UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 5

Centro de Informática Industrial

CEDIN

Título: “Visualizador web de la Interfaz Hombre-Máquina del SCADA Guardián del ALBA.”

Trabajo de Diploma para optar por el título de
Ingeniero Informático

Autor: Adolfo Yasser Santana Rojas.

Tutor(es): Ing. Yosvani Ramírez Martínez

Ing. Ariannys Garrido Saroza.

La Habana, junio de 2015

Año del 56 Aniversario del Triunfo de la Revolución

Agradecimientos.

A mis padres, quienes son mi razón de ser, son una gran inspiración para mí. Gracias por estar siempre ahí para aconsejarme y guiarme por buenos caminos.

A mis abuelas, Aleida y María Caridad por su cariño y constante preocupación, gracias por estar siempre pendientes de su nieto.

A mis tías y primos, a mi familia en general por poder contar con ellos en todo momento, gracias por su confianza en mí, el hecho de saber que me apoyan, me ayudan a seguir adelante.

A mis compañeros del antiguo 5101, por brindarme su apoyo, por compartir buenos y malos momentos, gracias por demostrarme el valor de la amistad.

A las personas que compartieron conmigo estos años, gracias por todos los momentos que pasamos juntos.

A mis tutores Yosvani y Ariannys, por ser unos guías inigualables en el desarrollo de este trabajo.

Al personal del glorioso laboratorio 23 del Centro de Informática Industrial (CEDIN) por sus recomendaciones.

Al tribunal por sus correcciones, porque me fueron de gran ayuda.

A todos, nos vemos cuando el destino tenga ganas de juntarnos, mientras Cuídense y sean muy felices.

Dedicatoria.

A mis padres que han sido un impulso para mí, a ellos les debo todo lo que soy. Mi madre por darme toda su confianza, comprensión, amor y por apoyar mis decisiones en los momentos más difíciles, muchas gracias por existir. Mi padre por ser un ejemplo todo el tiempo, por todos tus consejos y enseñanzas, que me han hecho ser el hombre que soy.

Ustedes son la razón por la que vivo y siento de corazón que tenerlos es poseer el mayor tesoro del mundo.

Declaración jurada de autoría.

Declaro ser el autor de la presente tesis: **"Visualizador web de la Interfaz Hombre-Máquina del SCADA Guardián del ALBA"** y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo el presente a los ____ días del mes de _____ del año 2015.

Firma del autor

Adolfo Yasser Santana Rojas

Firma del tutor

Ing. Yosvani Ramírez Martínez

Firma del tutor

Ing. Ariannys Garrido Saroza

Las Tecnologías de la Información y las Comunicaciones (TIC) son un factor de vital importancia en la transformación de la sociedad. El constante avance de las TIC y el crecimiento de las industrias han traído como consecuencia un aumento en la complejidad de los procesos industriales, los cuales se han convertido en un gran avance para el sector empresarial. Dentro de las aplicaciones informáticas que se utilizan para lograr un control eficiente de estos procesos, se encuentran los sistemas de supervisión, control y adquisición de datos, conocidos como *Supervisory Control and Data Acquisition (SCADA)*. En la actualidad una gran parte de estos sistemas han enfocado su diseño hacia aplicaciones web, pues estas permiten mayor extensibilidad, accesibilidad y estandarización.

La presente investigación surge a partir de la necesidad de los directivos, supervisores y personal autorizado de la empresa PDVSA, de visualizar la información del sistema SCADA Guardián del ALBA (GALBA), en dispositivos portátiles y estaciones de trabajo que no tengan instalado dicho sistema. El objetivo fundamental es desarrollar una aplicación web capaz de visualizar en tiempo real la información del sistema SCADA GALBA. Para el desarrollo de la solución propuesta se realiza un análisis de las principales herramientas, tecnologías y metodologías que se utilizan en la construcción de un software. El proceso estuvo guiado por el uso de las siguientes herramientas y tecnologías: Visual Paradigm como herramienta CASE, UML como lenguaje de modelado, HTML como lenguaje de etiquetado, SVG como tecnología para la creación de gráficos, WebSocket como mecanismo de comunicación, JavaScript como lenguaje de programación script respaldado por los *framework*[1] AngularJS y JQuery, desarrollado con el IDE Brackets. Para validar que los resultados alcanzados fueron los esperados se realizó un conjunto de pruebas: Caja blanca con la técnica Camino básico y Caja negra con la técnica Partición de equivalencia. Se logró corregir todas las no conformidades detectadas lo que significa que la solución propuesta está lista para ser desplegada. Finalmente se obtuvo un Visualizador web capaz de representar en tiempo real los datos provenientes del SCADA GALBA.

Palabras claves: HMI, Supervisar, objetos gráficos, SCADA.

[1] Abstracción de código común que provee funcionalidades genéricas que pueden ser utilizadas para desarrollar aplicaciones de manera rápida, fácil, modular y sencilla.

Abstract.

The Information Technology and Communications (ICT) are a vital factor in the transformation of society. The constant progress of ICT and the growth of industries have resulted in an increase in the complexity of industrial processes which has become a breakthrough for the business sector. Among the applications that are used for efficient control of these processes are monitoring systems, data acquisition and control, known as SCADA for its acronym in English. Today a large part of these systems have focused their design to web applications, as these allow for greater extensibility, accessibility and standardization.

This research arises from the need of managers, supervisors and authorized personnel of PDVSA, to display information system SCADA Guardian ALBA (GALBA), on portable devices and workstations that do not have this type of system installed. The main goal is to develop a web application capable of displaying real-time information GALBA SCADA system. For the development of the proposed solution an analysis of the main tools, technologies and methodologies used in the construction of software is done. The process was guided by the use of the following tools and technologies: Visual Paradigm as CASE tool, UML as modeling language, HTML as markup language, SVG as a technology for creating graphics, WebSocket as a means of communication, JavaScript as a language Script Programming backed by angularjs and JQuery framework, developed with the IDE brackets. To validate that the results obtained were as expected a set of tests were performed: White box with basic technical black box Road and the technical equivalence partition. It was possible to correct all nonconformities which means that the proposed solution is ready to be deployed. Finally a web viewer capable of displaying real-time data from the SCADA GALBA was obtained.

Keywords: HMI, Monitoring, graphic objects, SCADA.

Índice de Tablas.

Índice de Figuras.

En poco tiempo las Tecnologías de la Información y las Comunicaciones (TIC) han evolucionado considerablemente, hasta el punto de volverse casi imprescindible para algunos sectores de nuestra sociedad. La automatización de los procesos industriales se ha convertido en un gran avance para el sector empresarial, con su uso, se han revolucionado las producciones a gran escala, reduciendo el peligro de catástrofes, además de obtener grandes ahorros en tiempo, dinero y recursos.

En las industrias se utilizan diferentes herramientas para la ejecución, monitoreo y control de los procesos industriales. Recientemente estas herramientas han ganado en complejidad y funcionalidad de forma exponencial, lo que ha permitido una disminución considerable en la carga de trabajo de los trabajadores en cuanto a tareas se refiere, así como la peligrosidad en las labores que realizan. En la actualidad, los procesos industriales se monitorean y controlan mediante los sistemas SCADA. Estos sistemas proporcionan gran información de los procesos de forma oportuna para la toma de decisiones y centralizan el funcionamiento de una empresa en una reducida cantidad de estaciones de trabajo, mejorando altamente la eficacia de dichos procesos.

En la Universidad de las Ciencias Informáticas (UCI) se implementa un esquema estructurado de estudio-trabajo, en el cual se instruye a los estudiantes tanto en su formación docente como en su vida laboral, basados en la integración de procesos fundamentales como la formación, investigación y la producción entorno a una temática de un centro de desarrollo. La universidad a su vez cuenta con un Centro de Informática Industrial (CEDIN), perteneciente a la facultad 5, el cual está desarrollando varios software para contribuir a la automatización de procesos industriales, destacando entre los productos el sistema SCADA GALBA. Dicho sistema está compuesto por varios módulos que interactúan entre sí, comenzando el proceso por los controladores de dispositivos y finalizando en la Interfaz Hombre-Máquina (HMI). El HMI está compuesto por un ambiente de edición o editor, que es donde se realizan las configuraciones para la supervisión, y el ambiente de ejecución o visualizador, que permite al operador supervisar y controlar los procesos industriales.

El GALBA está orientado al trabajo en estaciones fijas como una aplicación de escritorio, el cual debe ser instalado sobre el sistema operativo Debian en cada ordenador destinado a la supervisión y el control de los procesos. Para realizar alguna modificación o actualización en el sistema se debe realizar en cada cliente por separado, lo que torna compleja la tarea de soporte a dicho sistema. Además posee una deficiente escalabilidad en la visualización de los despliegues, así como elevados consumos de los recursos de hardware. Las condiciones planteadas anteriormente provocan que los directivos, supervisores y el

personal autorizado de la empresa PDVSA, no puedan visualizar la información de los procesos en tiempo real mediante el Ambiente de Ejecución del HMI en estaciones de trabajo que no tengan instalado el sistema, limitando la disponibilidad de la aplicación. Esto ocasiona que los directivos necesiten trasladarse hacia las oficinas para poder supervisar los procesos, estando éstas en muchas ocasiones alejadas de las entidades administrativas, provocando además gastos innecesarios.

Teniendo en cuenta la situación planteada con anterioridad se define el siguiente **problema de investigación:** ¿Cómo lograr la disponibilidad del visualizador del HMI del SCADA Guardián del ALBA en todas las estaciones de trabajo y dispositivos portátiles?

Para dar solución a este problema se asume como **objeto de estudio:** El proceso de visualización de los Ambientes de Ejecución de los HMI en sistemas SCADA.

Enmarcado en el **campo de acción:** La disponibilidad y visualización del Ambiente de Ejecución del HMI en el SCADA Guardián del ALBA.

En concordancia con lo anterior se propone como **objetivo general:** Desarrollar una aplicación web que permita la disponibilidad del visualizador del HMI del SCADA Guardián del ALBA.

Para dar cumplimiento a los objetivos de esta investigación se definieron las siguientes **tareas investigativas**:

1. Establecimiento de los fundamentos teórico-metodológicos para el desarrollo de sistemas informáticos SCADA en el entorno web.
2. Análisis del estado del arte de las principales tecnologías, metodologías y herramientas para el desarrollo de aplicaciones web en tiempo real.
3. Análisis y diseño de la solución informática.
4. Implementación de la solución informática.
5. Validación de los resultados obtenidos mediante la realización de pruebas.
6. Valoración cualitativa de los resultados obtenidos en la validación de la solución propuesta.

Durante la investigación se llevan a cabo varios **métodos y técnicas** en la búsqueda y procesamiento de la información como son:
Métodos Teóricos.

- **Analítico-sintético:** Para el estudio de los conceptos empleados en los sistemas SCADA web, analizando todos los documentos elaborados, para la extracción de los elementos más importantes.

- **Histórico-lógico:** Para la comprensión de los antecedentes y las tendencias actuales referidas la evolución en el mundo de los sistemas SCADA web.

Métodos Empíricos.

- **Observación:** Se puso en práctica este método para conocer el funcionamiento existente en los despliegues del SCADA guardián del ALBA mediante el comportamiento de los dispositivos de campo en las propiedades de los objetos gráficos y sumarios empleados para la toma de decisiones de los operadores.
- **Consulta bibliográfica:** Empleada para consultar las fuentes de información relacionados con los tipos de los sumarios y objetos gráficos visualizados en los entornos web de los HMI en sistemas SCADAS.

La estructura del documento se resume en los siguientes acápites:

Capítulo 1. Fundamentación teórica:

Abarca las principales características de un sistema SCADA y los módulos que lo componen, haciendo énfasis en el de visualización. Se realiza un estudio de las principales tecnologías que se utilizan para el desarrollo de aplicaciones web. Además se fundamenta la selección de las herramientas, metodología y tecnologías para el desarrollo de la solución propuesta.

Capítulo 2. Análisis y diseño del sistema:

Comprende aspectos relacionados con el análisis y diseño del sistema. Se realiza el levantamiento de los requisitos del sistema a partir del modelo conceptual diseñado, además se identifican los casos de uso que contiene el sistema y se describe cada uno de ellos para guiar todo el proceso de desarrollo del software. Se expone el diseño del sistema a través de una descripción de los estilos y patrones arquitectónicos más utilizados y la aplicación de estos para la confección del modelo de diseño.

Capítulo 3. Implementación y pruebas:

Describe mediante el modelo de componentes la organización y las dependencias lógicas que existen entre los ficheros que conforman la aplicación, dando paso a la realización de una serie de pruebas para verificar el funcionamiento de la implementación realizada.

Capítulo 1: Fundamentación Teórica.

1.1 Introducción.

En el presente capítulo se engloban conceptos fundamentales asociados a los sistemas SCADA. Además se exponen las características principales de estos sistemas en la web. Se analizan las principales tendencias, tecnologías, metodologías y software utilizados en la actualidad para el desarrollo de aplicaciones web. A su vez se analizan y se fundamenta la selección de estas para el desarrollo de la solución propuesta.

1.2 Sistemas de Supervisión, Control y Adquisición de Datos SCADA.

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.

Los sistemas SCADA, comprenden las soluciones de aplicación que requieren de la captura de información de un proceso o planta industrial, la cual es utilizada para realizar una serie de análisis o estudios con los que se pueden obtener valiosos indicadores de los procesos para mejorar la eficacia del monitoreo y control del proceso y de la toma de decisiones operacionales apropiadas. A su vez se comunican con los dispositivos de campo y controlan el proceso de forma casi automática desde la pantalla de un ordenador u otra tecnología de comunicación. Otras de las características de los SCADA es 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. En la siguiente figura se describe la estructura básica de un SCADA (1).

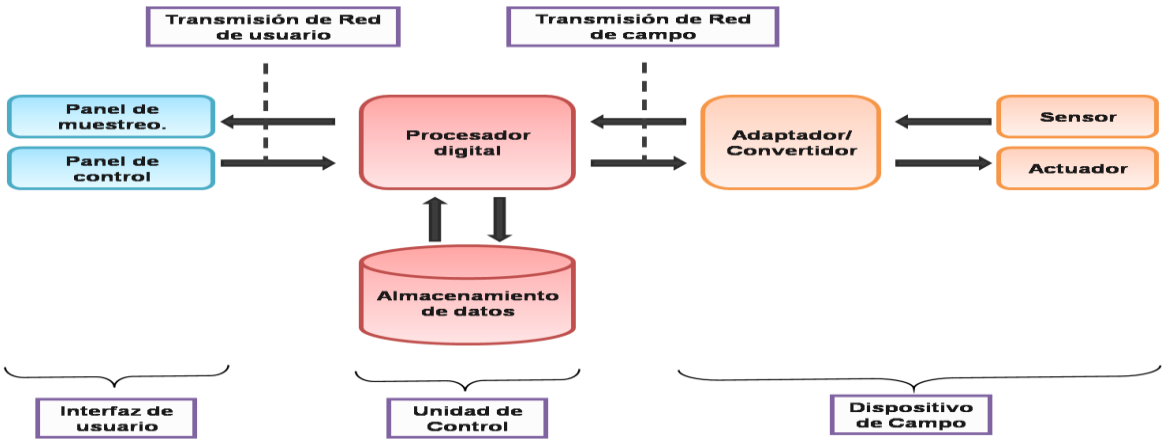


Figura 1: Estructura básica de un SCADA.

1.2.1 Sistema SCADA Guardián del ALBA.

Desde el año 2006 se comienza el desarrollo en la UCI de un SCADA en convenio con la Gerencia AIT de la empresa Petróleos de Venezuela S.A (PDVSA), conocido en sus inicios como SCADA Nacional o SCADA PDVSA y a partir del 2008 en que fue presentado en la "Cumbre del ALBA" se comienza a conocer como SCADA GALBA, donde se proyectó una futura instalación del sistema en los países integrantes de esta organización. El software, que se da como solución es realizado en cooperación de distintos equipos de desarrollo de la UCI, DST-AIT PDVSA, Centro de Desarrollo de Automática Integral (CEDAI), Universidad Central "Marta Abreu" de Las Villas (UCLV), Instituto Superior Minero Metalúrgico de Moa (ISMMM), Universidad de los Andes (ULA), DBAccess, IntelCom y la empresa de Ingeniería de Software y Calidad Aplicada (Isca). El SCADA "Guardián del ALBA" es un software que integra las funcionalidades de alto nivel que permite la solución de aplicaciones de supervisión y control de procesos, utilizando para ello una arquitectura distribuida de módulos que permite escalar a aplicaciones de gran envergadura.

1.3 Módulo Interfaz Hombre-Máquina (HMI).

Los sistemas SCADA cuentan con un HMI que permite representar o controlar los datos de un proceso determinado. 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. Las señales del proceso son conducidas al HMI por medio de dispositivos como tarjetas de entrada/salida en el ordenador, PLC (Controladores lógicos programables), PAC (Controlador de automatización programable), RTU (Unidades remotas de I/O) o DRIVER (Variadores de velocidad de motores). (2)

El HMI del sistema SCADA Guardián de ALBA está compuesto por dos ambientes de trabajo, el de Configuración o Editor que permite administrar la configuración del proyecto y

sus recursos; y el de Ejecución o Visualizador que permite ejercer control y supervisar los recursos que han sido configurados con anterioridad en el ambiente de configuración. A continuación se describen algunas de las funcionalidades que permiten estos ambientes:

- El editor permite crear un proyecto y administrar la configuración de sus recursos (Seguridad, Histórico, Adquisición y HMI). Posibilita la creación y el diseño de los despliegues que se visualizarán en el ambiente de Ejecución utilizando para ello los componentes necesarios para representar la información al operador que son asociados a su vez con los puntos(valores configurados para adquirir información del campo) que representan, dando la posibilidad el sistema de modificar todos los recursos que se definan. Permite la creación y edición de reportes, la definición de los usuarios bajo un esquema de privilegios identificado y la configuración de los grupos donde se almacenarán los datos del sistema.

- El visualizador permite adquirir datos del campo en forma continua, almacenar y mostrar información en forma de sumarios y gráfica, ejercer control y supervisión de los despliegues configurados en el ambiente de configuración, así como visualizar las alarmas que se detecten y afecten el comportamiento del proceso supervisado. Permite la autenticación de usuarios en el sistema. Crea una sesión para cada usuario autenticado, con sus respectivos privilegios sobre la herramienta y maneja los tiempos de expiración de la sesión.

Como se observa, los visualizadores de hoy en día ofrecen a los operadores las más sofisticadas técnicas de supervisión y control, brindado la posibilidad de simular las operaciones que se realizan en una planta industrial con los gráficos sinópticos, haciendo sumamente intuitivo la operación de una planta específica.

1.4 Visualizadores web para los sistemas SCADA.

Con el vertiginoso desarrollo de las aplicaciones web se han abierto las vías en el acceso y uso de la información en cualquier parte geográfica del mundo. Con los avances de esta tecnología cada vez se necesitan aplicaciones más rápidas, ligeras y robustas. A su vez 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; permitiendo así una alta disponibilidad de los recursos solicitados donde se tenga acceso a Internet, fácil integración de datos de diferentes fuentes y su respectiva centralización, racionalización del trabajo permitiendo la reducción del tiempo y dinero destinado al mantenimiento, menos requerimientos de memoria en comparación con programas instalados localmente. (1)

En la actualidad existe una gran variedad de sistemas SCADA que poseen una Interfaz Hombre-Máquina que incluyen una extensión para la supervisión y control de los procesos mediante la web. Entre los sistemas que se pueden encontrar que cumplen estas características, todos tienen solución con tecnologías bajo licencia propietaria, entre ellos se encuentran:

- **HMI 500:** La aplicación HMI 500 corresponde a la más reciente solución de EFACEC para la implementación de la Interfaz Hombre-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. (3)
- **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 un 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 *CSWorks* utiliza OPC. (4)
- **WebHMI:** Es un poderoso complemento para cualquier sistema de Genesis32 (5), 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 las extensiones necesarias 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.

1.5 Metodologías para el desarrollo de software.

Las metodologías de desarrollo de software asignan un proceso disciplinado en el que se ve paso a paso y detalladamente las tareas y actividades que se deben realizar para obtener un producto informático con excelente calidad. Existen numerosas propuestas de metodologías, por una parte están las metodologías tradicionales, que se centran en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas, documentación detallada y notaciones para el modelado donde el resultado final sería un proceso de desarrollo más complejo. Entre estas se pueden encontrar: RUP (*Rational Unified Procces*) y MSF (*Microsoft Solution Framework*). Por otra parte las metodologías ágiles le dan un mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con interacciones muy cortas, centrándose en el factor humano y en el producto de software. Entre estas la más destacadas hasta el momento son: XP (*Extreme Programming*), OpenUp y AUP (*Agile Unified Process*).

En la UCI actualmente existen 14 centros productivos, cada uno de estos centros se dedica al desarrollo de *software* y/o servicios empleando el uso de diferentes metodologías de desarrollo entre robustas y ágiles. A pesar de la variedad de metodologías usadas, se ha comprobado que muy pocos proyectos la aplican en su totalidad. Las diferencias entre estas metodologías no radica únicamente en los productos de trabajos que proponen o en sus roles, sino en su forma de planificar el proyecto y realizar las estimaciones del tiempo. Factor determinante en la culminación exitosa de todo desarrollo de software. Para lograr erradicar los problemas detectados, se decide escoger una metodología para ser adaptada a lo que ya la Universidad ha estado proponiendo como ciclo de vida de los proyectos, sin alejarse de lo que hasta el momento se ha trabajado e introducir la menor cantidad de cambios posibles.

Al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos) exigiéndose así que el proceso sea configurable. Se decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI. Con la adaptación de AUP que se propone para la actividad productiva de la UCI se logra estandarizar el proceso de desarrollo de software, dando cumplimiento además a las buenas prácticas que define CMMI-DEV v1.3. Se logra un lenguaje común en cuanto a fases, disciplinas, roles y productos de trabajos. De ahí que se escoge la utilización

de dicha metodología para guiar el proceso de desarrollo de la solución propuesta, cumpliendo con los estándares de calidad definidos por la universidad. (6)

Esta describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. Se caracteriza por estar dirigido por casos de uso, centrado en la arquitectura y por ser iterativo e incremental. Describe un enfoque simple del desarrollo del software usando técnicas y conceptos ágiles. Algunas técnicas usadas por AUP incluyen el desarrollo orientado a pruebas, modelado y gestión de cambios ágiles y refactorización de base de datos para mejorar la productividad. (7)

1.6 Tecnologías y Herramientas a utilizar.

Las herramientas y tecnologías escogidas para el desarrollo de un proyecto deben ser seleccionadas cautelosamente, ya que pueden suponer el fracaso de éste o pueden aumentar su complejidad, para lo cual se deben conocer cuáles son las distintas alternativas y las necesidades del proyecto.

1.6.1 Lenguaje de modelado.

Son un conjunto estandarizado de símbolos que facilita la modelación de un diseño de software. Se usan en combinación con una metodología de desarrollo de software para avanzar de una especificación inicial a un plan de implementación. Para la especificación y documentación del producto el lenguaje de modelado es esencial, ya que constituye un pilar para el mantenimiento del software, su posterior actualización y soporte.

Se selecciona como lenguaje el UML, debido a que se emplea para visualizar, especificar, construir y documentar los artefactos de la solución propuesta. Es un lenguaje estándar, fácil de aprender, y ofrece un conjunto de notaciones y diagramas estándar para mostrar la solución propuesta desde varias perspectivas. Está especialmente diseñado para apoyar un estilo de desarrollo iterativo e incremental y presenta tecnología orientada a objetos. Ayuda al usuario a entender la realidad de la tecnología y tiene una notación gráfica muy expresiva que permite representar todas las fases de un proyecto informático: desde el análisis con los casos de uso, el diseño con los diagramas de clases, objetos, hasta la implementación y configuración con los diagramas de despliegue. (8)

1.6.2 Tecnologías web para la visualización.

Las tecnologías web permiten visualizar de forma flexible y rentable la información, ofrecen muchas posibilidades útiles en la automatización. Resulta especialmente interesante para la

operación y monitorización de instalaciones. Para ello, la información binaria del mundo de las máquinas se adapta a la percepción humana. Esto quiere decir, que la tecnología web es ideal para la interacción entre el hombre y la máquina. Para una operación y monitorización flexible la tecnología web posee muchas ventajas porque es económica, independiente de las plataformas, de modo que se puedan usar dispositivos HMI independientemente del tipo de sistema de control.

1.6.2.1 HTML

Se selecciona HTML en el desarrollo de la solución propuesta para crear los elementos de la interfaz visual, permitiendo de esta manera que pueda ser visualizado en todo tipo de navegador web. El propio W3C1 define el lenguaje HTML como “un lenguaje reconocido universalmente y que permite publicar información de forma global”. Desde su creación, el lenguaje HTML ha pasado de ser un lenguaje utilizado exclusivamente para crear documentos electrónicos, a ser un lenguaje que se utiliza en muchas aplicaciones electrónicas como buscadores, tiendas online y banca electrónica. HTML5 es la actualización de HTML, el lenguaje en el que es creada la web. HTML5 también es un término de marketing para agrupar las nuevas tecnologías de desarrollo de aplicaciones web: HTML5, CSS3 y nuevas capacidades de JavaScript. (9)

La versión anterior y más usada: HTML4, carece de tecnologías necesarias para la creación de aplicaciones modernas basadas en un navegador. El uso fuerte de JavaScript ha ayudado a su mejora, gracias a *framework*s como jQuery, jQuery UI, ExtJS, AngularJS, entre otros. Ahora HTML5 es capaz de hacer esto sin necesidad de *plugins* y con una gran compatibilidad entre navegadores. (10)

1.6.2.2 CSS

CSS es un lenguaje de hojas de estilos, creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML[2]. CSS es seleccionado para el desarrollo de la solución propuesta porque imprescindible para crear páginas web complejas, además mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes. (11)

1.6.2.3 Scalable Vector Graphic (SVG)

[2] Versión avanzada del lenguaje de etiquetado de hipertexto basado en XML.

Se selecciona SVG en el desarrollo de la solución propuesta para la visualización de componentes gráficos más complejos, específicamente aquellos componentes que representan los diferentes dispositivos del sistema SCADA Guardián del ALBA.

Es un estándar de la W3C que define una gramática XML para describir gráficos de dos dimensiones. Cualquier programa, tales como un navegador que reconozca XML pueden mostrar la imagen usando la información proporcionada en el formato SVG. La parte del término "gráficos escalable" hace hincapié en que las imágenes SVG se pueden ampliar fácilmente, a diferencia de las imágenes especificadas en gráficos de mapa de bits. Así, el formato SVG permite la visualización de una imagen sobre una pantalla de ordenador de cualquier tamaño y resolución, ya sea una pequeña pantalla LCD en un teléfono celular o una gran pantalla CRT en una estación de trabajo. Además de la facilidad de la reducción del tamaño y la ampliación, SVG permite que los textos dentro de imágenes para ser reconocidas como tales, por lo que el texto puede ser localizado por un motor de búsqueda y fácilmente traducible a otros idiomas. Otra ventaja potencial sobre los formatos de imagen estándar de la web (GIF y JPEG) es su tamaño, en comparación con una imagen de mapa de bits, una imagen SVG puede ser mucho menor lo que disminuye el tiempo de carga. Los gráficos SVG pueden ser interactivos y dinámicos, con animaciones que pueden ser definidas y activas de modo declarativo o por medio de scripts. (12)

1.6.3 Mecanismo de comunicación en la web.

Una excelente comunicación entre servidores y clientes permite obtener mejores resultados en las aplicaciones web. La necesidad de tener una comunicación en tiempo real entre el servidor y el cliente cada día es mayor. Existen tecnologías que permiten que el servidor envíe datos al cliente al detectar un cambio. Entre estas se destacan las siguientes.

1.6.3.1 Mecanismo de comunicación asíncrona AJAX.

Es uno de los mecanismo de comunicación más utilizado en el mundo, el término AJAX es un acrónimo de *Asynchronous JavaScript* + *XML,* que se puede traducir como "JavaScript asíncrono + XML", éste usa el objeto *XMLHttpRequest* para enviar solicitudes asincrónicamente sin la necesidad de recargar la página web. (13)

Este mecanismo posee algunas técnicas, las cuales son:

- **AJAX Polling**

Es usada para simular una interacción con el servidor en tiempo real, consiste en abrir una conexión AJAX cada cierto intervalo de tiempo con objeto de recibir datos del servidor. Si el intervalo es lo suficientemente pequeño, la sensación es la de comunicación en tiempo real.

- **AJAX Long Polling**

Es creada para optimizar las desventajas que posee AJAX Polling. En este caso, si el servidor no tiene nada que enviar, la conexión se mantiene abierta esperando, como en AJAX Polling. En el momento que envía algo y se procesa, en lugar de esperar un intervalo de tiempo para establecer la conexión, se realiza inmediatamente otra petición AJAX.

- **AJAX Multipart Streaming**

Esta técnica permite mantener una conexión abierta con el servidor si se fija el tipo MIME de la respuesta como multipart/x-mixed-replace. Esta variante de comunicación la introdujo Netscape en 1995 como una característica de su navegador a la que llamó server push con la que pretendía que habilitar a los servidores a enviar nuevas versiones de los elementos de una página web. Sin embargo, los únicos navegadores que aceptan esta técnica son los que están basados en Gecko, el motor de Mozilla, como por ejemplo Firefox. (14)

1.6.3.2 Mecanismo de comunicación asíncrona WebSocket.

La especificación de HTML5 WebSockets, define la API de sockets que permite a las páginas web usar el protocolo Secure Socket web, para comunicación bidireccional con un host remoto. Se presenta la interfaz WebSocket y define un canal de comunicación bidireccional, que opera a través de una sola toma en la Web. HTML5 WebSockets pone a disposición del programador una conexión de socket, a través de Internet, con una sobrecarga mínima. Proporcionando una enorme reducción del tráfico innecesario en la red y disminuyendo la latencia. Estas soluciones se utilizan a menudo para empujar datos en tiempo real a los clientes, o incluso simular una conexión bidireccional, para mantener dos conexiones HTTP.

Para conectarse desde cliente web a un servidor utilizando HTML5 WebSockets, se crea una instancia nueva WebSocket y se le pasa la dirección URL del servidor. La especificación define un ws:// y un wss:// ,esquemas para indicar las conexiones WebSocket y las conexiones WebSocket seguras, respectivamente. Una conexión WebSocket se establece mediante la actualización del protocolo HTTP, al protocolo Secure Socket Web, durante la conexión inicial entre el cliente y el servidor, sobre la misma conexión subyacente TCP/IP. (15)

1.6.3.3 Mecanismo de comunicación asíncrona *Server Sent Event* (SSE).

SSE es una especificación del estándar HTML5 orientada a crear un protocolo de comunicación unilateral, mediante el cual se envían datos desde el servidor al cliente. Su funcionamiento comienza cuando el cliente abre una secuencia de eventos, mediante la

creación de un *EventSource*, que tiene una dirección URL de origen de evento como parámetro. El controlador de eventos *onMessage* se llamará cada vez que el origen genere un nuevo dato. Como AJAX, que se pueden comunicar de forma asíncrona desde el cliente al servidor y, ahora con SSE se puede hacer en la dirección opuesta, desde el servidor al cliente, una vez más de forma asincrónica (16)

1.6.3.4 Mecanismo de comunicación asíncrona extensible de mensajería y de presencia (XMPP).

Es una tecnología libre para la comunicación en tiempo real, que usa XML 5 como el formato base para el intercambio de información. En esencia, XMPP provee la forma de enviar pequeñas porciones de XML de una entidad a otra en el menor tiempo posible. Aun cuando XMPP fue desarrollado originalmente en la comunidad de código abierto *Jabber*, el protocolo en sí no es un proyecto de código abierto como Apache, sino más bien un estándar abierto como HTTP. Como resultado, XMPP es una tecnología abierta que no está ligado a ningún proyecto de software o empresa. Las especificaciones XMPP definen protocolos abiertos que se usan para la comunicación entre entidades de la red. XMPP define los protocolos y formatos de datos que alimentan en tiempo real las interacciones a través de Internet. (17)

Selección del mecanismo de comunicación asíncrona para el desarrollo de la aplicación.

Se realizó un análisis minucioso en el Centro de Informática Industrial (CEDIN) de las tecnologías anteriormente descritas para la creación del servidor (HMIServer), donde se obtuvo WebSocket como el mecanismo de comunicación utilizado para la transmisión de los datos hacia los clientes en la web.

1.6.4 Formato para el intercambio de datos.

Es la serialización que se utiliza en el intercambio de datos entre la solución propuesta y el servidor HMIServer del SCADA GALBA. Se emplea como formato JSON para transferir la información obtenida. Se seleccionó debido a que su simplicidad es perfecta para la transmisión de datos estructurados entre servidores y aplicaciones web a través de la red. Su facilidad de implementación le otorga un gran desempeño y lo convierten en una de las alternativas ideales al momento de reemplazar XML.

1.6.5 Lenguaje de programación.

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes. Tiene la capacidad de especificar, de forma precisa, cuáles son los datos que debe trabajar un equipo informático, de qué modo deben ser conservados o transferidos dichos datos y qué instrucciones debe poner en marcha la computadora ante ciertas circunstancias.

Se seleccionó JavaScript para programar la aplicación web. Porque es un lenguaje de programación interpretado que se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Comenzó como una manera de manipular elementos de las páginas web (como imágenes y campos de formulario), pero ha crecido enormemente. Además de las secuencias de código del lado cliente, en estos días se puede usar JavaScript para programar en una creciente variedad de plataformas. Puede escribir código del lado servidor (usando. NET o Node.js), aplicaciones de escritorio (que funcionan en todos los sistemas operativos) y las extensiones de aplicación (por ejemplo, para Firefox o Photoshop), aplicaciones móviles, y los scripts de línea de comandos (18)

1.6.6 Framework de desarrollo.

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. Entre los diferentes framewoks que existen se destacan por su facilidad y eficiencia en la implementación los de JavaScript porque permiten la reutilización de código, ofrecen animación, efectos, movimiento y formas dinámicas. (19)

En el desarrollo de software, un *framework* es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado. 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.

1.6.6.1 JQuery

Se decide utilizar jQuery para el desarrollo de la solución propuesta porque es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos,

desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Es un *framework* para javascript que permite desarrollar sitios de manera más dinámica y fácil; ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio. Entre sus características principales se encuentra la manipulación de la hoja de estilos CSS, los efectos y animaciones, las animaciones personalizadas y es compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+.5. A su vez permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX. (20)

1.6.6.2 AngularJS

Para aumentar las potencialidades del lenguaje JavaScript así como para mejorar el proceso de desarrollo se decidió utilizar AngularJS como *framework*. Porque es un potente *framework* para crear aplicaciones web en el lenguaje del cliente con JavaScript ejecutándose con el conocido single-page applications (aplicación de una sola página) que extiende el tradicional HTML con etiquetas propias (directivas). Permite extender HTML con etiquetas personalizadas, definir y vincular (data-binding) variables vista/controlador, consultas AJAX con peticiones HTTP, sistema óptimo de plantillas, manipulación de datos en JSON, inyección de dependencias, formularios de validación, desacoplamiento del DOM de JavaScript, internacionalización i18n y l10n, filtros, unidad de testeo, es compatible con los navegadores de última generación(Chrome, Firefox, Safari, Opera, Webkits, IE9+). AngularJS brinda muchas facilidades para hacer aplicaciones web, aplicaciones de gestión o de negocio, aplicaciones que funcionan en dispositivos y que tienen un rendimiento muy similar a las nativas e incluso aplicaciones de escritorio con unos frontales web, cada vez más habituales. (21)

1.6.7 Herramienta CASE.

Las herramientas CASE (*Computer Aided Software Engineering*) fueron diseñadas para aumentar la productividad en el desarrollo de software, reduciendo el costo de estos en términos de tiempo y dinero. Corresponden a diversas aplicaciones informáticas, que incluyen un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de un proyecto a desarrollar. Además facilitan el uso de las distintas metodologías propias de la ingeniería del software incluyendo AUP.

Se seleccionó como herramienta CASE el Visual Paradigm porque soporta el ciclo de vida completo del desarrollo de software, permite modelar los diagramas de la solución propuesta, realizar ingeniería tanto directa como inversa, a partir de un modelo relacional en SQL Server, MySQL, PostgreSQL, es capaz de desplegar todas las clases asociadas a las tablas (siguiendo el patrón de diseño Una Clase-Una Tabla). Soporta múltiples usuarios trabajando sobre el mismo proyecto, permite el control de versiones y generar la documentación automáticamente en formatos como web o PDF. Además este software de modelado UML ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. A su vez es una herramienta libre y multiplataforma. (22)

1.6.8 Entorno de desarrollo.

Un entorno de desarrollo integrado, conocido también como IDE (*Integrated Development Environment*) por sus siglas en inglés, es un programa informático compuesto por un conjunto de herramientas de programación que facilitan el desarrollo de aplicaciones. Un IDE puede denominarse como un entorno de programación, esto significa que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Los IDE proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, PHP, C#, Delphi, Visual Basic.

Se propone como IDE de desarrollo *Brackets* porque es una excelente herramienta de diseño y desarrollo web de código abierto que ha sido creado con tecnologías web. Entre sus características principales se destacan que es un editor de código sencillo y ligero, sin paneles de opciones ni barras de iconos que sobrecarguen la interfaz, permite editar los estilos CSS directamente en el documento HTML sin necesidad de abrir el archivo, brinda sugerencias de código para las etiquetas y atributos HTML, los selectores y propiedades de las hojas de estilo CSS y las palabras reservadas, variables locales, argumentos y nombres de propiedades de JavaScript. Es modular y extensible mediante extensiones. *Brackets* integra el navegador con un editor web, por lo que se puede trabajar directamente en el editor de código y ver los resultados simultáneamente en el navegador; se diferencia de los demás editores gracias a la facilidad de mostrar el código específico de acuerdo al contexto usado. (23)

1.7 Consideraciones parciales.

En el presente capítulo se describieron los sistemas de Supervisión, control y adquisición de datos (SCADA), el sistema SCADA Guardián del ALBA, el modulo Interfaz Hombre-Máquina (HMI) y los visualizadores web de los sistemas SCADA.

Después de haber analizado las metodologías, herramientas y tecnologías que hoy en día enriquecen a la informática se decidió utilizar para guiar el proceso de desarrollo de la solución propuesta la metodología AUP. Entre las tecnologías y las herramientas existentes se seleccionó el lenguaje de modelado el UML y el *Visual Paradigm* como herramienta CASE ya que se emplean para visualizar, construir y documentar los artefactos del sistema, para la visualización web se seleccionó las tecnologías estándares HTML para crear los elementos de la interfaz visual permitiendo que la solución propuesta pueda ser visualizada en más de un navegador web. CSS para controlar el aspecto o presentación de la interfaz. SVG para la visualización de los componentes gráficos más complejos, específicamente aquellos componentes que representan los diferentes dispositivos del sistema SCADA GALBA. Para el intercambio de datos se seleccionó el formato JSON porque su simplicidad es perfecta para la transmisión de datos estructurados entre el servidor HMIServer y la solución propuesta a través de la red. Como lenguaje de programación se seleccionó JavaScript y como *frameworks* de desarrollo AngularJS y JQuery. Finalmente como entorno de desarrollo Brackets por ser una excelente herramienta de diseño y desarrollo web de código abierto.

Capítulo 2: Análisis y diseño del sistema.

2.1 Introducción

En el presente capítulo se reflejan las actividades realizadas en los procesos de análisis y diseño de la solución propuesta; proceso que será guiado por la metodología de desarrollo AUP. En el mismo se realiza el modelo de dominio donde se describen las entidades que intervienen con el objetivo de facilitar la comprensión de los principales conceptos que se utilizarán en el proceso de negocio identificado. Se exponen los artefactos más importantes que describen el flujo normal de eventos que ocurren en el sistema, se realiza una descripción de la solución propuesta, planteándose los requisitos funcionales y no funcionales, el diagrama de casos de uso del sistema y la descripción de los casos de uso críticos.

2.2 Descripción de las acciones vinculadas al campo de acción.

La presente investigación está orientada al desarrollo de una aplicación web, capaz de conectarse con la capa de comunicaciones del SCADA Guardián del ALBA, a través del servidor HMIServer y adquirir la información (puntos, despliegues y alarmas) y visualizarla en un entorno web. La solución debe ser capaz de realizar el proceso de autenticación y carga de la configuración, así como de visualizar los diferentes despliegues, las últimas 5 alarmas emitidas por el sistema, el sumario de puntos, el sumario de dispositivos, el sumario de subcanales, el sumario de alarmas, así como los detalles de un punto, de los dispositivos y de subcanales.

2.3 Modelo de dominio.

El modelo de dominio es “la representación visual de los conceptos u objetos más importantes de un negocio, sus características y las relaciones entre dichos conceptos. Es el mecanismo fundamental para comprender el dominio del problema y para establecer conceptos comunes. Es un diccionario visual del dominio del problema.” (24)

A continuación se presentan las clases del dominio perteneciente a la solución.

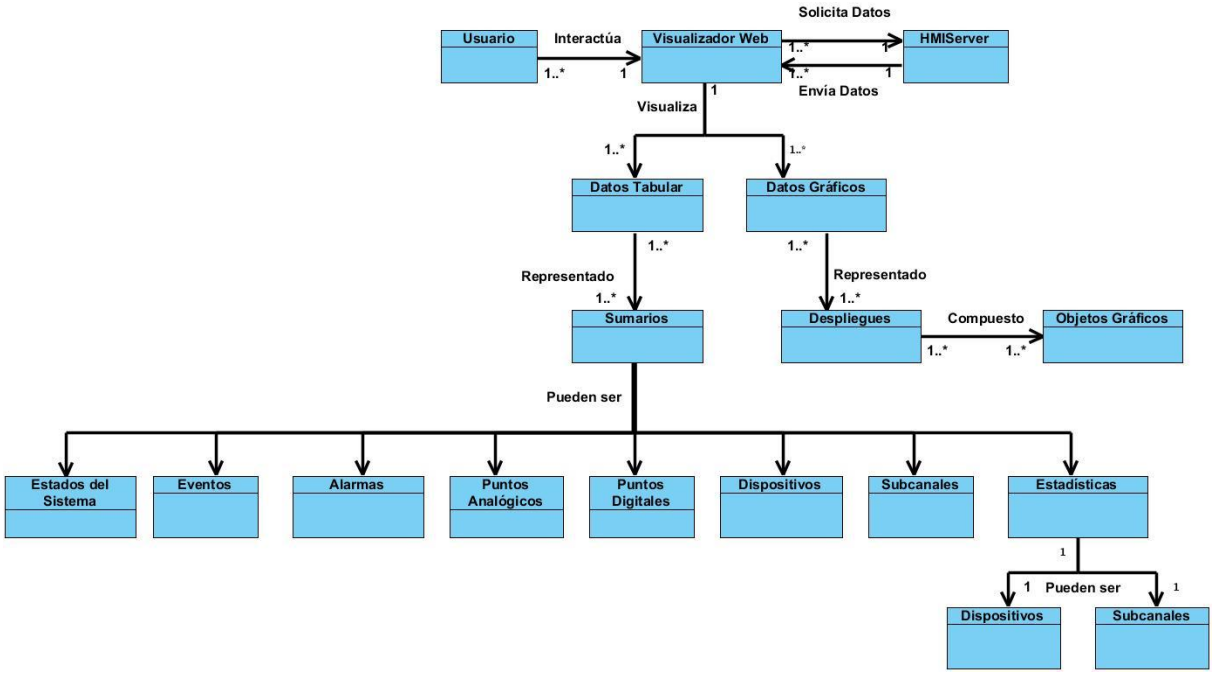


Figura 2: Modelo de dominio.

En la supervisión de los procesos, el Usuario interactúa con el Visualizador Web, el cual solicita los datos al HMIServer y éste le envía la información solicitada. Los datos se pueden visualizar de las siguientes maneras:

- Fomato Tabular mediante Sumarios: que pueden ser de Eventos, Alarmas, Puntos Analógicos, Puntos Digitales, Dispositivos, Subcanales, Estadísticas de Dispositivos, Estadísticas de Subcanales.
- Formato Gráfico mediante Despliegues: que están compuestos por objetos gráficos.

2.3.1 Descripción de los conceptos del dominio.

Tabla 1: Descripción de los conceptos del dominio.

Conceptos del dominio	Descripción
Usuario	Persona capacitada para interactuar con el visualizador web.
Visualizador Web	Herramienta encargada de la visualización de los datos.
HMIServer	Herramienta encargada en gestionar las peticiones provenientes del visualizador web.
Datos Tabular	Representación de la información en forma de tuplas.

Datos Gráficos	Representación de la información en forma de objetos gráficos.
Sumarios	Conjunto de elementos representados en tabla, y posee las siguientes acciones: filtro y paginado.
Despliegues	Conjunto de elementos representados mediante objetos gráficos, permitiendo visualizar los datos en tiempo real.
Objetos Gráficos	Representación gráfica de cada dato, en correspondencia con sus propiedades.
Eventos	Tipo de datos que pueden ser visualizados en un sumario, representan los sucesos ocurridos durante la ejecución de los procesos.
Alarmas	Tipo de datos que pueden ser visualizados en un sumario.
Puntos Analógicos	Tipo de datos que pueden ser visualizados en un sumario.
Puntos Digitales	Tipo de datos que pueden ser visualizados en un sumario.
Dispositivos	Tipo de datos que pueden ser visualizados en un sumario.
SubCanales	Tipo de datos que pueden ser visualizados en un sumario.
Estadísticas	Tipo de sumario; que se encarga de la recopilación de elementos dependiendo de un intervalo de tiempo y el tipo de elemento a visualizar.
Dispositivos	Tipo de datos que pueden ser visualizados en un sumario.
Subcanales	Tipo de datos que pueden ser visualizados en un sumario.
Estados del Sistema	Tipo de datos que pueden ser visualizados en un sumario.

2.4 Especificación de requisitos.

Conocer los requisitos del sistema que se va a desarrollar es el primer paso en el desarrollo de su implementación, de esta manera se reducirá el número de cambios que habrá que realizar en el producto debido a los cambios en sus requisitos. A partir del modelo del dominio presentado se realizó el levantamiento de los requisitos, los cuales están divididos en dos tipos: los funcionales y los no funcionales. A continuación se especifican los requisitos definidos para el desarrollo del sistema propuesto.

2.4.1 Requisitos funcionales.

Los requisitos funcionales (RF) del sistema propuesto se identificaron de acuerdo a las capacidades o condiciones que este debe cumplir. Los mismos expresan una especificación detallada de cómo reacciona el sistema a una entrada particular y cómo se comporta ante situaciones particulares, es decir: sus entradas, salidas y excepciones. (S. Pressman, 2001)

Teniendo en cuenta las funcionalidades que el sistema debe proveer se especificaron 28 RF los cuales se definen a continuación.

RF1: Iniciar la aplicación.

El sistema debe mostrar una interfaz inicial indicándole al usuario que la aplicación está inicializándose, posteriormente debe obtenerse la configuración de la aplicación e inicializar todos los componentes y los módulos que forman parte de la aplicación. Se debe establecer la comunicación con el servidor y comenzar el intercambio de información entre el controlador y la vista. Una vez que el usuario inicie sesión por primera vez el sistema debe mostrar el sumario reducido de alarmas.

RF2: Visualizar datos de usuario.

El sistema debe visualizar los datos del usuario tales como el nombre de usuario y el tiempo restante de sesión.

RF3: Mostrar fecha y hora actual.

El sistema debe visualizar la fecha y la hora actual en la parte inferior derecha de la interfaz. Las mismas deben estar sincronizadas con el servidor y actualizarse conforme vaya pasando el tiempo.

RF4: Interacción con el módulo de Seguridad.

El sistema debe permitir la interacción con el módulo de Seguridad para realizar las siguientes acciones:

> **RF4.1:** Autenticar usuario.
>
> El sistema debe poseer una interfaz de autenticación para que el usuario introduzca los datos correspondientes. La interfaz debe presentar los siguientes campos: Usuario y Contraseña.
>
> **RF4.2:** Cambiar usuario.

El sistema debe mostrar en un menú de acciones de usuario la opción de Cambiar usuario. Esta acción debe mostrar la interfaz de autenticación para ingresar los datos del nuevo usuario. La interfaz debe presentar los siguientes campos: Usuario y Contraseña.

RF4.3: Cambiar contraseña.

El sistema debe mostrar en la barra de navegación una opción que permite desplegar un menú, en el que se muestran la opción Cambiar contraseña. Esta acción debe mostrar una interfaz para ingresar la nueva contraseña.

RF4.4: Cerrar sesión.

El sistema debe mostrar en un menú de acciones de usuario la opción de cerrar la sesión.

RF5: Visualizar sumario reducido de las alarmas.

El sistema debe mostrar siempre el sumario reducido de alarmas, mostrando las 5 alarmas más relevantes de acuerdo a su tiempo de emisión.

RF6: Visualizar sumario de alarmas.

El sistema debe permitir al usuario desde el ambiente de ejecución visualizar el estado de las alarmas activas, ordenas por criticidad del sistema.

RF7: Visualizar sumario de puntos analógicos.

El sistema debe permitir al usuario desde el ambiente de ejecución visualizar la información principal de los puntos analógicos.

RF8: Visualizar sumario de puntos digitales.

El sistema debe permitir al usuario desde el ambiente de ejecución visualizar la información principal de los puntos digitales.

RF9: Visualizar sumario de dispositivos.

El sistema debe permitir al usuario desde el ambiente de ejecución visualizar la información principal de los dispositivos.

RF10: Visualizar sumario de subcanales.

El sistema debe permitir al usuario desde el ambiente de ejecución visualizar la información principal de los subcanales.

RF11: Visualizar sumario de histórico de eventos.

El sistema debe permitir al usuario desde el ambiente de ejecución visualizar los eventos generados por el sistema y almacenados en la base de datos históricos. Entre las funcionalidades que presenta este sumario se encuentran realizar consultas de tiempo y filtros sobre las mismas.

RF12: Visualizar sumario de estadísticas de dispositivos.

El sistema debe ser capaz de representar el sumario de estadísticas de los dispositivos según la información almacenada en históricos, para ello muestra contadores de estados y porcentajes de éxito para las comunicaciones.

RF13: Visualizar sumario de estadísticas de subcanales.

El sistema debe ser capaz de representar el sumario de estadísticas de los subcanales según la información almacenada en históricos, para ello muestra contadores de estados y porcentajes de éxito para las comunicaciones.

RF14: Visualizar monitor del sistema.

El sistema debe visualizar en la esquina inferior derecha una imagen que represente el estado del sistema, el cual debe actualizarse en dependencia de un cambio en el estado de la comunicación con los servicios del SCADA GALBA.

RF15: Visualizar sumario de estado del sistema.

El sistema debe ser capaz de representar el sumario de estado de los servicios del SCADA GALBA según la información almacenada en el Administrador de servicios.

RF16: Realizar paginado de sumarios.

El sistema debe permitir el paginado de los sumarios, de esta manera se dividen contenidos de gran longitud en varias páginas más cortas. La cantidad de datos a mostrar por página debe ser configurable.

RF17: Filtrar sumarios.

El sistema debe permitir filtrar los sumarios en dependencia de los campos que muestre cada sumario o de algún valor en específico, de esta manera se pueden realizar búsquedas por categorías.

RF18: Visualizar detalle de puntos analógicos.

El sistema debe permitir al usuario visualizar los detalles del punto analógico seleccionado.

RF19: Visualizar detalle de puntos digitales.

El sistema debe permitir al usuario visualizar los detalles del punto digital seleccionado.

RF20: Visualizar detalle de dispositivos.

El sistema debe permitir al usuario visualizar los detalles del dispositivo seleccionado.

RF21: Visualizar detalle de subcanales.

El sistema debe permitir al usuario visualizar los detalles del subcanal seleccionado.

RF22: Gestionar objeto gráfico de tipo medidor.

El sistema debe visualizar y actualizar el objeto gráfico de tipo medidor.

RF22.1: Crear objeto gráfico de tipo medidor.

El sistema debe visualizar el objeto gráfico de tipo medidor.

RF22.2: Actualizar objeto gráfico de tipo medidor.

El sistema debe actualizar el objeto gráfico de tipo medidor.

RF23: Gestionar objeto gráfico de tipo imagen.

El sistema debe visualizar y actualizar el objeto gráfico de tipo imagen.

RF23.1: Crear objeto gráfico de tipo imagen.

El sistema debe visualizar el objeto gráfico de tipo imagen.

RF23.2: Actualizar objeto gráfico de tipo imagen.

El sistema debe actualizar el objeto gráfico de tipo imagen.

RF24: Gestionar objeto gráfico de tipo botón.

El sistema debe visualizar y actualizar el objeto gráfico de tipo botón.

RF24.1: Crear objeto gráfico de tipo botón.

El sistema debe visualizar el objeto gráfico de tipo botón.

RF24.2: Actualizar objeto gráfico de tipo botón.

El sistema debe actualizar el objeto gráfico de tipo botón.

RF25: Gestionar objeto gráfico de tipo nivel.

El sistema debe visualizar y actualizar el objeto gráfico de tipo nivel.

RF25.1: Crear objeto gráfico de tipo nivel.

El sistema debe visualizar el objeto gráfico de tipo nivel.

RF25.2: Actualizar objeto gráfico de tipo nivel.

El sistema debe actualizar el objeto gráfico de tipo nivel.

RF26: Gestionar objeto gráfico de tipo etiqueta.

El sistema debe visualizar y actualizar el objeto gráfico de tipo etiqueta.

RF26.1: Crear objeto gráfico de tipo etiqueta.

El sistema debe visualizar el objeto gráfico de tipo etiqueta.

RF26.2: Actualizar objeto gráfico de tipo etiqueta.

El sistema debe actualizar el objeto gráfico de tipo etiqueta.

RF27: Visualizar sumario de despliegues.

El sistema debe visualizar el sumario de despliegues. El sumario debe presentar los siguientes campos: Nombre y Descripción.

RF28: Visualizar despliegue.

El sistema debe visualizar los despliegues previamente diseñados en el ambiente de edición, con sus respectivos componentes gráficos. A su vez debe permitir las siguientes acciones:

RF28.1: Abrir despliegue.

Cuando se ejecuta la acción de abrir despliegue el sistema debe visualizar en una pestaña los componentes gráficos previamente diseñados en el ambiente de edición para el despliegue seleccionado.

RF28.2 Cerrar despliegue.

Cuando se ejecuta la acción de cerrar despliegue el sistema debe cerrar todos los eventos y objetos que se estén visualizando en el despliegue seleccionado, así como la pestaña que este visualizando a dicho despliegue.

RF28.3 Cambiar de despliegue.

Cuando se ejecuta la acción de cambiar de despliegue el sistema debe activar y desactivar los despliegues que se visualizan.

RF28.4: Actualizar objetos gráficos del despliegue.

El sistema debe realizar la actualización de los valores de los puntos.

2.4.2 Requisitos no funcionales.

Los requisitos no funcionales (RNF) son las propiedades o cualidades que el sistema debe tener, para que sea un producto atractivo, usable, rápido y confiable. (S. Pressman, 2001)

Existen múltiples categorías para clasificar los requisitos no funcionales, siendo las siguientes las más representativas acumulando un total de 11 RNF para el sistema propuesto.

Usabilidad

RNF1: El sistema debe poder ser usado por cualquier persona que tenga conocimientos básicos de computación.

RNF2: El sistema deberá estar disponible en todo momento, limitado solamente por las restricciones que ésta tenga de acuerdo a las políticas de seguridad del usuario.

RNF3: Se debe lograr un producto altamente configurable y extensible, de manera que sea posible incorporar a éste nuevas funcionalidades, sin que se afecte el mismo.

Eficiencia

RNF4: El tiempo de respuesta estará dado por la cantidad de información a procesar, entre mayor cantidad de información mayor será el tiempo de procesamiento.

RNF5: Al igual que el tiempo de respuesta, la velocidad de procesamiento de la información, la actualización y la recuperación dependerán de la cantidad de información que tenga que procesar.

Confiabilidad

RNF6: Una vez que ocurra una excepción producto a un error simple el sistema debe informar y recuperarse ante el fallo, si por el contrario es debido a un error crítico que afecte el funcionamiento del sistema, debe informar el error y a continuación cerrarse.

Interfaz de Usuario

RNF7: El sistema debe tener una interfaz gráfica uniforme incluyendo pantallas, menús y opciones.

RNF8: Tanto los títulos de los componentes de la interfaz, como los mensajes para interactuar con los Usuarios, así como los mensajes de error, deberán ser en idioma español y tener una apariencia uniforme.

RNF9: Se deberá facilitar la entrada de datos a los usuarios, presentando campos de selección que permitan escoger los valores posibles con los que se podrá llenar un determinado elemento en la interfaz, haciendo que el proceso de llenado de datos sea lo más intuitivo posible, de tal forma que los usuarios se puedan adaptar fácilmente.

Software

RNF10: Como Sistema Operativo Linux o Windows, recomendándose utilizar cualquier distribución de Linux o Windows (7 o superior).

RNF11: Como navegador web se podrá utilizar Firefox(a partir de la versión 4), Chrome(a partir de la versión 8), recomendándose el último especificado.

2.5 Modelo de Casos de Uso del sistema.

El modelo de casos de uso representa un esquema que recoge las funcionalidades del sistema que se automatizan y determina el uso que tendrá desde el punto de vista del usuario (25). Para el sistema propuesto el modelo de casos de uso está compuesto por los actores que interactúan con él, los requisitos funcionales principales agrupados en casos de uso y la relación que existe entre estos y los actores.

2.5.1 Descripción de los actores del sistema.

Los actores representan un tipo de usuario del sistema y el conjunto de casos de uso al que estos tienen acceso define su rol global en el sistema. (25) En el sistema propuesto se identificó como actor el siguiente:

Tabla 2: Descripción del actor del sistema.

Actor	Descripción
Usuario	Persona encargada directamente del control de los procesos operacionales, de ejecutar y enviar los reportes durante su guardia.

2.5.2 Diagrama de Casos de Uso del sistema.

Un caso de uso (CU) se representa mediante un óvalo y proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico (25). Teniendo en cuenta lo planteado se decide agrupar los 28 RF en 16 CU, destacando solo los principales. A continuación se puede apreciar el diagrama de CU del sistema.

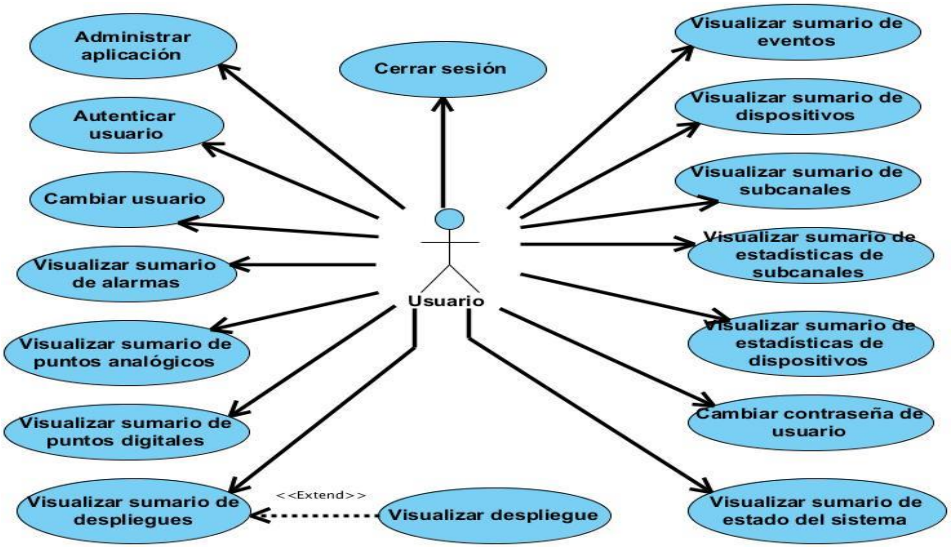


Figura 3: Diagrama de Casos de Uso.

El diagrama de CU del sistema presentado se puede apreciar que el usuario inicia los siguientes casos de uso: Administrar aplicación, Autenticar usuario, Cambiar usuario, Visualizar sumario de alarmas, Visualizar sumario de puntos analógicos, Visualizar sumario de puntos digitales, Visualizar sumario de despliegues, Cerrar sesión, Visualizar sumario de eventos, Visualizar sumario de dispositivos, Visualizar sumario de subcanales, Visualizar sumario de estadísticas de dispositivos, Visualizar sumario de estadísticas de subcanales,

Cambiar contraseña de usuario, Visualizar sumario de estado del sistema, Visualizar despliegue; este caso de uso es una extensión del CU Visualizar sumario de despliegue.

2.5.3 Descripción de los Casos de Uso.

Especificar los casos de uso del sistema permite entender las necesidades del cliente y proveer una visión que posibilite emprender el desarrollo del mismo (25). En este apartado se describirá solamente el CU: "Visualizar despliegue", por ser el que más impacta en la arquitectura del sistema. Para ver las restantes especificaciones consultar Anexo #1.

Tabla 3: Especificación del Caso de Uso "Visualizar despliegue".

Nombre del Caso de Uso	Visualizar despliegue
Actores	Usuario
Propósito	Abrir, cerrar, visualizar un despliegue.
Resumen	Este caso de uso permite al Usuario abrir y cerrar despliegue.
Referencias	RF22, RF23, RF24, RF25, RF26, RF28
Prioridad	Crítico
Precondiciones	El Usuario está registrado en el sistema y pertenece a un grupo operacional de privilegios. Los despliegues se encuentran configurados con parámetros de dirección y comandos permitidos.
Postcondiciones	Si el caso de uso finalizó correctamente el actor visualizó, abrió el despliegue.
Sección "Visualizar despliegue"	
Flujo Normal de los Eventos	
Acciones del Actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el actor se sitúa en el sumario de despliegue.	2. Muestra el listado de despliegues con un botón de visualización por cada despliegue del listado. Para abrir un despliegue: ver sección "Abrir despliegue". Para cerrar un despliegue: ver sección "Cerrar despliegue".
Sección "Abrir despliegue"	
Flujo Normal de los Eventos	
Acciones del Actor	**Respuesta del Sistema**

1. Presiona clic en el botón representado con una lupa del despliegue que desea visualizar.	2. Muestra una pestaña con el nombre del despliegue y con su contenido.
Sección "Cerrar despliegue"	
Flujo Normal de los Eventos	
Acciones del Actor	**Respuesta del Sistema**
1. Se dirige a la pestaña donde se encuentra el despliegue en cuestión y presiona Cerrar.	2. Cierra el despliegue, muestra la interfaz de la aplicación. Finaliza el caso de uso.
Prioridad	Alta

2.6 Patrones de Arquitectura.

La selección de un patrón arquitectónico es una decisión fundamental de diseño en el desarrollo de un sistema de software ya que provee un conjunto de subsistemas predefinidos, especifica sus responsabilidades e incluye reglas y pautas para la organización de las relaciones entre ellos.

En el desarrollo de un software, se hace necesario seleccionar diferentes patrones los cuales ayudan a que esté presente una buena estructura y organización que hace eficiente su funcionamiento. Estos patrones son una guía para cometer una determinada acción. Especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones, para organizar los distintos componentes.

2.6.1 Patrón arquitectónico Modelo Vista Vista Modelo.

El patrón *Model View ViewModel* (MVVM) es una variante de otro patrón muy conocido, llamado *Model - View - Controller* (MVC). Este patrón se emplea en un gran número de marcos; especialmente en el marco de aplicaciones web de uso extenso *Ruby on Rails*, además de ASP.NET MVC de Microsoft y AngularJS. No sólo se emplea en las aplicaciones web, sino que también se usa extensamente desde las aplicaciones de escritorio hasta las aplicaciones móviles.

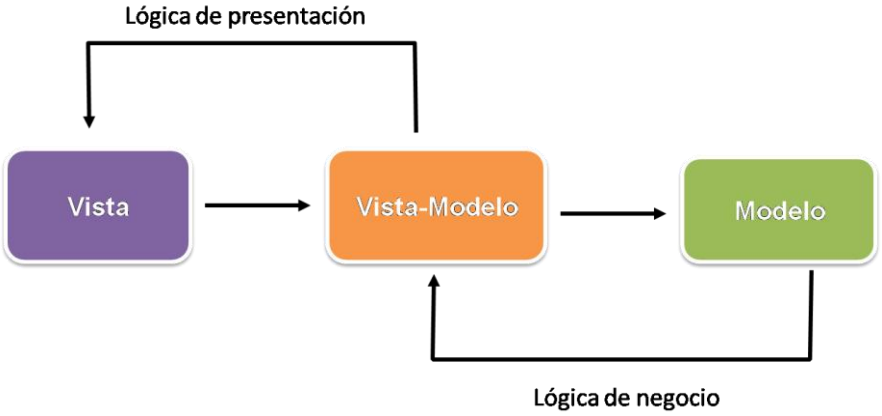


Figura 4: Representación del patrón arquitectónico Modelo Vista -Vista Modelo.

Como se describe en la figura anterior el patrón arquitectónico seleccionado está compuesto por 3 entidades:

-Modelo: representa el dominio del negocio: acceso a datos, clases modelos, reglas del negocio;

-Vista: figura la interfaz de Usuario de la aplicación.

-Vista-Modelo: resulta ser una capa intermedia entre el modelo y la vista, con el objetivo de procesar todas las peticiones que tenga la vista hacia el modelo, además de ocuparse del manejo de las reglas del negocio.

A continuación se realiza una representación del patrón arquitectónico utilizado en la solución propuesta con el *framework* AngularJS:

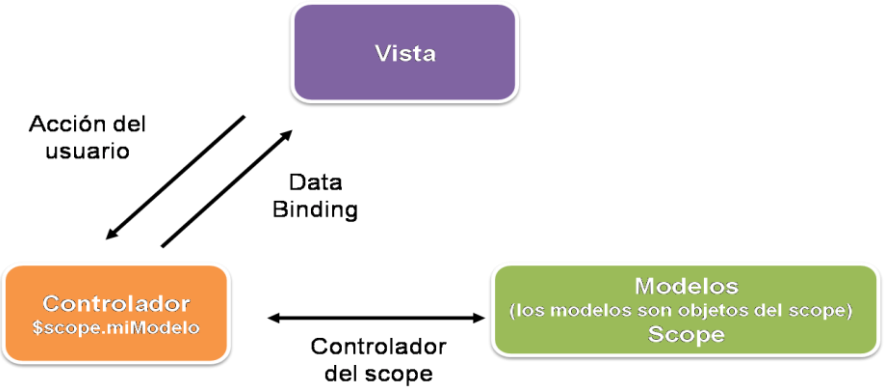


Figura 5: Patrón arquitectónico MVVM en el *framework* AngularJS.

El scope es el elemento conector entre el controlador y la vista; el Usuario realiza una acción en la vista y el controlador manipula dicha acción de entrada y modifica los datos del modelo

como sea necesario. Al poseer una data binding [3] de doble sentido los cambios dentro del modelo de datos se reflejan automáticamente en la interfaz y viceversa.

2.7 Patrones de diseño.

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software, es decir brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Esto provee las siguientes ventajas: reduce los esfuerzos de desarrollo y mantenimiento, mejora la seguridad informática, eficiencia y consistencia del diseño, y proporciona un considerable ahorro en la inversión. (26)

A continuación se describe la importancia de los patrones de diseño y como se aplican en el desarrollo de la estructura del sistema.

2.7.1 Pandilla de los Cuatro conocidos como GoF (*Gang of Four*).

Los patrones GoF se utilizan durante la implementación del sistema con el objetivo de poder solucionar los problemas que suelen ser comunes cuando se está desarrollando un software. A continuación se describen los que se seleccionaron para el desarrollo de la solución propuesta:

- **Observador (*Observer*):**

En un entorno orientado a eventos, como son los navegadores web, donde se busca constantemente la atención de un usuario, el patrón de observador, también conocido como el patrón Publicador-Suscriptor, es una excelente herramienta para gestionar la relación entre las personas y sus puestos de trabajo, o más bien, los objetos, sus acciones, y su estado. En cuanto a JavaScript, este modelo le permite volver a observar el estado de un objeto en un programa y ser notificado cuando se cambia. En el patrón de observador, hay dos papeles: el observador y el observado (ver o ser visto) La aplicación desarrollada utiliza este patrón para trabajar con todos los eventos generados por el usuario y para la interrelación con los datos publicados desde el servidor (Stefanov, 2010)

- **Decorador (*Decorator*):**

[3] Data binding es el proceso que establece conexiones entre la interfaz de Usuario y la lógica de una aplicación. Si las notificaciones y la configuración son las adecuadas los datos reflejan los cambios en cuanto estos suceden. En el caso de tener un data binding de doble sentido también quiere decir que cuando la interfaz de Usuario es modificada los datos que haya por debajo reflejarán los cambios.

Es un patrón de diseño estructural que promueve la reutilización de código y es una alternativa flexible a la sub-clasificación. Este patrón es también útil para modificar los sistemas existentes, donde es posible que desee añadir características adicionales a los objetos sin la necesidad de cambiar el código subyacente que los utiliza. (18). La aplicación desarrollada hace uso de este patrón específicamente en la interrelación entre las clases *tabDecorator* y, *tab*.

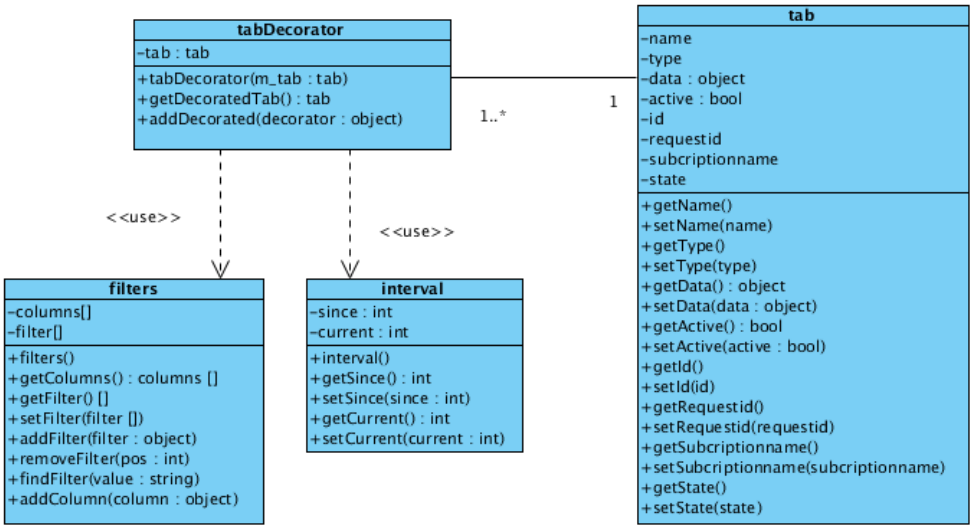


Figura 6: Representación del patrón Decorador.

- **Instancia Única (*Singleton*)**

Proporciona una forma de agrupar el código en una unidad lógica, que se puede acceder a través de una sola variable. Al asegurar que sólo existe un objeto único, se sabe que todo el código hace uso de los mismos recursos globales (18). La aplicación desarrollada hace uso de este patrón específicamente en la clase encargada de la seguridad (*securitySrv*)

2.7.2 Patrones Generales de Software para Asignar Responsabilidades (GRASP).

Para el diseño de la aplicación de monitoreo se tienen en cuenta los patrones GRASP, los cuales describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. A continuación se muestra una selección de estos patrones los cuales serán utilizados durante la realización del diseño de la aplicación:

- **Experto:**

Es el patrón encargado de asignar responsabilidades; expresa que siempre se debe asignar una responsabilidad al experto en información, o sea, a la clase que cuenta con la información necesaria para llevar a cabo la funcionalidad. Este patrón se pone de manifiesto en las clases *tabsSrv* y *securitySrv*.

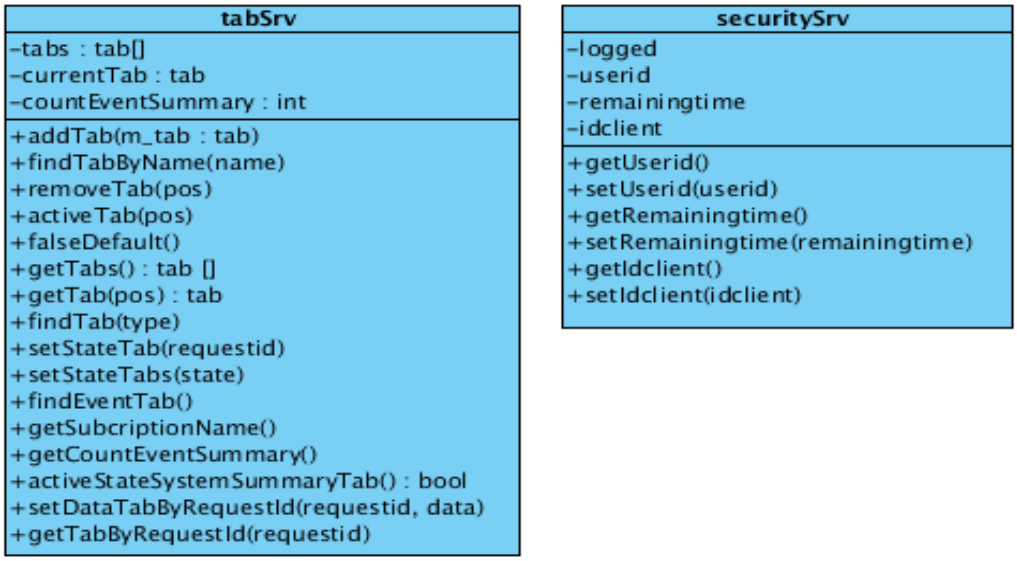


Figura 7: Representación del patrón Experto.

- **Creador:**

Permite asignar el responsable de la creación de una nueva instancia de alguna clase. Explica que clase es la encargada de crear objetos, en determinados escenarios de ejecución y guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito general de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Este patrón se evidencia en la clase *requestSrv*.

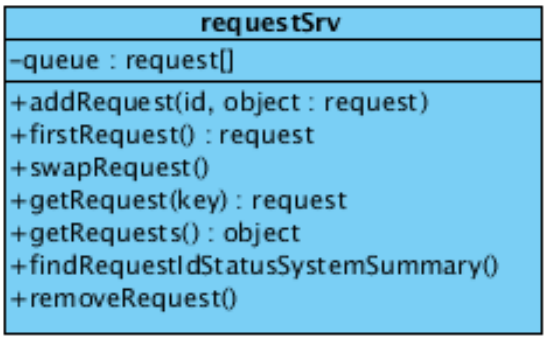


Figura 8: Representación del patrón Creador.

- **Bajo Acoplamiento:**

Permite impulsar la asignación de responsabilidades de manera que su localización no incremente el acoplamiento hasta un nivel que lleve a los resultados negativos que puede

producir un acoplamiento alto. No soporta el diseño de clases que son más independientes, lo que reduce el impacto del cambio. El mismo no se puede considerar de manera aislada a otros patrones como el Experto o el de Alta Cohesión, sino que necesita incluirse como uno de los diferentes principios de diseño que influyen en una elección, al asignar una responsabilidad.

- **Alta Cohesión:**

Sigue el principio de que cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable. A través de este patrón se simplifica el mantenimiento y las mejoras del funcionamiento del sistema. Las clases que están sobrecargadas de métodos poseen una alta cohesión por lo que se recomienda para un buen diseño la creación de los paquetes de servicio o clases agrupadas por funcionalidades que son fácilmente reutilizables.

2.8 Modelo de diseño.

Una vez que identificadas las clases que participarán en la solución del sistema propuesto se realiza el diagrama de clases del diseño. A diferencia del modelo conceptual, el diagrama de clases del diseño contiene las definiciones de las entidades del software en vez de conceptos del mundo real.

El diagrama de clases expresa la estructura u organización del software en términos de las clases. Es un reflejo abstracto de los componentes y las relaciones entre ellos. 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. Cuando se crea un diagrama de clases, se está modelando una parte de los elementos y relaciones que configuran la vista de diseño del sistema. A continuación se muestra el diagrama de clases del diseño pertenecientes al CU “Visualizar despliegue”. Para ver los restantes modelos del diseño consultar Anexo #2.

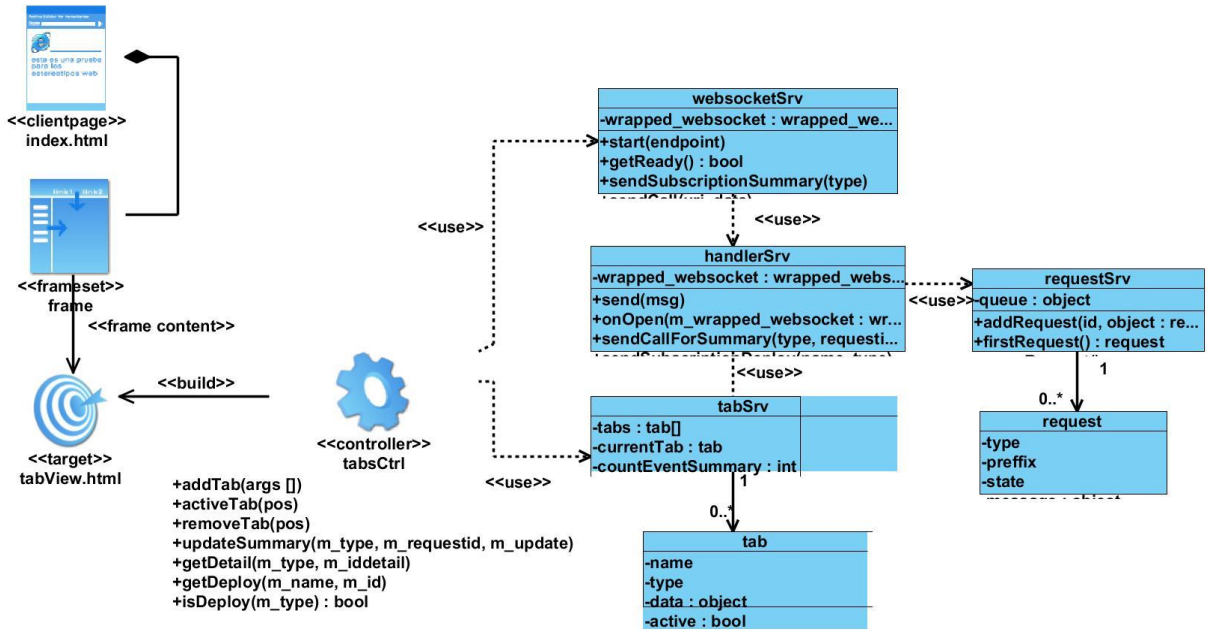


Figura 9: Modelo de diseño del Caso de Uso “Visualizar despliegue”.

El flujo de acciones del diagrama de clases del CU Visualizar despliegue se realiza de la siguiente forma: el usuario selecciona dentro de la pestaña llamada “Despliegue” perteneciente a la vista *tabView* el despliegue a visualizar haciendo clic al botón titulado “Visualizar”. Esta petición es recibida por el controlador *tabsCtrl*, encargado de crear la pestaña asociada al despliegue y el envío de datos al servidor para realizar la subscripción. Al adicionar la nueva pestaña es actualizado el contenedor de las pestañas *tabSrv* permitiendo al controlador *tabsCtrl* obtener las pestañas ya creadas y proveersela a la vista *tabView* para sus respectiva visualización. La clase *handlerSrv* es la encargada de parsear los datos provenientes del servidor y la asignación de datos a cada módulo del sistema.

2.9 Conclusiones parciales.

En el capítulo presentado se definió el modelo del dominio, el cual refleja el punto de partida de la solución propuesta. Se especificaron además los requisitos del sistema que permitieron identificar las funcionalidades con las que este contará y que darán respuesta a las necesidades del usuario. A raíz de esto se determinaron 28 requisitos funcionales los cuales se agruparon en 16 CU y 11 requisitos no funcionales. Por su impacto en la arquitectura del sistema se describió el CU *“*Visualizar despliegue*”,* el cual está dividido por secciones con el objetivo de lograr una mayor comprensión del mismo.

Se trataron además los temas más importantes referentes al diseño y se definió una arquitectura basada en patrones. Como patrón arquitectónico se propuso el modelo vista-

vista modelo (MVVM) y como patrones de diseño se seleccionaron los patrones GRASP y GoF los cuales posibilitaron definir una mejor arquitectura. Se diseñaron además los artefactos que permitieron dar continuación a la implementación del sistema, entre los que se encuentra el modelo del diseño del CU *"*Visualizar despliegue*"*.

Capítulo 3: Implementación y prueba.

3.1 Introducción.

En el presente capítulo se analizan los elementos necesarios para proceder a la implementación del sistema y se verifica el cumplimiento de los requisitos funcionales mediante las pruebas de software. Además se construye el modelo de implementación correspondiente al sistema propuesto, desglosándolo en el diagrama de componentes del CU más importantes y en la descripción de cada uno de estos componentes, de la misma forma se modela el hardware utilizado en la implementación a través del modelo de despliegue y se determinan los tipos de pruebas a realizar y los casos de pruebas que serán aplicados al sistema.

3.2 Modelo de implementación.

Para la composición física de la implementación del sistema se realizó el modelo de implementación, el cual está integrado por un conjunto de componentes, entre los que se encuentran ejecutables y librerías. Este modelo permitió definir una organización del código en términos de los subsistemas de implementación organizados en capas, implementar los elementos de diseño en términos de elementos de implementación, es decir programas ejecutables, finalmente probar y desarrollar los componentes como unidades.

Para conformar el modelo de implementación se realizaron dos diagramas fundamentales, el diagrama de componentes y el diagrama de despliegue, los cuales se describen a continuación.

3.2.1 Diagrama de Componentes.

Un componente es una parte física y reemplazable de un sistema. Las clases representan abstracciones lógicas y los componentes son elementos físicos del mundo real, siendo estos la implementación física de un conjunto de otros elementos lógicos, como clases y colaboraciones. El diagrama de componentes se representa como un grafo unido por medio de las relaciones de dependencia que existe entre los componentes del sistema y muestra un conjunto de ficheros relacionados entre sí para lograr una completa funcionalidad del sistema.

En el diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión de software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo.

Para visualizar la estructura general del sistema se generó el diagrama de componente, el cual permitió describir el modelo de implementación y las relaciones entre los elementos de dicho modelo.

A continuación se presenta el diagrama de componentes del CU “Visualizar Despliegue”, en donde los componentes se agrupan teniendo en cuenta el patrón arquitectónico Modelo - Vista – Vista_Modelo:

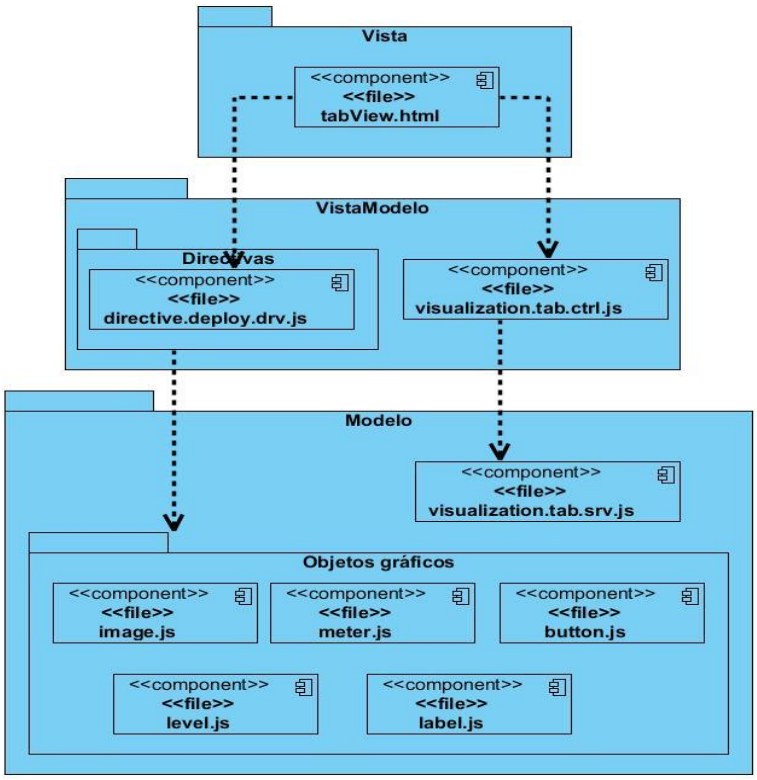


Figura 10: Diagrama de componentes del CU Visualizar despliegue.

El flujo de acciones del diagrama de componentes presentado se define de la siguiente forma: los componentes identificados se agrupan en tres paquetes de implementación básicos el modelo, la vista y la vista_modelo. En el modelo se encuentran los componentes que permiten poseer los datos atómicos de cada objeto perteneciente al despliegue. En la vista se encuentra el componente que permite la representación visual de los datos del despliegue. Finalmente en la vista_modelo, se agrupan los componentes que permiten el control de los datos del despliegue para su respectiva visualización.

3.2.2 Descripción del diagrama de componentes.

Para una mejor comprensión del diagrama presentado en la siguiente tabla se resumen algunos de los componentes utilizados y el propósito de cada uno de estos:

Tabla 4: Descripción de los componentes.

Componente	Propósito
visualization.tab.srv.js	Componente que posee los atributos atómicos asociados a la pestaña del despliegue.
image.js	Componente destinado a poseer las propiedades del objeto gráfico imagen.
meter.js	Componente destinado a poseer las propiedades del objeto gráfico medidor.
button.js	Componente destinado a poseer las propiedades del objeto gráfico botón.
level.js	Componente destinado a poseer las propiedades del objeto gráfico nivel.
label.js	Componente destinado a poseer las propiedades del objeto gráfico etiqueta.
visualization.tab.ctrl.js	Componente destinado a controlar los datos pertenecientes al despliegue.
directive.deploy.drv.js	Componente destinado a transformar los datos que provee el controlador de las pestañas para la correcta visualización de los objetos gráficos.
tabView.html	Componente destinado a representar visualmente los datos pertenecientes al despliegue.

3.3 Modelo de Despliegue.

Con el objetivo de proveer una descripción de la distribución física del sistema se realiza el modelo de despliegue, mediante el cual se muestran las relaciones físicas de los distintos nodos que componen el sistema.

Los diagramas de despliegue son los complementos de los diagramas de componentes que, unidos, proporcionan la vista de implementación del sistema. A su vez describen la topología del sistema, es decir la estructura de los elementos de hardware y el software que consumen. Además muestran las relaciones físicas de los distintos nodos que componen un sistema. A continuación se muestra el modelo de despliegue del sistema en donde los nodos representados mediante estereotipos se conectan mediante soportes bidireccionales.

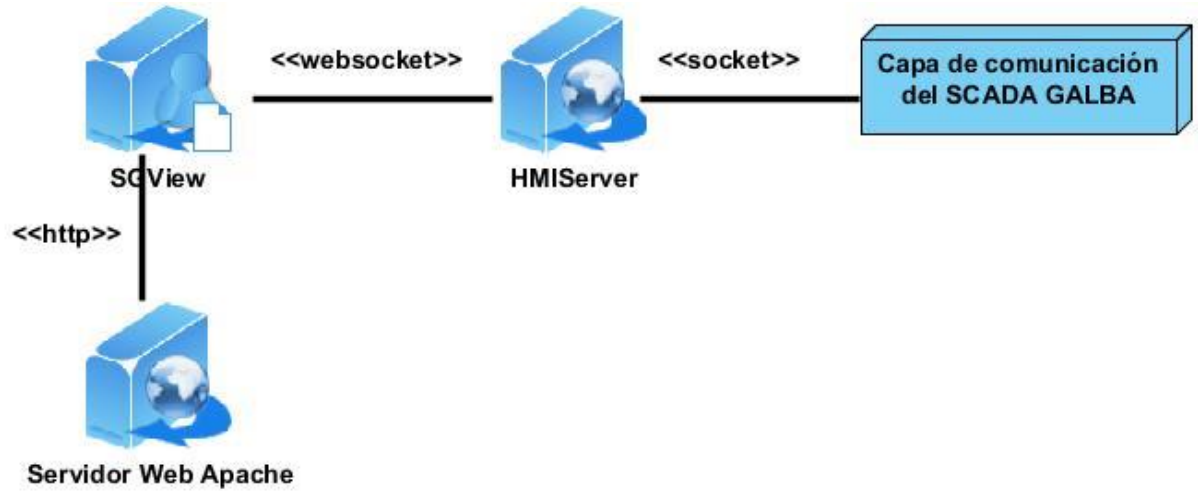


Figura 11: Modelo de despliegue del sistema.

En el diagrama presentado se observa como el software y el hardware trabajan juntos, en donde se especifica que el Visualizador Web (SGVIEW) depende del Servidor Web Apache para su publicación usando el protocolo de comunicación HTTP además de obtener los datos por el protocolo de comunicación WebSocket del HMIServer y éste último mediante TCP/IP se conecta a la capa de comunicación del SCADA GALBA.

3.4 Estándares de codificación

Uno de los instrumentos que facilitan la tarea de asegurar la calidad del software es la adopción de estilos y estándares de codificación. El uso de estos estándares tiene innumerables ventajas entre ellas lograr un estilo de código homogéneo asegurando su legibilidad y proveer una guía para el encargado del mantenimiento/actualización del sistema, con código claro y bien documentado. Además ayuda a mejorar el proceso de codificación haciéndolo en gran medida eficiente y en muchos casos reutilizables.

A continuación se exponen los diferentes estilos de codificación que se utilizaron en la implementación del sistema:

1- Los nombres de las variables y de las funciones deben seguir la notación *lowerCamelCase*[4] con el sufijo Srv para las clases servicios, Ctrl para las clases controladoras.

2- Un solo espacio después de cada delimitador coma.

3- Un solo espacio alrededor de los operadores. Por ejemplo: ===, &&, ||.

[4] Es un estilo de escritura que se aplica a frases o palabras compuestas. Establece que la primera letra de cada una de las palabras es minúscula.

4- Llaves para indicar la estructura del cuerpo de control, independientemente del número de declaraciones que contenga.

3.5 Validación y pruebas.

Las pruebas se pueden realizar basándose en dos esquemas diferentes: demostrar a través de pruebas de caja blanca que las operaciones internas se ajustan a lo especificado y que los componentes internos marchan bien, o mediante las pruebas de caja negra, conociendo la función del programa e intentar demostrar que las funciones están correctas.

Para probar que el sistema cumple con los requerimientos específicos para los cuales ha sido creado se usó la prueba de funcionalidad que permitió validar las funciones y los métodos del caso de uso “Visualizar Despliegue”. Se aplicó además la prueba de carga sometiendo a la aplicación a un régimen de carga de trabajo para saber el rendimiento de la misma y la prueba de estrés se utilizó para saber el tiempo en el que la aplicación comienza a fallar o es incapaz de responder a las peticiones de los usuarios.

3.5.1 Niveles de prueba

Los niveles de pruebas se realizan en determinados momentos del ciclo de vida del software, en este flujo de pruebas se aplicó las pruebas de desarrollador para verifica el correcto funcionamiento del sistema a través de varias pruebas y las prueba de sistema en donde el sistema fue sometido a una serie de pruebas para verificar que todas las herramientas hayan sido integradas y que las funciones cumplan satisfactoriamente con los requisitos funcionales.

3.5.2 Tipos de pruebas

Las pruebas se pueden realizar basándose en dos esquemas diferentes: demostrar a través de pruebas de caja blanca que las operaciones internas se ajustan a lo especificado y que los componentes internos marchan bien, o mediante las pruebas de caja negra, conociendo la función del programa e intentar demostrar que las funciones están correctas.

Para probar que el sistema cumple con los requerimientos específicos para los cuales ha sido creado se usó la prueba de funcionalidad que permitió validar las funciones y los métodos del caso de uso “Visualizar Despliegue”. Se aplicó además la prueba de carga sometiendo a la aplicación a un régimen de carga de trabajo para saber el rendimiento de la misma y la prueba de estrés se utilizó para saber el tiempo en el que la aplicación comienza a fallar o es incapaz de responder a las peticiones de los usuarios.

3.5.2.1 Pruebas de rendimiento.

Las pruebas de rendimiento se basan en comprobar que el sistema puede soportar el volumen de carga definido en la especificación, es decir, hay que comprobar la eficiencia. Como la solución propuesta es una aplicación web que interactúa con un servidor, se realizan pruebas para comprobar la capacidad que tiene al aceptar varias peticiones concurrentemente.

3.5.2.2 Método caja blanca.

Entre los métodos de pruebas se decidió aplicar el método de caja blanca, el cual permitió comprobar y examinar la estructura interna del sistema utilizando la técnica de camino básico. Esta técnica posibilitó obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. (24)

Para aplicar la técnica de Camino básico se tuvo en cuenta lo siguiente:

Camino básico

La técnica del camino básico es la más usada en el método caja blanca, la cual determina la complejidad ciclomática de una porción de código. La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Cuando se usa el camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y da un límite superior para el número de pruebas que se deben realizar (24) Esta complejidad se puede calcular de tres formas:

- ✓ El número de regiones del grafo de flujo coincide con la complejidad ciclomática.
- ✓ La complejidad ciclomática, V(G), de un grafo de flujo G, se define como: V(G)=A-N+2. Donde A es el número de aristas del grafo de flujo y N es el número de nodos del mismo.
- ✓ La complejidad ciclomática, V(G), de un grafo de flujo G también se define como V(G)=P+1. Donde P es el número de nodos predicado, es decir son los nodos a partir de los cuales se puede tomar más de un camino dentro del contenido en el grafo de flujo G.

3.5.2.3 Método de caja negra

Para comprobar el correcto funcionamiento de las interfaces del software se aplicó el método de caja negra empleando la técnica de partición de equivalencia, la cual permitió examinar los valores válidos y no válidos de las entradas al sistema. (Pressman, 2002)

Son 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. Estas pruebas permiten encontrar:

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

Para aplicar la técnica de partición de equivalencia se tuvo en cuenta lo siguiente:

Partición de Equivalencia

La técnica de partición de equivalencia es la más usada en el método caja negra, la cual divide el dominio de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. El diseño de estos casos de prueba para la partición de equivalencia se basa en la evaluación de las clases de equivalencia. Una clase de equivalencia representa un conjunto de estados válidos o inválidos para condiciones de entrada.

3.5.2.4 Aplicación de la prueba de Rendimiento

Para la aplicación de la prueba de rendimiento, se tuvo en cuenta los factores que son más críticos en la solución propuesta: tiempo de respuesta del servidor y consumo de los navegadores web utilizados. A continuación se exponen los resultados que arrojaron los diferentes factores a evaluar.

Es de suma importancia destacar que es el tiempo de respuesta del servidor el factor de mayor significancia. En la siguiente tabla se muestra el promedio de tiempo de respuesta del servidor, utilizando 16 ordenadores, distribuidos en 15 clientes y 1 servidor con las siguientes características:

- **Sistema operativo:**
  - Debian, versión 7.8 (wheezy), 32 bits.
- **Hardware:**
  - Memoria RAM: 1.7 GB.
  - Procesador: Intel® Core™ i3-2120 CPU @ 3.30GHz × 4.

Tabla 5: Tiempo de respuesta del servidor.

Cantidad de clientes conectados	Tiempo promedio de respuesta del servidor(s)
5	1.189
10	1.225
15	1.387

Como se aprecia anteriormente, a medida que se adiciona un nuevo cliente aumenta el tiempo de respuesta del servidor a los clientes conectados.

En el caso de consumo de los navegadores web utilizados, se hizo un análisis minucioso del consumo de memoria empleado para la visualización de los despliegues y sumarios solicitados por los clientes. A continuación se representa el resultado obtenido durante la prueba.

Tabla 6: Consumo de memoria de los navegadores web.

Navegador web	Consumo de memoria(%) con respecto al ordenador
Firefox	63.4
Google Chrome	51.5

Los datos arrojados indican que es el navegador web Google Chrome posee el mejor rendimiento en cuanto la visualización de los despliegues pertenecientes al SCADA GALBA.

De acuerdo con los resultados obtenidos, las pruebas se realizaron satisfactoriamente obteniéndose valores satisfactorios en concordancia con el tiempo de actualización que debe poseer el visualizador web para la representación de los procesos pertenecientes al SCADA GALBA.

3.5.2.5 Aplicación de la prueba de caja blanca

A continuación se muestra como se le aplica la prueba de caja blanca a la función *login*() perteneciente al CU “Autenticar usuario”, la cual permite que el usuario pueda autenticarse en el sistema. El código fue dividido por bloques de ejecución, los cuales están enumerados y a su vez constituyen los nodos del camino básico.

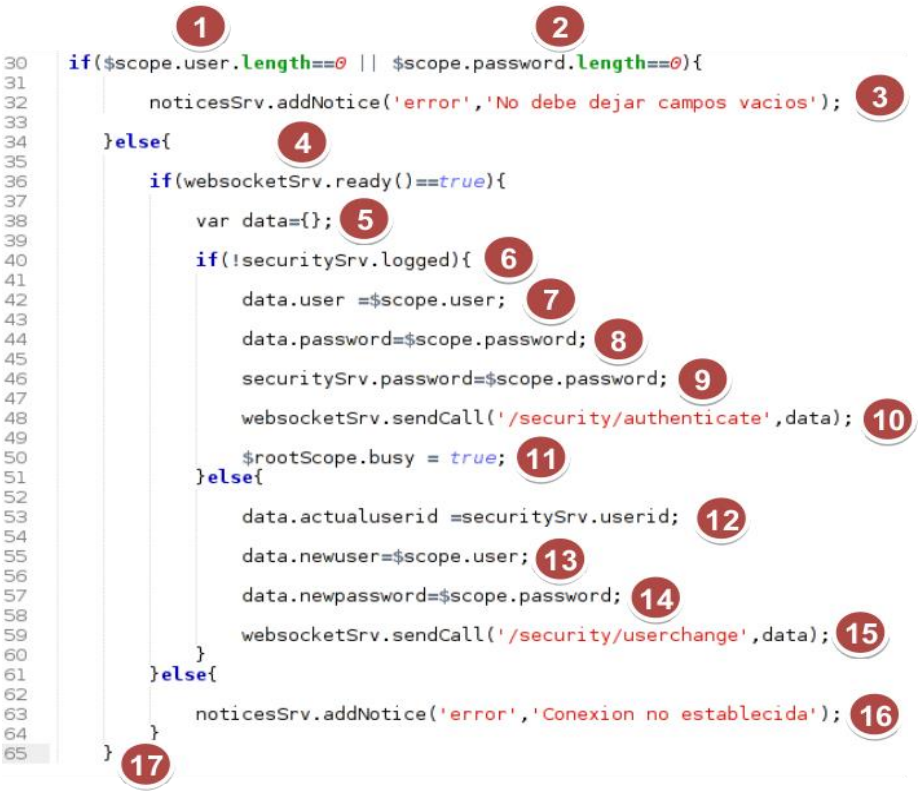


Figura 12: Fragmento de código perteneciente al caso de uso “Autenticar usuario”.

A continuación se representa el grafo de flujo necesario para calcular la complejidad ciclomática:

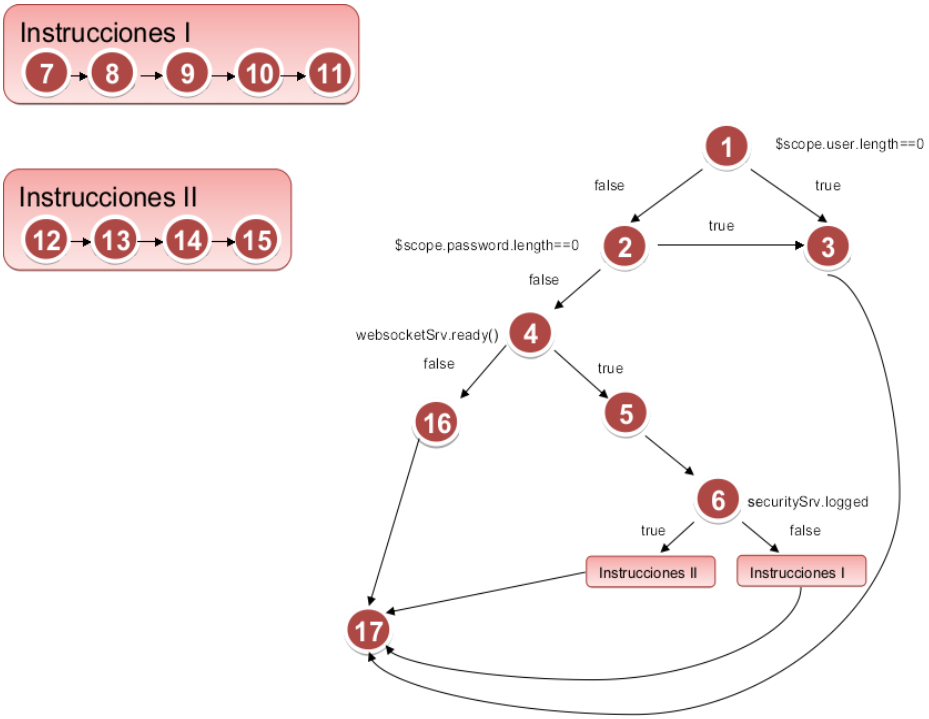


Figura 13: Grafo de flujo.

En la figura 11 se muestra el resultado que se obtuvo utilizando la técnica de camino básico. Las aristas indican los posibles caminos a seguir a partir del nodo correspondiente. Con el camino básico determinado, se aplica una de las tres formas para calcular la complejidad ciclomática, se utilizó la fórmula V(G) = A-N + 2, para la cual se obtuvo 19 artistas y 17 nodos, por lo tanto: V(G) = 19 - 17 + 2, quedando V(G) = 4. De la misma forma se pueden comprobar que las otras variantes explicadas de calcular la complejidad ciclomática arriban al mismo resultado. La complejidad ciclomática obtenida indica que existen 4 posibles casos de ejecución para la función, lo que es muy útil a la hora de diseñar los casos de prueba de caja negra.

3.5.2.6 Aplicación de la prueba de caja negra

Los casos de prueba que se presentan a continuación pretenden demostrar que las funciones del sistema son operativas, que la entrada de los valores válidos y no válidos se acepta de forma adecuada y que se produce un resultado correcto.

Condiciones de ejecución del Caso de Uso “Cambiar contraseña de usuario”.

1. El usuario debe de haberse autenticado en el visualizador web.

2. En el visualizador web se debe hacer clic en el icono asociado al cambio de contraseña.

Tabla 7: Caso de prueba de Caja negra del CU “Cambiar contraseña de usuario”.

Escenario	Variable	Descripción	Respuesta del sistema	Resultado de la prueba
1.1 Campos vacios.	contraseña_actual: string contraseña_nueva :string contraseña_confirmar:string	El usuario selecciona el icono que representa el cambio de contraseña. El sistema muestra una ventana para insertar los datos, el usuario al menos deja un campo vacío y da clic en el botón “Aceptar”.	Muestra el mensaje de error: “Existencia(s) de campo(s) vacio(s)”.	Correcto con respecto a la respuesta del sistema.
1.2 Contraseña actual válida.	contraseña_actual:string	El usuario selecciona el icono que representa el cambio de contraseña. El sistema muestra una ventana para insertar los datos, el usuario no inserta correctamente la contraseña actual y da clic en el botón “Aceptar”.	Muestra el mensaje de error: “Contraseña actual no válida”.	Correcto con respecto a la respuesta del sistema.
1.3 Nueva contraseña no confirmada.	contraseña_nueva :string contraseña_confirmar:string	El usuario selecciona el icono que representa el cambio de contraseña. El sistema muestra una ventana para insertar los datos, el usuario no confirma correctamente la nueva contraseña y da clic en el botón “Aceptar”.	Muestra el mensaje de error: “Nueva contraseña no confirmada”.	Correcto con respecto a la respuesta del sistema.

Condiciones de ejecución del Caso de Uso “Cambiar contraseña de usuario”.

1. El usuario debe de haberse autenticado en el visualizador web.

2. En el visualizador web se debe hacer clic en el icono asociado al sumario de estadísticas de subcanales.

3. El sistema debe haber recibido los datos correspondientes del servidor.

Tabla 8: Caso de prueba de Caja negra del CU “Visualizar sumario de estadísticas de subcanales”, sección 1 “Filtrar estadísticas de subcanales”.

Escenario	Variable	Descripción	Respuesta del sistema	Resultado de la prueba
No escoge ningún filtro	buscar: string	El usuario no selecciona ningún filtro, escribe en el campo de búsqueda lo que desea obtener y presiona la tecla Entrar.	Muestra el listado del resultado del filtro anterior.	Correcto con respecto a la respuesta del sistema.

Tabla 9: Caso de prueba de Caja negra del CU “Visualizar sumario de estadísticas de subcanales”, sección 2 “Realizar filtro personalizado”.

Escenario	Variable	Descripción	Respuesta del sistema	Resultado de la prueba
Palabra clave vacía.	criterio: string palabra clave :string	El usuario no inserta ningún valor como palabra clave.	El sistema muestra al mensaje: “Valor de filtro inválido.”	Correcto con respecto a la respuesta del sistema.
Filtro ya insertado.	criterio: string palabra clave :string	El usuario inserta un criterio que ya es un filtro.	El sistema muestra al mensaje: “Filtro ya existente.”	Correcto con respecto a la respuesta del sistema.

Tabla 10: Caso de prueba de caja negra del Caso de Uso “Visualizar sumario de estadísticas de subcanales”, sección 2 “Realizar filtro personalizado”.

Escenario	Variable	Descripción	Respuesta del sistema	Resultado de la prueba
-Filtro no seleccionado.	filtro: objeto	El usuario no selecciona el filtro que desea eliminar del panel de filtro y oprime el botón del panel (-).	El sistema muestra al mensaje: “No se ha seleccionado filtro a eliminar.”	Correcto con respecto a la respuesta del sistema.

3.5.2.7 Resultados de las pruebas

Después de realizar las pruebas funcionales mediante el método de caja blanca y caja negra, se comprobó el correcto funcionamiento de la interfaz y la codificación del sistema. Cada problema detectado en el desarrollo del sistema fue resuelto a raíz del trabajo continuo del desarrollador, con un total de 10 no conformidades encontradas en la primera iteración, 6 en la segunda y 4 en la tercera, las cuales se dividieron en significativas y no significativas. A continuación se representa lo expuesto anteriormente a través de la siguiente figura.

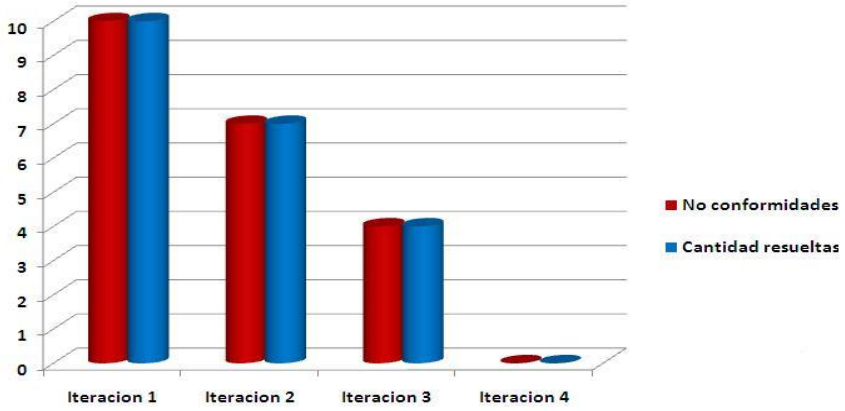


Figura 14: Resultados de las pruebas funcionales.

3.6 Conclusiones parciales

En el capítulo expuesto se construyó el modelo de implementación correspondiente al CU “Visualizar Despliegue”, para ello se realizó el diagrama de componentes en el que se definieron los componentes necesarios para el correcto funcionamiento de este CU y se describieron algunos de ellos para lograr una mayor comprensión acerca de su función. Además se definió el modelo de despliegue el cual muestra los elementos necesarios para realizar el despliegue e instalación del sistema. Una vez terminado el flujo de trabajo Implementación se pasó al flujo de trabajo de prueba, donde se aplicaron los métodos de Caja blanca empleando la técnica de camino básico y el método de Caja negra utilizando la técnica de partición de equivalencia. Cada dificultad detectada en el desarrollo del sistema fue resuelta, con un total de 10 no conformidades detectadas durante 3 iteraciones, que actualmente fueron satisfactoriamente resueltas.

Conclusiones Generales.

El desarrollo de una aplicación web que permita la disponibilidad del visualizador del HMI del SCADA Guardián del ALBA es un gran reto y una oportunidad que sin dudas posee un alto impacto social, ya que no existe otra solución similar en el CEDIN. El hecho además de que el sistema esté desarrollado con una arquitectura web, posibilita explotar los beneficios de la red, minimizando los costos de hardware en las organizaciones. Una vez concluida la presente investigación, se arriban a las siguientes conclusiones:

- La investigación del marco teórico de los visualizadores de procesos a partir de un SCADA orientado a la web permitió sentar las bases teóricas para la futura implementación de la solución propuesta.
- La combinación de los estilos y patrones de diseño definidos permitieron que la aplicación se implementara utilizando buenas prácticas de la programación y minimizando los errores de codificación.
- Las pruebas de rendimiento, caja blanca y caja negra realizadas sobre la solución propuesta permitieron corregir los riesgos funcionales detectados en el sistema.

Recomendaciones.

Luego de haber analizado los resultados del presente trabajo de diploma, se recomienda :

- Integrar la representación gráfica de la información mediante los gráficos de tendencia y gráfica XY.
- Integrar los componentes gráficos complejos pertenecientes al SCADA GALBA.
- Integrar la funcionalidad de control de los procesos pertenecientes al SCADA GALBA.

Bibliografía.

1. **Acedo Sánchez, José.** *Instrumentación y control básico de procesos.* Madrid : Díaz de Santos, 2006. pág. 509. ISBN 84-7978-759-77978-545-4.

2. **M. Romero, Ing. Diego.** Sistemas de Interfaz Humano-Máquina (HMI). *Sistemas de Interfaz Humano-Máquina (HMI).* Buenos Aires, Buenos Aires, Argentina : auto edición, 07 de 05 de 2007. Vol. I, 1.

3. **Frederico, Ulrich.** *Breve descripción del HMi 500.* Portugal : EFACEC Engenharia, S.A., 2010. pág. 2, Informativo.

4. **Inc., CSWorks.** CSWorks,Scalable automation solutions for the web. *CSWorks,Scalable automation solutions for the web.* [En línea] CSWorks Inc., 25 de 10 de 2010. [Citado el: 10 de 11 de 2014.] http://www.controlsystemworks.com/Product/Features/Web.aspx.

5. **Iconics.** Iconics, visualize your enterprise. *Iconics, visualize your enterprise.* [En línea] Iconics, 8 de 3 de 2011. [Citado el: 7 de 10 de 2014.] http://www.iconics.com/Home/Products/HMI-SCADA/GENESIS32.aspx.

6. **Universidad de las Ciencias Informáticas.** *Metodología de desarrollo para la actividad productiva de la UCI.* Universidad de las Ciencias Informáticas(UCI). La Habana : s.n., 2014. pág. 14.

7. **Osuna, Jessica Marianela.** *Metodologías Ágiles: Proceso Unificado Ágil.* México : s.n., 2014.

8. **Rumbaugh, James, Jacobson, Ivar y Booch, Graddy.** *El lenguaje Unificado de modelado.Manual de referencia.* 1. Estados Unidos : Addison-Wesley, 2005. pág.479. Vol. 1.

9. **Eguíluz Pérez, Javier.** *Introducción a XHTML.* México: Autoedición, 2009. pág.168.Vol. 1.

10. **Freddy Vega, John y Van Der Henst, Christian.** Aulaclic.com. *Aulaclic.com.* [En línea] 1, 12 de 08 de 2011. [Citado el: 15 de 11 de 2014.] http://www.aulaclic.es/guia-html5/. 1.

11. **Eguíluz Pérez, Javier.** *Introducción a CSS.* Acapulco:Autoedición, 2009.pág.251. Vol. 1.

12. **W3C.** W3C.Web design and applications. *W3C.Web design and applications.* [En línea] 1, 23 de 04 de 2013. [Citado el: 15 de 11 de 2014.] http://www.w3.org/standards/webdesign/graphics.

13. **Eguíluz Pérez, Javier.** *Introducción a AJAX.* s.l. : Autoedición, 2008. pág. 251. Vol. 1.

14. **Muñoz de la Torre Monzón, Ing. Arturo.** *Introducción a NodeJS a través de Koans.* Madrid : Autoedición, 2013. pág. 188. Vol. 1.

15. **Wang, Vanessa, Salim, Frank y Moskovits, Peter.** *The Definitive Guide to HTML5 Websocket.Build real-time applications with html5.* [ed.] Tony Pye. 1ra. New York City : Appress.Springer Sciences;Bussines Media, 2013. pág. 177. Vol.1.ISBN:978-1-4302-4740-1.

16. **Lubbers, Peter, Albers, Brian y Salim, Frank.** *Pro HTML5 Programming.Use html5 to create cutting-edge web applications.* [ed.] Tony Pye. 2da. New York City : Appress.Springer Sciences;Bussines Media, 2013. pág. 323. Vol. 1. ISBN:978-1-4302-3864-5.

17. **Saint-Andre, Peter, Smith, Kevin y Troncon, Remko.** *XMPP: The Definitive Guide.Building Real-Time Applications with Jabber Technologies.* [ed.] Mary E. Treseler. 1ra. Sebastopol : O'Reilly Media, 2009. pág. 306. Vol. 1. ISBN: 978-0-596-52126-4.

18. **Stefanov, Stoyan.***JavaScript Patterns.*[ed.] Mary Treseler.1ra. 1005 Gravenstein Highway North,Sebastopol :O' Reilly Media, 2010. pág. 206. Vol. 1.ISBN:978-0-596-80675-0.

19. **Tavárez, David.** Comparación de Frameworks en Javascript. *Comparación de Frameworks en Javascript.* [En línea] 26 de 05 de 2014. [Citado el: 15 de 11 de 2014.] http://www.maestrosdelweb.com/comparacion-frameworks-javascript/.

20. **Chafter, Jonathan y Swedberg, Karl.** *Learning Jquery.Better interaction,design, and web development with simple JavaScript techniques.* [ed.] Dayan Hyames, y otros. Cuarta edición. Birmingham : s.n., Junio 2013. pág. 404 páginas. ISBN 978-1-78216-314-5.

21. **Sánchez, Alex.** AngularJS.Framework Javascript para Webapps. *AngularJS.Framework Javascript para Webapps.* [En línea] 1, 04 de 03 de 2013. [Citado el: 17 de 11 de 2014.] http://www.lostiemposcambian.com/blog/javascript/angularjs-framework-javascript-para-webapps/.

22. **Visual Paradigm.** Visual Paradigm.UML CASE for software development. *Visual Paradigm.UML CASE for software development.* [En línea] Visual Paradigm, 03 de 09 de 2013. [Citado el: 20 de 11 de 2014.] www.visual-paradigm.com/product/vpuml.

23. **Martínez, Tania.** Brackets.io. Un editor de texto Open Source por Adobe. *Brackets.io. Un editor de texto Open Source por Adobe.* [En línea] 25 de 01 de 2014. [Citado el: 18 de 11 de 2014.] http://html5facil.com/tips/brackets-io-un-editor-de-texto-open-source-por-adobe/.

24. **S. Pressman, Roger.** *Ingeniería del Software.Un enfoque práctico.* [ed.] Darrel Ince. 5ta. s.l. : Mc Graw Hill, 2001. pág. 573. Vol. 1.

25. **Zuluaga Campuzano, Jorge Mario.** *Diseño de Sistema.* Universidad de Quindio, Facultad de Ciencias Humanas y Bellas Artes. Quindio : s.n., 2010. pág. 32.

26. **Harmes, Ross y Díaz, Dustin.** *Pro JavaScript.Design Patterns.* [ed.] Chris Mills y Tom Welsh. 1ra. New York City : Apress.Springer Bussines, 2008. Vol. 1. ISBN: 978-1-59059-908-2.

27. **Gauchat, Juan Diego.** *El gran libro de HTNL5 CSS3 y Javascript.* Primera edición. Barcelona : Marcombo, 2012. pág. 354. Vol. 1. ISBN 978-84-267-1782-5.

28. **Grigorik, Ilya.** *High Performance Browser Networking.* [ed.] Nash Courtney. Primera edición. Sebastopol : O'Reilly Media, 2013. Vol. I. ISBN 978-1-449-34476-4.

Referencias Bibliográficas.

1. **Acedo Sánchez, José.** *Instrumentación y control básico de procesos.* Madrid : Díaz de Santos, 2006. pág. 509. ISBN 84-7978-759-77978-545-4.

2. **M. Romero, Ing. Diego.** Sistemas de Interfaz Humano-Máquina (HMI). *Sistemas de Interfaz Humano-Máquina (HMI).* Buenos Aires, Buenos Aires, Argentina : auto edición, 07 de 05 de 2007. Vol. I, 1.

3. **Frederico, Ulrich.** *Breve descripción del HMi 500.* Portugal : EFACEC Engenharia, S.A., 2010. pág. 2, Informativo.

4. **Inc., CSWorks.** CSWorks,Scalable automation solutions for the web. *CSWorks,Scalable automation solutions for the web.* [En línea] CSWorks Inc., 25 de 10 de 2010. [Citado el: 10 de 11 de 2014.] http://www.controlsystemworks.com/Product/Features/Web.aspx.

5. **Iconics.** Iconics, visualize your enterprise. *Iconics, visualize your enterprise.* [En línea] Iconics, 8 de 3 de 2011. [Citado el: 7 de 10 de 2014.] http://www.iconics.com/Home/Products/HMI-SCADA/GENESIS32.aspx.

6. **Universidad de las Ciencias Informáticas.** *Metodología de desarrollo para la actividad productiva de la UCI.* Universidad de las Ciencias Informáticas(UCI). La Habana : s.n., 2014. pág. 14.

7. **Osuna, Jessica Marianela.** *Metodologías Ágiles: Proceso Unificado Ágil.* México : s.n., 2014.

8. **Rumbaugh, James, Jacobson, Ivar y Booch, Graddy.** *El lenguaje Unificado de modelado.Manual de referencia.* 1.Estados Unidos : Addison-Wesley, 2005. pág. 479. Vol. 1.

9. **Eguíluz Pérez, Javier.** *Introducción a XHTML.*México :Autoedición,2009. pág. 168. Vol. 1.

10. **Freddy Vega, John y Van Der Henst, Christian.** Aulaclic.com. *Aulaclic.com.* [En línea] 1, 12 de 08 de 2011. [Citado el: 15 de 11 de 2014.] http://www.aulaclic.es/guia-html5/. 1.

11. **Eguíluz Pérez, Javier.** *Introducción a CSS.* Acapulco : Autoedición,2009.pág.251.Vol. 1.

12. **W3C.** W3C.Web design and applications. *W3C.Web design and applications.* [En línea] 1, 23 de 04 de 2013. [Citado el: 15 de 11 de 2014.] http://www.w3.org/standards/webdesign/graphics.

13. **Eguíluz Pérez, Javier.** *Introducción a AJAX.* s.l. : Autoedición, 2008. pág. 251. Vol. 1.

14. **Muñoz de la Torre Monzón, Ing. Arturo.** *Introducción a NodeJS a través de Koans.* Madrid : Autoedición, 2013. pág. 188. Vol. 1.

15. **Wang, Vanessa, Salim, Frank y Moskovits, Peter.** *The Definitive Guide to HTML5 Websocket.Build real-time applications with html5.* [ed.] Tony Pye. 1ra. New York City : Appress.Springer Sciences;Bussines Media, 2013. pág. 177.Vol. 1.ISBN:978-1-4302-4740-1.

16. **Lubbers, Peter, Albers, Brian y Salim, Frank.** *Pro HTML5 Programming.Use html5 to create cutting-edge web applications.* [ed.] Tony Pye. 2da. New York City : Appress.Springer Sciences;Bussines Media, 2013. pág. 323. Vol. 1. ISBN:978-1-4302-3864-5.

17. **Saint-Andre, Peter, Smith, Kevin y Troncon, Remko.** *XMPP: The Definitive Guide.Building Real-Time Applications with Jabber Technologies.* [ed.] Mary E. Treseler. 1ra. Sebastopol : O'Reilly Media, 2009. pág. 306. Vol. 1. ISBN: 978-0-596-52126-4.

18. **Stefanov, Stoyan.** *JavaScript Patterns.* [ed.] Mary Treseler. 1ra.1005 Gravenstein Highway North,Sebastopol : O' Reilly Media,2010. pág. 206.Vol. 1.ISBN: 978-0-596-80675-0.

19. **Tavárez, David.** Comparación de Frameworks en Javascript. *Comparación de Frameworks en Javascript.* [En línea] 26 de 05 de 2014. [Citado el: 15 de 11 de 2014.] http://www.maestrosdelweb.com/comparacion-frameworks-javascript/.

20. **Chafter, Jonathan y Swedberg, Karl.** *Learning Jquery.Better interaction,design, and web development with simple JavaScript techniques.* [ed.] Dayan Hyames, y otros. Cuarta edición. Birmingham : s.n., Junio 2013. pág. 404 páginas. ISBN 978-1-78216-314-5.

21. **Sánchez, Alex.** AngularJS.Framework Javascript para Webapps. *AngularJS.Framework Javascript para Webapps.* [En línea] 1, 04 de 03 de 2013. [Citado el: 17 de 11 de 2014.] http://www.lostiemposcambian.com/blog/javascript/angularjs-framework-javascript-para-webapps/.

22. **Visual Paradigm.** Visual Paradigm.UML CASE for software development. *Visual Paradigm.UML CASE for software development.* [En línea] Visual Paradigm, 03 de 09 de 2013. [Citado el: 20 de 11 de 2014.] www.visual-paradigm.com/product/vpuml.

23. **Martínez, Tania.** Brackets.io. Un editor de texto Open Source por Adobe. *Brackets.io. Un editor de texto Open Source por Adobe.* [En línea] 25 de 01 de 2014. [Citado el: 18 de 11 de 2014.] http://html5facil.com/tips/brackets-io-un-editor-de-texto-open-source-por-adobe/.

24. **S. Pressman, Roger.** *Ingeniería del Software.Un enfoque práctico.* [ed.] Darrel Ince. 5ta. s.l. : Mc Graw Hill, 2001. pág. 573. Vol. 1.

25. **Zuluaga Campuzano, Jorge Mario.** *Diseño de Sistema.* Universidad de Quindio, Facultad de Ciencias Humanas y Bellas Artes. Quindio : s.n., 2010. pág. 32.

26. **Harmes, Ross y Díaz, Dustin.** *Pro JavaScript.Design Patterns.* [ed.] Chris Mills y Tom Welsh. 1ra. New York City : Apress.Springer Bussines, 2008. Vol. 1. ISBN: 978-1-59059-908-2.

27. **Gauchat, Juan Diego.** *El gran libro de HTNL5 CSS3 y Javascript.* Primera edición. Barcelona : Marcombo, 2012. pág. 354. Vol. 1. ISBN 978-84-267-1782-5.

28. **Grigorik, Ilya.** *High Performance Browser Networking.* [ed.] Nash Courtney. Primera edición. Sebastopol : O'Reilly Media, 2013. Vol. I. ISBN 978-1-449-34476-4.

Glosario de Términos.

CRT ("Tubo de rayos catódicos.")

Se llama genéricamente así a los monitores basados en un tubo de vacío donde un haz de electrones pinta la imagen en la pantalla.

CMMI-DEV

Modelo de Madurez de la Capacidad Integrado. Es un modelo que contienen las mejores prácticas que ayudan a las organizaciones a mejorar sus procesos.

Cross Domain

Mecanismo de seguridad de las comunicaciones en navegadores actuales. Evitan que un script (XMLHttpRequest de AJAX) o una aplicación (Flash, Silverlight) de una página web puedan acceder a un servidor web diferente del que residen.

DISPOSITIVO

Un dispositivo de campo es un sistema de transmisión de información (datos) que simplifica la instalación y operación de máquinas y equipamientos industriales utilizados en procesos de producción. El objetivo de un dispositivo de campo es sustituir las conexiones punto a punto entre los elementos de campo y el equipo de control. Típicamente son redes digitales, bidireccionales, multipunto, montadas sobre un bus serie, que conectan dispositivos de campo como PLCs/PACs, transductores, actuadores y sensores.

DRIVER

Es un programa que controla un dispositivo. Cada dispositivo, ya sea una impresora, un teclado, etc., debe tener un programa controlador.

JSON

Acrónimo de Notación de Objetos Javascript (*Javascript Object Notation*), es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.

HTTP

Protocolo usado para acceder a la Web (www). Se encarga de procesar y dar respuestas a las peticiones para visualizar una página web. Además sirve para el envío de información adicional como el envío de formularios con mensajes, etc.

PLC (“*Controlador Lógico Programable*”)

Equipo electrónico diseñado para programar y controlar procesos secuenciales en tiempo real. Por lo general, es posible encontrar este tipo de equipos en ambientes industriales.

Punto analógico

Se refiere a las direcciones de memoria que se configuran en el SCADA y que referencian a una variable capturada por un dispositivo de campo, o una variable del proceso. Se caracteriza por obtener la variable de una sola dirección de memoria de dispositivo.

Punto digital

Se refiere a las direcciones de memoria que se configuran en el SCADA y que referencian a una variable capturada por un dispositivo de campo, o una variable del proceso. Se caracteriza por poseer un rango 0-15 para los valores, cada uno es representado como estado, puede estar conformado por distintas direcciones de memorias de diferentes dispositivos.

RTU ("Unidades Terminales Remotas”)

Dispositivos de adquisición de datos y control en campo, cuya función principal es hacer la interfaz entre los equipos de instrumentación y control local y el sistema de adquisición de datos y control supervisorio.

SASL ("Capa de seguridad y autenticación simple")

Es un marco para proporcionar servicios de autenticación y seguridad de los datos en protocolos orientados a la conexión a través de mecanismos reemplazables. Ella proporciona una interfaz estructurada entre. protocolos y mecanismos. El marco resultante permite que los nuevos protocolos para la reutilización existente mecanismos y protocolos permite viejos para hacer uso de nuevos mecanismos. El marco también proporciona un protocolo para asegurar los posteriores intercambios de protocolo dentro de una capa de seguridad de datos.

SCADA

Acrónimo de *Supervisory Control And Data Acquisition* (Supervisión, Control y Adquisición de Datos) es un software para ordenadoresque permite controlar y supervisar procesos industriales a distancia. Facilita retroalimentación en tiempo real con los dispositivos de campo (sensores y actuadores), y controla el proceso automáticamente. Provee de toda la información que se genera en el proceso productivo (supervisión, control calidad, control de producción, almacenamiento de datos) y permite su gestión e intervención.

SUBCANAL

El subcanal de datos es un área donde se almacena información. Protecciones como *SecuROM* (*PC)* y *Libcrypt* (*PSX*) usan este canal. El canal principal es donde se almacenan el grueso de los datos (datos de usuario)

XML

Lenguaje de marcas extensible (*eXtensible Markup Language*) es un lenguaje de marcas desarrollado por el *World Wide Web Consortium*(W3C).

XMLHttpRequest (XHR)

Referido como XMLHTTP (*Extensible Markup Language* / *Hypertext Transfer Protocol*), es una interfaz empleada para realizar peticiones HTTP y HTTPS a servidores Web.

Anexo 1: Descripción de casos de uso.

Caso de uso "Administrar aplicación".

Objetivo	Iniciar la aplicación.
Actores	Usuario: "Inicia".
Resumen	Este caso de uso le permite al Usuario iniciar y detener el servicio del HMI Web, así como visualizar el monitor del sistema y mostrar la fecha y hora actual.
Complejidad	Media
Prioridad	Crítico.
Precondiciones	-
Postcondiciones	El Usuario inició la aplicación, visualizó la hora y/o se mostró el monitor del sistema.
Referencias	RF1 ,RF2,RF3,RF5
Flujo de eventos	
Flujo básico "Visualizar aplicación"	
Actor	**Sistema**
El actor siguiendo la ruta navegador Web -> SGVIEW.	El sistema muestra un mensaje indicando que se está conectando con el servidor.(ver alterno 1)
	3. Muestra la interfaz de la aplicación con el monitor del sistema, la hora y la fecha actual.
	4. Termina el Caso de Uso.
Flujo de eventos	
Flujo alterno "No pudo conectarse con el servidor"	
	3. Muestra el mensaje: "Conexión fallida, verifique su conexión con el servidor"

Caso de uso "Visualizar sumario de alarmas".

Objetivo	Operar en el sumario de alarmas del HMI-Web.
Actores	Usuario: "Inicia".
Resumen	Este caso de uso permite a los Usuarios visualizar procesos asociados a las alarmas.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	1.El Usuario está registrado en el sistema y pertenece a un grupo operacional de privilegios. Se ha generado el registro en el Sistema.
Postcondiciones	El Usuario pudo visualizar los datos correspondientes al sumario de alarmas.
Referencias	RF6,RF16,RF17
Flujo de eventos	
Flujo básico "Visualizar sumario de alarmas"	

Actor	**Sistema**
1.El caso de uso se inicia cuando el actor desde el HMI-Web, presiona clic en el icono que representa al sumario de alarmas.	2.Muestra una solapa con toda la información de las alarmas: Panel de búsqueda Botones para navegar por páginas Tabla de alarmas: -Fecha/Hora. -Recurso. -Causa. -Tipo.

	-Descripción. -No. Ocurrencias. -Prioridad. -Ayuda(representado con el signo de interrogación) -Grupo Para filtrar las alarmas: Ver sección 1 “Filtrar alarma”. Para paginar el listado de alarmas: Ver sección 2 “Paginar listado de alarmas”. Para cerrar sumario de alarmas: Ver sección 3 “Cerrar sumario de alarmas”.
Sección 1: “Filtrar alarma”	
Actor	**Sistema**
1. Posiciona el cursor en el campo editable buscar. Escribe los caracteres que considere necesarios para la búsqueda del recurso en el campo “Buscar” (ver alterno 1) y presiona el botón de actualización.	2. El sistema permite la edición de los campos: Buscar: Campo editable donde se introduce el valor del criterio. Criterio: Lista desplegable donde se muestran todos los criterios de búsquedas.
3. Escribe los caracteres que considere necesarios para la búsqueda del recurso en el campo “Buscar”	4. Va filtrando los recursos de acuerdo a los caracteres ingresados por el actor; el sistema muestra por defecto en el “Criterio” de búsqueda la primera propiedad de los recursos y realiza el filtrado de acuerdo a ese criterio o de acuerdo un criterio previamente seleccionado.
Flujo Alterno 1: “No escoge ningún filtro”.	

1. No selecciona ningún filtro, escribe en el campo de texto de búsqueda lo que desea obtener y presiona el botón de actualización.	2. Muestra el listado del resultado del filtro realizado con anterioridad.
Sección 2: "Paginar listado de alarmas"	
Actor	**Sistema**
1. Escoge la cantidad de elementos que desea observar por página en el campo de números de paginado que se encuentra al lado del campo de texto de filtros.	2. Muestra en el sumario la cantidad de elementos seleccionado por páginas. Retorna al paso 1.

Sección 3 "Cerrar sumario de alarmas"	
Actor	**Sistema**
1. Presiona clic en la pestaña correspondiente al sumario alarmas.	2. Cierra la pestaña sumario de alarmas. Finaliza el caso de uso.

Caso de uso "Visualizar sumario de de estadísticas de dispositivos".

Objetivo	Operar en el sumario de "Estadísticas de Dispositivos" desde el HMI web.
Actores	Usuario: "Inicia".
Resumen	Este caso de uso permite al Usuario visualizar las estadísticas de los dispositivos realizadas en el sistema.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El Usuario está registrado en el sistema y pertenece a un grupo operacional de privilegios. Se ha generado el registro en el Sistema.
Postcondiciones	Se mostraron los datos de la estadística de dispositivos; si el caso de uso no terminó correctamente se mostró el mensaje de error correspondiente. En ambos casos se ha generado el registro en el Sistema.
Referencias	RF12,RF16,RF17

Flujo de eventos	
Flujo básico "Visualizar sumario de estadísticas de dispositivos"	
Actor	**Sistema**
1.El caso de uso se inicia cuando el actor desde el HMI-Web, presiona clic en el icono que representa al sumario de estadísticas de dispositivos.	2. Muestra una solapa dividida por las columnas:Panel de búsqueda. Icono para ejecutar consulta a la base de datos. Icono para realizar filtro personalizado. Botones para navegar por páginas Tabla de dispositivos:

	-Id. -Recurso. -Descripción. -% Instantáneo. -Promedio Hora. Actual. -Promedio Hr. Anterior. -Promedio D. actual. -Promedio D. anterior. -Normal. -Error en ciclo de Scan. -Error CRC. -Error RX. -Error TX. -Error de configuración. -Reintentando Conexión. -Mensaje ilegal. -Error no definido. Para Filtrar estadísticas de dispositivo ": Ver sección 1 "Filtrar estadísticas de dispositivos". Para realizar filtro personalizado: Ver sección 2 "Realizar filtro personalizado". Para paginar el listado de estadísticas de dispositivos: Ver sección 3 "Paginar listado

	de estadísticas de dispositivos". Para cerrar sumario de estadísticas de dispositivos: Ver sección 4 "Cerrar sumario de estadísticas de dispositivos".
Sección 1 "Filtrar estadísticas de dispositivos"	
Actor	**Sistema**
1. Posiciona el cursor en el campo editable buscar. Escribe los caracteres que considere necesarios para la búsqueda del recurso en el campo (ver alterno 1) y se le da clic al botón de actualización.	2. El sistema permite la edición de los campos: Buscar: Campo editable donde se introduce el valor del criterio. Criterio: Lista desplegable donde se muestran todos los criterios de búsquedas.
3. Escribe los caracteres que considere necesarios para la búsqueda del recurso en el campo "Buscar"	4. Va filtrando los recursos de acuerdo a los caracteres ingresados por el actor; el sistema muestra por defecto en el "Criterio" un criterio no válido y realiza el filtrado de acuerdo a un criterio previamente seleccionado.
Flujo Alterno 1: "No escoge ningún filtro".	
5. No selecciona ningún filtro, escribe en el campo de texto de búsqueda lo que desea obtener y presiona el botón de actualización.	6. Muestra el listado del resultado del filtro realizado con anterioridad.
Sección 2 "Realizar filtro personalizado"	
Actor	**Sistema**
1. Presiona clic en el ícono que representa el filtro personalizado.	2. Despliega una ventana para la configuración de consulta, con los siguientes campos: -Criterio () -Operador de comparación (igual, mayor,

	menor, mayor que, menor que, no contiene, contiene) -Campo para colocar palabra clave -Panel de filtro -Botón de opción de búsqueda (Y/O) -Botón para agregar filtro al panel (+) - Botón para eliminar filtro del panel (-) Para adicionar un filtro ver sección 2.1 “Adicionar Filtro“. Para eliminar un filtro ver sección 2.2 “Eliminar Filtro“.
Sección 2.1 “Adicionar Filtro”	
Actor	**Sistema**
1. Presiona clic en Criterio.	2. Muestra una lista desplegable con los criterios de búsqueda: -Id. -Recurso. -Descripción. -% Instantáneo. -Promedio Hr. Actual. -Promedio Hr. Anterior. -Promedio D. actual. -Promedio D. anterior. -Normal. -Error en ciclo de Scan. -Error CRC.

	-Error RX. -Error TX. -Error de configuración. -Reintentando Conexión. -Mensaje llegal. -Error no definido.
3. Selecciona el criterio.	4. Permite seleccionar el criterio.
5. Escribe una palabra clave a buscar en el campo ubicado al lado del selector del operador de comparación.	6. Permite insertar un valor.
7. Presiona clic en Operador de comparación.	8. Muestra una lista despegable con los operadores de comparación: -Igual. -Mayor. -Menor. -Mayor que. -Menor que. -No contiene. -Contiene.
9. Selecciona el operador de comparación.	10. Permite seleccionar el operador de comparación.
11. Presiona clic en la opción de búsqueda.	12. Muestra una lista despegable con las opciones de búsqueda : -Y.

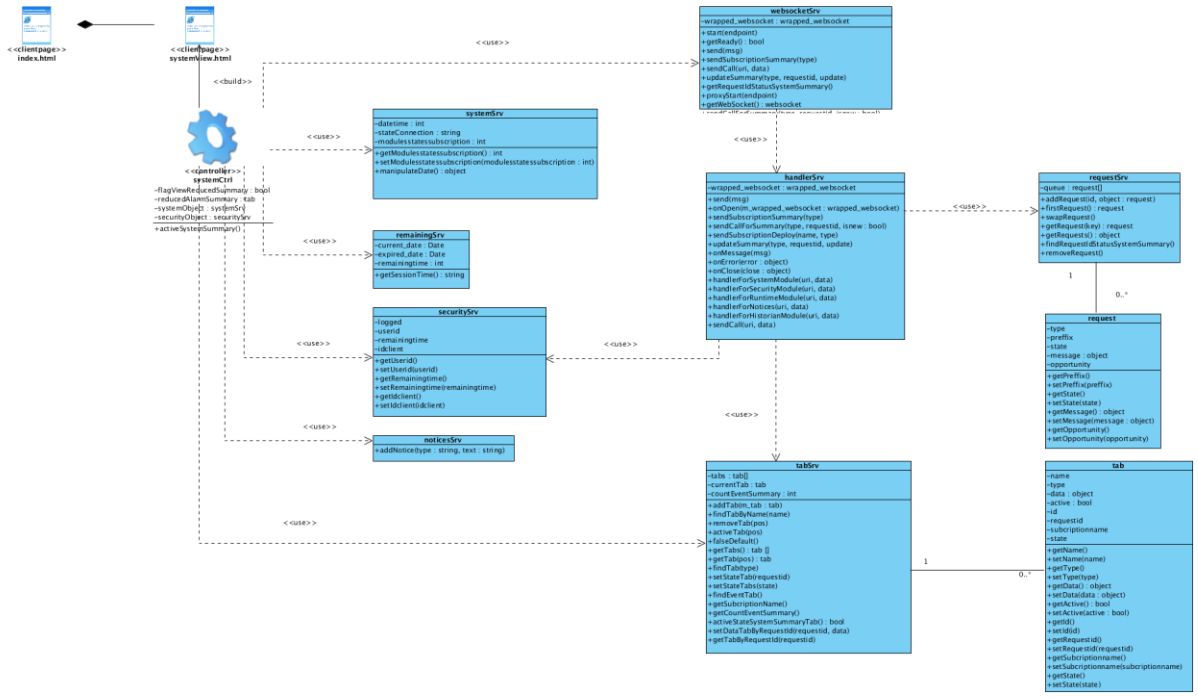
	-O.
13. Selecciona la opción de búsqueda.	14. Permite seleccionar la opción de búsqueda.
15. Oprime el botón para agregar filtro al panel (+)	16. Valida que el campo para colocar palabra clave no esté vacío (ver flujo alterno 1), que el filtro ya no esté insertado (ver flujo alterno 2), adiciona el filtro. Retorna a la sección 2.
Flujo Alterno 1: “Palabra clave vacía”	
	2.El sistema muestra al mensaje: “Valor de filtro inválido.” y vuelve a la sección 2.1.
Flujo Alterno 2: “Filtro ya insertado”	
	1.El sistema muestra al mensaje: “Filtro ya existente.” y vuelve a la sección 2.1.
Sección 2.2 “Eliminar Filtro”	
Actor	**Sistema**
El Usuario selecciona el filtro a eliminar del panel del filtro y presiona el botón del panel (-).	3.Valida que se haya seleccionado anteriormente el filtro (ver flujo alterno 1) a eliminar y elimina el filtro, confirmándolo con el mensaje “Filtro eliminado.”. Retorna a la sección 2.
Flujo Alterno 1: “Filtro no seleccionado”	
	1.El sistema muestra al mensaje: “No se ha seleccionado filtro a eliminar.” Retorna a la sección 2.2.
Sección 3: “Paginar listado de estadísticas de dispositivos”	
Actor	**Sistema**
1. Escoge la cantidad de elementos que desea observar por página en el campo de números de paginado que se encuentra al lado del campo de	2. Muestra en el sumario la cantidad de elementos seleccionado por páginas.

texto de filtros.	

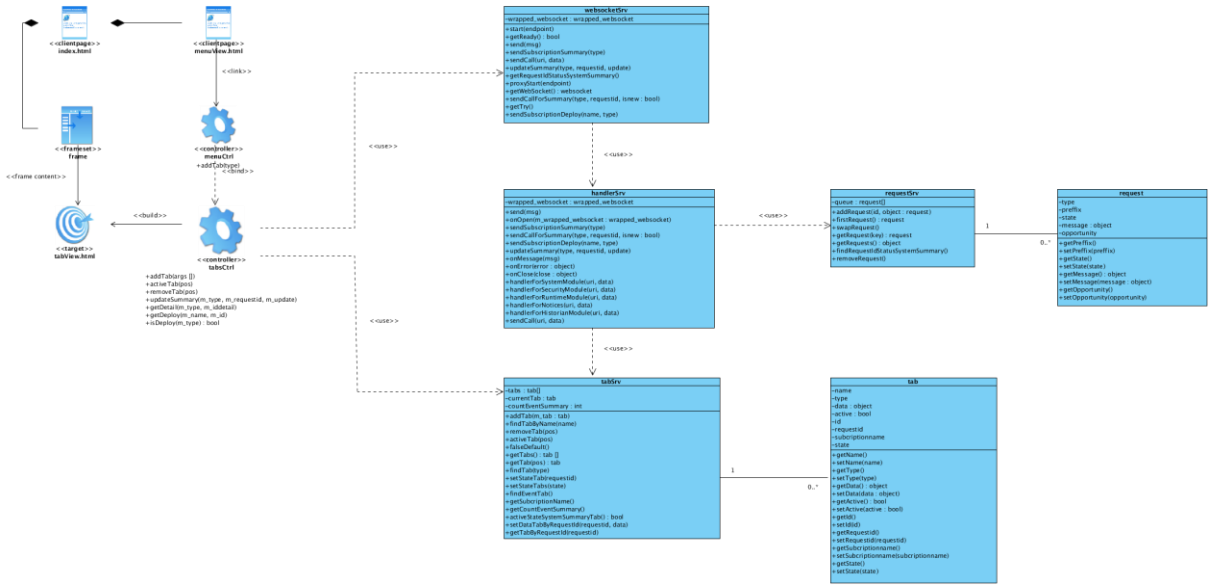
Sección 4: “Cerrar sumario de estadísticas de dispositivos”	
Actor	**Sistema**
1. Presiona clic en la pestaña correspondiente al sumario de estadísticas de dispositivos	2. Cierra la pestaña, se muestra la interfaz de la aplicación. Finaliza caso de uso.

Anexo 2: Modelo de diseño.

Modelo de diseño “Administrar aplicación”.



Modelo de diseño “Visualizar sumario de alarmas”.



Modelo de diseño “Visualizar sumario de de estadísticas de dispositivos”.