre-máquina (HMI), es la encargada de presentar la información en forma de gráficos representativos. Dicho módulo debe posibilitar la configuración y actualización del sistema, este reajuste debe realizarse sin reiniciar ninguno de los servidores, dicho proceso se conoce como configuración en caliente. La presente investigación tiene como objetivo desarrollar un mecanismo que permita la configuración en caliente de este módulo en el SCADA UX, sin necesidad de reiniciar o detener el sistema, posibilitando el refrescamiento de la configuración en las estaciones de trabajo donde se está utilizando el HMI, para lo cual es usado como lenguaje de programación C++ y como marco de trabajo (framework) la biblioteca gráfica QT.

PALABRAS CLAVE

SCADA, HMI, configuración en caliente

Índice

Índice de figuras

Índice de tablas

Introducción

En el amplio campo de la informática tiene lugar la automatización de procesos industriales, que no es más que el uso de sistemas o elementos computarizados y electromecánicos para controlar maquinarias y/o procesos industriales sustituyendo a operadores humanos. La automatización y supervisión de procesos industriales nace de la necesidad entre otras, de recolectar, almacenar y visualizar la información, además de las limitaciones de la visualización de los sistemas de adquisición y control; de manera que surgen sistemas que permiten supervisar y controlar variables de proceso a distancia, denominados SCADA.

Un SCADA (acrónimo de Supervisory Control And Data Adquisition) es una aplicación de software, especialmente diseñada para funcionar sobre ordenadores en el control de la producción, proporcionando comunicación con los dispositivos de campo (controladores autónomos, autómatas programables, entre otros). Además, envía la información generada en el proceso productivo a diversos usuarios, tanto del mismo nivel como hacia otros supervisores dentro de la empresa, es decir, que permite la participación de otras áreas como por ejemplo: control de calidad, supervisión, mantenimiento, entre otros. Dentro de las características de estos sistemas se encuentra su fácil adaptación a múltiples entornos; así como la posibilidad de integración con otros sistemas, ya sean gerenciales, de gestión u otros.

En sus orígenes, los sistemas SCADA eran controlados físicamente por empleados con distintos grados de acceso según la criticidad del sistema (lo cual se sigue manteniendo por cuestiones de seguridad y privilegios). Actualmente, los sistemas SCADA se han ido introduciendo poco a poco en nuevas tendencias tecnológicas, entre ellas la más famosa y de mayor auge, Internet. De esta forma es posible controlar los sistemas remotamente aprovechando las comunicaciones TCP/IP, consiguiendo por tanto, reducir costos y mejorando funcionalidades y eficiencia en los diferentes procesos de cada sistema en concreto.

Debido a las funciones que desempeñan dichos sistemas, algunos de éstos deben operar 24 horas al día, 7 días a la semana ofreciendo alta disponibilidad, este es el caso de procesos como la gestión de una central nuclear o los de una central eléctrica, entre otros muchos.

La naciente industria cubana del software cuenta con numerosos retos económicos y sociales, para acelerar la informatización de la sociedad cubana. La Universidad de las Ciencias Informáticas (UCI) forma parte de este desarrollo informático, siendo uno de los pilares del país en el desarrollo de software, su misión es formar profesionales comprometidos con la Revolución, partiendo de un modelo pedagógico flexible, que vincula dinámicamente el estudio con la producción y la investigación. El Centro de Informática Industrial (CEDIN) de la UCI es uno de los más importantes dentro de la universidad, ya que desarrolla productos que son una gran fuente de ingresos de

divisas al país. Tiene la misión de desarrollar productos y servicios informáticos de automatización industrial y computación gráfica, con un alto valor agregado y que cumplan las necesidades y expectativas de los clientes, potenciando la formación especializada y la investigación.

La universidad promueve la puesta en práctica de metodologías de trabajo reconocidas internacionalmente y la creación de soluciones informáticas en la producción de software, donde el CEDIN toma participación y protagonismo. Dicho centro tiene varias líneas de productos, siendo una de ellas el desarrollo de sistemas SCADA. Los integrantes de este proyecto tienen la responsabilidad de crear un sistema de este tipo, bajo paradigmas actuales y plataformas de software libre.

Actualmente el CEDIN no cuenta con un producto SCADA 100% propio. En entrevistas con los especialistas del centro se pudo constatar que se desarrollan varios sistemas de este tipo, entre los más conocidos están, el SCADA Guardián del ALBA (GALBA) el cual es un desarrollo en conjunto ALBET -PDVSA sujeto a una contratación que solo puede ser desplegado en Venezuela y Cuba; y el SCADA Agua Santiago que puede ser desplegado en el país, pero no puede ser comercializado con otras naciones porque se utilizaron algunos requisitos y módulos del SCADA GALBA.

Por todo lo anteriormente expuesto se evidencia la necesidad del desarrollo de un nuevo sistema SCADA. Para esto debe utilizarse el conocimiento y habilidades adquiridas en el desarrollo de los dos productos anteriores, con nuevas tecnologías, con una arquitectura abierta capaz de crecer o adaptarse según las necesidades de las empresas que lo necesite.

Debido a su magnitud y dificultad de desarrollo, los sistemas SCADA generalmente se desarrollan en módulos. Entre los módulos que componen un SCADA se encuentra el de interfaz hombre-máquina ó HMI (Human Machine Interface), que es el encargado de presentar la información en forma de gráficos representativos. Dicho módulo debe posibilitar, a los especialistas en mantenimiento, la configuración y actualización de la información del sistema. Este reajuste debe realizarse sin verse obligados a reiniciar los clientes HMI y ninguno de los servidores. El sistema SCADA UX que se encuentra actualmente en desarrollo, no cuenta con las posibilidades descritas anteriormente, lo cual provoca que:

1. Sea necesario detener el control de la aplicación, por parte del SCADA, cuando se necesite actualizar cambios en la configuración. Esto trae como consecuencia que se pierda información durante ese tiempo.
2. Se requiera reiniciar todos los clientes HMI para que estos actualicen los cambios realizados. Esto trae consigo que se pierda la visualización de los datos obtenidos por el SCADA.
3. Se impida trabajar sobre una configuración en dos clientes HMI simultáneamente. Esto trae como resultado retrasos en la conformación de las configuraciones, ya que es una sola persona la que tiene que crear toda la distribución de trabajo que el SCADA va a controlar.

Es por ello que se formuló el siguiente **problema científico:** ¿Cómo lograr la actualización de la configuración del módulo HMI sin necesidad de reiniciar o detener los servicios del SCADA-UX?

El **objeto de estudio** se centra en: la configuración de sistemas SCADA.

El **objetivo** de este trabajo consiste en: desarrollar un mecanismo para la configuración en caliente del módulo HMI del sistema SCADA UX.

Delimitando como **campo de acción**: la configuración en caliente de módulos HMI.

Para dar cumplimiento al objetivo se plantearon las siguientes **tareas investigativas**:

1. Actualización del estado del arte sobre el desarrollo de mecanismos para la configuración de módulos HMI.
2. Caracterización de las herramientas y tecnologías actuales para el desarrollo de mecanismos para la configuración en caliente del módulo HMI.
3. Diseño de las funcionalidades relacionadas con el mecanismo de configuración en caliente del módulo HMI del sistema SCADA UX.
4. Implementación y prueba del mecanismo de configuración en caliente.

Después de obtener toda la información relacionada con el tema del trabajo se plantea como **posible resultado:** Mecanismo que permitirá la configuración en caliente para la interfaz hombre máquina del sistema SCADA UX.

Desde el punto de vista metodológico se emplearon los siguientes métodos científicos:

Teóricos:

Analítico Sintético: Admite el análisis de todo el contenido de características específicas del mecanismo de configuración y a su vez llegar a conceptos y generalizaciones, de manera que permita sintetizar todo lo obtenido como un todo.

Análisis histórico-lógico: Es utilizado para analizar la evolución histórica de soluciones similares, las tendencias más recientes de los mecanismos de configuración y basándose en esos datos, complementar las características necesarias y deseables para la solución que se propone.

Modelación: Se emplea para representar gráficamente la solución que se propone.

Generalización: Permite sistematizar en cada capítulo de la investigación las pautas más significativas de la misma, así como llegar a conclusiones más objetivas y explícitas para cada caso.

Revisión Documental: Permite la consulta de la bibliografía confiable y más completa referente al tema de la investigación.

El presente Trabajo de Diploma está estructurado de la siguiente manera: Introducción, cuatro capítulos de contenido, Conclusiones, Recomendaciones, Referencias Bibliográficas, Bibliografía, Anexos, y Glosario de Términos.

Capítulo 1: “Fundamentación Teórica”, en este capítulo se analiza con mayor profundidad el objeto de estudio. Se explica además la metodología y algunas herramientas de desarrollo; así como las tecnologías actuales a considerar para el desarrollo del mecanismo de configuración.

Capítulo 2: “Análisis y diseño”, en esta sección se describen los casos de uso, y la descripción de los mismos, que permiten el desarrollo del mecanismo de configuración en caliente. También se muestra el diagrama de clases de diseño, y las descripciones de las clases que forman parte de este diagrama de clases del diseño.

Capítulo 3: “Solución propuesta”, en este capítulo se muestra el diagrama de despliegue, de componentes, pruebas realizadas al mecanismo de configuración en caliente y se obtiene además una versión del producto final.

Capítulo 1 “Fundamentación Teórica”

En este capítulo se abordarán conceptos esenciales relacionados con los sistemas SCADA tales como sus usos y funcionalidades, haciendo especial énfasis en los elementos afines al módulo de HMI. Además se describirán las herramientas y tecnologías a utilizar en el desarrollo del mecanismo tratado en este trabajo.

1.1. Introducción a los sistemas SCADA

Un sistema distribuido no es más que un conjunto de elementos, tanto de procesamiento como de almacenamiento que están conectados a través de una red y que para un usuario del sistema se comporta como un único computador. Existen cinco características que son responsables de la utilidad de los sistemas distribuidos: compartir recursos, apertura, concurrencia, tolerancia a fallas y transparencia. Se hace notar que estas características no son consecuencia automática de la distribución; el software debe ser cuidadosamente diseñado para asegurar que ellas sean conseguidas.

Un ejemplo de aplicación con un sistema distribuido son los sistemas SCADA. Estos sistemas son aplicaciones de software para la adquisición, supervisión y control de datos mediante la comunicación con los dispositivos de campo, con la finalidad de controlar procesos automatizados desde la pantalla del ordenador. Gracias a estos sistemas se pueden localizar los errores de forma rápida, minimizando así los períodos de paro en las instalaciones, esto repercute en la reducción de costes de mantenimiento.(1)

Los primeros SCADA eran simplemente sistemas de telemetría que proporcionaban reportes periódicos de las condiciones de campo vigilando las señales que representaban medidas y/o condiciones de estado en ubicaciones de campo remotas. Estos sistemas ofrecían capacidades muy simples de monitoreo y control, sin proveer funciones de aplicación alguna. La visión del operador en el proceso estaba basada en los contadores y las lámparas detrás de paneles llenos de indicadores. Mientras la tecnología se desarrollaba, los ordenadores asumieron el papel de manejar la recolección de datos, disponiendo comandos de control, y una nueva función de presentación de la información sobre una pantalla de CRT. Los ordenadores agregaron la capacidad de programar el sistema para realizar funciones de control más complejas. Los primeros sistemas automatizados SCADA fueron altamente modificados con programas de aplicación específicos para atender a requisitos de algún proyecto particular.

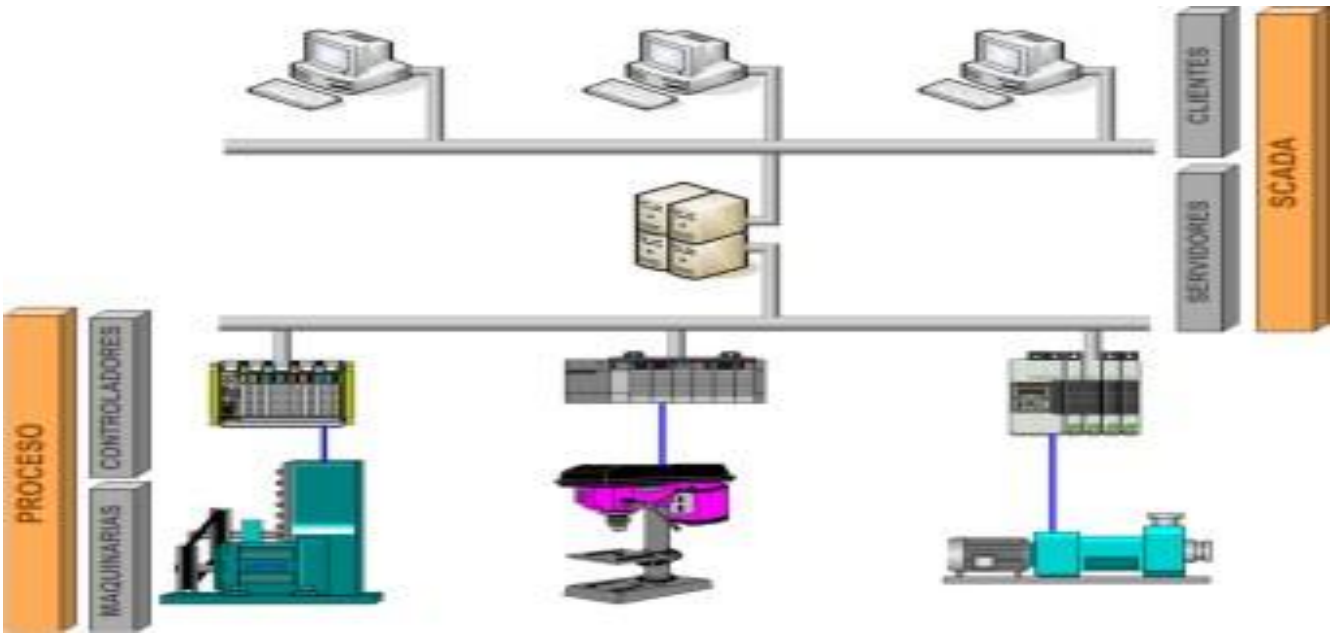

Figura 1: Sistema SCADA.

En la figura 1 se muestra la concepción general de un SCADA, para un mayor entendimiento, la figura se divide en dos, una parte que corresponde al proceso, aquí es donde se encuentran las maquinarias de las diferentes industrias y los controladores, que son los que están supervisando constantemente las variables de las maquinarias, estos controladores envían la información a los servidores, estos están en la segunda parte de la figura que corresponde al sistema de software como tal, los servidores son los encargados de entre otras cosas enviar la información a los clientes para que la presenten en forma de gráficos representativos.

La mayoría de los sistemas SCADA que son instalados hoy se están convirtiendo en una parte integral de la estructura de gerenciamiento de la información corporativa. Estos sistemas son vistos por la gerencia como un recurso importante de información. La mayoría de los vendedores principales de SCADA han reconocido esta tendencia, y están desarrollando rápidamente métodos eficientes para hacer disponibles los datos, mientras protegen la seguridad y funcionamiento del sistema SCADA. La arquitectura de los sistemas de hoy integra a menudo muchos ambientes de control diferentes, tales como tuberías de gas y aceite, en un solo centro de control.(1)

Las tareas de supervisión y control generalmente están más relacionadas con el software SCADA, en él, los operadores puede visualizar en la pantalla del computador de cada una de las estaciones remotas que conforman el sistema, los estados de ésta, las situaciones de alarma y tomar acciones físicas sobre algún equipo lejano, la comunicación se realiza mediante buses especiales o redes LAN. Todo esto se ejecuta normalmente en tiempo real, y están diseñados para brindar al operador de planta la posibilidad de supervisar y controlar dichos procesos. Estos sistemas actúan sobre los dispositivos instalados en la planta, como son los controladores,

autómatas, sensores, actuadores, registradores, etc. Además permiten controlar el proceso desde una estación remota, para ello el software brinda una interfaz gráfica que muestra el comportamiento del proceso en tiempo real. Generalmente se vincula el software al uso de una computadora o de un PLC, la acción de control es realizada por los controladores de campo, pero la comunicación del sistema con el operador es necesariamente vía computadora. Para alcanzar un nivel aceptable de tolerancia de fallas con estos sistemas, es común tener ordenadores SCADA redundantes operando en paralelo en el centro primario del control, y un sistema de reserva del mismo situado en un área geográficamente distante. Esta arquitectura proporciona la transferencia automática de la responsabilidad del control de cualquier ordenador que pueda llegar a ser inasequible por cualquier razón, a una computadora de reserva en línea, sin interrupción significativa de las operaciones.

Cuando hablamos de un sistema SCADA, no debemos centrarnos solo en las diferentes pantallas que nos muestran toda la información de forma sencilla e intuitiva. Detrás de todo esto está toda una amplia gama de elementos de supervisión y control, sistemas de comunicaciones y múltiples utilidades de software para hacer que el sistema funcione de forma segura, rápida y eficiente. A continuación se relacionan un conjunto de ventajas que proporcionan los sistemas SCADA:

- ✓ El actual desarrollo de los sistemas SCADA permite que se puedan utilizar sin ser un experto en la materia.
- ✓ Los autómatas son altamente configurables, por lo que permiten adaptarlos a las necesidades del momento y reconfigurarlos después en caso de ser necesario.
- ✓ Con la generación de alarmas y las herramientas de diagnóstico se puede localizar de forma rápida las fallas, esto permite reducir los paros productivos en las instalaciones y reduce de manera sustancial los costos de mantenimiento.
- ✓ Los protocolos de seguridad permiten la gestión segura y eficiente de los datos, garantizando la integridad, confidencialidad y disponibilidad.
- ✓ Monitorización: representa los datos con severas restricciones de tiempo, de forma tal que permita a los operadores de la planta conocer el estado de los procesos en todo momento.
- ✓ Además a través de este monitoreo de los diferentes procesos, haciendo uso extensivo de los avances tecnológicos, es posible identificar los problemas y encontrarles solución antes de un detenimiento no deseado y perjudicial de las actividades monitoreadas.
- ✓ Es posible operar remotamente con la mayoría de estos sistemas de control sobre los servicios o procesos productivos, la restauración tras una interrupción puede ser efectuada a distancia con mayor velocidad.

- ✓ El SCADA también ayuda a determinar la localidad de donde proviene la interrupción, la severidad de la misma y el número de personas afectadas, informaciones importantes para garantizar la asignación apropiada de los recursos necesarios para la recuperación. Sus reportes permiten analizar las causas de las diversas problemáticas.
- ✓ A través de la integración de una gran variedad de dispositivos electrónicos de control y red que facilitan el monitoreo y administración, los SCADA posibilitan la obtención de niveles de fiabilidad y robustez muy altos.
- ✓ Mediante el despliegue de un SCADA centralizado es posible reducir los costos de operación y mantenimiento debido a que se requiere menos personal para monitorear equipamientos de localidades remotas, resultando en un incremento de la efectividad del operador y menor número de viajes de mantenimiento.
- ✓ La habilidad de monitorear y controlar los procesos desde una única localidad, y al mismo tiempo recolectar y compartir en tiempo real los datos de los procesos, se ha convertido en una herramienta invalorable para el mejoramiento de las operaciones.
- ✓ Permiten brindar respuestas frente a las situaciones de emergencia de una forma mucho más rápida y efectiva porque los datos son analizados antes, durante y después de cada evento.
- ✓ Automatizaciones de muchas otras funciones específicas brindan eficiencias adicionales. El sistema pudiera incluso ser reiniciado remotamente en momentos que sea necesario por situaciones críticas o de alarma.

Estas ventajas mencionadas hacen que la utilización de sistemas SCADA sea una forma más eficiente, rápida y segura de controlar la producción de una industria.

En el mundo de los sistemas SCADA se encuentran una gran variedad de aplicaciones. Desde grandes sistemas privativos, muchos de los cuales se comercializan como soluciones completas acompañadas del hardware de supervisión y control, hasta no menos importantes soluciones de código abierto que generalmente, trabajan con hardware de un sinnúmero de fabricantes, lo que los hace en algunos casos más interoperables. Las plataformas más comunes sobre las que corren estos sistemas son Windows y GNU/Linux. En este mundo de las tecnologías, la automatización y el control, existen muchos sistemas SCADA importantes, a continuación se relacionan algunos de los más avanzados y reconocidos a nivel internacional.

1.2.1 PlantScape

Uno de los sistemas SCADA más utilizados actualmente a nivel mundial es PlantScape de Honeywell, este posee una herramienta de ingeniería del sistema, Control Builder, incorpora una librería con más de 50 bloques de control analógico predefinidos. Estos bloques han sido diseñados a partir de la experiencia de Honeywell en sistemas DCS. Mediante un proceso simple de selección y

copia ("drag-and-drop") el usuario puede fácilmente configurar nuevos lazos y ajustarlos. Mediante un mecanismo similar al anterior, el usuario puede construir módulos de control reusables y autodocumentados, agilizándose así tareas de ingeniería del sistema que tradicionalmente consumían mucho tiempo. La documentación se realiza utilizando el Control Builder, que la genera automáticamente. El lenguaje Hypertext Markup Language (HTML) le permite obtener una ayuda particularizada al momento de la configuración. Con sólo unos pequeños pasos, se llega al centro del problema. Los vídeos y los mapas de bits dan una imagen real e identifican las localizaciones de la planta. La conexión se puede realizar con dispositivos de hardware de terceros por medio del ControlNet y por conexión a una variedad de enlaces seriales de comunicación estándar. Esto permite una gran funcionalidad y versatilidad. PlantScape está disponible con aplicaciones normalmente utilizadas, que incluyen control de calderas, análisis de disparo de planta, análisis predictivo y muchas más. Se puede añadir el RMPCT, Robust Multivariable Predictive Control Technology (Tecnología robusta y multivariable de control predictivo), para aplicaciones que requieran algoritmos de control más complejos.(2)

Existe la posibilidad de añadir o modificar las funciones de control, sin incurrir en paradas, a medida que la instalación se va ampliando. Si se requiere, se puede enlazar con aplicaciones anteriores. Es posible controlar todo el sistema exteriormente. Los gráficos disponibles en una librería de funciones estándar ayudan, en términos de simplicidad y rapidez, en la ingeniería de cualquier sistema actualmente disponible. Los despliegues pueden ser animados y enlazados al proceso, para representar los eventos en tiempo real. Las adiciones o modificaciones a los gráficos se pueden realizar estando el sistema está en línea o no. Este sistema cuenta con más de 300 visualizaciones pre-grabadas, que comprenden la base para el monitoreo de un proceso con lo cual se disminuye el tiempo de implementación, entre estos se pueden mencionar:

1. Despliegues de puntos compuestos.
2. Despliegues del sumario de alarmas.
3. Despliegues del sumario de eventos.
4. Despliegues de tendencias.
5. Despliegues de reportes.
6. Despliegues de grupos de operadores.
7. Despliegues de diagnóstico del sistema.

Las funciones de alarma, tendencia, datos históricos y comunicación son estándares del sistema. Estos posibilitan una mayor rapidez que la combinación convencional PLC/SCADA. La clave está en

una fuerte integración, ya que una sola base de datos realiza todas las funciones. Dicha base de datos almacena la siguiente información:

- ✓ Configuración de la arquitectura del sistema y la relación entre los nodos.
- ✓ Configuración de la red local.
- ✓ Configuración del hardware de cada nodo del sistema.
- ✓ Configuración de los puntos de campo y puntos internos o de cálculo.
- ✓ Valores de cada uno de los parámetros correspondientes a cada punto de campo, interno o de cálculo.

Para la configuración de sistema, se utiliza la herramienta Quickbuilder, la información existente en la base de datos puede ser cargada en esta herramienta para modificarlo y verificar que esta como se desea. Entre las características fundamentales que provee esta herramienta se destacan:

1. Los cambios en la configuración pueden hacerse mientras el sistema está en línea.
2. Posee una interfaz intuitiva.
3. Selección de múltiples objetos.
4. Edición de múltiples puntos.
5. Informes normalizados.
6. Facilidad en el manejo de alarmas.
7. Fácil y rápida configuración de grupos y tendencias.(2)

1.2.2 ARGOS-SCADA

El proyecto Argos está siendo desarrollado para ser un SCADA con código abierto, asimismo, las herramientas que Argos proporciona actualmente sientan una base (con funcionalidades básicas) para implementar sistemas de supervisión en procesos automatizados.

La arquitectura clásica de estos sistemas permite inferir que los componentes de software que interactúan para formar un SCADA deben estar distribuidos en la totalidad de la red de supervisión, aunque pueden existir aplicaciones en las que todos los componentes de software se ejecuten dentro de la misma estación de trabajo.

Argos se ha diseñado con una arquitectura que permite adaptarse a los distintos esquemas de automatización moderna, en donde cada componente de software cuenta con estructuras de datos

de alto rendimiento que operan de manera distribuida ya sea en una plataforma de red o en una misma computadora.

Entre las principales herramientas que se proporcionan en el SCADA Argos, se encuentran los siguientes:

- ✓ Comunicación, estos procesos se encargan de establecer la comunicación con los equipos controladores de campo y pueden ejecutarse en uno o varios nodos.
- ✓ Escaneador, convierte los registros de todos los controladores en unidades de ingeniería, que posteriormente serán mostrados a los usuarios finales.
- ✓ Historiador, es un proceso configurado para almacenar información de manera permanente, principalmente usado para contar con gráficos de tendencias e históricos de alarmas y eventos.
- ✓ Los procesos Servidores y Clientes se encargan de la transferencia de información, ya sea entre nodos localizados remotamente, o que se estén ejecutando en el mismo servidor, para que pueda ser presentada al operador.
- ✓ Transmisor, su única función será escribir registros en el controlador.
- ✓ Por último, los procesos HMI serán los encargados de desplegar la información adquirida a los usuarios finales, mediante distintos recursos gráficos.(3)

Actualmente el software está disponible como un proyecto de código abierto, bajo licencia GPL, en el repositorio SourceForge.NET, su estado es ALPHA y en continuo desarrollo para generar los candidatos a lanzamiento (RC) y cuenta con los siguientes avances:

- ✓ Procesos de adquisición y administración de variables, eventos y alarmas.
- ✓ Procesos de envío y recepción de datos con los clientes HMI.
- ✓ Procesos para el almacenamiento en base de datos de las variables, eventos y alarmas.
- ✓ Capacidad para configuración a través de archivos en formato XML para todos los procesos anteriormente mencionados.
- ✓ Objetos gráficos diseñados para ser incorporados en aplicaciones de Ventanas usados para desplegar la información de las variables del proceso (como clientes HMI).

Para desarrollos futuros orientados a cubrir la mayoría de los requerimientos del sector industrial, en lo que a sistemas de control supervisor se refiere, se consideran los siguientes requerimientos:

- ✓ Desarrollo de manejadores para la adquisición de datos de la gran gama de dispositivos de instrumentación y control que actualmente se encuentran en el mercado e instalados en la mayoría de las plantas industriales.

- ✓ Simuladores para poder diseñar y desarrollar todo un sistema SCADA sin la necesidad de estar conectado a hardware industrial.
- ✓ Diseño y desarrollo de un entorno integrado para la configuración e implementación de todas las herramientas en un sistema SCADA.
- ✓ Diseño y desarrollo de un entorno que permita la creación de interfaces Web dinámicas compatibles con navegadores Web estándar.
- ✓ Procesos que administren la redundancia del sistema SCADA, asegurando de ésta manera la robustez del control supervisor sobre un proceso productivo.(3)

1.2.3 InTouch

InTouch es un SCADA de la rama Wonderware de la división de Administración de Operaciones de la compañía inglesa Invensys. Permite configurar alarmas y establecerles hasta 999 niveles de prioridad y hasta 8 niveles de jerarquía entre grupos de alarma con posibilidad de hasta 16 subgrupos para cada uno de ellos. Posibilita visualizar todas o un extracto de ellas de forma histórica y grabar en disco o imprimir en diferentes formatos personalizables. Las funciones de alarmas distribuidas incluyen reconocimiento global o selectivo, desplazamiento por la lista y visualización de alarmas procedentes de diferentes servidores en un único panel.(4)

Entre las características más significativas se destacan:

Gráficos orientados a objetos: Las aplicaciones son fáciles de editar y configurar, por lo que representan un menor tiempo de desarrollo. Se puede mover, redimensionar y animar objetos o grupos de ellos como si fueran imágenes estáticas. Dispone de todo tipo de herramientas de diseño: dibujos sencillos, alineación, trabajo en múltiples capas, espaciado, rotación, inversión, duplicación, copia, eliminación, etc.

SuitLink / OPC: Es un protocolo de comunicaciones elaborado por Wonderware de muy altas prestaciones para enlace de aplicaciones bajo TCP/IP o PROFIBUS, en el que se pueden configurar clientes OPC.

Gráficos de Tendencia Históricos y a Tiempo Real: Cada gráfico puede presentar hasta 16 plumas con referencias a variables y ficheros históricos independientes. Cada uno de los gráficos dispone, en tiempo de ejecución, de selección de variables, visualización del valor en la posición del cursor, ampliación, desplazamiento o centrado.

Programación: Posee un lenguaje de programación sencillo y extenso para la realización de cálculos en segundo plano, simulaciones, etc.

Lecturas y escrituras optimizadas: El uso de técnicas de excepción en lecturas/escrituras de variables enlazadas a segundas aplicaciones facilita la transferencia de datos de la forma más

rápida. Sólo se actualizan continuamente los puntos de comunicación de objetos visibles o los utilizados en alarmas, históricos o en programas de usuario.

Entre los beneficios de InTouch HMI/SCADA se pueden mencionar:

- ✓ Facilidad de uso que le permite a desarrolladores y operarios ser más productivos de manera simple y rápida.
- ✓ Gran integración de dispositivos y conectividad a prácticamente todos los dispositivos y sistemas.
- ✓ Sus capacidades de representación gráfica y la interacción con sus operaciones permiten entregar la información correcta a las personas correctas en el momento correcto.
- ✓ Migración de versiones de software sin interrupción, lo que significa que la inversión en sus aplicaciones HMI está protegida.

Además entre las capacidades se tienen:

- ✓ Gráficos de resolución independiente y símbolos inteligentes que visualmente dan vida a su instalación directamente en la pantalla de su computadora.
- ✓ Sofisticado sistema de scripting para extender y personalizar aplicaciones en función de sus necesidades específicas.
- ✓ Alarmas distribuidas en tiempo real con visualización histórica para su análisis.
- ✓ Modelamiento de tendencias históricas integradas y en tiempo real.
- ✓ Integración con controles Microsoft ActiveX y controles .NET.
- ✓ Librería extensible con más de 500 de objetos y gráficos prediseñados, "inteligentes" y personalizables.

De manera general, los sistemas Intouch y PlantScape son muy avanzados, pero tienen una gran limitante, son propietarios, no es posible tener acceso a su código fuente para ser estudiados por el personal del centro para tomar experiencia y verterla en nuestros desarrollos enfocados a la industria cubana. Hay que mencionar además sus altos precios de adquisición, en especial los grandes SCADA internacionales que en muchas ocasiones vienen acompañados del hardware lo que los hace aún más difíciles de adquirir para países sin un alto desarrollo económico como Cuba. El empleo de estos sistemas limita la soberanía tecnológica del país debido a que se deja el control de los procesos industriales claves en manos de un software desarrollado en muchos casos en países ideológicamente contrarios al proceso revolucionario cubano. En cambio el sistema Argos-SCADA es libre, por lo que es más factible su implantación en nuestro país.(4)

1.2. Niveles de un SCADA

Un sistema SCADA está formado por una amplia gama de elementos que colaboran para llevar a cabo las tareas de supervisión, control y adquisición de datos. De ahí que puedan subdividirse para su mejor comprensión en 4 niveles o partes esenciales:

1. **Nivel de instrumentación**: este nivel toma las variables físicas y las convierte en una señal equivalente que puede ser leída o interpretada por el hombre. El sistema SCADA maneja instrumentación de tipo electrónica donde la variable física se convierte en una señal eléctrica.
2. **Nivel de RTU**: las RTU (Remote Terminal Unit) o unidades terminales remotas, son dispositivos inteligentes con microprocesador que recogen, almacenan y procesan la información que viene de la instrumentación de campo y comandan elementos finales de control. Son programables, permitiendo alojar algoritmos de control y poseen un canal de comunicación para su interconexión.
3. **Nivel de comunicaciones**: es el encargado de tomar la información de las RTUs y transmitirla por el medio escogido hasta el centro de control. Dependiendo del costo, tamaño del sistema SCADA, distancias a las RTUs, disponibilidad del medio, la velocidad de transmisión y la confiabilidad requerida, se seleccionan los distintos soportes y medios de comunicación más apropiados.
4. **Nivel de Gestión o Centro de control**: está compuesto por un conjunto de computadoras, periféricos y software que realizan el procesamiento de las señales. Existe al menos una computadora con su hardware y software de base en la cual se mantiene ejecutando un software SCADA de cierta complejidad, que abarca diversas funciones. Usualmente existe también un equipo que sirve como interfaz de comunicaciones, cuya función es recibir la información de los diferentes canales de comunicación, procesarla y agruparla para enviarla a las restantes computadoras mediante una red.(5)

1.3. Módulos de un SCADA

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

- ✓ **Configuración**: Permite al usuario definir el entorno de trabajo de su SCADA adaptándolo a la aplicación particular que se desea desarrollar.

- ✓ **Interfaz gráfico del operador**: Proporciona al operador las funciones de control y supervisión de la planta. El proceso se representa mediante gráficos representativos almacenados

en el ordenador de proceso y generados desde el editor incorporado en el SCADA o importados desde otra aplicación durante la configuración del paquete.

- ✓ **Módulo de proceso**: Ejecuta las acciones de mando preprogramadas a partir de los valores actuales de variables leídas.

- ✓ **Gestión y archivos de datos**: Almacena y procesa ordenadamente los datos, de forma que otra aplicación o dispositivo pueda tener acceso a ellos.

- ✓ **Comunicaciones**: Proporciona la transferencia de información entre la planta y la arquitectura hardware que soporta el SCADA, y entre ésta y el resto de elementos informáticos de gestión.

Estos sistemas son capaces de recolectar información rápidamente de una gran variedad de fuentes, y presentarla a un operador de forma sencilla de interpretar. No constituye objeto de este trabajo abordar en profundidad esta temática, no obstante es importante presentar las principales funcionalidades de un sistema SCADA. Entre estas funcionalidades están:

Adquisición y almacenamiento de datos: Es la recolección, procesamiento y almacenamiento de la información recibida, en forma continua y confiable.

Supervisión de procesos: Es el monitoreo del funcionamiento del proceso, procesamiento estadístico de los datos y confección de reportes. Además se brindan notificaciones al operador del sistema sobre cambios detectados en la instalación.

Generación de Alarmas: Las alarmas se basan en la vigilancia de los parámetros de las variables del sistema. Son los sucesos no deseables, porque su aparición puede dar lugar a problemas de funcionamiento. Este tipo de sucesos requieren de la atención de un operario para su solución antes de que se llegue a una situación crítica que detenga el proceso. El resto de las situaciones normales, tales como puesta en marcha, paro, cambios de consignas de funcionamiento, consultas de datos, entre otras, serán los denominados eventos del sistema o sucesos. Los eventos no requieren de la atención del operador del sistema, registran de forma automática todo lo que ocurre en el sistema. También será posible guardar estos datos para su posterior consulta.

Control de procesos: Es el control de los procesos, actuando sobre los reguladores autónomos básicos como eventos y alarmas, o directamente sobre el proceso mediante las salidas conectadas.

Transmisión de datos: Es la transmisión de información entre dispositivos de campo y computadoras de control o entre dispositivos ubicados a un mismo nivel.

Presentación de la información: Es la representación gráfica de los datos mediante gráficos representativos. Esto es conocido como Interfaz del Operador o HMI (Human Machine Interface).

1.4. Módulos del SCADA UX

El SCADA UX es un sistema que integra las funcionalidades y beneficios que permiten una solución viable para aplicaciones de supervisión y control de procesos, utilizando para ello una arquitectura distribuida de módulos que permiten realizar el procesamiento en varios nodos independientes. Entre los módulos por los que está compuesto el SCADA UX están:

Drivers: En este módulo se encuentran los controladores que permiten al sistema operativo interactuar, controlar y comunicarse con un dispositivo en particular, posibilitando así la transmisión de datos entre redes de computadoras.

Procesamiento y Recolector: Es un framework en desarrollo de estructura modular que trabaja orientado a plugins o extensiones, para garantizar la ejecución en tiempo real y se encarga de la planificación de los procesos de lectura y escritura sobre los dispositivos dentro del sistema, actúa realizando diversas funciones como la conversión de datos para su mejor asimilación y posterior uso. También posee conexión con la capa Middleware.

BDH: Es el módulo encargado de manejar de forma clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante y se enviarán al HMI (Human Machine Interface), donde serán mostrados al usuario.

Middleware: Capa que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas.

HMI(Human Machine Interface): Es un módulo que permite desde cualquier PC cliente acceder a los datos que se encuentran en el servidor, este comprende todos los puntos de contacto entre el usuario y el equipo, de forma amena y evita tener que instalar software adicionales para que el sistema funcione.

Seguridad: Provee las funcionalidades necesarias para garantizar el trabajo autorizado por usuarios (autenticación y control de acceso).

Reportes: Emitir reportes o informes que consoliden la información adquirida para entregarla en un formato determinado, de tal manera que sea útil al personal que va dirigida. El funcionamiento del subsistema encargado de la generación de reportes puede dividirse en dos: Por un lado tenemos un diseñador de informes, por otro lado existe un motor para la generación de reportes que se encarga de extraer los datos de una o varias bases de datos.

Configuración: Es el encargado de almacenar, persistir y suministrar la información base para el funcionamiento de los demás módulos del SCADA. Cada módulo del SCADA, posee una biblioteca que permite establecer las comunicaciones con el servidor de configuración, a través del subsistema de comunicación, permitiendo crear, eliminar y modificar los recursos configurables del sistema.

1.5. Módulo de Configuración del SCADA UX

Generalmente los sistemas SCADA cuentan con funcionalidades para realizar la configuración, de modo que los operadores puedan definir el entorno de trabajo para adaptarlo a las necesidades de la aplicación. El sistema en que se enmarcará este trabajo, tiene un módulo de configuración dedicado a estos fines. Entre las prestaciones que se pueden encontrar en dicho módulo se tienen:

- ✓ Creación, modificación y organización de los recursos necesarios para la configuración de la aplicación definida.
- ✓ Gestionar los módulos del SCADA UX, permitiendo definir la ubicación de estos en los nodos físicos que se desee.
- ✓ Administración de usuarios que accederán al sistema, permitiendo la clasificación de estos según su importancia.
- ✓ Creación de grupos con privilegios que permiten o limitan el acceso y la acción de los usuarios sobre los recursos configurados en el sistema.
- ✓ Configuración de las comunicaciones, dando la facilidad para especificar los dispositivos que se utilizarán para recolectar los valores a monitorizar, además de los parámetros que estos necesitarán para su correcto funcionamiento.
- ✓ Creación de pantallas con imágenes y texto que simulen el proceso a controlar, brindando a los usuarios una mejor comprensión de la aplicación creada.

Éste módulo es el encargado de almacenar y suministrar la información base para el funcionamiento de los demás módulos del SCADA. Para lograr un funcionamiento eficiente del mismo es necesario garantizar la escalabilidad del módulo, de manera que soporte desde una pequeña configuración hasta una de un millón de puntos. El módulo debe ser lo más robusto posible para soportar cualquier problema de seguridad que atente contra la estabilidad del sistema. Además debe soportar la configuración en frío y en caliente.

Estas configuraciones son explicadas a continuación.

- ✓ **Configuración en frio**: Cada uno de los módulos del sistema se inicia y solicitan sus parámetros de arranque al Servidor de Configuración y posteriormente se sincronizan. Todos los datos y archivos temporales son limpiados, excepto los valores almacenados en el Servidor de Datos Históricos, que se mantienen de ejecuciones anteriores y el sistema de trazas. Al finalizar

esta acción el sistema se encuentra en la condición de Inicializado. Los módulos en la configuración en frío no se actualizan, para poder hacerlo tendrían que reiniciarse y volver a leer toda la configuración, lo cual es solventado con la configuración en caliente y operacional.

- ✓ **Configuración en caliente**: El sistema se encuentra en ejecución y se realizan modificaciones en la configuración. El sistema se actualiza con los nuevos datos de configuración sin afectar la ejecución de este.

1.6. Arquitectura del módulo de configuración del SCADA UX

La variante de asignar las mayores responsabilidades del módulo al servidor de configuración logra un mayor control del acceso al sistema de persistencia, ya que es el Servidor de Configuración es el único que puede acceder directamente a este, cuando un módulo necesita información de la configuración le hace el pedido al Servidor de Configuración mediante el agente, el servidor procesa el pedido y le da una respuesta (en caso de llevarla) a través del propio agente. Sin embargo, esta variante conlleva inconvenientes considerables. Al ser el servidor el único encargado de acceder al sistema de persistencia provoca que los datos que se manejen en una petición, ya sea de lectura o escritura, tendrán que viajar potencialmente dos veces por la red, en el caso de la obtención de una configuración la información va primero del sistema de persistencia al servidor y luego de este al agente, cuando se desee guardar una nueva configuración es al revés, la información va del agente al servidor y de este al sistema de persistencia. Esto a su vez conlleva que los datos que se envían del servidor a los agentes, o al revés, tienen que ser convertidos a un formato que ambos sean capaces de interpretar, provocando una considerable pérdida de tiempo ya que implica la conversión de los datos a este formato y luego la conversión de estos al formato usado internamente por el receptor de la información. Esta variante además atenta contra una rápida actualización del SCADA en la configuración en caliente, ya que desde el HMI se tendrá que hacer llegar al Servidor de Configuración la totalidad de la configuración para que a continuación el servidor se encargue de determinar las diferencias entre la nueva configuración y la que se encuentra almacenada, este proceso provoca que desde el HMI se envíe potencialmente una gran cantidad de información que ya se encuentra almacenada en el sistema de persistencia y que no es de utilidad alguna. Además, el proceso de informarle a cada agente de las actualizaciones puede ser lento siendo el propio servidor el encargado de enviarle esta información a cada uno de ellos.

La variante de delegar en los agentes la mayor parte de las responsabilidades del módulo tiene como principal desventaja que se hace más difícil garantizar un acceso controlado a los datos de la

configuración, lo cual puede atentar contra la integridad de estos. Esto está provocado porque son los agentes los encargados de acceder directamente al sistema de persistencia, ya sea para obtener la información que necesitan o para almacenar una nueva configuración. En este caso el Servidor de Configuración es el responsable de darles los permisos a los agentes para que ellos puedan acceder al sistema de persistencia, o sea, el servidor se encarga de determinar quién puede acceder en cada momento al sistema de persistencia. De esta forma se logra aumentar el rendimiento debido a que la información de configuración siempre se transporta directamente del sistema de persistencia al agente o viceversa, reduciéndose a la comunicación entre el servidor y los agentes al mínimo. Además se garantiza que se maneje estrictamente en cada momento la información que se necesita.

1.7. Módulo HMI del SCADA UX

La Interfaz Hombre-Máquina ó HMI (Human Machine Interface), es la que permite interactuar con el sistema presentando la información en forma de gráficos representativos de los elementos de la planta. Los sistemas HMI, se pueden pensar como una “ventana” de un proceso, la misma puede estar en dispositivos especiales como los paneles de un operador o en una computadora. Las señales de los procesos con conducidos al HMI por medio de dispositivos como tarjetas de entrada/salida en la computadora, PLC, RTU o DRIVE, todos estos dispositivos deben tener una comunicación que entienda el HMI.

Existen diferentes tipos de HMI, de los cuales se pueden reconocer claramente dos de ellos: Interfaces Gráficas de Usuario (GUI, sus siglas en inglés) e Interfaces de Usuario basadas en la Web. Existen varios tipos de interfaces que actualmente son utilizadas en menor escala, entre ellas están las basadas en líneas de comandos, las táctiles, basadas en gestos, las multi-pantallas y las basadas en texto. Los sistemas HMI facilitan mucho las tareas de diseño debido a que incorporan protocolos para comunicarse con los dispositivos de campo más conocidos, tienen herramientas para crear bases de datos dinámicas, permiten crear y animar pantallas de forma sencilla y además incluyen gran cantidad de librerías de objetos para representar los diferentes dispositivos de la industria como motores, tanques, indicadores e interruptores.

Los sistemas HMI están compuestos por un conjunto de programas y archivos. Hay programas que son para diseño y configuración del sistema y otros que son el motor mismo del sistema. Con los programas de diseño se crean moldes de pantallas para la visualización de los datos de los procesos, estos moldes son guardados en archivos que almacenarán la forma en la que serán visualizados los datos de las pantallas. Además, con estos programas se encargan de refrescar las variables de las bases de datos en la pantalla, y actualizarlas, si corresponde, por entradas del teclado o ratón. Las bases de datos son un lugar en la memoria de la computadora donde se

almacenan los datos requeridos para el proceso. Estos datos varían en el tiempo según cambien los datos de los procesos, por esta razón se denominan “bases de datos dinámicas”. Las bases de datos están formadas por bloques que pueden estar interconectados, estos bloques pueden recibir información de otros bloques o de los drivers y pueden enviar información hacia otros bloques o hacia los drivers. La comunicación entre los bloques de la base de datos y las señales de los procesos se realiza mediante manejadores o drivers. Estos drivers son los que manejan la comunicación entre el HMI y los distintos dispositivos de campo. (6)

En el SCADA UX, el módulo HMI presenta dos ambientes, el ambiente de edición que permite configurar los procesos, definir y gestionar las variables, editar los controladores, los comandos, las alarmas y opciones adicionales. El editor es el que permite a un operador definir el ambiente de trabajo del SCADA, adaptándolo mejor a la aplicación particular que se desea desarrollar. Por otra parte, la edición gráfica proporciona al operador las funciones de control y supervisión de la planta. El proceso se representa mediante gráficos representativos, almacenados en el ordenador o bien importados desde otras aplicaciones durante la configuración del paquete de software; y el ambiente de ejecución que es el encargado de posibilitar al HMI el monitoreo y supervisión de los procesos del sistema previamente configurados en el ambiente de edición. Se refiere a las características o componentes que estarán disponibles al usuario cuando éste inicie al sistema en el ambiente de ejecución.

1.8. Herramientas y tecnologías

A continuación se enunciarán las herramientas y tecnologías utilizadas, sin ellas no fuera posible darle cumplimiento al objetivo trazado.

Eclipse

Es un entorno de desarrollo integrado, de Código abierto y Multiplataforma. Mayoritariamente se utiliza para desarrollar lo que se conoce como "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Es una potente y completa plataforma de Programación, desarrollo y compilación de elementos tan variados como sitios web, programas en C++ o aplicaciones Java. No es más que un entorno de desarrollo integrado (IDE) que provee todas las herramientas y funciones necesarias para el trabajo, recogidas además en una atractiva interfaz que lo hace fácil y agradable de usar. (7)

Para el desarrollo de este trabajo se utilizó el IDE Eclipse debido a las ventajas que ofrece, entre ellas podemos enunciar que emplea extensiones, esto será de gran utilidad ya que, al no tener todas las funcionalidades incluidas no se consume memoria innecesaria, contribuyendo a la velocidad del desarrollo. Este mecanismo de módulos permite:

- ✓ Usar otros lenguajes de programación como son C/C++ y Python.

- ✓ Trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y sistema de gestión de base de datos.

Se provee soporte para Java y CVS en el SDK de Eclipse; esto provee una fácil integración con una amplia gama de elementos, entre ellos el framework Qt y con SVN, esto facilita el trabajo en equipo, el compartimiento de recursos y mantener el control de cambios de todo el trabajo.

C++

Es un lenguaje imperativo orientado a objetos derivado del C. En este trabajo se utilizó debido a sus múltiples ventajas, entre ellas el hecho de que como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción. Además, la definición "oficial" del lenguaje nos dice que C++ es un lenguaje de propósito general basado en el C, al que se han añadido nuevos tipos de datos, clases, plantillas, mecanismo de excepciones, sistema de espacios de nombres, funciones inline, sobrecarga de operadores, referencias, operadores para manejo de memoria persistente, y algunas utilidades adicionales de librería.(8)

Todo esto incrementa las posibilidades de lo que se puede hacer con el lenguaje, hace que sea más intuitivo y más fácil de aprender. Entre lo que se puede desarrollar con C++ se tiene:

- ✓ **Núcleos y drivers de dispositivos**: Tanto los drivers como el núcleo funcionan en un nivel realmente bajo de operaciones en el ordenador. Para escribir el núcleo del sistema operativo y acceder a las propiedades del hardware tales como los ciclos de la memoria, buses de entrada/salida, etc., se necesita un lenguaje que pueda comunicarse con el hardware con potencia. El núcleo Linux hace uso del lenguaje C con una pequeña parte en lenguaje ensamblador.
- ✓ **Bibliotecas y utilidades**: Las bibliotecas y utilidades básicas del sistema tales como mkdir, chmod, chown, head, tail, chroot, uptime, users también están escritas en lenguaje C.(9)
- ✓ **Gestores de paquetes y programas de configuración**: Los gestores de paquetes tales como yum, apt, dpkg, etcétera, también están escritos en C, que como veis es la estrella absoluta de los componentes "base" de un sistema GNU/Linux.
- ✓ **Entornos de Escritorio y gestores de ventanas**: La mayoría de gente usa un entorno de escritorio, y es que a día de hoy poco queda sólo en interfaz de línea de comandos. Los gestores de ventanas tales como metacity, kwin están desarrollados en C y requieren gcc para ser compilados. El entorno de escritorio, iconos, ventanas, barras de herramientas, etc, están basados en librerías específicas y hacen uso del lenguaje C.
- ✓ **Aplicaciones gráficas de usuario**: Este es el punto donde entran en juego una gran cantidad de lenguajes de programación, dado que básicamente es la capa más alta. Tenemos una

gran variedad: C, Python, Java, Perl y otros. Hay librerías GTK+, Tcl/Tk, Qt que son un frontend gráfico a tareas que corren por detrás en línea de comando.(10)

Visual Paradigm

Es una herramienta CASE: Ingeniería de Software Asistida por Computación. La misma fue muy importante en el desarrollo de este trabajo debido a que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Esta herramienta fue diseñada para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable, a través de la utilización de un enfoque Orientado a Objetos.(11)

En este trabajo el uso de esta herramienta fue de mucha ayuda ya que esta tiene una gran disponibilidad en múltiples plataformas (Windows, Linux), presenta un diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad, usa un lenguaje estándar UML que es común a todo el equipo de desarrollo lo que facilita la comunicación, posee capacidades de ingeniería directa e inversa, además el modelo y el código permanecen sincronizados en todo el ciclo de desarrollo, tiene soporte para varios idiomas, es fácil de instalar y actualizar, se pueden realizar diagramas de procesos de negocio - proceso, decisión, actor de negocio y documentos, tiene una gran interoperabilidad con modelos UML2 (metamodelos UML 2.x para plataforma Eclipse) a través de XMI, tiene gran integración con las bases de datos, se puede transformar de diagramas de Entidad-Relación en tablas de base de datos y de Sistemas Gestores de Bases de Datos existentes a diagramas de Entidad-Relación, además posee un potente generador de informes y un editor de figuras.

Qt

Es una biblioteca multiplataforma para desarrollar aplicaciones, las cuales pueden ser con o sin interfaz gráfica. Es utilizada principalmente por grandes compañías desarrolladoras de productos de software como Autodesk Maya, Dassault DraftSight, Google Earth, KDE, Adobe Photoshop Album, Skype, Qt Extended, VLC media player, VirtualBox y Mathematica. Funciona en todas las principales plataformas, y tiene un amplio apoyo. En este trabajo se hace uso de esta biblioteca porque utiliza el lenguaje de programación C++ de forma nativa, adicionalmente puede ser utilizado en varios otros lenguajes de programación a través de bindings. (12)

Además se utiliza para realizar todas las interfaces gráficas necesarias. La API (Interfaz de Programación de Aplicaciones) de la biblioteca cuenta con métodos para acceder a bases de datos

mediante SQL, así como uso de XML, 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. Qt es software libre y de código abierto.(13)

La biblioteca C++ Qt provee un rico conjunto de componentes para la construcción de aplicaciones, distribuidas en módulos.

- ✓ QtCore: Clases no gráficas utilizadas por los otros módulos.
- ✓ QtGui: Componentes para las interfaces gráficas de usuario.
- ✓ QtNetwork: Programación de redes.
- ✓ QtOpenGL: Soporte a OpenGL.
- ✓ QtScript: Evaluación de Qt Scripts.
- ✓ QtScriptTools: Componentes Qt Script adicionales.
- ✓ QtSql: Intregración con SQL.
- ✓ QtSvg: Clases para mostrar el contenido de ficheros SVG.
- ✓ QtWebKit: Clases para mostrar y editar contenido web.
- ✓ QtXml: Manejo del XML.
- ✓ QtXmlPattern: Motores XQuery y XPath para XML y modelo de datos personalizados.
- ✓ Phonon: Clases del Framework mutimedia.
- ✓ Qt3Support: Soporte de compatibilidad a clases de Qt 3.5.(14)

1.9. Metodologías

Lograr la construcción de un sistema informático eficiente, que cumpla con los requerimientos planteados, es una tarea realmente intensa y sobre todo difícil de cumplir. Las metodologías para el desarrollo del software imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Una metodología de desarrollo de software tiene como principal objetivo aumentar la calidad del software que se produce en todas y cada una de sus fases de desarrollo. No existe una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable. Las metodologías de desarrollo se pueden dividir en dos grupos de acuerdo con sus características y los objetivos que persiguen: ágiles y robustas.(15)

Las metodologías ágiles se caracterizan por hacer énfasis en la comunicación cara a cara, es decir, se basan en una fuerte y constante interacción, donde clientes y desarrolladores trabajan constantemente juntos, estableciéndose así una estrecha comunicación. Estas metodologías están orientadas al resultado del producto y no a la documentación; exige que el proceso sea adaptable,

permitiendo realizar cambios de último momento. Se puede hacer mención dentro de las metodologías ágiles a: XP (por sus siglas en inglés Extreme Programming), Scrum y Crystal Methodologies.

Las metodologías robustas o tradicionales Están guiadas por una fuerte planificación. Centran su atención en llevar una documentación exhaustiva de todo el proceso de desarrollo y en cumplir con un plan de proyecto, definido en la fase inicial del mismo. Entre las metodologías robustas se encuentran: MSF (por sus siglas en inglés Microsoft Solution Framework), MÉTRICA 3 y RUP (siglas de Rational Unified Process).

En este trabajo se utiliza RUP como metodología de desarrollo de software, por ser esta una metodología eficaz que se adapta a las características propias del software que se desarrolla. Esta metodología permite enfocarse en trabajar de forma organizada, donde se controla y documenta todo lo relacionado con el proyecto. Con la utilización de RUP se genera la documentación que es imprescindible para desarrollar el proyecto o para presentárselo al cliente. En esencia, esta metodología genera los artefactos necesarios. Está basada en componentes e interfaces bien definidas, y junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

Debido a que RUP es un proceso, en su modelación se definen como sus elementos principales:

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. (16)

El ciclo de vida de RUP se caracteriza por ser **dirigido por casos de uso** (CU). Los CU 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 (cómo se llevan a cabo). Además es **centrado en la arquitectura**, esta 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 CU relevantes desde el punto de vista de la arquitectura. El modelo de arquitectura se representa a través de vistas en las que se incluyen los diagramas de UML.(17)

Iterativo e Incremental: Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros.

Es una metodología que divide el desarrollo de software en 4 fases, la fase de inicio, en la que se determina la visión del proyecto; la fase de elaboración, en la que se determina la arquitectura óptima del software; la fase de construcción, en la que se obtiene una capacidad operacional inicial; y por último la fase de transición, donde se obtiene un entregable del producto. Cada una es desarrollada mediante un ciclo de iteraciones, que consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes.

En RUP se han agrupado las actividades en grupos lógicos definiéndose 9 flujos de trabajo principales, los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como flujos de apoyo.(18)

Modelamiento del Negocio: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.

Requerimiento: Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.

Análisis y Diseño: Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.

Implementación: Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.

Prueba: Busca los defectos a lo largo del ciclo de vida.

Instalación o despliegue: Produce un entregable del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.

Administración del proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.

Administración de configuración y cambios: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.

Ambiente: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización. (16)

Conclusiones parciales

Se realizó un estudio de los sistemas SCADA más avanzados existentes actualmente, enfatizando en los módulos de HMI y configuración. A esto se suma que se utilice la metodología tradicional RUP, que se adecua a las características del proyecto así como, las herramientas a utilizar que son el Visual Paradigm para lograr una óptima modelación de la problemática, como IDE el Eclipse, con el objetivo de realizar un excelente manejo del lenguaje de programación C++. Además se seleccionó como biblioteca a utilizar Qt, para lograr una buena respuesta y un uso aceptable de la memoria. A partir de este punto se comenzará el desarrollo de la propuesta de solución.

Capítulo 2 “Análisis y diseño”

En este capítulo se expondrá el flujo de trabajo de análisis y diseño realizado para modelar y dar respuesta al problema planteado. Se escogió realizar un Modelo de Dominio debido a que se adecúa mejor para la representación de conceptos, sus atributos y asociaciones con otros conceptos lo que permite modelar objetos y sus interacciones en el mundo real. Se va a hacer referencia a las capacidades que el mecanismo de configuración en caliente debe cumplir (Requerimientos funcionales), así como las cualidades que este debe tener (Requerimientos no funcionales). También se mostrará el diagrama de clases de diseño, y las descripciones de las clases que forman parte de este diagrama.

2.1 Modelo de dominio

El Modelo de Dominio es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Representa clases conceptuales del dominio del problema, conceptos del mundo real en lugar de componentes de software.(19)

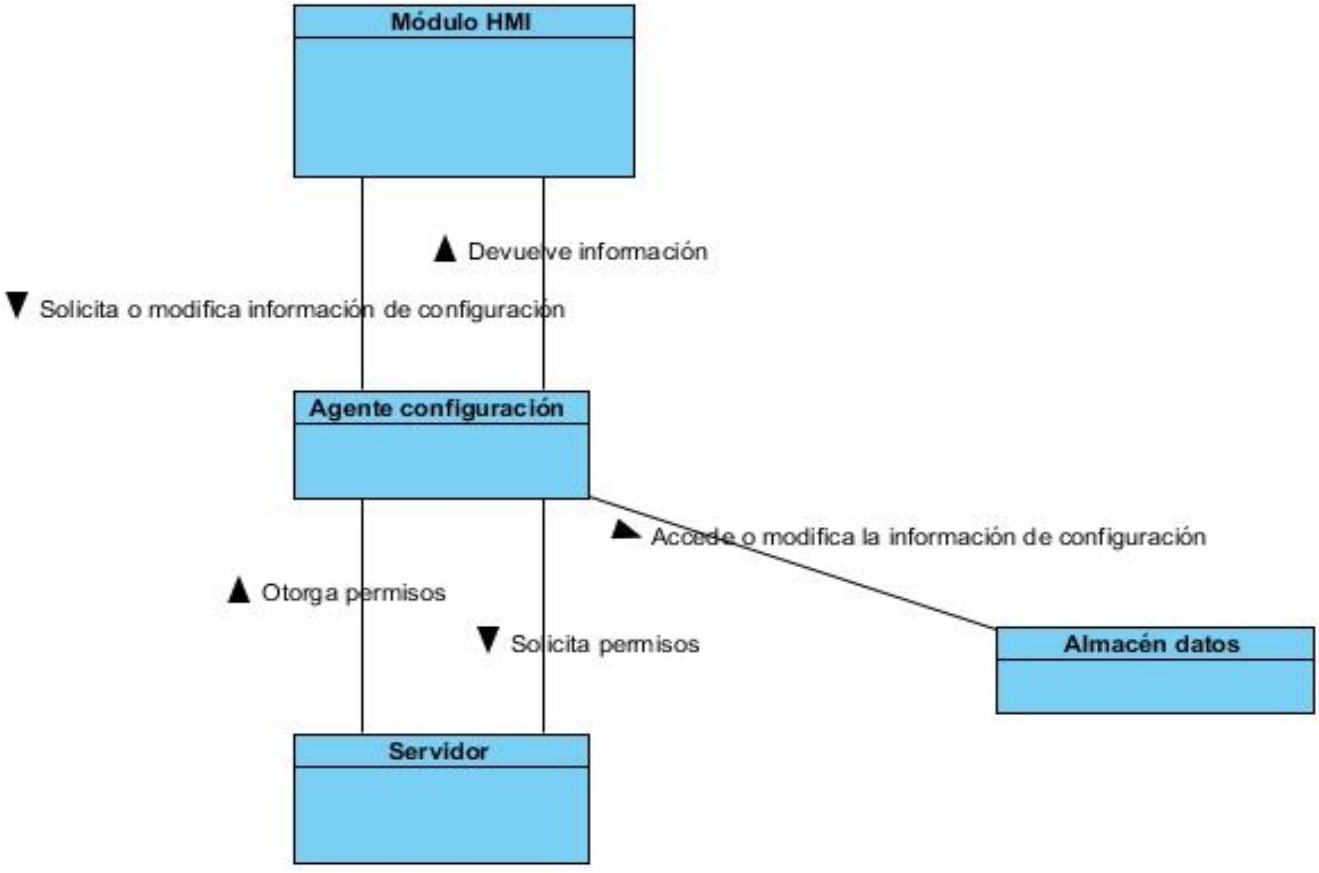

Figura 2: Modelo de dominio.

El **módulo de HMI** es el encargado de solicitar o modificar la información de configuración mediante el agente de configuración.

El **agente de configuración** es el mediador entre el módulo HMI, el servidor de configuración y los datos. Este le solicita permisos al servidor de configuración para poder acceder o modificar la información de configuración.

El **servidor de configuración** es el encargado de otorgar los permisos para ejecutar cualquier cambio en la configuración.

En el **almacén de datos** es donde se guarda toda la información referente a la configuración del sistema.

2.2 Modelo de datos

Para entender lo que es la información de configuración, se presenta el siguiente modelo de datos donde se puede observar las entidades que son configurables dentro del módulo HMI:

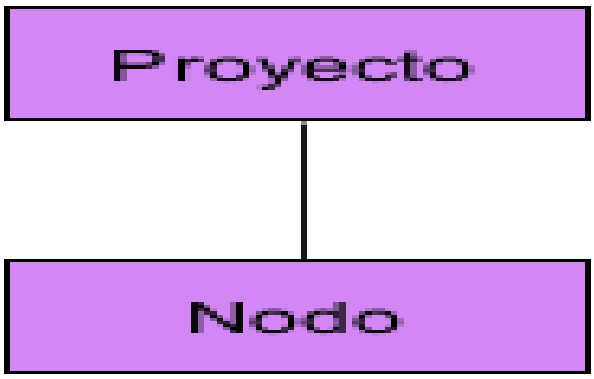

Figura 3: Modelo de datos, entidades generales.

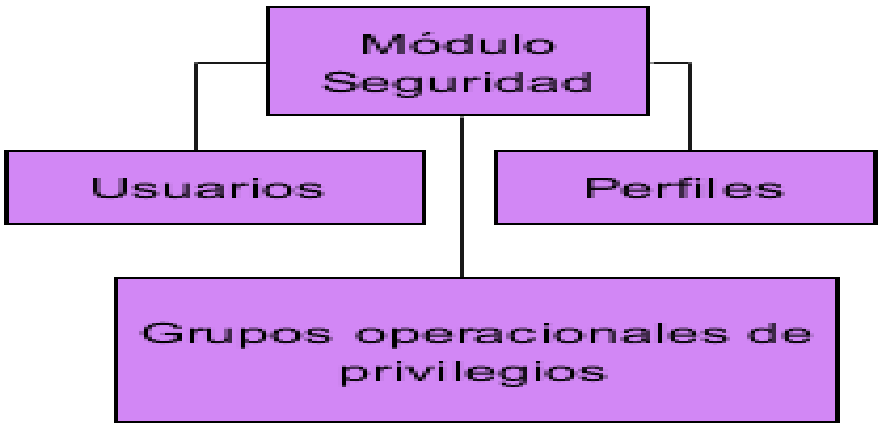

Figura 4: Modelo de datos, módulo de seguridad.

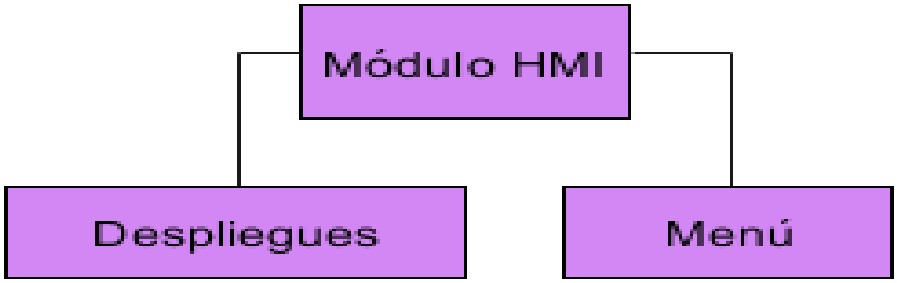

Figura 5: Modelo de datos, módulo de HMI.

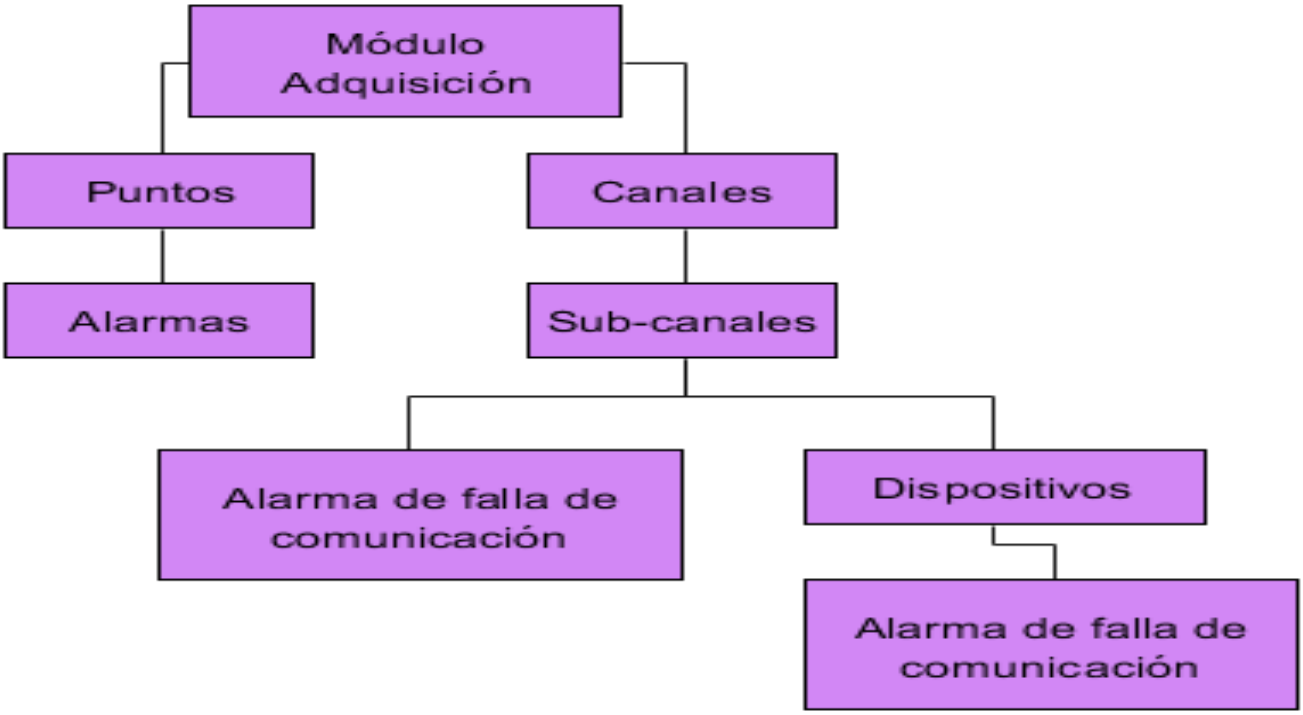

Figura 6: Modelo de datos, módulo de adquisición.

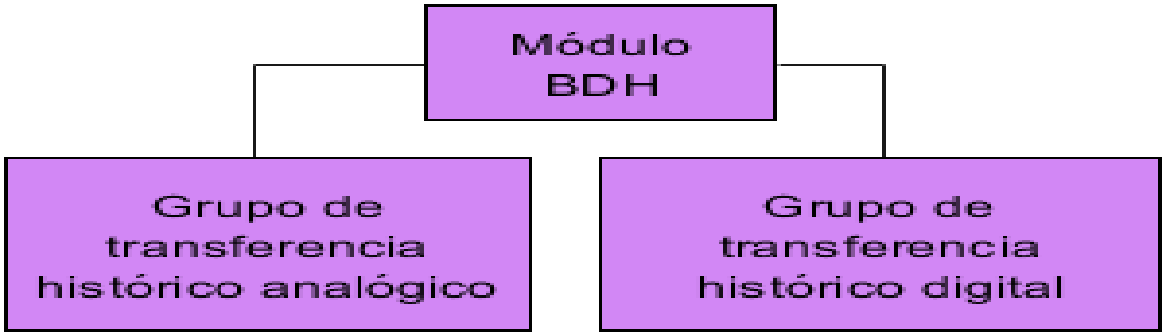

Figura 7: Modelo de datos, módulo de BDH.

Proyecto: Es la base de la configuración, representa el sistema o proceso en general a configurar. Se le pueden agregar varios nodos.

Nodo: Se define como la unidad física donde se instancia uno o varios módulos del sistema. En este sistema se pueden agregar tantos como se desee o también se puede tener solamente uno donde se configuren todos los módulos del sistema.

Módulos: El sistema se configura utilizando cuatro tipos de módulos (módulo HMI, módulo de adquisición, módulo de seguridad y módulo de base de datos histórica). En el módulo HMI se configuran los despliegues y menú, estos serán las principales interfaces de información e interacción de los operadores con el sistema en ejecución. En el módulo de adquisición se encuentran los puntos y se define la configuración de las comunicaciones (canales, sub-canales, dispositivos). En el módulo de seguridad se definen los usuarios, perfiles, y grupos operacionales de privilegios. Permitiendo así asignar permisos a los usuarios sobre los diferentes recursos del sistema. En el módulo de base de datos histórica se configuran los grupos de transferencias históricas analógicas y digitales, en esta base de datos se guardaran todos los eventos y acciones del sistema para su posterior consulta.

Puntos: Representan las variables a monitorizar y controlar en el proceso. Los valores de estas se recolectan en el campo y se muestran a los operadores mediante la interfaz a la cual están asociadas. Pueden ser de diferentes tipos, este sistema solo abarcará las analógicas y digitales. Los analógicos se utilizan para medir valores de unidades de ingeniería como pueden ser presión o temperatura, mientras que los digitales representan valores binarios, por ejemplo para monitorizar el estado de un interruptor (abierto o cerrado, encendido o apagado).

Canales: Representa el medio físico a través del cual se conectan un conjunto de dispositivos, el principal parámetro que se le configura a este recurso es el modo de acceso. La interacción con este medio físico en el sistema es independiente de la interacción con el resto de los dispositivos del SCADA.

Subcanales: A un canal se le pueden agregar varios sub-canales, estos representan el protocolo de comunicación médiate el cual se conectará uno o varios dispositivos. Nuestro sistema contará en un principio con manejadores para los siguientes protocolos: EthernetIP, ModbusTCP, OPC, ABEthernet, ModbusASCII y ModbusRTU.

Dispositivos: Los dispositivos son equipos electrónicos que pueden ser autómatas, PLC, reguladores autónomos, sensores inteligentes, controladores etc. Estos son el primer eslabón en la adquisición de los datos. A estos se le asocian variables previamente configuradas, cuyos valores se obtendrán mediante estos equipos.

Alarmas: Las alarmas se basan en la vigilancia de los parámetros de las variables del sistema. Se evidencian en los sucesos no deseables, porque su aparición puede dar lugar a problemas de funcionamiento. Estas requieren de la atención de un operario para su solución antes de que llegue a una situación crítica que afecte el proceso. A las alarmas en general que se configuran en este

sistema como en la mayoría de estos, se les asigna una severidad la cual podrá ser Alta, Media o Baja, este parámetro define la criticidad de la alarma, la de severidad Alta representa una alarma crítica que requiere la acción inmediata del operador. Otro parámetro asignado a las alarmas es la prioridad, de modo que si aparecen varias de forma simultánea, las de mayor prioridad aparecerán primero.

Alarma de falla de comunicación: En el sistema también podemos encontrar alarmas de falla de comunicación. Estas se le configuran a los sub-canales y los dispositivos. Se le especifica un número máximo de reintentos antes de que se active la alarma.

Despliegues: La herramienta para la configuración del sistema también permite crear pantallas con múltiples combinaciones de imágenes y texto las cuales representarán gráficamente las funcionalidades del proceso a controlar. Los objetos gráficos que se colocaran en estos despliegues brindan la posibilidad de asociarles las variables del sistema que van a representar. Por ejemplo es posible visualizar una variable analógica mediante un objeto en forma de barra el cual se mostraría según el valor de esta variable.

Usuarios: El sistema brinda la facilidad de definir usuarios que harán uso de la aplicación, a estos se le asignará un perfil.

Perfiles: El perfil definirá las características que pueden tener uno o varios usuarios. Por ejemplo los días que ese usuario trabajará y las horas que tendrá por sesión de trabajo. A los perfiles se le asocia un grupo operacional de privilegios.

Grupos Operacionales de privilegios: Este recurso permitirá agrupar usuarios y recursos permitiendo el acceso a estos según los permisos que se definen en un perfil al asociarle un grupo, estos permisos pueden ser lectura, escritura y configuración.

Grupos de transferencia a históricos: Estos grupos definen la forma de almacenamiento de las variables en la base de datos histórica del sistema, se le configura el tipo de almacenamientos y tipo de grupo. Se le asocian las variables que almacenaran según los parámetros configurados en este. Estos grupos se clasifican en analógicos y digitales.(20)

2.3 Captura de requisitos

Se hace necesario realizar una captura de requisitos que recoja de forma coherente los nuevos requerimientos para el mecanismo de configuración en caliente. Estos requisitos tienen como objetivo definir qué funcionalidades hacen falta para para dar solución a los mismos.

2.3.1 Requisitos funcionales

Los requisitos funcionales de un sistema describen lo que el sistema debe hacer. Definen con detalle la función del sistema, sus entradas, salidas, etc.

Se identificaron como requisitos funcionales:

RF1 Registrar cambios.

RF1.1 Registrar cambios en la entidad Proyecto.

RF1.2 Registrar cambios en la entidad Nodo.

RF1.3 Registrar cambios en la entidad Usuario.

RF1.4 Registrar cambios en la entidad Perfil.

RF1.5 Registrar cambios en la entidad Grupo Operacional de Privilegios.

RF1.6 Registrar cambios en la entidad Despliegues.

RF1.7 Registrar cambios en la entidad Menú.

RF1.8 Registrar cambios en la entidad Punto.

RF1.9 Registrar cambios en la entidad Alarma.

RF1.10 Registrar cambios en la entidad Canal.

RF1.11 Registrar cambios en la entidad Subcanales.

RF1.12 Registrar cambios en la entidad Dispositivo.

RF1.13 Registrar cambios en la entidad Grupo de Transferencia Histórico Analógico.

RF1.14 Registrar cambios en la entidad Grupo de Transferencia Histórico Digital.

RF2 Actualizar identificador.

RF2.1 Actualizar identificador en la entidad Proyecto.

RF2.2 Actualizar identificador en la entidad Nodo.

RF2.3 Actualizar identificador en la entidad Usuario.

RF2.4 Actualizar identificador en la entidad Perfil.

RF2.5 Actualizar identificador en la entidad Grupo Operacional de Privilegios.

RF2.6 Actualizar identificador en la entidad Despliegues.

RF2.7 Actualizar identificador en la entidad Menú.

RF2.8 Actualizar identificador en la entidad Punto.

RF2.9 Actualizar identificador en la entidad Alarma.

RF2.10 Actualizar identificador en la entidad Canal.

RF2.11 Actualizar identificador en la entidad Subcanales.

RF2.12 Actualizar identificador en la entidad Dispositivo.

RF2.13 Actualizar identificador en la entidad Grupo de Transferencia Histórico Analógico.

RF2.14 Actualizar identificador en la entidad Grupo de Transferencia Histórico Digital.

RF3. Salvar configuración en el servidor en ambiente de edición.

RF4 Cargar configuración desde el servidor en ambiente de edición.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales son aquellos que se refieren a las propiedades emergentes de éste como fiabilidad, usabilidad, restricciones en el diseño e implementación, etc.

Usabilidad

RnF1. El módulo debe mantener el mismo procedimiento de arranque y puesta en marcha que poseía antes de agregar una nueva funcionalidad.

Fiabilidad

RnF2.1. Integridad de datos: El sistema debe ser capaz de mantener la calidad de los datos de manera que garantice su integridad durante su recepción, procesamiento y almacenamiento en la base de datos de configuración.

Soporte

RnF3.1 El sistema debe poseer soporte metodológico y tecnológico para lograr su correcto funcionamiento.

RnF3.2 El sistema debe ser de fácil instalación, configuración y puesta en marcha.

RnF3.3 La arquitectura debe ser abierta y distribuida, modular, de capacidad escalable y tecnología actualizable de acuerdo a las necesidades operacionales y tendencias tecnológicas de las aplicaciones y componentes que se ejecutan o interactúan con el sistema.

RnF3.4 La programación del sistema debe estar orientada a objeto.

Restricciones de diseño

RnF4.1 Empleo del mismo conjunto de herramientas en el desarrollo del sistema: Se debe emplear el mismo conjunto de herramientas para el desarrollo del sistema, así como para futuros desarrollos. De manera tal que garantice la escalabilidad y compatibilidad entre los diferentes desarrollos.

2.4 Diagrama de casos de uso del sistema

Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas. O lo que es igual, un diagrama que muestra la relación entre los actores y los casos de uso en un sistema. Una relación es una conexión entre los elementos del modelo, por ejemplo la especialización y la generalización son relaciones. Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema al mostrar cómo reacciona una respuesta a eventos que se producen en el mismo.(21)

A continuación se presentará el diagrama de casos de uso del sistema.

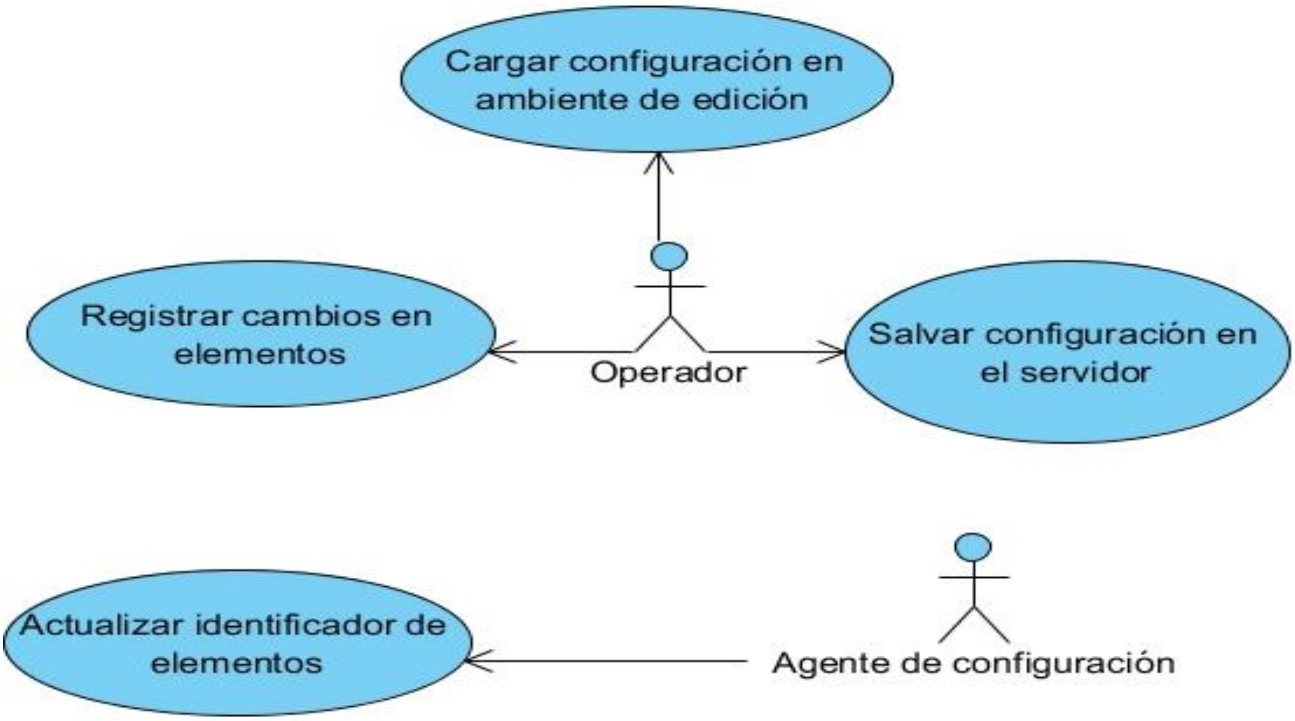

Figura 8: Diagrama de casos de uso del sistema.

2.5 Descripción de los casos de uso.

En las siguientes tablas se describen con detalles, el funcionamiento de los casos de uso expuestos.

CU Salvar configuración en el servidor.

Objetivo	Salvar configuración en el servidor.
Actores	Operador (Inicia)
Resumen	El caso de uso se inicia cuando el operador desea salvar la configuración en la que está trabajando en ese momento, ejecuta la acción y termina el caso de uso.
Complejidad	Media
Prioridad	Media
Precondiciones	
Postcondiciones	Se salvó la configuración en el servidor.
Flujo de eventos	
Flujo básico Salvar configuración en el servidor	
Actor	**Sistema**

	1.1 Hace clic en el menú “Archivo” de la barra de herramientas.	1.2 Despliega la opción “subir configuración al servidor”
	2.1 Selecciona la opción “Almacenar configuración en el servidor”.	2.2 Muestra una ventana para especificar la dirección IP y el puerto donde se encuentra el servidor.
	3.1 Introduce la dirección IP y el puerto donde se encuentra el servidor. 3.2 Hace clic en el botón Aceptar.(Alterno 1)	3.3 Valida que los parámetros de la conexión al servidor estén correctos.(Alterno 2) 3.4 Salva la configuración de cada una de las entidades en el servidor.
		4.1 Termina el caso de uso.
Flujos alternos		
Nº Evento 1 Selecciona la opción Cancelar.		
	Actor	**Sistema**
	1.1 En el paso 3.2 del flujo básico hace clic en el botón Cancelar.	1.2 Cierra la ventana de especificación de la dirección IP y puerto del servidor.
		2.1 Termina el caso de uso.
Flujos alternos		
Nº Evento 2 Parámetros de conexión incorrectos		
Actor		**Sistema**
		1.1 En el paso 3.3 del flujo básico, muestra mensaje de alerta informando que los parámetros de conexión al servidor son

		incorrectos.
		2.1 Regresa al paso 2.2 del flujo básico.
Requisitos no funcionales	No procede	
Asuntos pendientes	No procede	

CU Cargar configuración en el ambiente de edición.

Objetivo	Cargar configuración desde el servidor en ambiente de edición.	
Actores	Operador (Inicia)	
Resumen	El caso de uso se inicia cuando el operador desea cargar la configuración guardada anteriormente en el servidor, ejecuta la acción y termina el caso de uso.	
Complejidad	Media	
Prioridad	Media	
Precondiciones	Debe existir una configuración guardada en el servidor.	
Postcondiciones	Se cargó la configuración desde el servidor en ambiente de edición.	
Flujo de eventos		
Flujo básico Cargar configuración desde el servidor en ambiente de edición		
Actor		**Sistema**
	1.1 Hace clic en el menú “Archivo” de la barra de herramientas.	1.2 Despliega la opción “Cargar configuración desde el servidor”.
	2.1 Selecciona la opción “Descargar configuración desde el servidor”.	2.2 Muestra una ventana para especificar la dirección IP y el puerto donde se encuentra el servidor con la configuración que se desea cargar.
	3.1 Introduce la dirección IP y el puerto de donde se encuentra el servidor.	3.3 Valida que los parámetros de la conexión al

	3.2 Hace clic en el botón Aceptar. (Alterno 1)	servidor estén correctos. (Alterno 2) 3.4 Carga la configuración de cada una de las entidades desde el servidor.
		4.1 Termina el caso de uso.
Flujos alternos		
Nº Evento 1 Selecciona la opción Cancelar.		
	Actor	**Sistema**
	1.1 En el paso 3.2 del flujo básico hace clic en el botón Cancelar.	1.2 Cierra la ventana de especificación de la dirección IP y el puerto del servidor.
		2.1 Termina el caso de uso.
Flujos alternos		
Nº Evento 2 Parámetros de conexión incorrectos		
Actor		**Sistema**
		1.1 En el paso 3.3 del flujo básico, muestra mensaje de alerta informando que los parámetros de conexión al servidor son incorrectos.
		2.1 Regresa al paso 2.2 del flujo básico.
Requisitos no funcionales	No procede	
Asuntos pendientes	No procede	

CU Actualizar identificadores de los elementos.

Objetivo	Actualizar los identificadores de las entidades involucradas en el

	sistema.
Actores	Agente de configuración (Inicia)
Resumen	El caso de uso se inicia cuando el agente de configuración salva la configuración en el servidor, obtiene los nuevos identificadores de los elementos y termina el caso de uso.
Complejidad	Media
Prioridad	Media
Precondiciones	Debe haberse ejecutado el caso de uso Salvar configuración en el servidor.
Postcondiciones	Se actualizaron los identificadores de cada uno de los elementos.

Flujo de eventos		
Flujo básico Actualizar identificadores de los elementos		
	Actor	**Sistema**
	1.1 Detecta que se ha terminado de salvar la configuración. 1.2 Envía una petición para la actualización de los nuevos identificadores de los elementos.	1.3 Verifica la conexión con el servidor. (Alterno 1) 1.4 Muestra un mensaje preguntando si desea actualizar los identificadores de los elementos. 1.5 Obtiene los nuevos identificadores de los elementos.
		2.1 Actualiza los identificadores de cada una de las entidades.
		3.1 Termina el caso de uso.
Flujos alternos		
Nº Evento 1 No existe conexión con el servidor		
	Actor	**Sistema**
		1.1 En el paso 1.3 del flujo básico muestra un mensaje de alerta para informar que no existe

		conexión con el servidor.
		2.1 Termina el caso de uso.
Requisitos no funcionales	No procede	
Asuntos pendientes	No procede	

CU Registrar cambios en elementos

Objetivo	Registrar los cambios que se realizan sobre las entidades involucradas en el sistema.	
Actores	Operador(Inicia)	
Resumen	El caso de uso se inicia cuando el operador realiza un cambio sobre alguna de las entidades (adicionar, modificar, eliminar), el sistema lo registra y termina el caso de uso.	
Complejidad	Media	
Prioridad	Media	
Precondiciones		
Postcondiciones	Se registró el cambio realizado.	
Flujo de eventos		
Flujo básico		
Sección 1: Adicionar elemento.		
Actor		**Sistema**
	1.1 Hace clic derecho sobre el árbol de navegación.	1.2 Muestra la opción “Adicionar” de acuerdo al elemento en el que se encuentre.
	2.1 Selecciona la opción adicionar.	2.2 Muestra una ventana para que se especifiquen los datos del elemento a adicionar.
	3.1 Introduce los datos del elemento a adicionar. 3.2 Hace clic en el botón Aceptar.	3.3 Adiciona el nuevo elemento. 3.4 Registra el cambio

	(Alterno 1)	realizado.
		4.1 Termina el caso de uso.
Flujos alternos		
Nº Evento 1 Selecciona la opción Cancelar.		
	Actor	**Sistema**
	1.1 En el paso 3.2 del flujo básico hace clic en el botón Cancelar.	1.2 Cierra la ventana para la adición del elemento.
		2.1 Termina el caso de uso.
Sección 2: Modificar elemento		
Flujo básico		
	Actor	**Sistema**
	1.1 Hace clic derecho sobre el elemento que desea modificar.	1.2 Muestra la opción “Modificar” de acuerdo al elemento en el que se encuentre.
	2.1 Selecciona la opción “Modificar”.	2.2 Muestra una ventana con los datos del elemento que se desea modificar.
	3.1 Modifica los datos que desea del elemento escogido. 3.2 Hace clic en el botón Aceptar. (Alterno 1)	3.3 Modifica los datos del elemento 3.4 Registra el cambio.
		4.1 Termina el caso de uso.
Flujos alternos		
Nº Evento 1 Selecciona la opción Cancelar.		
	Actor	**Sistema**
	1.1 En el paso 3.2 del flujo básico hace clic en el botón Cancelar.	1.2 Cierra la ventana para la modificación del elemento.
		2.1 Termina el caso de uso.

Sección 3: Eliminar elemento		
Flujo básico		
	Actor	**Sistema**
	1.1 Hace clic derecho sobre el elemento que desea eliminar.	1.2 Muestra la opción “Eliminar” de acuerdo al elemento en el que se encuentre.
	2.1 Selecciona la opción “Eliminar”.	2.2 Elimina el elemento. 2.3 Verifica que la configuración del elemento en cuestión haya sido cargada desde el servidor. (Alterno 1) 2.4 Registra el cambio.
		3.1 Termina el caso de uso
Flujos alternos		
Nº Evento 1 La configuración no fue cargada del servidor.		
	Actor	**Sistema**
		1.1 En el paso 2.2 del flujo básico si la configuración no fue cargada del servidor el cambio no es registrado.
		2.1 Termina el caso de uso.
Requisitos no funcionales	No procede	
Asuntos pendientes	No procede	

2.6 Diagramas de interacción

En el análisis, las interacciones entre objetos se pueden representar a través de diagramas de secuencia o de colaboración. Para la solución que se propone, se emplean diagramas de secuencia,

que se utilizan para identificar secuencias de interacción detalladas y ordenadas cronológicamente.(22)

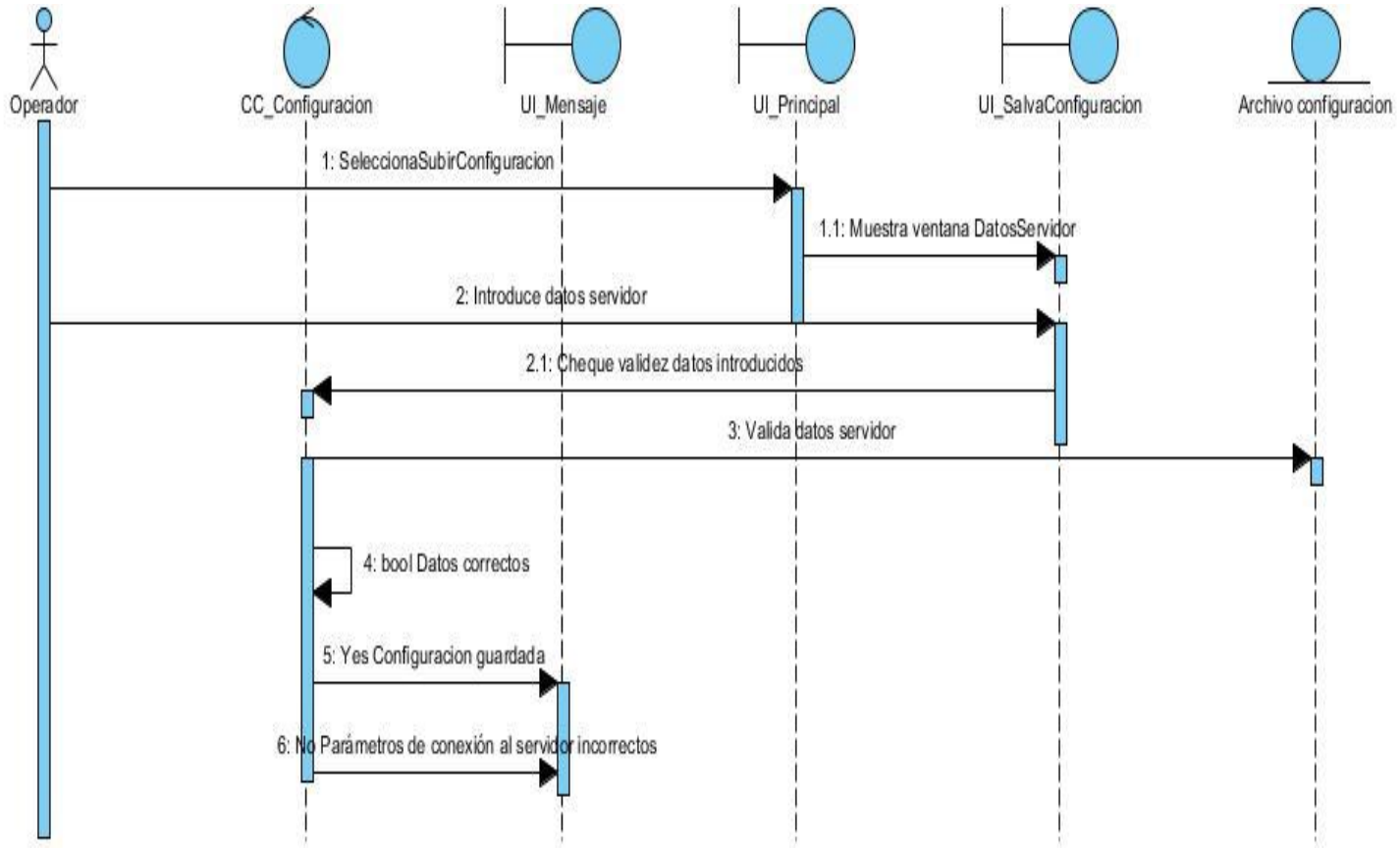

Figura 9: CU Salvar configuración en el servidor.

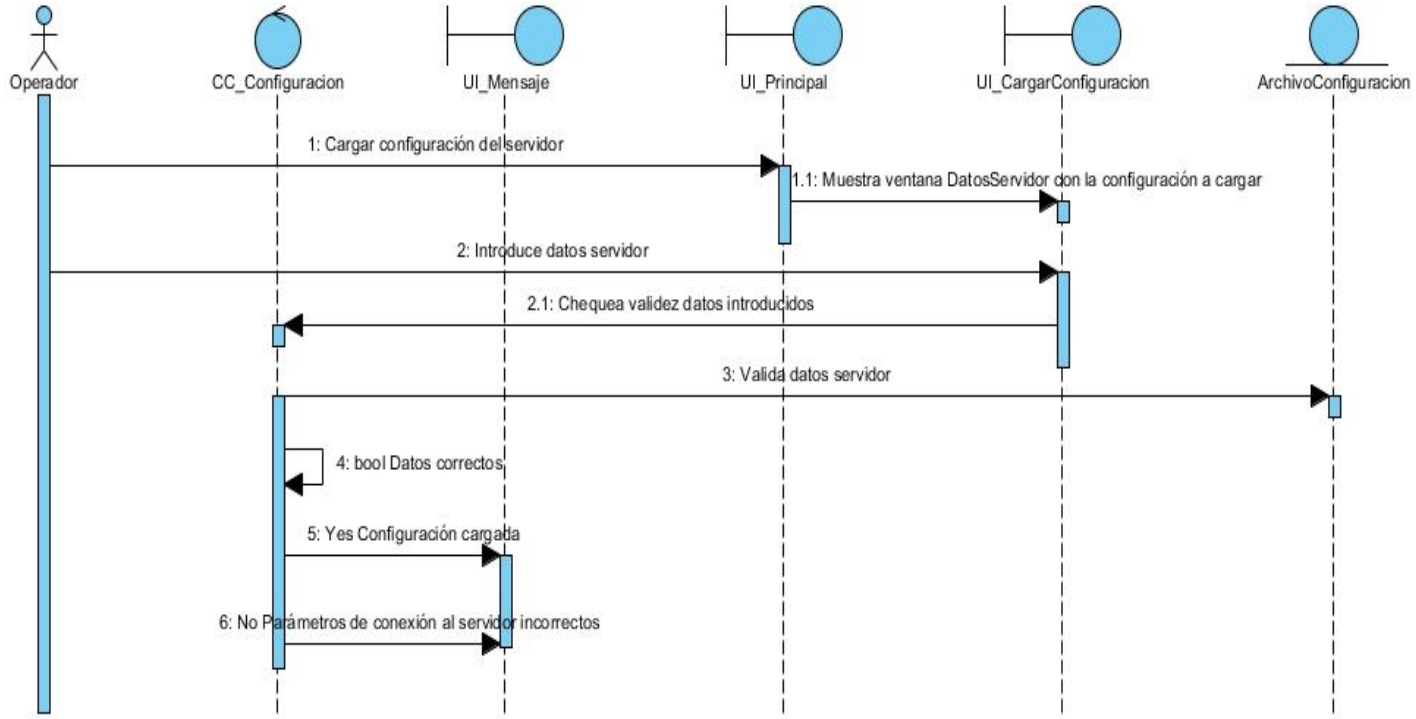

Figura 10: CU Cargar configuración en ambiente de edición.

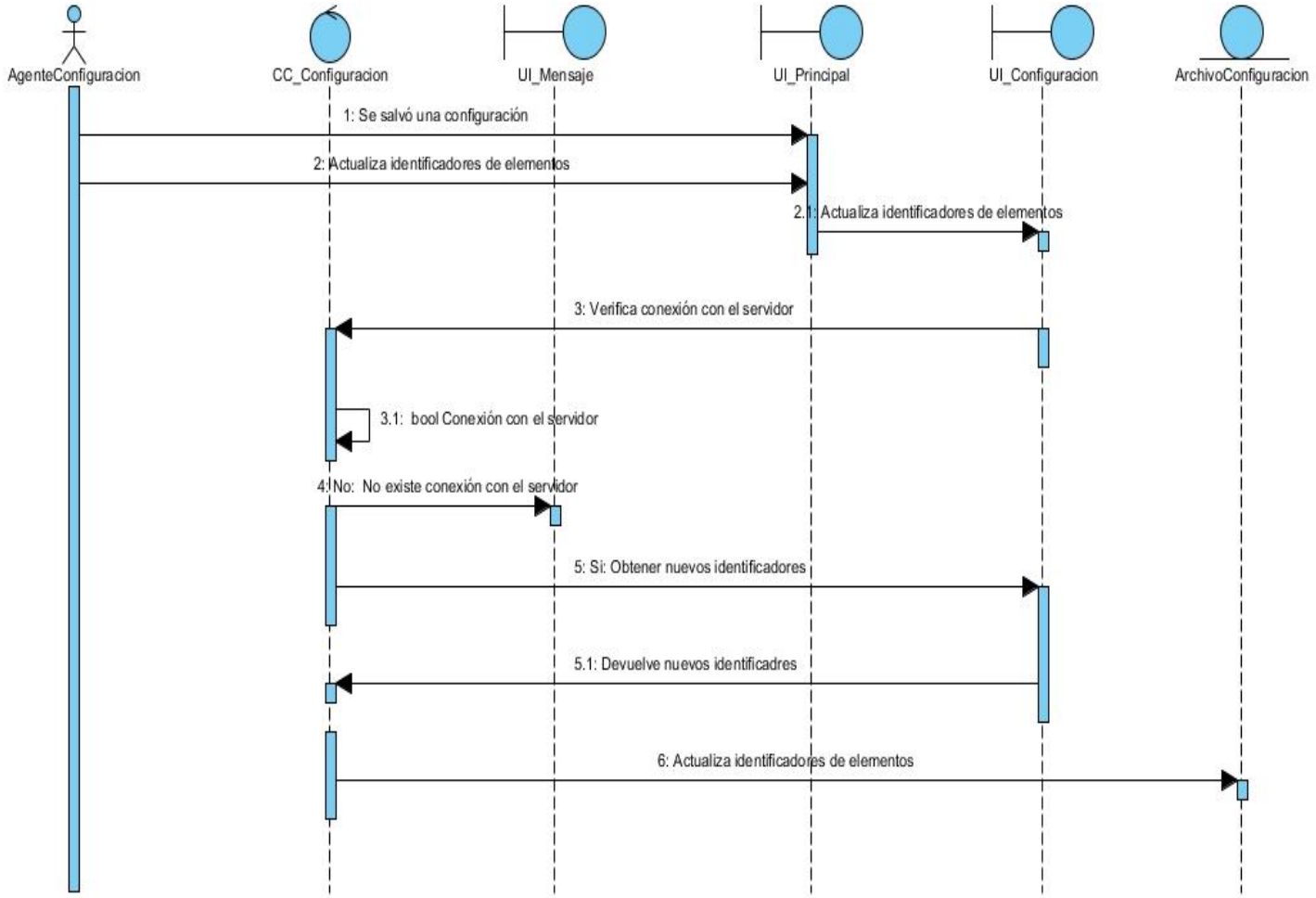

Figura 11: CU Actualizar identificadores de elementos.

2.7 Diagrama de clases del diseño

A continuación se presenta el diagrama de clases del diseño, para un mejor entendimiento se dividió en varias imágenes que se explicarán más adelante.

En las siguientes imágenes (Figura 10, 11, 12, 13 y 14), se muestran los detalles de las clases más importantes involucradas en la configuración. Se dividió por paquetes para una mayor organización lógica. Se tiene el paquete **Core** con la clase **ItemModel**, luego está el paquete **Common** con la clase **Configurable**, luego el paquete **Editor** con las clases correspondientes a la entidades Proyecto y Nodo y por último el paquete **SainuxExtension** que contiene los módulos de seguridad, con su clase principal **SecurityModule**; luego está el módulo de HMI y su clase principal **HMIModule;** además se tiene al módulo de BDH y su clase **BDHModule** y por último el módulo de adquisición con su clase principal **AdquisitionModule**.

En cada uno de los diagramas siguientes además de las clases principales ya mencionadas están las clases correspondientes a las entidades involucradas en cada módulo, en cada una de estas clases se incluyeron solo las funciones más importantes para esta investigación.

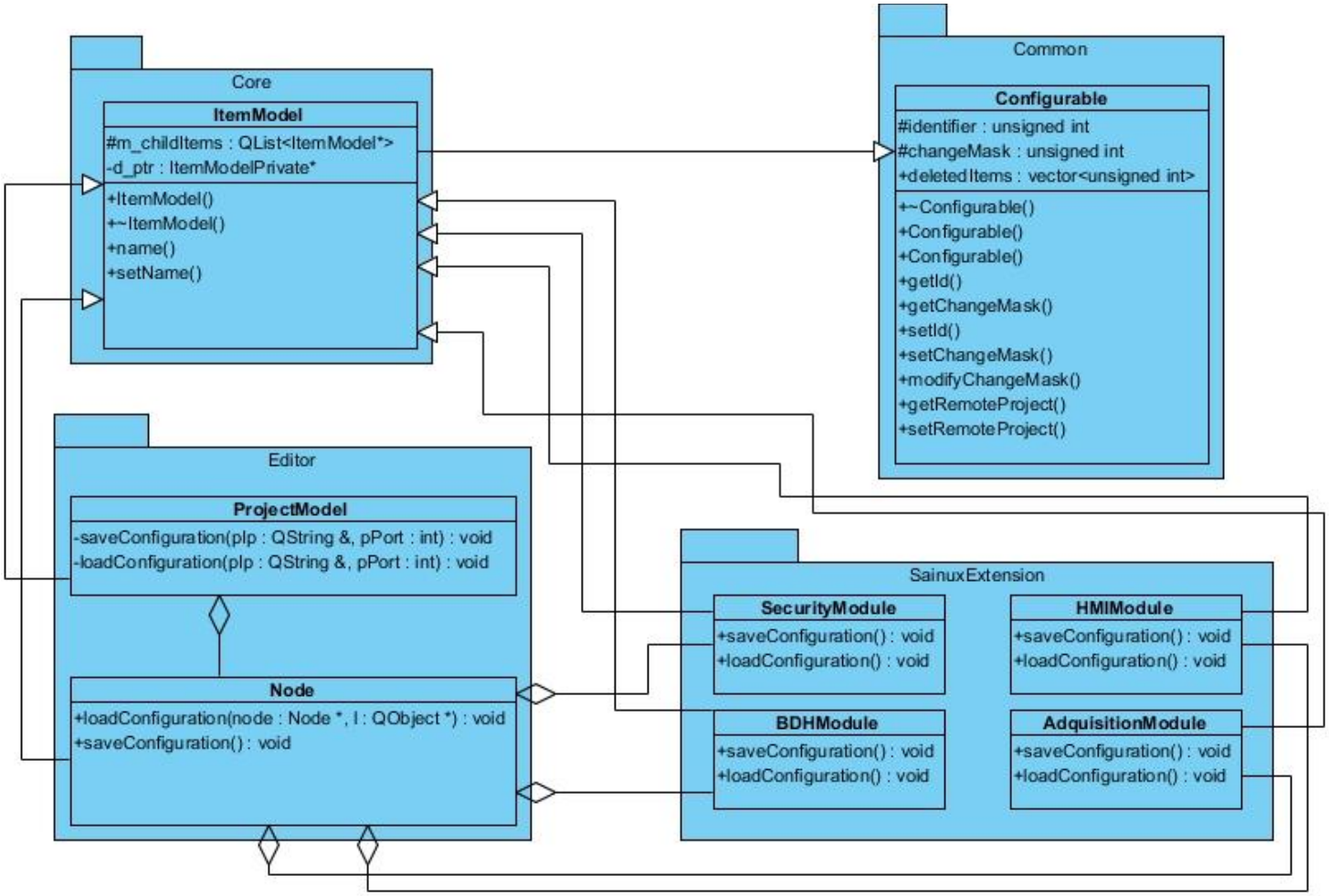

Figura 12: Diagrama de clases del diseño, entidades generales.

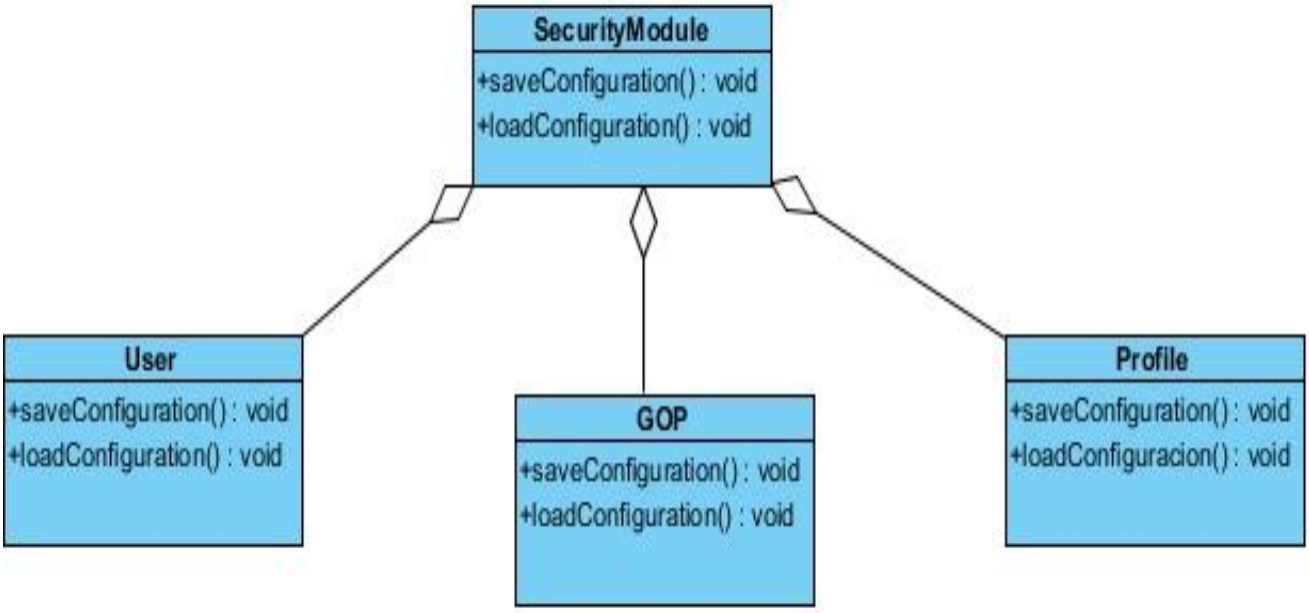

Figura 13: Diagrama de clases del diseño, módulo de seguridad.

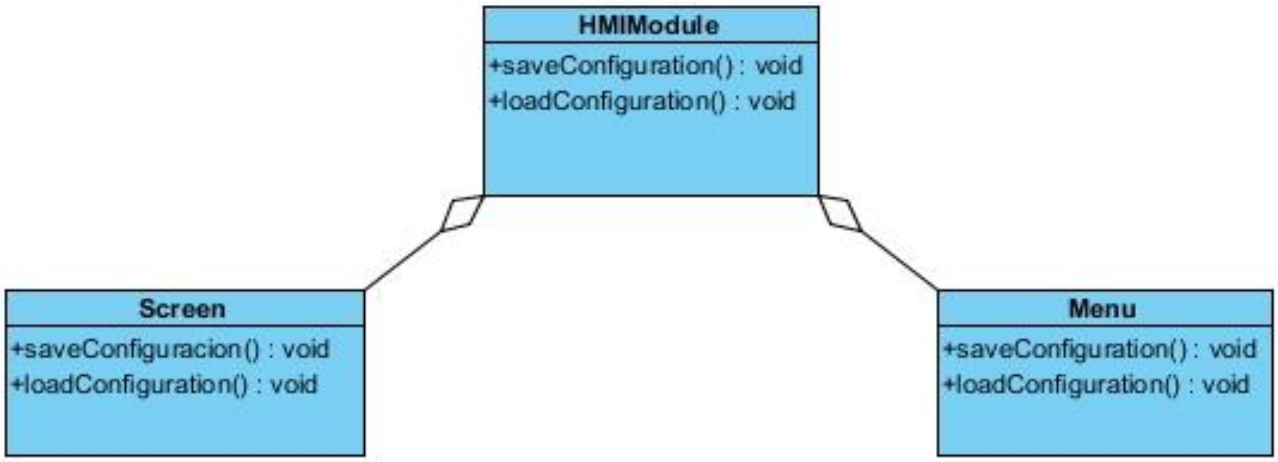

Figura 14: Diagrama de clases del diseño, módulo de HMI.

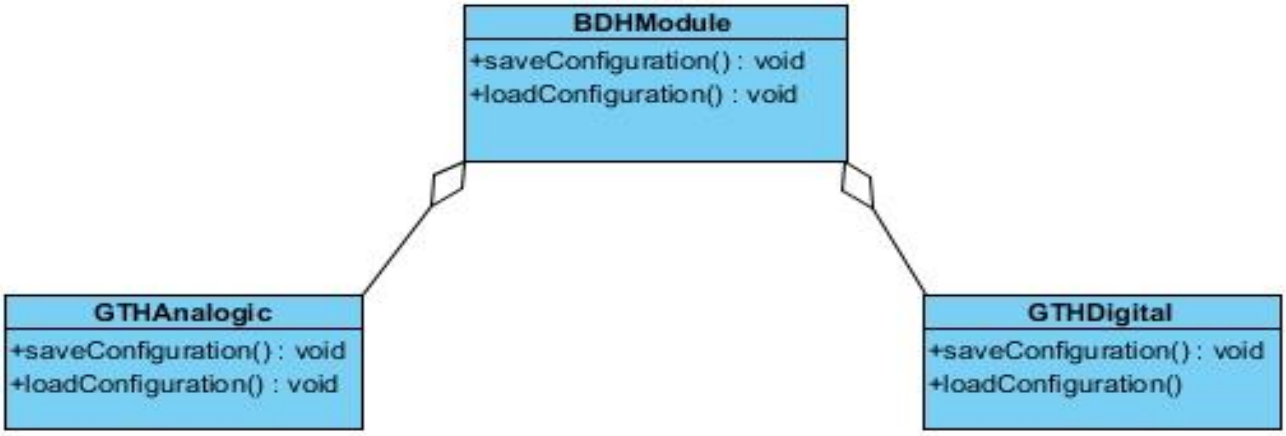

Figura 15: Diagrama de clases del diseño, módulo de BDH.

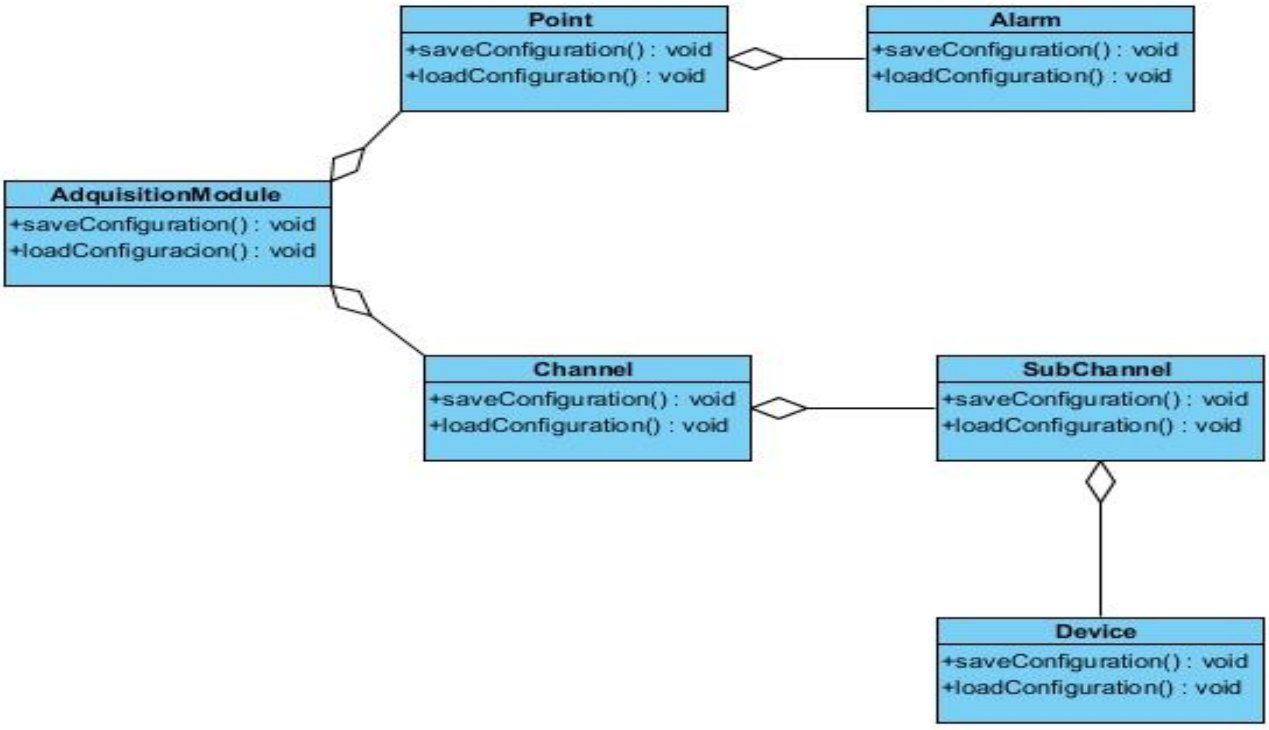

Figura 16: Diagrama de clases del diseño, módulo de Adquisición.

2.8 Descripción de clases

De las clases mencionadas anteriormente se consideran como las más importantes para esta investigación **ItemModel** y **Configurable**, a continuación se explican más detalles de las mismas.

Clase ItemModel

La clase **ItemModel** se encuentra dentro del paquete **Core** y tiene como propósito representar los elementos que se muestran en el árbol de navegación.

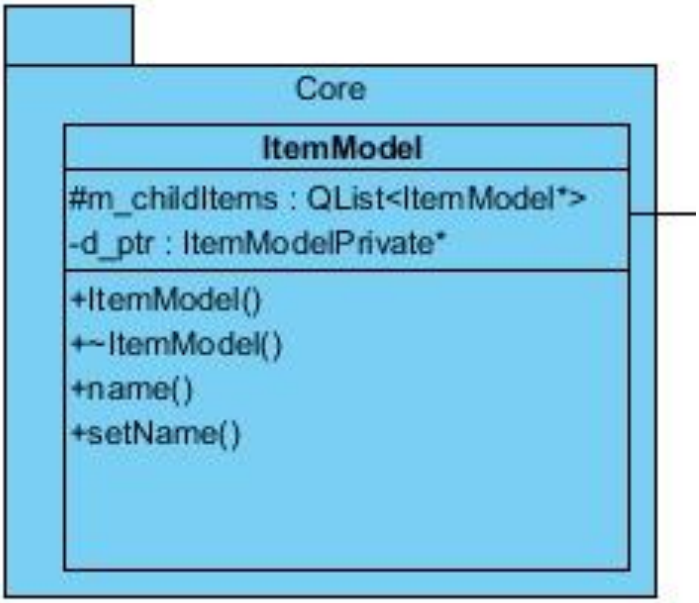

Figura 17: Clase ItemModel.

Descripción

Esta es una clase muy importante ya que contiene las características necesarias para poder incluirse en el árbol de navegación. Todas las clases correspondientes a las entidades que intervienen en el proceso de carga y salva de la configuración heredan de esta clase para poder ser visualizadas en el árbol.

Clase configurable

La clase **Configurable** se encuentra dentro del paquete **Common**, y tiene como propósito representar todos los elementos que se pueden configurar dentro de las entidades que se muestran en el árbol de navegación.

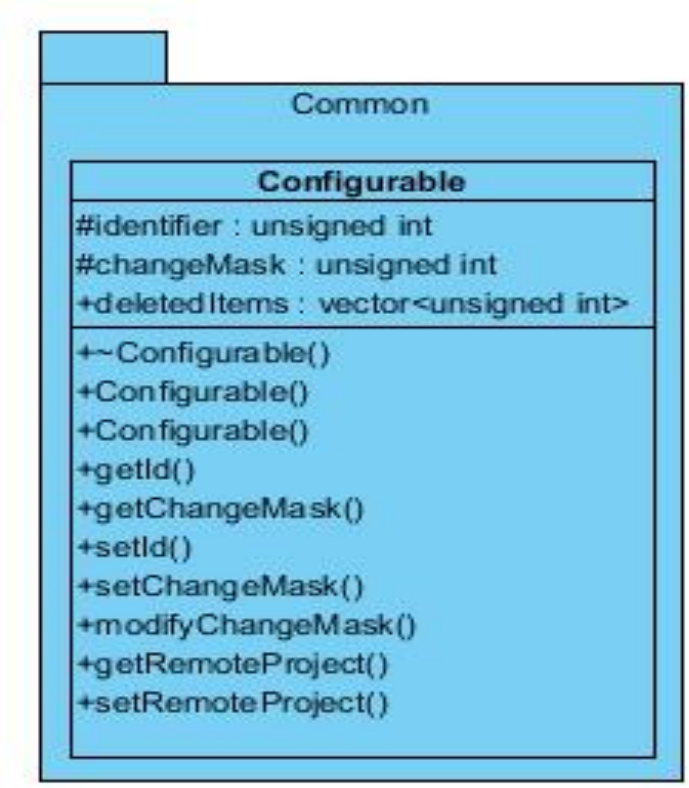

Figura 18: Clase Configurable.

Descripción

Esta clase es la encargada de contener los elementos que permiten que las entidades sean configurables. Dentro de las funciones de esta clase se encuentra **getRemoteProject()**, dicha función es la que devuelve si el elemento en cuestión (Proyecto, Nodo o algún módulo), es cargado del servidor o fue creado localmente por el operador. Conocer esto es muy importante debido a que en el proceso de registrar los cambios ocurridos en los elementos, si el mismo es creado localmente, se tiene en cuenta los cambios de tipo adicionar y modificar, y no los cambios de tipo eliminar. En cambio si el proyecto es cargado de un servidor se tienen en cuenta todos los tipos de cambios. Además tiene las funciones **getId()**, esta función se encarga de devolver el identificador de la entidad en cuestión, cada elemento solo tendrá un identificador después de haber sido salvada la configuración. También tiene la función **getChangeMask()**, esta se encarga de devolver la máscara de cambios de cada elemento del árbol, dicha máscara es un número que contiene la información referente a que elemento específico cambió dentro de la entidad en cuestión. Otra de las funciones son **setChangeMask()** y **modifyChangeMask()**, dichas funciones trabajan sobre la máscara de cambios, la primera cambia la máscara completa y la segunda cambia la máscara bit a bit, este trabajo con la máscara de cambios es muy importante porque es lo que va a permitir en el proceso de cargar y salvar la configuración, actualizar solo los cambios ocurridos, actualmente se realiza

actualizando todo el árbol aun cuando solo haya ocurrido un pequeño cambio en una de sus entidades. Por último se va a tratar **deletedItems**, esto es un vector de identificadores de elementos eliminados, aquí cada elemento guarda los identificadores de los hijos que hayan sido eliminados, es importante conocer que esto solo se tendrá en cuenta cuando el elemento seleccionado sea cargado desde el servidor de configuración.

Conclusiones parciales

El Modelo de Dominio, elaborado a partir de los principales conceptos involucrados en el entorno del problema, constituyó la antesala a la identificación de los requerimientos funcionales y no funcionales del mecanismo de configuración en caliente. Estos elementos permitieron la concepción de los casos de uso representados en el Modelo de Casos de Uso del Sistema, así como las descripciones textuales de cada uno. La unión de estos artefactos sirvió de guía en la elaboración del Modelo de Análisis con sus respectivos diagramas de interacción por cada caso de uso. De igual forma se confeccionó el Modelo de Diseño del mecanismo de configuración en caliente, entradas fundamentales para las actividades de implementación y prueba.

Capítulo 3: Solución propuesta

Los artefactos UML creados durante el Análisis y Diseño, como son: los diagramas de interacción y diagramas de clases del diseño del mecanismo de configuración en caliente, constituyen la entrada fundamental para las actividades de implementación y prueba. En la primera, se describe cómo los elementos del Modelo de Diseño se implementan en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares. Una vez realizadas las actividades de implementación, se requiere realizar las pruebas para detectar la presencia de errores, debido a la imposibilidad humana de trabajar y comunicarse de forma perfecta.

3.1 Modelo de implementación

El Modelo de Implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes podemos encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos. Este artefacto describe cómo se implementan los componentes, congregándolos en subsistemas organizados en capas y jerarquías, y señala las dependencias entre éstos.(16)

3.1.1 Diagrama de despliegue

Los diagramas de despliegue son los complementos de los diagramas de componentes que, unidos, proveen la vista de implementación del sistema. Describen la topología del sistema, la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos. Los diagramas de despliegue representan a los nodos y sus relaciones. Los nodos son conectados por asociaciones de comunicación tales como enlaces de red, conexiones TCP/IP.(23)

El siguiente diagrama de despliegue está compuesto por los nodos necesarios para todo el proceso de carga y salva de la configuración en caliente en el módulo de HMI. Dicho módulo utiliza el **servidor de comunicaciones** para enviar la solicitud requerida al **servidor de configuración**, y a su vez este interactúa con los datos del **servidor de base de datos**, todas estas conexiones se realizan el protocolo TCP/IP.

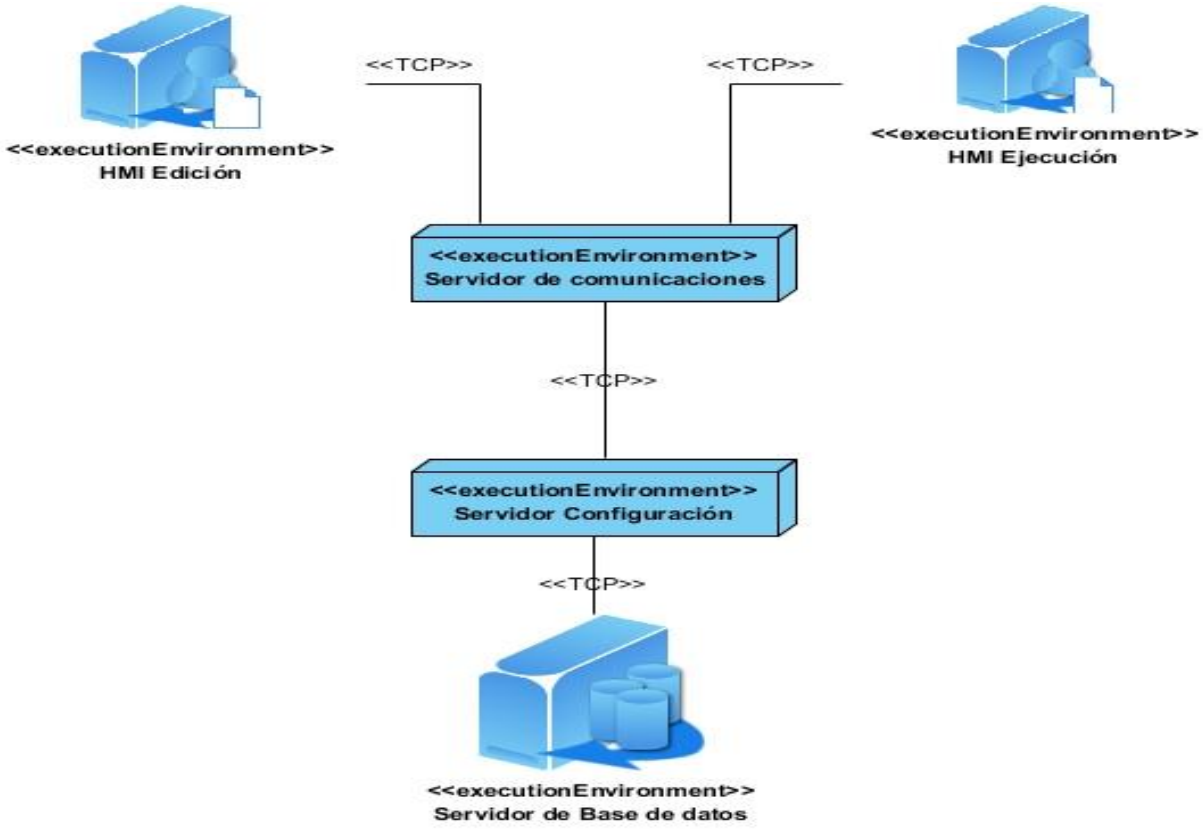

Figura 19: Diagrama de despliegue del mecanismo de configuración en caliente.

3.1.2 Diagrama de componentes

Por su parte, el diagrama de componentes es otro de los artefactos importantes que incluye el Modelo de Implementación. El mismo muestra elementos del modelo, tales como, los componentes y sus relaciones. Se utiliza para modelar la vista estática del sistema y muestra la organización y las dependencias lógicas entre los componentes de software, sean estos de código fuente, binarios o ejecutables.(24)

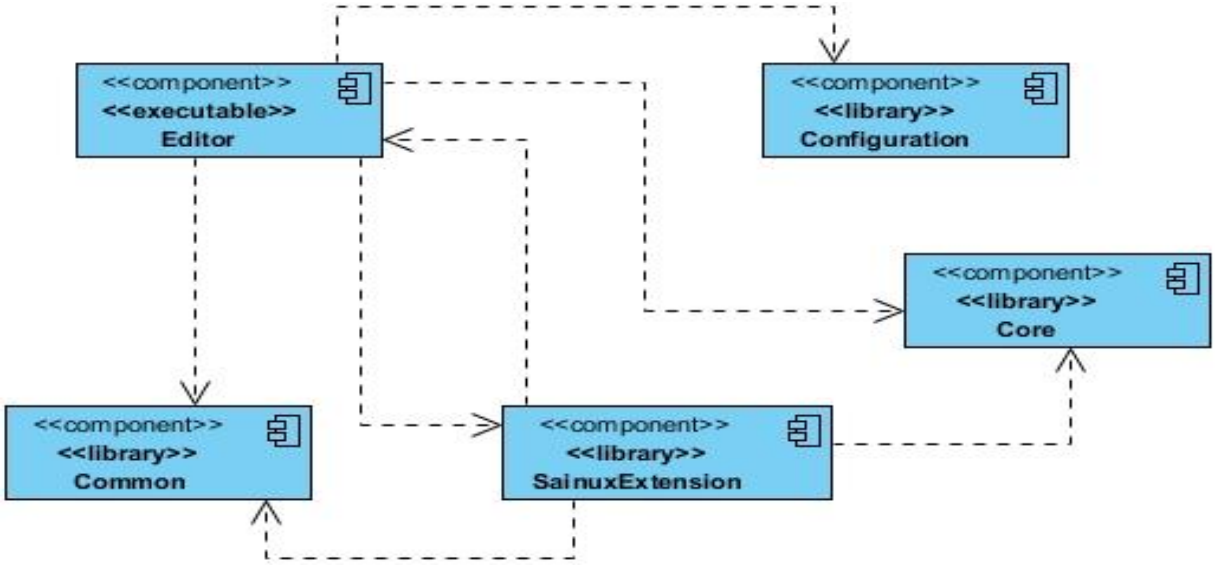

Figura 20: Diagrama de componentes del mecanismo de configuración en caliente.

A continuación se describen cada uno de los componentes presentados en la figura 18 correspondientes al mecanismo de configuración en caliente para el módulo de HMI.

- ✓ **Core**: Este componente es el núcleo de todo el módulo de HMI.
- ✓ **Common**: Aquí se encuentran algunas clases que son útiles y comunes para el resto del módulo de HMI.
- ✓ **Editor**: Interfaz gráfica del editor del módulo HMI.
- ✓ **SainuxExtension**: Es el encargado de manejar todas las entidades relacionadas con los elementos de los módulos de seguridad, HMI, BDH y Adquisición de datos.
- ✓ **Configuration**: Módulo de configuración.

3.2 Principales funcionalidades

A continuación se explicarán algunas de las principales funcionalidades implementadas.

En primer lugar se encuentra **registrar cambios**, un cambio ocurre cuando se adiciona, se modifica o se elimina un elemento. Para el caso de adicionar y modificar un elemento, se actualiza la máscara de cambios asociada a cada elemento, dicha máscara no es más que un valor entero convertido a binario, donde cada bit corresponde a un atributo configurable en el elemento seleccionado, estando el bit en 1 si se modifica y en 0 si no ha sido modificado. En el caso de que sea eliminar, se le adiciona el identificador del elemento que se va a eliminar a una estructura que contiene el nodo padre del mismo en el árbol. Esta funcionalidad es la que permite que solamente se actualicen los cambios en la base de datos y no que se tenga reescribir la configuración completa.

Otra de las funcionalidades comienza cuando el operador desea a **salvar la configuración**, para ejecutar esta acción se muestra una ventana para introducir los datos de conexión (dirección IP y

puerto), después de hecho esto se verifican estos datos, si están correctos se inicializa el agente configuración. Estando el mismo inicializado, se comienza a construir la estructura de los elementos que se han modificado, comenzando por el proyecto, luego los nodos y por último cada módulo. Esta estructura tiene que ser comprendida por el módulo de configuración. Cada elemento es el encargado de salvar su configuración y mandar a salvar la de sus descendientes en el árbol. En todo este proceso se tiene muy en cuenta la máscara de cambios, solo se salvarán los parámetros que se indican en la máscara de cambios. Por último se manda al agente de configuración a que salve toda la estructura creada.

Cuando el operador manda a **cargar la configuración**, se muestra una ventana para introducir la dirección IP y el puerto, luego de verificar que son correctos se inicializa el agente de configuración con estos datos. Luego, si el agente de configuración se inicializó correctamente, se procede a realizar el proceso inverso al anterior, el módulo de configuración brinda una estructura que el módulo de HMI puede comprender para poder cargar los datos (identificador, nombre, descripción, etc.) referentes al proyecto, luego los de los nodos y por último los de cada módulo que se encuentre en el servidor. Cada elemento es el responsable de cargar su configuración y mandar a cargar la de sus hijos en caso de tener Esta carga de la configuración se realiza teniendo en cuenta la máscara de cambios, así se optimiza este proceso ya que solo se cargan los elementos indicados en dicha máscara.

3.3 Patrones de diseño

Un patrón se plantea como una solución a un problema en un contexto, codifica conocimiento específico acumulado por la experiencia en un dominio determinado.

Los Patrones de Diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software. Un patrón de diseño es una solución a un problema de diseño, además, estos pretenden proporcionar catálogos de elementos reusables en el diseño de sistemas software, formalizar un vocabulario común entre diseñadores y estandarizar el modo en que se realiza el diseño.(25)

3.3.1 Patrones GRASP

GRASP, Acrónimo de General Responsability Assignment Software Patterns (Patrones de Software para la Asignación General de Responsabilidad). Estos patrones describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades.(26)

Patrón alta cohesión

El patrón alta cohesión plantea asignar una responsabilidad de modo que la cohesión siga siendo alta, la cohesión es una medida de la fuerza con la que se relacionan las clases y el grado de focalización de las responsabilidades de un elemento. Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable.

Una clase con baja cohesión hace muchas cosas no afines o realiza un trabajo excesivo. Estas a menudo representan un alto grado de abstracción o han asumido responsabilidades que deberían haber delegado a otros objetos.

Este patrón fue utilizado en el diseño de la aplicación de manera general; donde se agruparon las clases, según la premisa de que cada clase debe implementar las operaciones que estén sobre la misma área funcional.

Patrón Experto

El propósito del patrón Experto es asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. La responsabilidad de la creación de un objeto o la implementación de un método debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtendrá un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento). (27)

Dicho patrón se evidencia en la definición de las clases de acuerdo a las funcionalidades que deben realizar a partir de la información que manejan. Específicamente en las clases de cada una de las entidades, cada una de ellas será la responsable de implementar la lógica del elemento que representa (Proyecto, Nodo, módulos).

Patrón Creador

El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debe conectar con el objeto producido en cualquier evento. Al escoger este patrón, se da soporte al bajo acoplamiento.

El patrón creador se percibe en las clases que representan una entidad que tiene la responsabilidad de crear otra, tal es el caso de la clase ProjectModel que se encarga de crear la clase Nodo, esto se logra debido a que ProjectModel conoce los datos necesarios para crear los nodos.

3.3.2 Patrones GOF

Los patrones Gang-of-Four (“pandilla de los cuatro”) o comúnmente llamados Patrones GOF, descritos en el libro Design Patterns (Gama 1995) definen un catálogo con 23 patrones básicos. Según el libro GOF existen 3 tipos de patrones:

- ✓ *De Creación*: abstraen el proceso de creación de instancias.
- ✓ *Estructurales*: se ocupan de cómo clases y objetos son utilizados para componer estructuras de mayor tamaño.

- ✓ *De Comportamiento*: atañen a los algoritmos y a la asignación de responsabilidades entre objetos.(25)

Patrón Singleton

El patrón de diseño Singleton (instancia única) como patrón de creación, está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella. Este patrón se implementa creando en una clase un método que crea una instancia del objeto sólo si todavía no existe alguna. Para asegurar que la clase no puede ser instanciada nuevamente se regula el alcance del constructor (con atributos como protegido o privado).

Este patrón ha sido escogido para ponerlo en uso en el módulo Core, el mismo solo se instancia una vez cuando inicia la aplicación, ya sea en ambiente de edición o de ejecución, y persiste hasta que se cierra la aplicación, logrando tener una instancia global de este módulo.

Patrón Observer

La mayoría de librerías para el desarrollo de aplicaciones con GUI como Qt, está diseñado para ser dirigidos por eventos, de ahí el famoso sistema de ranuras y señales (Signals and Slots) de Qt que no es más que un patrón de diseño Observer bastante curado. Señales y slots es una construcción del lenguaje introducido en Qt, lo que hace que sea fácil de implementar el patrón Observer, evitando código repetitivo.(28)

La idea principal detrás del patrón de comportamiento *Observer* es que existe una entidad con estados cambiantes y una o más entidades observándola. Los observadores esperan a que la entidad observada les informe de un cambio de estado a través de un evento que puede ser de su interés, por lo que los observadores se registran con la entidad observada.(27)

Cuando ocurre un evento, la entidad observada mira en su lista de observadores y notifica a aquellos que se registraron para recibir eventos de ese tipo. Los observadores dejaron instrucciones detalladas de cómo puede la entidad observada ponerse en contacto con ellos para recibir los eventos.

Este patrón se pone de manifiesto de forma general en toda la aplicación, con el uso de los signals y slots que propone la biblioteca Qt, logrando actualizar las vistas cuando cambia el estado en alguna entidad.

3.4 Pruebas

Las pruebas de software son un elemento crítico para la garantía de la calidad del software y representan una revisión final de las especificaciones, del diseño y de la codificación. El objetivo fundamental de las pruebas es descubrir diferentes clases de errores con la menor cantidad de tiempo y de esfuerzo. Aunque las pruebas no pueden asegurar la ausencia de defectos; sí pueden demostrar que existen defectos en el software.(16)

Estas actividades se planean con anticipación y se realizan de manera sistemática. Cuando se aplican pruebas a un software es necesario tener en cuenta el objetivo que se persigue, debido a que las pruebas son agrupadas por niveles que se encuentran en distintas etapas del proceso de desarrollo.(29)

A las funcionalidades implementadas se le realizaron pruebas unitarias de caja negra. Una prueba unitaria es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. Las pruebas de caja negra se basan en la especificación del programa o componente a ser probado para elaborar los casos de prueba. El componente se ve como una “Caja Negra” cuyo comportamiento sólo puede ser determinado estudiando sus entradas y las salidas obtenidas a partir de ellas. No obstante, como el estudio de todas las posibles entradas y salidas de un programa sería impracticable se selecciona un conjunto de ellas sobre las que se realizan las pruebas.(30)

A continuación se enuncian los distintos escenarios (EC) de prueba correspondientes a cada caso de uso.

3.4.1 Casos de prueba

Caso de prueba CU Salvar configuración en el servidor.

Escenario	Descripción	Dirección IP	Respuesta del sistema	Resultado de la prueba
EC 1.1 Salvar Configuración.	1 Seleccionar menú “Archivo”. 2 Seleccionar opción “Almacenar configuración en el servidor”. 3 Introducir dirección IP del	V	1.1 El sistema muestra la opción “Almacenar configuración al servidor.” 2.1 El sistema muestra una ventana para que se introduzca	El resultado de esta prueba fue satisfactorio.

	servidor correcta. 4 Presionar botón Aceptar.		la dirección IP del servidor. 4.1 El sistema valida que los parámetros de conexión al servidor sean correctos. 4.2 El sistema salva en el servidor la nueva configuración.	
EC 1.2 Selecciona la opción Cancelar.	1 Introducir dirección IP del servidor. 2 Presionar el botón Cancelar.	V	2.1 El sistema cierra la ventana de dialogo de dirección IP. 2.2 El sistema cancela el proceso de salvado de la configuración.	El resultado de esta prueba fue satisfactorio.
EC 1.3 Parámetros de conexión incorrectos.	1 Introducir dirección IP del servidor incorrecta. 2 Presionar el botón Aceptar.	I	2.1 El sistema muestra un mensaje de alerta especificando que los parámetros de conexión son incorrectos.	El resultado de esta prueba fue satisfactorio.

Caso de prueba CU Cargar configuración en el ambiente de edición.

Escenario	**Descripción**	**Dirección IP**	**Respuesta del sistema**	**Resultado de la prueba**
EC 1.1 Cargar configuración ambiente edición.	1 Seleccionar menú “Archivo”. 2 Seleccionar la opción “Descargar configuración del servidor”.	V	1.1 El sistema muestra la opción “Descargar configuración del servidor”. 2.1 El sistema	El resultado de esta prueba fue satisfactorio.

	3 Introducir dirección IP del servidor. 4 Presionar el botón Aceptar.		muestra una ventana para que se introduzca la dirección IP del servidor. 4.1 El sistema valida que los parámetros de conexión al servidor sean correctos. 4.2 Carga la configuración de cada una de las entidades.	
EC 1.2 Selecciona la opción Cancelar.	1 Introducir dirección IP del servidor. 2 Presionar el botón Cancelar.	V	2.1 El sistema cierra la ventana de dialogo de dirección IP. 2.2 El sistema cancela el proceso de cargar la configuración.	El resultado de esta prueba fue satisfactorio.
EC 1.3 Parámetros de conexión incorrectos.	1 Introducir dirección IP del servidor incorrecta. 2 Presionar el botón Aceptar.	I	2.1 El sistema muestra un mensaje de alerta especificando que los parámetros de conexión son incorrectos.	El resultado de esta prueba fue satisfactorio.

Caso de prueba CU Actualizar identificadores de los elementos.

Escenario	Descripción	Conexión con el servidor	Respuesta del sistema	Resultado de la prueba
EC 1.1 Actualizar identificadores.	1 Detectar que se ha salvado la configuración 2 Pedir actualizar los identificadores de los elementos.	V	2.1 El sistema verifica que exista conexión con el servidor. 2.2 El sistema obtiene los nuevos	El resultado de esta prueba fue satisfactorio.

			identificadores de los elementos. 2.3 El sistema actualiza los identificadores de las entidades.	
EC 1.2 No existe conexión con el servidor.		I	1.1 El sistema muestra un mensaje de alerta especificando que no existe conexión con el servidor.	El resultado de esta prueba fue satisfactorio.

Caso de prueba CU Registrar cambios en elementos.

SC 1 Adicionar elemento.

Escenario	Descripción	Datos elemento	Respuesta del sistema	Resultado de la prueba
EC 1.1 Adicionar elemento.	1 Seleccionar árbol de navegación. 2 Seleccionar tipo de elemento a adicionar. 3 Introducir datos de elemento a adicionar. 4 Presionar el botón Aceptar.	V	1.1 El sistema muestra la opción “Adicionar”. 2.1 El sistema muestra una ventana para especificar los datos del elemento a adicionar. 4.1 El sistema adiciona el nuevo elemento. 4.2 El sistema registra el cambio realizado.	El resultado de esta prueba fue satisfactorio.
EC 1.2 Selecciona la opción Cancelar.	1 Introducir datos de elemento a adicionar.	V	2.1 El sistema cierra la ventana para introducir los datos del	El resultado de esta prueba

	2 Presionar el botón Aceptar.		elemento a adicionar. 2.2 Cancela el proceso de adición del elemento.	fue satisfactorio.

SC 2 Modificar elemento

Escenario	Descripción	Datos elemento	Respuesta del sistema	Resultado de la prueba
EC 1.1 Modificar elemento.	1 Seleccionar elemento a modificar 2 Seleccionar opción “Modificar” 3 Modificar datos del elemento seleccionado. 4 Presionar el botón Aceptar.	V	1.1 El sistema muestra la opción “Modificar”. 2.1 El sistema muestra una ventana para modificar los datos del elemento seleccionado. 4.1 El sistema modifica los datos del elemento seleccionado. 4.2 El sistema registra el cambio realizado.	El resultado de esta prueba fue satisfactorio.
EC 1.2 Selecciona la opción Cancelar.	1 Modificar datos del elemento seleccionado. 2 Presionar el botón Aceptar.	V	2.1 El sistema cierra la ventana para modificar los datos del elemento seleccionado. 2.2 Cancela el proceso de modificación de los datos del elemento.	El resultado de esta prueba fue satisfactorio.

SC 3 Eliminar elemento.

Escenario	Descripción	Configuración del servidor	Respuesta del sistema	Resultado de la prueba

EC 1.Eliminar elemento.	1 Seleccionar elemento a eliminar 2 Seleccionar opción “Eliminar”	V	1.1 El sistema muestra la opción “Eliminar”. 2.1 El sistema elimina elemento seleccionado. 2.2 El sistema verifica que se haya cargado la configuración del servidor. 2.3 El sistema registra el cambio realizado.	El resultado de esta prueba fue satisfactorio.
EC 1.2 Selecciona la opción Cancelar.	1 Cargar configuración del servidor incorrecta.	I	1.1 El sistema no registra el cambio realizado.	El resultado de esta prueba fue satisfactorio.

3.5 Resultados de las pruebas

Las pruebas diseñadas se realizaron a lo largo de todo el ciclo de desarrollo de la aplicación. Como resultado, se obtuvo una aplicación con alta funcionalidad y 100% de cumplimiento de los objetivos trazados. La ejecución de estas pruebas permitió localizar errores y que estos fueran corregidos mucho antes de dar los toques finales a la aplicación. De esta forma, se obtiene una aplicación cuyas funcionalidades han sido comprobadas con anterioridad, lo que garantiza la confiabilidad en la aplicación y la obtención de resultados satisfactorios durante su uso.

Conclusiones parciales

En el desarrollo del capítulo se especificaron los resultados obtenidos. Se presentó la distribución física del sistema y sus componentes mediante el diagrama de componentes y el diagrama de despliegue; estos permitieron un mejor entendimiento de la distribución física y lógica del sistema. Se explicaron las funciones que se consideraron críticas para el desarrollo del mecanismo de configuración en caliente.

Finalmente, se valida el mecanismo desarrollado mediante pruebas de caja negra, realizando casos de prueba a todas las funcionalidades. Este proceso permitió detectar las no conformidades que éste presentaba para darle solución a las mismas.

Conclusiones generales

Para la implementación y correcto funcionamiento de aplicaciones de automatización y control, el mecanismo de configuración en caliente es uno de los requisitos críticos que debe poseer un sistema de este tipo, constituye una técnica de vital importancia para garantizar la integridad de los datos que se manejan y la estabilidad funcional del mismo. Durante el desarrollo de la investigación se cumplieron los objetivos propuestos, estableciéndose mecanismos de carga y salva de la configuración en sistemas SCADA, concluyéndose que:

- ✓ Con el desarrollo de un mecanismo capaz de realizar la configuración en caliente en el módulo HMI se evitó la pérdida de visualización de la información debido al reinicio de los servicios del SCADA UX.
- ✓ Mediante varias iteraciones de pruebas al software y luego de solventar las no conformidades encontradas se pudo satisfacer correctamente los requisitos funcionales planteados para el mecanismo.
- ✓ Con el resultado del presente trabajo se da cumplimento a los requisitos planteados por el proyecto SCADA UX correspondientes a la configuración en caliente del módulo HMI.

Recomendaciones

Con el objetivo de mejorar las funcionalidades implementadas se plantean las siguientes recomendaciones:

- ✓ Aplicar técnicas de optimización de código a las nuevas funcionalidades con el objetivo de mantener y mejorar el código desarrollado.
- ✓ Incorporar mecanismos para la resolución de conflictos, que puede presentar en el proceso de salvar la configuración desde dos clientes HMI.

Bibliografía

1. SÁNCHEZ ELÍAS, L. *Mecanismo de disponibilidad para el Configurador SCADA UX.* 2011.

2. HONEYWELL. *PlantScape* Disponible en: http://www.honeywellsp.com/hw_productos_servicios/hw_industrial/Hw_Control_Automatizacion_Industrial.htm.

3. *Argos Scada* Disponible en: http://www.cintal.com.ve/argos/.

4. WONDERWARE. *Intouch* Disponible en: http://www.wonderware.com.

5. *Sistemas SCADA* Disponible en: http://www.automatas.org/redes/scadas.htm.

6. *Sistemas HMI.* Disponible en: http://www.generatecnologias.es/sistemas_hmi.html.

7. *Eclipse en Ecured* Disponible en: http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado.

8. *C++ en EcuRed* Disponible en: http://www.ecured.cu/index.php/C%2B%2B.

9. *Curso Visual* Disponible en: http://www.monografias.com/trabajos5/visualcurso/visualcurso.shtml.

10. *Lenguajes Orientados a Objetos.* Disponible en: http://www.ciao.es/Opiniones/Lenguajes_Orientado_a_Objetos_C__72248.

11. *Visual Paradigm en Ecured* Disponible en: http://www.ecured.cu/index.php/Visual_Paradigm.

12. *13 razones para utilizar qt.* Disponible en: http://www.celularis.com/software/13-razones-para-utilizar-qt/.

13. *Qt en Ecured.* Disponible en: http://www.ecured.cu/index.php/Qt.

14. NOKIA. *Qt* Disponible en: http://www.qt.nokia.com/.

15. JACOBSON, I., BOOCH, GRADY Y RUMBAUGH, JAMES. *El Proceso Unificado de Desarrollo de Software.* 2000.

16. PRESSMAN, R. S. *Ingeniería de Software, un enfoque práctico.* Sexta edición ed.

17. *Desarrollando aplicaciones informáticas con el Proceso de Desarrollo Unificado (RUP).* Disponible en: http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm.

18. MABLE. *Metodologías de desarrollo.* Disponible en: http://www.marblestation.com/?p=644.

19. *Modelo de dominio en Ecured.* Disponible en: http://www.ecured.cu/index.php/Modelo_de_dominio.

20. SEHARA DRIGGS, Y. L. *Implementación de un modelo para la configuración de un sistema SCADA.* 2008.

21. *Diagrama de casos de uso del sistema.* Disponible en: http://www.slideshare.net/javi2401/diagramas-de-casos-de-uso-presentation.

22. *Diagrama de Interacción.* Disponible en: http://users.dcc.uchile.cl/~psalinas/uml/interaccion.html.

23. *Diagrama de despliegue en Ecured.* Disponible en: http://www.ecured.cu/index.php/Diagrama_de_despliegue.

24. *Arquitectura de software: Diagrama de componentes.* Disponible en: http://es.scribd.com/doc/7884665/Arquitectura-de-Software-II-Diagrama-de-Componentes-y-Despliegue.

25. *Patrones de diseño en Ecured.* Disponible en: http://www.ecured.cu/index.php/Patrones_de_dise%C3%B1o_de_software.

26. *Patrones GRASP en Ecured.* Disponible en: http://www.ecured.cu/index.php/Patrones_de_Asignaci%C3%B3n_de_Responsabilidades.

27. LARMAN, C. *UML Y PATRONES: Introducción al análisis y diseño orientado a objetos.* 2004. vol. Tomo 1, pag. 185 - 215 p.

28. RIBEAUX CEDEÑO, L. *Cliente de Grabación del sistema de Video Vigilancia, Suria Vision.* 2012.

29. LEYVA ABRAHANTES, D. *Diseño e implementación de un módulo para la realización de presentaciones web reusables sobre Moodle.* 2011.

30. GONZÁLEZ TAMAYO, J. C. *Desarrollo de funcionalidades al módulo de procesamiento del Scada UX.* 2011.