UNIVERSIDAD DE LAS CIENCIAS INFORMATICAS

Facultad 6

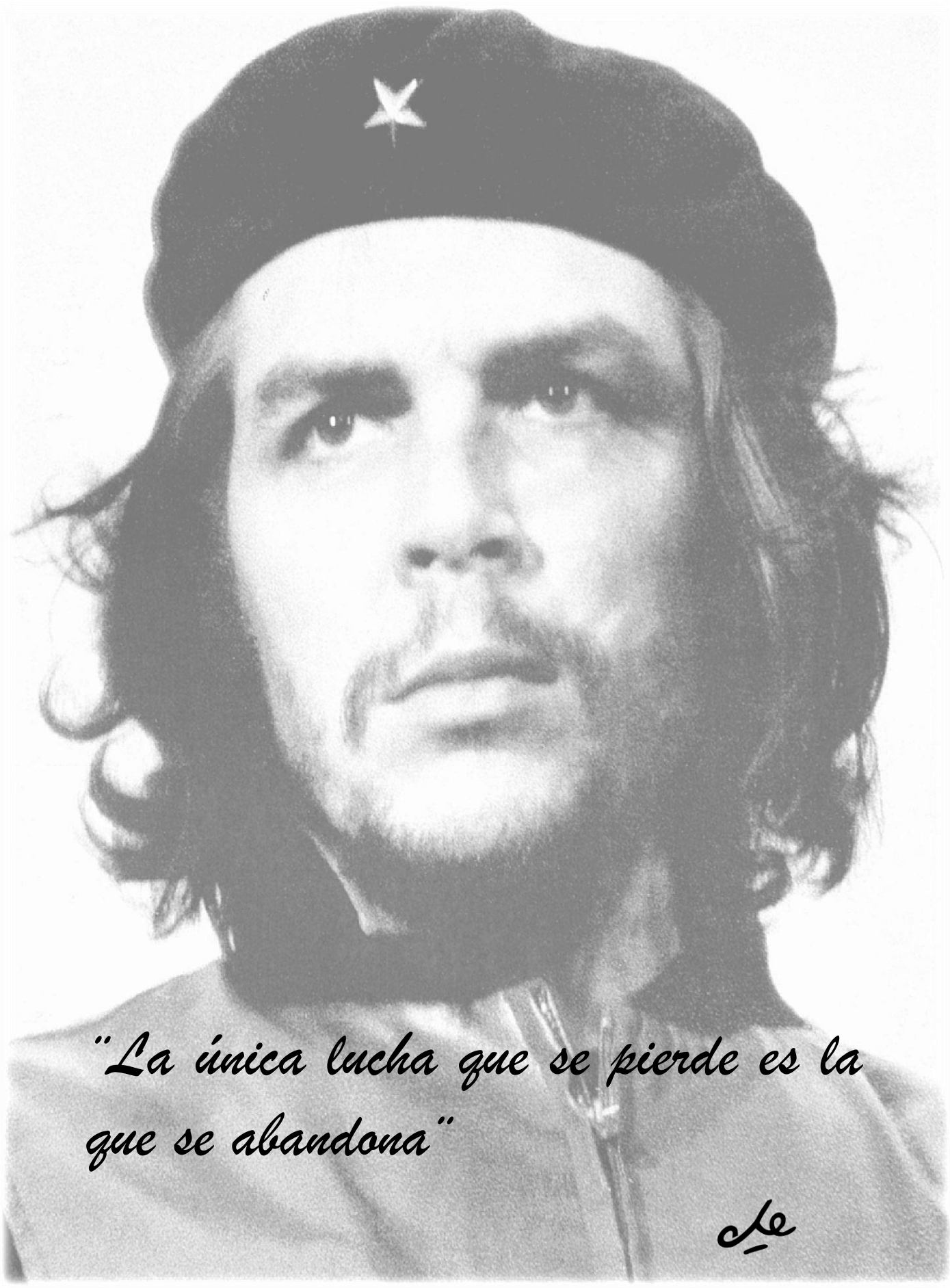
Trabajo de Diploma para optar por el título de Ingeniera en Ciencias Informáticas

Sistema para la visualización del consumo energético en la UCI

Autora: Yuleidys Portillo Caro

Tutores: MSc. Enrique Carreras Paz
MSc. Zobeida Rosa Pérez López-Chávez
C-tutor: Ing. Harold Williams Guerra Carrancá

La Habana, 2015
´´Año 57 de la Revolución´´


"La única lucha que se pierde es la que se abandona"

Declaración de autoría

Declaro ser autora de la presente tesis que tiene por título: Sistema para la visualización del consumo energético en la UCI y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de _____ del año 2015.

Yuleidys Portillo Caro

_____

Firma del autor

MSc. Zobeida Rosa Pérez López-Chávez

_____

Firma del tutor

MSc. Enrique Carreras Paz

_____

Firma del tutor

Ing. Harold Williams Guerra Carrancá

_____

Firma del tutor

Datos de Contacto

Autor

- Yuleidys Portillo Caro
- Universidad de las Ciencias Informáticas
- Correo electrónico: yuleidy@uci.cu

Tutores

- MSc. Enrique Carreras Paz, profesor Asistente
- Especialista A Centro de Informática Industrial (CEDIN)
- Teléfono Trabajo: 78373831
- Correo electrónico: enriquec@uci.cu

- MSc. Zobeida Rosa Pérez López-Chávez, profesor Asistente
- Jefe de Departamento de tecnología Facultad 5
- Teléfono Trabajo: 78372489
- Correo electrónico: zobeida@uci.cu

- Ing. Harold Williams Guerra Carrancá
- Administrador de redes y especialista en soporte y mantenimiento Laboratorios farmacéuticos AICA
- Teléfono Trabajo: 72712155-1012
- Correo electrónico: haroldg@aica.cu

Resumen

Para la economía del país la energía eléctrica es un eslabón primordial, por lo que es relevante contribuir al ahorro de la misma. La Universidad de la Ciencias Informáticas (UCI) es una de las entidades que más energía eléctrica consume a nivel nacional, pues posee una amplia red de alumbrado público y una cantidad considerable de equipos eléctricos. Debido a esto es de vital importancia el control del consumo. La entidad cuenta con un departamento de Atención a Programas Energéticos que se encarga de analizar el consumo de energía eléctrica recogido de los dispositivos instalados en las áreas de la universidad. Con estas mediciones es necesario elaborar informes históricos, mostrar mayores consumos, entre otros datos estadísticos. Todo este proceso se realiza con dificultades, partiendo desde la lectura a vista de los valores hasta los cálculos manuales, considerando además no poder tener estos datos actualizados en cortos períodos de tiempo. El presente trabajo describe una solución a dicha problemática con la implementación de un sistema que visualiza el consumo energético en la UCI y brinda reportes importantes asociados a este, a partir de su integración con un sistema supervisor SCADA que mide los parámetros eléctricos de los diferentes dispositivos de medición en cada área de la universidad.

Palabras claves: Consumo de energía, control energético, monitoreo.

Contenido

Índice de Tablas y Figuras

Introducción

Las Tecnologías de la Información y las Comunicaciones (TIC) ofrecen una oportunidad única para que los países en vías de desarrollo den un salto en su evolución económica, política y social. El logro de obtener un acceso fácil y rápido a la información correcta de manera oportuna, hacer más corta la distancia, minimizar los errores y optimizar el tiempo en la obtención de los datos, permite que la informática sea una de las herramientas más importante usada para la gestión de la información. (1).

Es significativo para toda organización, la planificación estratégica que incluya la gestión de la información, de manera que garantice el cumplimiento de la misión y la visión de la entidad, siendo esta la forma de visualizar los cambios que se desean y planificar las alternativas que se llevarán a cabo para la toma de decisiones.

En la actualidad, es común escuchar acerca de la importancia de la gestión de la información, a pesar de no ser un tema nuevo. Las estrategias existentes para esta gestión deben responder a los nuevos tipos de demandas, resultantes de la aparición de tendencias más modernas en las organizaciones. Este es un tema prioritario que ocupa, cada vez más, un espacio mayor en la economía de los países a escala mundial.

En Cuba, la gestión de la información relacionada con el tema de energía es un proceso que representa uno de los pilares fundamentales para el desarrollo nacional. Para la economía del país la energía es un eslabón primordial, por lo que es relevante contribuir al ahorro de la misma. Debido a esto se fundó el Programa de Ahorro de Electricidad en Cuba (PAEC), con el objetivo de reducir la máxima demanda del sistema y la tasa de crecimiento anual del consumo según los planes establecidos.

Una de las entidades que más energía eléctrica consume en el país es la Universidad de las Ciencias Informáticas (UCI). Esta cuenta con un número considerable de equipos consumidores de energía eléctrica y una amplia red de alumbrado público. El centro universitario tiene un departamento de Atención a Programas Energéticos encargado, entre otras funciones, de analizar el consumo de la energía eléctrica. El departamento recibe periódicamente un plan de consumo energético establecido por el Ministerio de Educación Superior, en el que está indicado la energía eléctrica que se debe consumir.

Con el objetivo de mejorar su trabajo, el departamento distribuyó el centro universitario por áreas, de esta manera realiza una distribución del plan entre cada una de ellas, dependiendo de un estudio basado en los datos que se obtienen de los dispositivos medidores.

Para obtener, visualizar y comprobar los datos del consumo energético la dirección del departamento trabaja con un Sistema de Supervisión que mide los parámetros eléctricos de los diferentes dispositivos de medición en cada área.

En estos momentos los cálculos del consumo energético de la universidad se hacen manualmente una vez al día, así como las lecturas de los dispositivos eléctricos. Esto provoca que algo de mínima complejidad pueda demorar mucho y que además se introduzcan errores, propios del ser humano. Además, no se tiene un estado actualizado en cada instante de tiempo de la información recogida, pues los indicadores solo están disponibles en los dispositivos de medición ubicados en un área específica de la entidad, distanciados del local donde radican los especialistas encargados del análisis y reporte. Debido a esto, no es posible precisar en todo momento cuáles son las áreas que cumplen realmente con el plan trazado, si el plan general se está cumpliendo o no, el horario en que sobrepasan el plan y qué área lo hace y si está correctamente ajustado el consumo asignado a cada área respecto a sus necesidades. Al no poder monitorizar todas las variables de consumo es imposible tomar decisiones acertadas en el momento oportuno para prevenir picos negativos o despilfarro de energía eléctrica.

Por lo antes planteado la dirección del departamento de Atención a Programas Energéticos, no puede tomar decisiones ante el despilfarro de energía eléctrica en la UCI en cualquier instante del día; como por ejemplo: medidas diferenciadas de distribución de consumo energético para las distintas áreas de la universidad.

Teniendo en cuenta la situación problemática anteriormente descrita, se plantea el siguiente ***problema a resolver***: ¿Cómo contribuir a la toma de decisiones ante el despilfarro de energía eléctrica en la UCI, en cualquier instante del día?

Este problema se enmarca dentro del ***objeto de estudio***: la medición de variables eléctricas en redes de distribución, teniendo como ***campo de acción***: la medición de variables eléctricas para el análisis del consumo energético en la UCI.

Se determina como ***objetivo general*** de la investigación: Desarrollar un sistema informático que permita visualizar el consumo energético en la UCI, contribuyendo al análisis de este en cualquier instante del día, para la correcta toma de decisiones ante el despilfarro de energía eléctrica.

Al analizar la posible solución al problema, se procede a plantear la siguiente ***hipótesis***: Si se desarrolla un sistema informático que visualice la información del consumo energético de la UCI en cualquier instante del día, se disminuirá el tiempo y los errores asociados al procesamiento de la información, permitiendo la correcta toma de decisiones ante el despilfarro de energía eléctrica.

Para dar cumplimiento al objetivo de la investigación se plantean las siguientes ***tareas de la investigación:***

1. Elaboración del marco teórico-referencial asociado a la medición de variables eléctricas en redes de distribución y a su empleo en reportes de consumo de energía eléctrica.
2. Selección de herramientas, tecnologías y metodologías de desarrollo, afines a la elaboración del sistema informático para la visualización del consumo energético en la UCI.
3. Desarrollo del sistema informático.
4. Implementación del sistema informático.
5. Aplicación de pruebas para la validación del funcionamiento del sistema informático.

Para la investigación se emplean los siguientes Métodos Teóricos:

Analítico-Sintético: Este método permite sintetizar la información necesaria para el desarrollo de la aplicación Web de gestión de la información del consumo energético. Se utiliza al estudiar la documentación necesaria referente a la medición y consumo de energía, permitiendo la extracción de los elementos más importantes que se relacionan con el objeto de estudio.

Histórico-Lógico: Este método es utilizado en la consulta de bibliografía referente al consumo energético y la automatización. También permite estructurar la documentación investigada de manera organizada, para así tener un mejor entendimiento de la misma.

Para la investigación se emplean los siguientes Métodos Empíricos:

Observación científica: Mediante la observación se comprobó la necesidad de la investigación a partir de la información obtenida referente al despilfarro de energía eléctrica en la UCI. (Anexo 1)

En el proceso de investigación se aplican técnicas como:

Entrevista: Esta técnica es utilizada para recopilar información referente a la energía eléctrica y al consumo energético. Esta entrevista fue dirigida a un especialista del grupo de supervisión de energía eléctrica de la universidad. (Anexo 2)

Estructura del documento

El presente trabajo de diploma está conformado por: Resumen, Introducción, tres capítulos, Conclusiones, Recomendaciones, Referencias bibliográficas y Anexos.

En el ***Capítulo 1*** (Fundamento teórico), se realiza una caracterización de los principales conceptos asociados al dominio del problema. Se hace un análisis del estado del arte que precede a la realización de este trabajo, en el cual se definen temas relacionados con el desarrollo de aplicaciones Web para la gestión de la información asociada al consumo energético a nivel mundial y se selecciona un conjunto de tecnologías y herramientas para el desarrollo de la solución propuesta.

En el ***Capítulo 2*** (Características y diseño de la aplicación Web), se describen las características de la aplicación Web a desarrollar, definiendo el dominio y los requisitos de la misma, también se describen los patrones empleados para estructurar la aplicación y se diseñan los diagramas generados por la metodología seleccionada.

En el ***Capítulo 3*** (Implementación y pruebas de la aplicación Web), se realiza la descripción de la implementación y las mediciones para validar el patrón de calidad, se efectúan las pruebas al sistema para evaluar su funcionamiento mostrando sus resultados.

Capítulo 1 Fundamentos teóricos de la investigación

1.1- Introducción

En este capítulo se abordan conceptos básicos relacionados con el objeto de estudio, las variables eléctricas utilizadas para el cálculo del consumo energético y elementos del proceso en sí. Se analizan algunas soluciones existentes utilizadas para la gestión de la información del consumo energético en Cuba y el mundo. Además, se seleccionan las tecnologías y herramientas para el desarrollo de la aplicación propuesta, a partir de las que comúnmente se emplean en este tipo de herramientas informáticas.

1.2- Conceptos asociados al Proceso de Gestión de Consumo Eléctrico

En este epígrafe se analizan los conceptos asociados al proceso de gestión del consumo eléctrico, así como las fórmulas que se utilizan para calcular las variables eléctricas que son relevantes para el desarrollo de la investigación.

1.2-1. Energía eléctrica

Se denomina energía eléctrica a la forma de energía que resulta de la existencia de una diferencia de potencial entre dos puntos. La energía eléctrica se manifiesta como corriente eléctrica (movimiento de cargas eléctricas negativas, o electrones, a través de un cable conductor metálico como consecuencia de la diferencia de potencial que un generador esté aplicando en sus extremos). (2)

1.2.1 Potencia eléctrica

La potencia eléctrica es la relación de paso de energía de un flujo por unidad de tiempo; es decir, la cantidad de energía entregada o absorbida por un elemento en un tiempo determinado (2). La unidad para la medición de la potencia según el Sistema Internacional de Unidades es el vatio (*watt*) y la energía consumida por un dispositivo eléctrico se mide en vatios-hora (*Wh*) (2). Normalmente las empresas que suministran energía eléctrica a la industria y los hogares, en lugar de facturar el consumo en vatios-hora, lo hacen en kilovatios-hora (*kWh*).

Existen dos tipos de potencia según del tipo de corriente que circula:

Potencia en corriente continua: Cuando se trata de corriente continua (*CC*) la potencia eléctrica desarrollada en un cierto instante por un dispositivo de dos terminales, es el producto de la diferencia de potencial entre dichos terminales y la intensidad de corriente que pasa a través del dispositivo. Por esta razón la potencia es proporcional a la corriente y a la tensión. Esto es:

$$P = \frac{dw}{dt} = \frac{dw}{dt} \cdot \frac{dq}{dt} = I \cdot V$$

Donde *I* es el valor instantáneo de la intensidad de corriente y *V* es el valor instantáneo del voltaje. Si *I* se expresa en amperios y *V* en voltios, *P* estará expresada en watts (*vatios*). Igual definición se aplica cuando se consideran valores promedio para *I, V* y *P*. (3)

Potencia en corriente alterna: Cuando se trata de corriente alterna (*AC*) sinusoidal, el promedio de potencia eléctrica desarrollada por un dispositivo de dos terminales es una función de los valores eficaces o valores cuadráticos medios, de la diferencia de potencial entre los terminales y de la intensidad de corriente que pasa a través del dispositivo. Si a un circuito se aplica una tensión sinusoidal *v (t)* con velocidad angular ω y valor de pico V_0 de forma:

$$v\,(t) = V_0\,.\,\mathrm{sin}(\omega t)$$

Esto provocará, en el caso de un circuito de carácter inductivo (caso más común), una corriente *I* desfasada un ángulo θ respecto de la tensión aplicada:

$$i\,(t)\mathrm{I}_0\,.\,\mathrm{sin}(\omega t - \theta)$$

Donde, para el caso puramente resistivo, se puede tomar el ángulo de desfase como cero. La potencia instantánea vendrá dada como el producto de las expresiones anteriores:

$$p\,(t) = V_0\,.\mathrm{I}_0\,.\,\mathrm{sin}(\omega t) \cdot \mathrm{sin}(\omega t - \theta)$$

Mediante trigonometría y sustituyendo los valores del pico por los eficaces:

$$p\,(t) = V.\mathrm{I}\,.\,\mathrm{cos}(\theta) - V \cdot I\ \mathrm{cos}(2\omega t - \theta)$$

Se obtiene así para la potencia un valor constante $V \cdot I\cos(\theta)$ que se denomina potencia activa y otro variable en el tiempo $V \cdot I\cos(2\omega t - \theta)$que se denomina potencia fluctuante. (3)

1.2.2 Consumo de energía eléctrica

Cuando los equipos eléctricos están funcionando generan un consumo de energía eléctrica en función de la potencia que tengan y del tiempo que estén en funcionamiento. En Cuba, el consumo de energía eléctrica en los hogares se contabiliza mediante un dispositivo precintado que se instala en los accesos a la vivienda, denominado contador, y que cada mes revisa un empleado de la compañía suministradora de la electricidad anotando el consumo realizado en ese período. En el caso de la industria el consumo se contabiliza mediante un metrocontador inteligente, donde se definen diferentes variables asociadas al consumo, encontrándose entre estas: la energía activa, la energía aparente, la lectura pico y la lectura del día.

El kilovatio hora (*kWh*) es la unidad de energía en la que se factura normalmente el consumo doméstico o industrial de electricidad. Equivale a la energía consumida por un equipo eléctrico cuya potencia fuese un kilovatio (*kW*) y estuviese funcionando durante una hora. La cantidad de energía consumida o entregada se obtiene multiplicando la potencia del equipo (*Potencia Activa*) por el tiempo durante el cual trabaja el mismo. (4)

Por lo que se puede definir el consumo eléctrico o energía consumida por:

*Consumo eléctrico = Potencia Eléctrica (W) * tiempo de uso en horas (h)*

Este resultado se expresa en *kW/h*.

En esta investigación, se define el consumo como la medición realizada cada un espacio de tiempo de 1 minuto, se puede afirmar que el consumo eléctrico de los sistemas monofásicos y trifásicos medidos en la universidad sería:

Consumo eléctrico monofásico = $PotenciaActiva = I \cdot V \cdot fp$

Consumo eléctrico trifásico = $PotenciaActiva = \sqrt{3} \cdot I \cdot V \cdot fp$

Donde la *I* viene dada en Amperes (*A*) y el voltaje en Volt (*V*).

1.2.3 Potencia aparente

La potencia aparente de un circuito eléctrico de corriente alterna (*S*), es la suma vectorial de la potencia que el mismo disipa y se transforma en calor o trabajo (*P*) expresado en vatios (*W*) y la potencia utilizada para la formación de los campos electromagnéticos de sus componentes, (se identifica con la letra *Q* y se mide en voltamperios reactivos). Esto significa que la potencia aparente representa la potencia total desarrollada en un circuito. (5). La relación entre todas las potencias aludidas es:

$$S^2 = P^2 \cdot Q^2$$

La fórmula de la potencia aparente es: (3)

$$S = I \cdot V$$

1.2.4 Potencia trifásica

En ingeniería eléctrica un sistema trifásico es un sistema de producción, distribución y consumo de energía eléctrica, formado por tres corrientes alternas monofásicas de igual frecuencia y amplitud (y por consiguiente, valor eficaz) que presentan una cierta diferencia de fase entre ellas, en torno a 120°, y están dadas en un orden determinado. Cada una de las corrientes monofásicas que forman el sistema se designa con el nombre de fase (3). El sistema trifásico presenta una serie de ventajas como:

- La economía de sus líneas de transporte de energía (hilos más finos que en una línea monofásica equivalente) y de los transformadores utilizados.
- Un elevado rendimiento de los receptores, especialmente motores, a los que la línea trifásica alimenta con potencia constante y no pulsada, como en el caso de la línea monofásica.

La representación matemática de la potencia activa en un sistema trifásico equilibrado (las tres tensiones de fase tienen idéntico valor y las tres intensidades de fase también coinciden) está dada por la ecuación:

$$P_{3\varphi} = \sqrt{3} \cdot I \cdot V \cos(\theta)$$

Siendo *I* la intensidad de línea y *V* la tensión de línea (no deben emplearse para esta ecuación los valores de fase). Para reactiva y aparente:

$$Q_{3\varphi} = \sqrt{3} \cdot I \cdot V \sin(\theta)$$

$$S_{3\varphi} = \sqrt{3} \cdot I \cdot V$$

1.2.5 Factor de potencia

El factor de potencia es un indicador del correcto aprovechamiento de la energía eléctrica. Se define como la relación entre la potencia activa (*kW*) usada en un sistema y la potencia aparente (*kVA*) que se obtiene de las líneas de alimentación.

Todos los equipos electromecánicos que están constituidos por devanados o bobinas, tales como motores y transformadores necesitan la denominada corriente reactiva para establecer campos magnéticos necesarios para su operación. La corriente reactiva produce un desfase entre la onda de tensión y la onda de corriente, si no existiera la corriente reactiva la tensión y la corriente estarían en fase y el factor de potencia seria la unidad.

El desfase entre las ondas de tensión y corriente, producido por la corriente reactiva se anula con el uso de condensadores de potencia, lo que hace que el funcionamiento del sistema sea más eficaz y por lo tanto, requiera menos corriente lo que técnicamente se denomina compensación. (6)

Si se define como factor de potencia (*fp)* tendríamos que el cálculo de la potencia activa para los circuitos monofásicos y trifásicos sería:

Potencia activa circuitos monofásicos: $P = I \cdot V \cos(\theta)$

Potencia activa circuitos trifásicos: $P_{3\varphi} = \sqrt{3} \cdot I \cdot V \cos(\theta)$

1.3 Dispositivos de medición

Para el análisis de las variables eléctricas se necesitan de algunos dispositivos eléctricos que realicen su medición. La mayoría de las empresas que fabrican estos dispositivos proveen un gran número de productos orientados a la supervisión de las instalaciones eléctricas tales como: instrumentación analógica, instrumentación digital, contadores de

energía de uso interno, analizadores de redes eléctricas, transformadores de corriente, analizadores portátiles y software de supervisión y control.

Existen en el mundo diversidad de dispositivos de medición de variables eléctrica, entre ellos podemos mencionar:

- Multímetros
- Vatihorímetro
- Metrocontadores
- Analizadores

En la universidad existen instalados varios dispositivos eléctricos encargados de recoger información de las variables eléctricas, uno de ellos es el contador trifásico digital multifunción CIRWATT C que es de 2 o 4 cuadrantes, con precisión 1 en activa y 2 en reactiva, medida directa o indirecta, con la posibilidad de programar hasta 3 contratos.

Atendiendo a las demandas y necesidades del mercado eléctrico, CIRCUTOR disidió incluir 2 puertos de comunicaciones, respetando los protocolos de comunicación IEC y MODBUS, y apuesta a los contadores tipo C al sistema de comunicaciones, PLC.

En la pantalla principal de reposo aparecen los indicadores visuales:

- Sentido de la energía
- Reactiva capacitiva / reactiva
- Cuadrante en el que está trabajando
- Las fases activadas y su sentido
- Tarifa activa por contrato

El contador también dispone de 3 alarmas que indican el estado del contador o de la instalación crítica, no crítica y de batería. Este dispositivo eléctrico CIRWATT C (7) es idóneo para suministros de BT y de MT hasta los 450 *kW*.

Otro de los dispositivos eléctrico que se encuentran instalados en la entidad son los analizadores de redes de potencia WM14 DIN y WM14 96 en su versión básica y avanzada pueden utilizarse en aquellas aplicaciones donde se necesite medir y controlar los principales parámetros eléctricos y transmitirlos a través de pulsos o señales digitales

a un PLC o un PC. El equipo WM14 es un modelo compacto disponible para montaje en panel o carril DIN (8).

El WM14 es la evolución natural del WM12. Además de las ventajas y funciones de medida proporcionadas por el indicador multifunción algunas de las ventajas son:

- La medida de energía activa y reactiva total y parcial con salidas de pulso, para supervisar no sólo los parámetros de carga típicos sino también los consumos.
- Medida de la corriente térmica, a través de una fase, y registro de las demandas máximas. Con esta información el personal de mantenimiento podrá saber si las protecciones contra las sobre intensidades han sido correctamente prefijadas y en el caso de producirse un disparo, podrán saber cuál es su intensidad real.
- Función de contador horario. Incorporado a una máquina o grupo generador, el instrumento indica cuánto tiempo ha estado funcionando una máquina, ahorrando el costo del clásico contador horario externo.
- Permite estimar y planificar el coste correcto del “uso de máquinas” y/o del mantenimiento mecánico.
- Control OR/AND de hasta 16 variables seleccionadas para proporcionar una carga o un control de línea más completo a través de 2 salidas digitales (8).

1.4 Estudio de soluciones existentes

En este epígrafe se abordan las soluciones informáticas existentes a nivel internacional y nacional; concluyendo en la importancia del desarrollo de un sistema informático que dé solución a la situación de la gestión energética en la UCI.

1.4.1 Aplicaciones para la gestión energética a nivel internacional

Con el desarrollo de las nuevas tecnologías han surgido nuevas formas de abordar la problemática de la gestión de la información energética. Cada día se desarrollan equipos modernos capaces de controlar un número mayor de variables eléctricas, optimizando el control de la calidad y el consumo de energía. Para lograr un óptimo aprovechamiento de las funcionalidades que poseen estos equipos se hace necesaria la puesta en marcha de productos de software que exploten estas funcionalidades y gestionen la información de una manera eficiente.

Power Studio

Es un sistema privativo para la gestión energética, que fue creado por la empresa CIRCUTOR. Este software tiene como principal función, la comunicación con los equipos CIRCUTOR y la elaboración posteriori de tablas y gráficas de los históricos registrados. A continuación se muestran algunas de las principales ventajas del mismo (9)

- Módulo de alarmas.
- Visualización de parámetros en tiempo real.
- Software Multipuesto (Servidor Web) mediante pantallas estáticas.
- Gran versatilidad y muy fácil uso.
- Acceso a través de Internet con contraseña y posibilidad de creación de perfiles de acceso.

Power Monitoring Expert

Es un sistema muy completo desarrollado por la empresa *Schneider Electric*, que permite el monitoreo del consumo y análisis de variables eléctricas, además de agua, gas y otros. Permite organizar y asignar costos por locales y/o zonas de la entidad, identificación de pérdidas, reportes especializados y validaciones de la calidad de la energía y redes de distribución. Permite analizar picos de demanda y mejorar la asignación de la energía según la demanda de un área en cuestión. Ayuda en la comparación de consumos entre diferentes períodos de tiempo (10).

1.4.2 Aplicaciones para la gestión energética a nivel nacional

En Cuba se han desarrollado diversas aplicaciones con el objetivo almacenar las variables eléctricas haciendo uso de diferentes equipos de medición (metro contadores y analizadores) para la supervisión de la calidad y control de la energía eléctrica.

En la Empresa Eléctrica de Villa Clara surgió un proyecto que haciendo uso de los metro contadores digitales programables (ABB A1700, DTSD 341 y CIRCUTOR) se puso a prueba en la Facultad de Ingeniería Eléctrica de la Universidad Central de Las Villas “Marta Abreu”. Como parte de este proyecto se implementaron dos aplicaciones, un software que lee y almacena en una base de datos los valores de las variables

eléctricas que ofrecen estos equipos y una aplicación para la visualización y supervisión de las mismas. Además se desarrolló una aplicación Web en la que se ofrecían algunas de las funcionalidades que posee el Sistema de Supervisión antes mencionado.

En el año 2005 la Empresa Eléctrica de la provincia de Cienfuegos desarrolló un sistema similar, la cual permite la lectura de los metro contadores DTSD 341 Shangsha, este sistema está implementado en el lenguaje de programación Delphi y cuenta solo, para la supervisión de los parámetros de lectura, con el software de los metro contadores, no aprovechando así todas sus funcionalidades. En el Instituto Superior Politécnico "José Antonio Echeverría" (CUJAE) se realizó una aplicación que presta funcionalidades con el mismo propósito, a pesar de lo anterior, la misma está desarrollada para la lectura sobre los analizadores programables digitales y no cuenta con una aplicación capaz de monitorizar las variables (9).

1.4.3 Aplicación para la gestión energética en la UCI

En la UCI existe un departamento de Atención a Programas Energéticos encargado de supervisar, analizar el consumo de la energía eléctrica y distribuir el plan de consumo de la universidad entre las áreas, con las cuales cuenta. El departamento para la lectura de los dispositivos eléctricos disponibles en la red cuenta con un Sistema de Supervisión y Control (SCADA) desarrollado por el centro CEDIN. Este sistema abre un gran número de posibilidades desde el punto de vista de la supervisión energética como herramienta para el control y ahorro de energía, esta última con una gran importancia para el país. También permite la configuración de las diferentes variables, gestionando el cálculo del consumo en los diferentes centros consumidores de energía de la universidad.

A pesar de las numerosas ventajas que ofrecen las soluciones previamente descritas (a escala nacional e internacional) no son factibles en algunos casos, o no son posibles en otros, el uso de estas como aplicaciones para el control del consumo eléctrico en la UCI.

Por un lado, las aplicaciones disponibles en el mercado internacional, a pesar de ser soluciones sumamente robustas, tienen un propósito mucho más general asociado al análisis de redes eléctricas. Su compra, además del costo, implica el uso de módulos y funcionalidades inútiles o poco útiles en el entorno de la universidad.

Por otro lado, las aplicaciones desarrolladas en el entorno nacional son aplicaciones a la medida, desarrolladas para resolver un problema específico en las entidades que les dieron origen. Además de esto, están asociadas a modelos específicos de metrocontadores muy diferentes a los existentes en la UCI.

Si se analiza el sistema SCADA, este carece en la actualidad de una aplicación que pueda publicar los datos de forma visible a los usuarios, por lo que se hace necesario el desarrollo de una plataforma Web que los visualice, lo cual constituye el objetivo principal de este trabajo. Con esto, se podrá aprovechar la ventaja que ofrece el Sistema SCADA para el monitoreo de las variables eléctricas y control de las redes.

1.5 Tecnologías asociadas al desarrollo de herramientas informáticas para la gestión del consumo eléctrico

La selección de las tecnologías y herramientas para desarrollar la aplicación permite establecer, de forma organizada, todo los elementos técnicos a utilizar, con el objetivo de alcanzar una mayor integración, robustez, facilidad y calidad en su elaboración. Esta selección tiene como principal criterio que las tecnologías y herramientas sean de software libre o estén distribuidas bajo licencias adquiridas por la universidad.

1.5.1 Metodología de desarrollo

En el proceso de desarrollo de software, la metodología define “Quién debe hacer Qué, Cuándo y Cómo debe hacerlo”, e influye directamente en el transcurso del desarrollo del sistema. Actualmente no existe una metodología de desarrollo universal y las características de cada proyecto exigen que el proceso sea configurable. Dentro de las más conocidas se encuentran RUP, DSDM y XP.A continuación se describen y analizan las antes mencionadas.

RUP

La metodología *RUP (Proceso Racional Unificado, de sus siglas en inglés)* (11), divide el proceso de desarrollo en nueve flujos de trabajo y cuatro fases. Esta metodología tiene tres características fundamentales: que el proceso es iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura. El que sea iterativo e incremental le permite al equipo de desarrollo, en un tiempo estimado, obtener una pequeña parte del producto, de acuerdo a una fase que pase por todas las disciplinas, esto se conoce como iteración.

Permite además, según cada iteración, un crecimiento del producto, conocido como incremento. La característica de ser dirigido por casos de uso ayuda también al equipo de trabajo a desarrollar el sistema según los requisitos solicitados por el cliente, pues a través de los mismos estos se modelan y reciben seguimiento. Poseer la característica de estar centrado en la arquitectura, indica cómo puede construirse el sistema y en qué orden, además de respetarse en cada flujo la base arquitectónica definida previamente. Todas estas características fueron destacas por los autores del proceso unificado desde sus inicios.

Metodología DSDM

La metodología *Método de Desarrollo de Sistemas Dinámicos* (DSDM) (12)es un proceso dinámico y modular que puede ser implementado tanto bajo metodologías ágiles como tradicionales. Al igual que la metodología *XP, DSDM* incluye al cliente como parte esencial en el proceso de desarrollo del producto.

Entre sus premisas fundamentales está la de tratar de evitar el roce constante entre directivos y desarrolladores, dándole a estos últimos cierto nivel de autoridad sobre la toma de decisiones en el desarrollo, ahorrando el tiempo innecesario en analizar algunos tipos de cambios en el sistema. Plantea un modelo iterativo e incremental centrado en la entrega frecuente de versiones funcionales del producto con el objetivo de detectar y resolver rápidamente los errores que puedan aparecer. Según *DSDM*, es mejor entregar a tiempo algo lo suficientemente bueno que resuelva situaciones críticas, que entregar todo perfecto al final. Esta filosofía se aplica tanto al código fuente como a otros artefactos como documentación, requerimientos o modelos de datos. Este proceso cuenta con cinco fases o etapas.

Programación extrema (XP)

Programación Extrema XP (acrónimo de *Extreme Programming*, del inglés) (13), es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. Esta metodología se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes y simplicidad en las soluciones implementadas. Se define como especialmente adecuada para proyectos con

requisitos imprecisos y muy cambiantes y donde existe un alto riesgo técnico. Entre los artefactos que se utilizan en *XP* vale la pena mencionar las tarjetas de historias; son tarjetas comunes de papel en que se escriben breves requerimientos de un rasgo. Tienen una granularidad de diez o veinte días, se usan para estimar prioridades, alcance y tiempo de realización.

Fundamentos de la metodología seleccionada

A partir de las características destacadas de las metodologías estudiadas, se decide seleccionar RUP. A continuación se describen los motivos fundamentales de la selección:

- Es necesario documentar completamente el proyecto, como requerimiento del centro. RUP permite una documentación detallada a través de numerosos artefactos en cada flujo de trabajo.
- Se puede desarrollar en un corto - mediano plazo (6 - 9 meses), por lo que la metodología se puede adaptar a un desarrollo iterativo en este período, lo cual es una característica de RUP.
- La arquitectura reviste vital importancia en el caso de una aplicación que hibrida desarrollo Web y desktop y que además debe ser escalable. En el caso de la aplicación propuesta, debe permitir el consumo de servicios del SCADA en futuras versiones, por lo que una de las características de RUP la hace ideal para ello: ser centrada en la arquitectura.
- El equipo de desarrollo tiene un alto conocimiento de esta metodología.

1.5.2 Lenguaje de modelado

Un lenguaje de modelado es un conjunto de signos o modelos que permiten transformar el lenguaje común a un lenguaje más técnico, con el objetivo de obtener una mejor comprensión, visualización y descripción del sistema a desarrollar, se utiliza para la creación de diagramas de casos de uso y de clases del diseño, modelos de datos, entre otros. (14)

UML

El *UML (Lenguaje Unificado de Modelado)*, se utiliza para especificar, visualizar y documentar los esquemas de los sistemas de software orientado a objeto. Está compuesto de muchos elementos de esquematización que representan las diferentes

partes de un sistema y es utilizado para crear diagramas, que representa alguna parte o punto de vista del mismo. La versión usada es la *2.0.*

1.5.3 Herramientas CASE

Las herramientas *CASE (Ingeniería de Software Asistida por Ordenador, por sus siglas en inglés)* son aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de un software, reduciendo el costo del mismo en términos de tiempo y dinero. Ayudan al ingeniero a desarrollar y mantener el *software*. Ofrecen la verificación y mantenimiento de la consistencia de la información del proyecto. Hacen más fácil la planificación y gestión del proyecto informático, la aplicación de técnicas de reutilización y reingeniería y el establecimiento de estándares en el proceso de desarrollo y documentación (14)**.** Estas herramientas pueden ayudar durante el ciclo de vida de desarrollo del software en disimiles tareas, como son el diseño del proyecto, cálculo de costos, implementación básica del código, compilación automática y documentación. Algunas de estas herramientas son:

Visual Paradigm

Es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Además de ello soporta todos los diagramas y modelos típicos del desarrollo de software, lo que lo hace abierto a cualquier metodología. Agiliza la construcción de aplicaciones y se comercializa bajo licencia privativa. Entre las principales características de esta herramienta *CASE* se destacan que: soportan el lenguaje de modelado *UML*, se puede realizar la ingeniería inversa, genera código fuente y archivos de Bases de Datos (15).

Rational Rose

Rational Rose es una herramienta *CASE* que cubre todo el ciclo de vida de un proyecto, concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases. Está completamente relacionada a *RUP* desde su concepción. Brinda la posibilidad de modelar y visualizar los procesos de negocio, además permite destacar las oportunidades para aumentar la eficiencia y el trabajo en equipo dando la posibilidad de que haya varias personas a la vez trabajando, para ello posibilita que cada desarrollador opere en un espacio de trabajo privado y que

tenga un control exclusivo sobre la propagación de los cambios en este espacio de trabajo. Otra de las utilidades que brinda esta herramienta *CASE*, es la Ingeniería Inversa que permite, mediante el código de un programa, se obtenga información sobre su diseño (14).

A pesar de ser una excelente herramienta *CASE*, presenta un alto consumo de los recursos de la máquina para su funcionamiento de forma eficiente. Esto provoca dificultades a la hora de editar los diagramas y trabajar con ellos. Al igual que *Visual Paradigm* se comercializa bajo licencia privativa.

Fundamentos de herramienta CASE seleccionada

Las herramientas descritas son utilizadas en disímiles escenarios a escala mundial dadas sus características, que plantean importantes ventajas y beneficios al proceso de desarrollo de sistemas informáticos. *Rational Rose* está respaldada por una importante empresa y comunidad que garantizan su eficiencia y calidad, pero existe gran preferencia cuando se utilizará expresamente la Metodología *RUP*. Por otro lado, los costos de licencia, sus *plugins* y módulos adicionales también son comercializados.

Por otro lado, *Visual Paradigm*, la cual se considera la herramienta a utilizar, se plantea como potente herramienta multiplataforma, de código abierto en muchos de sus módulos principales y costos medios en cuestión de pago de licencias. Está sugerida como herramienta principal de modelado en los Proyectos de Producción de la UCI, institución que ha adquirido su licencia. Es vinculable a Entornos de Desarrollo Integrado (*IDEs*, del inglés) como *NetBeans*, muy empleado en diversos lenguajes de importancia como *Java* y *PHP*, que pudieran ser considerados para el desarrollo de la propuesta. Al igual que *Rational Rose* permite ingeniería inversa y generación de código fuente de diversos lenguajes, con fuerza en *Java*, tecnología sobre la que está basada. Es configurable a entornos colaborativos, en los cuales diversas personas pueden trabajar al mismo tiempo sobre un mismo proyecto.

1.5.4 Sistemas gestores de base de datos

Un Sistema Gestor de Base de Datos (SGBD) es un programa que permite crear y mantener una Base de Datos (BD), asegurando su integridad, confidencialidad y seguridad. El propósito general de los SGBD es manejar de manera clara, sencilla y

ordenada un conjunto de datos que posteriormente se convertirán en información relevante para una organización. Estos sistemas permiten describir los elementos de datos con su estructura, sus interrelaciones y sus validaciones. Entre los más utilizados se encuentran *Oracle*, *PostgreSQL* y *MySQL* (16).

Oracle

Es un SGBD que utiliza una arquitectura Cliente-Servidor. Es considerado como uno de los sistemas de BD más completos, destacando su: soporte de transacciones, estabilidad, escalabilidad y el ser multiplataforma (17). Entre las principales ventajas se pueden citar:

- El manejo BD relacional que hace uso de los recursos del sistema informático en todas las arquitecturas de *hardware*.
- Soporta la mayoría de los lenguajes de computación al igual que 26 idiomas diferentes.
- Se ejecuta automáticamente en más de 80 arquitecturas de *hardware* y *software* distinto sin tener la necesidad de cambiar su código fuente.
- Está disponible en múltiples plataformas como *Windows*, *Linux*, todas las versiones de *Unix* ofrecidas por diversas empresas como *IBM, Sun, Digital, HP* y *Sequent* (17).

PostgreSQL

Es un SGBD que comenzó como un proyecto en la Universidad Bekerley de California. Este proyecto sigue actualmente un activo proceso de desarrollo a nivel mundial por un equipo de desarrolladores y contribuidores de código abierto (17). Puede funcionar en múltiples plataformas y a partir de la versión 8.0, también en *Windows* de forma nativa. Para las versiones anteriores existen versiones binarias para este sistema operativo, pero no tienen respaldo oficial. Sus ventajas más notables son:

- Está publicado bajo licencia *BSD*.
- Es orientado a objetos.
- Está diseñado para ambientes de alto volumen.
- Aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas.
- Soporta la integridad referencial, que se utiliza para asegurar la validez de los datos de las BD.

- La flexibilidad del *API* (Interfaz de Programación de Aplicaciones, del inglés) de *PostgreSQL* ha permitido proporcionar soporte al desarrollo fácilmente para el RDBMS15 *PostgreSQL*. Estas interfaces incluyen *Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++,* y *Pike*.
- Usa una arquitectura de proceso por usuario, Cliente-Servidor *PostgreSQL* (17).

MySQL

Es un Sistema de gestión de bases de datos relacional, licenciado bajo *GPL* de *GNU*, permite soportar una gran carga de forma eficiente, es uno de los gestores de bases de datos más usados en el mundo, debido a su rapidez y facilidad de uso. En febrero del 2008 es desarrollado por *SUN Microsystems* y en el 2009, *Oracle* realizó la adquisición de *SUN* con todos sus productos. Este SGBD proporciona sistemas de almacenamiento transaccional y no transaccional, las funciones *SQL* que presenta están implementadas usando una biblioteca altamente optimizada, usa fácilmente múltiples núcleos y es bastante rápido de implementar (18).

Fundamentos del SGBD seleccionado

La investigación realizada permite proponer como gestor de BD el *PostgreSQL* en su versión **9.2.4.1**, ya que:

- Está publicado bajo licencia *BSD*, que pertenece al grupo de licencias de software libre.
- Está pensado para ambiente de alto volumen de información.
- Soporta la integridad referencial, que se utiliza para asegurar la validez de los datos de las BD.
- El proyecto SCADA utiliza este sistema SGBD en su proceso de automatización.

1.5.5 Lenguajes de programación

Los lenguajes de programación están diseñados para permitir la comunicación entre el usuario y las computadoras. Usando los mismos se puede especificar al ordenador las instrucciones que desea ejecutar, con un lenguaje relativamente próximo al lenguaje humano. Se utilizan para crear *software*, que faciliten el trabajo y están integrados por símbolos y reglas semánticas y sintácticas que detallan la estructura y el significado de

sus elementos y expresiones, las instrucciones que forman dicho programa son conocidas como código fuente (17).

J2EE

La edición empresarial de Java, es una plataforma para el desarrollo de aplicaciones Web complejas, que contiene una serie de tecnologías que permiten escribir aplicaciones del lado del servidor, para proporcionar servicios desde redes *TCP/IP*, usando como lenguaje de programación a *Java*.

HTML

De las siglas *(Lenguaje de marcas de hipertexto)*, es el lenguaje de marcado más utilizado para la construcción de páginas Web. Es un lenguaje de composición de documentos y especificación de ligas de hipertexto. El mismo define la síntesis y coloca instrucciones especiales que no muestra el navegador, aunque sí le indica como desplegar el contenido del documento, incluyendo texto, imágenes y otros medios soportados. Su versión 5, permite el desarrollo de aplicaciones con una mayor vistosidad, calidad e integración. Esta versión es soportada por las versiones actuales de los navegadores *Mozilla Firefox*, *Chrome*, *Chromiun*, *Safari* e *Internet Explorer* (19).

CCS

De las siglas *(Hojas De Estilo En Cascada).*Este lenguaje se usa para organizar y controlar la presentación y aspecto de una página Web. Es utilizado principalmente por los diseñadores y programadores, para elegir la multitud de opciones de presentación que este brinda, como son los colores, los tipos y tamaños de letras y las alineaciones de capas. Se selecciona la versión 3 de este lenguaje, ya que presenta muchas características que enriquecen la apariencia de las páginas Web, como son los bordes redondeados, el sombreado de cajas de texto, las animaciones y las transiciones (19).

Java Script

Es un lenguaje de tipo *Script* compacto, basado en objetos y guiado por eventos, diseñado específicamente para el desarrollo de aplicaciones Cliente-Servidor dentro del ámbito de internet. Como ventaja clave de este lenguaje, se puede mencionar que es interpretado por todos los navegadores modernos, debido a que este lenguaje está

provisto de una implementación del Modelo de Objetos del Documento (*DOM)* estándar. Algunas de las características principales de este lenguaje son:

- Es un lenguaje muy sencillo y tiene abundante documentación de referencia en Internet.
- Tiene una curva de aprendizaje elevada.
- Con el uso de este lenguaje se pueden crear páginas Web dinámicas, menús desplegables, manipular datos y crear aplicaciones Web usando poca memoria. (20).

Este lenguaje se utiliza en el desarrollo de la aplicación, para validar los campos de los formularios y para el uso de la biblioteca *High Charts*, que permite la utilización de gráficas de múltiples formas. La versión a usar es la 1.8.5.

Fundamentos del lenguaje seleccionado.

Como lenguaje de programación para el desarrollo de la aplicación Web se utiliza el *Java*. Este lenguaje es uno de los más divulgados en la universidad, por lo que existe abundante documentación del mismo, también con el uso de este, se pueden desarrollar aplicaciones Web de forma sencilla logrando la velocidad, estabilidad y seguridad requerida. Por otro lado, las tecnologías asociadas a servicios Web están muy fuertemente respaldadas por las tecnologías *Java*. Además de este lenguaje se utilizaron los lenguajes *HTML5, CSS3, JAVASCRIPT 1.8.5.*

1.5.6 IDE

Un *IDE (Entorno de Desarrollo Integrado)* es un conjunto de herramientas de desarrollo de software para programadores. Generalmente están compuestos por un editor de código, un compilador, un depurador y un constructor de interfaz gráfica de usuario. Es creado para el desarrollo de aplicaciones, usando diversos lenguajes de programación. Cuando el lenguaje que soporta el mismo, es orientado a objetos, incluyen navegador de clases, inspector de objetos y diagrama de jerarquía de clases (21).

Netbeans

Es un entorno integrado de desarrollo de *software*, gratuito y de código abierto, usado por algunos desarrolladores. Contiene herramientas para crear aplicaciones profesionales

para el escritorio, la empresa, la Web y los equipos móviles con lenguajes de programación como *Java*, *PHP*, *C/C++* y *Ruby*.

Es fácil de instalar, de usar y posee versiones para varias plataformas. La versión de *NetBeansIDE*8.0, provee mejoras en el rendimiento, menor consumo de memoria y mejor respuesta cuando se trabajan con proyectos grandes. Algunas de las tecnologías y lenguajes soportadas por este *IDE* son: *C/C++, Java*,

Java Script, *SOAP*, *Java EE* y *XML* (22). Se selecciona este *IDE* en su versión 8.0 ya que:

- Es un entorno Integrado de Desarrollo gratuito.
- Permite la edición, depuración y compilación del código *Java*.
- Permite el rápido desarrollo de una interfaz visual, haciendo uso de los componentes *SWING*.
- Soporta *Java2*, *HTML5*, *CCS3*, y *JSP2.1*
- Posee versiones para varias plataformas.

1.5.7 Bibliotecas usadas

Las bibliotecas son archivos externos a la aplicación desarrollada, que contienen funciones. Las mismas simplifican el desarrollo del *software* al ser reutilizables.

High Charts

Es una biblioteca escrita en *Java Script* basada en *jQuery*, que permite la creación de gráficas. La misma ofrece un método fácil e interactivo para insertar gráficas en un sitio o aplicación Web; esta biblioteca es compatible con todos los navegadores modernos. No es comercial y no se necesita el permiso de los autores para su implementación en sitios Web personales o con fines de lucro. Es de código abierto y todas las características pueden ser personalizadas permitiendo una gran flexibilidad (23). La versión a usar es la 3.0.5.

JQuery 2.0

Es una biblioteca de *JavaScript* que permite simplificar la manera de interactuar con los documentos *HTML*, manipular el árbol *DOM*, manejar eventos, desarrollar animaciones

(*FLV*) y agregar interacción con la técnica *AJAX* a las páginas Web. Es la biblioteca de *Java Script* más utilizada (24), es libre y de código abierto que posee un doble licenciamiento bajo la Licencia *MIT* y la Licencia Publica General *GPL* v2, permitiendo su uso en proyectos libres y privativos.

1.5.8 Servidor Web

Apache Tomcat 8.0

Para la ejecución del código *Java* de los módulos de la aplicación se utiliza el servidor *Apache Tomcat 8.0*. Este servidor vine integrado al entorno de desarrollo *NetBeans*, aunque puede ser adquirido e instalado de manera independiente, incluso como *CGI*. *Tomcat* es el más potente servidor *Java* de entre los competidores, pues garantiza eficiencia y calidad en el servicio de aplicaciones (25).

Como principales elementos para la presente investigación se tuvieron en cuenta algunas de sus características como:

- Alto rendimiento
- Eficiencia y rapidez en las respuestas.
- Seguridad de la aplicación y sus datos.
- Alta compatibilidad y acoplamiento para servicios Web.

1.6 Conclusiones parciales

A partir del estudio de posibles soluciones a escala internacional y nacional, al problema de la investigación, se pudo constatar la viabilidad y necesidad de la elaboración del sistema propuesto. Los sistemas existentes no se ajustan a los problemas y especificaciones de la universidad, siendo muy generales en algunos casos o muy específicos en otros. Se identificaron las tecnologías, herramientas y metodología a utilizar en el desarrollo del sistema informático para la visualización de los valores asociados al consumo energético, se garantiza con ellas la calidad de *software*, la posibilidad de ejecución multiplataforma y la documentación necesaria de apoyo.

Capítulo 2 Características y Diseño de la Aplicación

2.1 Introducción

En el presenta capítulo se comienza el *Proceso de Desarrollo de Software* siguiendo la metodología *RUP*. Se parte de un estudio del entorno en que tomará parte fundamental la aplicación, a partir de la mejora de procesos mediante automatización y posteriormente se comienza el diseño de la misma, partiendo de los elementos arquitectónicos más significativos y definiendo cada una de las partes lógicas y físicas que le conformarán. Se muestran los elementos de las 4+1 vistas de la arquitectura.

2.2 Modelado del Negocio

El propósito principal de esta fase en el proceso de desarrollo de *software* es dar un entendimiento del negocio especificando los conceptos del dominio del problema que permitan determinar las necesidades operacionales que dan paso a la fase inicial del sistema.

Modelo de dominio

El Modelo de Dominio (*figura 1*) es un caso especial de un modelo de negocio que tiene como objetivo fundamental comprender y describir las entidades fundamentales dentro del dominio del problema incluyendo a los principales conceptos y actores del sistema. Se propone representar elementos del mundo real y establecer las relaciones entre ellos como paso previo al diseño de la propuesta. Además de esto, el Modelo de Dominio o Diagrama Conceptual es un artefacto más del Modelo de Negocio de *RUP* y el único que se puede realizar cuando no hay procesos claros de negocio

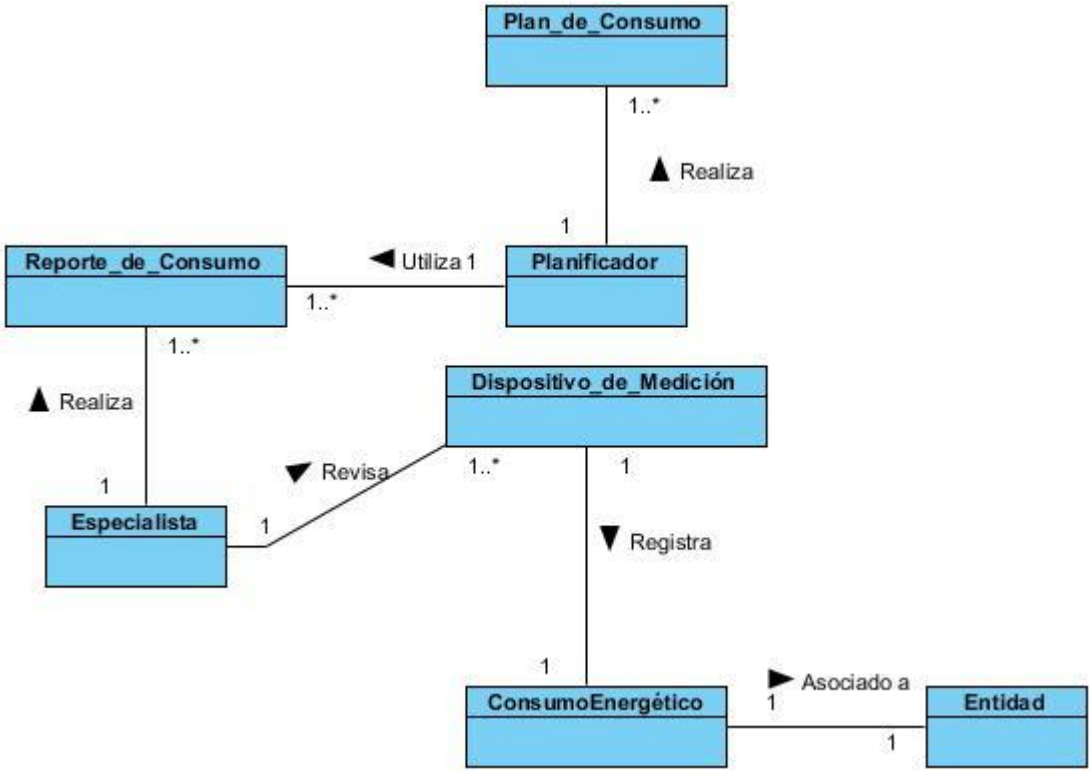


Fig. 1 Modelo de dominio

Descripción de los conceptos del dominio

Especialista: Representa un trabajador de la entidad, encargado de realizar la revisión de los dispositivos de medición e informar sobre el consumo.

Reporte de Consumo: Informe detallado por área que muestra el consumo real durante un determinado período de tiempo.

Consumo Energético: Medida de gasto de electricidad asociado a un área o entidad en un período de tiempo determinado. Se muestra en unidades de *Kw/h.*

Entidad: Representa un área o local dentro de una institución dada o la institución en sí.

Dispositivo de medición: Representa los dispositivos utilizados para el control del consumo energético: metro contadores, contadores digitales, etc.

Planificador: Encargado de realizar la estimación del consumo de una determinada área. Comúnmente es un ente dentro de la Administración Central del Estado que planifica la demanda.

Plan de Consumo: Representa la media estimada que debe consumir un área determinada en un período de tiempo establecido (por lo general 30 -365 días).

2.3 Descripción de los requisitos funcionales

Los requisitos funcionales definen las condiciones o capacidades que el sistema será capaz de realizar. Estos describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Estos requisitos, al tiempo que avanza el proyecto de *software*, se convierten en los algoritmos, la lógica y gran parte del código del sistema (26)**.**

En el caso de la propuesta que se describe en este trabajo, el sistema debe ser capaz de:

- **RF.1**Visualizar consumo total por áreas.
- **RF. 2** Visualizar consumo diario por áreas.
- **RF. 3** Visualizar consumo semanal por áreas.
- **RF. 4** Visualizar consumo mensual por áreas.
- **RF.5** Visualizar consumo anual por áreas.
- **RF.6** Visualizar históricos de consumo por fecha y hora.
- **RF.7**Visualizar entidad más consumidora.
- **RF.8**Visualizarentidad menos consumidora.
- **RF.9**Visualizarpromedio de consumo.
- **RF.10**Visualizarmayor consumo.
- **RF.11**Visualizarmenor consumo.
- **RF.12** Actualizar estado de consumo.

A partir de los requisitos se definen los Caso de Uso del Sistema (*CUS*), elemento base de la Metodología *RUP*.

2.3 Descripción de los requisitos no funcionales

Los requisitos no funcionales son las exigencias de cualidades que se le imponen al sistema que se desarrolla. Estos son los encargados de definir la usabilidad, los elementos de *hardware* y de *software* necesarios para el funcionamiento, la apariencia, la seguridad, *etc*. (26)

RNF 1. Interfaz gráfica o apariencia externa:

El sistema debe poseer una interfaz gráfica uniforme incluyendo pantallas, menús y opciones. Tanto los títulos de los componentes de la interfaz, como los mensajes para interactuar con los usuarios, deberán ser en idioma español y tener una apariencia uniforme guiada por los colores blanco y azul. Los mensajes definidos deberán ser lo suficientemente informativos para dar a conocer la severidad de los mismos. Los colores que indiquen el consumo se corresponderán con los valores mostrados (variedades para tipos de indicadores de días, semanas, meses, *etc*.).

RNF 2. Fiabilidad:

El sistema deberá prever contingencias que pueden afectar la prestación estable y permanente del servicio y garantizar la capacidad para capturar excepciones.

RNF 2.1. Seguridad:

RNF 2.1.1. Integridad:

- Garantizar la entereza en toda la información que se almacene, bajo rigurosa protección contra alteraciones indebidas.

RNF 2.1.2. Disponibilidad:

- Garantizar la disponibilidad de la información en todo momento.
- Garantizar velocidad en los procesos ejecutados por el sistema.
- Estimar tiempos aceptables del sistema fuera de línea.

RNF 3. Usabilidad:

El sistema debe permitir un alto nivel de facilidades de uso, basado en el cumplimiento de los siguientes aspectos:

- Proporcionar a los usuarios toda la documentación necesaria para lograr un entendimiento total de las acciones a realizar en el sistema.
- Permitir una navegación sencilla, tanto a los usuarios con conocimientos avanzados de informática como a los usuarios más inexpertos, esto se logrará a partir de una

estructura de la información correcta, la cual en todo momento los usuarios tendrán conocimiento del lugar donde se encuentra en el sistema.

RNF 4. Eficiencia:

RNF 4.1. Rendimiento:

- Tener alta velocidad de procesamiento o cálculo, tiempo de respuesta y de recuperación, y disponibilidad.
- Optimizar la ejecución de procesos de sistema para disminuir la congestión de recursos.

RNF 4.2. Capacidad:

- Alojar un gran número de clientes y de transacciones en el sistema.
- Soportar múltiples conexiones al sistema de manera simultánea.
- Considerar el crecimiento esperado en el volumen de datos.

RNF 5. Software:

Algunos de los requerimientos mínimos de software son los siguientes:

Para las PCs Clientes:

- Un navegador Web como: *Mozilla Firefox, Zafari* u otro navegador que cumpla con los estándares *W3C*.
- Sistema operativo: *GNU/Linux y Windows*.

Para las PCs Servidores:

- Sistema operativo *GNU/Linux o Windows* (*XP* o Superior).
- Servidor *Web Tomcat 7.0* o superior.
- *PostgreSQL* como Sistema Gestor de Base de Datos.

RNF 6. *Hardware*:

Para las PCs Clientes:

- Se requiere tengan tarjeta de red.

- Al menos 256 MB de memoria RAM.
- Procesador 512 *MHz* como mínimo.

Para los Servidores:

- Se requiere tarjeta de red.
- El Servidor de Aplicación debe tener como mínimo 2 GB de RAM.
- El Servidor de base de datos debe tener como mínimo 2 GB de RAM.
- Procesador 3 *GHz* como mínimo.

2.4 Modelado del Sistema

El modelado del sistema permite describir cada uno de sus componentes físicos y lógicos. A partir del análisis de los requisitos de *software*, se definen los casos de uso y, a partir de cada uno de ellos, la composición del sistema para que su realización llegue a feliz término. Uno de los elementos fundamentales es el Diagrama de Casos de Uso del Sistema.

El diagrama de *CUS* es el modelo de las funciones deseadas para el sistema y su entorno, que sirve como contrato entre el cliente y los desarrolladores. Estas funciones por lo general agrupan uno o más requisitos funcionales definidos en el flujo de trabajo de Modelado, específicamente en la disciplina de Requisitos.

Una vez recopilados los requisitos funcionales del sistema, se muestra el diagrama de caso de uso del sistema (*figura 2*), en el cual se representan los actores y su relación con el sistema:

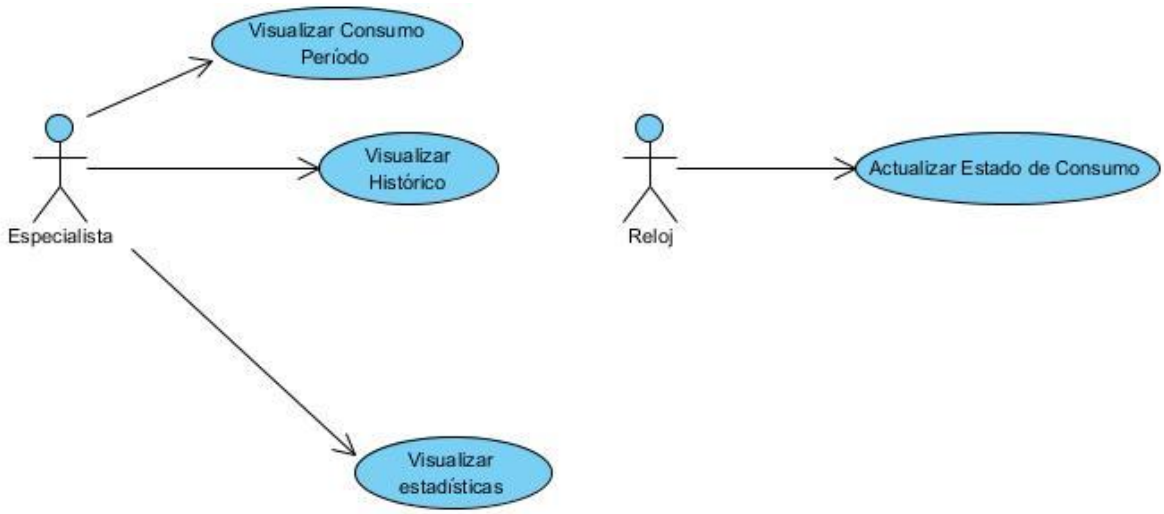


Fig. 2 Diagrama de Casos de Uso del Sistema

2.4.1 Descripción de los Casos de Uso del Sistema

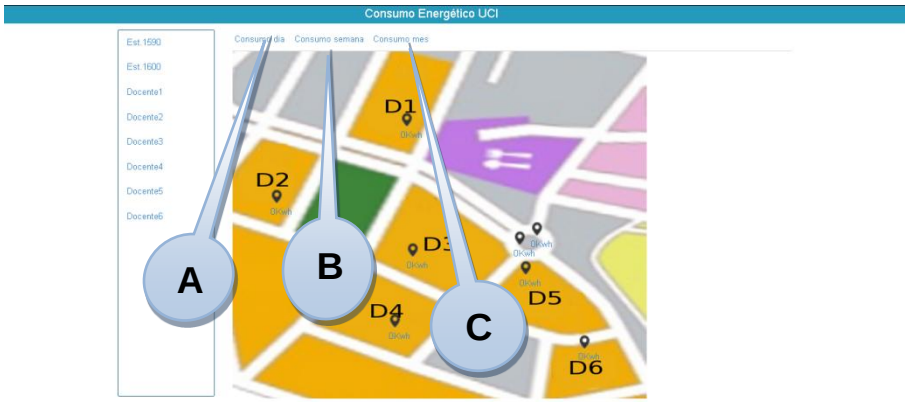
Es sumamente importante dejar descritos de manera detallada cada uno de los *CUS* identificados. Esto permite a los miembros del equipo de desarrollo poder entender y descomponer cada caso de uso para su diseño e implementación. A continuación se describen algunos de los principales *CUS* (*tabla 1*).

Tabla 1 Descripción del Caso de Uso

Caso de Uso:	Mostrar Consumo Período
Actores:	Especialista
Propósito:	Mostrar resumen del consumo de energía de un determinado centro consumidor dado un período, a partir de estadísticas y gráficas.
Resumen:	El caso de uso se inicia cuando el Especialista desea ver el consumo de un ente consumidor en un determinado período. Termina cuando se muestra el reporte de consumo.

Precondiciones:	
Referencias:	RF.1 - RF.5
Prioridad:	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el usuario selecciona una de las opciones de consumo de área y luego un determinado período. Pueden ser: **-**Consumo día - Consumo semana - Consumo mes	**2.** El sistema realiza la operación según la opción seleccionada por el usuario. **-** Si selecciona “Consumo día”, ver sección “Consumo día”. **-** Si selecciona “Consumo semana”, ver sección “Consumo semana”. **-** Si selecciona “Consumo mes”, ver sección “Consumo mes”.
	3. El caso de uso termina cuando el sistema procesa la información según la acción realizada por el usuario y actualiza la visualización en los gráficos.
Prototipo de Interfaz	

Interfaz 1



Sección “Consumo día”	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción Consumo día (A).	**2.** El sistema muestra la interfaz de consumo día
	3. El sistema actualiza los gráficos para el consumo.
Sección “Consumo semana”	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción Consumo semana (B).	**2.** El sistema muestra la interfaz de consumo semana

	3. El sistema actualiza los gráficos para el consumo.
Sección “Consumo Mes”	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción Consumo mes (C).	**2.** El sistema muestra la interfaz de consumo mes
	3. El sistema actualiza los gráficos para el consumo.

2.5 Principios arquitectónicos

Una de las principales características de *RUP* es que todo el trabajo se realiza "centrado en la arquitectura", por lo que la definición de su base y posterior implementación son procesos claves.

2.5.1 Estilos y patrones arquitectónicos y de diseño

La definición oficial de *Arquitectura de Software* propuesto por la *IEEE* resume que:

“La arquitectura de software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán y los principios que sustentan su diseño y evolución”

Un patrón se define como una solución probada con éxito que aparece una y otra vez ante determinado tipo de problema en un contexto dado. Los patrones se definen por un nombre, un problema, una solución y las consecuencias de su aplicación. Éste define una

posible solución correcta para un problema de diseño dentro de un contexto dado, describiendo las cualidades invariantes de todas las soluciones. (27)

Por otro lado, los estilos también resuelven problemas, solo que mucho más generales y se relacionan con elementos que no solo tienen que ver con el software y su comportamiento, sino que además con las normas que este sigue para la comunicación, interacción con otros sistemas, tratamiento de la información e incluso tecnologías que le soportan.

Estilos y Patrones utilizados

A partir de los elementos necesarios y características que debe tener la aplicación se utilizan los siguientes estilos y patrones:

Patrones GRASP

Los patrones Generales de *Software* para la Asignación de Responsabilidades, *GRASP*, describen los principios fundamentales de la atribución de responsabilidades a objetos, expresados en forma de patrones (26).

Estos constituyen un apoyo para la enseñanza que ayuda a uno a entender el diseño de objetos esencial, y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable (26).

Experto: Es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él no se pretende designar una idea oscura ni extraña; expresa simplemente la "intuición" de que los objetos hacen cosas relacionadas con la información que poseen (26).

Este patrón está presente en la clase *app.java* de la aplicación *desktop*, que es la encargada de iniciar y detener la actualización de datos.

Creador: Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento (26).

El patrón creador se pone de manifiesto en la clase *index.jsp* cuando se crea una instancia de las clases del paquete *funcionalidades*.

Bajo Acoplamiento: Es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño. El acoplamiento es una medida de fuerza que indica la dependencia de un componente con otro (26).

Otros patrones:

Singleton: El propósito de este patrón, según (28)**,** es permitir que una clase sólo tenga un ejemplar, lo cual le permitiría un acceso global desde varias entidades y podría ser reutilizado sin la necesidad de crearlo nuevamente. Este patrón se pone de manifiesto en la clase *funciones.java* donde se crea una instancia única del modelo con la clase *Conexion_metro*, utilizada en todas las llamadas de reporte de resultados.

2.6 Diseño de la aplicación

El Modelo de Diseño describe, dentro de la metodología *RUP*, los principales componentes lógicos y físicos, a partir de clases y sus relaciones, que conformarán el sistema a desarrollar. Generalmente parte de los elementos definidos en la línea Base de la Arquitectura y fundamenta el uso de los estilos y patrones establecidos. En el caso del sistema propuesto, está conformado por dos módulos principales:

- Módulo de actualización (*Desktop*): Encargado de la actualización constante de los valores de las variables eléctricas a partir de la información proveniente del *SCADA*, con el objetivo de evitar la sobrecarga en el consumo directo de la información de su base de datos y de hacerlos disponibles en todo momento para ser visualizados en el módulo Web. Se ejecuta como un hilo de procesamiento constante en el servidor.
- Módulo de Visualización (Web): Encargado de la visualización gráfica de los valores de consumo energético por áreas, almacenados por el módulo *desktop*, mostrando además comparaciones y estadísticas asociadas al consumo.

Fundamentos de la utilización del módulo Desktop y del módulo Web

Se decide hacer una aplicación *desktop* y una aplicación Web porque la base de datos original (*SCADA*) que almacena el consumo de los dispositivos de medición, es muy extensa y además no está *indexada*. Esto conlleva a que, cuando la aplicación Web está

conectada directamente a la base de datos del *SCADA*, el tiempo de respuesta de la consulta es muy prolongado. Para optimizar este proceso, se tomó como medida hacer una aplicación *desktop* que tomara los últimos 5 datos recogidos por la base de datos *SCADA* y con esto acortar el tiempo de respuesta de las consultas. Por otro lado la aplicación Web facilita el acceso desde la red.

A continuación se describen los diagramas de clases del diseño para los módulos anteriormente mencionados (figuras 3 y 4).

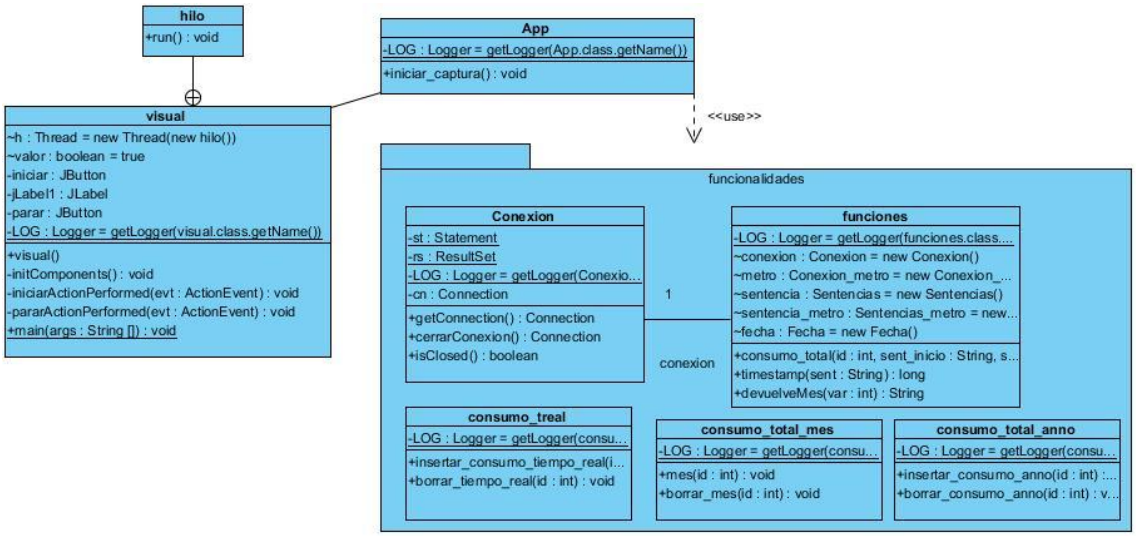


Fig. 3 Diagrama de Clases del Diseño módulo Desktop

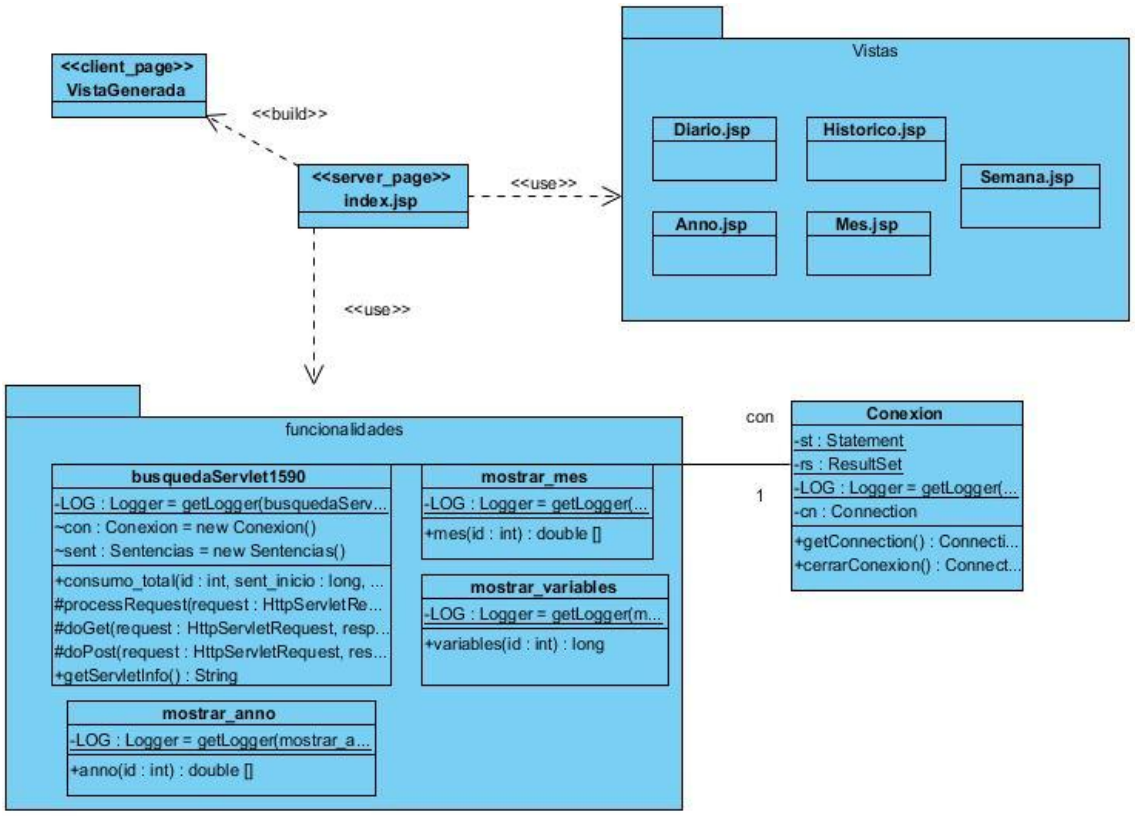


Fig. 4 Diagrama de Clases del Diseño módulo Web

2.7 Manejo de Datos

El manejo de información persistente es de vital importancia para las aplicaciones de visualización. Esto permite tener disponibilidad de los datos en todo momento para su uso y/o análisis. El uso de Bases de datos es la práctica más general en el caso de los sistemas informáticos.

En el caso de la aplicación propuesta en la presente investigación se hace necesario el almacenamiento temporal de la información del consumo de las diferentes áreas de la entidad en una base de datos local, utilizando tablas independientes para cada información.

RUP plantea entre sus flujos de trabajo el relacionado con la modelación de la base de datos, donde se tiene en cuenta la información a almacenar, sus características y relaciones. A continuación se muestra el Modelo Físico de Datos del sistema (*figura 5*).

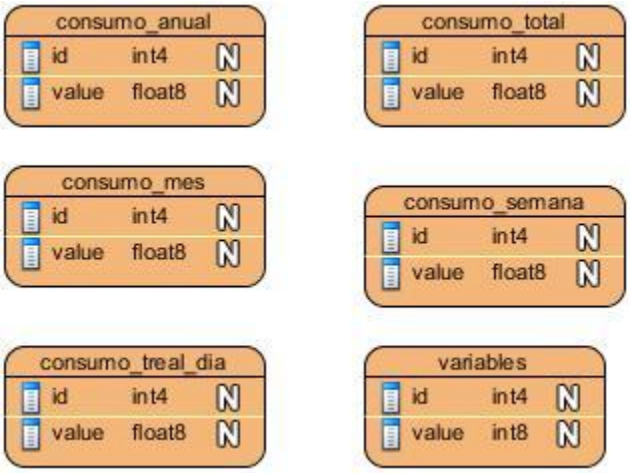


Fig. 5 Modelo Físico de Datos

2.8 Diagrama de Despliegue

El diagrama del despliegue es una red de símbolos de nodos conectados por líneas que muestran las asociaciones de comunicación. Son, en gran medida, semejantes a los diagramas de objetos. Generalmente, muestran las instancias individuales del nodo implicadas en un sistema. Muestra la configuración de los nodos de proceso y las instancias de componentes y objetos que residen en ellos (26). A continuación se visualiza el Diagrama de Despliegue de la aplicación (*figura 6*).

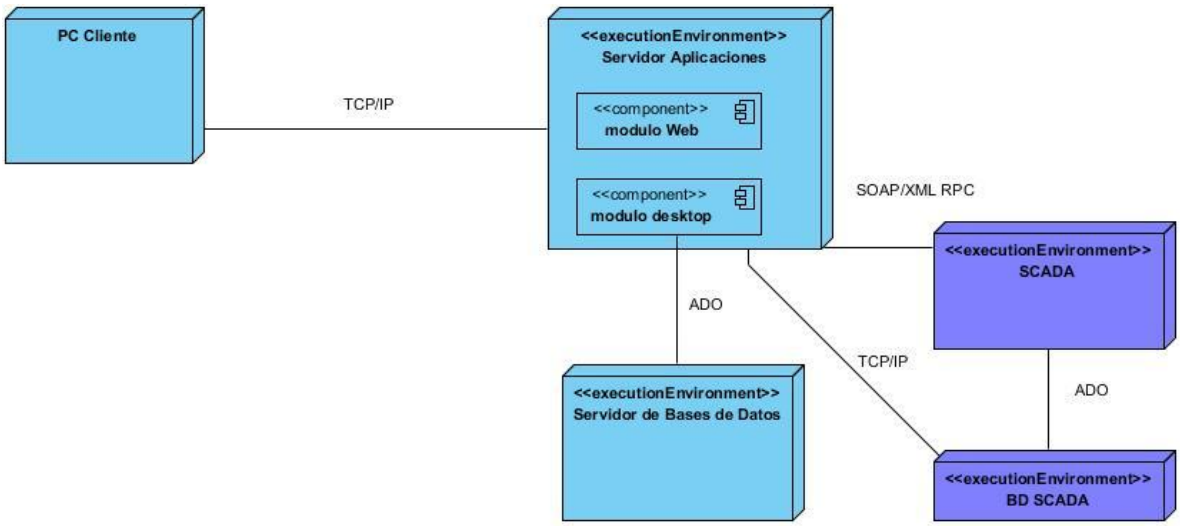


Fig. 6 Diagrama de Despliegue

Es importante destacar que los elementos resaltados en color diferente representan entes de una aplicación externa: el *SCADA*. Estos nodos son accedidos mediante la aplicación a partir del consumo de Servicios Web y acceso a Datos.

2.9 Conclusiones parciales

Una vez analizado el entorno en el cual debe impactar la aplicación, e interactuar con los clientes para identificar las funciones que debe realizar el sistema, se pudieron identificar los requisitos funcionales. A partir de ellos, sus características y las implicaciones en cada caso, se definieron los elementos principales de la base arquitectónica, a partir de la identificación de clases y sus interacciones. Una vez tenida la vista física y lógica del sistema, respetando los requerimientos no funcionales y los principios básicos de disponibilidad de la información y funcionamiento de la entidad, se plantearon las variantes para el despliegue. Todos estos elementos permiten la continuación de los flujos de trabajo y fases de *RUP* para la completa implementación y terminación del sistema.

Capítulo 3 Implementación y Pruebas

3.1 Introducción

En el presente capítulo se analiza la etapa de implementación, la cual denota el estado actual del sistema en términos de componentes. A fin de garantizar el correcto funcionamiento de la solución se definen las pruebas del *software*, las cuales garantizan la calidad del *software* y la revisión final del cumplimiento de las especificaciones del diseño

3.2 Componentes de la aplicación

Se puede definir un componente como: “Una parte modular, desplegable y reemplazable de un sistema que encapsula implementación y expone un conjunto de interfaces”. (27)O sea, describen la organización de los elementos físicos que implementan el sistema y se emplea particularmente para modelar la vista de implementación.

Un componente es una parte del sistema sustituible, casi independiente e importante que desempeña una función clara en el contexto de una arquitectura bien definida (11). Algunos estereotipos de componentes son los siguientes:

<<*executable*>> programa que puede ser ejecutado en un nodo.

<<*file*>> fichero que contiene código fuente o datos.

<<*library*>> librería estática o dinámica.

<<*table*>> tabla de base de datos.

<<*document*>> documento.

A continuación se visualizan los Principales Componentes de la aplicación (*figura 7*).

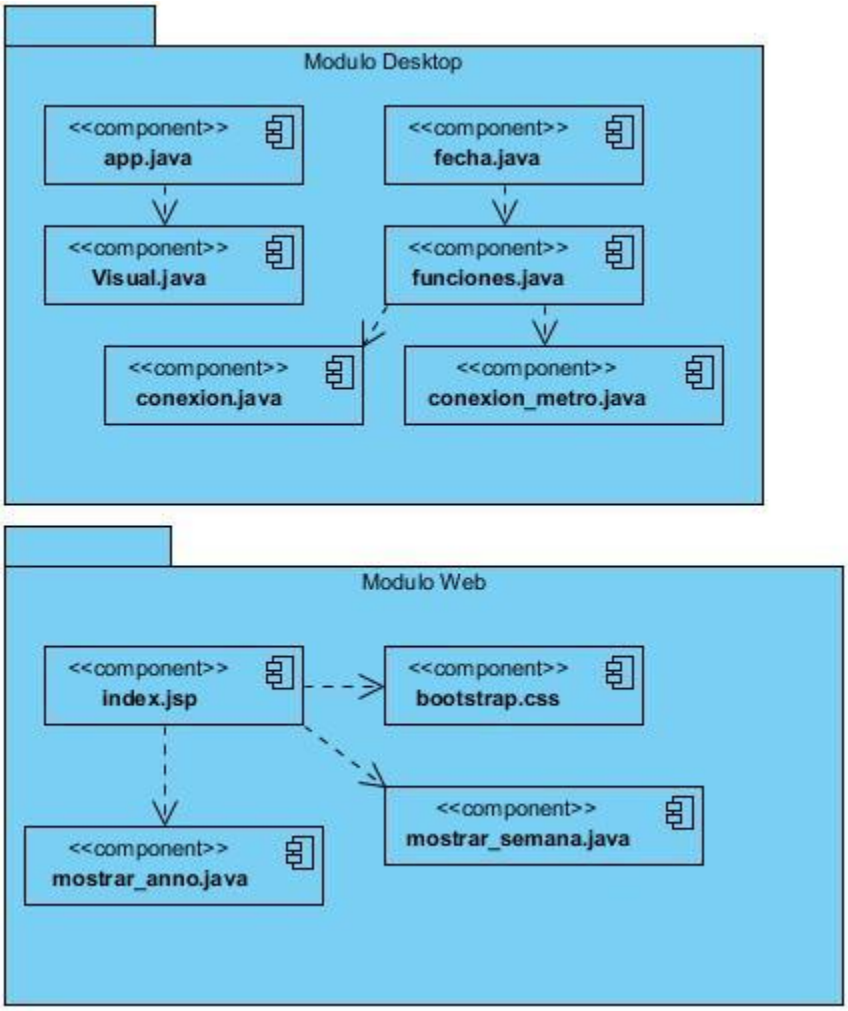


Fig. 7 Principales componentes

3.3 Estándar de codificación

El estándar en el código de programación es muy importante en cualquier proyecto, principalmente si este involucra muchos desarrolladores. Esto asegura que el código sea de alta calidad, que contenga una cantidad baja de errores y sea fácil de mantener.

Es importante que durante la codificación se consideren permanentemente los siguientes criterios de calidad:

- **Facilidad de Comunicación**: Proporcionar al usuario entradas y salidas fácilmente asimilables.

- **Auto descripción:** Proporcionar en el código, explicaciones sobre la implantación realizada.

- **Simplicidad:** La implantación realizada debe hacerse de la forma más comprensible posible.

A continuación la reglas que rigen el estándar de codificación utilizado por el equipo:

1. Los nombres de los métodos y objetos comienzan con minúscula, en caso de tener dos o más palabras a partir de la segunda, el primer carácter de cada una se escribe en mayúscula o se mantendrá minúscula separado por *guión bajo (_)*. Ejemplo: getConnection (), metro, consumo_total_dia_actual ().

2. Las variables locales se denotan usando minúsculas y en caso de estar compuestas por dos palabras se separan por el carácter *guión bajo (_).*

3 Se usan comentarios para describir el objetivo de cada método y de algunas variables. Estos solo intentan describir el objetivo de la función o de la palabra reservada debajo, no describe el funcionamiento interior. Para comentar funciones se usa el doble asterisco entre barras "/**/" y para variables la doble barra "//".

4. Solo se declara una instrucción por línea.

5. No se declara más de una variable por línea.

3.4 Pruebas

Las pruebas del *software* son una actividad de control de la calidad realizada en los proyectos para asegurar el correcto funcionamiento del *software*. Tienen como objetivos la verificación de la correcta implementación de los requisitos explícitamente establecidos, la adecuada integración de los componentes que conforman el sistema y la ejecución de casos de prueba que permitan detectar el mayor número de No conformidades y corregirlas antes de la entrega del *software* al cliente.

RUP define cuatro niveles de prueba: unidad, integración, sistema y aceptación, dentro de los cuales se encuentran diferentes tipos de pruebas, como son:

- Pruebas de unidad: se realizan durante la fase de construcción, específicamente en el flujo de trabajo de implementación; las cuales se basan en probar los componentes implementados como unidades individuales. Las pruebas de unidad están divididas en dos grupos, pruebas de caja blanca y pruebas de caja negra.
- Pruebas de integración: se llevan a cabo durante la fase de construcción, las mismas involucran a un número creciente de módulos y terminan probando el sistema como conjunto. Estas pruebas se pueden plantear desde un punto de vista estructural o funcional. Verifican que los componentes interaccionan entre sí de un modo apropiado después de haber sido integrados en el sistema. Se toman como casos de prueba los casos de uso del diseño. Para ello se utiliza el diagrama de secuencia correspondiente y se diseñan combinaciones de entrada y salida del sistema que lleven a distintas utilizaciones de las clases y en consecuencia de los componentes, que participan en el diagrama.
- Pruebas de sistema: prueban que el sistema funciona globalmente de forma correcta. Cada prueba del sistema prueba combinaciones de casos de uso bajo condiciones diferentes. Se prueba el sistema como un todo probando casos de uso unos detrás de otros y si es posible, en paralelo. En este nivel existen una gran variedad de pruebas que se utilizan según el interés que se tenga respecto al funcionamiento del *software*.
- Pruebas de aceptación: se realizan para permitir que el cliente valide y verifique todos los requisitos pactados. Estas pruebas las realiza el usuario final en lugar del responsable del desarrollo del sistema. El cliente es quien impone los requisitos pues quien mejor que él para dar fe de su satisfacción.

Las pruebas de Caja Negra se aplican a la interfaz del *software* permitiendo obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales del programa con el fin de encontrar la mayor cantidad de no conformidades existentes en el producto (27). En el caso del sistema propuesto, no es posible identificar entradas al sistema, pues toda la información que se muestra en los reportes proviene directamente de la Base de Datos. No es necesario realizar pruebas de integración ni entre módulos, pues son independientes, ni entre sistemas, dado que el sistema propuesto no tiene relación directa con ningún otro.

Se propone el uso de Pruebas de Sistema, en este caso pruebas de funcionalidad, rendimiento y estrés.

3.4.1 Pruebas de Funcionalidad

El objetivo fundamental de las pruebas de sistema es establecer que se cumplen todos los requerimientos funcionales sin errores y que el sistema funciona correctamente. Para elaborar o seleccionar qué pruebas hacer, se hace necesario analizar los casos de uso propuestos y sus escenarios. En el caso de las pruebas realizadas, para cada una de las ejecuciones de *JMeter* no se generó ningún error en la respuesta. Se realizaron además pruebas a las funcionalidades, con el objetivo de evaluar su correctitud. En este caso se consideró como error una demora en la respuesta, a una petición, superior a los 10 segundos.

A continuación se detallan los principales resultados de cada una de las pruebas realizadas:

Tabla 2 Resultados de Pruebas de Funcionalidad

Página accedida	**Error**	**Tiempo de respuesta medio (segundos)**
vistas/entidades/Consumo_dia.jsp	No	3.674
vistas/entidades/Consumo_mes.jsp	No	4.034
vistas/entidades/estadísticas.jsp	**Si (Valor de consumo promedio semana signo negativo)**	8.327
vistas/entidades/estadísticas.jsp	Si (Tiempo de espera superado)	**11.204**
vistas/entidades/estadísticas.jsp	No	7.367

vistas/E1590/Diario.jsp	No	6.210
vistas/E1590/Anno.jsp	No	4.210
vistas/E1590/Semana.jsp	No	3.945

Si se analiza en resumen mostrado en la tabla anterior, solamente se encontraron 2 errores en un mismo escenario, asociados esencialmente a la demora de actualización de los datos necesarios para realizar todos los cálculos estadísticos, desde la base de datos y a un problema algorítmico ya solucionado antes de la siguiente iteración. Para cada caso se realizaron 3 ejecuciones, determinando en el caso del tiempo de respuesta el tiempo medio. El resto de las funcionalidades procedieron sin error alguno, demostrando que la propuesta desarrollada es correcta.

3.4.2 Pruebas de rendimiento y carga

Para probar el rendimiento se ejecutó la aplicación y se realizó un monitoreo del consumo de memoria y de la Unidad Central de Procesamiento (*CPU*). El sistema se ejecutó en una estación de trabajo, lo que permitió observar cómo se comportaban estos consumos en cuanto a la cantidad de usuarios que se conectan al mismo para los valores de consumo. Para la simulación del proceso se utilizó el *software JMeter*, que permite simular entornos de prueba para carga, rendimiento y estrés. A continuación se muestran las características de la estación, los niveles de consumo de memoria y *CPU* durante el funcionamiento del sistema (*tabla 2*)

Estación de trabajo:

-Sistema Operativo:

Windows 7

-*Hardware*:

Procesador: *AMD E450 APU with RADEON - 1.65 GHz*

Memoria *RAM*: 6 GB

Disco: 800 GB

Tabla 2: Prueba de rendimiento para la estación de trabajo

Cantidad de usuarios conectados	**Porciento de uso del CPU**	**Memoria RAM consumida(MB)**
10	76.0141	1.3411
20	81.0341	2.235
50	84.9798	2.379

Luego de analizar los resultados se puede verificar que, aunque hay un ligero incremento del consumo de memoria y uso del *CPU* a medida que incrementa el número de usuarios, este incremento no es significativo, por lo que la respuesta del sistema es sumamente estable y correcta, no mostrando sobrecargas o sobreconsumo.

JMeter también permite simular pruebas de carga y estrés, basadas en solicitud/respuesta *HTML* y verificación de tiempos de respuesta y acceso, por lo que se consideró en el escenario de pruebas propuesto. Las siguientes gráficas (*figuras 8 y 9*) muestran los resultados de 20 simulaciones con 10 usuarios y 10 simulaciones con 50 usuarios conectados a la aplicación:

Fig. 8 Resultados de Test de JMeter para 10 usuarios en 20 ejecuciones

Si se analizan los resultados mostrados, la media de tiempo que demora el sistema en dar respuesta a todas las peticiones, a partir de los resultados de 20 ejecuciones es de alrededor de 2 segundos, lo cual es ligeramente rápido. Se puede observar también una capacidad para 16, 629 peticiones/minuto. En ninguna de las pruebas se produjo error alguno.

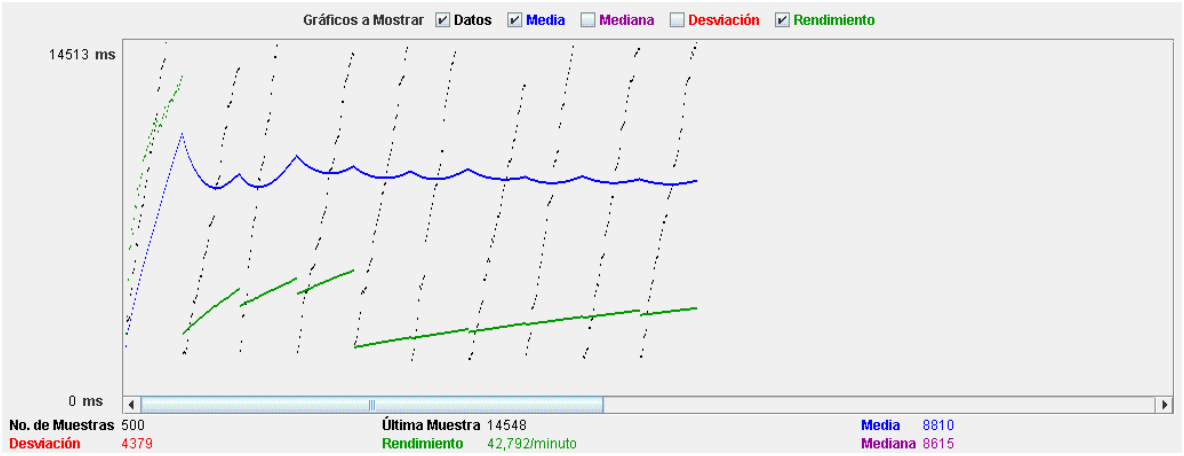


Fig. 9 Resultados de Test de JMeter para 50 usuarios en 10 ejecuciones

Si se analiza el resultado de la segunda gráfica, el tiempo aumenta a 8,8 segundos, lo cual no es significativo con respecto a la previa ejecución si se considera que ha crecido en 5 veces el número de usuarios. La capacidad de soporte a peticiones es, en este caso de 42, 792 peticiones/minuto.

3.4.3 Análisis de los resultados

Teniendo en cuenta los elementos anteriormente descritos, se puede constatar que el sistema es correcto, pues luego de resolver algunas no conformidades detectadas, en todas las ejecuciones realizadas no se encontró ningún error. Por otro lado, los tiempos de respuesta cumplen satisfactoriamente los requerimientos establecidos para un sistema de capacidad para un número de escala media de usuarios o sistemas clientes conectados. El consumo de recursos, a pesar de estar soportado sobre una tecnología que usualmente puede generar carga, es relativamente bajo.

3.5 Conclusiones parciales

A partir de la definición de los principios básicos de la arquitectura y el modelo de diseño de la aplicación se pudieron definir los componentes básicos a implementar, constituyendo el modelo de implementación, que es la base funcional del sistema. Tener en cuenta los componentes principales y sus relaciones permitió cumplir con los requerimientos arquitectónicos y funcionales establecidos, garantizando la completitud y correctitud del sistema. El uso de un estándar de codificación permite mejorar las acciones de soporte de la aplicación y una mejor estructuración del código fuente. La ejecución de las pruebas demostró que la aplicación cumple con las expectativas y requerimientos planteados.

Conclusiones Generales

- El estudio preliminar de soluciones existentes para el control del consumo de energía eléctrica, permitió conocer claramente la factibilidad y pertinencia de la investigación, dejando la necesidad de este tipo de herramientas en la entidad para el apoyo a la toma de decisiones con respecto al despilfarro de energía.
- Las pruebas realizadas al sistema (de funcionalidad, de rendimiento y de estrés) demuestran que se da cumplimiento a los requerimientos establecidos y se obtiene un *software* con la calidad y completitud deseada.
- La propuesta desarrollada ayuda a mejorar el proceso de análisis del consumo de energía en la UCI, a partir de la visualización del proceso de revisión de consumo y reportes especializados, favoreciendo la toma de decisiones por parte de los directivos en función de disminuir el despilfarro de energía eléctrica, cumpliéndose con esto el objetivo de la investigación.

Recomendaciones

Se recomienda continuar con el proceso de desarrollo del sistema informático, con el fin de agregarle nuevas funcionalidades asociadas al estudio de la demanda de cada área, según las necesidades que surjan. Además, publicar la documentación para que futuros investigadores puedan tomarla como referencia para nuevos desarrollos.

Bibliografía y Referencias

1. **Penzias, Arno.** *Ideas e información.La gestión en un mundo de alta tecnología.* Madrid España : Fundesco, 1990. ISBN:84-86094-69-0.

2. **Balcells, Josep, y otros.** *Eficiencia en el uso de la energía eléctrica.* 1ra. s.l. : Marcombo S.A, 2000. 9788426716958.

3. **Agundez, Miguel Ángel y Simancas, Julián Martínez.** *Energía eléctrica. Manual básico para juristas 1.* s.l. : La ley, 2014. 978-84-9020-365-1.

4*Guía para determinar el consumo de energía eléctrica*2005http://www.cubasolar.cu/Biblioteca/Energía/Energía33/HTML/artículo06.htmlISSN 1028-9925

5*El factor de potencia y la eficiencia energética*2005http://www.cubasolar.cu/Biblioteca/Energía/Energía29/HTML.htmISSN/1028-9925

6. **Renzetti, Mario A.** www.e29.com.mx. [En línea] Mario A. Renzetti, 2008. [Citado el: 6 de 10 de 2014.] http://www.e29.com.mx/pdf/FactordePotencia.pdf.

7. www.circutor.es. Viladecavalls (Barcelona) España : s.n. s/n 08232.

8. www.eca.gov.uk. [En línea] s/n 00042.

9. **Gómez, Duniel Rodriguez y García, William Claro.** *Desarrollo de una aplicación web para la supervisión de parámetros eléctricos de un sistema de gestión de energía.* Facultad 5, UCI. La habana : s.n., 2007.

10. **Schneider Electric.** Power Monitoring Expert. *Elcepa - Instalaciones y control.* [En línea] 2013. [Citado el: 03 de 02 de 2015.] http://www.elecpa.es/es/wp-content/uploads/Power_Monitoring_Expert.pdf.

11. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El proceso unificado de desarrollo de software.* 2000.

12. **Hernández, Yasmany Núñez.** *Despliegue del SCADA-UX para Energía UCI.* Facultad 5, UCI. LA habana : s.n., 2012.

13. **Beck, Kent.** *Extreme Programming Explained.* s.l. : Pearson Education Corporate Sales Division, 1999. ISBN/0201616416.

14. **Frías, Roberley Cuadra.** *Desarrollo de una librería de componentes de interfaz gráfica de usuario basada en Swing Application Framework y AWT Framework para el desarrollo de la capa de presentación en aplicaciones de software de escritorio.* Facultad 3, UCI. La habana : s.n., 2012.

15. **González, Yenisleydi Pérez.** *Manejador ODBC para Sistemas de Supervisión y Control de Procesos Automatizados.* Facultad 5, UCI. La habana : s.n., 2011.

16. **Téllez, Isnaudy Pérez y Sañudo, Yasmany González.** *SISTEMA PARA LA GESTIÓN DE LA INFORMACIÓN DE LOS MEDICAMENTOS, TURNOS MÉDICOS Y VACUNAS EN EL HOSPITAL ERNESTO CHÉ GUEVARA.* Facultad 5, UCI. La habana : s.n., 2012.

17. **Guerrero, Zorilin Alonso y González, Genry Leyva.** *Sistema de Captura y Transcripción de Audio.* Facultad 9, UCI. 2009.

18. **Suehring, Steve.** *Mysql Bible.* New York : John Wiley & Sons, 2002. Vol. 1. ISBN/0764549324.

19. **Hogan, Brian P.** *Hogan B.P. HTML5 and CSS3: Develop with Tomorrow's Standards Today.* s.l. : Pragmatic Programmers, 2011. ISBN/1-934356-68-9.

20. **Wright, Tim.** *Learning JavaScript: A Hands-On Guide to the Fundamentals of Modern.* 2012. Vol. 1. ISBN/978-0321832740.

21. **Ibargollín, Neybel González.** *Módulo de Análisis para el Sistema de Video Vigilancia Suria en Qt.* Facultad 6, UCI. La habana : s.n., 2012.

22. **BÖCK, Heiko.** The Definitive Guide to NetBeans Platform. New York : Springer-Verlang New York, 2009.

23. **Kuan, Joe.** Learning Highcharts. 2012.

24. **Chaffer, Jonathan y Swedberg, Karl.** Learning jQuery : Better Interaction Design and Web Development with Simple JavaScript Techniques. 2007.

25. **Vukotic, Aleksa y Goodwill, James.** *Apache Tomcat 7.* s.l. : Springer Verlag GmbH, 2011. ISBN/1430237236.

26. **Larman, Craig.** *UML y PATRONES. Introducción al análisis y diseño orientado a objeto.* 2008.

27. **PRESSMAN.** Ingeniería del Software: Un enfoque práctico. 2005. [En línea] 2005.

28. **Pavon Mestras, Javier.** *Patrones de Diseño Orientado a Objetos.* 2004.

29. www.si3ea.gov.co. [En línea] Libertad y orden UPME, COLCIENCIAS Colombia. [Citado el: 6 de 10 de 2014.] http://www.si3ea.gov.co/Portals/0/Gie/Tecnologias/factor.pdf.

30. **Sommerville, Ian.** *Ingeniería de software.* 2005.

31. **Pressman.** *Ingeniería de requisitos parte 1.*

32. **Larman, Craig.** *UML y Patrones: Una introducción al desarrollo Orientado a Objetos. Análisis y Diseño. 2004.* 2004. ISBN 0-13-148906-2.

33. **Pressman.** *Ingeniería de Requistos Parte 1, Cap 07.* 2002.

34. **Sommerville, Ian.** *Ingeniería de software .* 2005.

35. **Pressman, Roger.** *Software Engineering: A Practitioner´s Approach.* New York : s.n., 2010.

36. —. Capítulo 8: Modelado del Análisis. *Ingeniería de Software.* New York : s.n., 2008.

37. —. *Ingeniería de Software. Un enfoque práctico. 5ta Edición.* México : s.n., 2002.

38. **Pressman, Roger S.** *Ingeniería del Software.* 2005.

39. **Larman, Craig.** *UML y Patrones Introducción al análisis y diseño orientado a objetos.* México : PRENTICE HALL, 1999.

40. **Pressman, Roger S.** Ingeniería de Software. Un enfoque práctico. *Ingeniería de Software. Un enfoque práctico.* s.l. : Mc Graw Hill.

41. **Pressman, Roger.** *Ingenieria de Software: Un enfoque práctico.* 2006.

42. —. *Ingeniería del Software.* 2005.

43. **W3C.** W3C. *W3C.* [En línea] 2013. [Citado el: 25 de Marzo de 2014.] http://www.w3.org/standards/webdesign/audiovideo.

44. **W3CEspaña.** W3CEspaña. [En línea] 2014. [Citado el: 5 de enero de 2014.] http://www.w3c.es/.

45. **Sommerville, Ian.** *Ingeniería de software. Septima edición.* 2005.

46. **Pressman, Roger S.** *Ingenieria de Software. Un enfoque Práctico.* 2010.

Anexos

Anexo 1- Guía de observación

Soy la estudiante que cursa la carrera de Ingeniería en Ciencias Informáticas que se estudia en la UCI, realizo un proyecto de investigación en el que usted forma parte de la población analizada. Considero que tiene información importante para el desarrollo de este trabajo por lo que solicito su colaboración y garantizo el carácter confidencial de sus respuestas.

Gracias por adelantado.

_____.

Objetivo: Conocer los elementos que intervienen en la gestión de la información del consumo energético en la UCI, haciendo uso del método empírico observación.

1. ¿Cómo se gestiona en la actualidad el consumo energético en la UCI?

__Manual. ___Digital. ___Otros.

Especificar para otros.

_____
____.

2. Explique brevemente el proceso de gestión de la información del consumo energético en la UCI.

_____
_____
_____
_____
_____
_____.

3. Qué mecanismos son utilizados para medir el consumo energético en la UCI.

Especificar mecanismo(s)

_____
_____
_____
_____
_____
_____.

4. Cuántos dispositivos eléctricos de medición existen en la UCI.

5. Cree usted que sea necesario desarrollar un sistema para contribuir a la gestión de la información del consumo energético.

Si _____. No _____.

Explique.

_____
_____
_____
_____
_____
_____.

Anexo 2- Entrevista a especialista del grupo de Atención a Programas Energéticos de la UCI.

Soy la estudiante que cursa la carrera de Ingeniería en Ciencias Informáticas que se estudia en la UCI, realizo un proyecto de investigación en el que usted forma parte de la población analizada. Considero que tiene información importante para el desarrollo de este trabajo por lo que solicito su colaboración y garantizo el carácter confidencial de sus respuestas.

Gracias por adelantado.

_____.

Objetivo: Gestionar en tiempo real la información del consumo energético de la UCI mediante una aplicación Web.

1. ¿Cómo se gestiona en la actualidad el consumo energético en la UCI?

__Manual. ___Digital. ___Otros.

Especificar para otros.

_____
____.

2. Explique brevemente el proceso de gestión de la información del consumo energético en la UCI.

_____
_____
_____
_____
_____
_____.

3. Qué mecanismos son utilizados para medir el consumo energético en la UCI.

Especificar mecanismo(s)

_____
_____
_____
_____
_____
_____.

4. Cuántos dispositivos eléctricos de medición existen en la UCI.

5. Cree usted que sea necesario desarrollar un sistema para contribuir a la gestión de la información del consumo energético.

Si _____. No _____.

Explique.

_____
_____
_____
_____
_____
_____.