Universidad de las Ciencias Informáticas

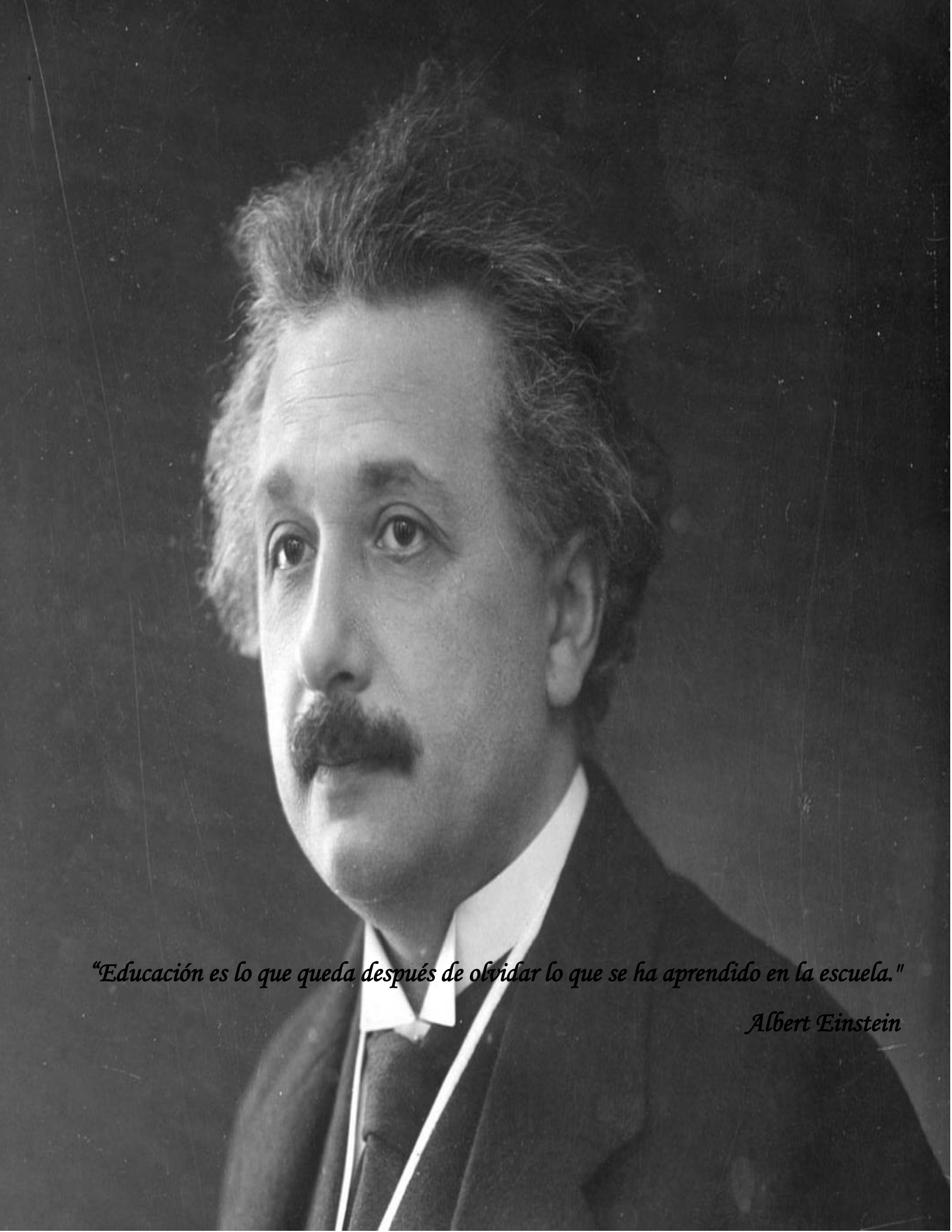
Facultad 6

Título: Sistema Integrado de Soporte del Centro DATEC

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

Autor(es): Danisleidys Aldana Kindelán
Roilan Navarro Orduñez
Tutor(es): Ing. Niurka Martínez Durán
Ing. Alexander Delgado Gutiérrez

La Habana, Cuba
Junio 2014
"Año 56 de la Revolución"



"Educación es lo que queda después de olvidar lo que se ha aprendido en la escuela."

Albert Einstein

DECLARACIÓN DE AUTORÍA

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Danisleidys Aldana Kindelán

_____

Firma del Autor

Roilán Navarro Orduñez

_____

Firma del Autor

Ing. Niurka Martínez Durán

_____

Firma del Tutor

Ing. Alexander Delgado Gutiérrez

_____

Firma del Tutor

DATOS DE CONTACTO

Tutores:

Ing. Niurka Martínez Durán

Especialidad de graduación: Ingeniero en Ciencias Informáticas

Años de graduado: 6

Correo Electrónico: nduran@uci.cu

Ing. Alexander Delgado Gutiérrez

Especialidad de graduación: Ingeniero en Ciencias Informáticas

Años de graduado: 2

Correo Electrónico: adgutierrez@uci.cu

AGRADECIMIENTOS

Agradecimientos de Danisleidys

Hoy se cumplen no solos mis sueños, también los sueños de aquellos que me vieron crecer, reír y llorar, en este día tan maravillo quiero dedicarle este trabajo de diploma especialmente a mi madre ***Damaris*** *por demostrarme fortaleza cuando más lo necesitaba y por ser el ejemplo de mujer luchadora que todos quieren tener, a mi abuela* ***Damacia Belén*** *por ser la voz de mi conciencia, por mimarme y ser más que una madre para mí. También quiero dedicarles este trabajo a mi padre* ***Yuri****, a mi tío* ***Tomas*** *y a mi abuelo* ***Juan Kindelán****. Le agradezco al* ***señor*** *por probarme su existencia y su grandeza.*

A los tutores ***Niurka*** *y* ***Alexander*** *por su total entrega y dedicación hacia nosotros. Al profesor* ***Hugo*** *por ser mi guía en esta universidad. A mis amistades* ***Ana****,* ***Elena*** *y* ***Ramoncito*** *por los momentos felices de mi vida. A mi dúo de tesis* ***Roilán*** *por su dedicación y entrega en la meta que nos propusimos. A* ***Abel*** *por ser mi paño de lágrimas y ser un buen compañero. A todos los* ***profesores*** *y* ***amistades*** *que de una forma u otra han contribuido para llegar a la meta propuesta.*

Agradecimientos de Roilán

Primeramente quiero agradecerle a una persona muy especial, por todo su amor y cariño, por su apoyo incondicional en todas mis decisiones y por haber echo de mí lo que ahora soy.

Gracias por todo, gracias por existir, gracias ***Ada Iris****, mi mamá. A todos aquellos que conocí a los largo de la carrera, a los que están aquí presentes y a los que no, en especial a* ***Reynaldo****,* ***Yoelkis****,* ***Jaciel*** *por haber compartido estos años juntos y haber pasado momentos inolvidables.*

A mis tutores ***Niurka*** *y* ***Alexander*** *por dedicarnos su tiempo y por sus consejos. A mi dúo de tesis* ***Danisleidys*** *por su preocupación y ayudarme a realizar esta meta que nos hemos propuesto.*

DEDICATORIA

Dedicatoria de Danisleidys

*Les dedico este trabajo de diploma especialmente a mi **madre** y a mi **abuela** por ser las mejores personas del mundo, la luz que me ha guiado siempre, por ser el brazo, la fortaleza y la destreza que a mí me falta. Agradezco haberlas conocido y haber nacido en su seno.*

Dedicatoria de Roilán

*Les dedico el presente trabajo a toda mi **familia** por el gran apoyo que me han brindado, en especial a mi **mamá**, mis **hermanos** y a mis **abuelos** que son mi razón de ser.*

RESUMEN

Para una mejor gestión del área de Tecnología que atiende el grupo de Soporte del Centro de Tecnologías de Gestión de Datos (DATEC) se creó el Sistema para la Gestión de Información de Incidencias, Descargas y Recursos del Centro de Tecnologías de Gestión de Datosv2.0. Actualmente las necesidades del grupo han cambiado debido a la reestructuración de sus áreas. El presente trabajo de diploma tiene como objetivo desarrollar un Sistema Integrado de Soporte del Centro DATEC. La solución propuesta garantiza una mayor estabilidad del área de Tecnología e integra las otras áreas del grupo del grupo de Soporte como la de Seguridad Informática y Control de Activos Fijos propiciando una mejor calidad en los servicios ofrecidos. Como resultado se obtuvo un Sistema Integrado de Soporte del Centro DATEC que mantiene los beneficios obtenidos en el sistema anterior incorporando nuevas funcionalidades como la gestión de la clasificación de útil y gestión de la clasificación de Activo Fijo Tangible permitiendo realizar una gestión más amplia sobre los recursos ubicados en los locales. Realiza la gestión dinámica de las prestaciones de los activos tecnológicos permitiendo una mayor rapidez en la obtención de las mismas. Permite la realización de auditorías de seguridad informática a los recursos de cómputo de acuerdo a las normas regidas por la universidad y el centro.

Palabras Claves:

Activo Fijos, Seguridad Informática, Gestión de Información.

ABSTRACT

For better management of the area served by the group of Technology Support Center Data Management Technologies (DATEC) System for Information Management Issues, Downloads and Resources Center Management Technology was established Datosv2.0 . Currently the group's needs to have changed due to the restructuring of their areas. This diploma work has the objective to develop an Integrated Support Centre DATEC. The proposed solution ensures greater stability in the area of technology and integrates other areas of the group Support group as Information Security and Control of Fixed Assets promoting a better quality of services offered. As a result an Integrated Support Centre DATEC maintaining the benefits achieved in the previous system incorporating new features such as the management of the classification and management of useful classification of Tangible Fixed Assets allowing a wider management of resources was obtained located on the premises. Making the dynamic management of the performance of the technology assets enabling faster in obtaining them. Let’s give you information security audits to computing resources according to the rules governed by the university and center.

Keywords:

Fixed Assets, Information Security, Information Management.

TABLA DE CONTENIDO

TABLA DE CONTENIDO

INTRODUCCIÓN

La innovación y uso de las Tecnologías de la Información y las Comunicaciones (TIC) es un proceso acelerado en el desarrollo científico técnico a escala mundial. De ellas depende en gran medida el desarrollo de la economía de los países en el presente siglo. Se puede decir que las TIC no son más que aquel conjunto de tecnologías que permiten transmitir, procesar y difundir información de manera constante. Además son consideras las bases para reducir la brecha digital sobre la cual se construye una sociedad de información y economía del conocimiento.

El uso de las TIC en Cuba es un proceso priorizado para satisfacer las necesidades actuales de información. Además de lograr un continuo avance, posibilita una mejora constante en los sectores de la economía y la sociedad. Esto ha provocado una acertada transformación cuyas ventajas expresan el ahorro de recursos, la comunicación y la actualización de la información. Enmarcado al proceso de informatización de la sociedad se encuentran la mayor parte de las instituciones académicas y de investigación entre las que se encuentra la Universidad de la Ciencias Informáticas (UCI).

La UCI vincula el proceso docente-educativo con la producción de sistemas informáticos. En correspondencia con esto la universidad dedica prolongados esfuerzos a la distribución de sus recursos. La misma está conformada por una estructura organizativa fragmentada en áreas por objetivos, una de estas áreas es la encargada de la producción de software, que agrupa a los centros productivos entre los que se encuentra el Centro de Tecnologías de Gestión de Datos (DATEC).

El centro DATEC es el encargado de proveer soluciones integrales, productos y servicios relacionados con las tecnologías de gestión de datos, permitiendo la formación y capacitación de profesionales integrales con un alto nivel científico-productivo para el desarrollo de investigaciones afines a él. Esta entidad posee varios departamentos: Almacenes de Datos, PostgreSQL e Integración de Soluciones y a su vez tiene asociado a él tres grupos de trabajo: Calidad, Mercadotecnia y Soporte.

El Grupo de Soporte se encarga de los procesos encaminados a la gestión de los recursos del centro para la realización de software. Este grupo de trabajo actualmente gestiona las áreas de Tecnología, Seguridad Informática y Control de Activos Fijos. El área de Tecnología se encarga de gestionar los recursos informáticos de la entidad, sus prestaciones y cómo están distribuidos por locales y por puestos de

trabajo. El área de Seguridad Informática se enfoca en la protección de la infraestructura e información computacional guiándose por las normas de seguridad regidas por el centro y la universidad, algunas de estas son la instalación de antivirus, cortafuegos y el control de acceso de los usuarios. Por último, el área Control de Activos Fijos controla los medios básicos (MB) asociados a los locales de la entidad así como la realización de sus movimientos.

Para tener el control del área de Tecnología se realizó el Sistema para la Gestión de Información de Incidencias, Descargas y Recursos del Centro de Tecnologías de Gestión de Datosv2.0. Este sistema cumplió con las necesidades para lo que fue creado brindándole al grupo un conjunto de requisitos útiles para el desempeño de sus actividades. A pesar de las funcionalidades que este sistema brinda, posee la limitante de solo gestionar el área de Tecnología dejando fuera de sus alcances las otras áreas que administra el Grupo de Soporte.

En el área de Tecnología en la cual el sistema centra todas sus funcionalidades, la actualización de las prestaciones que brindan los recursos de cómputo es introducida por el usuario, resultando este proceso muy extenso debido a la cantidad de recursos existentes. Los roles asignados a los usuarios poseen permisos invariables, impidiendo el proceso de conceder o denegar un permiso una vez establecidos los mismos. No se generan ni se actualizan expedientes técnicos como una nueva regla de negocio a tener en cuenta. Este sistema carece de un Control de Activos Fijos debido a que los movimientos de los recursos son registrados en Excel posibilitando pérdidas de información. Además para la realización de las auditorías informáticas se necesita un personal encargado que ejecute scripts sobre las computadoras y recopile la información de estas.

DATEC crea servicios informáticos que agrupa tanto a los sistemas de información, como a los denominados sistemas de inteligencia empresarial o de negocios, cuyo propósito fundamental es apoyar el proceso de toma de decisiones de las diferentes entidades. Debido a esto, este centro mantiene estrecha colaboración con importantes empresas de alto nivel nacional e internacional entre las cuales se encuentra el Ministro de Salud Pública, Oficina Nacional de Estadísticas e Información, Ministerio del Comercio Exterior de Cuba, Unión de Jóvenes Comunistas entre otras. Esto requiere fundamentalmente que la distribución de los recursos de cómputo sea acorde a las prioridades que se presenten para satisfacer de manera gradual las demandas realizadas al centro.

INTRODUCCIÓN

Por todo lo expuesto anteriormente se define como **problema de investigación** ¿Cómo satisfacer las necesidades actuales del grupo de Soporte perteneciente al centro DATEC? La investigación tiene como **objeto de estudio:** Sistemas de gestión de información. Centrado al **campo de acción:** Sistemas de gestión de información para los servicios de soporte**.** Para darle solución a la problemática planteada se plantea como **objetivo general:** Desarrollar el Sistema Integrado de Soporte para obtener una gestión de información general del Grupo de Soporte del Centro DATEC. Desglosado en los siguientes **objetivos específicos:**

- ✓ Analizar las tecnologías y tendencias actuales referentes al desarrollo de Sistemas de Gestión de Información.
- ✓ Fundamentar la selección de la metodología, herramientas y tecnologías a utilizar en el desarrollo del Sistema Integrado de Soporte del Centro DATEC.
- ✓ Diseñar la propuesta del Sistema Integrado de Soporte del Centro DATEC.
- ✓ Implementar la propuesta del Sistema Integrado de Soporte del Centro DATEC.
- ✓ Validar el Sistema Integrado de Soporte del Centro DATEC.

Para darle cumplimiento a los objetivos plantados se definieron las siguientes **tareas de la investigación:**

- ✓ Análisis de los Sistemas de Gestión para la Información existentes en el mundo.
- ✓ Análisis del Sistema para la gestión de Información, Incidencias, Descargas y Recursos del Centro de Tecnologías de Gestión de Datos.
- ✓ Análisis de los procesos vinculados al Sistema Integrado de Soporte del Centro DATEC.
- ✓ Definición de las características del sistema a implementar.
- ✓ Rediseño del sistema a implementar.
- ✓ Rediseño de la interfaz gráfica del sistema a implementar.
- ✓ Implementación del sistema rediseñado en la capa de presentación de datos.
- ✓ Implementación del sistema rediseñado para satisfacer los requisitos funcionales.
- ✓ Realización de pruebas de caja negra al sistema implementado.
- ✓ Realización de pruebas de carga al sistema implementado.
- ✓ Realización de pruebas de stress al sistema implementado.

Para guiar la investigación se exponen las siguientes **preguntas científicas**:

- ✓ ¿Cómo se realiza el proceso de gestión de información?
- ✓ ¿Qué entorno de desarrollo requiere el sistema a implementar?
- ✓ ¿Existen funcionalidades acordes a las actuales exigencias?
- ✓ ¿Qué se puede hacer para cumplir con las necesidades del Grupo de Soporte?
- ✓ ¿Se ajustan las funcionalidades implementadas a las demandas del Grupo de Soporte?

Para llevar a cabo este trabajo de diploma se definieron los siguientes **métodos de investigación científica**:

Analítico-Sintético: El análisis permite la división mental del fenómeno en sus múltiples relaciones y componentes para facilitar su estudio y la síntesis establece mentalmente la unión de las partes previamente analizadas. Se utiliza para el análisis de documentos, con el fin de extraer los elementos más importantes relacionados con los Sistemas de Gestión de Información, para sintetizar y mostrar los resultados obtenidos.
Histórico-Lógico: Presuponen el estudio detallado de todos los antecedentes, causas y condiciones históricas en que surgió y se desarrolló un objeto o proceso determinado. Se utiliza para actualizar la evolución de los Sistemas de Gestión de Información existentes en el mundo, así como las limitantes y logros en Cuba.
Modelación: Es el método mediante el cual se crean abstracciones para representar la realidad compleja y ofrece información sobre el objeto que se estudia. Se utiliza para la representar los diagramas tales como el modelo de dominio, clases de diseño, casos de uso, modelo de datos, componentes y despliegue para un mejor desarrollo del sistema.

El presente trabajo de diploma está desglosado en los siguientes capítulos:

Capítulo 1: Fundamentos teóricos.
En el capítulo se describen los conceptos y definiciones fundamentales respecto a los Sistemas de Gestión de Información (SGI). Se realiza un análisis de sistemas existentes en el mundo y en Cuba orientados a los SGI, caracterizando fundamentalmente los componentes de la versión 2.0 del Sistema para la gestión de información de incidencias, descargas y recursos de DATEC. Además se fundamenta el uso de las distintas herramientas, tecnologías y metodología utilizadas para el desarrollo del sistema propuesto.

Capítulo 2: Características y diseño del sistema.

En el presente capítulo se describe el análisis y diseño del Sistema Integrado de Soporte del Centro DATEC. Se realiza el modelo de dominio y el levantamiento de requisitos funcionales y no funcionales para lograr el sistema que cumpla con las actuales exigencias del grupo de Soporte. Mediante el diagrama de caso de uso se representa la relación de los actores con los mismos y además se describe el caso de uso más significativo. Se brinda un acercamiento a la implementación del sistema generando los diagramas de interacción y modelo de clases siguiendo el estilo arquitectónico y los patrones de diseño seleccionados. Se muestra el modelo de datos con la descripción de las tablas más significativas.

Capítulo 3: Implementación y pruebas del sistema.

En el presente capítulo se describe cómo los elementos del modelo del diseño se implementan en términos de componentes y se muestra una descripción detallada de los paquetes de implementación. Se presenta el mapa de navegación del software desarrollado y algunas imágenes del mismo para brindar una mejor comprensión del sistema. Posteriormente se procede a la validación de la aplicación mediante las pruebas de software de caja negra, carga y stress, para comprobar la operatividad de las principales funcionalidades. Y finalmente se muestra el diagrama de despliegue, con una breve descripción de los nodos que lo conforman.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

Introducción

En el capítulo se describen los conceptos y definiciones fundamentales respecto a los Sistemas de Gestión de Información (SGI). Se realiza un análisis de sistemas existentes en el mundo y en Cuba orientados a los SGI, caracterizando fundamentalmente los componentes de la versión 2.0 del Sistema para la gestión de información de incidencias, descargas y recursos de DATEC. Además se fundamenta el uso de las distintas herramientas, tecnologías y metodología utilizadas para el desarrollo del sistema propuesto.

1.1 Sistemas de gestión para la información (SGI) existentes en el mundo

La gestión de la información se puede identificar como la disciplina que se encarga de todo lo relacionado con la obtención de la información adecuada enmarcada a maximizar los beneficios derivados del uso de la información y minimizar el coste de adquisición. (1) La misma genera nuevos conocimientos que posibilitan calidad y eficiencia en los servicios y productos brindados. Sirven además para integrar o centralizar la gestión de la información dentro de una empresa y sostienen todos los procesos de negocio y de soporte de la organización.

La investigación sobre los SGI mostró que en el mundo existe una fuerte inclinación de su utilización por estar orientados a varios ámbitos sociales. A continuación se analizan algunas soluciones existentes vinculadas a los servicios de soporte:

SISE

El proyecto Sistema Integrado De Soporte Especializado (SISE v. 1.0.1) surge de las necesidades y dificultades que se presentan en la Empresa Procesos y Servicios S.A.S en Colombia. En este sistema se automatiza la administración de peticiones de ayuda técnica. Además implementa una guía para la solución de problemas a los activos tecnológicos de la compañía y de esta manera obtener la satisfacción de los clientes internos y externos, logrando así un mejoramiento en los índices de productividad y en el control de los procesos internos que lleva la organización.

SISE cuenta con el apoyo tecnológico de procesos y servicios S.A.S, compañía dedicada a la administración, procesamiento y gestión de la información en varias compañías importantes del medio. Este presenta el servicio de externalización en sus procesos de digitalización, gestión documental, interventoría en procesos, conciliación documental bancaria, gestión y depuración de bases de datos, radicación y captura de citas médicas. A pesar de ser una compañía líder en su medio y constante crecimiento no cuenta con la manera adecuada de realizar los procedimientos de soporte técnico interno y para mejorar sus políticas de calidad. Requiere de un aplicativo tipo escritorio que gestione dichos procesos y preste el servicio técnico de una manera integrada. SISE busca integrar los procesos tipo Mesa de Apoyo integrando un inventario de los activos tecnológicos con los que cuenta la compañía. (2)

El sistema implementa servicios útiles para la empresa para la que fue desarrollada pero el mismo no resulta ser una solución viable debido a que no se ajusta a las necesidades del grupo de Soporte. Sus funcionalidades para la gestión de los recursos tecnológicos son escasas ya que solo gestiona las incidencias y los reportes de estas. Entre sus funcionalidades no se reflejan el área de Seguridad Informática ni el Control de Activos Fijos. Además es un software propietario, lo cual genera gastos en actualizaciones, suplementos, servicios basados en internet y soporte técnico.

GESIMED

Sistema de información interactivo para la red de bibliotecas médicas en Villa Clara surgió por no poseer un sistema que agrupe, organice, gestione, controle y divulgue la información dentro de la organización. Este sistema permite:

- ✓ El control del registro de usuarios.
- ✓ El control de acceso a los gestores de información.
- ✓ Almacenamiento de los productos informativos garantizando la integridad y privacidad.

El sistema vela por el funcionamiento optimizando y perfeccionando sus componentes, teniendo en cuenta los usuarios, el ambiente, sus necesidades y sus solicitudes. A través del mismo se optimizan los servicios de información especializados de toda la red incrementando su fluidez, facilitando al usuario la información que le pueda ser útil, aumentando así la calidad y efectividad de los servicios con un mayor ajuste creando productos informativos con un alto valor agregado.

Es un sistema de información que da la posibilidad de buscar, localizar, procesar, almacenar y difundir la información a través de los procesos automatizados. Su diseño y sus procesos están enfocados en primer plano al usuario. Es proveedor de fuentes de información tanto externas como internas. Cuenta con un sistema de reportes estadísticos tanto para medir la usabilidad del sistema GESIMED, como cantidad de usuarios atendidos por categorías y la cantidad de artículos y productos informativos que se ofertan al usuario en un tiempo determinado. (3)

Este sistema es muy práctico para la gestión de las redes de bibliotecas medicas de Villa Clara pero en el no existen funcionalidades viables para lo que se quiere desarrollar debido a que están encaminadas a un área de negocio diferente y su posible adaptación seria costoso en tiempo. Además posee una gestión parcial de los usuarios por lo que existen usuarios navegando en la aplicación sin ser autenticados. Sus funcionalidades no están enmarcadas los recursos tecnológicos de la entidad ni a la seguridad de los mismos.

CEDRUX

El Sistema Integral de Gestión (CEDRUX) es un sistema de gestión empresarial que se desarrolló en la UCI. Cuenta con un subsistema para el control de los recursos de cómputo que pretende servir de apoyo a la toma de decisiones en las entidades. El subsistema es capaz de gestionar recursos de forma dinámica, tributa mediante operaciones al sistema contable de la empresa. Incluye además temas como la agrupación de recursos y la generación de documentos. Entre las funcionalidades de CEDRUX está la de centralizar la información de la contabilidad de la empresa lo que posibilita una mejor administración, gestión, agilización de procesos, optimización del tiempo y control sobre los recursos. (4)

A pesar de las funcionalidades que brinda este no resulta ser una solución factible para lo que se necesita. Aunque mantiene una gran administración de sus recursos, esta administración está encaminada a la depreciación monetaria de los mismos, funcionalidad que en la actualidad no es útil para lo que se quiere desarrollar. Por su pesada arquitectura es imposible dividir las funcionalidades que son necesarias e incorporarles otras, y una posible disolución de las mismas contribuiría a un gasto de recursos y tiempo mayor que los beneficios que contraería.

1.2 Sistema para la gestión de información de incidencias, descargas y recursos del Centro de Tecnologías de Gestión de Datos v2.0

Este sistema surgió para ayudar a la toma de decisiones de los directivos del Centro de Tecnologías de Gestión de Datos. Esta aplicación permite la gestión de departamentos, proyectos, locales, recursos. En la gestión de locales se encuentra asociado el campo gestión de categoría de local que es concebido para administrar las categorías que se tienen en el centro. También posibilita la gestión de los usuarios y sus categorías, puestos de trabajo y usuarios por puesto de trabajo permitiendo realizar una gestión más amplia sobre los recursos ubicados en los locales y proyectos del usuario.

Este sistema posee otras funcionalidades como la gestión de incidencias por puesto de trabajo, solicitud de descarga personal y el estado de las incidencias donde estas se pueden administrar o reportar. Para una mejor seguridad los usuarios del sistema también tienen la opción de cambiar sus contraseñas y se emplea la política de control de acceso basado en roles. Además los usuarios pueden optar por una solicitud de descargas y ver el estado en que estas se encuentran. Gestionar un nuevo recurso adicional correspondiente a un local, la visualización de trazas y de reportes son también actividades contempladas en él. (5)

Este sistema cumple con las pautas para la cual fue desarrollado, es un sistema extensible al cual se le pueden agregar varios módulos, pero debido a que el Grupo de Soporte ha ampliado su marco de trabajo este presenta algunas limitaciones que imposibilitan la satisfacción de sus necesidades actuales. Aunque resulta ser una solución útil para los directivos del centro solo contempla el área de Tecnología. Además entre las funcionalidades de este producto no controla las medidas de seguridad regidas por la universidad, posee un control parcial de sus recursos ya que no controla los medios básicos de la entidad y la actualización de las computadoras es entrada por el usuario.

Luego de un estudio realizado a las soluciones existentes se concluye que estos sistemas no son factibles para lo que se quiere desarrollar. Entre sus funcionalidades no se encuentran la generación y actualización de expedientes técnicos, una nueva regla del negocio a tener en cuenta. Muchos no abarcan el control de los activos tecnológicos ni automatizan las prestaciones que estos brindan. Algunos se crearon bajo la licencia de software privativo lo cual genera un gasto económico y no permiten realizarles auditorías de seguridad informática a las computadoras. Una vez planteadas estas inconvenientes se decidió realizar un

sistema que represente las necesidades actuales del centro tomando como base la filosofía y parte de la arquitectura del Sistema para la Gestión de información de incidencias, descargas y recursos del Centro de Tecnologías de Gestión de Datos v2.0, sin realizar una nueva versión del mismo debido a que el negocio ha cambiado significativamente.

1.3 Metodología, tecnologías y herramientas.

A continuación se presentan las descripciones de la metodología, herramientas y tecnologías utilizadas para dar solución al problema. Para definir la selección se tomaron en cuenta las necesidades existentes, tendencias actuales, el entorno donde se desplegará el sistema y la arquitectura implementada en la versión 2.0 del Sistema para la gestión de información de incidencias, descargas y recursos del Centro de Tecnologías de Gestión de Datos.

1.3.1 Metodología de desarrollo de software

Las metodologías de desarrollo de *software* son un conjunto de procedimientos, técnicas, herramientas y soporte documental para el desarrollo de un producto (*software*). Se basan en una combinación de los modelos de proceso genéricos (cascada, evolutivo e incremental). Son para estructurar, planear y controlar el proceso de desarrollo del *software*. Constituyen una guía que define las tareas y actividades que se deben realizar para obtener un *software* de buena calidad. (6)

Metodologías ágiles

Las metodologías ágiles son orientadas a la interactuación con el cliente y el desarrollo incremental del *software*. Mostrando versiones parcialmente funcionales de la aplicación al cliente en intervalos cortos de tiempo, para que pueda evaluar y sugerir cambios en el producto según se va desarrollando. Se enmarca más en la capacidad de respuesta ante un cambio realizado que en el seguimiento estricto de un plan. Consiste básicamente en ajustarse estrictamente a una serie de reglas que se centran en las necesidades del cliente para lograr un producto de buena calidad en poco tiempo. Entre ellas se destacan: *XP (eXtreme Programming), Scrum, ASD (Adaptive Software Development) y OpenUP*. (7)

OpenUP

OpenUP es una variante ágil del Proceso Unificado que aplica métodos iterativo e incremental dentro de un ciclo de vida estructurado y está orientado a equipos de desarrollo pequeños. Abraza una filosofía pragmática y ágil que se centra en la naturaleza de colaboración de desarrollo de *software*. Es un proceso

de desarrollo iterativo del *software* que es mínimo, completo, y extensible. El proceso es mínimo cuando solamente el contenido fundamental es incluido; es completo cuando puede ser manifestado como todo el proceso para construir un sistema y es extensible cuando puede ser utilizado como fundamento sobre el cual el contenido de proceso se pueda agregar o adaptar según lo necesitado. *OpenUP* como metodología de desarrollo es conducida por el principio de colaboración para alinear intereses y para compartir su comprensión. Se presenta la metodología *OpenUP* por las siguientes características:

Fases del ciclo de vida de OpenUP:

- ✓ Concepción: Primera de las 4 fases en el proyecto del ciclo de vida, acerca el entendimiento del propósito y los objetivos además de obtener suficiente información para confirmar que el proyecto debe hacer. El objetivo de esta fase es capturar las necesidades de los stakeholder en los objetivos del ciclo de vida para el proyecto.
- ✓ Elaboración: Es la segunda de las 4 fases del ciclo de vida del *OpenUP* donde se trata los riesgos significativos para la arquitectura. El propósito de esta fase es establecer la base la elaboración de la arquitectura del sistema.
- ✓ Construcción: Esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El propósito de esta fase es completar el desarrollo del sistema basado en la arquitectura definida.
- ✓ Transición: Es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, y evalúa la funcionalidad y performance del último entregable de la fase de construcción. (5)

Concluyendo lo anteriormente expuesto, se decide seleccionar a *OpenUP* para enfrentar el desarrollo del *software* propuesto debido a que permite detectar errores con prontitud a través de un ciclo iterativo y tiene un enfoque orientado al cliente con iteraciones cortas. Esta también es apropiada para proyectos pequeños y de bajos recursos, permite disminuir las probabilidades de fracaso en los proyectos pequeños e incrementar las probabilidades de éxito. Además es la metodología que se utilizó en el sistema anterior, el cual se va a tomar como base.

1.3.2 Lenguaje de modelado de sistema.

Los lenguajes de modelados como su nombre lo indica son lenguajes que transmiten de forma gráfica un mensaje o una idea a través de gráficos, tablas y documentaciones. Estos generan artefactos que pueden

ser representaciones gráficas o documentos explicativos. En la siguiente sección se hará referencia a UML como lenguaje de modelado.

UML

El Lenguaje de Modelado Unificado (UML), es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

UML sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas *software* como para la arquitectura *hardware* donde se ejecuten. Otro objetivo de este modelado visual es que sea independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML se puedan implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos).

UML es además un método formal de modelado. Esto aporta las siguientes ventajas:

- ✓ Mayor rigor en la especificación.
- ✓ Permite realizar una verificación y validación del modelo realizado.
- ✓ Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener la visión en el diseño, de más alto nivel, de la estructura de un proyecto.
- ✓ Se puede usar para diferentes tipos de sistemas.
- ✓ Consolida muchas de las notaciones y conceptos más usados orientados a objetos.
- ✓ Es fácilmente entendible. (8)

Considerando lo anteriormente planteado se decidió utilizar el UML debido a que es el lenguaje de modelado de sistemas de *software* más conocido y utilizado en la actualidad. Modela estructuras complejas, reemplaza a decenas de notaciones empleadas con otros lenguajes y las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.

1.3.3 Herramienta CASE

Las herramientas CASE son muy utilizadas en la actualidad porque son aplicaciones que ayudan a aumentar la productividad del desarrollo de *software* reduciendo costo, tiempo y dinero de los sistemas en confección. Con estos tipos de herramientas se puede diseñar, implementar a partir de un diseño realizado, compilar automáticamente, detectar errores y brindarle la seguridad al equipo de trabajo sobre el avance de la solución.

Visual Paradigm

Visual Paradigm es una herramienta CASE que permite construir el diagramado UML, como son los flujos de eventos del sistema, las clases, todo lo que es documentación tanto de desarrollo como procesos de negocio. La versión Visual Paradigm UML 6.4 *Community Edition*, alternativa libre y gratuita de este *software*, permite aumentar la calidad del *software*, a través de la mejora de la productividad en el desarrollo y mantenimiento del *software*. Aumenta el conocimiento informático de una empresa ayudando así a la búsqueda de soluciones para los requisitos.

Proporciona características tales como la ingeniería inversa, la generación de código y de informes. Permite invertir el código fuente de los programas, archivos ejecutables y binarios en modelos UML al instante, creando de manera simple toda la documentación. Otra de las ventajas que ofrece es la navegación intuitiva entre el modelo visual y el código, además de permitir la sincronización entre el código fuente y el modelo en tiempo real o bajo demanda. (9)

Teniendo en cuenta estas características se decide utilizar como herramienta CASE el Visual Paradigm 6.4, debido a que permite el modelado visual para todos los tipos de diagramas UML. Posibilita el uso de un lenguaje estándar común a todo el equipo de desarrollo facilitando la comunicación. Posee un modelo y código que permanece sincronizado en todo el ciclo de vida del desarrollo de software. Además permite realizar ingeniería inversa de bases de datos y es multiplataforma.

1.3.4 Lenguajes de programación.

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente. (10)

PHP

PHP o *Hypertext Pre Processor* es un lenguaje de programación de uso general de código del lado del servidor diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos.

Está especialmente diseñado para incrementar el dinamismo de las páginas web, aprovechando los recursos de las redes informáticas y los escasos requerimientos de *hardware* que solicita el lenguaje para que sus aplicaciones funcionen correctamente. Además, permite aplicar técnicas de programación orientada a objetos. No requiere definición de tipos de variables aunque se pueden evaluar por el tipo que estén manejando en tiempo de ejecución. El código fuente escrito en este lenguaje es invisible al navegador ya que es el servidor el encargado de ejecutar el código y enviar su resultado HTML al cliente, logrando una programación segura y confiable. Posee una amplia documentación, dado que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda. (11)

Por todo lo anteriormente analizado se seleccionó para la programación del lado del servidor el lenguaje de programación PHP5, debido a todas las facilidades de uso que brinda para la realización de sistemas sobre la web. Además su código fuente es invisible al navegador web y al cliente, ya que es el servidor el que se encarga de ejecutarlo y enviar el resultado al navegador. Es un lenguaje multiplataforma y posee la capacidad de expandir su potencial utilizando módulos.

JavaScript

JavaScript es un lenguaje de programación interpretado. Se define como orientado a objetos basado en prototipos débilmente tipados y dinámicos. Se utiliza principalmente en su forma del lado del cliente (*client-side*), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. Se usa en aplicaciones externas a la web, por ejemplo en documentos y aplicaciones de escritorio entre otras. Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del *Document Object Model* (DOM). (12)

Por todo lo anteriormente planteado en la programación del lado del cliente será utilizado el lenguaje Java Script ya que permite la creación de páginas web de forma dinámica además es un lenguaje que resulta fácil de aprender. Los programas escritos en este lenguaje se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. No es necesario declarar los tipos de variables que van a utilizarse y las referencias a objetos se comprueban en tiempo de ejecución.

Bash

Bash es un programa informático cuya función consiste en interpretar órdenes. Está basado en la shell de *Unix* y es compatible con POSIX (*Portable Operating System Interface*). Fue escrito para el proyecto GNU y es el intérprete de comandos por defecto en la mayoría de las distribuciones de Linux. Este permite utilizar todos los comandos del sistema, llamadas a otros script Bash, funciones internas, y rutinas en otros lenguajes. Estas características hacen que Bash sea una herramienta muy potente que nos permite evitar la realización de tareas repetitivas. (13)

Por lo anteriormente expuesto se utiliza Bash para el desarrollo de los scripts que se van a ejecutar en las estaciones de trabajo, debido a que permite ejecutar los scripts en los diferentes sistemas operativos Unix/Linux/Windows que posean las herramientas para interpretar este lenguaje.

1.3.5 Framework y librería de desarrollo

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un *framework* proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y fácil de mantener. Por último, un *framework* facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

Symfony

Symfony2 es un completo *framework* diseñado para optimizar el desarrollo de las aplicaciones web. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.

Symfony2.3 ha sido ideada para utilizar al máximo las nuevas características de PHP 5.3 o superior y por eso es uno de los *frameworks* PHP con mejor rendimiento. Su arquitectura interna está completamente desacoplada, lo que permite reemplazar o eliminar fácilmente aquellas partes que no encajen en los proyectos.

Algunas de las ventajas que Symfony brinda son:

- ✓ Fácil de instalar y configurar en la mayoría de plataformas.
- ✓ Independiente del sistema gestor de bases de datos.
- ✓ Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- ✓ Utiliza Doctrine2 como herramienta ORM (*Object Relational Mapper* - Mapeo Relacional de Objeto).
- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros. (14)

Analizando lo anteriormente planteado se decide utilizar para el desarrollo del sistema Symfony2.3, teniendo en cuenta las características expuestas, que sostienen una actualización tecnológica mayor en correspondencia con la implementación de la segunda versión.

ExtJS

Es una librería JavaScript de alto rendimiento, compatible con la mayoría de navegadores, usado para crear páginas web dinámicas. Este permite crear aplicaciones complejas utilizando componentes predefinidos.

Ventajas:

- ✓ Existe un balance entre Cliente – Servidor: La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo.
- ✓ Eficiencia de la red: El tráfico de red puede disminuir al permitir que la aplicación elija qué información desea transmitir al servidor y viceversa. (15)

Por todo lo antes expuesto se decide utilizar ExtJS 3.4.0 debido a que representan una estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación que ayuda a la rápida confección de sitios web.

Lycan

Es un IDE para el desarrollo de aplicaciones enriquecidas para la web. Facilita el desarrollo de componentes de una manera intuitiva, rápida y cómoda. Se reutiliza y configura para otros desarrollos más especializados en el propio dominio (como desarrollo de reportes o encuestas). Lycan es utilizado por un número creciente de desarrolladores de Ext JS en el entorno ideal de la UCI. Se ha optimizado para su ejecución en Mozilla Firefox navegador preferencial de la mayoría de sus desarrolladores. Los ciclos de desarrollo son variables durando entre 2 y 3 meses entre cada actualización y 6 meses entre cada liberación. (16)

Teniendo en cuenta lo anteriormente planteado se utiliza Lycan porque tiene mecanismos flexibles y escalables de extensión a nivel de plataforma. Además posee documentación de Ingeniería que permita la transferencia a una comunidad.

1.3.6 Servidor web

Los servidores web se encargan de contestar peticiones y entregar como resultado un documento HTML utilizando el protocolo HTTP. El funcionamiento de un servidor web está dado por: esperar peticiones en el puerto TCP indicado, recibir una petición, buscar el recurso y enviarlo utilizando la misma conexión por la que recibió la petición. Las peticiones son realizadas con el uso del protocolo HTTP y respondidas con códigos HTML que el cliente es el encargado de descifrar en fuentes, colores y formas exhibidas en dichas páginas.

Apache

Apache2.2 constituye una tecnología gratuita de código abierto. Es un servidor web flexible y rápido, continuamente actualizado y adaptado a los nuevos protocolos. Permite la creación de sitios web dinámicos mediante el uso de *Server Side Includes* (SSI) por sus siglas en inglés, de lenguajes de scripting como PHP, JavaScript, Python, entre otros. Es multiplataforma y posee una arquitectura modular que admite ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la API de programación de módulos, para el desarrollo de módulos específicos. Tiene una alta configuración en la creación y gestión de logs y soporta personalizar la respuesta ante los posibles errores que se puedan generar en el servidor. (17)

Partiendo de las características analizadas se resolvió usar para la implementación del sistema el servidor web Apache2.2 teniendo en cuenta además el lenguaje de programación web escogido para la programación del lado del servidor.

1.3.7 Servidor de base de datos

Los servidores de bases de datos surgen por la necesidad de las empresas de manejar grandes y complejos volúmenes de datos, al tiempo que requieren compartir la información con un conjunto de clientes (que pueden ser tanto aplicaciones como usuarios) de una manera segura. (18)

PostgreSQL 9.2

Es un sistema de gestión de base de datos relacional orientada a objetos, libre y publicado bajo la licencia BSD (*Distribución de Software Berkeley*). Es un gestor de bases de datos de código abierto avanzado, ofreciendo control de concurrencia multi-versión, soportando casi todas las sintaxis SQL como sub-consultas, transacciones y funciones definidas por el usuario, cuenta además con un amplio conjunto de enlaces con lenguajes de programación como C, C++, Java y Python . PostgreSQL está dirigido por una comunidad bien organizada de desarrolladores y organizaciones comerciales, de ahí el nivel de aceptación y de calidad del mismo.

Entre sus características se encuentran:

- ✓ Poseer una gran escalabilidad. Es capaz de ajustarse al número de Unidades Centrales de Procesamiento (*Central Processing Units*, CPU por sus siglas en inglés) y a la cantidad de memoria que posee el sistema de forma óptima.
- ✓ Tener la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.
- ✓ La simplificación del proceso de administración de licencias de software, que no es necesario cuando se usa software libre.
- ✓ Poder lidiar con grandes volúmenes de datos.

PostgreSQL 9.2 cuenta con soporte nativo para JSON (*JavaScript Object Notation*), que abarca los índices, las mejoras de replicación y de desempeño proporcionando un mecanismo eficaz para la creación y el almacenamiento de documentos para APIs web. Con la adición de una escalabilidad lineal de hasta 64 núcleos, las exploraciones de índices y las reducciones en el consumo de energía de la CPU hace

que esta versión haya mejorado significativamente la escalabilidad y la flexibilidad de desarrollo para las cargas de trabajo más exigentes. (19)

Teniendo en cuenta lo antes analizando, para el desarrollo del sistema se decidió utilizar el gestor de bases de datos PostgreSQL 9.2; por las particulares expuestas y además por la existencia en el centro de una línea de investigación relacionada con este gestor, que garantiza facilidades de experiencias y conocimientos que se pueden adquirir.

1.3.8 Entorno integrado de desarrollo

Para la ejecución del sistema se hizo necesario escoger un entorno integrado de desarrollo que proporcionara el uso y la integración de todas las tecnologías y lenguajes antes mencionados, por lo que se decidió utilizar el NetBeans IDE 7.4. Las mejoras realizadas a esta última versión en cuanto a la programación web pueden ser apreciadas en la siguiente descripción.

NetBeans IDE

NetBeans IDE 7.4 es un entorno modular basada en estándares de desarrollo integrado (IDE), escrito en el lenguaje de programación Java. El proyecto NetBeans consiste en un IDE de código abierto con todas las funciones escritas en el lenguaje de programación Java y una plataforma de aplicaciones de cliente enriquecido, que puede ser utilizado como un marco genérico para crear cualquier tipo de aplicación.
NetBeans IDE 7.4 amplía el soporte avanzado de desarrollo HTML5 para las aplicaciones Java EE y PHP.
Nuevas Características más importantes:

Para JavaScript

- ✓ Soporte de edición para AngularJS, marcos Knockout y ExtJS.
- ✓ Navegador y plegado de código en archivos JSON.
- ✓ Finalización de código mejorada con una mayor precisión.

Para PHP

- ✓ Características de HTML5 disponibles en las aplicaciones.
- ✓ Apoyo estático de análisis de código. (20)

Se eligió este IDE para la implementación porque permite la interacción con las librerías de ExtJS y la generación automática de códigos para los lenguajes que soporta.

1.3.9 Generador de reportes

Los generadores de reportes son herramientas complementarias de los sistemas de información. Proveen una forma transparente al usuario para realizar consultas a la base de datos y obtener información de ella en forma de reporte. En el mundo existen varios generadores de reporte entre los que se encuentran JasperReports, AgataReports y GDR 1.8, este último desarrollado en la UCI.

GDR 1.8

El Generador Dinámico de Reportes (GDR) ofrece una solución de reporte que le brinda a los negocios una herramienta reditual e inteligente, destinada a mejorar la rapidez y calidad de la adopción de decisiones en todos los niveles de una organización. El sistema Generador Dinámico de Reportes constituye una aplicación RIA, (del inglés, *Rich Internet Applications*) puesto que combina las ventajas que ofrecen las aplicaciones web y las aplicaciones tradicionales o de escritorio. Constituye una de las herramientas que posibilitan la inteligencia de negocios ya que genera vistas agregadas de datos para mantener a la gerencia informada sobre el estado de su negocio, además de que:

- ✓ Proporciona soporte a todo el ciclo de vida de los reportes.
- ✓ Provee accesibilidad a la información emitiendo reportes a través de un navegador Web u otros formatos estándares.
- ✓ Está orientada al usuario final, brindando herramientas como el diseñador de reportes que permite la confección de informes de forma interactiva en la web. (21)

JasperReports 4.5

JasperReports es una herramienta de creación de informes *Java open source* que tiene la habilidad de entregar contenido rico en el monitor, a la impresora o a ficheros PDF, HTML, XLS, CSV y XML. Es uno de los motores de informes Java más utilizado del mundo. Permite combinar fuentes de datos y producir documentos para su visualización, impresión o exportación a una variedad de formatos. Está escrito completamente en Java y puede ser usado en gran variedad de aplicaciones de Java, incluyendo J2EE o aplicaciones Web, para generar contenido dinámico.

Su propósito principal es ayudar a crear documentos de tipo páginas, preparados para imprimir en una forma simple y flexible. JasperReports se usa comúnmente con *iReports*, unos *front-end* gráficos *open source* para la edición de los informes. Algunas de sus funciones son:

- ✓ Presentación de *dashboards*, tablas, tablas cruzadas, gráficos y *widgets*.
- ✓ Diseño con salida continua o por páginas para su impresión o visualización en la web.

- ✓ Sub-informes para diseños altamente complejos. (22)

Como herramienta para la generación de reporte se decidió usar JasperReports por todas las características y funcionalidades que brinda. JasperReports presenta una arquitectura escalable permitiendo que los reguladores de consultas controlen el uso de los recursos del sistema y los virtualizadores de informes optimicen el consumo de la memoria. Además la segunda versión del generador GDR va a utilizar como motor de reportes JasperReports.

OpenSSH

OpenSSH es una versión libre de SSH (*Secure SHell*, en español: intérprete de órdenes segura) y sirve para acceder a computadoras remotas a través de una red. Este usa técnicas de cifrado que hacen que la información que viaja por el medio de comunicación vaya de manera no legible, evitando de forma efectiva que terceras personas puedan realizar escuchas, secuestros de las conexiones y otros ataques a nivel de red. Además permite manejar por completo la computadora mediante un intérprete de comandos. (23)

Teniendo en cuenta las características expuestas, se decidió utilizar OpenSSH para la extracción de los datos de las computadoras.

1.3.10 Herramientas de Pruebas

Con el avance de las tecnologías, el proceso de las pruebas de *software* también ha evolucionado por lo que se hace necesario la creación de herramientas para apoyar, agilizar y facilitar dicho proceso. El flujo de pruebas es muy importante en el desarrollo de un *software* para verificar y revelar la calidad del mismo, por lo que se hace imprescindible la investigación del uso de herramientas para verificar que estos se cumplan. Las pruebas de software cuentan con una serie de herramientas que automatizan y ayudan a un mejor resultado de las mismas. Existen herramientas tales como:

Jmeter 2.3.4

JMeter es una herramienta de carga para llevar acabo simulaciones sobre cualquier recurso de Software. Inicialmente fue diseñada para pruebas de estrés en aplicaciones web, ahora su arquitectura ha evolucionado no sólo para llevar a cabo pruebas en componentes habilitados en Internet (HTTP), sino además en Bases de Datos, programas en Perl y prácticamente cualquier otro medio. JMeter soporta aserciones para asegurarse que los datos recibidos son correctos, por cookies de hilos, configuración de variables y una variedad de reportes. Se utiliza habitualmente para analizar las

prestaciones (el rendimiento) de recursos dinámicos y estáticos, tales como, objetos Java, bases de datos, etc. También se utiliza con frecuencia para diseñar test de carga, emulando grandes niveles de concurrencia sobre los servidores. (24)

Se decide seleccionar a la herramienta JMeter 2.3.4 para la realización de las pruebas debido a que permite el diseño y automatización de las pruebas funcionales a través de una interfaz GUI (*Graphical User Interface*, Interfaz Gráfica de Usuario) a modo de diseñador. Además permite la visualización de los resultados de la ejecución de las pruebas de múltiples formas, obteniendo diferentes métricas.

Conclusiones

En el presente capítulo se realizó una investigación de los sistemas existentes en el mundo en el campo de las aplicaciones dedicadas a la gestión de soporte e información. Se efectuó un análisis de la segunda versión del Sistema para la Gestión de Información de Incidencias, Descargas y Recursos del Centro de Tecnologías de Gestión de Datos. Estos análisis permitieron dar comienzo al desarrollo de un sistema capaz de cubrir las necesidades actuales del grupo de Soporte. Teniendo en cuenta esto, se exhibe una síntesis de las tecnologías, herramientas y metodología a utilizar durante la elaboración del software fundamentando su selección. Se decidió usar como lenguaje de programación del lado del cliente JavaScript auxiliándose de la librería de ExtJS 3.4.0 y el lenguaje de programación PHP del lado del servidor. Se estableció utilizar como framework de desarrollo Symfony2.3, como IDE el NetBeans 7.4, como gestor de base de datos PostgreSQL 9.2, como metodología de desarrollo OpenUP y como herramienta CASE Visual Paradigm 6.4. Se definió como herramienta de prueba Jmeter 2.3.4 y como generador de reportes JasperReports 4.5**.** Este ambiente de desarrollo seleccionado apoya la implementación del sistema propuesto, brindando agilidad y variedad de funciones a los desarrolladores.

CAPÍTULO 2: CARACTERÍSTICAS Y DISEÑO DEL SISTEMA

Introducción

En el presente capítulo se describe el análisis y diseño del Sistema Integrado de Soporte del Centro DATEC. Se realiza el modelo de dominio y el levantamiento de requisitos funcionales y no funcionales para lograr el sistema que cumpla con las actuales exigencias del grupo de Soporte. Mediante el diagrama de caso de uso se representa la relación de los actores con los mismos y además se describe el caso de uso más significativo. Se brinda un acercamiento a la implementación del sistema generando los diagramas de interacción y modelo de clases siguiendo el estilo arquitectónico y los patrones de diseño seleccionados. Se muestra el modelo de datos con la descripción de las tablas más significativas.

2.1 Modelo de Dominio

Un modelo del dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. También se les denomina modelos conceptuales, modelo de objetos del dominio y modelos de objetos de análisis, ya que ayuda a comprender los conceptos clave de un negocio o un dominio de problema. (25) A continuación se presenta el modelo de dominio permitiendo mostrar el entorno real del grupo de Soporte del Centro DATEC.

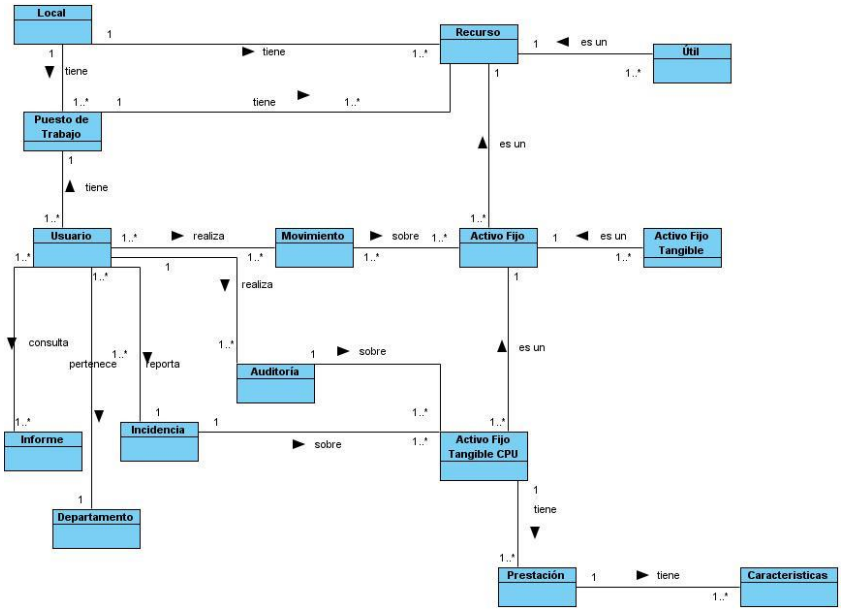


fig. 1: Modelo del Dominio

Descripción del modelo del dominio

El Grupo de Soporte atiende varios locales asociados al Centro DATEC. Estos locales contienen recursos y puestos de trabajos los cuales son asignados a los usuarios. Los usuarios por su parte pertenecen a un departamento, además poseen la facilidad de reportar incidencias y realizar auditorías a los recursos de tipo Activo Fijo Tangible CPU. Los usuarios también pueden realizarle consultas a los informes relacionados con el centro y realizarle movimiento a todos los Activos Fijos de la entidad. Los recursos son de tipo Útil o Activo Fijo que estos últimos a su vez son de tipo Activo Fijo tangible o Activo Fijo tangible CPU. Los activos fijos tangibles de tipo CPU tiene asociados un conjunto de prestaciones que estas a su vez poseen características. A continuación se describen las clases del modelo del dominio:

Local: Es el área física que va a contener los recursos asociados al centro.
Usuario: Personal que tendrá asociado recursos en la entidad.
Puesto de Trabajo: Espacio físico compuesto por un conjunto de recursos asignados a los usuarios para el desarrollo de sus actividades.
Auditoría: Chequeo de seguridad informática que se le realiza a un recurso activo fijo tangible de tipo CPU para ver si este cumple con las medidas de seguridad establecidas por el centro.
Informe: Información a la que pueden acceder los usuarios para ver el estado de los recursos del centro.
Recurso: Se refiere a todos los recursos que se encuentran dentro de la entidad, estos pueden ser o no asignados a los usuarios.
Activo Fijo: Es un tipo de recurso y se refiere a todos los recursos que contienen un número de inventario asociado.
Activo Fijo Tangible CPU: Activo fijo tangible tecnológico capaz de almacenar información. Esta posee un conjunto de prestaciones.
Prestación: Conjunto de informaciones técnicas que poseen los CPU. Están asociadas a ella un conjunto de características.
Características: Conjunto de cualidades que poseen las prestaciones de trabajo. Estas pueden ser variables. Ejemplo de estas son: Modelo, Serie, Tamaño entre otros.
Activo Fijo Tangible: Es un tipo recurso activo fijo que contiene características en la entidad y un número de inventario. Este activo no posee prestaciones.

Útil: Es un tipo de recurso que contiene un número de serie y representan componentes adicionales que pueden existir en un local determinado.

Incidencias: Es un reporte realizado por un usuario sobre cualquier situación fuera de lo normal en un recurso Activo Fijo Tangible CPU. Permite informar el estado técnico de un activo fijo.

Movimiento: Constituye un cambio de posición que se realiza a los activos fijos, que se puede hacer de un local a otro o en el mismo de una forma simple o masiva.

Departamento: Es la estructura con mayor nivel organizacional.

2.2 Requisitos del sistema

Un requisito del sistema es una necesidad documentada sobre el contenido, forma o funcionalidad de un producto o servicio que se utilizan como datos de entrada en la etapa de diseño. Son una condición o capacidad que un usuario necesita para poder resolver un problema o lograr un objetivo. (26)

2.2.1 Requisitos Funcionales

Los requisitos funcionales definen el comportamiento interno de un *software*, son condiciones que el sistema ha de cumplir. Estos muestran las funcionalidades que deben satisfacerse para cumplir con las especificaciones de *software*. (27) Para el desarrollo del Sistema Integrado de Soporte del Centro DATEC se toma como base el sistema anterior del cual se modificarán 12 requisitos funcionales. A pesar de estos modificados se definieron 36 requisitos de los cuales algunos se describen a continuación:

RF23: Adicionar Activo Fijo Tangible CPU:

Descripción: Permite registrar en el sistema un nuevo Activo Fijo Tangible CPU.

Entradas: número de inventario (Alfanumérico 1 a 15 caracteres), dirección IP (Cadena de caracteres que se correspondan con una dirección IP válida), puesto de trabajo (Campo de selección), prestaciones (Lista).

Prototipo elemental de interfaz gráfica de usuario

fig. 2: Interfaz de usuario Adicionar Activo Fijo Tangible CPU

RF26: Modificar Activo Fijo Tangible CPU:

Descripción: Permite modificar el recurso seleccionado en el sistema.

Entradas: número de inventario (Alfanumérico 1 a 15 caracteres), dirección IP (Cadena de caracteres que se correspondan con una dirección IP válida), puesto de trabajo (Campo de selección), prestaciones (Lista de prestaciones).

Prototipo elemental de interfaz gráfica de usuario:

fig. 3: Interfaz de usuario Modificar Activo Fijo Tangible CPU

RF30: Mover recurso:

Descripción: Permite registrar un movimiento de un recurso a un local o a un puesto de trabajo.

Entradas: tipo de movimiento (Campo de selección), tipo de recurso (Campo de selección), Local Origen (Campo de selección), Local Destino (Campo de selección), puesto de trabajo (Campo de selección),

recursos (Lista de CPU), Autorizado por (Cadena de Caracteres (Debe empezar con letra inicial mayúscula)), Aprobado por (Cadena de Caracteres (Debe empezar con letra inicial mayúscula)), Recibido por (Cadena de Caracteres (Debe empezar con letra inicial mayúscula)).

Prototipo elemental de interfaz gráfica de usuario:

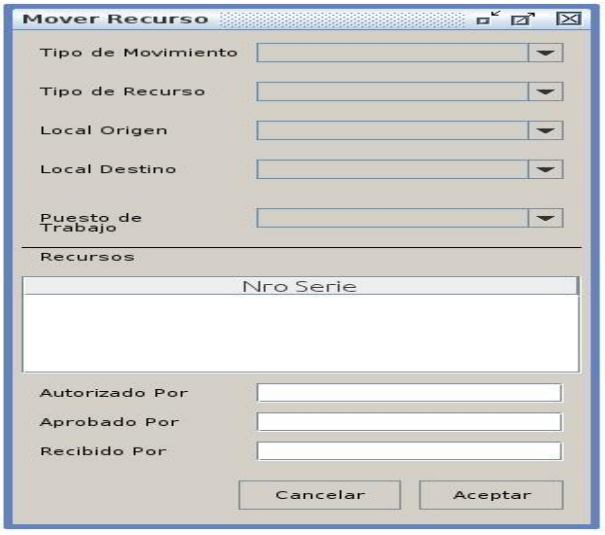


fig. 4: Interfaz de usuario Mover Recurso

RF31: Cargar Prestación:

Descripción: Permite cargar las prestaciones de los CPU.

Entradas: recursos (lista de recursos CPU), usuario (Cadena de caracteres, (Debe empezar con letra inicial minúscula)), contraseña (Cadena de caracteres).

Prototipo elemental de interfaz gráfica de usuario

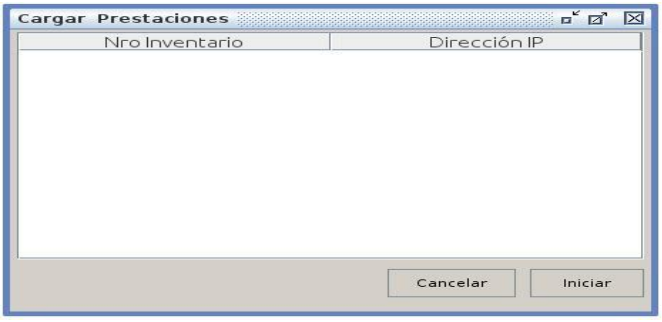


fig. 5: Interfaz de usuario Cargar Prestación

Las descripciones de los requisitos funcionales que no fueron especificados en el documento se encuentran en el artefacto Especificación de Requisitos del Expediente de Proyecto.

2.2.2 Requisitos no Funcionales

Después de haberse definido los requisitos funcionales se definen los requisitos no funcionales ya que especifican los criterios que se deben usar para juzgar el funcionamiento de un sistema. Los requisitos no funcionales que se identificaron son:

RNF1.Software

El servidor donde se instalará la aplicación debe cumplir con los siguientes requisitos:

- ✓ Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 12.04 o superior, Debian 6 GNU/Linux o superior.
- ✓ Paquetes: apache2, php5, libapache2-mod-php5, php5-cli, php5-pgsql, php5-sqlite, libssh2-php, libssh2-1-dev, php5-soap, php-apc, php5-curl, ssh.
- ✓ Navegador: Mozilla Firefox versión 4.0 o superior.

El servidor donde se instalará la base de datos debe cumplir con los siguientes requisitos:

- ✓ Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 12.04 o superior, Debian 6 GNU/Linux o superior.
- ✓ PostgreSQL versión 9.0 o superior.
- ✓ PgAdmin III o algún administrador para PostgreSQL.
- ✓ PostgreSQL debe estar correctamente configurado para aceptar conexiones vía TCP/IP utilizando el método de autenticación por md5.

RNF2.Hardware

Las PC clientes debe cumplir con los siguientes requisitos de hardware:

- ✓ Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- ✓ Memoria RAM mínimo 256 MB.
- ✓ Un mínimo de 1GB de espacio en disco.

Las PC Servidor debe cumplir con los siguientes requisitos de hardware:

- ✓ Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador. Memoria RAM mínimo 1Gb.
- ✓ Disco Duro con 10Gb de capacidad para instalar el sistema.

RNF3.Eficiencia

El sistema responderá de manera inmediata siempre que se cumplan los requisitos de hardware necesarios:

- ✓ Para 100 peticiones concurrentes el sistema responderá en menos de 3 segundos.
- ✓ Para 250 peticiones concurrentes el sistema responderá en menos de 8 segundos.

RNF4.Portabilidad

El sistema deberá ser desarrollado de forma tal que sea multiplataforma.

RNF5.Seguridad.

La información solo podrá ser vista por los usuarios con el nivel de acceso requerido para ello; permitiendo que las funcionalidades del sistema se muestren de acuerdo a lo permisos del usuario que esté activo.

2.3 Modelo de casos de uso

El diagrama de casos de uso documenta el comportamiento de un sistema desde el punto de vista del usuario. Por tanto los casos de usos determinan los requisitos funcionales del sistema, los cuales representan las funcionalidades que un sistema puede ejecutar. (28)

Caso de Uso

Un caso de uso es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios u otros sistemas. (28)

Actores de Sistema

Los actores pueden representar roles jugados por usuarios humanos, hardware externo, u otros sujetos. Un actor no necesariamente representa una entidad física específica, sino simplemente una faceta particular de alguna actividad que es relevante a la especificación de sus casos de uso asociados.

Tabla. 1 Descripción de los Actores del Sistema

Actor	Descripción
Administrador	Es el encargado de administrar todos los usuarios existentes en el sistema y de otorgarle los roles a los mismos. Además tiene acceso a todas las funcionalidades del sistema.
Usuario	Son todos los usuarios del sistema. Sus funcionalidades en el mismo están determinadas por el rol que le conceda el administrador.

LDAP	El Protocolo Ligero de Acceso a Directorios (*Lightweight Directory Access Protocol*, LDAP por sus siglas en inglés) es un conjunto de protocolos de acceso a directorios de información. De él se utilizan las prestaciones para la autenticación, regidas por los elementos: usuario y contraseña, que representan a todos los usuarios del dominio UCI.

2.3.1 Diagrama de Casos de Usos

Se define que un diagrama de casos de usos documenta el comportamiento de un sistema desde el punto de vista del usuario. Muestra cómo los actores del sistema se relacionan con las funcionalidades desde su entrada en el mismo.

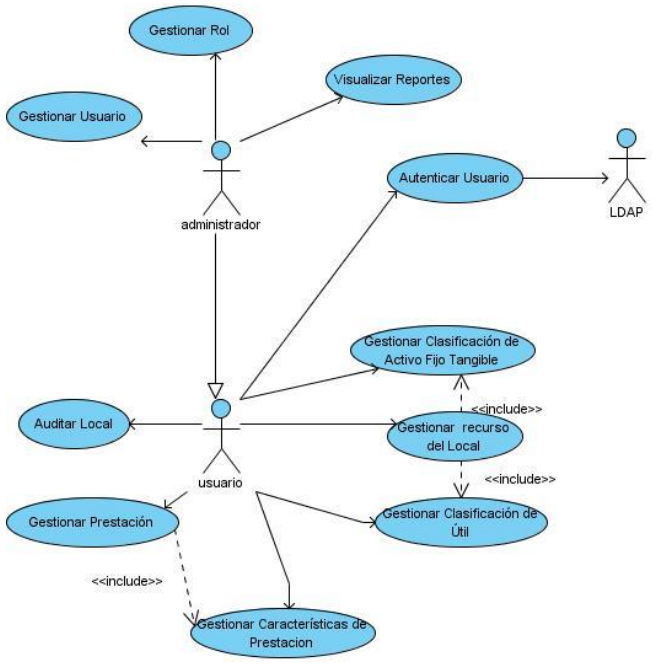


fig. 6: Diagrama de Casos de Usos

En la figura 6 se muestran 10 casos de uso del sistema, de ellos 4 son casos de usos significativos (Gestionar Recurso del local, Gestionar Prestación, Gestionar Usuario, Gestionar Rol) y 4 serán modificados del sistema anterior (Gestionar Rol, Gestionar Usuario, Autenticar Usuario, Visualizar Reportes). El actor Administrador tendrá la tarea de asignar los roles y los permisos a los usuarios que intervienen en el proceso a través del

caso de uso Gestionar Rol. Este es el único que puede realizar todas las funcionalidades del sistema. El actor Usuario realizará las funcionalidades definidas en los permisos del rol que represente.

Se definieron las relaciones entre las funcionalidades a través de relaciones de inclusión (<<include>>). Se utilizaron los patrones CRUD (Crear, Mostrar, Modificar y Eliminar) que se aplican cuando se quiere realizar altas, bajas, cambios y consultas a algunas de las entidades del sistema e inclusión concreta que se aplica cuando un flujo puede extender del flujo de otro caso de uso así como ser realizado en sí mismo. Se tuvo como relación entre los actores el patrón múltiples actores, donde el actor usuario puede tener el mismo rol sobre los casos de usos del administrador en dependencia de sus permisos. Se utilizó también el patrón múltiples actores roles diferentes ya que el caso de uso Autenticar Usuario interactúa con actor Usuario y con LDAP.

2.3.2 Descripción de los casos de uso del sistema

Mediante la descripción de los casos de uso del sistema se detalla la secuencia de eventos que los actores llevan a cabo para completar un determinado proceso a través del sistema. La siguiente tabla describe los procesos adicionar, modificar, eliminar, mover y cargar prestaciones del caso de uso Gestionar recurso a local.

Tabla. 2 Descripción de casos de usos del sistema.

Objetivo	Gestionar Recurso del Local.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el Usuario de la aplicación va a realizar algunas de las acciones para gestionar recurso a local de la misma (adicionar, modificar, listar, eliminar, mover y cargar), finalizando así el CU.
Complejidad	Media.
Prioridad	Crítico.
Precondiciones	El usuario ha sido autenticado en el sistema con los permisos necesarios.

Postcondiciones	En dependencia de la acción del Usuario: • Se registra un nuevo Activo Fijo Tangible. • Se registra un nuevo Activo Fijo Tangible CPU. • Se registra un nuevo Útil. • Se modifican los datos de un Activo Fijo Tangible. • Se modifican los datos de un Activo Fijo Tangible CPU. • Se modifican los datos de un Útil. • Se elimina un Recurso del Local. • Se mueve un recurso. • Se obtienen las prestaciones.

Flujo de eventos

Flujo básico "Gestionar Recurso a Local"

	Actor	**Sistema**
1.	Selecciona la opción Recursos del Local.	
2.		Muestra un listado de todos los recursos del local existentes en dependencia de su tipo.
3.		Permite realizar las siguientes acciones: a. Adicionar un nuevo recurso del local: ir a la **sección 1 "Adicionar Activo Fijo Tangible".** b. Adicionar un nuevo recurso del local: ir a la **sección 1 "Adicionar Activo Fijo Tangible CPU".** c. Adicionar un nuevo recurso del local: ir a la **sección 1 "Adicionar Útil".** d. Modificar los datos de un recurso del local**:** ir a la **sección 2 "Modificar Activo Fijo Tangible".** e. Modificar los datos de un recurso del local**:** ir a la **sección 2 "Modificar Activo Fijo Tangible CPU".** f. Modificar los datos de un recurso del local**:** ir a la **sección 2 "Modificar Útil".** g. Eliminar recurso del local: ir a la **sección 3 "Eliminar recurso del local".** h. Mover recurso del local: ir a la **sección 4 "Mover**

		recurso". i. Cargar las prestaciones: ir a la **sección 5 "Cargar Prestación".**
4.		Termina el caso de uso.
Flujos alteros		
2.a No existen usuarios registrados		
1.		Muestra la lista de los recursos vacía y continua al paso 3 del flujo normal de eventos permitiendo solamente la opción de adicionar.
Sección 1: "Adicionar Activo Fijo Tangible"		
Flujo básico "Adicionar Activo Fijo Tangible"		
	Actor	Sistema
1.	Selecciona la opción Adicionar.	
		Muestra el tipo de recurso a seleccionar.
	Selecciona el tipo Activo Fijo Tangible.	
2.		Muestra un formulario para introducir los datos de un Activo Fijo Tangible.
3.	Introduce los datos necesarios para adicionar un nuevo Activo Fijo Tangible: MB, puesto de trabajo, clasificación y descripción, selecciona la opción Aceptar.	
4.		Valida que los datos introducidos son correctos y/o los campos obligatorios no están vacíos.
5.		Verifica que no exista un Activo Fijo Tangible con el mismo MB.
6.		Adiciona el nuevo Activo Fijo Tangible.
7		Muestra el mensaje de confirmación "La operación se realizó satisfactoriamente".
8		Actualiza el listado de los recursos.
9		Termina el caso de uso.
Flujos alternos		
3.a Operación cancelada		
	Actor	**Sistema**
1.		Muestra el mensaje: "¿Realmente desea salir sin enviar las últimas modificaciones?".
4.a Datos incorrectos y/o campos obligatorios vacíos		
	Actor	**Sistema**
1.		Señala en rojo los campos con problemas y no activa la opción de Aceptar.

5.a Recurso ya existe		
	Actor	**Sistema**
1.		Muestra el mensaje “El elemento ya existe en la base de datos”.
Sección 1: “Adicionar Activo Fijo Tangible CPU”		
Flujo básico “Adicionar Activo Fijo Tangible CPU”		
	Actor	**Sistema**
1.	Selecciona la opción Adicionar.	
		Muestra el tipo de recurso a seleccionar.
	Selecciona el recurso Activo Fijo Tangible CPU.	
2.		Muestra un formulario para introducir los datos de un Activo Fijo Tangible CPU.
3.	Introduce los datos necesarios para adicionar un nuevo Activo Fijo Tangible CPU: puesto de trabajo, MB, IP y prestaciones, selecciona opción Aceptar.	
4.		Valida que los datos introducidos son correctos y/o los campos obligatorios no están vacíos.
5.		Verifica que no exista un Activo Fijo Tangible CPU con el mismo MB o IP.
6.		Adiciona el nuevo Activo Fijo Tangible CPU.
7.		Muestra el mensaje de confirmación “La operación se realizó satisfactoriamente”.
8.		Actualiza el listado de los recursos.
9.		Termina el caso de uso.
Flujos alternos		
3.a Operación cancelada		
	Actor	**Sistema**
1.		Muestra el mensaje: “¿Realmente desea salir sin enviar las últimas modificaciones?”.
4.a Datos incorrectos y/o campos obligatorios vacíos		
	Actor	**Sistema**
1.		Señala en rojo los campos con problemas y no activa la opción de Aceptar.
5.a Recurso ya existe		
	Actor	**Sistema**
1.		Muestra el mensaje “El elemento ya existe en la base de datos.”

Sección 1: "Adicionar Útil"		
Flujo básico "Adicionar Útil"		
	Actor	**Sistema**
1.	Selecciona la opción Adicionar.	
2.		Muestra el tipo de recurso a seleccionar.
3.	Selecciona el tipo de recurso Útil.	
4.		Muestra un formulario para introducir los datos de un Útil.
5.	Introduce los datos necesarios para adicionar un nuevo Útil: puesto de trabajo, clasificación, número de serie y descripción, selecciona opción Aceptar.	
6.		Valida que los datos introducidos son correctos y/o los campos obligatorios no están vacíos.
7.		Verifica que no exista un Útil con el mismo número de serie.
8.		Adiciona el nuevo Útil.
9.		Muestra el mensaje de confirmación "La operación se realizó satisfactoriamente".
10.		Actualiza el listado de los recursos.
11.		Termina el caso de uso.
Flujos alternos		
3.a Operación cancelada		
	Actor	**Sistema**
1.		Muestra el mensaje: "¿Realmente desea salir sin enviar las últimas modificaciones?".
	Actor	**Sistema**
1.		Señala en rojo los campos con problemas y no activa la opción de Aceptar.
5.a Recurso ya existe		
	Actor	**Sistema**
1.		Muestra el mensaje "El elemento ya existe en la base de datos".
Sección 2: "Modificar Activo Fijo Tangible"		
Flujo básico "Modificar Activo Fijo Tangible"		
	Actor	**Sistema**

1.	Elige el usuario deseado y selecciona la opción Modificar.	
2.		Muestra un formulario con la información del recurso seleccionado con los siguientes campos: puesto de trabajo, MB, clasificación y descripción.
3.	Introduce la nueva información que desea actualizar e indica guardar los cambios.	
4.		Valida que los datos introducidos son correctos y/o los campos obligatorios no están vacíos.
5.		Verifica que no exista otro recurso con el mismo MB.
6.		Actualiza el recurso con la nueva información.
7.		Muestra el mensaje de confirmación “La operación se realizó satisfactoriamente”.
8.		Actualiza el listado de los recursos.
9.		Termina el caso de uso.
Flujos alternos		
3.a Operación cancelada		
	Actor	**Sistema**
1.		Muestra el mensaje: “¿Realmente desea salir sin enviar las últimas modificaciones?”.
4.a Datos incorrectos y/o campos obligatorios vacíos		
	Actor	**Sistema**
1.		Señala en rojo los campos con problemas y no activa la opción de Aceptar.
5.a Recurso ya existe		
	Actor	**Sistema**
1.		Muestra el mensaje “El elemento ya existe en la base de datos”.
Sección 2: “Modificar Activo Fijo Tangible CPU”		
Flujo básico “Modificar Activo Fijo Tangible CPU”		
	Actor	**Sistema**
1.	Elige el usuario deseado y selecciona la opción Modificar.	
2.		Muestra un formulario con la información del recurso seleccionado con los siguientes campos: MB, IP y prestaciones.
3.	Introduce la nueva información que desea actualizar e indica guardar los cambios.	
4.		Valida que los datos introducidos son correctos y/o los campos obligatorios no están vacíos.

5.		Verifica que no exista otro recurso con los mismos MB o IP.
6.		Actualiza el recurso con la nueva información.
7.		Muestra el mensaje de confirmación “La operación se realizó satisfactoriamente”.
8.		Actualiza el listado de los recursos.
9.		Termina el caso de uso.
Flujos alternos		
3.a Operación cancelada		
	Actor	**Sistema**
1.		Muestra el mensaje: “¿Realmente desea salir sin enviar las últimas modificaciones?”
4.a Datos incorrectos y/o campos obligatorios vacíos		
	Actor	**Sistema**
1.		Señala en rojo los campos con problemas y no activa la opción de Aceptar.
5.a Recurso ya existe		
	Actor	**Sistema**
1.		Muestra el mensaje “El elemento ya existe en la base de datos”.
Sección 2: “Modificar Útil”		
Flujo básico “Modificar Útil”		
	Actor	**Sistema**
1.	Elige el usuario deseado y selecciona la opción Modificar.	
2.		Muestra un formulario con la información del recurso seleccionado con los siguientes campos: puesto de trabajo, clasificación, número de serie y descripción.
3.	Introduce la nueva información que desea actualizar e indica guardar los cambios.	
4.		Valida que los datos introducidos son correctos y/o los campos obligatorios no están vacíos.
5.		Verifica que no exista otro recurso con el mismo número de serie.
6.		Actualiza el recurso con la nueva información.
7.		Muestra el mensaje de confirmación “La operación se realizó satisfactoriamente”.
8.		Actualiza el listado de los recursos.
9.		Termina el caso de uso.

Flujos alternos		
3.a Operación cancelada		
	Actor	**Sistema**
1.		Muestra el mensaje: "¿Realmente desea salir sin enviar las últimas modificaciones?".
4.a Datos incorrectos y/o campos obligatorios vacíos		
	Actor	**Sistema**
1.		Señala en rojo los campos con problemas y no activa la opción de Aceptar.
5.a Recurso ya existe		
	Actor	**Sistema**
1.		Muestra el mensaje "El elemento ya existe en la base de datos"
Sección 3: "Eliminar Recurso del Local"		
Flujo básico "Eliminar Recurso del Local"		
	Actor	**Sistema**
1	Elige el recurso deseado y selecciona la opción Eliminar.	
2		Muestra mensaje de confirmación: "¿Está seguro que quieres eliminar un recurso?".
3	Confirma acción de eliminar.	
4		Elimina el recurso seleccionado.
5		Actualiza el listado de recursos.
6		Termina el caso de uso.
Sección 4: "Mover Recurso"		
Flujo básico "Mover Recurso"		
	Actor	**Sistema**
1	Selecciona la opción Mover	
2		Muestra un formulario para introducir los datos del movimiento.

3	Introduce los datos necesarios para realizar un movimiento: Tipo de movimiento, Tipo de recurso, Local origen, Local destino, Puesto de trabajo, recursos, autorizado por, aprobado por recibido por y selecciona opción Aceptar.	
4		Valida que los datos introducidos son correctos y//o los campos obligatorios no están vacíos.
5		Actualiza el recurso con la nueva información.
6		Muestra el mensaje de confirmación “La operación se realizó satisfactoriamente”
7		Termina el caso de uso.
Flujos alternos		
3.a Operación cancelada		

	Actor	**Sistema**
1.		Muestra el mensaje “El elemento ya existe en la base de datos”
4.a Datos incorrectos y/o campos obligatorios vacíos		
	Actor	**Sistema**
1.		Señala en rojo los campos con problemas y no activa la opción de Aceptar.
Sección 4: “Cargar Prestación”		
Flujo básico “Cargar Prestación”		
	Actor	**Sistema**
1.	Selecciona un local.	
2.	Selecciona la opción Cargar Prestaciones.	
3.		El sistema muestra el listado de los recursos.
4.	Selecciona el o los CPU a cargar.	
5.		El sistema muestra un formulario para introducir los datos: usuario y contraseña.
6.	El usuario introduce los datos.	
7.	Selecciona la opción iniciar.	
8.		El sistema carga las prestaciones.

9.		Envía un mensaje de éxito y termina el caso de uso.
Ú	**CU Incluidos**	- Gestionar Clasificación de Útil, Gestionar Clasificación de Activo Fijo Tangible.
	CU Extendidos	-
Requisitos no funcionales	-	
Asuntos pendientes	-	

2.4 Modelo de diseño

El modelo de diseño describe la realización de casos de usos y sirve como una abstracción del modelo de aplicación y su código fuente. Se utiliza como parte esencial para las actividades en ejecución y prueba, se basa en el análisis y los requisitos de la arquitectura del sistema. Representa los componentes de aplicación y determina su colocación adecuada y el uso dentro de la arquitectura en general del sistema. (29)

2.4.1 Diagrama de Clases del Diseño

Los diagramas de clases del diseño son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se maneja en el sistema y los componentes que se encargan del funcionamiento y la relación entre uno y otro. Describen gráficamente las especificaciones de las clases del software y de las interfaces en una aplicación. En su diseño contiene como información (clases, métodos, información sobre los tipos de los atributos, dependencias). (26) A continuación se presentan las descripciones de los componentes del marco reutilizable Lycan que se utilizan para el desarrollo del sistema:

Nombre: ManagePlugin

Descripción: La clase ManagePlugin es una clase en JavaScript que permite agregar comportamientos específicos a la clase maestro (CU_Master) con la que interactúe; tales como la incorporación de un menú contextual y la barra de tarea inferior de las acciones que en ese contexto se pueden acometer (Adicionar, Actualizar, Buscar, Eliminar, entre otras). Implementa los comportamientos que serán reutilizados en la realización del caso de uso que se implemente.

Nombre: Record

Descripción: La clase Record es una clase en JavaScript que representa los atributos propios del objeto que se gestiona. Un record (Record) se corresponde a un formulario (DetailForm), donde el formulario gestiona todos los atributos o elementos que describen al objeto que se gestiona y el record es la instancia de dicho objeto.

Nombre: DetailForm

Descripción: es una clase en JavaScript correspondiente a una interfaz visual que se encarga de mostrar los detalles o campos correspondientes al objeto que se gestione. Implementa un conjunto de funciones que serán reutilizadas en la realización del caso de uso que se implemente, logrando especificar las funcionalidades necesarias para la gestión de los campos descriptivos del objeto en cuestión.

Nombre: Repositoy

Descripción: Es la clase de PHP ejecutada por Symfony como realización del CU definida como CU_Repository, donde los métodos que se implementan por un criterio determinado realizan la búsqueda de la entidad en la Base de Datos. Cada entidad (entity) tiene asociado un repositorio (repository).

Nombre: DirectStore

Descripción: La clase DirectStore es una clase en JavaScript que representa el listado de objetos o record que se gestionan, donde la clase CU_Store hereda de la misma para la realización del CU que se implementa. Un store (CU_Strore) está compuesto por varios objetos o record (CU_Record), donde se adjuntan y se listan en la clase Recursos para ser mostrados visualmente, permitiendo el manejo de los comportamientos o acciones que se implementan en la realización del CU en cuestión a través de la clase ManagePlugin.

Nombre: Controller

Descripción: clase de PHP que es creada y ejecutada por Symfony como realización del CU, los métodos que se implementan permiten generar el API y re-direccionar las llamadas de ExtDirect. El producto final del controlador es: un objeto de respuesta Symfony2.

A continuación se presenta el diagrama de clases del caso de uso más significativo Gestionar recuro del Local.

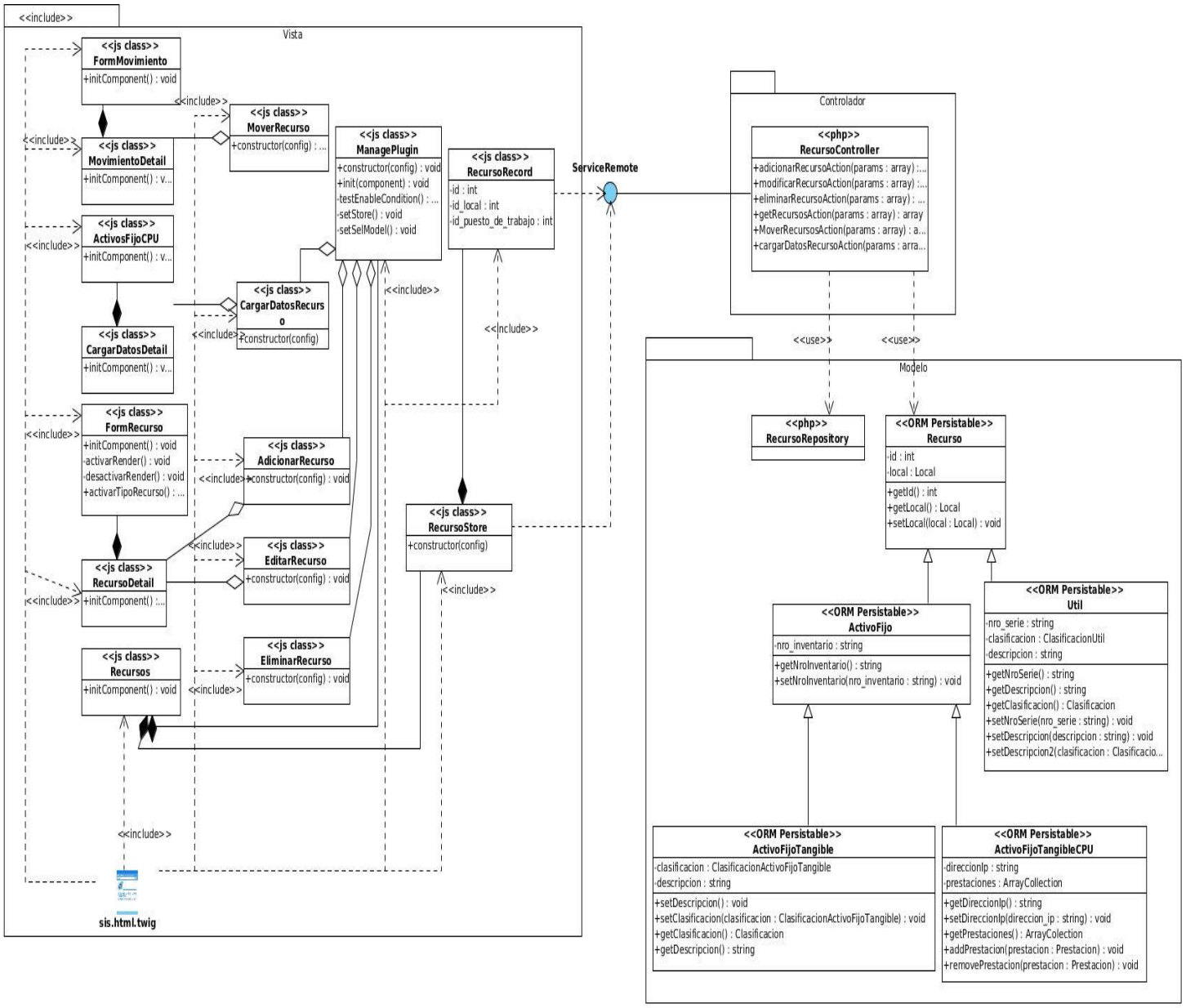


fig. 7: Diagrama de Clases del Diseño Gestionar Recursos del Local

El diagrama de clases del diseño (ver figura 7) contiene las principales clases con sus respectivos métodos y atributos para el caso de uso gestionar Recursos del Local. Las acciones implementan las funcionalidades que permite el caso de uso, en este caso editar, adicionar, eliminar y mover recurso, cada una representada a través de las clases JavaScript AdicionarRecurso, EditarRecurso, EliminarRecurso y Mover Recurso

respectivamente. Las acciones de edición y adición controlan una vista representada por la clase RecursoDetail la cual contiene una clase denominada FormRecurso que permite insertar o editar un objeto de tipo recurso que puede ser Activo Fijo Tangible CPU, Activo Fijo Tangible o Útil. La acción mover controla la vista MovimientoDetail la cual contiene una clase denominada FormMovimiento que permite insertar un movimiento. Otras de las acciones como cargar controlan la vista CargarDatosDetail la cual contiene un grid ActivoFijoCPU que permite insertar un movimiento. Luego que se realiza la edición o adición, la clase RecursoDetail entrega a la acción correspondiente ya sea AdicionarRecurso o EditarRecurso a la clase RecursoRecord asociada a los datos capturados en el formulario, la acción utiliza la capacidad del RecursoRecord para comunicarse con el servidor y ejecutar una petición de adición o edición según la acción que se realice en ese momento.

Los servicios a consumir en el lado del cliente serán definidos en un API perteneciente a las clases RecursoRecord y RecursoStore. En el API mencionado anteriormente se especifican cada uno de las funciones a llamar en el controlador del lado del servidor dependiendo de las acciones que se ejecuten en el lado del cliente. Otro de los elementos fundamentales en la vista es la clase Recursos la cual se encarga de mostrar los recursos registrados en la base de datos. Esta clase contiene un ManagePlugin y un RecursoStore que es el encargado de representar la lista de recurso. En este diagrama de clases se evidencia la reutilización del marco Lycan además muestra el patrón arquitectónico que se utiliza para el desarrollo del sistema.

2.4.2 Patrón de arquitectura

Para la creación del sistema se utilizó el *framework* Symfony el cual propone una arquitectura Modelo-Vista-Controlador (MVC), para una mejor organización y manejo de las vistas. Este patrón separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos.

Modelo: Agrupa los datos y la lógica de la aplicación.
Vista: Se encarga de mostrar los datos recogidos en el modelo, generalmente a través de una interfaz.
Controlador: Se encarga de procesar las interacciones con el usuario y realizar los cambios apropiados en el modelo o en la vista.

Ventajas: Brinda soporte de vistas múltiples, es decir, que la interfaz de usuario puede mostrar múltiples vistas de los mismos datos simultáneamente, debido a que la vista se encuentra separada del modelo y no existe dependencia directa entre ellos.

En el sistema las clases JS generan las interfaces formando parte de la capa Vista. Las clases Controller forman la capa Controladora, debido a que son estas las encargadas de responder las peticiones de los usuarios. Por último la capa Modelo la integrarán las clases EntityRepository, teniendo en cuenta que ellas representan los datos del dominio de la aplicación.

2.4.3 Patrones de diseño utilizados

Patrones GRAP

GRASP es un acrónimo que significa *General Responsibility Asignment Software Patterns* (patrones generales de software para asignar responsabilidades). Estos describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones.

Experto: Asigna una responsabilidad a un experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. 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. (30) Este patrón se aplica en la clase controladora AdicionarRecurso del sistema ya que ella es experta en la realización de su funcionalidad que consiste en adicionar un nuevo recurso al sistema.

fig. 8: Patrón Experto

Creador: El patrón creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que deba conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. (30) Este patrón se utiliza en la clase ManagePlugin ya que es

una clase en JavaScript que permite agregar comportamientos específicos a la clase EliminarUsuario que interactúa con ella.

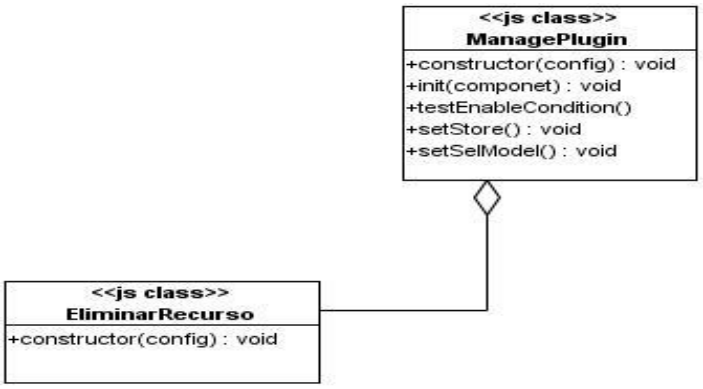


fig. 9: Patrón Creador

Bajo Acoplamiento: Es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño. Soporta el diseño de clases más independientes, que reducen el impacto de los cambios, y también más reutilizables, que acrecientan la oportunidad de una mayor productividad. (30) Se aplica en las clases del modelo, pues existe poca dependencia entra las clases de acceso a datos y las de abstracción de datos, lo que permite mayor reutilización. Se pueden modificar las clases del modelo sin que se afecten las del controlador.

Alta Cohesión: Propone asignar responsabilidades a las clases evitando que estas realicen un trabajo excesivo y que las tareas no sean afines. Es un patrón evaluativo que el desarrollador aplica al valorar sus decisiones de diseño. (30) Por ejemplo este patrón se aplica en la clase EditarRecurso debido a que es una de las encargadas de mostrar a la clase RecursoDetails la información del recurso seleccionado.

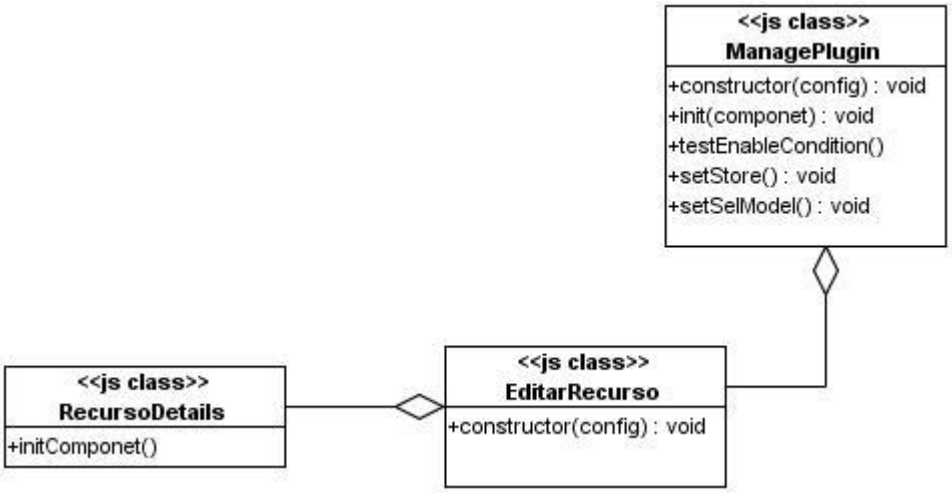


fig. 10: Patrón Alta Cohesión

Controlador: El patrón controlador resuelve el problema: ¿Quién debería ser el responsable de gestionar un evento de entrada al sistema?. Este asigna una responsabilidad de recibir o manejar un mensaje de evento del sistema a una clase. Posibilita la organización y claridad en el diseño. Favorece el bajo acoplamiento. (30) Este patrón se ve aplicado en todas las clases controladoras ya que son las encargadas de ejecutar eventos de entradas externas en el sistema (Adicionar, Modificar, Listar, Eliminar y Mover).

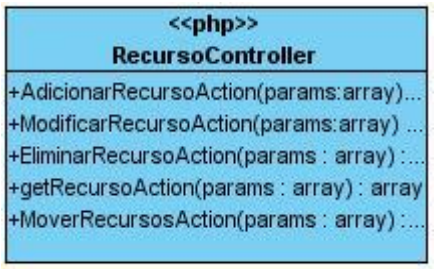


fig. 11: Patrón Controlador

Patrones GOF

Fachada: Proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define la interfaz de alto nivel que hace que sea más fácil de utilizar el subsistema. Se utiliza en la clase AppSIS la cual hace función de interfaz principal de la aplicación y a la vez es la responsable de crear las interfaces visuales

de los módulos Seguridad y Centro, proporcionando una interfaz de alto nivel que hace que la aplicación sea más fácil de usar.

Decorador: Añade dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender funcionalidades. Este patrón se evidencia en la clase FormRecurso porque este podrá ser modificado en dependencia del tipo de recurso a adicionar.

Singleton: Garantiza que una clase sólo tenga una instancia, y proporciona un punto de acceso global a ella. Este patrón se utiliza en la clase Centro ya que en ella se crea una única instancia de cada clase controladora en la vista.

Observador: Define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambia de estado se notifica y actualizan todos los objetos automáticamente. Se aplica en la clase ManagePlugin ya que esta se mantiene observando a la clase Recursos en espera de que esta notifique que un recurso ha sido seleccionado para activar las funcionalidades de editar y eliminar.

Adaptador: Convierte la interfaz de una clase en otra distinta que es la que esperan los clientes. Además permiten que cooperen clases que de otra manera no podrían hacerlo, pues tendrían interfaces incompatibles.

Constructor: Separa la construcción de un objeto complejo de su representación, de forma que el mismo proceso de construcción pueda crear diferentes representaciones. (31)

2.4.4 Diagrama de Interacción

Los diagramas de interacción describen secuencias de intercambios de mensajes entre los roles que implementan el comportamiento de un sistema. Son utilizados para el modelado de los aspectos dinámicos de un sistema, proporcionan una vista integral de su comportamiento. Esto conlleva modelar instancias concretas, componentes y nodos junto con los mensajes enviados entre ellos que representan su interacción. Los diagramas de interacción tienen dos formas de manifestarse:

- ✓ Diagramas de secuencia.
- ✓ Diagramas de colaboración.

Un diagrama de secuencia destaca el orden temporal de los mensajes, ilustra los objetos que se encuentran en un escenario, y la secuencia de mensajes intercambiados entre ellos para llevar a cabo la funcionalidad descrita; el diagrama de colaboración a su vez destaca la organización estructural de los objetos que envían y reciben mensajes. (32)

A continuación se muestran los diagramas de secuencia Adicionar Activo Fijo Tangible del caso de uso Gestionar Recursos del Local:

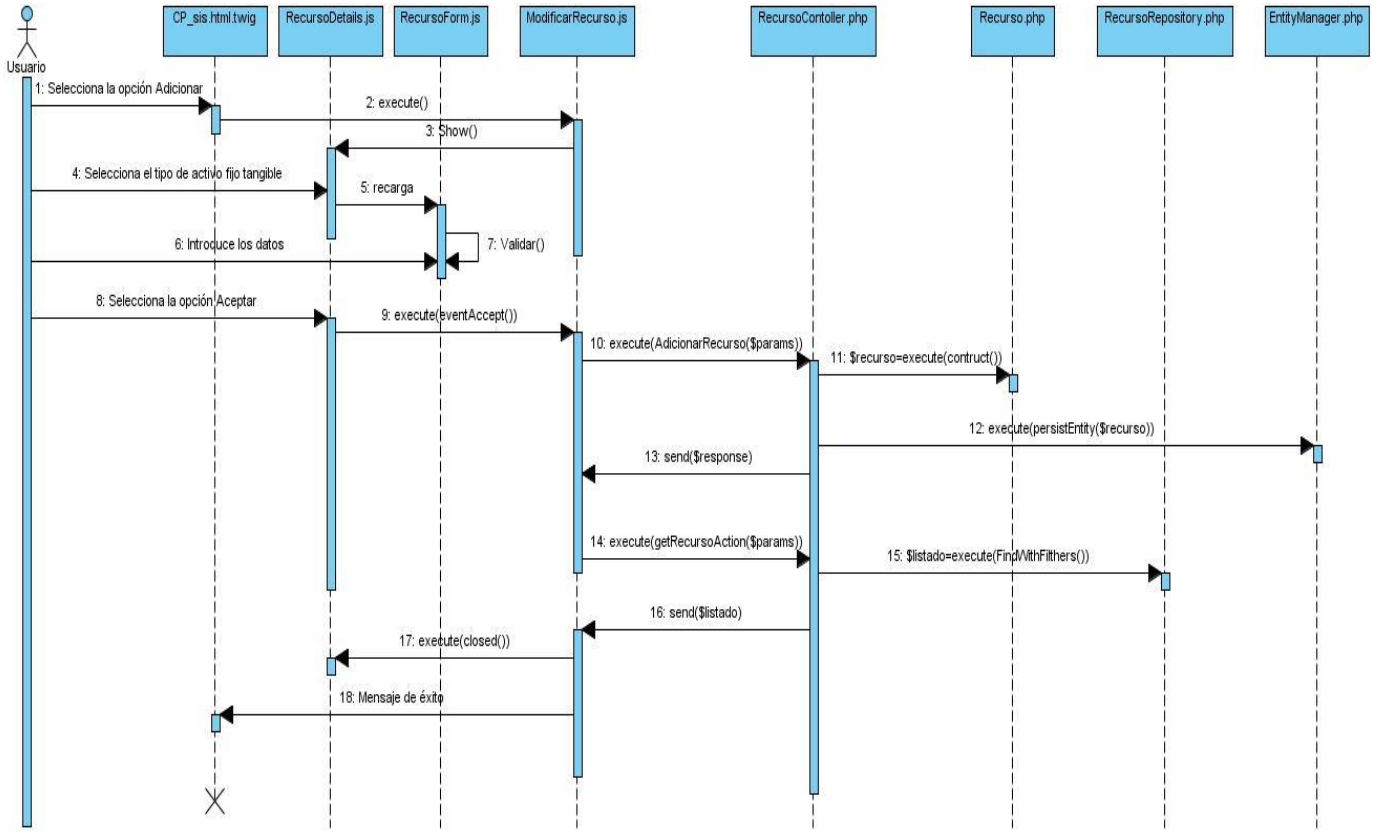


fig. 12: Diagrama de Secuencia Adicionar Activo Fijo Tangible CPU

En el diagrama de secuencia de la figura anterior el actor Usuario selecciona la opción Adicionar Recurso a la página cliente CP_sis.html.twig, esta ejecuta una acción en la clase AdicionarRecurso y muestra los detalles en la clase RecursoDetail. El Usuario selecciona el tipo de recurso Activo Fijo Tangible CPU a la clase RecursoDetails y la misma recarga el formulario. El Usuario introduce los datos en el formulario, este los valida y selecciona la opción aceptar en la clase RecursoDetails, la cual ejecuta un evento sobre la clase AdicionarRecurso ejecutando en la clase RecursoController el método AdicionarRecursoAction($params). Después se ejecuta una consulta sobre la clase Recurso $recurso=execute(construct). La clase RecursoController ejecuta una consulta sobre la clase EntityManager para mantener la persistencia y facilitar la búsqueda en la base de datos. La clase RecursoController envía una respuesta si la consulta fue satisfactoria a la clase AdicionarRecurso la cual envía hacia la clase controladora la lista de los recursos

getRecursoAction($params) ejecutándole a la clase RecursoRepository la consulta de la lista, obteniendo esa lista y enviándola a la clase AdicionarRecurso. Después le envía a la clase details un evento de cerrar el formulario y le envía a la página cliente un mensaje de éxito.

2.5 Modelo de Datos

Un modelo de datos es una colección de conceptos que se emplean para describir la estructura de una base de datos. Esa colección de conceptos incluye entidades, atributos y relaciones. A continuación se muestra el modelo de datos del Sistema Integrado de Soporte del Centro DATEC. (33)

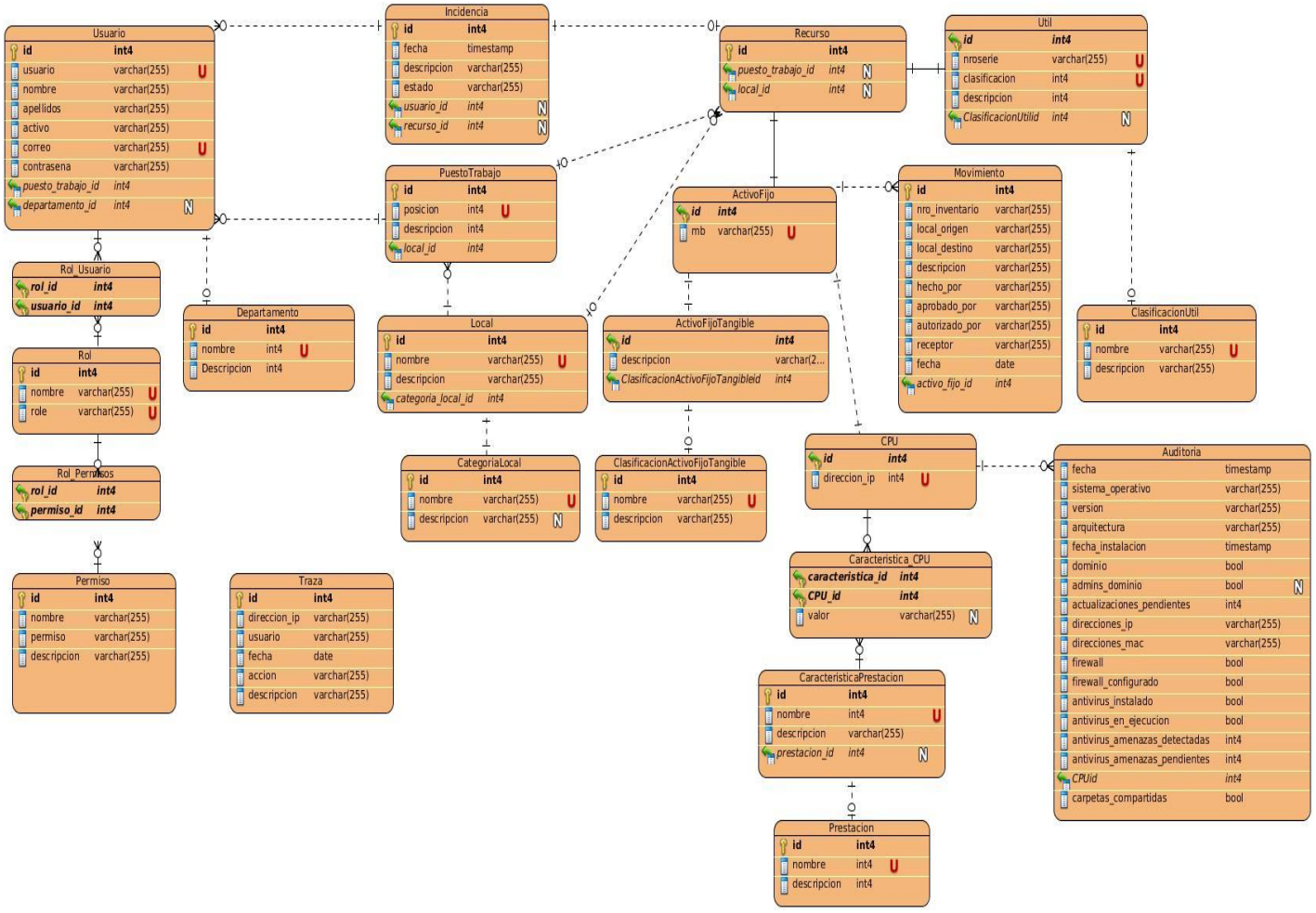


fig. 13: Modelo de datos

En la modelo de datos anterior se aprecia las relaciones existentes entre las tablas de la base de datos asociada a la aplicación. A continuación se muestra las descripciones de las tablas:

Usuario: Tabla que contiene los usuarios que accederán al sistema.
Rol: Tabla que contiene la información de los roles que posteriormente se le asignan los usuarios.
Rol_Usuario: Permite la relación de asignación de cada usuario con un rol determinado.
Permisos: Tabla que contiene toda la información de los permisos.
Rol_Permisos: Tabla que contiene todos los permisos que pueden tener los roles para poder asignárselos a los usuarios posteriormente.
Departamento: Tabla que contiene toda la información de los departamentos del centro.
Local: Posee la información de todos los locales registrados en el sistema.
Categoría de local: Tabla que contiene toda la información de las categorías que pueden tener los locales.
Puesto de Trabajo: Contiene la información de los puestos de trabajo de los locales definidos.
Recurso: Tabla que posee toda la información de los recursos del sistema en dependencia de su tipo.
ActivoFijo: Tabla que contiene la información de los activos fijos en dependencia de su tipo.
ActivoFijoTangible: Tabla que contiene la información de todos los activos fijos de tipo Activo Fijo Tangible.
CPU: Posee toda la información de todos los activos fijos tangibles CPU.
Útil: Posee toda la información de todos los recursos de tipo útiles en la entidad.
ClasificaciónActivoFijoTangible: Contiene toda la información de la clasificación de los activos fijos tangibles.
ClasificaciónÚtil: Contiene toda la información de la clasificación de todos los recurso útiles de la entidad.
Prestación: Permite ver las características de todas las prestaciones de los CPU.
Características: Posee las características de todas las prestaciones.
Caracteristicas_CPU: Contiene las características de un CPU.
Movimiento: Almacena el movimiento que se realizara a un recurso ya sea de tipo simple o masivo.
Trazas: Almacena la información relacionada con todas las trazas del sistema.
Incidencias: Almacena la información de las incidencias que se le reportan a los recursos de tipo activo fijo tangible CPU.

Conclusiones

En el presente capítulo fueron abordados algunos de los artefactos propuestos por la metodología OpenUP para el desarrollo del *software*. Uno de ellos es el modelo de dominio, con el cual se describe el proceso que guía el funcionamiento de la aplicación. Con la especificación de los requisitos funcionales y no funcionales se obtiene una orientación para la creación del producto y una visión de los patrones de diseños GRASP y GOF que se siguen para una mejor implementación del sistema. La realización de los diagramas de las clases del diseño muestra la estructura interna del producto y da una medida de los métodos que se deben desarrollar y donde deben ser ubicados. A su vez se realiza una descripción de los procesos implícitos en la aplicación a través de los diagramas de secuencias y se obtiene la estructura de la base de datos a partir de la confección del diagrama de entidad de relación.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA

Introducción

En el presente capítulo se describe cómo los elementos del modelo del diseño se implementan en términos de componentes y se muestra una descripción detallada de los paquetes de implementación. Se presenta el mapa de navegación del *software* desarrollado y algunas imágenes del mismo para brindar una mejor comprensión del sistema. Posteriormente se procede a la validación de la aplicación mediante las pruebas de *software* de caja negra, carga y *stress*, para comprobar la operatividad de las principales funcionalidades. Y finalmente se muestra el diagrama de despliegue, con una breve descripción de los nodos que lo conforman.

3.1 Diagrama de componentes

El diagrama de componentes es una parte física de un sistema (módulo, base de datos, programa ejecutable). Se puede decir que un componente es la materialización de una o más clases, porque una abstracción con atributos y métodos pueden ser implementados en los componentes. Los diagramas de componentes son utilizados para modelar la vista estática y dinámica del sistema, mostrando la organización y dependencias entre los componentes. (34)

A continuación se presente el diagrama de componentes del caso de uso Gestionar Recurso del Local:

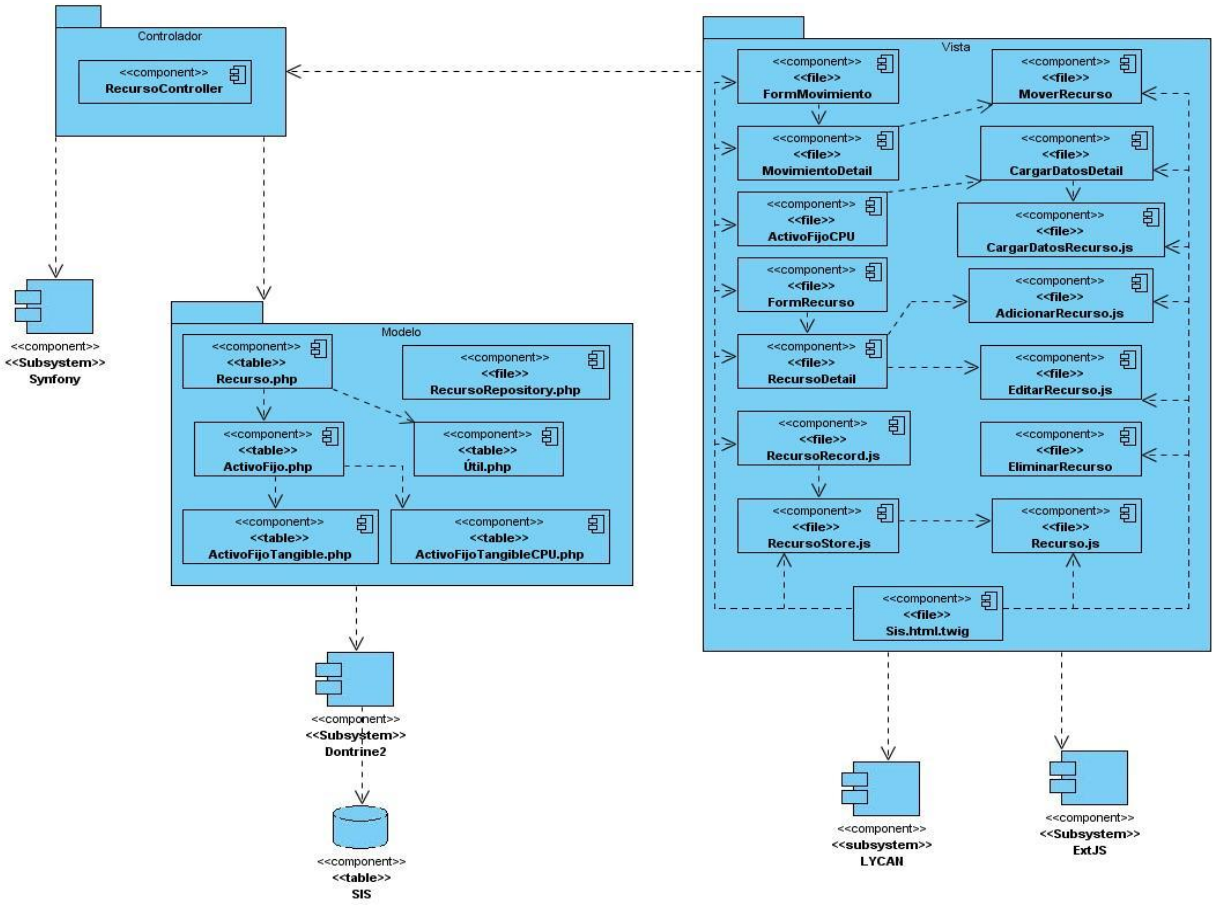


fig. 14: Diagrama de componentes del caso de uso Gestionar Recursos del Local.

- ✓ **Vista:** contiene todas las interfaces de usuario relacionadas al caso de uso Gestionar Recursos del Local, convertidas en componentes <<file>>. La Vista tiene relación de dependencia con el <<subsystem>> ExtJS.
- ✓ **Controlador:** contiene todas las acciones relacionadas al caso de uso Gestionar Recursos del Local, para darle respuesta a las peticiones del usuario, convertidas en componentes <<file>>. El Controlador tiene relación de dependencia con el <<subsystem>> Symfony.
- ✓ **Modelo:** contiene todas las tablas de la base de datos asociadas al caso de uso Gestionar Recursos del Local, convertidas en componentes <<table>>. El Modelo tiene relación de dependencia con el <<subsystem>> Doctrine2.

- ✓ **ExtJS:** paquete que se utiliza en la capa Vista para el diseño de las interfaces, convertido en un componente de tipo <<subsystem>>.
- ✓ **Symfony:** paquete que se utiliza en la capa Controlador para la implementación de las acciones, convertido en un componente de tipo <<subsystem>>.
- ✓ **Doctrine2:** paquete que se utiliza en la capa Modelo para convertir las tablas de la base de datos en clases, convertido en un componente de tipo <<subsystem>>.

3.2 Mapa de navegación del sistema

Un mapa de navegación es la representación gráfica de la organización de la información de una estructura web. Permite elaborar escenarios de comportamiento de los usuarios y expresa todas las relaciones de jerarquía y secuencia. En el diseño del mapa de navegación se debe seleccionar la página de entrada al sitio web, ordenar de manera jerárquica las páginas con los contenidos (por niveles o categorías) y establecer los vínculos entre ellas permitiendo una navegación hipertextual. (35) A continuación se presentan el mapa de navegación del sistema:

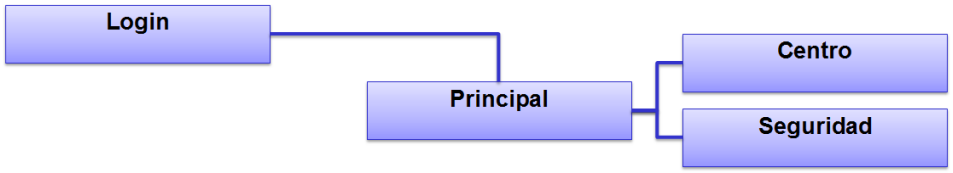


fig. 15: Mapa de navegación del sistema hasta el primer nivel.

La imagen anterior muestra el primer de acceso de los usuarios a la aplicación donde estos se autentican con las reglas previamente determinadas y acceden a la página principal donde se encuentran los módulos Centro y Seguridad. Una vez accedido al módulo Centro (ver figura 16) los usuarios tienen la posibilidad de gestionar los departamentos y los locales con sus categorías (ver figura 17) además podrán acceder a la pestaña recursos donde se gestionan las prestaciones de los mismos y se le asigna la clasificación de los recursos Activos Fijos tangibles y Útiles respectivamente (ver figura 18). También tendrán la facilidad de gestionar las incidencias ocurridas a los recursos de tipo Activo Fijo Tangible CPU, de administrar las trazas que no son más que los registros de navegación de los usuarios en la aplicación y podrán visualizar los reportes para ver los informes del estado de los recursos.

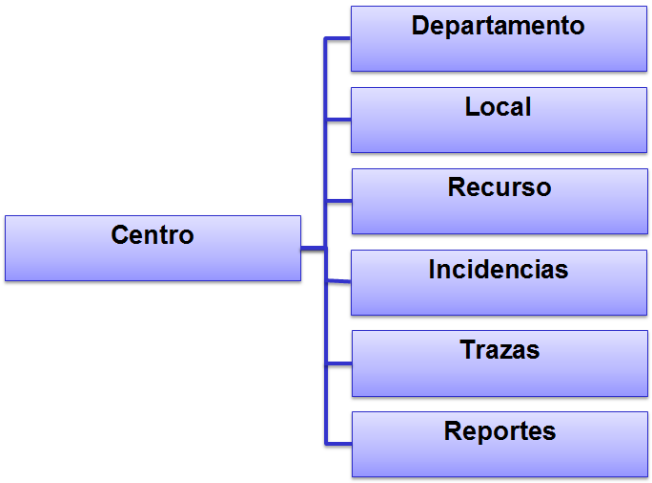


fig. 16: Fragmento del Mapa de navegación del sistema a partir del módulo Centro.

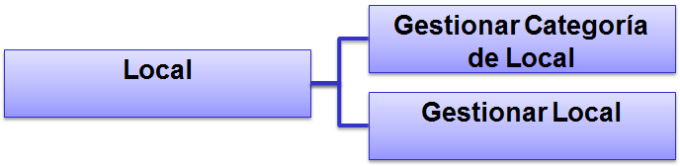


fig. 17: Fragmento del Mapa de navegación del sistema a partir de la pestaña Recurso

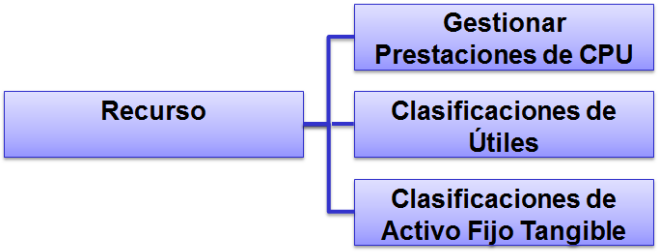


fig. 18: Fragmento del Mapa de navegación a partir de la pestaña Recurso

Por otra parte, los usuarios tienen la opción de acceder al módulo Seguridad donde estos pueden gestionar los roles y sus permisos correspondientes para navegar en la aplicación además de gestionar los usuarios asignándoles los roles previamente definidos.

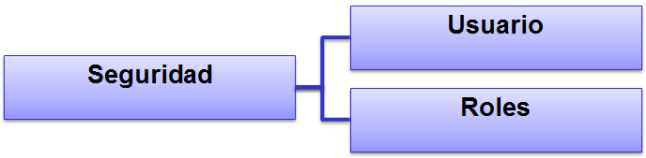


fig. 19: Fragmento del Mapa de navegación del sistema a partir del módulo Seguridad.

3.2.1 Funcionalidades del sistema

El Sistema para la gestión de información de incidencias, descargas y recursos del Centro de Tecnologías de Gestión de Datos v2.0 no permite que los usuarios puedan autenticarse por el dominio obligando al usuario generar sus datos localmente.

fig. 20: Autenticar Usuario.

La figura 20 muestra la interfaz gráfica del autentificar a los usuarios que deseen acceder a los servicios del Sistema Integrado de Soporte Del Centro DATEC. Posee tres campos no opcionales para la entrada de datos, correspondientes al usuario, la contraseña y el modo de autenticación. Para esto se definió un proveedor de autenticación personalizado el cual recoge los datos capturados del formulario. Si la autenticación del usuario es de manera local la validación se realiza en dependencia del usuario registrado en el sistema, de otra manera si el usuario se autentica por el dominio los datos se validan en el servidor del dominio accediendo al servicio LDAP de la universidad.

Cargar Prestación

En el Sistema para la gestión de información de incidencias, descargas y recursos del Centro de Tecnologías de Gestión de Datos v2.0 las prestaciones son insertadas por los usuarios además no permite la realización de auditorías de seguridad informática a las computadoras.

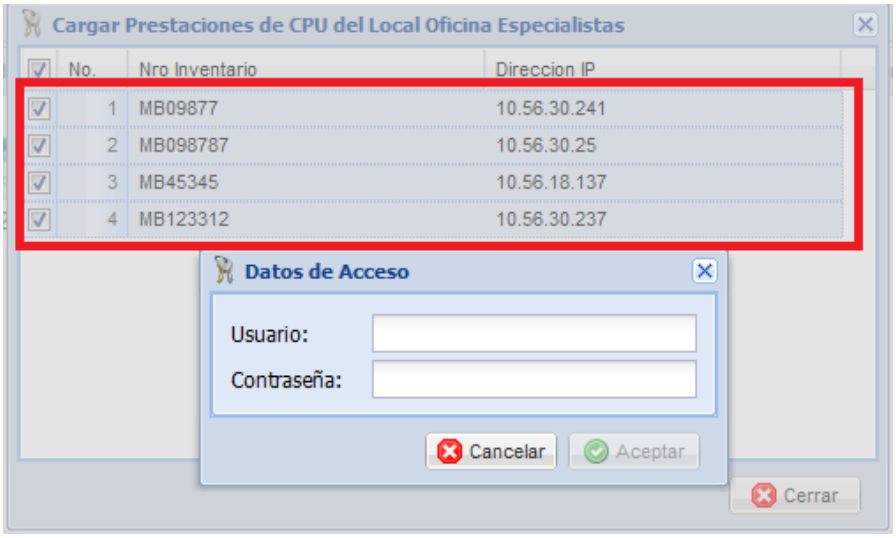


fig. 21: Cargar Prestaciones de CPU.

Actualmente en el Sistema Integrado de Soporte Del Centro DATEC, a los recursos de Activo Fijo Tangible CPU se les realizan auditorías de seguridad informática y se le actualizan dinámicamente las prestaciones, donde el usuario selecciona los recursos que desea actualizar o auditar, e introduce los datos de acceso enviándolos al servidor (ver figura 21). El servidor establece una conexión con las computadoras por el protocolo ssh y obtiene la información necesaria a través de la ejecución de un script en bash, dependiendo en gran medida del sistema operativo instalado. Durante el trascurso de la operación se muestra el progreso como se muestra en la figura 22. La implementación de estas funcionalidades permite al usuario un consumo no excesivo de sus horas de trabajo proporcionándole tiempo para el desarrollo de otras actividades.

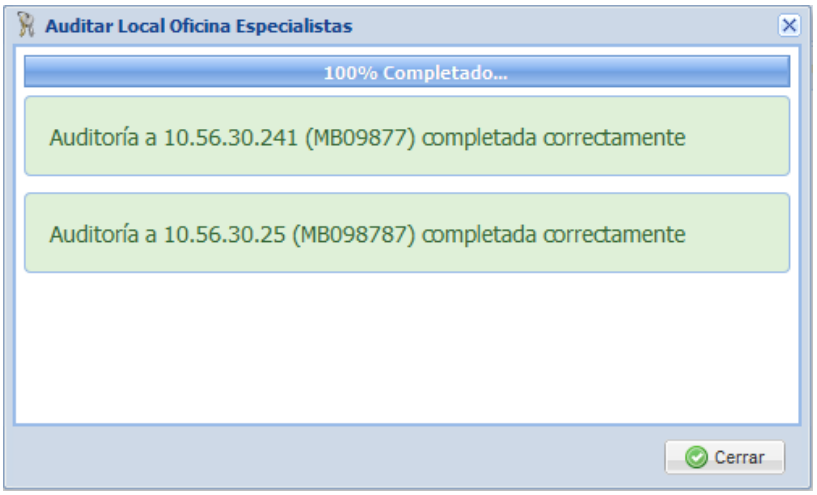


fig. 22: Auditar Local.

Visualizar Reportes

En el Sistema para la gestión de información de incidencias, descargas y recursos del Centro de Tecnologías de Gestión de Datos v2.0 los reportes se generan con TCPDF.

fig. 23: Visualizar Reportes

Actualmente los reportes en el Sistema Integrado de Soporte Del Centro DATEC se generan con JasperReports. Este permite el obtener los reportes en diferentes formatos como PDF, DOC y ODT entre otros posibilitando obtener una información determinada sin necesidad de una navegación excesiva en la aplicación.

3.3 Pruebas de Software

El único instrumento adecuado para determinar el estado y la calidad de un producto de *software* es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del *software* o al sistema de *software* en su totalidad, con el objetivo de medir el grado en que el *software* cumple con los requerimientos. Entre las pruebas que se le realizan al *software* están las pruebas de caja negra y la pruebas de rendimiento (Carga y *Stress*). (36)

3.3.1 Pruebas de Caja Negra

Las pruebas de caja negra denominadas pruebas de comportamientos, se centran en los requisitos funcionales del *software*. Estas permiten derivar conjuntos de condiciones de entrada que ejercitarán por completo todos los requisitos funcionales. Las pruebas de caja negra tratan de encontrar errores en las siguientes categorías:

- ✓ Funciones incorrectas o faltantes.
- ✓ Errores de Interfaz.
- ✓ Errores de estructura de datos o base de datos externas.
- ✓ Errores de comportamiento y desempeño.
- ✓ Errores de inicialización y término.

Existen diversos métodos de caja negra en los cuales se encuentra los métodos gráficos de prueba, análisis de valores límite y partición equivalente.

La partición equivalente es un método de prueba de caja negra que divide el dominio de la entrada de un programa en la clase de datos a partir de los cuales pueden derivarse casos de prueba. Se esfuerza por definir un caso de prueba que descubra ciertas clases de errores. Las clases de equivalencia se definen de acuerdo a las siguientes directrices:

- ✓ Si una condición de entrada especifica un rango, se definen una clase de equivalencia válida y dos no válidas.

- ✓ Si una condición de entrada requiere de un valor especifico, se definen una clase de equivalencia válida y dos no válida.
- ✓ Si una condición de entrada especifica un miembro de un conjunto, se define una clase de equivalencia válida y otra no válida.
- ✓ Si una condición de entrada es booleana, se define una clase de equivalencia válida y otra no válida. (37)

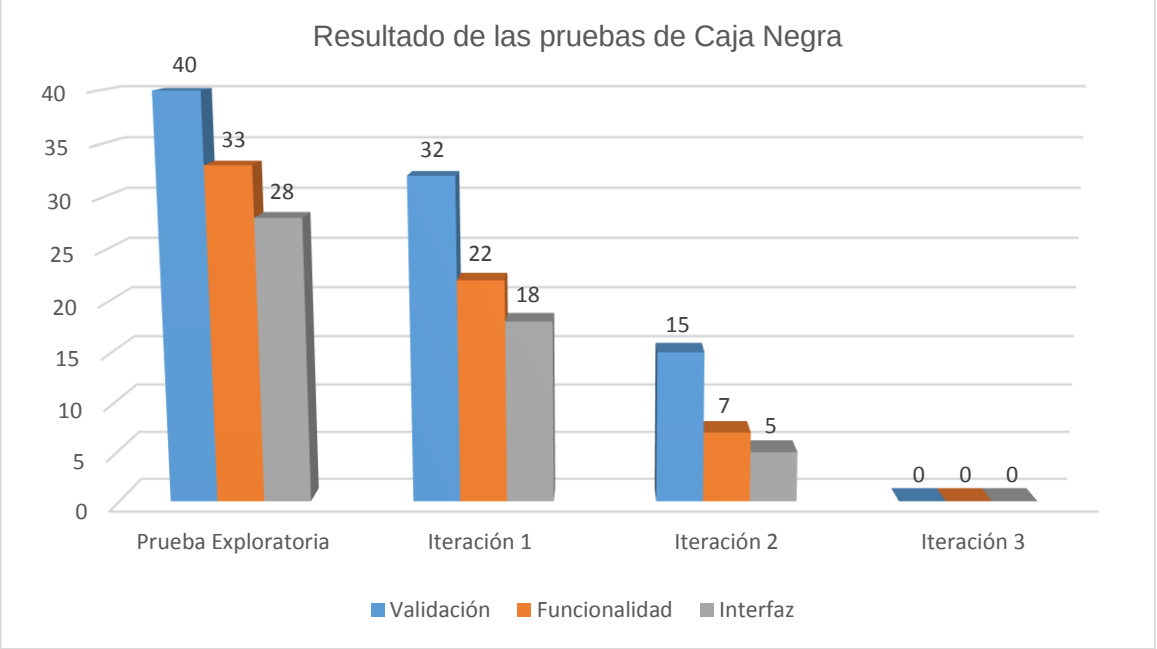
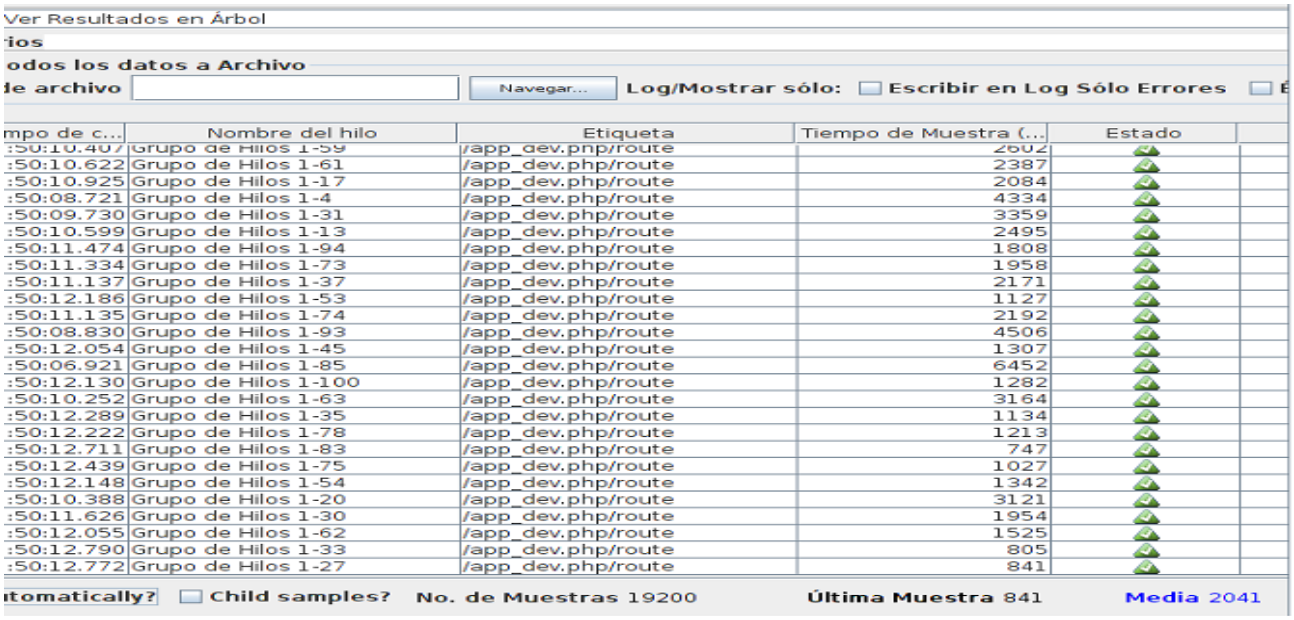


fig. 24: Resultados de la prueba de Caja Negra.

Después de realizar las pruebas de caja negra a través de los casos de prueba de los diferentes casos de usos, se realizó una prueba exploratoria en busca de no conformidades en el sistema. En el gráfico anterior se muestra los principales datos obtenidos durante la validación del sistema, donde se aprecia una prueba exploratoria en la que se encontraron 40 errores de no validación, 33 de funcionalidad y 28 de interfaz. En la primera y segunda iteración se corrigieron estos errores y al realizarse la tercera iteración no se encontraron errores de ningún tipo. Estas pruebas permitieron comprobar el correcto funcionamiento del módulo y la correcta validación de los campos, verificando que solo acepten los caracteres válidos.

3.3.2 Pruebas de rendimiento (Carga y Stress)

El objetivo de las pruebas de rendimiento es determinar si el usuario está satisfecho con la velocidad de la aplicación. Las pruebas de carga determinan el volumen de trabajo necesario para que el sistema funcione en hora punta y la de stress sirve para obtener datos, sobre la carga del sistema, que ayuden a realizar el dimensionamiento del sistema. (38) En esta tabla siguiente se puede observar que los tiempos de respuestas obtenidos son expresados en milisegundos (1000 milisegundos equivalen a 1 segundo).


Ver Resultados en Árbol

ios

odos los datos a Archivo

le archivo Navegar... Log/Mostrar sólo: Escribir en Log Sólo Errores

mpo de c...	Nombre del hilo	Etiqueta	Tiempo de Muestra (...	Estado
:50:10.407	Grupo de Hilos 1-59	/app_dev.php/route	2602	
:50:10.622	Grupo de Hilos 1-61	/app_dev.php/route	2387	
:50:10.925	Grupo de Hilos 1-17	/app_dev.php/route	2084	
:50:08.721	Grupo de Hilos 1-4	/app_dev.php/route	4334	
:50:09.730	Grupo de Hilos 1-31	/app_dev.php/route	3359	
:50:10.599	Grupo de Hilos 1-13	/app_dev.php/route	2495	
:50:11.474	Grupo de Hilos 1-94	/app_dev.php/route	1808	
:50:11.334	Grupo de Hilos 1-73	/app_dev.php/route	1958	
:50:11.137	Grupo de Hilos 1-37	/app_dev.php/route	2171	
:50:12.186	Grupo de Hilos 1-53	/app_dev.php/route	1127	
:50:11.135	Grupo de Hilos 1-74	/app_dev.php/route	2192	
:50:08.830	Grupo de Hilos 1-93	/app_dev.php/route	4506	
:50:12.054	Grupo de Hilos 1-45	/app_dev.php/route	1307	
:50:06.921	Grupo de Hilos 1-85	/app_dev.php/route	6452	
:50:12.130	Grupo de Hilos 1-100	/app_dev.php/route	1282	
:50:10.252	Grupo de Hilos 1-63	/app_dev.php/route	3164	
:50:12.289	Grupo de Hilos 1-35	/app_dev.php/route	1134	
:50:12.222	Grupo de Hilos 1-78	/app_dev.php/route	1213	
:50:12.711	Grupo de Hilos 1-83	/app_dev.php/route	747	
:50:12.439	Grupo de Hilos 1-75	/app_dev.php/route	1027	
:50:12.148	Grupo de Hilos 1-54	/app_dev.php/route	1342	
:50:10.388	Grupo de Hilos 1-20	/app_dev.php/route	3121	
:50:11.626	Grupo de Hilos 1-30	/app_dev.php/route	1954	
:50:12.055	Grupo de Hilos 1-62	/app_dev.php/route	1525	
:50:12.790	Grupo de Hilos 1-33	/app_dev.php/route	805	
:50:12.772	Grupo de Hilos 1-27	/app_dev.php/route	841	

tomatically? Child samples? No. de Muestras 19200 Última Muestra 841 Media 2041

fig. 25: Resultados de la prueba de Carga.

Las pruebas de rendimiento se realizaron con la herramienta de pruebas Jmeter 2.3.4. Para la ejecución de la prueba de carga se tomó una muestra de 100 personas obteniendo como resultado un tiempo que no sobrepasa los 3 segundos. Además para la ejecución de la prueba de stress se tomó una muestra de 250 obteniéndose como resultado un tiempo que no sobrepasa los 8 segundos.

3.4 Modelo de Despliegue

El diagrama de despliegue muestra las relaciones físicas entre los componentes hardware y software en el sistema final. Es un grafo de nodos unidos por conexiones de comunicación donde cada nodo puede contener instancias de componentes. En general un nodo puede ser una unidad de computación de algún tipo, desde un sensor hasta una computadora central y las instancias de componentes de *software* pueden estar unidas por relaciones de dependencia. (39)

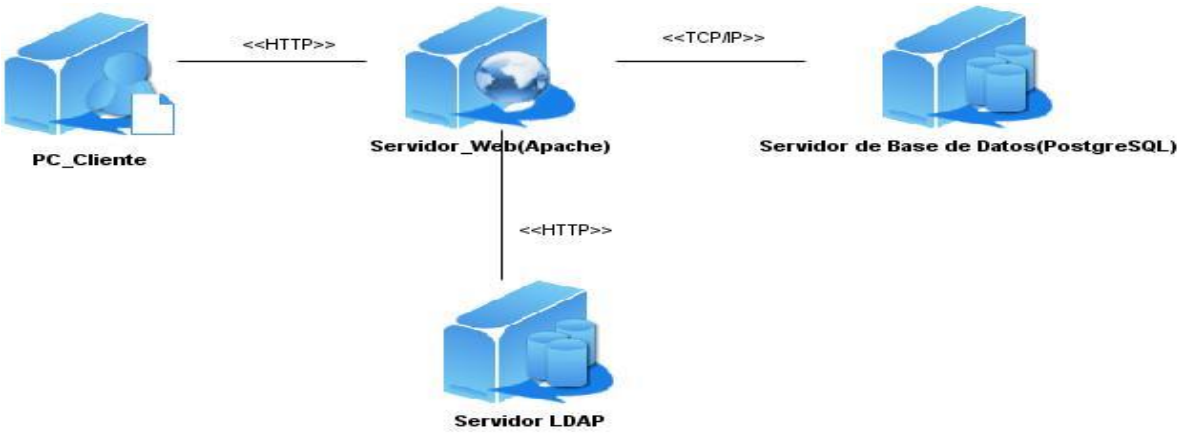


fig. 26: Modelo de Despliegue.

Descripción de los nodos

Nodo PC_Cliente: Ordenador desde donde el cliente accede al sistema.

Nodo Servidor Web: Ordenador donde se ejecuta el sistema.

Nodo Servidor de Base Datos: Almacena la información del que se obtiene del sistema.

Nodo Servidor LDAP: Base de datos donde se encuentran todos los usuarios de la universidad.

HTTP: Protocolo de transferencia utilizado para conectar la computadora del cliente con el servidor donde está el sistema y este último con el servidor de los usuarios de la universidad (LDAP).

TCP/IP: Protocolo para conectar el servidor de aplicaciones con las bases de datos.

Conclusiones

En este capítulo se presentó el diagrama de componentes del caso de uso significativo en análisis permitiendo mostrar las dependencias de los componentes físicos que conforman el sistema. Se realizó el mapa de navegación y se expusieron imágenes del sistema permitiendo una mejor navegabilidad en el sistema implementado. Se logró la implementación del sistema que posibilitó darle cumplimiento a los requisitos funcionales definidos. Para la validación del sistema se realizaron las pruebas de caja negra identificando en una prueba exploratoria 40 errores de no validación 33 de funcionalidad y 28 de interfaz que fueron resultas permitiendo mostrar un sistema listo para su ejecución. Se obtuvo la métrica del rendimiento del sistema mediante las pruebas de Carga y Stress para comprobar la funcionalidad del sistema.

CONCLUSIONES GENERALES

Con la realización de esta investigación se le dio cumplimiento a los objetivos previamente planteados:

- ✓ El estudio de los sistemas existentes en el mundo vinculados con la gestión de información permitió comprender la estructura de estos sistemas.
- ✓ El estudio y selección de la metodología, herramientas y tecnologías permitió sentar las bases para el desarrollo del Sistema Integrado de Soporte del Centro DATEC
- ✓ La realización del análisis y diseño del Sistema Integrado de Soporte del Centro DATEC permitió confeccionar los artefactos y diagramas correspondientes con la metodología escogida para un mejor entendimiento del negocio.
- ✓ La implementación del Sistema Integrado de Soporte del Centro DATEC dio cumplimiento a los 36 requisitos funcionales definidos, a los 12 requisitos funcionales modificados y los 5 no funcionales identificados.
- ✓ El diseño y puesta en práctica de las pruebas de software permitió comprobar el funcionamiento del Sistema Integrado de Soporte del Centro DATEC.
- ✓ Se obtuvo un Sistema Integrado de Soporte del Centro DATEC que centralizó las áreas de Grupo de Soporte y mejoró la ya implementada en el sistema anterior.

RECOMENDACIONES

Luego de haber analizado los resultados de este trabajo de diploma se hace las siguientes recomendaciones:

- ✓ Incorporar un módulo de métricas que permita evaluar indicadores a través de los datos recopilados por el sistema.
- ✓ Perfeccionar los algoritmos de las funcionalidades Auditar Local y Cargar Prestaciones utilizando los hilos de ejecución para mejorar el tiempo de respuesta.

REFERENCIAS BIBLIOGRÁFICAS

1. **Perez, Mario Montero.** Glossarium-Bltri. Glossarium-Bltri. [En línea] 18 de 11 de 2009. [Citado el: 24 de 11 de 2013.] http://glossarium.bitrum.unileon.es/Home/gestion-de-la-informacion..

2. **Omar, Demis Rojas.** Coleciones digital Unminuto. Coleciones digital Unminuto. [En línea] Corporación Universitaria un Minuto de Dios, 2012. [Citado el: 2013 de 11 de 28.] http://repository.uniminuto.edu:8080/jspui/handle/10656/1244..

3. **Peres, Ezequiel Martines.** IV CONGESO INTERNACIONAL INFORMATICA DE SALUD. IV CONGESO INTERNACIONAL INFORMATICA DE SALUD. [En línea] 2007. [Citado el: 23 de 11 de 2013.] www.bvs.hn/cu-2007/ponencias/SLD/SLD176.pdf. 4.

4. **Rodriguez, Kenia Riveron Ovalle y Yaidel.** IMPLEMENTACIÓN DEL MÓDULO DESPACHO DEL SUBSISTEMA INVENTARIO DEL SISTEMA INTEGRAL DE GESTIÓN CEDRUX. La Habana : s.n., 2007.

5. **Santana, Darlon.** Sistema para la gestión de incidencias, descargas y recursos del Centro DATEC . La Habana : s.n., 2013.

6. **Flores, Carmina Lizeth Torres.** Establecimiento de una Metodología de Desarrollo de Software. 2008.

7. **Cockburn, Alistair.** Agile Software Development. 2006.

8. **Object Management Group. .** Object Management Group. . Object Management Group. . [En línea] 14 de 02 de 2014. http://www.uml.org/..

9. **Presman, Roger S.** Ingeneria de Software, un enfoque práctico. 2007.

10. **Gutierrez, Jorge A Saavedra.** El mundo Informático. El mundo Informático. [En línea] [Citado el: 3 de 2 de 2014.] http://jorgesaavedra.wordpress.com/about/. .

11. **Vasquez, Mariño Carlos.** Programación en PHP5. Nivel Básico. 2008.

12. **Perez, javier Eguíluz.** Introducción a JavaScript. Introducción a JavaScript. [En línea] 25 de 03 de 2009. [Citado el: 02 de 03 de 2014.] http://librosweb.es/javascript/..

13. **Certad, Claudio Concepción.** fraterneo GNU/Linux. fraterneo GNU/Linux. [En línea] [Citado el: 20 de 02 de 2014.] http://fraterneo.blogspot.com/2011/01/que-es-bash.html .

14. **Pontencier, Fabier.** Synfony, la guía definitiva. 2008.

15. CodeManía. CodeManía. [En línea] 09 de 2013. [Citado el: 20 de 03 de 2014.] http://codemania.cubava.cu/2013/09/que-es-extjs/.. .

16**. Lobo, Armando Robert.** Documento Vision. La Habana : s.n., 2007.

REFERENCIAS BIBLIOGRÁFICAS

17. [En línea] [Citado el: 02 de 03 de 2013.] http://linux.ciberaula.com/articulo/linux_apache_intro.

18. **Menendez, Rafael.** Informática Base de Datos. [En línea] 09 de 01 de 2000. [Citado el: 05 de 04 de 2014.] http://www.um.es/docencia/barzana/DIVULGACION/INFORMATICA/sgbd.html. .

19. POSTGRESQL. POSTGRESQL. [En línea] [Citado el: 07 de 03 de 2014.] http://www.postgresql.org/about/. .

20. NetBeans. NetBeans. [En línea] [Citado el: 07 de 03 de 2014.] https://netbeans.org/community/releases/74/ .

21**. Ravelo, Raidel Oceguera.** Generador dinámico de reportes. La Habana : s.n., 2012**.**

22. JasperSof Corporation. JasperSof Corporation. JasperSof Corporation. [En línea] 15 de 03 de 2014. http://www.jaspersoft.com/es/node/38063.

23. OpenSSH. OpenSSH. [En línea] 16 de 03 de 2014. [Citado el: 20 de 05 de 2014.] http://www.openssh.com/.

24. OsmosisLatina. OsmosisLatina. [En línea] 20 de 10 de 2005. [Citado el: 05 de 04 de 2014.] http://www.osmosislatina.com/jmeter/basico.htm**. .**

25. **Larman, Graig.** Modelo del Dominio. 2003.

26. **Presman, Roger S.** Ingeniería de Software: un enfoque práctico. 2003 : s.n**.**

27. **Somerville, Ian.** Ingeniería de Software 7ma Edicion. 2011.

28. **Tello, Jesus Caceres.** [En línea] [Citado el: 07 de 04 de 2014.] http://www2.uah.es/jcaceres/capsulas/DiagramaCasosDeUso.pdf**.**

29. **Varela, Felix.** UML y Patrones. La Habana : s.n., 2004**.**

30. **Larman, Graig.** UML y Patrones. 2003.

31. Ingeniería de Software 7ma Edición. Ingeniería de Software 7ma Edición. [En línea] 2012. [Citado el: 07 de 04 de 2014.] http://www.filecrop.com/ian-sommerville-ingenieria-de-software-septima-edicion.pdf.html.

32. Popkin Software and Systems. Popkin Software and Systems. [En línea] [Citado el: 04 de 03 de 2014.] ttp://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/doc-modelado-sistemas-uml.pdf..

33. ALEGSA.COM.AR. DICCIONARIO DE INFORMÁTICA. DICCIONARIO DE INFORMÁTICA. [En línea] [Citado el: 02 de 04 de 2014.] http://www.alegsa.com.ar/Dic/modelo%20de%20datos%20de%20base%20de%20datos.php.

34. [En línea] [Citado el: 10 de 04 de 2014.] virtual.usalesiana.edu.bo/web/practica/archiv/componen1.ppt.

35. Mapa de Navegacion. Mapa de Navegacion. [En línea] [Citado el: 10 de 05 de 2014.] http://www.arquitecturadeinformacion.cl/como/mapa.html ..

36. Gestión de Calidad y Puebas de Software. Gestión de Calidad y Puebas de Software. [En línea] 2005. [Citado el: 10 de 05 de 2014.] http://www.pruebasdesoftware.com/laspruebasdesoftware.htm..

37. **Pressman, Roger.** Tecnicas de Prueba. . s.l. : Vol. Cap 14, parte 1.

38. Globe TESTIN. Globe TESTIN. [En línea] [Citado el: 29 de 04 de 2014.] http://www.globetesting.com/pruebas-de-rendimiento/ .

39. **Vilas, Ana Fernandez.** Diagrama de Despliegue. Diagrama de Despliegue. [En línea] http://www-gris.det.uvigo.es/~avilas/UML/node50.html.

BIBLIOGRAFÍAS

1. **Perez, Mario Montero.** Glossarium-Bltri. Glossarium-Bltri. [En línea] 18 de 11 de 2009. [Citado el: 24 de 11 de 2013.] http://glossarium.bitrum.unileon.es/Home/gestion-de-la-informacion..

2. **Omar, Demis Rojas.** Coleciones digital Unminuto. Coleciones digital Unminuto. [En línea] Corporación Universitaria un Minuto de Dios, 2012. [Citado el: 2013 de 11 de 28.] http://repository.uniminuto.edu:8080/jspui/handle/10656/1244..

3. **Peres, Ezequiel Martines.** IV CONGESO INTERNACIONAL INFORMATICA DE SALUD. IV CONGESO INTERNACIONAL INFORMATICA DE SALUD. [En línea] 2007. [Citado el: 23 de 11 de 2013.] www.bvs.hn/cu-2007/ponencias/SLD/SLD176.pdf. 4.

4. **Rodriguez, Kenia Riveron Ovalle y Yaidel.** IMPLEMENTACIÓN DEL MÓDULO DESPACHO DEL SUBSISTEMA INVENTARIO DEL SISTEMA INTEGRAL DE GESTIÓN CEDRUX. La Habana : s.n., 2007.

5. **Santana, Darlon.** Sistema para la gestión de incidencias, descargas y recursos del Centro DATEC . La Habana : s.n., 2013.

6. **Flores, Carmina Lizeth Torres.** Establecimiento de una Metodología de Desarrollo de Software. 2008.

7. **Cockburn, Alistair.** Agile Software Development. 2006.

8. **Object Management Group. .** Object Management Group. . Object Management Group. . [En línea] 14 de 02 de 2014. http://www.uml.org/..

9. **Presman, Roger S.** Ingeneria de Software, un enfoque práctico. 2007.

10. **Gutierrez, Jorge A Saavedra.** El mundo Informático. El mundo Informático. [En línea] [Citado el: 3 de 2 de 2014.] http://jorgesaavedra.wordpress.com/about/. .

11. **Vasquez, Mariño Carlos.** Programación en PHP5. Nivel Básico. 2008.

12. **Perez, javier Eguíluz.** Introducción a JavaScript. Introducción a JavaScript. [En línea] 25 de 03 de 2009. [Citado el: 02 de 03 de 2014.] http://librosweb.es/javascript/..

13. **Certad, Claudio Concepción.** fraterneo GNU/Linux. fraterneo GNU/Linux. [En línea] [Citado el: 20 de 02 de 2014.] http://fraterneo.blogspot.com/2011/01/que-es-bash.html .

14. **Pontencier, Fabier.** Synfony, la guía definitiva. 2008.

15. CodeManía. CodeManía. [En línea] 09 de 2013. [Citado el: 20 de 03 de 2014.] http://codemania.cubava.cu/2013/09/que-es-extjs/.. .

16**. Lobo, Armando Robert.** Documento Vision. La Habana : s.n., 2007.

17. [En línea] [Citado el: 02 de 03 de 2013.] http://linux.ciberaula.com/articulo/linux_apache_intro.

18. **Menendez, Rafael.** Informática Base de Datos. [En línea] 09 de 01 de 2000. [Citado el: 05 de 04 de 2014.] http://www.um.es/docencia/barzana/DIVULGACION/INFORMATICA/sgbd.html. .

19. POSTGRESQL. POSTGRESQL. [En línea] [Citado el: 07 de 03 de 2014.] http://www.postgresql.org/about/. .

20. NetBeans. NetBeans. [En línea] [Citado el: 07 de 03 de 2014.] https://netbeans.org/community/releases/74/ .

21**. Ravelo, Raidel Oceguera.** Generador dinámico de reportes. La Habana : s.n., 2012**.**

22. JasperSof Corporation. JasperSof Corporation. JasperSof Corporation. [En línea] 15 de 03 de 2014. http://www.jaspersoft.com/es/node/38063.

23. OpenSSH. OpenSSH. [En línea] 16 de 03 de 2014. [Citado el: 20 de 05 de 2014.] http://www.openssh.com/.

24. OsmosisLatina. OsmosisLatina. [En línea] 20 de 10 de 2005. [Citado el: 05 de 04 de 2014.] http://www.osmosislatina.com/jmeter/basico.htm**. .**

25. **Larman, Graig.** Modelo del Dominio. 2003.

26. **Presman, Roger S.** Ingeniería de Software: un enfoque práctico. 2003 : s.n**.**

27. **Somerville, Ian.** Ingeniería de Software 7ma Edicion. 2011.

28. **Tello, Jesus Caceres.** [En línea] [Citado el: 07 de 04 de 2014.] http://www2.uah.es/jcaceres/capsulas/DiagramaCasosDeUso.pdf**.**

29. **Varela, Felix.** UML y Patrones. La Habana : s.n., 2004**.**

30. **Larman, Graig.** UML y Patrones. 2003.

31. Ingeniería de Software 7ma Edición. Ingeniería de Software 7ma Edición. [En línea] 2012. [Citado el: 07 de 04 de 2014.] http://www.filecrop.com/ian-sommerville-ingenieria-de-software-septima-edicion.pdf.html.

32. Popkin Software and Systems. Popkin Software and Systems. [En línea] [Citado el: 04 de 03 de 2014.] ttp://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/doc-modelado-sistemas-uml.pdf..

33. ALEGSA.COM.AR. DICCIONARIO DE INFORMÁTICA. DICCIONARIO DE INFORMÁTICA. [En línea] [Citado el: 02 de 04 de 2014.] http://www.alegsa.com.ar/Dic/modelo%20de%20datos%20de%20base%20de%20datos.php.

34. [En línea] [Citado el: 10 de 04 de 2014.] virtual.usalesiana.edu.bo/web/practica/archiv/componen1.ppt.

35. Mapa de Navegacion. Mapa de Navegacion. [En línea] [Citado el: 10 de 05 de 2014.] http://www.arquitecturadeinformacion.cl/como/mapa.html ..

36. Gestión de Calidad y Puebas de Software. Gestión de Calidad y Puebas de Software. [En línea] 2005. [Citado el: 10 de 05 de 2014.] http://www.pruebasdesoftware.com/laspruebasdesoftware.htm..

37. **Pressman, Roger.** Tecnicas de Prueba. . s.l. : Vol. Cap 14, parte 1.

38. Globe TESTIN. Globe TESTIN. [En línea] [Citado el: 29 de 04 de 2014.] http://www.globetesting.com/pruebas-de-rendimiento/ .

39. **Vilas, Ana Fernandez.** Diagrama de Despliegue. Diagrama de Despliegue. [En línea] http://www-gris.det.uvigo.es/~avilas/UML/node50.html.