Universidad de las Ciencias Informáticas

Facultad 1

Título: Desarrollo del módulo de configuración del HMI web para el sistema de supervisión y control de equipamientos a distancia.

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

Autor(es): Julio César Cruz Membrado

Ramón Ortega Estévez

Tutor(es): Ing. Sucel Castro Castro

Ing. Yader Coca Ribas

Ciudad de La Habana, julio de 2010.
¨Año 52 de la Revolución¨

DECLARACIÓN DE AUTORÍA

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

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

_____	_____
Julio César Cruz Membrado	Ing. Sucel Castro Castro
Autor.	Tutora.
_____	_____
Ramón Ortega Estévez	Ing. Yader Coca Ribas.
Autor.	Tutor.

"Nunca consideres al estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber"

Albert Einstein

DATOS DE CONTACTO

Ing. Sucel Castro Castro: Graduada de Ingeniera en Ciencias Informáticas.

Correo electrónico: sccastro@uci.cu

Ing. Yader Luis Coca Ribas: Graduado de Ingeniero en Ciencias Informáticas.

Correo electrónico: ylcoca@uci.cu

AGRADECIMIENTOS

De Julio:

A mi familia, en especial a mi mamá Osnidia, por apoyarme siempre en todas mis decisiones.

A mi papá Julio, por nunca olvidarse de mí, aunque estuviera lejos.

A papi Osmani por guiarme bien por el camino de la vida.

A mi esposa Lily, por estar siempre ahí para mí y brindarme todo su amor y confianza.

A mi tía Carmita, a mi abuelita y a Tamara, que siempre lo dieron todo por verme graduado.

A mis suegros Lidia y Miguelito, por su exigencia constante y ser como mis propios padres.

A la Universidad de las Ciencias Informáticas por todas las oportunidades que me brindó.

A mis compañeros de estudio que compartieron conmigo a lo largo de la carrera.

De Ramón:

Es necesario dejar constancia, como un humilde homenaje de reconocimiento, por la ayuda brindada para realizar este trabajo a:

Todas las persona que de una manera u otra tuvieron confianza en que Sí se puede afrontar y vencer grandes retos. A mis amigos y compañeros que se manifestaron de muchas maneras y que nos alentaron y estimularon con sus palabras y acciones, a seguir adelante a pesar de las dificultades que la vida nos impone.

A mi familia y amigos, que se esmeraron en ayudarme y contagiarme con su energía positiva y con su entusiasmo necesario para cumplir esta meta y arribar a ella con la satisfacción del deber cumplido. Muy especial a mi familia que conoce cuanto sacrificio y rigor cuesta alcanzar elevados niveles de profesionalidad en el campo de la Informática. A mis mejores amigos en especial a: Rachel Olivera, Arley Fonseca, Sergio Cisneros, Jorgito Díaz, Orlando Pérez, Luis O Cuza, Yoandris Aroche que siempre me dieron su apoyo incondicional.

DEDICATORIA

De Julio:

Quiero dedicar este trabajo que es el resultado de todos mis estudios, a toda mi familia, en especial a mi madre, por ser lo más grande que existe y porque estoy seguro de que sin ella nunca lo hubiera logrado.

A una de las cosas más lindas del mundo: a ti mi Lily.

De Ramón:

A:

Mi familia, que siempre ha estado junto a mí, con plena confianza en mí, y además porque cada éxito obtenido lo siente como suyos, a mi mamá, papá, y hermana, y en primer lugar a mi madre, pues sin su amorosa, inagotable, paciente e ilimitada participación, no hubiera sido posible terminar con éxitos.

A:

La revolución, por conquistar la libertad y la dignidad plena de todos los cubanos, por darnos la oportunidad de estudiar y formarnos integralmente como futuros revolucionarios y oficiales de las FAR, con una elevada cultura de valores humanos y éticos.

RESUMEN

El presente Trabajo de Diploma surge a partir de la necesidad que existe en el Ministerio de las Fuerzas Armadas Revolucionarias de supervisar y controlar los equipamientos a distancia.

Para la realización del Trabajo de Diploma es de vital importancia conocer acerca de los sistemas SCADA (Sistemas de Supervisión, Control y Adquisición de Datos). Los SCADA proporcionan gran cantidad de datos que se entregan a usuarios fuera del ambiente de control con los que se realizan estudios de comportamientos, detección de irregularidades y toma de decisiones. Las aplicaciones SCADA que se comercializan en el mercado son sistemas propietarios con un alto costo para las empresas que los adquieren, a los clientes no se les proporcionan el código fuente de forma que no pueden adicionar funcionalidades ni modificar las existentes, además de todo esto la seguridad del sistema queda en manos del proveedor del software ya que los usuarios del mismo no tienen conocimiento del funcionamiento interno del programa. De aquí la necesidad de la realización de un sistema basado en código abierto, que cumpla con todos los requerimientos y necesidades existentes.

Se presenta el análisis, diseño e implementación de un configurador del módulo HMI (Human Machine Interface) web del SCADA, que permite la configuración y control de equipamientos a distancia en tiempo real.

Además incluye un estudio del estado del arte actual de este tipo de sistemas a nivel internacional y nacional, así como un análisis de las herramientas, metodologías, tendencias y técnicas para lograr una solución más óptima.

Palabras Claves

Control, configuración, despliegue, HMI, SCADA, supervisión.

TABLA DE CONTENIDOS

Introducción

Ante las agresivas declaraciones hechas sobre Cuba el primero de mayo de 2003 por el presidente norteamericano George W. Bush, quien situó a la Isla como uno de los 60 oscuros rincones del mundo donde podían realizar un golpe militar sin previo aviso; se tomó la decisión en un pleno del Partido celebrado en julio de ese propio año de incrementar la capacidad defensiva del país, lo cual incluyó un grupo de medidas, entre ellas la tarea Triunfo.

Fue en ese momento que el Ministro de las Fuerzas Armadas Revolucionarias le ordenó a la Industria Militar llevar a cabo la misión, que significó adentrarse en la producción y modernización de medios de combate. Era una actividad a desarrollar de manera cooperada con 40 organismos y entidades, pertenecientes al Ministerio de la Industria Sideromecánica (SIME), al Ministerio de la Industria Básica (MINBAS) y al Ministerio de Ciencias, Tecnología y Medio Ambiente (CITMA).

"Nos ha correspondido la misión de modernizar, perfeccionar y producir armamentos y técnica militar con altas cualidades combativas, como son la movilidad, el poder de fuego y la protección, entre otras, que se corresponden con nuestra doctrina militar de la Guerra de Todo el Pueblo".

"Dentro de este contexto, un impacto importante lo tienen las comunicaciones y la electrónica, para garantizar el mando ininterrumpido de las tropas y contrarrestar las acciones del enemigo en el terreno radioelectrónico". [13]

El Ministerio de las Fuerzas Armadas Revolucionarias (MINFAR), institución militar básica del Estado con la misión fundamental de combatir al agresor, está compuesta por diversas especialidades como: Defensa Antiaérea, Lucha Radioelectrónica, Logística, entre otras. Todas estas especialidades constan de una gran cantidad de armamento y técnica rusa que no se encuentra automatizado, lo que trae consigo que el trabajo con este armamento y técnica sea engorroso y se necesite más personal para trabajar con ellos. Además de una imperiosa necesidad de supervisar y controlar el equipamiento a distancia.

Teniendo en cuenta el análisis anteriormente expuesto, se generó la siguiente **Situación Problémica**: Los sistemas informáticos con que cuenta el MINFAR, presenta imposibilidades para supervisar y

controlar a distancia los equipamientos en tiempo real.
Teniendo en cuenta la situación antes planteada se presenta el siguiente **Problema Científico:** ¿Cómo lograr la configuración de un despliegue de un HMI desde la web para la supervisión y control de equipamientos a distancia?

El **Objeto de Estudio** que enmarca esta problemática es el proceso de configuración de la visualización de las Interfaces Hombre-Máquinas.

Este trabajo propone como **Objetivo General** desarrollar un configurador web para el sistema de supervisión y control de equipamientos a distancia.

Se ha decidido como **Campo de Acción** la configuración de despliegues desde un HMI web para el sistema de supervisión y control de equipamientos a distancia.

Como **Idea a defender** se tiene que si se implementa el submódulo ambiente de configuración del módulo HMI, se dará un paso importante en el desarrollo del sistema de supervisión y control en las FAR permitiendo así un mejor trabajo con la técnica y el armamento existente.

Para dar cumplimiento al objetivo planteado se definieron los siguientes **Objetivos Específicos**:

- Asimilar herramientas y tecnologías para el desarrollo de la aplicación.
- Realizar un estudio de otras Interfaces Hombre - Máquina existentes.
- Obtener la modelación del proceso de configuración de la visualización para el sistema de supervisión y control de equipamientos a distancia.
- Capturar los requisitos para el desarrollo de un sistema para la supervisión y control de equipamientos a distancia.
- Realizar análisis, diseño e implementación de un sistema para la supervisión y control de equipamientos a distancia.

Para la realización del trabajo se tienen en cuenta algunos métodos tradicionales investigativos. A continuación se mencionarán cada uno de ellos y de qué forma se ponen de manifiesto en la investigación.

Los **métodos teóricos** aplicados en la investigación son los métodos histórico-lógico puesto que se realizó un estudio de las tecnologías que existen actualmente para poder realizar la selección de las que se van a utilizar de acuerdo a las características propias del sistema a desarrollar y el método de análisis y síntesis, pues se realiza un análisis de la bibliografía que se utilizó para el estudio del tema.

Los **métodos empíricos** son los que describen y explican las características fenomenológicas del objeto, representan un nivel de la investigación cuyo contenido procede de la experiencia y es sometido a cierta elaboración racional.

Entre los métodos empíricos que se utilizan está el método de la observación. Con la aplicación del mismo se puede conocer la realidad mediante la percepción directa de los objetos y fenómenos; a través de este método se pudo conocer la esencia de la problemática definida, lo que ayudó al planteamiento del problema científico, además de permitir conocer el proceso definido como objeto de estudio, lo cual influye a la hora de tener un conocimiento más detallado de lo que se quiere, lo que hace falta hacer y cómo hay que hacerlo. Otro método empírico utilizado es la entrevista, nos posibilitó obtener información luego de conversaciones planificadas con las personas con experiencia del tema.

El documento quedará estructurado de la siguiente manera:

- **Capítulo 1:** Fundamentación teórica. Estado del arte.

Se definen los conceptos fundamentales sobre las tecnologías que se abordaron en la concepción del sistema. Se describen las herramientas y metodologías empleadas en la construcción del software, así como un análisis del estado del arte.

- **Capítulo 2:** Características del sistema.

Se realiza una descripción detallada de las características del sistema, se definen los requisitos

funcionales y no funcionales que guiarán el desarrollo del módulo de configuración.

- **Capítulo 3:** Diseño de la arquitectura de software.

En este capítulo se presentarán los diagramas de clases de diseño, donde se definen las responsabilidades de las clases y sus relaciones, los diagramas de secuencia y los patrones a utilizar.

- **Capítulo 4:** Implementación y pruebas.

Se codifican las funcionalidades y se realizan las pruebas para validar el código y detectar los errores.

Capítulo 1: Fundamentación Teórica. Estado del Arte.

1.1 Introducción

La necesidad de supervisar y controlar a distancia un proceso ha sido una premisa para el hombre moderno. Los primeros mecanismos de supervisión eran sencillos sistemas de telemetría que solo proporcionaban reportes de las condiciones de campo y brindaban parámetros de las unidades remotas. La llegada de nuevos avances tecnológicos proporcionó cada vez mejores soluciones a los distintos problemas. Las computadoras con mayor capacidad de cálculo son capaces de realizar tareas más complejas permitiendo adicionar gran cantidad de funcionalidades que hacen los sistemas más completos.

En este capítulo se realiza una fundamentación teórica y un estudio del estado del arte del tema, así como una explicación detallada sobre las metodologías y herramientas utilizadas para la concepción del módulo de configuración.

1.2 Características de los sistemas SCADA

El objetivo principal de la automatización es disminuir la intervención humana en la ejecución de los procesos de producción. Con vista a cumplir esa exigente meta se han desarrollado en los últimos años algunos sistemas de Supervisión, Control y Adquisición de Datos (SCADA) [1], utilizados para la gestión y el control de procesos industriales. La presencia de estos sistemas aumenta los niveles de eficiencia, minimiza los costos y optimiza los procesos de producción.

El término SCADA usualmente se refiere a un sistema central que monitorea y controla un sitio completo o un sistema que se extiende sobre una gran distancia (kilómetros / millas). La instalación de un sistema SCADA necesita de sensores, actuadores, controladores, redes, comunicaciones, base de datos, Interfaz Hombre-Máquina (HMI, por sus siglas en inglés), entre otros. [1]

El sistema permite 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. Además, provee de toda la información que se genera en el proceso productivo a diversos usuarios.

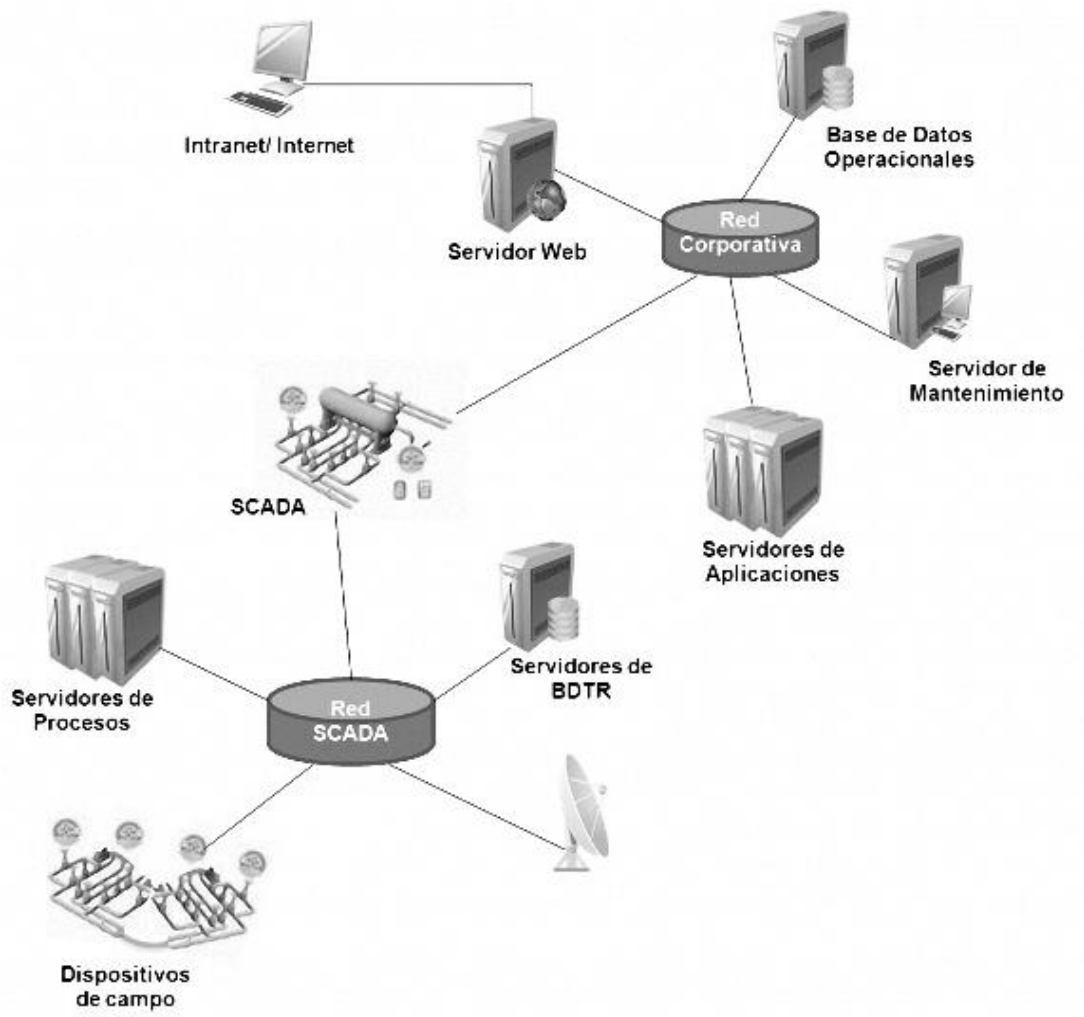

Figura 1: Representación de un sistema SCADA

La mayoría de los sistemas SCADA, debido a su magnitud y complejidad están formados por diferentes módulos o subsistemas, como son los de: procesamiento de datos, base de datos, manejadores y HMI. Dentro del módulo HMI encontramos dos sub- módulos: el ambiente de

configuración y el ambiente de ejecución, que están en estrecha relación con los demás módulos ya que son los encargados de la interacción del usuario con el sistema.

Algunas de las industrias que utilizan los sistemas SCADA son en las de gestión de recursos hidráulicos, energía eléctrica, señales de tráfico, sistemas de control ambiental, control de oleoductos, redes de distribución de gas natural y en sistemas de fabricación, entre otras. Pero esto es sólo a modo ilustrativo, pues existen escenarios muy variados donde se hace necesario la supervisión, el control de procesos y la accesibilidad de forma inmediata a la información de una manera rápida, confiable, segura y con el mínimo coste posible.

Otros beneficios que brindan estos sistemas al personal autorizado de la organización, donde se ponga en marcha el sistema, es el acceso a los comportamientos generales del sistema, como estadísticas del proceso, detección de fugas y la representación de los sinópticos gráficos configurados en los despliegues. Además, que el acceso al sistema pudiera realizarse desde instalaciones distantes de las estaciones de trabajo donde se encuentra instalado el sistema. Donde no se dispone en muchas ocasiones con los medios necesarios para acceder al sistema desde un ordenador central que tenga las aplicaciones necesarias para visualizar un sistema SCADA, se cuenta con dispositivos de baja capacidad de procesamiento, como puede ser: un teléfono móvil con navegación Web, un dispositivo portátil con un sistema operativo embebido; o un ordenador sin ningún software de sistemas SCADA, sin importar el sistema operativo.

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

Interfaz Hombre - Máquina o HMI (Human Machine Interface)

El término Interfaz Hombre-Máquina es usado frecuentemente en el contexto de los Sistemas de Computación y los Sistemas Electrónicos, constituye una capa intermedia que los independiza y permite la intercomunicación entre ambas partes.

Existen en la actualidad diferentes tipos de HMI, se pueden identificar claramente dos de ellas: Interfaces Gráficas de Usuario (GUI) y las Interfaces de Usuario basadas en la Web. Existen otra variedad de interfaces que actualmente se utilizan en menor escala, por ejemplo: interfaces basadas en líneas de comandos, táctiles, basadas en gestos, multi-pantallas e interfaces basadas en texto.

El módulo de HMI en el SCADA se encarga de representar, en un ordenador, los procesos que ocurren en el campo, muestra los componentes implicados, los sensores, las estaciones remotas, y el sistema de comunicación dándole al operador total control. Éste módulo es el que permite al operador estar en contacto directo con el sistema, realizar la supervisión y el control del proceso en general.

Está compuesto por dos partes fundamentales: el ambiente de configuración o editor y el ambiente de ejecución.

El ambiente de configuración permite configurar varios procesos o partes de ellos, aquí se definen y gestionan las variables, 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.

El ambiente de ejecución se encarga de visualizar las animaciones y los objetos definidos en el editor, muestra lo que está ocurriendo en el campo en tiempo real, es el que envía los comandos a

las estaciones remotas, quién recibe los valores de las variables, interactúa con la mayoría de los operadores pues se emplea para supervisar el proceso de manera directa. Al ser el módulo que se encarga de brindar el control total sobre el proceso de producción, la interfaz de usuario brinda un conjunto de funcionalidades primarias, entre ellas la generación de reportes, impresión, análisis de variables, visualización de la tendencia de indicadores, configuración de los manejadores para la comunicación y acceso a las alarmas. Los HMI representan las actividades que se pueden realizar en un SCADA mediante los despliegues; los despliegues están formados por sinópticos gráficos que toman la forma de los objetos del campo.

Tienen vínculos con bases de datos para proporcionar a los gráficos de tendencias, datos de diagnóstico y manejo de la información así como un cronograma de procedimientos de mantenimiento, información logística, esquemas detallados para un sensor o máquina en particular, incluso sistemas expertos con guía de resolución de problemas.[2]

1.2.2 ¿Qué es un HMI 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 la 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)*, 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.

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.

El módulo de HMI Web para sistemas SCADA está compuesto un cliente Web y un servidor HMI Web, que se comunican haciendo uso del protocolo de comunicación HTTP (*HyperText Transfer Protocol*). Éste módulo brinda las funcionalidad convencionales de un SCADA, permite a los usuarios autorizados estar en contacto directo con el sistema, realizar la supervisión, la adquisición de datos y el control del proceso 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 acceder a 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.

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.

Los servidores HMI Web gestionan la información que le llega desde los dispositivos de campo, el manejo de eventos y crean las representaciones de los sinópticos que luego se visualizará en el cliente Web. El servidor es el encargado de crear, modificar y eliminar 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 de los servidores 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.

1.2.3 Ventajas y desventajas de un HMI Web.

Ventajas:

- No requieren instalación, pues usan 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 en desde cualquier regionalización 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.

Desventajas:

- 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.

1.3 Sistemas SCADA en el mercado

En la actualidad los sistemas SCADA tienen buena aceptación en el mercado. Podemos mencionar algunos de estos sistemas:

- La compañía USDATA [10] ofrece el producto **Factory Link 7**. Esta solución SCADA para recolectar información crítica de los procesos fue diseñada específicamente para MS Windows 2000 bajo la plataforma multicapa de DNA. Utiliza la tecnología estándar de objetos para la importación de datos externos, con lo que se reduce el costo de propiedad de los sistemas.

  Muchas de las funcionalidades típicas en un ambiente de manufactura ya se encuentran pre construidas y almacenadas en una biblioteca para que el usuario desarrolle aplicaciones en tiempo récord.
- La compañía Advantech [11] ofrece el **Paradym-31** que provee un ambiente gráfico de programación compatible con MS Windows, que permite construir programas de control en tiempo real, tales como los tradicionales Controladores Lógicos Programables.

  Este software es capaz de brindar una solución completa de automatización. Por ser compatible con la norma IEC 1131-3 se reduce significativamente el costo de programación y entrenamiento. El usuario puede construir sus propias funciones lógicas y generar reportes automatizados especiales.
- Nematron ofrece el **SCADA Paragon**, software poderoso y flexible, permite construir aplicaciones para una completa visualización del operador, MMI, supervisión de control y adquisición de datos (SCADA).

  Debido a que las funciones para reparación de errores se encuentran integradas en los módulos de control, HMI y SCADA, todas ellas comparten una sola base de datos, facilitando así la programación y localización de errores. La misma base de datos creada para el sistema de control se usa para configurar entradas y salidas, pantallas de operador, adquisición de datos y otras aplicaciones. Se programa utilizando diagramas de flujo eficientes integrados al

popular lenguaje de escalera. Soporta las normas abiertas como OPC, ActiveX, COM/DCOM, etcétera, e incluye capacidades avanzadas de diagnóstico, por lo que se facilita el mantenimiento y la capacitación del personal técnico.[14]

- La compañía SIEMENS presenta el **HYBREX** (HybrexExpertSystem), herramienta de simulación que permite realizar cambios virtuales en la planta y observar sus resultados sin ningún riesgo. Está específicamente orientada a procesos de laminado en plantas siderúrgicas y se puede utilizar en cualquiera de las etapas del ciclo de vida de la planta, desde construcciones nuevas hasta plantas en procesos de optimización y modernización.[14]

En el estudio realizado de estos y otros sistemas, se puede observar que no resuelven los problemas que se presentan pues son altamente propietarios, además de estar especializados en un área determinada.

1.4 Metodología, tecnología y herramientas utilizadas en la solución.

1.4.1 Metodología Orientada a Procesos

La Metodología Orientada a Procesos (MOP) constituye que una nueva metodología para el análisis, implementación y documentación de sistemas orientados a objetos, creada con el objetivo de producir un software más robusto, predecible, reutilizable y de fácil mantenimiento. Para la realización del presente Trabajo de Diploma se hace uso de esta metodología creada en la Unidad de Compatibilización, Integración y Desarrollo de software para la defensa (UCID).

La producción de software engloba dos áreas de procesos perfectamente definibles: los procesos propios de la construcción del producto de software y las bases de su gestión.

En la MOP el ciclo de vida del proyecto se considera como parte del ciclo de vida del producto, el cual representa un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en la adquisición, el desarrollo, la explotación y el mantenimiento de un producto.

El ciclo de vida de la MOP se descompone en el tiempo en cinco fases secuenciales que son: Inicio, Modelación, Construcción, Explotación Experimental, Despliegue. Al final de cada fase los representantes de los grupos de roles presentes en el proyecto realizan una evaluación para determinar si los objetivos se cumplieron y así presentar a Consejo Técnico Formal para su evaluación y dar paso o no a la fase siguiente.

Figura 2: Etapas del ciclo de vida del proyecto

Las principales características del ciclo de vida son:

- Las fases son secuenciales y su transferencia debe ser precedida por un proceso de revisión o liberación del Centro de Calidad y su aprobación en Consejo Técnico Formal.

- El nivel del personal es bajo al comienzo, alcanza su nivel máximo en la fase de construcción y decae rápidamente cuando el proyecto se aproxima a su conclusión.
- La participación de los interesados es alta en las etapas de Inicio y Modelación, baja en la etapa de Construcción y vuelve a subir en las etapas finales del proyecto.

1.4.2 Descripción de las fases

En la **fase de Inicio** se logra una visión preliminar de la problemática a resolver, se identifica el alcance preliminar del proyecto, se especifican los involucrados y las líneas de desarrollo ejecutoras del proyecto constituyéndose el equipo de desarrollo y se estiman los recursos necesarios que deberán ser asignados al mismo.

En la **fase de Modelación** se definen los procesos del dominio del problema, se estiman los principales riegos que presenta el proyecto y se especifica la forma de mitigarlos, se identifican las necesidades del usuario de las que se derivan los requerimientos del producto a desarrollar, se determina la factibilidad operativa, técnica y/o económica de continuar el proyecto.

Se define la arquitectura del software, determinando las iteraciones (subsistemas y módulos) en las que se dividirá el producto y la forma de construcción, se valida y establece para disponer de cimientos sólidos sobre los que se basará el grueso del esfuerzo durante la fase de Construcción. Se realiza la planificación detallada de la siguiente fase.
Los principales hitos de esta fase son:

- Análisis de las necesidades del usuario.
- Especificación de la Arquitectura.
- Planeamiento de la construcción del software.

En la **fase de Construcción** se deben aclarar los requisitos restantes y completar el desarrollo del sistema sobre una base estable de la arquitectura. En esta fase todas las características, componentes y requerimientos deben ser integrados, implementados y probados en su totalidad, obteniendo una versión estable del producto comúnmente llamada versión beta. Se realiza la planificación detallada de la siguiente fase.

Los principales hitos de esta fase son:

- Versión Funcional del producto (Versión Beta).
- Manuales de usuario y de instalación.
- Liberación de calidad del producto.
- Planeación de la Explotación Experimental.

La **fase de Explotación Experimental** tiene como propósito asegurar que el software está disponible para realizar las pruebas de aceptación por un grupo de usuarios. Incluye las pruebas del producto como parte de su preparación para ser entregado, y la realización de ajustes en respuesta a la retroalimentación recibida de los usuarios. En este punto del ciclo de vida la retroalimentación de los usuarios debe enfocarse fundamentalmente en ajustes específicos y de corto alcance al producto junto a otros temas como configuración, instalación, y usabilidad.

Los principales hitos de esta fase son:

- Solución Estable.
- Planeamiento para el despliegue.
- Aceptación de los resultados.

En la **fase de Despliegue** se realiza la generalización del producto en las entidades y órganos según lo aprobado en el Cronograma de implantación. Durante el proceso de implantación por lo general no es necesaria la participación de los integrantes del equipo de desarrollo.

1.4.3 Roles involucrados en el proceso

Los roles son una definición abstracta que especifican el comportamiento y las responsabilidades de un individuo, o de un grupo de individuos trabajando juntos como un equipo. Una persona puede desempeñar diversos roles, así como un mismo rol puede ser representado por varias personas. Sus responsabilidades abarcan tanto el llevar a cabo un

conjunto de actividades como responder por la elaboración de un conjunto de artefactos. Además describen cómo los individuos deberán comportarse en el contexto de un proyecto. [4]

Analistas: agrupa los roles que están involucrados fundamentalmente en la identificación y descripción de los procesos de negocio y extracción y especificación de los requisitos del software. Este grupo está formado por los siguientes roles:

- **Analista del proceso de negocio:** responsable de definir la arquitectura del negocio, los procesos, su descripción y propuestas de mejora. Además es responsable de detallar la especificación de la organización o parte de ella.
- **Analista del sistema:** dirige y coordina el proceso de extracción de requisitos y especifica los detalles de cada una de las funcionalidades del sistema.

Desarrolladores: organiza los roles que están involucrados fundamentalmente en el diseño de la arquitectura, el diseño detallado y la implementación del software. En este grupo están los roles:

- **Desarrollador de interfaz de usuario:** coordina el diseño de la interfaz de usuario, utiliza los requisitos de usabilidad y crea prototipos candidatos de interfaz de usuario de acuerdo a ellos. Encargado de realizar el trabajo artístico que requiera el proyecto (iconos, pantalla de splash, gráficos, CSS, etc.).
- **Diseñador:** responsable del diseño de parte del sistema, dentro de los límites de: los requisitos, la arquitectura, y el proceso de desarrollo del proyecto.
- **Diseñador de BD:** dirige el diseño de la estructura de almacenamiento de datos persistentes que se utilizará en el sistema.

1.4.4 Tecnologías y herramientas

La arquitectura de software selecciona, combina e integra las tecnologías informáticas existentes para construir un software teniendo en cuenta los requisitos funcionales y no funcionales del cliente.

Las herramientas definidas por la arquitectura para la solución se describen a continuación:

1.4.4.1 Herramienta Case

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como realizar un diseño del proyecto, cálculo de costes, implementación de parte del código a partir del diseño dado, compilación automática y documentación o detección de errores.

Visual Paradigm for UML 6.4

Es una herramienta CASE, desarrollada por la compañía Visual Paradigm International, que utiliza UML como lenguaje de modelado.

- Presenta las siguientes características:
  - Soporta el ciclo de vida completo del software: análisis, diseño, implementación y despliegue.
  - Permite la captura de requisitos, el dibujo de diagramas UML, la realización de ingeniería inversa y generación de código PHP.

- Se integra con las siguientes herramientas:
  - Eclipse/IBM WebSphere
  - Builder
  - Net Beans IDE
  - Oracle JDeveloper
  - BEA Weblogic

- Está disponible en varias ediciones, cada una destinada a necesidades específicas: Enterprise, Professional, Community, Standard, Modeler y Personal.

La versión empleada en la solución es el Visual Paradigm for UML 6.4, lanzada el 20 de Octubre de 2008. Esta versión incluye siete nuevas mejoras:

1. Importa diagramas de Microsoft Office Visio a Visual Paradigm.
2. Soporta patrones de diseño.
3. Soporta las tres formas del Diagrama Entidad Relación, conceptual, lógica y física.
4. Mejora la trazabilidad de elementos utilizando el historial de revisiones.
5. Soporta líneas anidadas.
6. Muestra como elemento estereotipado del modelo un ícono de imagen.
7. Muestra y oculta elementos del diagrama según se desee. [5]

1.4.4.2 Lenguaje de modelado

UML 2.0

UML (Unified Modeling Language) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. Prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan [8]. UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. UML es una consolidación de muchas de las notaciones y conceptos más usados orientados a objetos. Actualmente se publicó la versión 2.0, que proporciona a los analistas, arquitectos y desarrolladores; herramientas cada vez más potentes que les posibilita aprovechar mejor los modelos y generar así una mayor cantidad de código reduciendo en gran medida el ciclo de desarrollo de sus aplicaciones.

1.4.4.3 Tecnología del lado del servidor

PHP 5.2.5

PHP es un acrónimo recursivo que significa PHP Hypertext Pre-processor fue creado por Rasmus Lerdorf, es un lenguaje interpretado de propósito general, ampliamente usado y que está diseñado especialmente para el desarrollo web, para la programación de servidores web de código abierto, es un lenguaje de alto nivel, cuyas órdenes se incrustan entre las etiquetas de las páginas web que han sido escritas en HTML. PHP es software libre esto significa que puede descargarse de Internet, modificar su código, copiarse, etc. sin ningún costo. Aunque es un lenguaje multiplataforma, ha sido

concebido para entornos UNIX y es en este sistema operativo donde se pueden aprovechar mejor sus prestaciones.

PHP ofrece un sinfín de funciones para la explotación de bases de datos de una manera llana y sin complicaciones. Brinda interfaces para el acceso a la mayoría de las bases de datos comerciales tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite

A continuación se muestra una pequeña lista de algunas de sus funcionalidades:

- Tratamiento de imágenes
- Gestión de archivos
- Gestión de bases de datos
- Funciones de correo electrónico
- Gestión de archivos PDF, Flash

1.4.4.4 Tecnologías del lado del cliente

JavaScript

JavaScript se implementó por primera vez en la versión beta de Netscape Navigator 2.0 en junio de 1995, por la empresa Netscape Communications Corporation. Es un lenguaje de programación interpretado utilizado en la realización de páginas Web. Una de sus principales características es que es orientado a objetos.

La verdadera fuerza de JavaScript es la compatibilidad con los distintos navegadores, incluso con los más anticuados. Además de que es interpretado por todos los navegadores modernos. [6]

Frameworks

Un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Proporciona una estructura que fuerza al desarrollo de código más legible. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio y permite separar en capas la aplicación.

ExtJS 2.2

ExtJS 2.2 es un framework de presentación completamente OO (Orientado a Objetos) de JavaScript, es multiplataforma y hace uso de la tecnología AJAX (Asynchronous JavaScript and XML). Este framework brinda múltiples posibilidades para el trabajo con las validaciones y manejo de errores en el cliente. Además permite la personalización de temas de estilos y provee el trabajo con una amplia configuración e intenso trabajo con las hojas de estilo CSS (Cascade Style Sheet). Basa toda su funcionalidad en JavaScript a través de librerías YUI, jQuery, o haciendo uso de la librería nativa, así en tiempo de ejecución carga y crea todos los objetos HTML a través del uso intenso de DOM (Document Object Model). Cuenta con dos licencias, una comercial y otra Open Source.

Ventajas:

- Posee una gran cantidad de widgets.
- El diseño está completamente separado de la funcionalidad dándole más flexibilidad.
- Funciones comunes fáciles de implementar, como validaciones, ventanas (con minimizar y maximizar), grillas editables lo que lo hace muy parecido a aplicaciones de escritorio.
- Posee una amplia documentación y comunidad de desarrollo.

Desventajas:

- El tiempo de aprendizaje es similar al de aprender a programar en un nuevo lenguaje.

Navegador

Mozilla Firefox 3.0

Mozilla Firefox es un navegador web libre descendiente de Mozilla Application Suite, desarrollado por la Corporación Mozilla, la Fundación Mozilla y un gran número de voluntarios externos.

Incluye navegación por pestañas, corrector ortográfico, búsqueda progresiva, marcadores dinámicos, un administrador de descargas y un sistema de búsqueda integrado que utiliza el motor de búsqueda

que desee el usuario. Además se pueden añadir funciones a través de complementos desarrolladas por terceros.

1.4.4.5 Software Libre

Es todo software cuyo "código fuente", es conocido y además puede ser difundido libremente. Otro aspecto importante es que el autor de dicho software lo puede publicar y permitirle a cualquier persona o institución la modificación o el simple uso de su obra. Existen dos formas de distribuir el software, una es bajo las llamadas licencias sin restricciones de tipo BSD y la otra es bajo el uso de la GPL de la Free Software Foundation que plantea que se puede modificar el software pero sí y sólo sí, el nuevo producto mantiene las mismas condiciones de uso y licencia del original y a su vez siga estando libremente al alcance de los que lo deseen.

Distribución de Linux: Debian GNU/Linux

Debian GNU/Linux es un sistema operativo libre, desarrollado por más de mil voluntarios alrededor del mundo, que colaboran a través de Internet. La dedicación de Debian al software libre, su base de voluntarios, su naturaleza no comercial y su modelo de desarrollo abierto la distingue de otras distribuciones del sistema operativo GNU. [9]

La combinación de la filosofía y metodología de Debian, las herramientas GNU, el núcleo Linux, y otro software libre importante, forman una distribución de software única llamada Debian GNU/Linux. Esta distribución está formada por un gran número de paquetes. Cada paquete en la distribución contiene ejecutables, scripts, documentación e información de configuración, y tiene un encargado, quien es el principal responsable de mantener el paquete actualizado, hacer un seguimiento de los informes de fallo y comunicarse con los autores principales del programa empaquetado.

La atención que pone Debian a los detalles, permite producir una distribución de alta calidad, estable y escalable. La instalación puede configurarse fácilmente para cumplir diversas funciones, desde cortafuegos reducidos al mínimo, a estaciones de trabajo científicas o servidores de red de alto rendimiento.

Debian es especialmente popular entre los usuarios avanzados debido a su excelencia técnica y a sus comités siempre atentos a las necesidades y expectativas de la comunidad Linux. Debian también introdujo muchas características a Linux, que ahora son comunes.

Por ejemplo, Debian fue la primera distribución de Linux en incluir un sistema de gestión de paquetes para una fácil instalación y desinstalación del software. Además, también fue la primera que podía actualizarse sin necesidad de reinstalarla.

Debian continúa siendo líder en el desarrollo de Linux. Su proceso de desarrollo es un claro ejemplo de lo bien que puede funcionar el modelo «Open Source»; incluso para tareas tan complejas, como construir y mantener todo un sistema operativo.

Lo que más distingue a Debian de otras distribuciones GNU/Linux es su sistema de gestión de paquetes. Estas herramientas otorgan al administrador de un sistema Debian total control sobre los paquetes instalados, incluyendo la capacidad de instalar un sólo paquete o actualizar el sistema operativo por completo. También es posible proteger paquetes individualmente de forma que no se actualicen. También puede indicar al sistema de gestión de paquetes qué programas ha compilado usted mismo y qué dependencias cumplen.

Para proteger el sistema contra “caballos de Troya” y otros programas malévolos, los servidores de Debian verifican que los paquetes provienen de sus auténticos encargados. Los desarrolladores de Debian también ponen gran cuidado en configurarlos de forma segura. Se publican parches muy rápidamente si se descubren problemas de seguridad en los paquetes ya distribuidos. Con el sencillo sistema de actualización de Debian, puede descargar e instalar parches de seguridad automáticamente a través de Internet. [9]

Entorno de desarrollo

Zend Studio for Eclipse 6.0

Zend Studio es un entorno integrado de desarrollo para la programación en PHP. Forma parte, junto con Zend Platform de lo que la empresa Zend Technologies propone para el desarrollo Web, Zend Studio para la parte cliente y Zend Platform para la parte del servidor. Esta herramienta no requiere instalación previa de PHP, ni de ningún otro entorno de ejecución Java. Ha

establecido el autocompletamiento de código, el plegado de código, inserción automática de paréntesis y corchetes, detección de errores de sintaxis en tiempo real y funciones de depuración. Posee también un manual de PHP integrado y un cliente FTP integrado. El programa entero está escrito en el lenguaje de programación Java, lo que a veces supone que no funcione tan rápido como otras aplicaciones de uso diario. Sin embargo, esto ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para sistemas operativos como Windows, Linux y Macintosh.

Programación Orientada a Objetos

La Programación Orientada a Objetos (POO u OOP según siglas en inglés) es un paradigma de programación que define los programas en términos de "clases de objetos", objetos que son entidades que combinan estado (datos), comportamiento (procedimientos o métodos) e identidad (propiedad del objeto que lo diferencia del resto). La programación orientada a objetos expresa un programa como un conjunto de estos objetos, que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener y reutilizar. Según Grady Booch, es un “método de implantación en el que los programas se organizan como colecciones cooperativas de objetos, cada uno de los cuales representa una instancia de alguna clase, y cuyas clases son miembros de una jerarquía de clases unidas mediante relaciones de herencia. En tales programas, las clases suelen verse como estáticas, mientras que los objetos suelen tener una naturaleza mucho más dinámica, promovida por la existencia de la ligadura dinámica y el polimorfismo”. La programación orientada a objetos es una forma de programar. Introduce nuevos conceptos, que superan y amplían conceptos antiguos ya conocidos.

1.5 Conclusiones

Se ha hecho mención de los principales conceptos que se abordarán a lo largo de todo este trabajo de diploma. Se profundizó además en las tecnologías y las herramientas que se utilizarán en la solución, debido a las ventajas que estas presentan y a la arquitectura definida en los proyectos desarrollados en la Unidad de Compatibilización, Integración y Desarrollo de Software para la Defensa (UCID).

Capítulo 2: Características del Sistema.

2.1 Introducción

En el presente capítulo se realiza una descripción detallada de las características del sistema, se definen los requisitos funcionales y no funcionales que guiarán el desarrollo del módulo de configuración. Se explica detalladamente el funcionamiento del mismo.

2.2 Objetivos estratégicos de la organización

El Centro de la Modernización de la Técnica y el Armamento en la UCID presenta los siguientes objetivos estratégicos:

- Construir soluciones de software que contribuyan a la modernización de la técnica y el armamento militar y al aumento de la disposición combativa.
- Estandarizar el desarrollo de software en el marco de las Comunicaciones, la Automática y la Instrumentación Virtual.
- Compatibilizar e integrar los avances tecnológicos obtenidos con las instituciones que desarrollen temáticas afines a la Automática y las Telecomunicaciones (fundamentalmente la UCI y los Centros de Investigación y Desarrollo de las FAR).

El presente trabajo de diploma está enmarcado en el primer objetivo estratégico.

Causas que originan la situación problémica

Actualmente los procesos para controlar los equipamientos a distancia son escasos, y los sistemas existentes en las FAR usan una tecnología atrasada que no está a la altura de las circunstancias y los tiempos que corren. Por lo que se genera la siguiente situación problémica: los sistemas informáticos con que cuenta el MINFAR presenta imposibilidades para supervisar y controlar los equipamientos a distancia y en tiempo real. A raíz de esta problemática se realiza la siguiente propuesta de solución:

2.3 Propuesta de sistema

Para la realización de los despliegues los sistemas SCADA necesitan de un configurador, el cual permita realizar una configuración de todos los elementos necesarios para el mismo. Con la solución planteada debe funcionar de la siguiente manera:

Primero: El configurador contará con un mecanismo de seguridad para el acceso, evitando el uso indebido por personas no autorizadas.

Segundo: Después de agregado un componente al despliegue se debe permitir establecer una serie de valores para sus atributos que se adapten a las necesidades requeridas.

Tercero: Una vez realizada la configuración del despliegue dará la posibilidad de que esta sea manipulada por el módulo pertinente del SCADA.

2.4 Procesos de negocio

A continuación se muestran los procesos del negocio identificados para el desarrollo del módulo de configuración.

- Despliegue.
    - Creación de despliegue.
    - Configuración de despliegue.

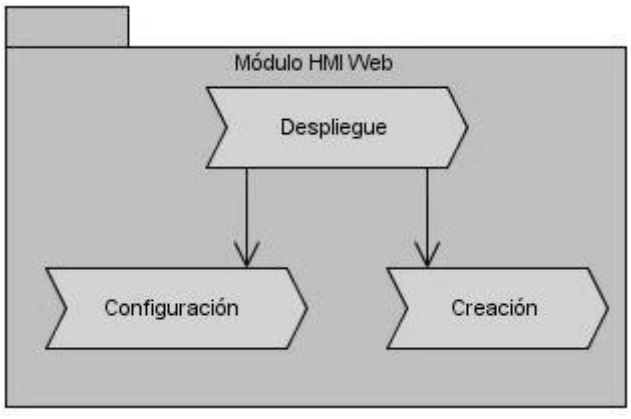

Figura 3: Mapa general de procesos.

2.5 Modelación de los procesos de negocio

La modelación de proceso de negocio permite realizar una exploración del dominio del problema, con el fin de lograr comprensión por parte del equipo de desarrollo de los procesos que se realizan actualmente en la entidad y la relación que existe entre estos. [4]

2.5.1 Creación de despliegue

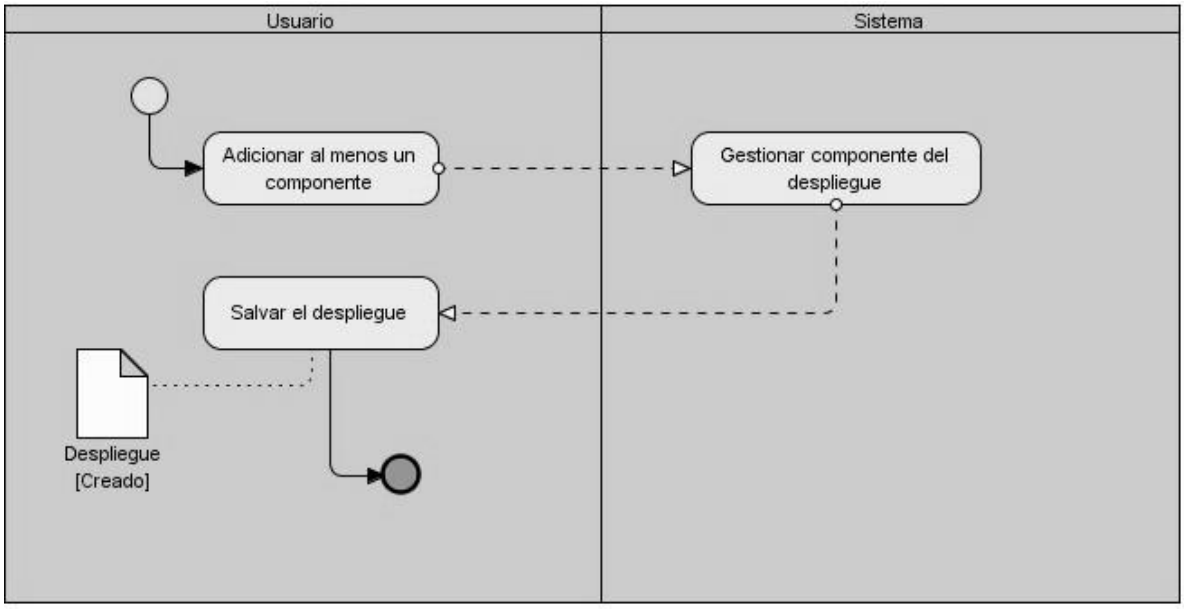

Figura 4: Proceso Creación de despliegue.

2.5.2 Configuración de despliegue

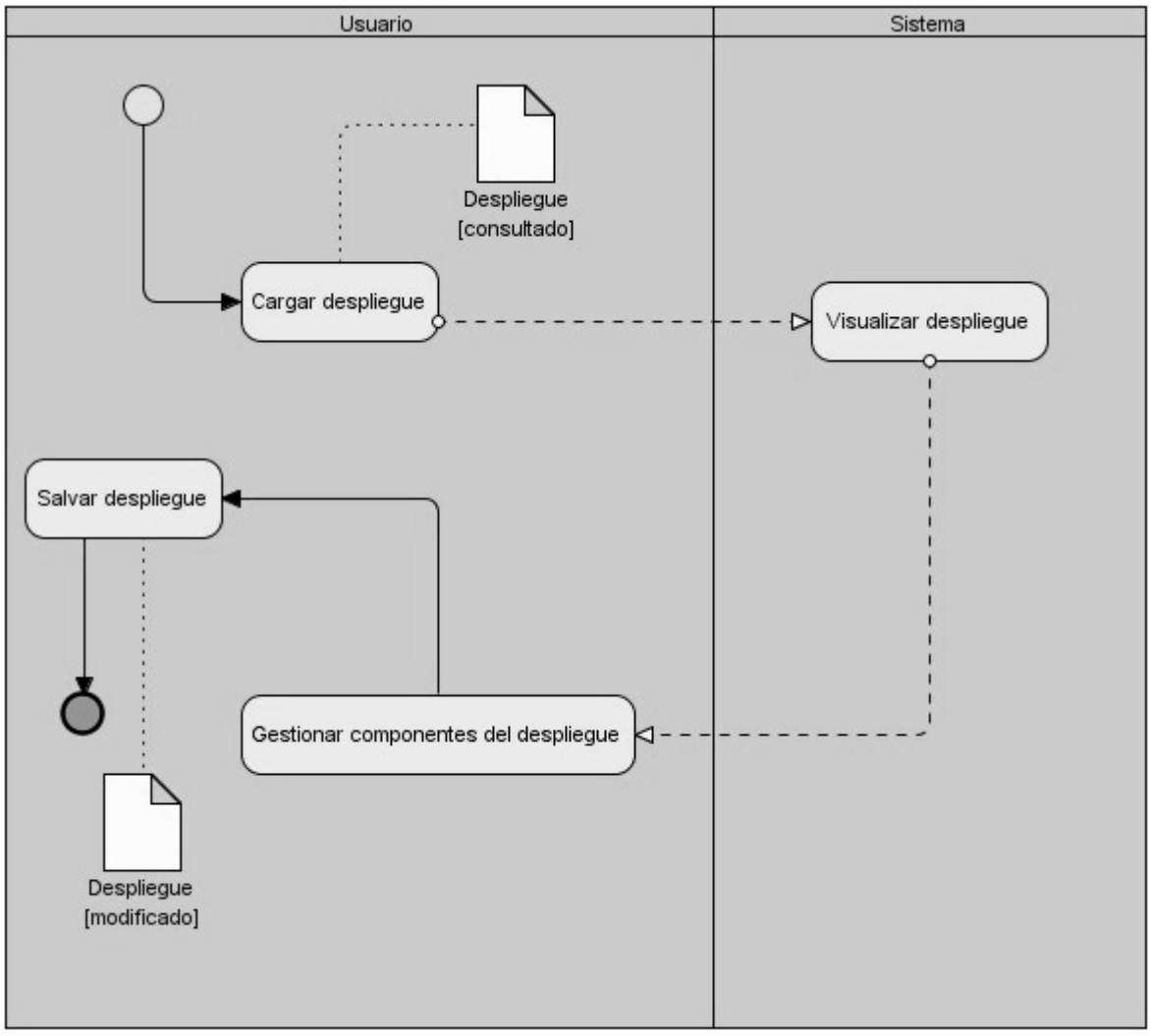

Figura 5: Proceso Configuración de despliegue.

2.6 Descripción de los procesos de negocio

2.6.1 Creación de despliegue

Objetivos	Crear un nuevo despliegue donde se puedan gestionar sus componentes. Permitir salvar el despliegue creado.
Evento(s) que lo generan	No procede.
Precondiciones	El usuario debe tener los privilegios para realizar esa acción.
Poscondiciones	Adicionar al menos un componente para que se pueda conformar el despliegue.
Marco jurídico	No procede.
Clientes internos	No procede.
Clientes externos	No procede.
Entradas	No procede.
Salidas	Un despliegue.

Tabla 1: Descripción del proceso Creación de despliegue.

2.6.2 Configuración de despliegue

Objetivos	Cargar despliegue para gestionar sus componentes. Salvar despliegue con las modificaciones realizadas.
Evento(s) que lo generan	No procede.
Precondiciones	El usuario debe tener los privilegios para realizar esa acción.
Poscondiciones	No procede.
Marco jurídico	No procede.
Clientes internos	No procede.
Clientes externos	No procede.
Entradas	Un despliegue realizado previamente para poder configurarlo.
Salidas	Un despliegue modificado.

Tabla 2: Descripción del proceso Configuración de despliegue.

2.7 Modelo conceptual

Como parte de la investigación del dominio del problema se desarrolla el modelo conceptual, el cual permite representar cada uno de los conceptos significativos del dominio del problema (cosas del mundo real), las relaciones existentes y los atributos que los componen. En la figura 6 se muestra una representación gráfica del modelo a través de un diagrama de clases UML, y luego se explican cada uno de sus conceptos y los atributos de estos conceptos.

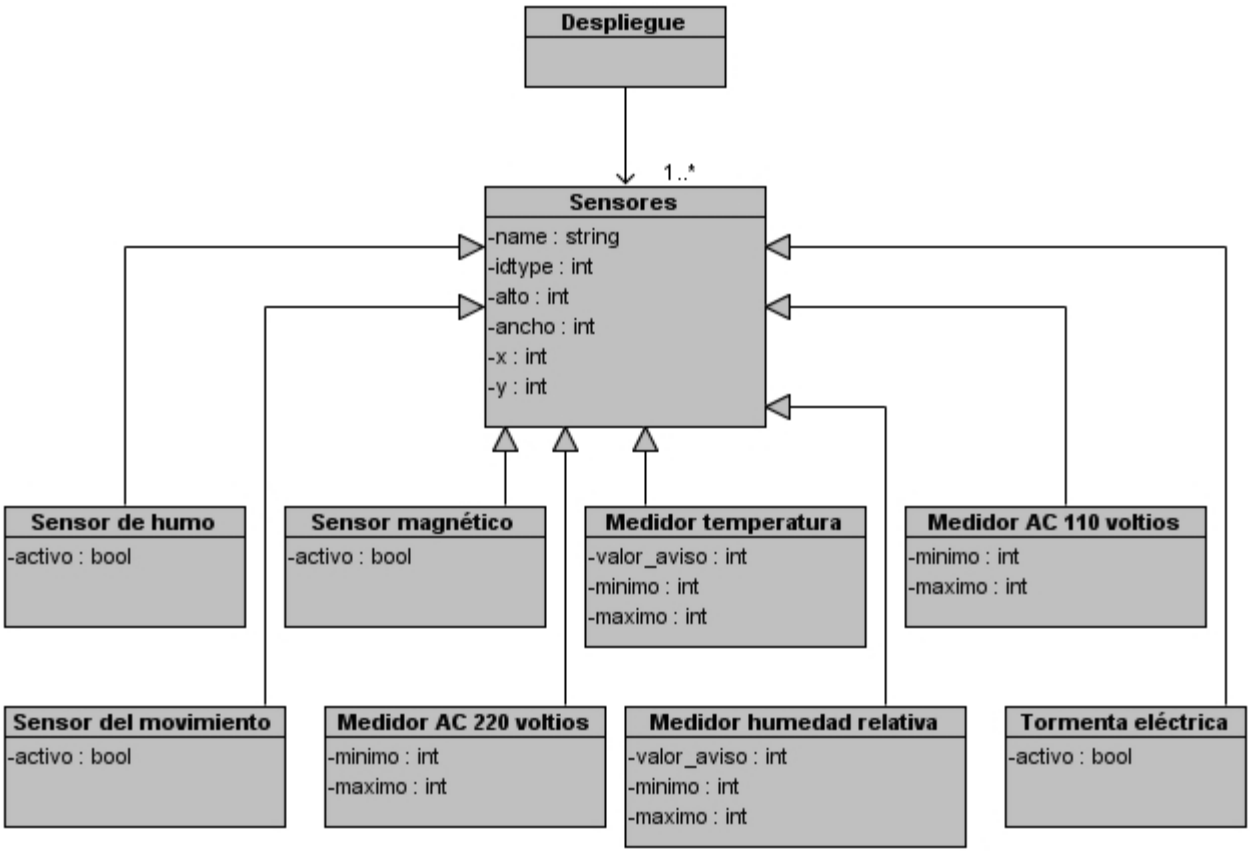

Figura 6: Modelo conceptual

2.7.1 Descripciones de las entidades del Modelo conceptual

A continuación se hace una descripción de las entidades del modelo conceptual presentado anteriormente, haciendo uso de las plantillas definidas en la metodología del Proceso de desarrollo y gestión de proyectos de la UCID.

Sensores

Nombre de la entidad	Sensores.					
Descripción de la entidad	Identifica un dispositivo de detección instalado en el lugar del despliegue.					
Nombre del atributo	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clase válidas	**Clases no válidas**
Idtype	Identifica al sensor.	Int.	No.	Si.	Valores enteros.	----
Name	Contiene el nombre del sensor.	String.	No.	Si.	Cadena de caracteres.	----
Alto	Contiene la altura de la imagen que representa al sensor en el configurador.	Int.	No.	No.	Valores entre 60 y 200.	Valores fuera de rango.
Ancho	Contiene el ancho de la imagen que representa al sensor en el configurador.	Int.	No.	No.	Valores entre 60 y 200.	Valores fuera de rango.
X	Posición horizontal en el configurador.	Int.	No.	No.	Valor mayor que 206.	Valores menores que 206.

Y	Posición vertical en el configurador.	Int.	No.	No.	Valor mayor que 75.	Valores menores que 75.

Tabla 3: Sensores.

Sensor de humo

Nombre de la entidad	Sensor de humo.					
Descripción de la entidad	Dispositivo de detección de humo.					
Nombre del atributo	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clase válidas	**Clases no válidas**
Activo	Informa si el dispositivo detecta o no humo.	Bool.	No.	Si.	----	----

Tabla 4: Sensor de humo.

Sensor magnético

Nombre de la entidad	Sensor magnético.
Descripción de la entidad	Dispositivo que indica si una puerta de un local está abierta o no.

Nombre del atributo	Descripción	Tipo	¿Puede ser nulo?	¿Es único?	Restricciones	
					Clase válidas	Clases no válidas
Activo	Informa si se abrió o no la puerta del local.	Bool.	No.	Si.	----	----

Tabla 5: Sensor magnético.

Sensor de movimiento

Nombre de la entidad	Sensor movimiento.					
Descripción de la entidad	Este dispositivo detecta si en un local existe movimiento lo que puede ser considerado como un acceso de personal no autorizado.					
Nombre del atributo	Descripción	Tipo	¿Puede ser nulo?	¿Es único?	Restricciones	
					Clase válidas	Clases no válidas
Activo	Indica si existe movimiento o no según su valor.	Bool.	No.	Si.	----	----

Tabla 6: Sensor de movimiento.

Medidor Temperatura

Nombre de la entidad	Medidor temperatura.
Descripción de	Mide la temperatura existente.

la entidad						
Nombre del atributo	**Descripción**	**Tipo**	**¿Puede ser nulo?**	**¿Es único?**	**Restricciones**	
					Clase válidas	**Clases no válidas**
valor_aviso	Valor que se establece para indicar que la temperatura se acerca al valor máximo o mínimo establecido.	Int.	No.	No.	Valores enteros.	Cadenas de caracteres.
Mínimo	Valor mínimo establecido en la configuración.	Int.	No.	No.	Valores enteros.	Cadenas de caracteres.
Máximo	Valor máximo establecido en la configuración.	Int.	No.	No.	Valores enteros.	Cadenas de caracteres.

Tabla 7: Medidor de temperatura.

Medidor Humedad Relativa

Nombre de la entidad	Medidor humedad relativa.
Descripción de la entidad	Mide la humedad relativa en el local.

Nombre del atributo	Descripción	Tipo	¿Puede ser nulo?	¿Es único?	Restricciones	
					Clase válidas	Clases no válidas
valor_aviso	Valor que se establece para indicar que la humedad se acerca al valor máximo o mínimo establecido.	Int.	No.	No.	Valores enteros.	Cadenas de caracteres.
Mínimo	Valor mínimo establecido en la configuración.	Int.	No.	No.	Valores enteros.	Cadenas de caracteres.
Máximo	Valor máximo establecido.	Int.	No.	No.	Valores enteros.	Cadenas de caracteres.

Tabla 8: Medidor de humedad relativa.

Para consultar las descripciones de los componentes restantes remitirse al Anexo 1.

2.8 Definición de requisitos

El propósito de la definición de requisitos es especificar las condiciones o capacidades que el sistema debe cumplir y las restricciones bajo las cuales debe operar, logrando un entendimiento entre el equipo de desarrollo y el cliente, y especificando las necesidades reales de forma que satisfaga sus expectativas. [4]

2.8.1 Requisitos funcionales.

Los requisitos funcionales constituyen las capacidades o condiciones que el sistema debe tener, o sea, las utilidades expuestas por la solución para los usuarios, los cuales deben redactarse en el lenguaje del cliente.

A continuación se enumeran las funcionalidades:

1. **Crear despliegue:** El sistema debe permitir al usuario crear un nuevo despliegue.
   - 1.1 **Adicionar componentes del despliegue.**
   - 1.2 **Eliminar componentes del despliegue.**
   - 1.3 **Modificar componentes del despliegue.**
   - 1.4 **Salvar despliegue**
2. **Configurar despliegue:** El sistema debe dar la posibilidad al usuario de configurar un despliegue previamente elaborado.
   - 2.1. **Cargar despliegue**
   - 2.2. **Salvar despliegue.**
   - 2.3. **Adicionar componentes al despliegue.** Esta operación se realiza sobre un despliegue ya creado.
   - 2.4. **Eliminar componentes del despliegue.** Esta operación se realiza sobre un despliegue ya creado.
   - 2.5. **Modificar componentes del despliegue.** Esta operación se realiza sobre un despliegue ya creado.

2.8.1.1 Descripción de los requisitos funcionales.

Para visualizar las descripciones de los requisitos funcionales debe remitirse al Anexo 2.

2.8.2 Requisitos no funcionales

Los requerimientos no funcionales especifican las propiedades o cualidades que debe tener la solución a desarrollar. Representan las características que hacen al producto atractivo, usable, rápido o confiable. Estos requisitos pueden marcar la diferencia entre un producto bien aceptado y otro con poca aceptación. A continuación se listan estas características:

Requerimientos de usabilidad

- Podrá ser usado por personas con conocimientos básicos en el manejo de computadoras.
- Tendrá siempre visible la opción de Ayuda, lo que posibilitará un mejor aprovechamiento por parte de los usuarios de sus funcionalidades.

Requerimientos de confiabilidad

- La información manejada estará protegida de acceso no autorizado y de divulgación.
- Deben montarse sistemas de respaldo eléctrico para mantener la vitalidad de los servicios.

Requerimientos de rendimiento

- Los tiempos de respuestas deben ser generalmente rápidos ante cada una de las acciones que solicite el usuario, al igual que la velocidad de procesamiento de la información, para ello debe garantizarse una velocidad de conexión rápida.
- Se garantizará que todos los procesos de configuración sean eficientes.

Requerimientos de apariencia o interfaz externa

- La interfaz debe ser de fácil comprensión en su funcionamiento permitiendo la utilización del sistema sin mucho entrenamiento, es decir de fácil uso y con rápida respuesta del sistema.
- Interfaces uniformes y con los mismos colores y diseños.
- Se debe garantizar que los colores de la interfaz de la aplicación sean claros.
- Imágenes claras y con la correcta visualización de su contenido.
- Mensajes sin ambigüedades.

Requerimientos de licencias y patentes

- Se deben utilizar herramientas libres.

Requerimientos de portabilidad

- El sistema debe funcionar en sistemas de la familia GNU/Linux y Windows.

Requerimientos políticos- culturales

- Debe respetar los términos empleados normalmente, por los especialistas en el tema de la esfera que se automatiza.

Requerimientos legales

- Se garantizará la protección de los datos, privacidad y derecho a la información.

Requerimientos de seguridad

- Los usuarios deberán autenticarse para entrar al sistema.
- Cuando se intente realizar cualquier acción irreversible, existirá una opción de advertencia antes realizar dicha acción.

Requerimientos de software

- Sistema Operativo: Windows y Linux.

Requerimientos de hardware

- Procesador: 1.00 GHz
- Memoria RAM: 512 MB
- Disco Duro: 5 MB

2.9 Conclusiones

En el presente capítulo se realizó una descripción detallada de las características del sistema y se definieron los requerimientos que guiarán el desarrollo del módulo de configuración. Además se identificaron los conceptos alrededor del mismo lo que nos permite avanzar al siguiente capítulo.

Capítulo 3: Diseño de la arquitectura de software.

3.1 Introducción

En este capítulo se presentará el diseño de la arquitectura de software, así como los diagramas de clases de diseño, los diagramas de secuencia y los patrones a utilizar.

3.2 Arquitectura de software

La arquitectura de software es el conjunto de decisiones significativas sobre la organización del sistema a automatizar, la selección de los elementos estructurales y sus interfaces, de los cuales el sistema está compuesto junto con su comportamiento. Describe los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente.

3.2.1 Modelo Vista Controlador (MVC)

Es un estilo basado en un patrón de diseño que plantea la separación de diferentes clases en dependencia de la función que realizan de modo tal que sea posible manejar dinámicamente la forma en que se procesan solicitudes y se gestiona la manera en que se muestran resultados al usuario final. En otras palabras separa la presentación del dominio de la aplicación. A simple vista ya se tienen ventajas.

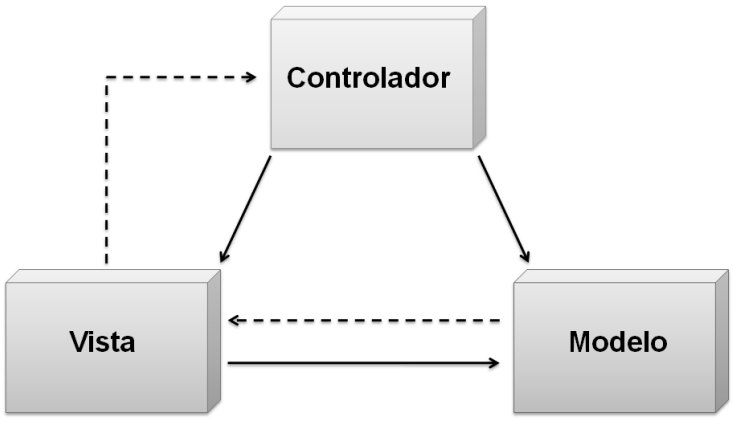

Figura 7: Modelo Vista Controlador.

Modelo

Es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el Modelo y sus Vistas, y notificar a las Vistas cuando cambia el Modelo.

Vista

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

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. [16]

3.3 Patrones de diseño

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

Contribuyen a reutilizar diseño, identificando aspectos claves de la estructura de un diseño que puede ser aplicado en una gran cantidad de situaciones. La importancia de la reutilización del diseño no es despreciable, ya que provee numerosas ventajas: reduce los esfuerzos de desarrollo y mantenimiento, mejora la seguridad, eficiencia y consistencia de los diseños, y proporciona un considerable ahorro en la inversión.

Mejoran la flexibilidad, modularidad y extensibilidad, factores internos e íntimamente relacionados con la calidad percibida por el usuario.

3.3.1 Patrón Decorator

El patrón Decorator dentro de la clase *Zend_View* se puede utilizar para:

- Adicionar responsabilidades a objetos individuales dinámicamente sin afectar otros objetos.
- Para agregar responsabilidades que pueden ser retiradas
- Cuando no es práctico adicionar responsabilidades por medio de la herencia.

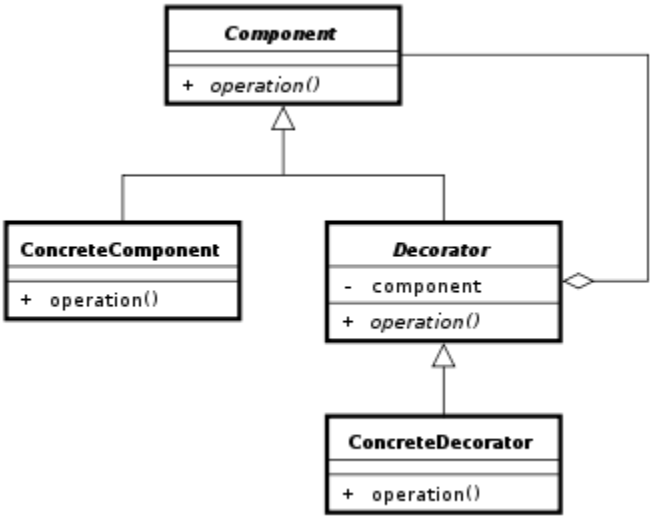

Figura 8: Patrón Decorator

Ventajas

- Es más flexible que la herencia estática.
- Permite que la adición de nuevas responsabilidades (nuevas clases de Decorators) sean independiente de las clases los Objetos que ellas extienden.

Desventajas

- Un Decorator y su Component no son idénticos. Desde el punto de vista de la identidad de los objetos, un DecoratorComponent no es idéntico al Component.
- El patrón Decorator hace que hayan muchos objetos pequeños que son muy parecidos. [17]

3.3.2 Patrón Front Controller.

El patrón Front Controller implica que todas las solicitudes son dirigidas a un único script PHP que se encarga de instanciar al controlador frontal y redirigir las llamadas.

3.3.3 Patrón Singleton

El patrón de diseño Singleton (instancia única) garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.

Figura 9: Patrón Singleton

Ventajas

- Solo existe una instancia de la clase Index.
- Las clases que requieran acceder a la instancia de Index lo consiguen mediante el método de recuperación de la instancia Index: getInstance().

3.3.4 Patrones GRASP

3.3.4.1 Patrón Controlador

Problema: ¿Quién debería encargarse de atender un evento del sistema?
Solución: Asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema, a una clase que represente una de las siguientes opciones:

- El sistema global.
- La empresa u organización global. (controlador de fachada).

- Algo en el mundo real que es activo (por ejemplo, el papel de una persona) y que pueda participar en la tarea (controlador de tareas).

Aplicación: En el sistema cada requisito funcional cuenta con la clase PrincipalController, encargada de manejar los eventos y la lógica del negocio del sistema relacionados al mismo.

3.3.4.2 Patrón Experto

Problema: ¿Quién debiera ser el responsable de conocer la información?
Solución: La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada.

Hay que tener en cuenta que esto es aplicable mientras se esté considerando los mismos aspectos del sistema:

- Lógica de negocio
- Persistencia a la base de datos
- Interfaz de usuario

Aplicación: El sistema implementa este patrón en forma de servicios; ejemplo de su aplicación: la clase principalController.php al necesitar leer datos de un XML.

3.4 Prototipos Interfaz de usuario

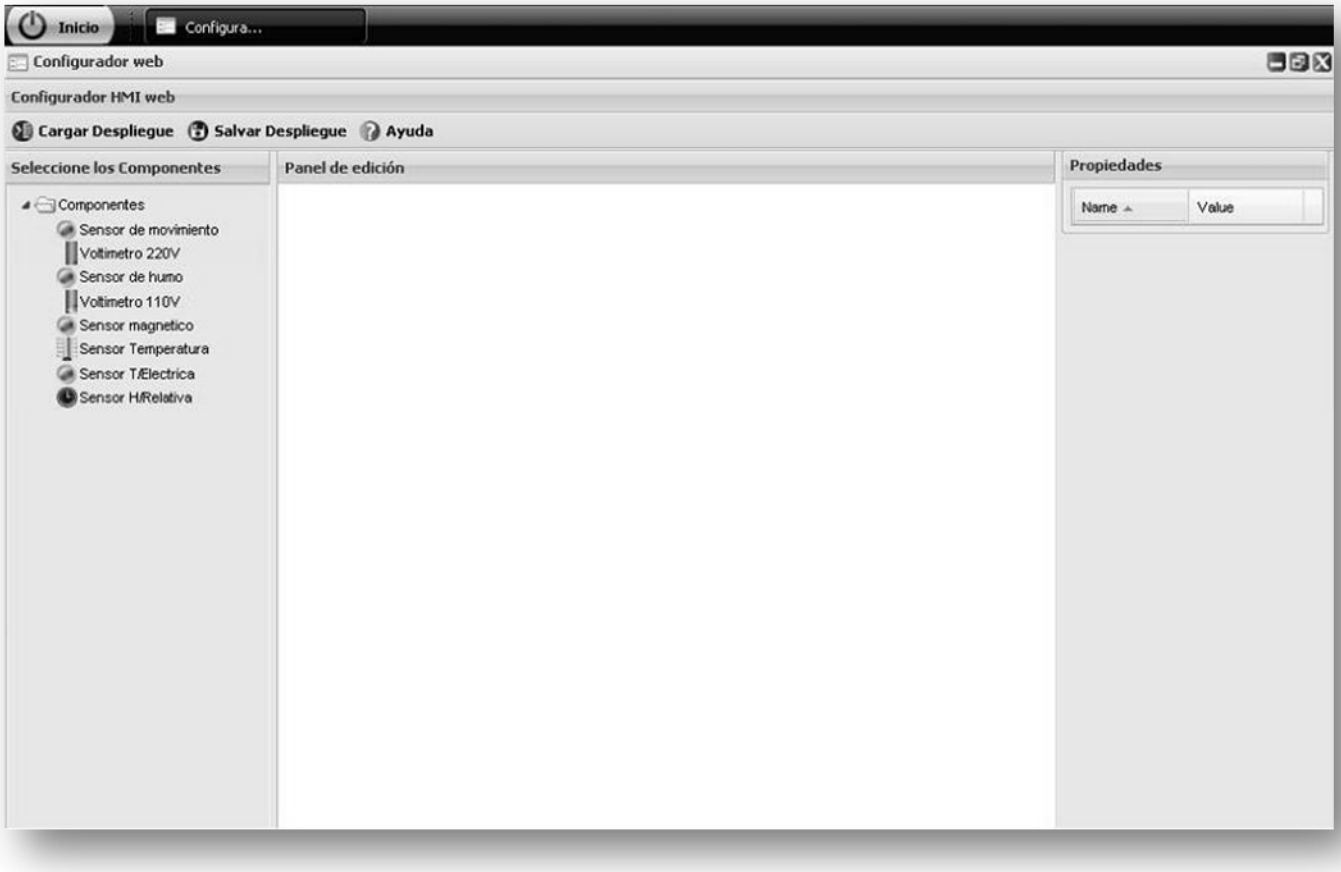

Figura 10: Panel de configuración

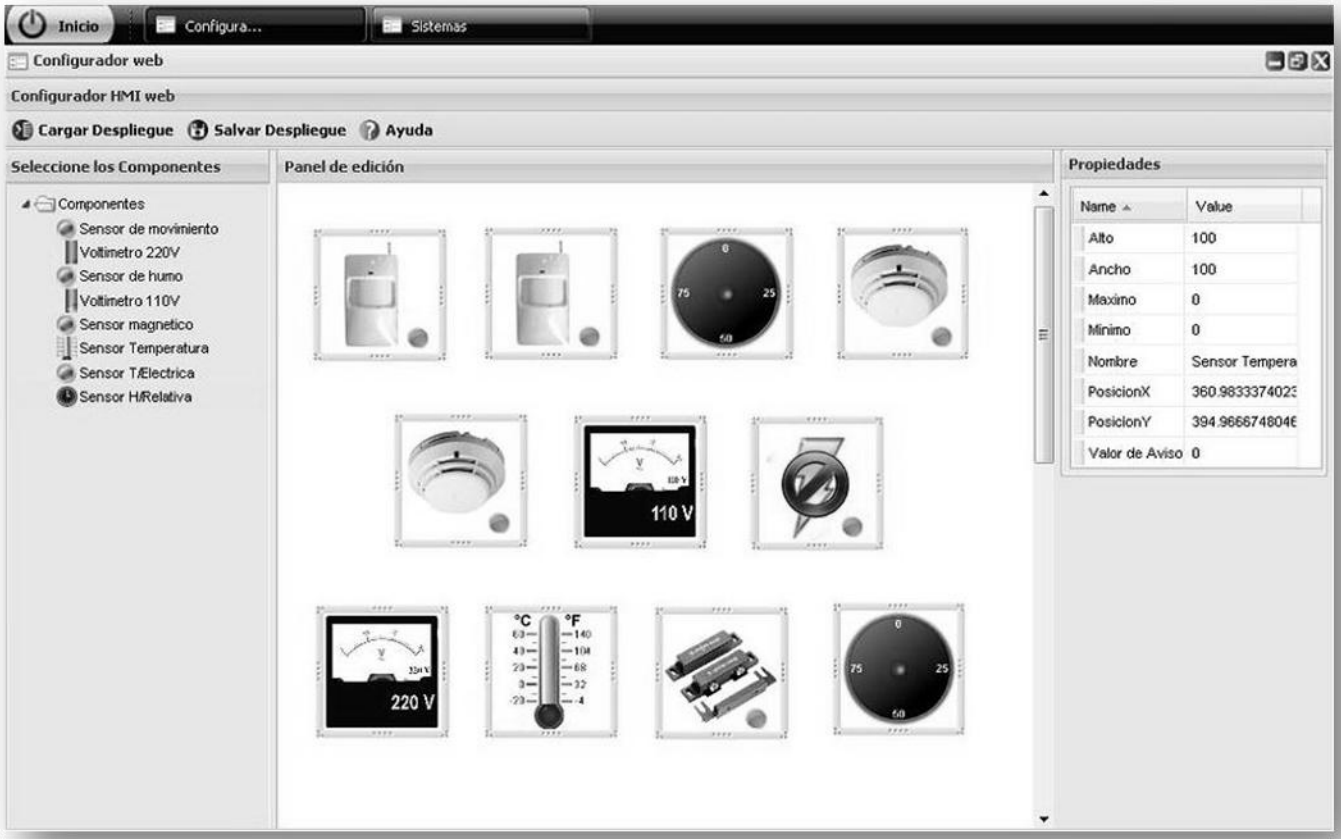

Figura 11: Panel de configuración con componentes añadidos.

3.5 Diagramas de secuencia

Un diagrama de secuencia muestra las interacciones entre objetos ordenadas en secuencia temporal. Muestra los objetos que se encuentran en el escenario y la secuencia de mensajes intercambiados entre los objetos para llevar a cabo la funcionalidad descrita por el escenario. Los diagramas de secuencia, formalmente diagramas de traza de eventos o de interacción de objetos, se utilizan con frecuencia para validar los procesos. Documentan el diseño desde el punto de vista de los procesos.

A continuación se presentarán los diagramas de secuencia por cada escenario de los requisitos funcionales identificados.

RF 1 Crear despliegue.

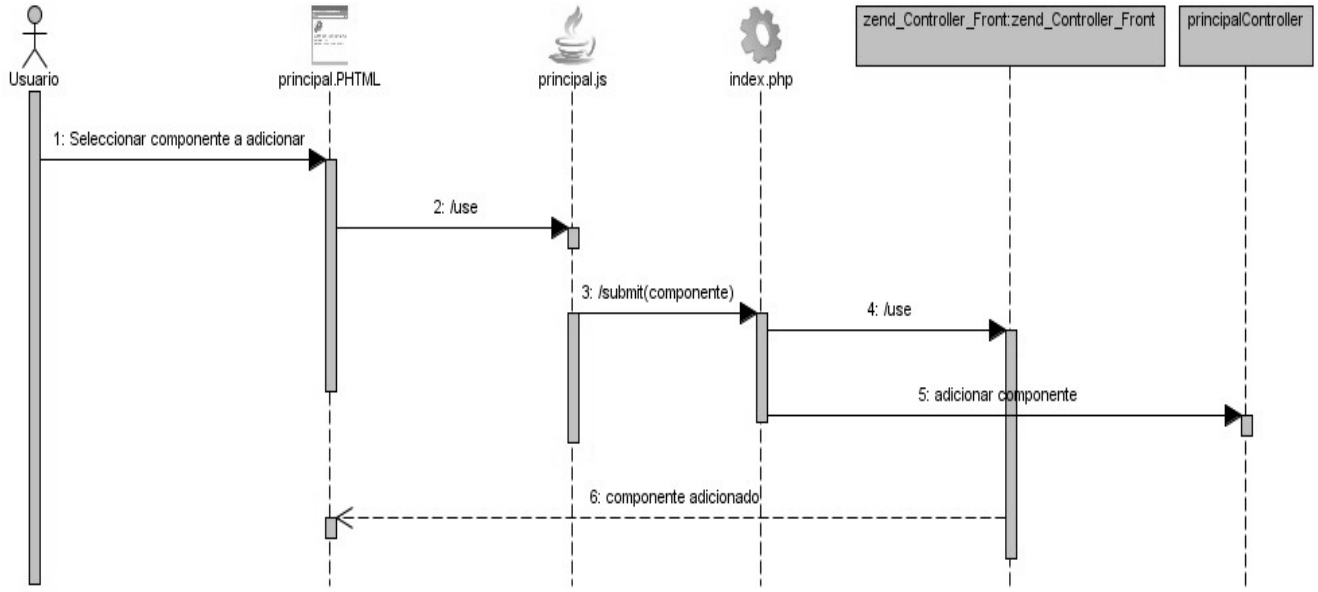

Figura 12: DS del escenario Adicionar componentes del despliegue.

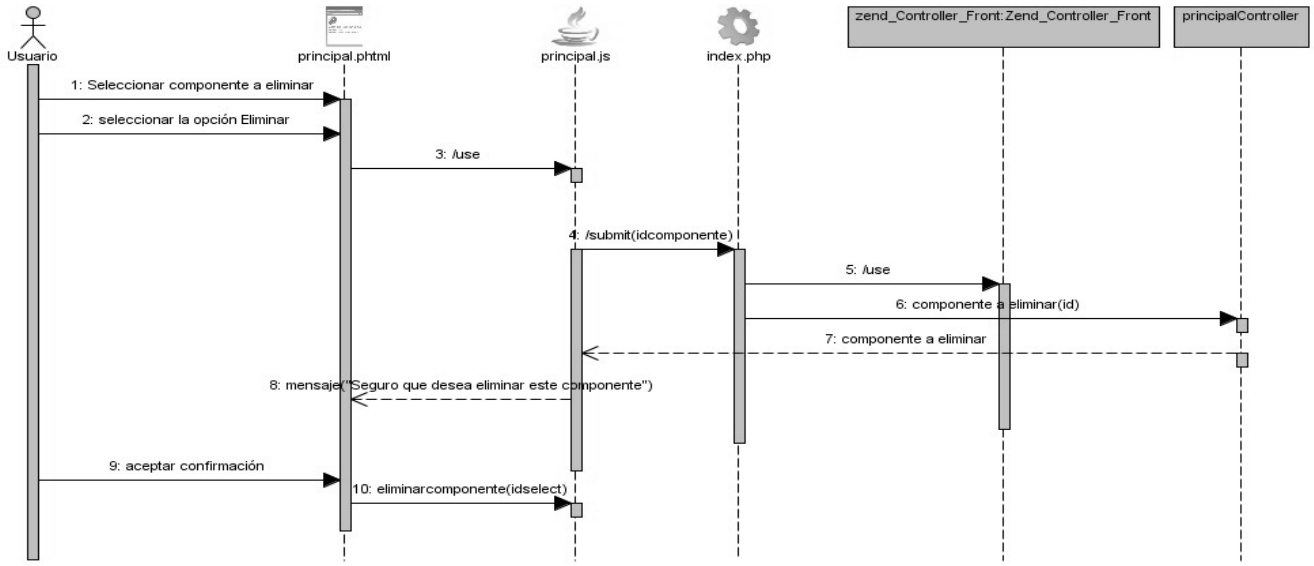

Figura 13: DS del escenario Eliminar componentes del despliegue.

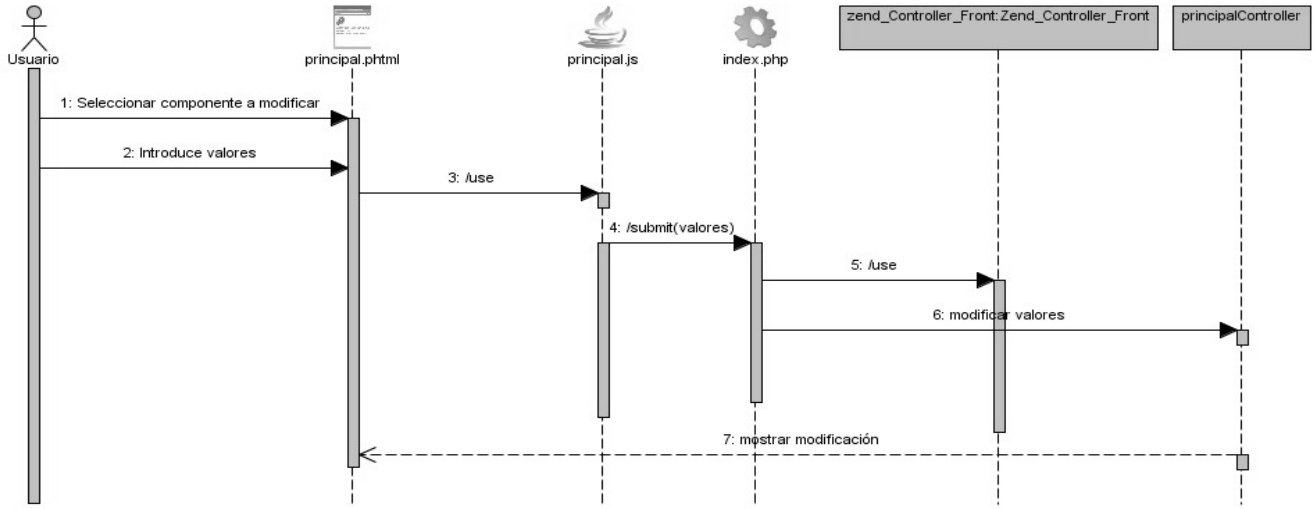

Figura 14: DS del escenario Modificar componentes del despliegue.

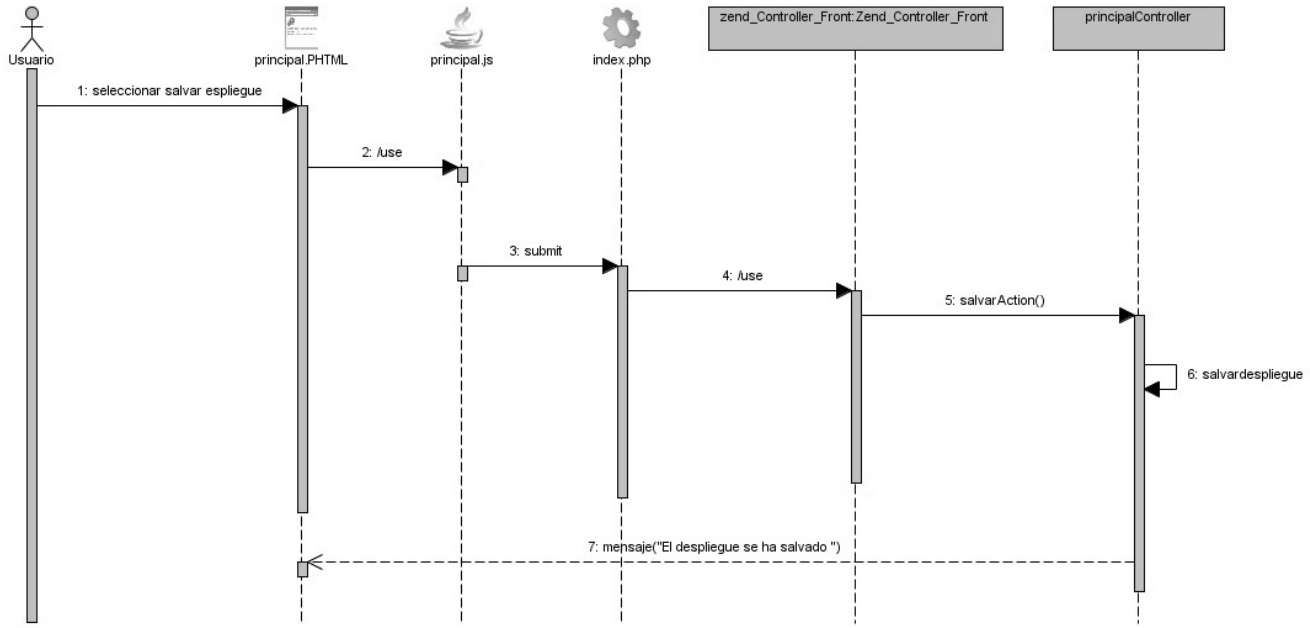

Figura 15: DS del escenario Salvar despliegue.

RF 2 Configurar despliegue

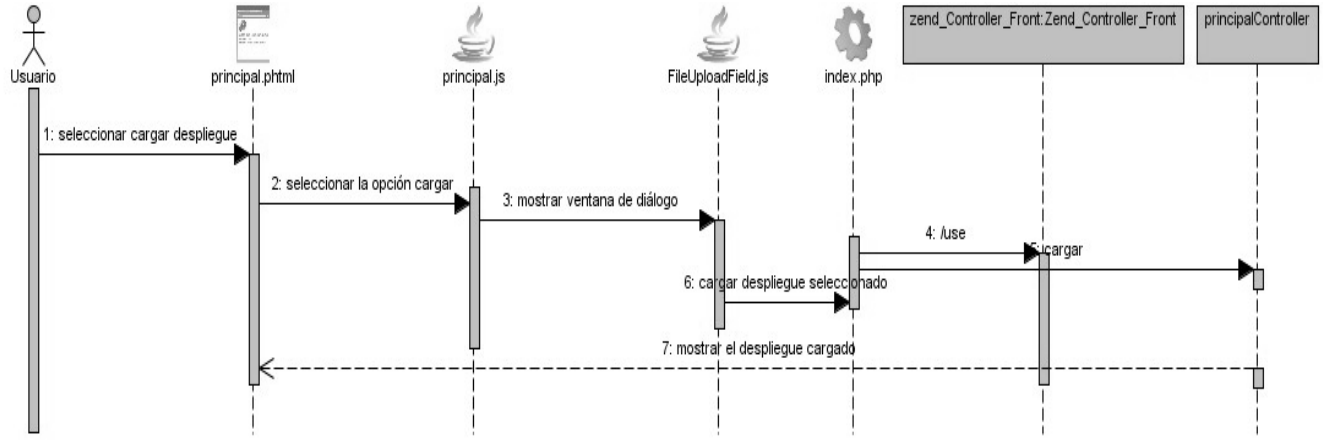

Figura 16: DS del escenario Cargar despliegue.

3.6 Diagrama de clases

Los diagramas de clases son diagramas de estructura estática que muestran las clases del sistema y sus interrelaciones. Los diagramas de clase son el pilar básico del modelado con UML, siendo utilizados tanto para mostrar lo que el sistema puede hacer, como para mostrar cómo puede ser construido.

Los diagramas de clases son los más utilizados en el modelado de sistemas orientados a objetos. Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema. Principalmente, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas. Los diagramas de clases también son la base para un par de diagramas relacionados: los diagramas de componentes y los diagramas de despliegue. Los diagramas de clases son importantes no sólo para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas ejecutables, aplicando ingeniería directa e inversa.
Para consultar los diagramas de clases genéricos remitirse a Anexo 1.

3.6.1 Diagrama de clases del diseño con estereotipos web.

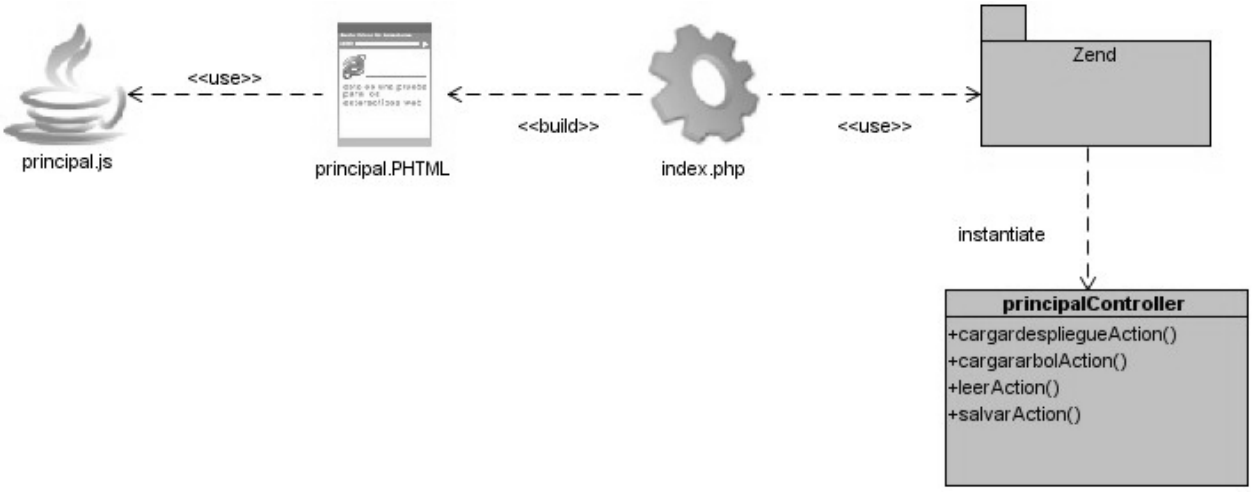

Figura 17: Diagrama de clases del diseño.

3.6.2 Descripción del diagrama de clases

Conocido el diagrama de clases del sistema que se desarrolla, se hace necesario especificar las características de cada una de las clases que lo componen. Las clases incluidas en el módulo son:

- Principal
- PrincipalController

Para consultar las descripciones consultar Anexo 3.

3.7 Conclusiones

En este capítulo se alcanzó la definición de la arquitectura del sistema especificando las clases correspondientes al patrón MVC. Se realizaron los diagramas de clases del diseño con estereotipos web aplicando los patrones de diseño descritos así como los diagramas de secuencia.

Capítulo 4: Implementación y pruebas.

4.1 Introducción

En el presente capítulo se traducen los elementos del diseño en elementos de implementación. Se describe cómo los elementos del modelo del diseño se implementan en términos de componentes, se presentan algunos de los diseños de casos de pruebas que auxiliarán las pruebas de caja negra.

4.2 Diagrama de componentes

Un diagrama de componentes es un diagrama de tipo UML, que representa cómo un sistema de software es dividido en componentes y muestra sus dependencias. En él se situarán librerías, tablas, archivos, ejecutables y documentos que formen parte del sistema.

A continuación se muestra el diagrama de componentes del sistema:

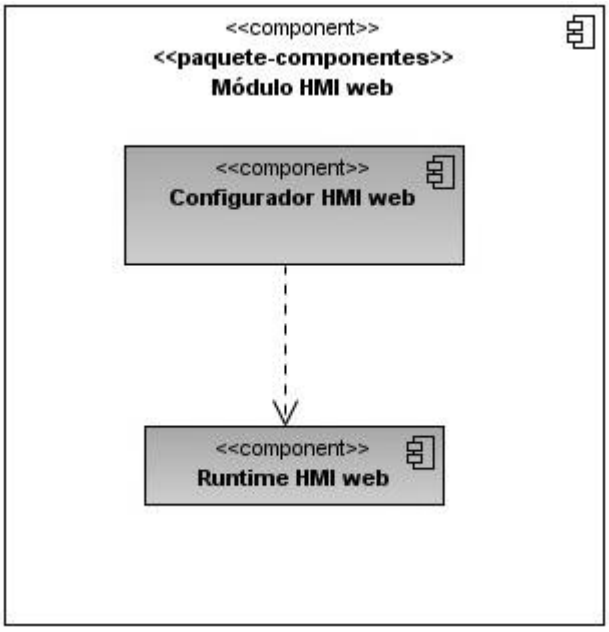

Figura 18: Diagrama de componentes.

4.3 Matriz de integración

La matriz de integración contiene todos los componentes definidos en el subsistema, de forma matricial, y en las intercepciones se especifican los servicios que consume el componente en la vertical del horizontal.

4.3.1 Matriz de integración de componentes externa

	Componentes Externos
Componentes Internos	**Runtime HMI web**
Configurador HMI web	Configuración de un despliegue en un XML.

Tabla 9: Matriz de componentes externa.

4.4 Ejecución de las pruebas de software

Las pruebas del software son la actividad más común de control de la calidad realizada en los proyectos de desarrollo, aunque para lograr un aseguramiento de la calidad del software más eficaz, se deben incluir otras técnicas como por ejemplo revisiones de modelos y documentos, que comúnmente no se realizan desde las primeras fases de desarrollo.

Las pruebas constituyen una etapa imprescindible durante el proceso de desarrollo del software, pues permiten detectar y corregir el máximo de errores posibles antes de la entrega al cliente del software desarrollado, por lo que el éxito de las mismas puede mejorar la percepción de calidad del usuario final y lograr su satisfacción. [4]

4.4.1 Descripción de los métodos empleados para la realización de las pruebas

Pruebas de caja Negra.

La prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. O sea, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene. [15]

Objetivos.

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

Descripción.

Se llevan a cabo sobre la interfaz del software, y es completamente indiferente el comportamiento interno y la estructura del programa.

Los casos de prueba de la caja negra pretenden demostrar que:

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

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

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

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

4.4.2 Diseño de casos de prueba.

4.4.2.1 Diseño de casos de prueba para el escenario Adicionar componentes

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Adicionar componentes	El objetivo de este requisito es adicionar un componente al despliegue.	EP 1.1: Adicionar un componente al despliegue.	- El usuario podrá adicionar un componente al despliegue. - El sistema muestra un componente que simula el equipo a configurar.

Tabla 10: Descripción del caso de prueba del escenario Adicionar componentes.

Para consultar los juegos de datos consultar Anexo 4.

4.4.2.2 Diseño de casos de prueba para el escenario Eliminar componentes

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario

1: Eliminar componentes	El objetivo de este requisito es eliminar un componente al despliegue.	EP 1.1: Eliminar un componente al despliegue.	– El usuario podrá eliminar un componente al despliegue. – El sistema muestra la confirmación del componente que desea eliminar del despliegue, que puede ser aceptada o no.

Tabla 11: Descripción del caso de prueba del escenario Eliminar componentes.

Para consultar los juegos de datos consultar Anexo 4.

4.4.2.3 Diseño de casos de prueba para el escenario Modificar componentes

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Modificar componentes	El objetivo de este requisito es modificar un componente al despliegue.	EP 1.1: Modificar un componente al despliegue.	– El usuario podrá modificar las propiedades de un componente del despliegue. – El sistema muestra los cambios que se le realizó al componente.

Tabla 12: Descripción del caso de prueba del escenario Modificar componentes.

Juegos de datos (Medidor de Temperatura)

Id del escenario	Escenario	Alto	Ancho	Máximo	Mínimo	Valor de aviso	Posición X	Posición Y	Nombre	Respuesta del sistema	Resultado de la prueba
EP 1.1	Modificar componente correctamente	V (65).	V (100).	V (40).	V (25).	V (38).	V (210).	V (78).	V (Medidor Temperatura).	Componente modificado.	El sistema modifica correctamente.
	Modificar componente incorrectamente	I (30).	V (100).	V (40).	V (25).	V (38).	V (210).	V (78).	V (Medidor Temperatura).	Muestra un mensaje indicando que los valores deben estar entre 60 y 200.	Muestra el cartel correctamente.
		V (65).	I (250).	V (40).	V (25).	V (38).	V (210).	V (78).	V (Medidor Temperatura).	Muestra un mensaje indicando que los valores deben estar entre 60 y 200.	Muestra el cartel correctamente.
		V (65).	V (80).	I (11a).	V (25).	V (38).	V (210).	V (78).	V (Medidor Temperatura).	El sistema no permite la entrada de letras en ese	El sistema recupera su valor inicial.

										campo.	
		V (80).	V (80).	V (40).	I (gfd).	V (38).	V (210).	V (78).	V (Medidor Temperatura).	El sistema no permite la entrada de letras en ese campo.	El sistema recupera su valor inicial.
		V (65).	V (80).	V (40).	V (25).	I (12d3).	V (210).	V (78).	V (Medidor Temperatura).	El sistema no permite la entrada de letras en ese campo.	El sistema recupera su valor inicial.
		V (65).	V (80).	V (40).	V (25).	V (38).	I (200).	V (78).	V (Medidor Temperatura).	No permite valores fuera del rango.	El sistema recupera su valor inicial.
		V (65).	V (80).	V (40).	V (25).	V (38).	V (210).	I (30).	V (Medidor Temperatura).	Valor incorrecto.	El sistema recupera su valor inicial.
		V (65).	V (80).	V (40).	V (25).	V (38).	V (210).	V (78).	I (Vacío).	El sistema no permite campos en blanco.	El sistema recupera su valor inicial.

Tabla 13: Juego de datos para modificar el Medidor de Temperatura.

Para consultar los juegos de datos pertenecientes a los sensores restantes consultar Anexo 4.

4.4.2.4 Diseño de casos de prueba para el escenario Salvar despliegue

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Salvar despliegue	El objetivo de este requisito es permitir que el usuario pueda salvar el despliegue realizado.	EP 1.1: Salvar el despliegue.	– El usuario podrá salvar el despliegue realizado. – El sistema guarda el despliegue.

Tabla 14: Descripción del caso de prueba del escenario Salvar despliegue.

Para consultar los juegos de datos consultar Anexo 4.

4.4.2.5 Diseño de casos de prueba para el escenario Cargar despliegue

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario

1: Cargar despliegue	El objetivo de este requisito es permitir que el usuario pueda cargar un despliegue previamente realizado.	EP 1.1: Cargar despliegue.	– El usuario podrá cargar un despliegue. – El sistema muestra los componentes del despliegue cargado.

Tabla 15: Descripción del caso de prueba del escenario Cargar despliegue.

4.4.3 No conformidades

No conformidad	Solución
Errores de taxonomía. Se detectaron palabras secundarias con mayúsculas.	Se corrigieron en los casos necesarios.
En el panel de propiedades, el campo **Nombre** admite dejarlo en blanco.	Se validó correctamente para evitar que esto se mantuviera.

Tabla 16: Descripción de las no conformidades

4.5 Conclusiones

Este epígrafe brindó soporte a la implementación del sotware, en el mismo se relizaron las pruebas de correspondientes a cada uno de los requerimentos funcionales del sistema y se archivan los resultados obtenidos para poder demostrar la eficiencia del producto.

Conclusiones

1. Se realizó un estudio satisfactorio de las Interfaces Hombre – Máquina existentes facilitando así la mejor comprensión de estos sistemas para la lograr una solución más óptima.
2. Se obtuvo la modelación del proceso de configuración para el sistema de supervisión y control de equipamientos a distancia.
3. Se realizó el análisis, diseño e implementación de un sistema de supervisión y control de equipamientos a distancia.
4. Dando cumplimiento al objetivo general se desarrolló finalmente el módulo de configuración del HMI web para el sistema de supervisión y control de equipamientos a distancia.

Recomendaciones

Se recomienda:

- Adicionar nuevas funcionalidades al sistema para lograr una versión más completa para el módulo de configuración.
- Presentar la presente investigación en eventos científicos relacionados con el tema.

Bibliografía

Acevedo Sánchez, José. 2006. Instrumentación y Control Básico de Procesos., Díaz de Santos, 2006.

Acevedo Sánchez, José. 2002. *Control Avanzado de Procesos.,* Díaz de Santos, 2002.

Sistemas SCADA. 2005. [2009]. [Disponible en: http://www.automatas.org/redes/scadas.htm]

Proceso de Desarrollo y Gestión de Proyectos de Software (1ra Versión)". (2009)Documento de trabajo no. 1.Unidad de Compatibilización Integración y Desarrollo De Software para la Defensa.

Visual Paradigm International. New Releases and Feature Enhancements of Visual Paragim Products. Visual Paradigm. [Citado el: 18 de marzo de 2010.] [Disponible en: http://www.visual-paradigm.com/product/news.jsp#VPUML50].

The Definitive JavaScript Resource: JavaScript Tutorials, Free Java Scripts, Source Code and Other Scripting Resources.[Citado el: 3 de febrero de 2010.] [Disponible en: http://www.javascript.com]

PostgreSQL:Documentation:Manuals:PostgresSQL8.3 [Disponible en: http://www.postgresql.org/docs/8.3/interactive/intro-whatis.html]

Booch, Grady, Jacobson, Ivar and Rumbaugh, James. 2007. *El Lenguaje Unificado de Modelado.* ADDISON-WESLEY, 2007. 978-84-7829-087-1.

Álvarez, Miguel Angel. Zend Studio.[Disponible en:http://www.desarrolloweb.com]

Galá, María de las Nieves. Unión de la Industria Militar: Modesta, pero eficiente:*Trabajadores*. 2010 [Disponible en:http://www.trabajadores.cu/news/union-de-la-industria-militar-modesta-pero-eficiente]

Autómatas Industriales. *SCADA.* [Citado el: 3 de febrero de 2010.] [Disponible en:http://www.automatas.org/redes/scadas.htm].

Mañas, J. A. (1994). Pruebas. [Disponible en: http://www.lab.dit.upm.es: http://www.lab.dit.upm.es/~lprg/material/apuntes/pruebas/testing.htm]

Universidad Gran Canaria. Tutorial Java. Arquitectura MVC.[Citado el 17 de mayo de 2010].[Disponible en: http://www.ulpgc.es/otros/tutoriales/java/Apendice/arq_mvc.html]

Garcia, Walter. Patrones de diseño.[Citado el: 17 de mayo 2010][Disponible en: http://agamenon.uniandes.edu.co/~pfiguero/soo/PatronesDiseno/Decorator/Decorator.htm]

Referencias bibliográficas

[1]. Acevedo Sánchez, José. 2006. Instrumentación y Control Básico de Procesos., Díaz de Santos, 2006.

[2]. Acevedo Sánchez, José. 2002. *Control Avanzado de Procesos.,* Díaz de Santos, 2002.

[3]. Sistemas SCADA. 2005. [2009]. [Disponible en: http://www.automatas.org/redes/scadas.htm]

[4].Proceso de Desarrollo y Gestión de Proyectos de Software (1ra Versión)". (2009)Documento de trabajo no. 1.Unidad de Compatibilización Integración y Desarrollo De Software para la Defensa.

[5].Visual Paradigm International. New Releases and Feature Enhancements of Visual Paragim Products. Visual Paradigm. [Citado el: 18 de marzo de 2010.] [Disponible en: http://www.visual-paradigm.com/product/news.jsp#VPUML50].

[6].The Definitive JavaScript Resource: JavaScript Tutorials, Free Java Scripts, Source Code and Other Scripting Resources.[Citado el: 3 de febrero de 2010.] [Disponible en: http://www.javascript.com]

[7].PostgreSQL:Documentation:Manuals:PostgresSQL8.3 [Disponible en: http://www.postgresql.org/docs/8.3/interactive/intro-whatis.html]

[8].Booch, Grady, Jacobson, Ivar and Rumbaugh, James. 2007. *El Lenguaje Unificado de Modelado.* ADDISON-WESLEY, 2007. 978-84-7829-087-1.

[9].[Disponible en: http://d-i.alioth.debian.org/manual/es.s390/ch01s03.html]

[10].[Disponible en:http://www.usdata.com/]

[11].[Disponible en:http://www.advantech.com/]

[12].Alvarez, Miguel Angel. Zend Studio.[Disponible en:http://www.desarrolloweb.com]

[13].Galá, María de las Nieves. Unión de la Industria Militar: Modesta, pero eficiente:*Trabajadores*. 2010 [Disponible en:http://www.trabajadores.cu/news/union-de-la-industria-militar-modesta-pero-eficiente]

[14].Autómatas Industriales. *SCADA.* [Citado el: 3 de febrero de 2010.] [Disponible en:http://www.automatas.org/redes/scadas.htm].

[15].Mañas, J. A. (1994). Pruebas. [Disponible en: http://www.lab.dit.upm.es: http://www.lab.dit.upm.es/~lprg/material/apuntes/pruebas/testing.htm]

[16].Universidad Gran Canaria. Tutorial Java. Arquitectura MVC.[Citado el 17 de mayo de 2010].[Disponible en: http://www.ulpgc.es/otros/tutoriales/java/Apendice/arq_mvc.html]

[17]. Garcia, Walter. Patrones de diseño.[Citado el: 17 de mayo 2010][Disponible en: http://agamenon.uniandes.edu.co/~pfiguero/soo/PatronesDiseno/Decorator/Decorator.htm]

Glosario de términos

RAM (Random Access Memory): Memoria de acceso aleatorio, es un tipo de memoria temporal que pierde sus datos cuando se queda sin energía (por ejemplo: al apagar la computadora).

Sensores: Un sensor es un dispositivo capaz de transformar de forma automática magnitudes físicas o químicas, llamadas variables de instrumentación, en magnitudes eléctricas. Las variables de instrumentación dependen del tipo de sensor y pueden ser por ejemplo: temperatura, intensidad lumínica, distancia, aceleración, inclinación, desplazamiento, presión, fuerza, torsión, humedad, etc.

Widget (Objeto gráfico): Objeto gráfico que puede contener gráficos vectoriales o imágenes raster, (imágenes de mapas de bits) y presenta un conjunto de propiedades que pueden ser editadas y asociadas a puntos (variables) del SCADA o a expresiones que contengan variables del sistema. Por medio de los objetos gráficos se pueden crear animaciones en función de los valores de los puntos asociados. Los widgets pueden agruparse para formar nuevos objetos gráficos más complejos. Un ejemplo de widget puede ser un contenedor de textos.