Universidad de las Ciencias Informáticas

"Facultad 5"

Título: "Desarrollo de una aplicación web para la supervisión y control de los procesos automatizados por el SCADA cubano."

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

Autores: Yanay Wong Asencio.

Abigail Armero Moreno.

Tutor: Ing. Enerys Mesa Morales

Co-tutor: Ing. Ariel Guerra Garayta

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2010.

Yanay Wong Asencio

_____

Firma del autor

Abigail Armero Moreno

_____

Firma del autor

Enerys Mesa Morales

_____

Firma del tutor

Datos de Contacto

Nombre y apellidos: Enerys Mesa Morales
Institución: Universidad de las Ciencias Informáticas (UCI).
Título: Ingeniero en Ciencias Informáticas.
e-mail: emesa@uci.cu

Ingeniero en Ciencias Informáticas, en la Universidad de Ciencias Informáticas (UCI) en el 2009, Profesor de la UCI en adiestramiento.

Nombre y apellidos: Ariel Guerra Garayta
Institución: Universidad de las Ciencias Informáticas (UCI).
Título: Ingeniero en Ciencias Informáticas.
e-mail: agarayta@uci.cu

Ingeniero en Ciencias Informáticas, en la Universidad de Ciencias Informáticas (UCI) en el 2009, Profesor de la UCI en adiestramiento.

Dedicatoria

A la Revolución, por formarnos bajo los preceptos socialistas.

A nuestro Comandante en Jefe, Fidel Castro, máximo impulsor de la universidad del futuro.

A mi mamita del alma a ella que es la luz en mi camino.

A mi papito que quiero tanto por estar siempre apoyándome.

A mi hermano por ser mi segundo padre a ti te debo lo que soy.

A mis tías las mas grandiosas por ser siempre un ejemplo a seguir.

Yanay.

A Fidel Castro Ruz y a la Revolución.

A mis padres por ser siempre una guía en mi formación.

A mis hermanos que son mi vida.

Abigail.

Agradecimientos

A toda mi familia por ser siempre un apoyo para mí en especial a mi mama por ser la persona más maravillosa del mundo, muchas gracias por estar siempre ahí con tu sonrisa, palabra y ternura.

A mi hermano que significas tanto para mí, a ti te quiero agradecer que seas siempre mi motor impulsor desde chiquita ya siento que puedes estar orgulloso de mi.

A mi papito querido gracias por tus consejos y apoyo.

A mis tías Eva y Cary que han seguido muy de cerca mi carrera dándome siempre su amor y confianza.

A mi primita Lienny que es mi hermanita querida.

A todos mis amigos sin excluir ninguno que han sido siempre un apoyo en mi carrera, en especial a las chicas Oadis,Yanet,Misleidys,Maydelin,Yadira,Adria;también a mis chicos Alejandro, Roberto, Juan Carlos, El Chino muchas gracias por estar siempre en todos los momentos dando siempre una palabra o una sonrisa amiga.

A mi novio Rachi gracias por tu amor en todos los momentos y por darme siempre esa palabra de seguridad que hace que confié en que todo saldrá bien.

A todos los mis profesores de los que he aprendido tanto en todos estos años.

Al profesor Millet por confiar en mi desde el principio de la carrera siempre será mi inspirador.

A mi tutora y amiga Enerys por sus consejos y preocupación.

A mi gran compañero de tesis y amigo Abigail por tener siempre ese carácter positivo por tu dedicación y confianza que hizo que en todos estos momentos de trabajo juntos fuéramos un gran equipo optimista.

Yanay.

Agradecimientos

A mis padres les agradezco toda mi vida porque siempre han estado cerca de mí en lo buenos y en los malos momentos, constituyen mi principal guía y sin ellos no hubiera alcanzado mi sueño, les debo todo los logros que he alcanzado, por brindarme siempre su apoyo y confianza, enseñándome que sí se puede pese a las dificultades. A ustedes que representan mi más valioso tesoro, no sé que hubiera sido sin su unidad inquebrantable.

A mis hermanos les agradezco por todos los momentos agradables que pasamos juntos, su amor y confianza en mí han constituido mi principal arma de batalla ante situaciones adversas, son y serán siempre mi orgullo.

A mis abuelos y a mis tías y tíos por contribuir en gran medida a mi educación.

A mis primas y primos, a toda mi familia.

A todos mis profesores que de una manera u otra han contribuido en mi aprendizaje, a ellos les debo mis conocimientos.

A mis tutores por brindarme su consejo y ayuda en el desarrollo de este trabajo, les estoy muy agradecido.

A mi novia por estar siempre a mi lado apoyándome y por hacerme sentir tan especial.

A todos mis amigos, gracias por su amistad incondicional.

A mi súper compañera de tesis Yanay, mejor no la quiero, excelente amiga con su espíritu y optimismo en salir adelante ante cualquier situación.

Abigail.

Resumen

SCADA proviene del acrónimo Supervisory Control and Data Acquisition (*Control de Supervisión y Adquisición de Datos*), es un sistema que recopila datos de varios sensores en una fábrica. El proyecto SCADA, fue elaborado por el centro de informática industrial de la Universidad de la Ciencias Informáticas como resultado del convenio entre Cuba-Venezuela y su principal objetivo fue el despliegue en las empresas de PDVSA (*Petróleos de Venezuela*). Como consecuencia de las necesidades existentes en empresas cubanas se encuentra en desarrollo el SCADA cubano, un producto que incorpora características propias y adaptables al sistema industrial nacional, manteniendo requisitos y algunos casos de usos críticos de su precedente. El objetivo de este trabajo es desarrollar una aplicación web como parte del módulo HMI (*Human Machine Interface*, por sus siglas en inglés) que permita la configuración, supervisión y control de los procesos automatizados mediante la web, solución antes vista en aplicaciones de escritorio.

Palabras claves: supervisar, controlar, SCADA, HMI

Tabla de Contenidos

Introducción

La informática industrial se enfoca en la aplicación de métodos y técnicas de las ciencias informáticas a los distintos ámbitos de la industria mediante el tratamiento automático de la información proveniente de los procesos industriales utilizando ordenadores o computadoras que hacen más eficiente y precisa la correcta toma de decisiones. En la industria actual han proliferado los llamados sistemas SCADA como su nombre indica, dichos sistemas cuentan con las funcionalidades de control, supervisión y adquisición de datos. Recopilan datos de sensores en una fábrica, permiten comunicarse mediante señales de entrada y salida con los dispositivos de campo (sensores, actuadores, controladores) para controlar el proceso desde la pantalla del ordenador, que es configurada por el usuario y puede ser modificada con facilidad.(2)

En el centro de informática industrial de la Universidad de la Ciencias Informáticas fue elaborado un sistema SCADA mediante un convenio entre Cuba y la República de Venezuela con el objetivo principal de desplegarlo en las empresas de Petróleos de Venezuela S.A (PDVSA). Como consecuencia de las necesidades existentes en empresas cubanas se encuentra en desarrollo el SCADA cubano, un producto que incorpora características propias y adaptables al sistema industrial nacional, manteniendo requisitos y algunos casos de usos críticos de su precedente, soportado sobre tecnologías avanzadas en software libre, posibilitando así la soberanía tecnológica.

La mayoría de los sistemas SCADA, debido a su magnitud y complejidad están formados por diferentes módulos o subsistemas funcionales, entre ellos se pueden encontrar el procesamiento de datos, manejadores e Interfaz hombre maquina (*HMI*, por sus siglas en inglés). La directriz de dicho módulo es desarrollar aplicaciones de escritorio debido a la importancia que se le atribuye a supervisar y controlar los procesos industriales en tiempo real. Sin embargo tal tendencia trae consigo que el monitoreo de la información por parte de los directivos requiera de su presencia en la planta. Tales dirigentes, que conforman la gerencia de estas empresas toman decisiones no críticas a gran escala donde el tiempo real no es tan necesario como la disponibilidad de la información.

Mediante el uso de la tecnología web, aplicaciones de este tipo pueden ser accesibles desde cualquier punto de la red a través de una computadora o dispositivos que soporten esta tecnología, lo que contribuye a un desarrollo personalizado de manera profesional y a la automatización de procesos mediante esta vía.

De la situación anterior se deriva la necesidad de contar con una herramienta informática que mediante la web permita conocer el estado de los procesos industriales de una planta desde cualquier localización, y así supervisar y controlar procesos de interés para facilitar la toma de decisiones a nivel administrativo.

Para dar respuesta a esta **situación problemática** se considera el siguiente **problema científico** ¿Cómo supervisar y controlar los procesos automatizados por los sistemas SCADA's mediante la web? Según el problema científico expuesto anteriormente se plantea como **objeto de estudio** los procesos de supervisión y control web.

Para dar solución al problema planteado se ha trazado como **objetivo general de la investigación:** Desarrollar una aplicación web para la supervisión y el control de los procesos automatizados por el SCADA cubano**.** Y como **campo de acción** los procesos de supervisión y control web para el SCADA cubano.

Para dar cumplimiento a estos objetivos se definieron las siguientes **tareas de investigación:**

- Revisión bibliográfica de las aplicaciones web para sistemas SCADA para la confección del estado del arte.
- Definición de las tecnologías existentes para el desarrollo de un HMI-Web en sistemas SCADA.
- Conceptualización de las necesidades en las aplicaciones web de SCADA para el levantamiento y especificación de requisitos.
- Puntualización del diseño del software para la definición de la arquitectura.
- Implementación de los elementos necesarios para cumplir con las funcionalidades identificadas.
- Valoración de los resultados obtenidos para asegurar la calidad de la herramienta.

La idea a defender es la siguiente: Desarrollando una aplicación web para la supervisión y control de los

procesos automatizados por el SCADA cubano, se logrará la supervisión y control de procesos automatizados a través de la web. Para el cumplimiento de estos objetivos se llevan a cabo varios **métodos y técnicas en la búsqueda y procesamiento de la información** como son:

A nivel teórico:

- **Método analítico-sintético:** Para el estudio de los conceptos empleados en los sistemas SCADA Web, analizando todos los documentos elaborados por desarrolladores, para la extracción de los elementos más importantes.

- **Análisis histórico–lógico:** Para la comprensión de los antecedentes y las tendencias actuales referidas a la evolución en el mundo de los sistemas SCADA Web.

A nivel empírico:

- **Experimento:** Elaboración de una aplicación web HMI para la supervisión y control de los procesos automatizados por el SCADA CUBANO.

Capítulo 1: Conceptos, herramientas y tecnologías para la supervisión y el control en aplicaciones web.

1.1. Introducción

El presente capítulo proyectará un análisis de los sistemas SCADA's, reflejándose sus principales características y su comportamiento mediante la web. Además se realizará un estudio de las tecnologías, herramientas y tendencias más utilizadas actualmente en el desarrollo de aplicaciones web para procesos industriales, así como las ventajas y desventajas de los procesos automatizados.

1.2. Características de los sistemas SCADA's.

El objetivo principal de la automatización es disminuir la intervención del operador humano en la ejecución de los procesos de producción de las empresas. Con vista a cumplir esa exigente meta se han desarrollado en los últimos años algunos sistemas denominados SCADA (2).

Los sistemas SCADA son aplicaciones de control de producción que se comunican con los dispositivos de campo y controlan el proceso de forma automática desde la pantalla del ordenador, proporcionando de esta forma información del proceso a diversos usuarios entre los que se encuentran los operadores, supervisores, mantenedores entre otros. Estos sistemas SCADA´s cuentan con una interfaz Hombre - Máquina o HMI que presenta los datos a un operador (humano) y a través del cual se controla el proceso.

Los sistemas HMI son pensados como una "ventana de un proceso". Esta ventana puede estar en dispositivos especiales como paneles de operador o en un ordenador. Los sistemas HMI en ordenadores se los conoce también como software HMI o de monitorización y control de supervisión. Las señales del proceso son conducidas al HMI por medio de dispositivos como tarjetas de entrada/salida en el ordenador, PLC's (Controladores lógicos programables), PACs (Controlador de automatización programable), RTU (Unidades remotas de I/O) o DRIVER's (Variadores de velocidad de motores). Todos estos dispositivos deben tener una comunicación que entienda el HMI. (3) Los sistemas de interfaz entre usuario y planta, están basados en paneles de control e indicadores lumínicos, instrumentos de medidas y botones, que están representados en la pantalla de un ordenador.

Otras de las características son la adquisición y el almacenamiento de información en bases de datos de tiempo real, representación gráfica de los procesos de manera animada, creación de gráficos de tendencia

a partir de los valores de las variables en el tiempo, conectividad con otros sistemas (locales y compartidos) y explotación de los datos para la gestión de la calidad, control estadístico, administración y generación de informes. Los sistemas SCADA´s son de arquitectura abierta, capaces de crecer o adaptarse según las características cambiantes de las empresas, que se relacionan con total facilidad y de forma transparente al usuario con los equipos de campo y el resto de los ordenadores involucrados en el proceso, programas en general fáciles de instalar y configurar, con pocas exigencias de hardware y que muestren interfaces amigables a los operadores y demás interesados. (4)

1.2.1. Funcionalidades de los sistemas SCADA

Resulta de interés presentar las principales funcionalidades que se pueden observar en un sistema SCADA.

Un SCADA debe cumplir tres funciones principales:

- Adquisición de datos para recoger, procesar y almacenar la información recibida.
- Supervisión para observar desde un monitor la evolución de las variables de control.
- Control para modificar la evolución del proceso, actuando sobre los reguladores autónomos básicos (consignas, alarmas, menús, etc.) o directamente sobre el proceso mediante las salidas conectadas.

En este tipo de sistemas interactúan ordenadores, unidades remotas, sistemas de comunicación, que efectúan las tareas de supervisión, gestión de los datos y el control total de los procesos entre otras tareas entre las que se encuentran:

- Adquirir información de campo.
- Procesar información de campo
- Generar alarmas.
- Almacenar la información adquirida por largos períodos de tiempo.
- Generar informes.
- Cambiar estados de los dispositivos de campo.

- Permitir la interacción a través de interfaces gráficas. (4)

Partiendo de lo anteriormente dicho se pueden definir un conjunto de bloques funcionales que aparecen en un SCADA:

- Comunicación con el Campo.
- Procesamiento de la información.
- Almacenamiento de la información.
- Interacción con el Usuario. (4)

1.2.2. Módulos de los sistemas SCADA.

- **Manejadores**

  Aseguran mediante una interfaz genérica la comunicación del sistema de supervisión y control con los distintos dispositivos que existen en el campo, ya sean autómatas, PLC[1], sensores inteligentes, etc.

- **Núcleo de procesamiento de datos**

  Representa el núcleo principal del procesamiento de los datos, es el encargado del procesamiento y análisis de la información recogida del campo a través de los manejadores. Una vez procesada esta información, es enviada al módulo que la requiera.

- **Base de datos históricos**

  Aquí se implementa el mecanismo encargado del almacenamiento de la información recibida desde el campo, así como la sucesión de alarmas y eventos generados. La información almacenada es utilizada por varias aplicaciones del sistema.

- **Middleware**

  El Middleware es la capa de software, que se encarga de la comunicación entre los diferentes procesos distribuidos de medio y alto nivel, que forman parte del sistema SCADA y el principal

[1] (Controladores lógicos programables)

elemento que caracteriza su complejidad es la gestión de las comunicaciones.

- **Ambiente de configuración**

  Esta aplicación permite configurar varios procesos o partes de ellos, aquí se definen y gestionan las variables, la configuración de los manejadores, los comandos, las alarmas y variadas opciones adicionales. Este ambiente funciona como una aplicación de diseño tradicional, con la peculiaridad que los sinópticos se confeccionan a partir de objetos y primitivas básicas predefinidas, que se pueden agrupar, combinar, transformar, importar y exportar entre otras.

- **Ambiente de ejecución (run-time)**

  Aplicación encargada de la visualización, la cual permite representar gráficamente de manera fidedigna los procesos operacionales con los que el usuario interactúa. A través de esta aplicación se pueden visualizar las condiciones operacionales, valores de variables, navegar entre despliegues, visualizar alarmas, enviar comandos, entre otros. Estas pueden estar determinadas por la visualización de escritorio o mediante la web. (4)

1.3. Características de las aplicaciones web.

Con el surgimiento y desarrollo de las aplicaciones web se han abierto las posibilidades en cuanto al acceso y uso de información desde lugares geográficos distantes del mundo. Con los avances en esta tecnología cada vez se demandan aplicaciones más rápidas, ligeras y robustas.

En la actualidad, el acelerado crecimiento de los sistemas de comunicación han hecho de Internet una tecnología portadora de una gran variedad de servicios, entre los que se destacan el de correo electrónico, transferencia de ficheros, servicio de información, servicios web, de televisión y telefonía. También ha permitido que el acceso a estos servicios pueda realizarse desde gran variedad de dispositivos entre los que se encuentran teléfonos móviles, los PDA (Personal Digital Assistant, de sus siglas en ingles), las computadoras, entre otros.

Mediante las aplicaciones web es posible conocer un evento distante de forma rápida. Facilita el uso de

recursos, los servicios y brinda accesibilidad independientemente del tipo de hardware, software, infraestructura de red, idioma, cultura y localización geográfica. (5) Estos avances han permitido que la internet sea usada en una amplia variedad de aplicaciones web y complejos sistemas que antes solo eran posibles con soluciones cliente/servidor de escritorio. La mayoría de los sistemas de supervisión y control de procesos a distancias basados en internet han enfocado su diseño hacia aplicaciones web. ‘

1.4. Selección de las tecnologías en el HMI web

Los módulos de HMI web para sistemas SCADA's está compuesto por un un cliente web y un servidor HMI web, que se comunican haciendo uso del protocolo de comunicación HTTP (HyperText Transfer Protocol, por sus siglas en ingles). Éste módulo está orientado a variables, el valor atómico de las mismas es donde se va a guardar la información de supervisión en el SCADA , Se utiliza la biblioteca QT (6) para la visualización gráfica y el framework EXTJS (7) en la visualización web. Estas permiten a los usuarios autorizados estar en contacto directo con el sistema, realizar la supervisión, y el control de los procesos en general.

- **Cliente web**

El cliente web es una aplicación web que funciona sobre un navegador web estándar, recibe y muestra los datos del servidor HMI web. Mediante la aplicación cliente los usuarios interactúan con el proceso. Una característica especial del cliente web de HMI es que es un cliente ligero, que no requiere ningún software adicional para realizar las funciones convencionales del SCADA. Permite visualizar la información proporcionada por el servidor HMI web. La comunicación con el servidor HMI web se realiza a través del protocolo HTTP, el cual permite la transferencia de archivos hipertexto.

- **Protocolo de comunicación HTTP**

El Protocolo de Transferencia de HiperTexto (*Hypertext Transfer Protocol*) es un protocolo cliente-servidor que manipula los intercambios de información entre los clientes web y los servidores HTTP. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. Un cliente establece una conexión con un servidor y envía un mensaje con los datos de la solicitud. El servidor responde con un mensaje similar, que contiene el estado de la operación y su posible resultado.

Una vez que se establece la conexión, el protocolo TCP (*Transmission Control Protocol*, de sus siglas en inglés) se encarga de mantener la comunicación y garantizar un intercambio de datos libre de errores. Una transacción HTTP está formada por un encabezado seguido, opcionalmente, por el dato.

El encabezado especificará la acción requerida del servidor, el tipo de dato retornado, o el código de estado. El uso de campos de encabezados enviados en las transacciones HTTP le da gran flexibilidad al protocolo. Estos campos permiten que se envíe información descriptiva en la transacción, permitiendo así la autenticación, cifrado e identificación de usuario. Un encabezado es un bloque de datos que precede a la información de la transacción, por lo que muchas veces se hace referencia a él como metadato.

HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores. El desarrollo de aplicaciones web necesita frecuentemente mantener estado. Para esto se usan las cookies, que es información que un servidor puede almacenar en el sistema cliente. Esto le permite a las aplicaciones web instituir la noción de "sesión", y también permite rastrear usuarios ya que las cookies pueden guardarse en el cliente por tiempo indeterminado.

En general, una aplicación que inicia una comunicación con otra se la califica como cliente. Los usuarios finales invocan aplicaciones cliente cuando utilizan un servicio de red. Cada vez que se ejecuta una aplicación cliente, esta contacta con el servidor, le envía una solicitud de servicio y espera la respuesta o resultados del servicio. El proceso cliente es el encargado de llevar a cabo la interacción con el usuario y de mostrar los resultados de las peticiones de servicio. En la mayoría de las ocasiones los clientes son más fáciles de diseñar que los servidores, y no suelen precisar privilegios especiales del sistema para poder funcionar.

- **Servidor Web**

Un servidor es un programa que espera peticiones de servicio por parte de un cliente. Recibe la petición del cliente, procesa la información, ejecuta el servicio solicitado y retorna los resultados al cliente. No existe una interacción directa entre el usuario y el servidor, de esto ya se encarga la aplicación cliente.

El servidor HMI web gestiona la información que le llega desde los dispositivos de campo y crea las representaciones de los sinópticos que luego se visualizará en el cliente Web. El servidor es el encargado actualizar los componentes gráficos de la aplicación, atendiendo a las peticiones de los usuarios y las actualizaciones de los valores de las variables de los dispositivos del campo.

Otra funcionalidad del servidor HMI web es que provee los datos del campo y los valores que se almacenan en base de datos histórica de forma inmediata. Gestionan el sistema de comunicación dándole al operador control total mediante un navegador web. (4)

1.5. Ventajas y desventajas de un HMI web.

Ventajas:

- No requiere instalación, pues usa tecnología web, lo cual nos permite el aprovechamiento de todas las características del Internet. Son fáciles de usar, los usuarios con conocimientos básicos sobre el sistema a operar puede manipular la aplicación.
- Alta disponibilidad, ya que puede realizar consultas desde cualquier región geográfica donde tenga acceso a Internet y a cualquier hora.
- Datos centralizados y fácil integración de datos de múltiples fuentes.
- Se obtiene una reducción de costos, puesto que se racionaliza el trabajo, se reduce el tiempo y dinero dedicado al mantenimiento.
- Menos requerimientos de memoria: Las demandas de memoria RAM (Random Access Memory) por parte del usuario final son más razonables que en los programas instalados localmente. (4)

Desventajas:

- Acceso limitado, la necesidad de conexión permanente y rápida a Internet hacen que el acceso a estas aplicaciones no esté al alcance de todos.
- La interactividad no se produce en tiempo real, en las aplicaciones web cada acción del usuario conlleva un tiempo de espera excesivo hasta que se obtiene la reacción del sistema.
- Elementos de interacción muy limitados. En comparación con el software de escritorio, las posibilidades de interacción con el usuario que ofrecen las aplicaciones web (mediante

formularios principalmente) son muy escasas, aunque se ha avanzado mucho con la introducción del AJAX (Asynchronous Javascript And XML) en la construcción de páginas web.

- Diferencias de presentación entre plataformas y navegadores. La falta de estándares ampliamente soportados dificulta el desarrollo de las aplicaciones. (4)

1.6. HMI web existentes en mundo

En la actualidad existen gran variedad de sistemas SCADA's que cuentan con una Interfaz Hombre – Máquina pero no todos incluyen una extensión para la supervisión y control de los procesos a través de la web. Entre los sistemas que podemos encontrar que cumplen estas características, todos tienen solución con tecnologías bajo licencia propietaria, entre ellos se encuentran:

- **WebHMI:** Es un poderoso complemento para cualquier sistema de Genesis32 (8), proporcionando conectividad de Internet remoto o de Intranet para el sistema de la empresa. Usando nada más que Internet Explorer de Microsoft, un PC remoto al instante puede navegar visualizar la información de cualquier producto de GENESIS32 ICONICS en la red. Basado en la arquitectura DNA de Microsoft, WebHMI ofrece automáticamente los "plug-ins" necesarios para que la estación del navegador pueda ser usado como parte de la solución de una red global. No sólo ver, sino también tomar el control del operador muestra el tiempo real con animación, tendencias de los datos, informes y alarmas. Se puede integrar la aplicación HMI en los navegadores de Internet tradicionales para realizar el seguimiento a distancia y bajo coste de la información clave de fabricación. WebHMI se completa con una corbata de servidor de seguridad en la entrada de datos y la interacción en tiempo real con su aplicación se controla todo el sistema.
- **Control System Works:** Es un marco basado en web para la construcción de HMI / SCADA soluciones de control de procesos utilizando Microsoft.NET y Microsoft Silverlight ofrece sistema abierto con ilimitadas posibilidades de personalización; potentes gráficos contemporáneos; funciona a través de barreras de red y cortafuegos; cliente de administración cero; ricas y transparentes posibilidades de configuración, alta disponibilidad. La aplicación de prueba de CS Works utiliza OPC. (9)

- **HMI 500: La** aplicación HMI 500 corresponde a la más reciente solución de EFACEC para la implementación de la Interfaz Humana Máquina de sistemas SCADA destinada con propósito para gestionar localmente, vía web, complejos sistemas distribuidos de automatización, supervisión, control y protección. La aplicación es del tipo Web server y puede ser utilizada en diferentes tipos de plataforma de hardware, funcionando bajo el sistema operativo WINDOWS XP. (10)

1.7. Herramientas y tecnologías a utilizar:

1.7.1. Framework en aplicaciones web vs. sitios web

Suele confundirse los conceptos de aplicación web con sitio web, lo cual lleva a la eterna discusión sobre cuál es el mejor biblioteca independientemente de la eficiencia de los mismos. Una aplicación web intenta portar las clásicas aplicaciones de escritorio hacia entornos web, que son utilizadas a través de los distintos navegadores, agregando portabilidad y capacidad de acceder desde diferentes dispositivos.

Un sitio web es un portal, una página, un grupo de páginas, una serie de documentos dinámicos o no. En ambos casos se puede mejorar la experiencia del usuario utilizando javascript y en efecto, frameworks. Aparte de la eficiencia, simplicidad, rendimiento, documentación, forma de uso, entre otros factores más, la finalidad de lo que se quiere crear con estos frameworks es uno de los más importantes a la hora de elegir entre uno u otro.

Otro factor importante que se debe considerar es la curva de aprendizaje y la facilidad de uso cuando se crean proyectos donde intervienen más de una persona. Una variable a considerar es la buena práctica y los patrones de programación aplicados.

1.7.2. Características de los frameworks:

Un framework es una abstracción de código común que provee funcionalidades genéricas que pueden ser utilizadas para desarrollar aplicaciones de manera rápida**,** fácil**,** modular y sencilla**,** ahorrando tiempo y esfuerzo. Entonces, un framework es concreto y también “incompleto”. Concreto porque es, desde un

punto de vista simple, un conjunto de componentes; incompleto, porque por sí mismos no pueden ser utilizados, ya que guían a la solución de problemas de programación recurrentes, por lo general, no son la solución específica completa. (11)

Entre los diferentes framewoks que existen se destacan por su facilidad y eficiencia en la implementación los de javascript pues permiten la reutilización de código, ofrecen animación, efectos, movimiento y formas dinámicas.

En su mayoría, los frameworks de javascript proveen componentes para:

- **Compatibilidad**. Agregan la posibilidad de escribir código javascript totalmente compatible con todos los navegadores y motores Javascript más utilizados. Esto aumenta la portabilidad y eliminan la incompatibilidad entre navegadores y sus motores intérpretes javascript.
- **Comunicación asíncrona** Ajax (Asynchronous Javascript And XML*)*. Usando este acercamiento, es fácil utilizar XMLHttpRequest para manejar y manipular los datos en los elementos de un sitio bien, aumentando la interactividad y experiencia del usuario.
- **DOM**. Maximizan la capacidad de agregar, editar, cambiar, eliminar elementos de manera dinámica agregando librerías que facilitan usar DOM (Document Object Model).
- **Validación de Formularios**. Permiten de una manera relativamente fácil validar campos dentro de uno o varios formularios. Esto, desde el punto de vista del desarrollador, simplifica y reduce el código para procesar dichos formularios, ya que los datos llegan previamente validados, reduciendo los errores de tipos de datos.
- **Efectos visuales**. Utilizando la manipulación de los elementos, se pueden crear efectos visuales y animaciones. Entre los efectos se encuentran: Aparecer y desaparecer, redimensionamiento, mover, aparecer y desaparecer, entre otros.
- **Almacenamiento Client-side**[2]. En adición provee funciones para leer y escribir cookies. También proveen una abstracción de almacenamiento que permite a las aplicaciones web guardar datos del lado del cliente, persistente y de manera segura.
- **Manejo JSON (***JavaScript Object Notation***)**. Incrementa al máximo el manejo de datos, que pueden

[2] Aplicación en Cliente.

ser utilizados para presentar informaciones de manera dinámica y en tiempo de ejecución.

- **Manejo de Eventos**. Esta característica agregada, permite reaccionar de una manera u otra dependiendo de las acciones del usuario.
- **Recibidores de Datos**. Permiten utilizar diferentes formatos de datos como XML, HTML, Texto, JSON entre otros.
- "**Arrastra y Suelta**". Mejor conocido como Drag and Drop. Es una funcionalidad que brinda la posibilidad de arrastrar elementos dentro de una misma página que interactúe con el resto de los elementos. (11)

Cada framework tiene características que los diferencian porque cada uno de ellos cumple con funcionalidades en dependencia del entorno en que se desarrollen, algunos de ellos son más rápidos y eficientes que otros en dependencia de la aplicación que se esté realizando.

1.7.3. Biblioteca EXTJS:

Es una biblioteca de Javascript para la creación de aplicaciones enriquecidas del lado del cliente. EXTJS (7) realmente brilla en la fabricación de aplicaciones web. Ofrece una gran cantidad de widgets[3] (12) para crear interfaces de usuario complejas. Sus características principales son: gran desempeño, componentes de interfaz de usuario personalizables, con buen diseño y documentación.

Originalmente fue construido como una extensión de YUI. Incluye intemporalidad con **JQuery (**biblioteca o framework de Javascript**)** (13) y **Prototype (**biblioteca o framework de Javascript**)**. (14) Posee controles para campos de textos, incluyendo áreas de texto. Controladores selectores de fecha, campos numéricos, para radiobox y checkbox.

Contiene además componentes para crear y manipular dataGrids donde goza de cierta ventaja sobre otros frameworks. Es posible crear "ventanas" con barras de herramientas y menús con estilo de aplicaciones de escritorio, diálogos modales y eventos. ExtJS puede ser adquirido bajo licencias libres y comerciales. La compañía detrás de este framework ofrece cursos de capacitación y un extenso soporte. (11)

[3] *componentes de una interfaz de usuario*

1.7.4. Ventajas de EXTJS

- Código reutilizable
- Independiente o adaptable a frameworks diferentes (prototype,jquery,YUI)
- Orientada a la programación de interfaces tipo desktop en el web.
- El API es homogeneizado independientemente del adaptador usado .Los controles siempre se verán igual
- Soporte comercial
- Una extensa comunidad de usuarios. (15)

1.7.5. Librería Dojo Toolkit:

Dojo Toolkit (15) es un framework Javascript que permite el desarrollo de aplicaciones web enriquecidas en el cliente y Ajax. Es popular porque está integrada en numerosos IDEs (*Un entorno de desarrollo integrado)* y otros frameworks para desarrollo de webs. Contiene un sistema de empaquetado inteligente, componentes de código en Javascript que puede ser utilizado para enriquecer sitios web con varias características que trabajan a través de la mayoría de los navegadores. Dojo resuelve asuntos de usabilidad comunes como ser la navegación y detección del browser, soportar cambios de URL en la barra de URLs (*Localizador de Recursos Uniforme*) para luego regresar a ellas.

Dojo es llamado “la navaja suiza del ejército de las bibliotecas del Javascript”. Proporciona una gama más amplia de opciones en una sola biblioteca y hace un trabajo muy bueno que apoya los nuevos y viejos buscadores.

Características:

- Múltiples puntos de entrada
- Independencia del interprete
- Unifica estándares de codificación:
- Construido en base a estándares de codificación de los proyectos

Determinar cuál framework es mejor es prácticamente imposible. Es importante destacar que estos

framework son simples herramientas que nos ayudan a realizar diferentes tareas de diferentes tipos, todos basados en el mismo lenguaje, javascript. Lo correcto es seleccionar uno u otro dependiendo la utilidad y la capacidad de saber cómo utilizarlo.

Dojo Toolkit, en su incorporación con Zend son bibliotecas que prometen mucho, pero no goza de la popularidad de EXTJS es la mejor opción, inclusive se podría crear un sistema operativo web utilizando este framework de forma relativamente fácil. Lo importante es saber utilizar las herramientas con las que se cuentan y sacarle el máximo provecho. (11)

1.7.6. Biblioteca Qt

Qt es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario. La biblioteca la desarrolla la que fue su creadora, la compañía noruega Trolltech, actualmente renombrada a Qt Software, y que desde junio de 2008 es propiedad de Nokia. Qt es utilizada en KDE, un entorno de escritorio para sistemas como GNU/Linux o FreeBSD, entre otros. Utiliza el lenguaje de programación C++ de forma nativa y además existen bindings para C, Python (PyQt), Java (Qt Jambi), Perl (PelrQt), entre otros. El API de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL, así como uso de XML y una multitud de otros para el manejo de ficheros, además de estructuras de datos tradicionales.

1.7.7. Metodología Utilizada.

Una metodología de desarrollo del software define quien está haciendo que, cuándo, y como alcanzar un objetivo específico, proporciona normas para el desarrollo eficiente del software con calidad al captar las mejores prácticas que el estado actual de la tecnología permite.

Open UP. Constituye un proceso unificado de desarrollo de corta duración, aplicado de manera iterativa e incremental dentro de un ciclo de vida estructurado en tres capas. Open Up adopta una pragmática y ágil filosofía centrado en el proceso colaborativo de desarrollo de software que puede ser aplicado a una gran variedad de proyectos en diferentes direcciones. En un proyecto organizado sobre Open Up, los esfuerzos personales se convierten en micro incrementos, estos proveen un ciclo de retroalimentación relativamente corto que permite flexibilidad y mejor adaptación a las decisiones tomadas dentro de cada iteración.

Esta metodología divide el proyecto en iteraciones que son planeadas sobre intervalos de tiempo definido en semanas, dichas iteraciones se centran dando cumplimiento a los objetivos definidos previamente en el plan para cada una, por parte del equipo de desarrollo donde cada ciclo iterativo debe concebir como resultado un demo o un ejecutable con funcionalidades específicas. Open Up en cada iteración del ciclo de vida, incrementa progresivamente los objetivos de las iteraciones anteriores añadiendo nuevas funcionalidades a las versiones estables del software que se tiene hasta el momento.

Dentro del ciclo de vida tiene 4 fases: Intercepción, Elaboración, Construcción y Transición. La metodología provee al cliente y al equipo con la visibilidad y los puntos de decisión durante el proyecto, deciden que hacer o no para el mejor aprovechamiento del tiempo y en el plan de proyecto se define el ciclo de vida de desarrollo donde el resultado final es una versión de la aplicación. Asumimos este metodología para el desarrollo de nuestro modelo porque nuestro polo de producción basa sus proyectos en ella y se han obtenido resultados relevantes en cuanto la agilización de producción de software para equipos de desarrollo y además la propuesta que tenemos a desarrollar no constituye un software de alta complejidad ni está determinado por procesos críticos donde se necesite poseer una amplia documentación sobre su ciclo de desarrollo. (17)

1.7.8. Herramienta Case seleccionada.

Visual Paradigm para UML como herramienta CASE

Visual Paradigm para UML es una herramienta Case de tipo cruzado de ciclo de vida, lo cual significa que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue, incluye también actividades como la gestión de proyectos y la estimación. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Incluye además importación desde Rational Rose, exportación/importación XMI, generador de informes, editor de figuras, integración con MS Visio, plug-in, integración IDE con Visual Studio, IntelliJ IDEA, Eclipse, NetBeans y otros. Apoya un conjunto de lenguajes tanto en la generación del código como en la Ingeniería Inversa por ejemplo Java, C

+ +, CORBA IDL, PHP, XML Schema, Ada y Python. (18)

1.8. Lenguajes utilizados para el desarrollo:

1.8.1. Lenguaje de Programación Interpretado: Javascript.

Javascript es un lenguaje de scripts desarrollado por Netscape para incrementar las funcionalidades del lenguaje HTML. Se utiliza embebido en el código HTML y XHTML, entre las etiquetas <script> y </script>. Dentro de sus características más importantes se pueden encontrar:

- Es un lenguaje interpretado, es decir, no requiere compilación. El navegador del usuario se encarga de interpretar las sentencias Javascript contenidas en una página HTML y ejecutarlas adecuadamente.
- Es un lenguaje orientado a eventos. Cuando un usuario presiona el cursor sobre un enlace o mueve el puntero sobre una imagen se produce un evento. Mediante este lenguaje se puede desarrollar Scripts que ejecuten acciones en respuesta a estos eventos.
- Es un lenguaje orientado a objetos. El modelo de objetos de Javascript está reducido y simplificado, pero incluye los elementos necesarios para que los Scripts puedan acceder a la información de una página y puedan actuar sobre la interfaz del navegador. (19)

1.8.2. Notación JSON

JSON (*JavaScript Object Notation*) es un formato sencillo para el intercambio de información. El formato JSON permite representar estructuras de datos y objetos (arrays asociativos) en forma de texto. La notación de objetos mediante JSON es una de las características principales de Javascript y es un mecanismo definido en los fundamentos básicos del lenguaje. En los últimos años, JSON se ha convertido en una alternativa al formato XML, ya que es más fácil de leer y escribir, además de ser mucho más conciso. No obstante, XML es superior técnicamente porque es un lenguaje de marcado, mientras que JSON es simplemente un formato para intercambiar datos. (20)

1.8.3. DOM (Document Object Model)

DOM es un conjunto de utilidades específicamente diseñadas para manipular documentos XML. Por extensión, DOM también se puede utilizar para manipular documentos XHTML y HTML. Técnicamente,

DOM es una API de funciones que se pueden utilizar para manipular las páginas XHTML de forma rápida y eficiente. Antes de poder utilizar sus funciones, DOM transforma internamente el archivo XML original en una estructura más fácil de manejar formada por una jerarquía de nodos. De esta forma, DOM transforma el código XML en una serie de nodos interconectados en forma de *árbol*. El árbol generado no sólo representa los contenidos del archivo original (mediante los nodos del árbol) sino que también representa sus relaciones (mediante las ramas del árbol que conectan los nodos). Aunque en ocasiones DOM se asocia con la programación web y con Javascript, la API de DOM es independiente de cualquier lenguaje de programación. De hecho, DOM está disponible en la mayoría de lenguajes de programación comúnmente empleados. (20)

1.8.4. AJAX (Javascript asíncrono + XML)

AJAX significa Asynchronous Javascript And XML, en español: Javascript y XML Asíncrono. Es una técnica de desarrollo que permite crear aplicaciones interactivas Hay que dejar bastante claro que AJAX no es una tecnología, sino que usa otras tecnologías. Para crear aplicaciones interactivas con AJAX, se combinan tres tecnologías, XHTML y CSS para mostrar los datos, Javascript y DOM para interactuar de forma dinámica con los datos mostrados o a mostrar y, XML y XSLT que manipula e intercambia los datos con un servidor web de manera de sincronizada. Además de estas tres tecnologías, AJAX también puede usar alguna que otra más, hasta puede usar texto llano. (7)

En el presente capítulo se analizaron conceptos y características referentes al SCADA, Se abordo las herramientas, tecnologías a utilizar, que posibilitarán el diseño y la implementación del sistema a desarrollar. Después de este estudio realizado se determinó usar la librería de Javascript EXT-JS por ser totalmente novedosa y con una arquitectura flexible que proporciona un desarrollo rápido en aplicaciones web. (20)

Capítulo 2: Características y descripción de la aplicación.

2.1. Introducción

En el presente capítulo se realiza una descripción del alcance que tendrá la propuesta de solución permitiendo tener una concepción general del sistema. Serán definidos los requisitos funcionales y no funcionales, el diagrama de casos de uso, secuencia y clases del diseño, permitiendo de esta forma modelar la vista interna de la aplicación.

2.2. Funcionamiento y principales características del SCADA cubano.

Un sistema SCADA es un sistema de arquitectura abierta, capaz de crecer o adaptarse según las necesidades cambiantes de una empresa. Un sistema de este tipo se debe comunicar con total facilidad y de forma transparente al usuario con el proceso industrial de la planta y el resto de la industria.

El SCADA existente en el centro de informática industrial (CEDIN) de la Universidad de la Ciencias Informáticas fue elaborado mediante un convenio entre Cuba-Venezuela. Actualmente a partir de las necesidades SCADA ´´Guardián del ALBA´´ se encuentra en desarrollo el SCADA cubano, un producto con características propias empresariales y adaptables al sistema industrial, manteniendo del anterior SCADA, los requisitos y algunos casos de uso críticos que pueden sufrir cambios en dependencia de la complejidad que conlleve.

2.2.1. Módulo HMI del SCADA cubano.

Desde de comienzos del curso 2009-2010 en el (CEDIN) se ha perfeccionado la línea HMI revaluando tecnologías gráficas existentes de software libre, lo que posibilita la incorporación de tecnologías avanzadas utilizándose el biblioteca Qt por las diferentes facilidades que brinda basado en plugin y como nueva herramienta para el desarrollo de la línea .

Posteriormente a esta nueva incorporación se comienzan a desarrollar herramientas para la configuración de sistemas SCADA's y proyectos para la visualización de escritorio y web lo que darían un valor agregado a las soluciones HMI. Los módulos del SCADA cubano han sufrido cambios en dependencia de las necesidades y quedan conformados con la siguiente estructura.

2.2.2. Módulos del SCADA cubano:

Dentro del estudio se ha reevaluado las funcionalidades de los módulos del SCADA:

- **Módulo de recolección:** Se determino la incorporación de un planificador de tareas (RTE).
- **Módulo de drivers:** Se le agrego una capa de la interfaz genérica para incluir los nuevos protocolos de comunicación**.**
- **Módulo de bases de datos históricos:** Extendió sus funcionalidades para la conexión con distintos servidores de bases de datos.
- **Módulo de comunicación middleware:** Se hizo un estudio de las tecnologías existentes y se desarrolló usando ICE pues permite incorporar mayor funcionalidad para el SCADA como puede ser la redundancia.
- **Módulo HMI:** Está compuesto por tres sub-módulos (ejecución, editor, reportes).

2.3. Conceptos en el dominio del problema.

2.3.1. Características de las alarmas del sistema de supervisión y control.

Sumario de alarmas.

El sumario de alarmas es una interfaz gráfica que concentra las condiciones de procesos críticas, medias y baja presentes en el sistema y cuyo objetivo primordial es guiar al operador a la detección del origen de la falla y supervisar la ejecución de las medidas de corrección automatizada o manual. Existe, a parte del sumario de alarmas reducido, un sumario de Alarmas expandido. (24)

Nivel de Severidad

La jerarquía de la alarma se define por su nivel de severidad, lo cual determina su tratamiento, según lo siguiente:

Severidad 1 (Alarmas Críticas)

Las alarmas también conocidas como “Alarmas Críticas”, son aquellas que requieren acción inmediata del operador:

1. Después que la acción correctiva automática ha sido activada.
2. Cuando la acción correctiva automática no está disponible.

En ambos casos el retardo o el fracaso de la acción correctiva pueden causar heridas al personal, producir daños ambientales o afectar las instalaciones, dañar y automática/o causar otras pérdidas sustanciales.

Los ejemplos de alarmas críticas son: fuego, gas, agujero de tubería, inundación de tanque y las alarmas de desviación de proceso que requieren la activación de proyectos de emergencia.

Severidad 2 Media

El objetivo de estas alarmas es advertir al operador que es necesario tomar medidas a fin de eliminar la desviación que activó la alarma y evitar la ejecución de alguna acción correctiva automática o manual. Un ejemplo típico de esto son las “pre alarmas”.

Severidad 3 (Advertencia de Alarmas)

Alarmas también conocidas como “Advertencia de alarmas”, usado para objetivos de información relacionados con desviaciones de proceso, equipo de proceso estado anormal, estado de cierre no incluido en prioridad 1 ó 2, alarmas de sistema, etc.

Diagrama de Estado para el tratamiento de alarmas. (24)

Según el Diagrama de Estado si se declara una condición de alarma por primera vez se visualiza en el Sumario con Parpadeo y con un sonido. Si ésta se reconoce por parte del operador y la causa que disparó la alarma desapareció, entonces desaparece la alarma del Sumario. Si por el contrario al ser reconocida la alarma no ha desaparecido la causa que la provocó ésta se silencia, pero permanece en el sumario sin parpadear, hasta tanto no desaparezca la causa, que es cuando desaparece entonces del sumario.

Representación de la Alarma en Sumario

Las exigencias siguientes serán explicadas para cada alarma según su asignación de severidad, clasificación y prioridad.

Severidad 1 o Alarmas Críticas

Importante: La guía de colores y posición en el sumario para una alarma crítica se muestra en la tabla 1.

Tabla 1 Guía de colores y posición en sumario, por omisión:

Estado de Alarma	Representación en Sumario	Posición en sumario
Alarma Activa No Reconocida	**Fondo Negro, Letras Rojas, intermitencia hacia Fondo Rojo, Letras Negras cada 0,5 Seg.**	Primeras filas de sumario ordenadas por su prioridad. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarma Activa Reconocida	**Fondo Negro, Letras Rojas, en modo estático**	Posterior al grupo de alarmas con Severidad baja no reconocidas. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarma No Activa NO reconocida	**Fondo Negro, Letras Rojas, intermitencia hacia Fondo Negro, Letras Verdes cada 0,5 Seg.**	Posterior al grupo de alarmas con Severidad baja reconocidas. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarmas No Activas reconocidas	**Desaparecen del sumario**	

Severidad 2 Media

El uso de un sumario de alarmas críticas en una Interfaz Hombre Máquina compartida con otras alarmas o despliegues. Las alarmas de severidad 2 pueden ser reconocidas individualmente o en grupo. La guía de colores y posición en el sumario para una alarma de severidad media se muestra en la tabla 2.

Tabla 2 Guía de colores y posición en sumario, por omisión:

Estado de Alarma	Representación en Sumario	Posición en sumario
Alarma Activa No Reconocida	**Fondo Negro, Letras Naranja, Fondo Naranja, Letras Negras cada 0,5 Seg.**	Posterior al Grupo de alarmas con severidad crítica no reconocida ordenadas por su prioridad. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarma Activa Reconocida	**Fondo Negro, Letras Naranja, en modo estático**	Posterior al grupo de alarmas con Severidad Alta reconocidas. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarma No Activa NO reconocida	**Fondo Negro, Letras Naranjas, intermitencia hacia Fondo Negro, Letras Verdes cada 0,5 Seg.**	Posterior al grupo de alarmas con Severidad crítica no activas no reconocidas. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarmas No Activas reconocidas	**Desaparecen del sumario**	

Severidad 3 Baja

Las alarmas de severidad 3 pueden ser reconocidas individualmente o en grupo.

La guía de colores y posición en el sumario para una alarma crítica se muestra en la tabla 3.

Tabla 3 Guía de colores y posición en sumario, por omisión:

Estado de Alarma	Representación en Sumario	Posición en sumario
Alarma Activa No Reconocida	**Fondo Negro, Letras Amarillas, intermitencia hacia Fondo Amarillo, Letras Negras cada 0,5 Seg.**	Posterior al Grupo de alarmas con severidad media no reconocida ordenadas por su prioridad. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarma Activa Reconocida	**Fondo Negro, Letras Amarillas, en modo estático**	Posterior al grupo de alarmas con Severidad media reconocidas. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarma No Activa NO reconocida	**Fondo Negro, Letras Rojas, intermitencia hacia Fondo Negro, Letras Verde cada 0,5 Seg.**	Posterior al grupo de alarmas con Severidad media no activas no reconocidas. Alarmas con la misma prioridad se comportan en modo LIFO dentro de su grupo.
Alarmas No Activas reconocidas	**Desaparecen del sumario**	

Ver anexo 1: **Representación del sumario de alarmas del ambiente de ejecución web.**

Acciones posibles dentro del sumario.

- ***Reconocer*****:** Ejecuta la acción de reconocimiento de la o las alarmas seleccionadas.

- ***Reconocer Todas**:* Ejecuta la acción de reconocimiento de todas las alarmas en el sumario.
- ***Silenciar Alarma:*** Ejecuta la acción de silenciado de todas las alarmas en el sumario, en caso de que se genere una nueva alarma posterior a la acción, la misma no será afectada.
- ***Silenciar Todas:*** Ejecuta la acción de silenciado de la ó las alarmas seleccionadas en el sumario, en caso de que se genere una nueva alarma posterior a la acción, la misma no será afectada.
- ***Filtros:*** El sumario permite el filtrado de la información, por los campos Fecha, Recurso, Grupo, Causa ó tipo, en cuyo caso el sumario congela la información, estos filtros aplican sólo al sumario en modo extendido. (24)

2.4. Especificación de los Requisitos de Software.

La especificación de requisitos comprende la descripción completa del comportamiento del sistema que se va a desarrollar constituyendo una base arquitectónica que con posterioridad se agrupará en casos de uso y describirá los procesos que realizará la futura aplicación que se propone. Además, de definir el funcionamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que este pueda poseer.

Requisitos Funcionales

Los **requisitos funcionales** son capacidades o condiciones que el sistema debe tener. Para la aplicación web que se desea desarrollar, ellos son:

RF1 El sistema debe permitir Autenticar Usuario.

RF1.1- El sistema debe permitir que el usuario pueda autenticarse con usuario y contraseña para acceder al mismo.

RF2 El sistema debe posibilitar Gestionar Alarmas.

RF2.1 -El sistema debe permitir silenciar una alarma.

RF2.2 -El sistema debe permitir silenciar todas las alarmas.

RF2.3 -El sistema debe posibilitar reconocer una alarma.

RF2.4 -El sistema debe posibilitar reconocer todas las alarmas.

RF2.5 -El sistema debe posibilitar filtrar las alarmas.

RF3 El sistema debe permitir Gestionar Comandos.

RF3.1- El sistema debe posibilitar enviar comandos de navegación entre despliegues.

RF3.2- El sistema debe posibilitar enviar comandos de actualización del despliegue.

RF3.3- El sistema debe posibilitar enviar comandos de actualización de un objeto dentro del despliegue.

Requisitos no funcionales:

Los requisitos **no funcionales** son propiedades o cualidades que el producto debe cumplir con el objetivo de lograr un producto atractivo, usable y confiable.

Para la aplicación que se desea desarrollar los requisitos no funcionales son los siguientes:

Usabilidad

CSUS1 La aplicación web deberá visualizarse con calidad en los principales navegadores existentes (Internet Explorer, Opera, Firefox y otros).

Fiabilidad

CSFI1 La aplicación deberá satisfacer los requisitos de alto rendimiento por parte del servidor HMI de forma tal que garantice su integridad y procesamiento.

Funcionamiento

CSFU1- Los tiempos de respuestas del sistema serán aproximadamente de 5 segundos.

Soportabilidad

CSSO1 El sistema debe ser de arquitectura abierta y distribuida.

Interfaces de usuarios

IU1 El Diseño del sistema debe estar enfocado a las características de las aplicaciones web, debe ser sencillo y funcional, fácil de navegar y de rápida adaptación para los usuarios.

Implementación

CSIM1 El sistema debe ejecutarse en diversas plataformas de hardware y software.

CSIM2 EL sistema se realizará utilizando el framework EXT JS, para la visualización web. Para la modelación de los diagramas UML y como metodología de desarrollo de software el OpenUp.

CSIM3 El sistema debe cumplir con los lineamientos necesarios para la producción de software libre y la comunidad de desarrollo y soporte.

Portabilidad:

CSPO1 El sistema debe ser multiplataforma, debe ejecutarse de manera óptima sin importar el sistema operativo que utilice el usuario.

Seguridad:

CSSE1 El sistema debe garantizar que la información sea registrada, visualizada, eliminada y actualizada únicamente por la persona que posea los privilegios correspondientes.

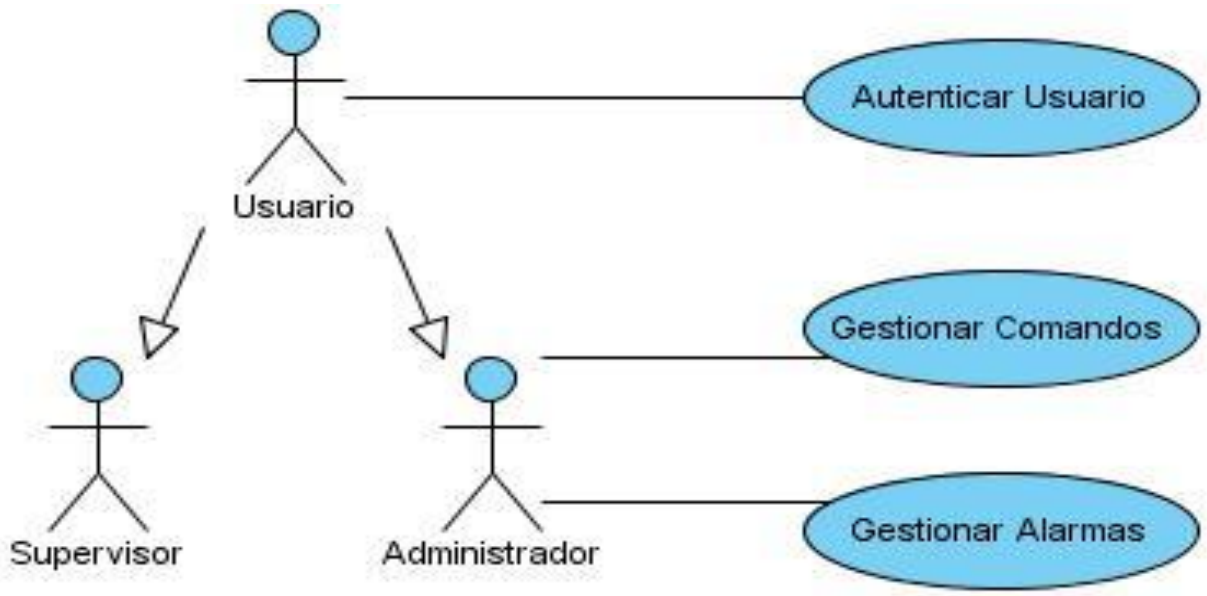
CSSE2 El sistema debe disponer de mecanismos de seguridad que garanticen el acceso y la manipulación autorizada y asegurando la confidencialidad, la integridad y la disponibilidad de la misma.

2.5. Descripción del Sistema

Actores del Sistema. Actor	Descripción
Usuario	Encargado de interactuar con el sistema, luego de autenticado podrá acceder a las funcionalidades del sistema entre las que se incluyen la gestión de alarmas y eventos.

Diagrama de Casos de Uso del Sistema.

El modelado de Casos de Uso se realiza con el objetivo de modelar de una forma simple y efectiva los requisitos del sistema desde el punto de vista del usuario. El diagrama de casos de uso constituye una visión general que se ha identificado para satisfacer los requerimientos funcionales del sistema.



2.5.1. Descripción de los Casos de Uso del Sistema.

La descripción detallada de los casos de uso muestra la expansión que permitirá entender a través de su descripción detallada, los procesos que se encuentran asociados a cada uno de ellos y así comprender como se propone que el sistema cumpla con los requerimientos establecidos con anterioridad.

Caso de Uso:	Autenticar Usuario
Actores:	Usuario
Resumen:	El caso de uso se inicia cuando el sistema envía los datos de inicio de sesión

	al servidor HMI.
Precondiciones:	El usuario debe tener privilegios de acceso a la aplicación.
Flujo Normal de Eventos	
1- El usuario entra los datos en el formulario. (Alterno 2.3)	2- El sistema verifica si usuario y contraseña son correctos. 2.1- El sistema muestra una interfaz gráfica.
Flujos alternos	2.3-El sistema emite un mensaje informando que el usuario y la contraseña son incorrectos.

Caso de Uso:	Gestionar Alarmas
Actores:	Administrador
Resumen:	El caso de uso se inicia cuando el administrador decide Gestionar Alarmas seleccionando la opción que desea realizar (silenciar, silenciar todas, reconocer, reconocer todas).
Precondiciones:	La aplicación debe haber cargado correctamente los datos de las alarmas. Y debe seleccionar la funcionalidad (silenciar, silenciar todas, reconocer, reconocer todas) dando clic derecho encima de la alarma.
Escenario "Silenciar Alarma"	
1- El administrador selecciona una alarma para silenciar.	2- El sistema verifica que la alarma seleccionada no haya sido silenciada con anterioridad.
3- El administrador selecciona la opción silenciar alarma.	4- El sistema silencia la alarma dejando de producir el sonido que emite.

Flujo Normal de Eventos	
cenario “Silenciar todas las Alarmas”	
1- El administrador selecciona todas las alarmas.	2- El sistema verifica que las alarmas seleccionadas no hayan sido silenciada con anterioridad.
3- El administrador selecciona la opción silenciar todas las alarmas.	4- El sistema silencia todas las alarmas dejando de producir el sonido que emite.
Escenario “Reconocer Alarma”	
1- El administrador selecciona la alarma a reconocer.	2- El sistema verifica que la alarma seleccionada no esté reconocida.
3- El administrador selecciona la opción reconocer alarma.	4- El sistema reconoce la alarma y cambia el color de su estado.
Escenario “Reconocer todas las Alarmas”	
1- El administrador selecciona la opción reconocer todas las alarmas.	2- El sistema verifica que todas las alarmas que no estén reconocidas.
3- El administrador selecciona la opción reconocer todas las alarma.	4- El sistema reconoce todas las alarmas y cambia los colores de estado.

Caso de Uso:	Gestionar Comandos
Actores:	Administrador
Resumen:	El caso de uso se inicia cuando el usuario decide gestionar eventos entre los que se incluyen los eventos de navegación, actualización y actualización de

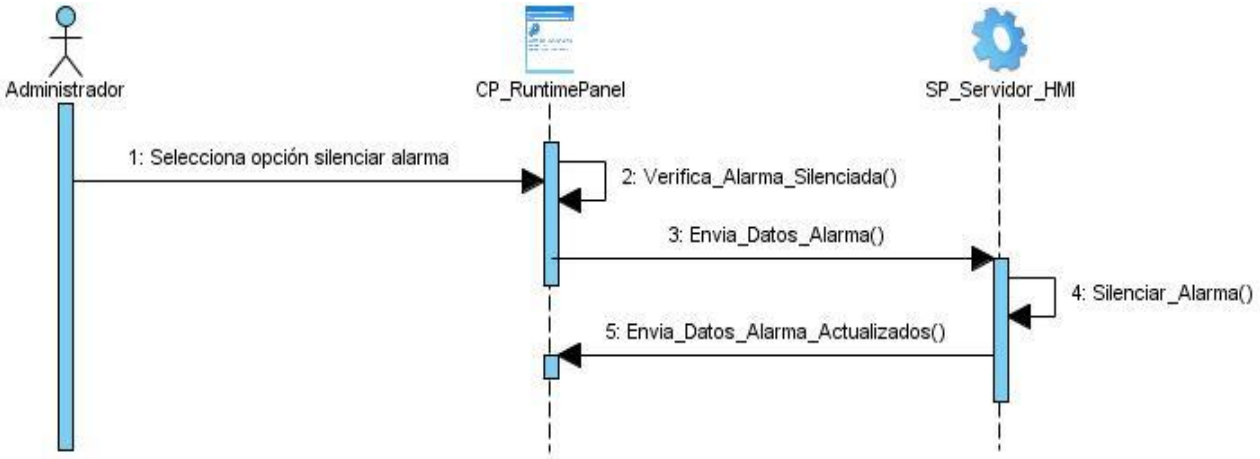
	objetos dentro del despliegue.
Precondiciones:	Debe haber conexión con el servidor HMI.
Flujo Normal de Eventos	
Escenario "Enviar comando de navegación entre despliegues"	
1- El administrador ejecuta la acción de mostrar otro despliegue.	2- El sistema verifica que el despliegue que se desea mostrar no es el despliegue actual. 2.1- El sistema muestra un nuevo despliegue.
Escenario "Enviar comando de actualización del despliegue"	
	1- El sistema verifica cuales objetos del despliegue han cambiado su estado. 1.1- El sistema actualiza los objetos del despliegue que han cambiado su estado.
Escenario "Enviar comando de actualización de un objeto dentro del despliegue"	
1- El administrador ejecuta la acción actualizar un objeto dentro del despliegue.	2- El sistema verifica si el objeto dentro del despliegue.

2.6. Diagramas de Secuencia

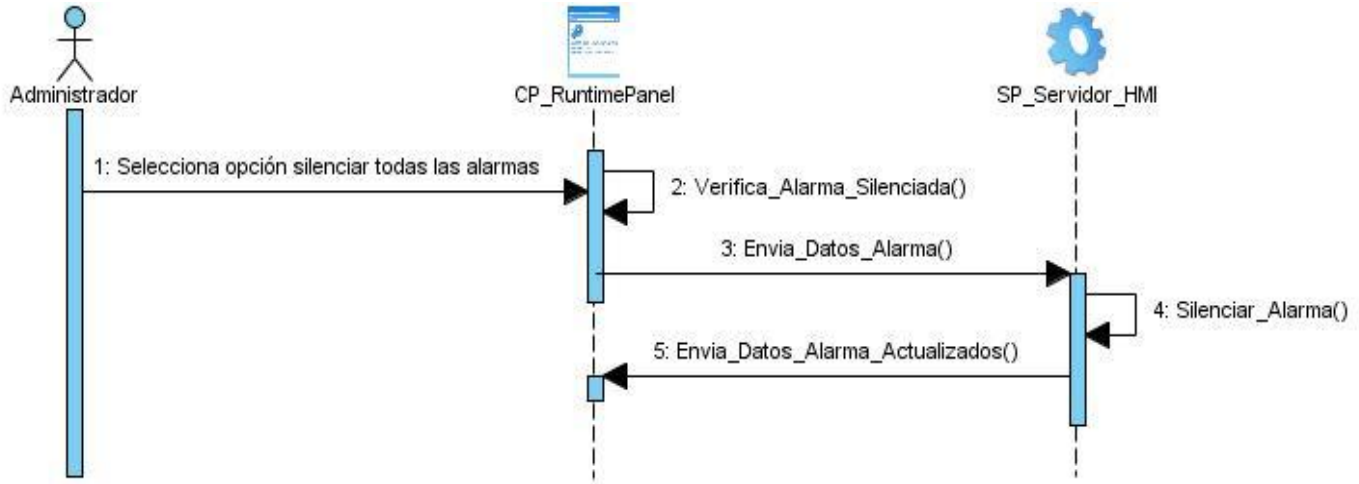
Los diagramas de secuencias muestran la interacción entre objetos y el orden secuencial en el que ocurren dichas interacciones, es decir cómo se comunican los objetos entre sí, logrando de esta forma representar los mensajes en función del tiempo.

A continuación se muestran los diagramas de secuencia para cada uno de los escenarios de los casos de uso del sistema.

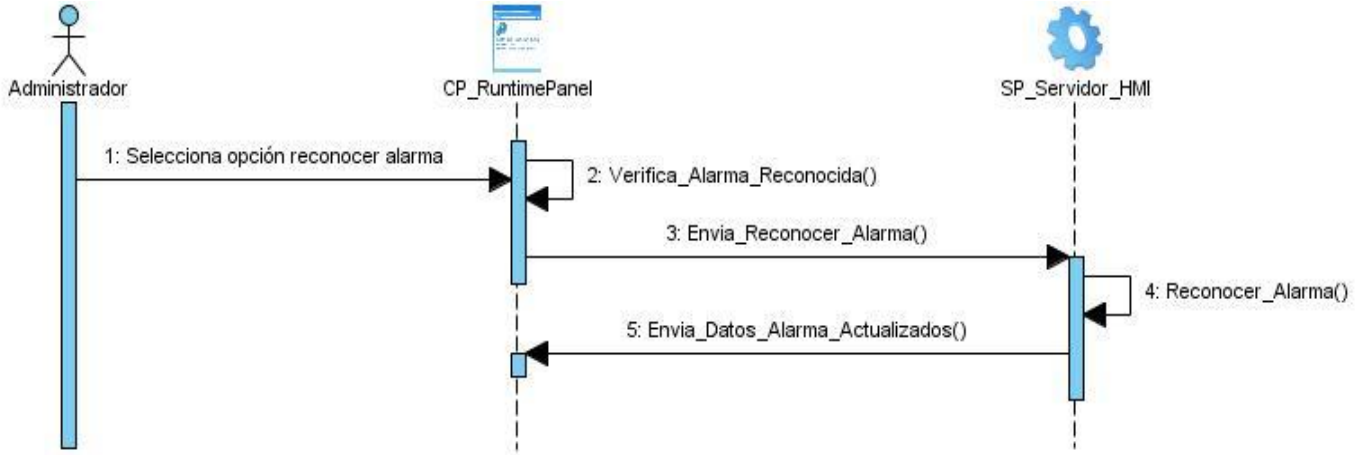
2.6.1. Diagrama de Secuencia: Caso de Uso Gestionar Alarmas (Escenario Silenciar Alarma).



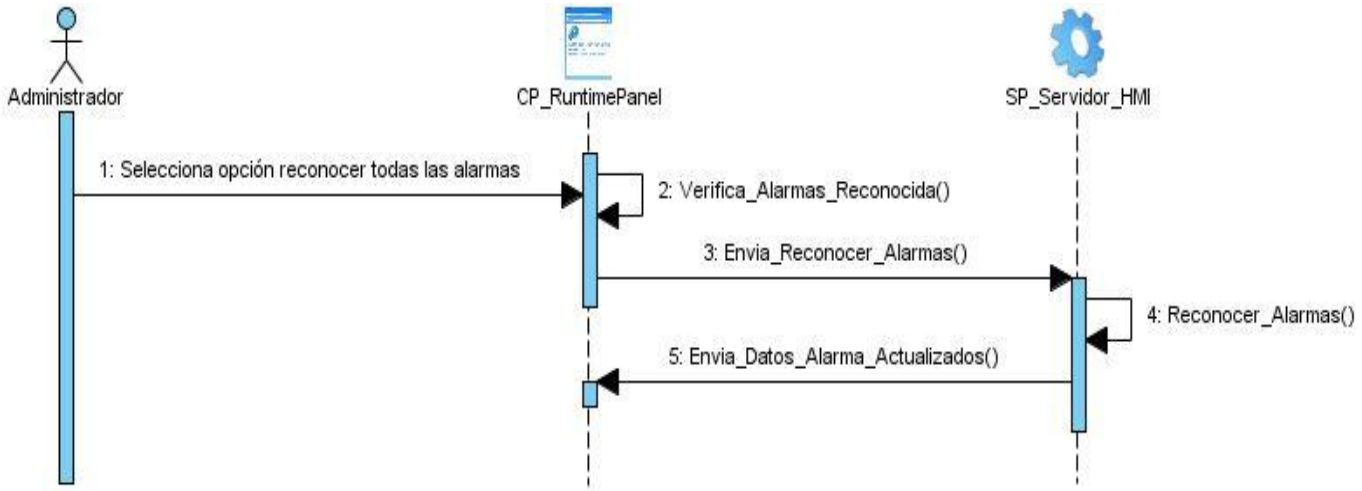
2.6.2. Diagrama de Secuencia: Caso de Uso Gestionar Alarmas (Escenario Silenciar todas las Alarma).



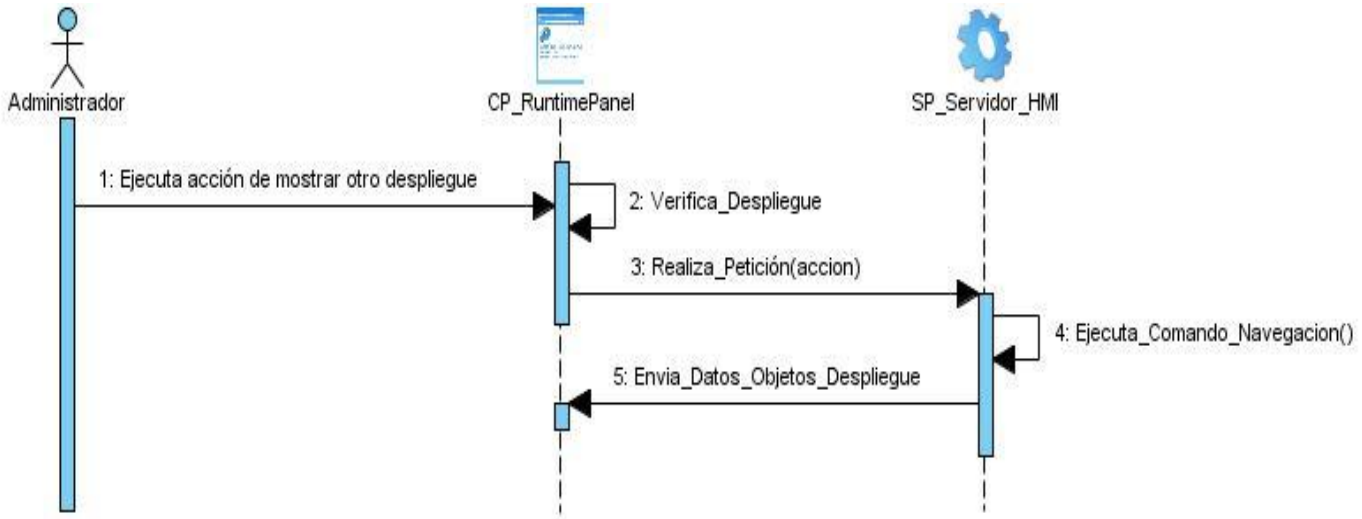
2.6.3. Diagrama de Secuencia: Caso de Uso Gestionar Alarmas (Escenario reconocer alarmas).



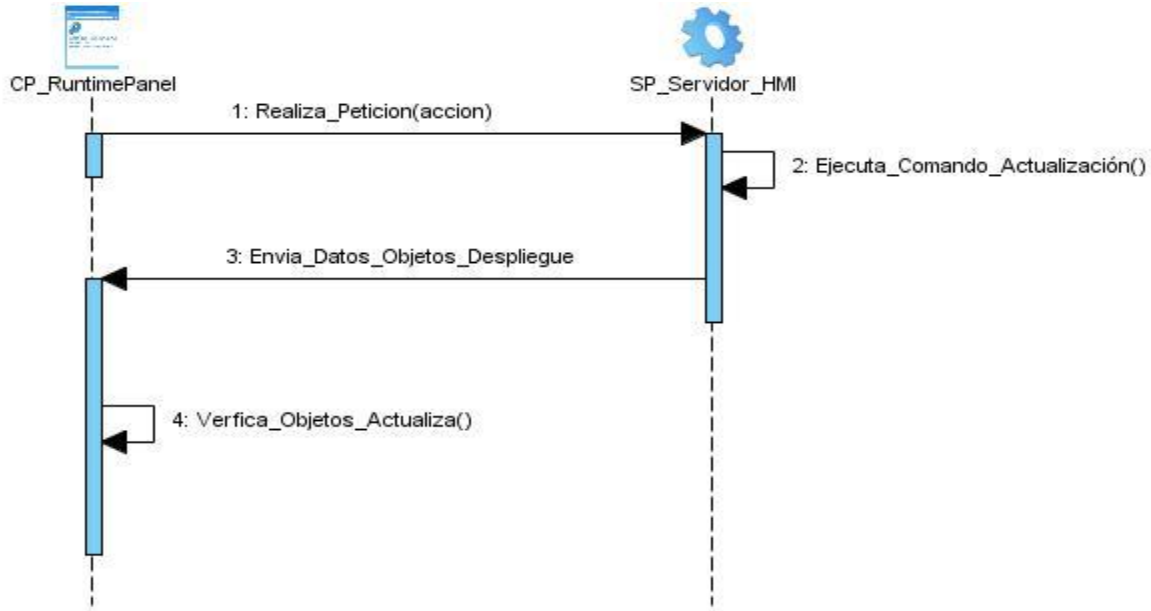
2.6.4. Diagrama de Secuencia: Caso de Uso Gestionar Alarmas (Escenario reconocer todas las alarmas).



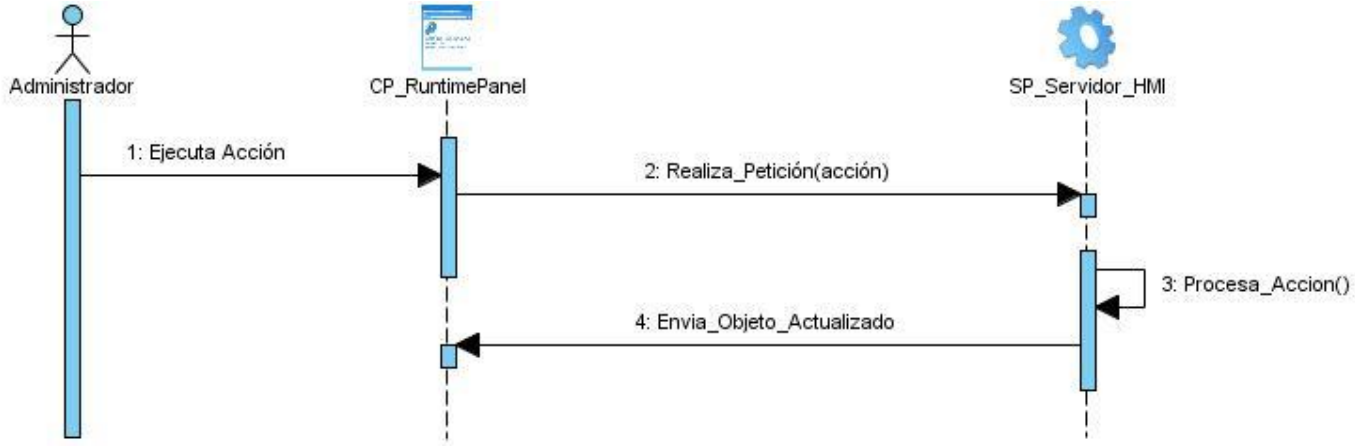
2.6.5. Diagrama de Secuencia: Caso de Uso Gestionar Comandos (Escenario Enviar comando de navegación entre despliegues).



2.6.6. Diagrama de Secuencia: Caso de Uso Gestionar Comandos (Escenario Enviar comando de actualización del despliegue).



2.6.7. Diagrama de Secuencia: Caso de Uso Gestionar Comandos (Escenario Enviar comando de actualización de un objeto dentro del despliegue).



2.7. Arquitectura de la aplicación web

El objetivo principal de la Arquitectura del Software es aportar elementos que ayuden a la toma de

decisiones y, al mismo tiempo, proporcionar conceptos y un lenguaje común. Para conseguirlo, la arquitectura del software se centra seleccionar y combinar estilos y patrones arquitectónicos.

Para el desarrollo de la propuesta de solución se identificó el patrón Modelo-Vista-Controlador (MVC) como estilo arquitectónico, el cual permite un desarrollo de forma modular y un mejor soporte para futuras actualizaciones del producto, dividiendo la aplicación tres componentes distintos: el modelo, la vista y el controlador, lo cual tiene como resultado final una aplicación muy robusta, de forma tal que con los cambios que se realicen en una parte de la aplicación, las demás no se vean afectadas.

Como patrón de diseño estructural se selecciona el Fachada al proporcionar una interfaz unificada para un conjunto de interfaces en un subsistema, haciéndolo más fácil de usar logrando reducir la complejidad y minimizar dependencias.

2.7.1. Patrón "Modelo-Vista-Controlador"

Al incorporar el modelo de arquitectura MVC a un diseño, las piezas de un programa se pueden construir por separado y luego unirlas en tiempo de ejecución. Si uno de los Componentes, posteriormente, se observa que funciona mal, puede reemplazarse sin que las otras piezas se vean afectadas.

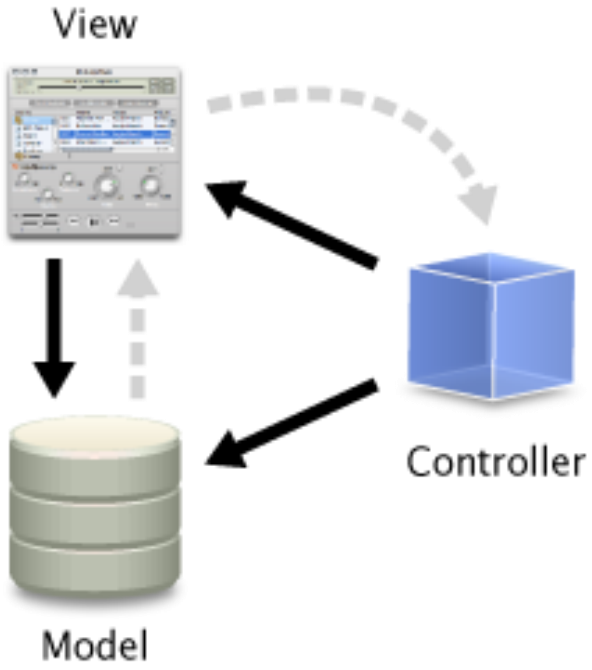


Figura 1: Representación del patrón Modelo - Vista – Controlador

Elementos del patrón:

- **Modelo**. El modelo administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador).

- **Vista.** Maneja la visualización de la información. Es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del modelo y muestra los datos al usuario. Interactúa con el Modelo a través de una referencia al propio Modelo.

- **Controlador.** Interpreta las acciones del ratón y el teclado, informando al modelo y/o a la vista para que cambien según resulte apropiado. Es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo. Cuando se realiza algún cambio,

entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo.

Tanto la vista como el controlador dependen del modelo, el cual no depende de las otras clases. Esta separación permite construir y probar el modelo independientemente de la representación visual. La separación entre la vista y el controlador puede ser secundaria en aplicaciones de clientes ricos y, de hecho, muchos frameworks de interfaz implementan ambos roles en un solo objeto. En aplicaciones web, por otra parte, la separación entre la vista (el browser) y el controlador (los componentes del lado del servidor que manejan los requerimientos de HTTP) está mucho más taxativamente definida (21).Existe herramientas de desarrollo que incorporan en las clases de la vista gran parte o todo el procesamiento de eventos. Con lo que el controlador queda semioculto dentro de la vista (22) esta variante del MVC recibe el nombre de MD (*Model Delegate,* por sus siglas en inglés) donde Delegate incluye la vista y el controlador. (26)

2.7.2. Patrón diseño estructural Fachada

Intención:

Se aplica cuando se quiere proporcionar una interfaz sencilla para un subsistema complejo, y desacoplar un subsistema de sus clientes y de otros subsistemas, haciéndolo más independiente y portable, si se quiera dividir los sistemas en niveles las fachadas serían el punto de entrada a cada nivel. **Fachada** conoce las clases del subsistema y delega las peticiones de los clientes en los objetos del subsistema.

Clases del subsistema: Implementan la funcionalidad del subsistema y llevan a cabo las peticiones que les envía la fachada, aunque no la conocen.

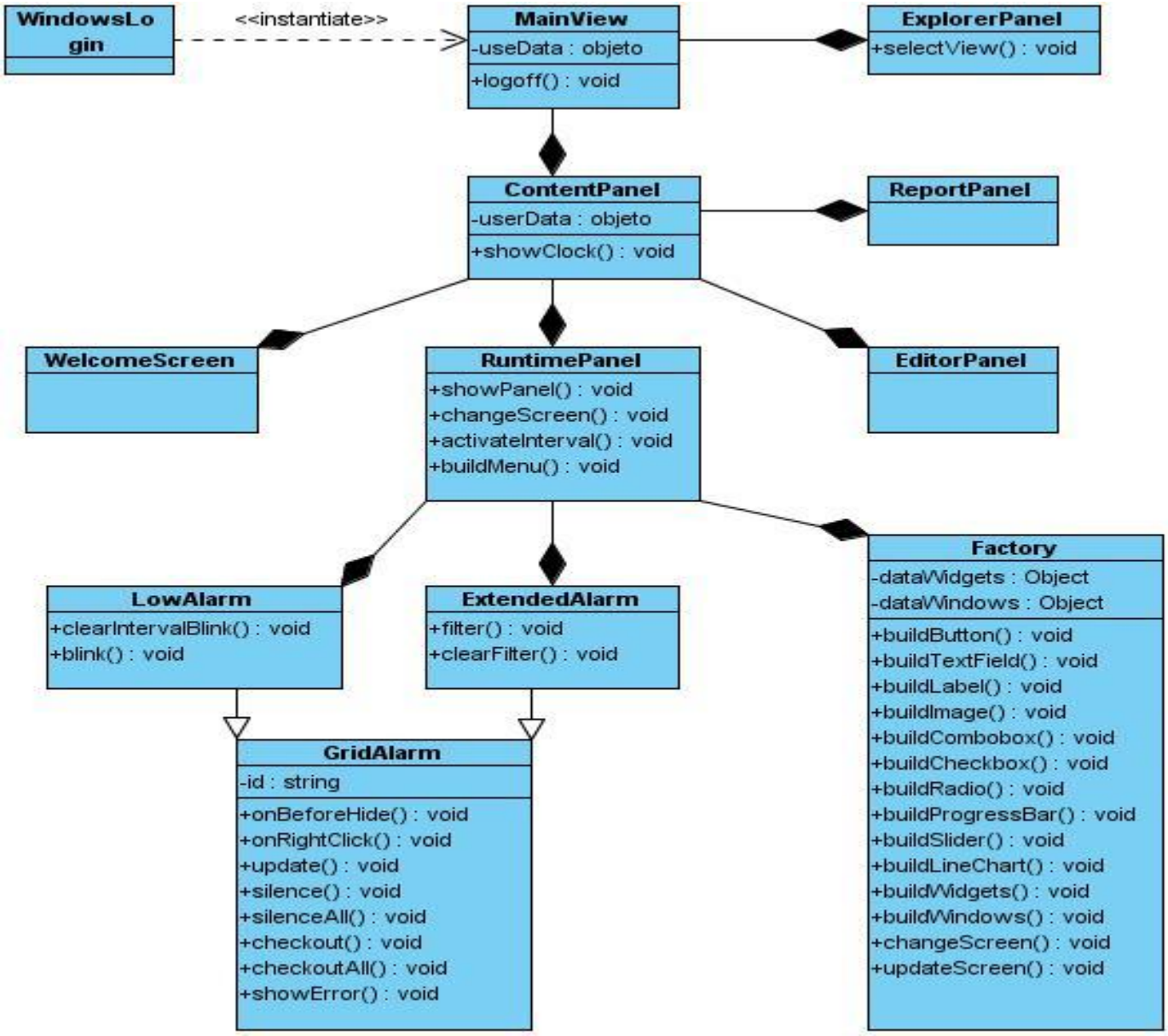
Colaboraciones: Los clientes se comunican con el subsistema a través de la fachada, que reenvía las peticiones a los objetos del subsistema apropiados y puede realizar también algún trabajo de traducción. Los clientes que usan la fachada no necesitan acceder directamente a los objetos del sistema.

Ventajas e inconvenientes

Oculta a los clientes de la complejidad del subsistema y lo hace más fácil de usar. Favorece un acoplamiento débil entre el subsistema y sus clientes, ayuda a dividir un sistema en capas y reduce dependencias de compilación. No evita que se usen las clases del sistema si es necesario, dejando la puerta del subsistema abierta por si es necesario. (23)

2.8. Diagrama de Clases del Diseño

Los diagramas de clases describen la estructura del sistema, creando el diseño conceptual de la información que se manejará, y los componentes que se encargarán del funcionamiento y la relación entre uno y otro.


WindowsLogin
<<instantiate>>
MainView
-useData : objeto
+logoff() : void
ExplorerPanel
+selectView() : void
ContentPanel
-userData : objeto
+showClock() : void
ReportPanel
WelcomeScreen
RuntimePanel
+showPanel() : void
+changeScreen() : void
+activateInterval() : void
+buildMenu() : void
EditorPanel
LowAlarm
+clearIntervalBlink() : void
+blink() : void
ExtendedAlarm
+filter() : void
+clearFilter() : void
Factory
-dataWidgets : Object
-dataWindows : Object
+buildButton() : void
+buildTextField() : void
+buildLabel() : void
+buildImage() : void
+buildCombobox() : void
+buildCheckbox() : void
+buildRadio() : void
+buildProgressBar() : void
+buildSlider() : void
+buildLineChart() : void
+buildWidgets() : void
+buildWindows() : void
+changeScreen() : void
+updateScreen() : void
GridAlarm
-id : string
+onBeforeHide() : void
+onRightClick() : void
+update() : void
+silence() : void
+silenceAll() : void
+checkout() : void
+checkoutAll() : void
+showError() : void

2.8.1. Descripción de las Clases del Diseño

Nombre: MainView	
Descripción: Es la clase encargada de implementar la interfaz para incorporar los componentes visuales de la aplicación.	
Tipo de clase: Interfaz	
Atributo	**Tipo**
- userData	- objeto
- explorerPanel	- objeto
- runtimePanel	- objeto
- editorPanel	- objeto
- reporterPanel	- objeto
Para cada responsabilidad:	
Nombre:	- logoff(): void
Descripción:	- Cerrar sesión de usuario. No devuelve nada (void).

Nombre: ContentPanel	
Descripción: Componente visual que proporciona un área para mostrar los diferentes ambientes de trabajo (ejecución, edición y reportes).	
Tipo de clase: Interfaz	
Atributo	**Tipo**
- userData	- objeto
Para cada responsabilidad:	
Nombre:	- showClock(): void
Descripción:	- Muestra la hora del sistema y el tiempo de sesión de usuario. No devuelve nada (void).

Nombre: ExplorerPanel	
Descripción: Componente visual para mostrar el explorador de proyectos.	
Tipo de clase: Interfaz	
Atributo	**Tipo**
-	-
Para cada responsabilidad:	
Nombre:	- selectView(): void
Descripción:	- Selecciona la vista que desee el usuario. No devuelve nada (void).

Nombre: WelcomeScreen	
Descripción: Componente visual para mostrar informaciones o datos de interés para el usuario.	
Tipo de clase: Interfaz	
Atributo	**Tipo**
-	-
Para cada responsabilidad:	
Nombre:	-
Descripción:	-

Nombre: RuntimePanel	
Descripción: Ambiente visual para supervisar y controlar los procesos automatizados	
Tipo de clase: Interfaz	
Atributo	**Tipo**
- runtimeScreen	- objeto

- lowAlarmPanel	- objeto
Para cada responsabilidad:	
Nombre:	- getToolbar(): void
Descripción:	- Construye una barra de herramientas de forma dinámica con accesos directos a cada uno de los despliegues del SCADA. No devuelve nada (void).
Nombre:	- sendCommand (options: objeto): void
Descripción:	- Permite enviar un comando con los parámetros recibidos. No devuelve nada (void).
Nombre:	- changeScreen(id: string): void
Descripción:	- Permite navegar de un despliegue hacia otro despliegue con id igual al que se pasa por parámetro. No devuelve nada (void).
Nombre:	- activateInterval(): void
Descripción:	- Activa un temporizador, el cual permite realizar la función de actualizar la información de las alarmas y de los objetos visuales del despliegue cada un intervalo de tiempo. No devuelve nada (void).
Nombre:	- showPanel(idPanel: string): void
Descripción:	- Muestra el panel que tiene un id igual al que se pasa por parámetro. No devuelve nada (void).

Nombre: LowAlarm	
Descripción: Componente visual para mostrar el panel de alarmas reducidas	
Tipo de clase: Interfaz	
Atributo	**Tipo**
- gridAlarm	- objeto
Para cada responsabilidad:	
Nombre:	-blinkAlarm(): void

Descripción:	Activa un temporizador que permite realizar la función de parpadeo de las alarmas en dependencia de los estados que tome cada un intervalo de tiempo. No devuelve nada (void).
Nombre:	- clearIntervalBlink() : void
Descripción:	- Desactivar temporizador del parpadeo de las alarmas. No devuelve nada (void).

Nombre: GridAlarm		
Descripción: Componente visual para mostrar información de las alarmas.		
Tipo de clase: Interfaz		
Atributo	**Tipo**	
- id	- string	
Para cada responsabilidad:		
Nombre:	-checkoutAlarm(): void	
Descripción:	- Ejecuta la acción de reconocimiento de la alarma seleccionada. No devuelve nada (void).	
Nombre:	-checkoutAllAlarm(): void	
Descripción:	- Ejecuta la acción de reconocimiento de todas las alarmas seleccionada. No devuelve nada (void).	
Nombre:	- silenceAlarm(): void	
Descripción:	Ejecuta la acción de silenciar una alarma seleccionada. No devuelve nada (void).	
Nombre:	- silenceAllAlarm(): void	
Descripción:	Ejecuta la acción de silenciar todas las alarmas. No devuelve nada (void).	

Nombre:	- update(): void
Descripción:	Ejecuta la acción de emitir un sonido en dependencia del tipo de alarma. No devuelve nada (void).

Nombre: ExtendedAlarm		
Descripción: Componente visual para mostrar el panel de alarmas extendidas.		
Tipo de clase: Interfaz		
Atributo	**Tipo**	
Para cada responsabilidad:		
Nombre:	- filter (begin: Date, end: Date, priority: string, type: string): void	
Descripción:	**-** Realiza el filtrado de las alarmas atendiendo los datos que se pasan por parámetro. No devuelve nada (void).	
Nombre:	**-** clearFilter(): void	
Descripción:	**-** Limpia los filtros.	

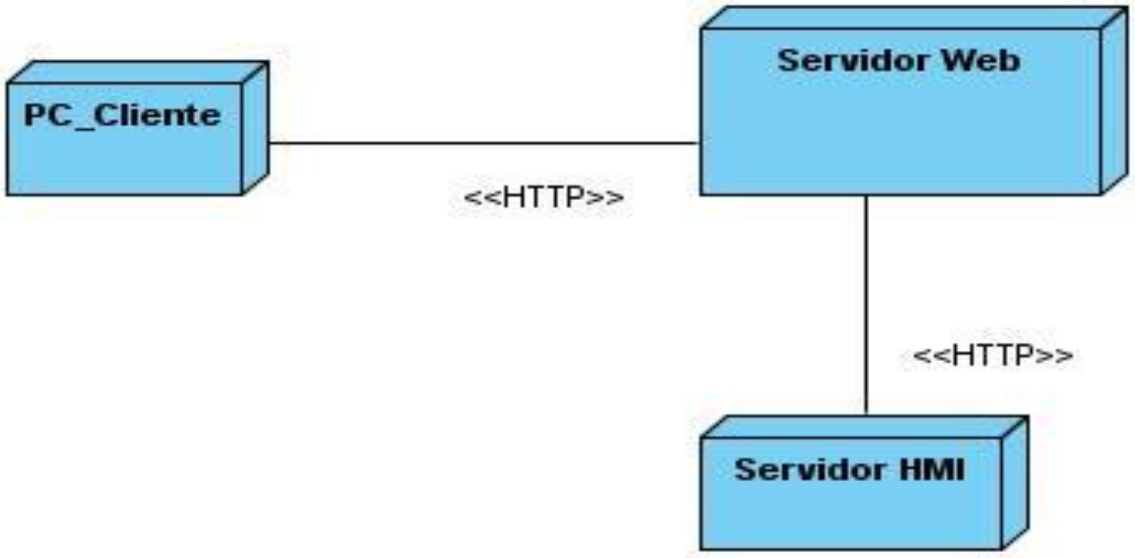
Nombre: Factory	
Descripción: Clase encargada de construir los objetos que se visualizan en los despliegues del SCADA.	
Tipo de clase: Controladora	
Atributo	**Tipo**
- canUpdate	- bolean
- DataWindows	- objeto
- DataWidgets	- objeto
Para cada responsabilidad:	
Nombre:	- buildButton(record: objeto): void

Descripción:	- Construye un botón dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildTextField (record: objeto): void
Descripción:	- Construye un textfield dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildLabel (record: objeto): void
Descripción:	- Construye un label dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildImage (record: objeto): void
Descripción:	- Construye una imagen dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildCombobox (record: objeto): void
Descripción:	- Construye un combobox dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildCheckbox (record: objeto): void
Descripción:	- Construye un checkbox dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildRadio (record: objeto): void
Descripción:	- Construye un radio dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildProgressBar (record: objeto): void
Descripción:	- Construye un progressBar dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildSlider (record: objeto): void
Descripción:	- Construye una slider dentro del despliegue atendiendo a las propiedades que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildWidgets (store: objeto): void

Descripción:	- Construye todos los objetos de un despliegue atendiendo a los datos de cada uno de los objetos que se pasan por parámetro. No devuelve nada (void).
Nombre:	- buildWindows (store: objeto): void
Descripción:	- Construye el despliegue que se muestra por defecto en el sistema atendiendo a los datos que se pasan por parámetro. No devuelve nada (void).
Nombre:	- changeScreen(panel: objeto, id: string): void
Descripción:	- Permite la navegación de un despliegue a otro.
Nombre:	- updateScreen(): void
Descripción:	- Actualiza los objetos del despliegue que se está mostrando. No devuelve nada (void).
Nombre:	- showError(): void
Descripción:	- Muestra un mensaje de error en caso de que no se obtengan los datos de los objetos. No devuelve nada (void).

2.9. Diagrama de Despliegue

El diagrama de despliegue es el que modela los sistemas mostrando el comportamiento y las características estáticas a través de las clases, ofrece como queda la distribución física de los nodos para la implantación de nuestro sistema.



Es de vital importancia, para toda la aplicación web fundamentalmente en el ambiente de ejecución, la presencia de un sumario de alarmas , esto hace más cómoda la gestión de alarmas y permite con gran facilidad el control de los procesos .En este capítulo se ha realizado la descripción de la mayoría de los elementos implementados, se han seleccionado las principales funcionalidades y se ha descrito detalladamente las clases más significativas que intervienen en cada uno de los escenarios propuestos.

Capítulo 3: Validación y seguimiento a la solución.

3.1. Introducción

En el presente capítulo se realiza el proceso de pruebas para validar el correcto funcionamiento de las funcionalidades del sistema de supervisión y control web, también se analizaran las no conformidades con el objetivo de realizar acciones correctivas y preventivas que garanticen la calidad del desarrollo de la aplicación web.

3.2. Pruebas.

Un factor de vital importancia para la producción de un software es lograr la calidad del producto, y precisamente las pruebas que se le realicen constituyen un papel protagónico pues con la realización de un conjunto finito de casos de pruebas es posible verificar el comportamiento del producto durante su ciclo de vida. Este proceso además de garantizar la calidad del software, tiene como objetivo comprobar la satisfacción de los requisitos y localizar para subsanar, el mayor número de deficiencias antes de liberar el producto final. El tamaño y complejidad del proceso de pruebas depende del tamaño y complejidad del producto que se está desarrollando y para ejecutarlo es necesario definir nivel, método y técnicas de pruebas a utilizar. Para el sistema desarrollado se decidió realizar las siguientes pruebas definidas a continuación:

3.2.1. Nivel de Prueba: Sistema. (25)

(12) A este nivel es verificado que cada elemento interactúe de forma adecuada, que sea alcanzado la funcionalidad y el rendimiento del sistema total, sean validados los requisitos establecidos comparándolos con el sistema construido.

3.2.2. Tipo de Prueba: Funcionalidad.

Este tipo de prueba asegura el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. Verificar en la aplicación web todas las funcionalidades requeridas por el sistema.

Método de Prueba: Caja Negra. (25) Es denominada prueba de comportamiento, se centra en los requisitos funcionales del software, permite obtener conjuntos de condiciones de entrada que ejerciten

completamente todos los requisitos funcionales de un programa y está referida a las pruebas que se llevan a cabo sobre la interfaz del software examinando algunos aspectos del modelo fundamental del sistema sin tener mucho en cuenta la estructura lógica interna del software.

3.3. Técnicas de Pruebas.

Para obtener resultados óptimos se hace necesario combinar adecuadamente tres técnicas de prueba del método de Caja Negra, que estarán encaminadas a probar que el software responde correctamente a las entradas de los usuarios, garantizando de esta forma que su respuesta se corresponde a los requisitos funcionales.

3.3.1. Técnica de Prueba: Particiones o Clases de Equivalencia. (25)

Método de prueba de Caja Negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. La partición equivalente se dirige a una definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar. Constituye una técnica algebraica que trata cada parámetro como un modelo donde unos datos son equivalentes a otros. Se realiza dividiendo un rango excesivamente amplio de posibles valores reales a un conjunto reducido de clases de equivalencia, entonces es suficiente probar un caso de cada clase, pues los demás datos de la misma clase son equivalentes. El diseño de casos de prueba según esta técnica consta de dos pasos:

1. Identificar las clases de equivalencia.
2. Identificar los casos de prueba.

3.3.2. Técnica de Prueba: Valores Límites. (25)

Esta técnica es utilizada muchas veces como complemento de la partición de equivalencia. Su fundamento se basa en que la mayor cantidad de errores se encuentra en los límites del rango de valores de entrada de un programa, se prueban los valores limites del rango incluyendo los valores fuera del

rango que estén más cercanos a él. Se consideran como condiciones límites aquellas que se encuentran en los márgenes de la clase de equivalencia, tanto de entrada como de salida.

3.3.3. Técnica de Prueba: Conjetura de Errores. (25)

Esta técnica está basada mayormente en la intuición y la experiencia adquirida por el que diseña las pruebas. No tiene un procedimiento estándar definido sin embargo es muy utilizado por las personas dedicadas a realizar pruebas al software.

3.4. Ambiente de Prueba.

Las computadoras utilizadas para el desarrollo de las pruebas poseían los siguientes requerimientos técnicos de software y hardware: Tarjetas Madre Intel, con capacidad de disco duro de 150 gigas, microprocesador Intel Core2Duo E4500 con velocidad de 2.20 GHz y 1.0 giga byte de memoria RAM. Poseían una velocidad de conexión de 100 megabits por segundo. El sistema operativo en el cual se desarrollaron las pruebas fue Ubuntu.

3.5. Diseño de Casos de Pruebas.

La sección está dedicada a mostrar los casos de pruebas diseñados que poseen la responsabilidad de probar todas las funcionalidades definidas para el software, durante las realización de las mismas fueron probados los tres Casos de Uso descritos de la aplicación.

3.5.1. Caso de Uso: Gestionar Alarmas

Este caso de uso permite (silenciar, silenciar todas, reconocer, reconocer todas) estas opciones contribuyen a gestionar las alarmas de manera eficaz.

Las pruebas realizadas a este caso de uso son las siguientes:

1. Silenciar una alarma seleccionada.
2. Silenciar todas las alarmas seleccionadas.
3. Reconocer una alarma seleccionada.

4. Reconocer todas las alarmas seleccionadas.
5. Verificar si la alarma emite sonido correcto de acuerdo al tipo de severidad.
6. Verificar si la alarma emite intermitencia de colores en correspondencia a su estado.

Caso de Prueba (CP1) Silenciar Alarma

1. Breve Descripción

Este Caso de Prueba permite comprobar la funcionalidad de silenciar una alarma seleccionada en el panel de alarmas.

2. Flujo Central

2.1-El usuario selecciona en el panel de alarmas en la opción silenciar alarma.

2.2- El sistema silencia la alarma.

3. Condiciones de Ejecución

- Que el actor se autentique con los privilegios correspondientes.

4. Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
	Se debe cargar los datos de las alarmas que se envían por	Se debe mostrar un mensaje que indique que no se han cargado correctamente los	Se muestra el mensaje correctamente.	

	el formato JSON.	datos.		
Seleccionar la opción: **Silenciar Alarma.**		El sistema silencia la alarma correspondiente	El sistema ha silenciado la alarma seleccionada correctamente.	

Caso de Prueba (CP2) Silenciar todas las Alarmas.

1. Breve Descripción

Este Caso de Prueba permite comprobar la funcionalidad de silenciar todas las alarmas seleccionadas del panel de alarmas.

2. Flujo Central

2.1-El usuario selecciona en el panel de alarmas en la opción silenciar todas alarmas.

2.2- -El sistema silencia las alarmas.

3. Condiciones de Ejecución

- Que el actor se autentique con los privilegios correspondientes.

4. Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones

Seleccionar la opción: **Silenciar todas las Alarma.**		El sistema silencia las alarmas correspondientes	El sistema silencia todas las alarmas seleccionadas correctamente.	

Caso de Prueba (CP3) Reconocer una Alarma.

1. Breve Descripción

Este Caso de Prueba permite comprobar la funcionalidad de Reconocer una alarma seleccionada del panel de alarma.

2. Flujo Central

2.1-El usuario selecciona en el panel de alarmas en la opción reconocer alarma.

2.2- El sistema reconoce la alarma.

3. Condiciones de Ejecución

- Que el actor se autentique con los privilegios correspondientes.

4. Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Seleccionar la opción:		El sistema reconoce la alarma	El sistema reconoce la	

Reconocer Alarma.		correspondiente.	alarma seleccionada correctamente.	
	Se muestran errores ortográficos en el menú de las alarmas	Se debe revisar los códigos necesarios para mostrar la ortografía correctamente.	Se erradicaron los errores ortográficos presentados.	

Caso de Prueba (CP4) Reconocer todas las alarmas.

1. **Breve Descripción**

Este Caso de Prueba permite comprobar la funcionalidad de Reconocer todas las alarmas seleccionadas del panel de alarma.

2. **Flujo Central**

2.1-El usuario selecciona en el panel de alarmas en la opción reconocer todas las alarmas.

2.2- El sistema reconoce todas las alarmas seleccionadas.

3. **Condiciones de Ejecución**

- Que el actor se autentique con los privilegios correspondientes.

4. **Iteraciones**

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Seleccionar la opción: Reconocer todas las Alarmas.		El sistema reconoce todas las alarmas correspondientes.	El sistema reconoce la alarma seleccionada correctamente.	

Caso de Prueba (CP5) Verificar si la alarma emite sonido correcto de acuerdo al tipo de severidad.

1. **Breve Descripción**

Este Caso de Prueba permite comprobar si las alarmas en dependencia del nivel de severidad que tomen emiten el sonido correspondiente.

2. **Flujo Central**

2.1- El usuario selecciona en el explorador de proyectos la opción de ir a la interfaz runtime.

2.2- El sistema muestra el panel de alarma y en dependencia del nivel que tome (critica, media o baja) emite el sonido correspondiente a su estado.

2.3- El sistema emite el sonido de las alarmas seleccionadas.

3. **Condiciones de Ejecución**

- Que el actor se autentique con los privilegios correspondientes.

4. **Iteraciones**

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Verificar que la alarma de severidad crítica emite el sonido correspondiente.		El sistema debe emitir sonido de alarma crítica repetitivamente en el navegador Internet Explorer.	El sistema emite sonido correctamente.	
Verificar que la alarma de severidad media emite el sonido correspondiente.		El sistema debe emitir sonido de alarma media repetitivamente en el navegador Internet Explorer.	El sistema emite sonido correctamente.	
Verificar que la alarma de severidad baja emite el sonido correspondiente.		El sistema debe emitir sonido de alarma baja.	El sistema emite sonido correctamente.	
	Verificar que las alarmas emiten sonido en el navegador Mozilla Firefox.	El sistema debe emitir los sonidos correspondientes en el navegador Mozilla Firefox repetitivamente.	El sistema no emite el sonido esperado en el navegador Mozilla Firefox.	Deben estar instalados los plugins necesarios para que el sonido se emita correctamente.

1. **Caso de Prueba (CP6) Verificar si la alarma emite intermitencia de colores en correspondencia a su estado.**

1. **Breve Descripción**

Este Caso de Prueba permite comprobar si las alarmas en dependencia del nivel de severidad que tomen (critica, media, baja) procesan su intermitencia de colores en dependencia de su estado.

2. **Flujo Central**

2.1-El usuario selecciona en el explorador de proyectos la opción de ir a la interfaz runtime.

2.2- El sistema muestra el panel de alarma y en dependencia del nivel de severidad que tome (critica, media o baja) emite la intermitencia correspondiente a su estado.

2.3-El sistema emite intermitencia según su nivel de severidad y estado de las mismas.

3. **Condiciones de Ejecución**

- Que el actor se autentique con los privilegios correspondientes.

4. **Iteraciones**

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Verificar si la alarma de severidad critica es activa no reconocida.		El sistema debe mostrar la alarma con fondo negro y letras rojas y con intermitencia de	El sistema muestra fondo negro y letras rojas y con intermitencia de	

		fondo rojo y letras negras.	fondo rojo y letras negras.	
Verificar si la alarma de severidad crítica es activa reconocida.		El sistema debe mostrar fondo negro y letras rojas de manera estática.	El sistema muestra fondo negro y letras rojas de manera estática.	
Verificar si la alarma de severidad critica es no activa no reconocida.		El sistema debe mostrar **Fondo Negro, Letras Rojas, intermitencia hacia Fondo Negro, Letras Verdes.**	El sistema muestra **Fondo Negro, Letras Rojas, intermitencia hacia Fondo Negro, Letras Verdes.**	
Verificar si la alarma de severidad media es activa no reconocida.		El sistema debe mostrar **Fondo Negro, Letras Naranja, intermitencia hacia Fondo Naranja, Letras Negras**	El sistema muestra **Fondo Negro, Letras Naranja, intermitencia hacia Fondo Naranja, Letras Negras**	
Verificar si la alarma de severidad media es activa		El sistema debe mostrar **Fondo Negro, Letras Naranja, en modo**	El sistema muestra **Fondo Negro, Letras Naranja, en**	

reconocida.		estático	modo estático	
Verificar si la alarma de severidad media es no activa no reconocida.		El sistema debe mostrar Fondo Negro, Letras Naranjas, intermitencia hacia Fondo Negro, Letras Verdes	El sistema muestra Fondo Negro, Letras Naranjas, intermitencia hacia Fondo Negro, Letras Verdes	
Verificar si la alarma de severidad baja es activa no reconocida.		El sistema debe mostrar Fondo Negro, Letras Amarillas, intermitencia hacia Fondo Amarillo, Letras Negras	El sistema muestra Fondo Negro, Letras Amarillas, intermitencia hacia Fondo Amarillo, Letras Negras	
Verificar si la alarma de severidad baja es activa reconocida.		El sistema debe mostrar Fondo Negro, Letras Amarillas, en modo estático	El sistema muestra Fondo Negro, Letras Amarillas, en modo estático	
Verificar si la alarma de severidad baja es no activa no		El sistema debe mostrar Fondo Negro, Letras Rojas,	El sistema muestra Fondo Negro, Letras Rojas,	

reconocida.		**intermitencia hacia Fondo Negro, Letras Verdes.**	**intermitencia hacia Fondo Negro, Letras Verdes.**	
	Se muestran errores ortográficos.	Se debe revisar los códigos necesarios para mostrar la ortografía correctamente.	Se erradicaron los errores ortográficos presentados.	

Caso de Uso: Gestionar Comandos.

Este caso de uso permite gestionar comandos de (navegación, actualización y actualización de objetos dentro del despliegue) estas opciones contribuyen a gestión de los comandos.

Las pruebas realizadas a este caso de uso son las siguientes:

2. Enviar comando de navegación entre despliegues.
3. Enviar comando de actualización del despliegue.
4. Enviar comando de actualización de un objeto dentro del despliegue.

Caso de Prueba (CP7) Enviar comando de navegación entre despliegues.

1. Breve Descripción

Este Caso de prueba permite comprobar la funcionalidad de enviar comando de navegación entre despliegues en el ambiente de ejecución web.

2. Flujo Central

2.1-El usuario ejecuta una acción mostrar un nuevo despliegue.

2.2- El sistema verifica que el despliegue a mostrar no está siendo mostrado.

2.3-El sistema muestra un nuevo despliegue.

3. Condiciones de Ejecución

- Que el actor se autentique con los privilegios correspondientes.

4. Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Ejecutar acción de navegación entre despliegues**.**		El sistema debe verificar que el despliegue a mostrar no está siendo mostrado y debe visualizar el nuevo despliegue.	Se muestra correctamente el nuevo despliegue.	
	Al intentar cambiar de despliegue los objetos no se muestran correctamente.	El sistema debe mostrar un mensaje de error indicando que no hay conexión con el servidor.	El sistema muestra el mensaje correctamente.	
	Navegar hacia el mismo despliegue.	El sistema debe mostrar un mensaje de información	El sistema muestra el mensaje	

		indicando que se encuentra situado en el mismo despliegue.	correctamente.	

Caso de Prueba (CP8) Enviar comando de actualización del despliegue.

5. Breve Descripción

Este Caso de prueba permite comprobar la funcionalidad de ejecutar un comando de actualización en el ambiente de ejecución web.

6. Flujo Central

2.1- El sistema verifica que la acción se corresponda a un comando de actualización.

2.2-El sistema muestra una vista de despliegue actualizada.

7. Condiciones de Ejecución

- Que el actor se autentique con los privilegios correspondientes.

8. Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Ejecutar acción de actualización del despliegue.		El sistema debe enviar comandos de actualización del despliegue automáticamente actualizando los objetos que hayan	Los objetos que han cambiado su estado se actualizan correctamente.	

		cambiado su estado.		

Caso de Prueba (CP8) Enviar comando de actualización de un objeto dentro del despliegue.

9. Breve Descripción

Este Caso de prueba permite comprobar la funcionalidad de ejecutar un comando de actualización de objeto en el ambiente de ejecución web.

10. Flujo Central

2.1-El usuario ejecuta una acción.

2.2- El sistema verifica que la acción se corresponda a un comando de actualización de objeto.

2.3-El sistema muestra una vista de despliegue con el objeto actualizado.

11. Condiciones de Ejecución

- Que el actor se autentique con los privilegios correspondientes.

12. Iteraciones

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
Ejecutar acción de actualización de un objeto dentro del despliegue**.**		El sistema debe verificar que la acción que se corresponde es un comando de actualización de un objeto y seguidamente	El sistema realiza comando de actualización de un objeto y se muestra correctamente la actualización del	

		debería mostrar otra vista del despliegue con el objeto actualizado.	objeto seleccionado.	
	Al intentar actualizar un objeto dentro del despliegue este no se actualiza correctamente.	El sistema debe mostrar un mensaje de error indicando que no hay conexión con el servidor.	El sistema muestra el mensaje correctamente.	

3.6. Resumen de las Pruebas.

Como resultado de la ejecución de las pruebas realizadas, fueron detectadas cuatro no conformidades y observaciones sobre la respuesta del sistema ante los casos de pruebas diseñados.

Elemento	No	No conformidad	Aspecto correspondiente	Etapa	Importancia	Recomendación

Aplicación	1	El sistema debería mostrar el sonido de la alarma en el navegador Mozilla Firefox repetitivamente.	update()		X	Revisar que este instalado el plugin necesario para que funcione correctamente.
Aplicación	2	Errores de ortografía.				Revisar los códigos necesarios para mostrar la ortografía correctamente.
Aplicación	3	El sistema no carga los datos de las alarmas que se envían por el formato JSON			X	Revisar si el url especificado está bien direccionado.
Aplicación	4	El sistema no muestra los objetos correctamente.			X	Verificar que hay conexión con el servidor HMI.

Con la realización de los casos de pruebas se logra poner en práctica las diferentes funcionalidades implementadas y determinar las no conformidades para la corrección de los errores que posteriormente garantizará la calidad del software.

Conclusiones

Al término de la investigación se cumplieron todos los objetivos propuestos y se puede concluir que:

1. La obtención de la aplicación web permitirá el acceso a la visualización de los procesos industriales del SCADA cubano desde cualquier localización.
2. La aplicación web desarrollada constituye un aporte al módulo HMI pues permite supervisar y controlar los procesos industriales del SCADA cubano.
3. La incorporación de nuevas funcionalidades mediante la web permitió la gestión de alarmas y comandos garantizado una nueva alternativa web antes vista solamente en soluciones de escritorio.

Recomendaciones

Aunque los objetivos propuestos de la investigación fueron cumplidos, durante el transcurso de su desarrollo, ha surgido como recomendación para futuras versiones de la aplicación desarrollada:

1. Exportar otros módulos ya implementados en soluciones de escritorio hacia la web por ejemplo el módulo de reportes, y el de edición.

Referencias Bibliográficas

1. **Ramos, José Enrique García.** *II curso de introducción al sistema operativo GNU/LINUX.* 2005.

2. **Jose, Acevedo Sánchez.** *Instrumentación y Control Básico de Procesos.* 2006.

3. **M.Romero, Ing Diego.** *Sistemas de Interfaz Humano Maquina(HMI).* 2007.

4. **Garayta, Ariel.** *WebUInt,Interfaz Hombre-Máquina del SCADA en la Web.* 2009.

5. **Rodrigo.** Artículos tecnológicos. [En línea] 2010. http://culturacion.com/2010/01/caracteristicas-de-una-aplicacion-web/.

6. QT creator. [En línea] Nokia Corporation, 2008-2010. http://qt.nokia.com/.

7. EXTJS. [En línea] http://www.extjs.com/.

8. genesis32. [En línea] 2009. http://www.iconics.com/products/genesis32.asp.

9. Trabajos de Sistemas de Control. [En línea] 2010. http://www.freedownloadmanager.org/es/downloads/Trabajos_de_Sistemas_de_Control_60815_p/.

10. **Ulrich, Frederico.** Breve Descripción del HMI 500. [En línea] 25 de 03 de 2008.

11. **David Tavárez .** Comparación de Frameworks en Javascript. [En línea] http://www.maestrosdelweb.com/editorial/comparacion-frameworks-javascript/.

12. Que son los widgets. [En línea] 11 de Marzo de 2010. http://software.suite101.net/article.cfm/que-son-los-widgets.

13. Sitio oficial JQuery. [En línea] 2010. http://jqueryui.com/.

14. Sitio oficial de Prototype. [En línea] 2006-2007. http://www.prototypejs.org/.

15. Sitio oficial de Dojo. [En línea] http://www.dojotoolkit.org/docs/.

16. slideshare.net. [En línea] 2009. http://www.slideshare.net/almarag/ext-js-y-frameworks-javascript..

17. OpenUP. *OpenUP.* [En línea] 16 de 11 de 2008. http://epf.eclipse.org/wikis/openup/...

18. Visual Paradigm for UML. [En línea] 05 de 2007. http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_

14720_p/.

19. *Manual de Javascript.* 2005.

20. **Pérez, Javier Eguíluz.** *Manual de Ajax.* 2008.

21. **Kiccillof, Carlos Reynoso – Nicolás.** *Estilos y Patrones.* UNIVERSIDAD DE BUENOS AIRES : s.n., 2004.

22. ARQUITECTURA Modelo Vista Controlador. [En línea] 20 de Agosto de 2007. http://exequielc.wordpress.com/2007/08/20/arquitectura modelovistacontrolador..

23. **Carreras, Carlos.** *Patrones de Diseño.* Universidad Politécnica de Madrid : s.n.

24. *Introducción al Módulo HMI. Modo de Ejecución.* **ALBA, Sistema Supervisor Guardián del.**

25. **Juristo, Natalia, Moreno, Ana M. y Vegas, Sira.** *Técnicas de Evaluación de Software.* 2006.

26. **Rosés, Francesc.** *JTABLE, TABLEMODEL Y RENDIMIENTO.* Julio 2004.

GLOSARIO DE TERMINOS

ALBA: Alternativa Bolivariana para las Américas.

API: Siglas de APPLICATION PROGRAM INTERFACE. El Interfaz para programas de aplicación es un conjunto de convenciones de programación que establecen cómo se invoca un servicio desde un programa.

CSS: Se explica que CSS son las siglas Cascading Style Sheets, que en español se traduciría por Hojas de Estilo en Cascada.

Cookies: Una **cookie** (pronunciado ['ku.ki]; literalmente *galleta*) es un fragmento de información que se almacena en el disco duro del visitante de una página web a través de su navegador, a petición del servidor de la página.

CORBA: (*Common Object Request Broker Architecture* — arquitectura común de intermediarios en peticiones a objetos), es un estándar que establece una plataforma de desarrollo de sistemas distribuidos facilitando la invocación de métodos remotos bajo un paradigma orientado a objetos.

DOM: Document Object Model, de sus siglas en ingles.

Framework: es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

GNU: Acrónimo que significa que GNU no es Unix además de ser el nombre del proyecto iniciado por Richard Stallman con el objetivo de crear un sistema operativo completamente.

HTTP: Protocolo de transferencia de hipertexto usado en cada transacción de la Web, representado por sus siglas en inglés HTTP (HyperText Transfer Protocol).

HTML: HyperText Markup Language, por sus siglas en inglés. Es un Lenguaje de Marcas de Hipertexto, muy usado para describir la estructura y el contenido en forma de texto de páginas Web.

IDE: Entorno de Desarrollo Integrado reconocido por sus siglas en inglés Integrated Development Environment (IDE).

PHP: Lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas, se puede encontrar embebido en las páginas HTML.

PYTHON: Lenguaje de programación creado por Guido van Rossum en el año 1990. Es comparado habitualmente con TCL, Perl, Scheme, Java y Ruby. En la actualidad Python se desarrolla como un proyecto de código abierto, administrado por la Python Software Foundation.

Plugin: Complemento o extensión, constituye una aplicación que se relaciona con otra para aportarle una funcionalidad nueva

UCI: Universidad de las Ciencias Informáticas.

XML (Extensive Markup Language, Lenguaje de Marcas extensible**)**:
El XML es considerado como un metalenguaje de definición de documentos estructurados mediante marcas o etiquetas. Se trata de un estándar del W3C cuyo objetivo es crear unas reglas básicas para permitir el intercambio de información estructurada entre aplicaciones, y en particular, entre aplicaciones web.

XSLT: (siglas de Extensible Stylesheet Language Transformations, lenguaje de hojas extensibles de transformación), que permite convertir documentos XML de una sintaxis a otra.