Universidad de las Ciencias Informáticas

Facultad de Ciencias y Tecnologías Computacionales

Sistema para la Informatización del proceso de Auditoría y Control de los Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autor: Tania Ramírez Ramírez

Tutor: M.Sc. Omar Mar Cornelio

Ing. Bernardo Hernández González

La Habana, junio de 2017

Declaración de Autoría

Declaro por este medio que yo, Tanía Ramírez Ramírez, con carné de identidad 94100716430 soy la autora de este trabajo y que autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste, firmamos la presente declaración jurada de autoría en La Habana a los __ días del mes de junio del año 2017.

Ing. Bernardo Hernández González

Tutor

M.Sc. Omar Mar Cornelio

Tutor

Tania Ramírez Ramírez

Autor

Datos de Contacto

Autores:
Tania Ramírez Ramírez
Universidad de las Ciencias Informáticas (UCI)
e-mail: tramirez@estudiantes.uci.cu

Tutores:
M.Sc Omar Mar Cornelio
Universidad de las Ciencias Informáticas (UCI)
e-mail: omarmar@uci.cu

Ing. Bernardo Hernández González
Universidad de las Ciencias Informáticas (UCI)
e-mail: bhernandez@uci.cu

Dedicatoria

Dedico este trabajo a mis padres, los que me convirtieron en la persona que soy hoy y me guiaron siempre por el buen camino, los amo. A mi familia, por brindarme su apoyo y estar siempre presente. A mis amistades que compartieron conmigo estos cinco años de universidad, los cuales serán inolvidables. A mis tutores, por todo el apoyo y conocimiento, por acompañarme durante todo el proceso. A todas las personas que de una forma u otra ayudaron en la realización de este trabajo.

Resumen

Como parte de las normas del sistema de control interno cubano, se establece que el 10% de los Activos Fijos Tangibles (AFT) sean auditados mensualmente, representando tarea medular dentro de cualquier entidad. En la Facultad de Ciencias y Tecnologías Computacionales de la Universidad de las Ciencias Informáticas, se realiza a través del testeo manual de los medios contra un inventario previamente impreso, lo que genera un proceso engorroso e ineficiente. La investigación que a continuación se presenta consiste en el análisis, diseño, implementación y evaluación de una aplicación informática encaminada a agilizar el proceso de Auditoría y Control de los Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales. El desarrollo del sistema se encuentra marcado por el empleo de tecnologías y herramientas de actualidad apegadas a la política de soberanía tecnológica que impulsa el país. La solución obtenida mejora el empleo de los recursos humanos y materiales, permitiendo obtener resultados de manera rápida y fiable a partir de la realización del proceso de auditoría y control.

Palabras clave: aplicación informática, auditoría y control, activo fijo tangible.

Abstract

As part of the rules of the Cuban internal control system, it is established that 10% of Fixed Tangible Assets (AFT) be audited monthly, representing a core task within any entity. In the School of Sciences and Computational Technologies of the University of the Computer Science, it is done through the manual testing of the resources with a previously printed inventory, which generates a cumbersome and inefficient process. The research that follows is the analysis, design, implementation and evaluation of a computer application aimed at streamlining the process of Auditing and Control of Fixed Tangible Assets in the Faculty of Sciences and Computational Technologies. The development of the system is marked by the use of current technologies and tools attached to the policy of technological sovereignty that drives the country. The solution obtained improves the use of human and material resources, allowing results to be obtained quickly and reliably from the completion of the audit and control process.

Keywords: computer application, audit and control, tangible fixed assets.

Índice de Contenido

Introducción 10

Capítulo 1: Fundamentación Teórica del Proceso de Auditoría y Control de los Activos Fijos Tangibles 15

1.1 Conceptos asociados al dominio del problema 15

1.1.1 Activos Fijos Tangibles 15

1.1.2 Auditoría y Control 16

1.1.3 Aplicación Informática 18

1.2 Caracterización del Objeto de Estudio 20

1.2.1 El proceso de Auditoría y Control de los AFT en Cuba 20

1.2.2 El proceso de Auditoría y Control de los AFT en la UCI 21

1.3 Análisis de soluciones similares 23

1.4 Metodología de Desarrollo de Software 27

1.4.1 UML 2.0: el lenguaje de soporte a la metodología seleccionada 30

1.4.2 Visual Paradigm for UML 8.0: la herramienta CASE 31

1.5 Sistema Gestor de Bases de Datos 31

1.5.1 PostgreSQL 9.5: el Sistema Gestor de Bases de Datos 32

1.6 Lenguaje de Programación a utilizar 33

1.7 Entorno Integrado de Desarrollo 35

1.8 Conclusiones parciales 36

Capítulo 2: Descripción y Diseño de la Solución Propuesta 37

2.1 Propuesta de solución 37

2.2 Modelo de negocio 38

2.2.1 Reglas del negocio a considerar 38

2.2.2 Actores y trabajadores del negocio 39

2.2.3 Diagrama de Casos de Uso del Negocio 39

2.2.4 Descripción textual del Caso de Uso del Negocio "Realizar Auditoría" 40

2.2.5 Diagrama de actividades del Caso de Uso del Negocio: "Realizar Auditoría" 41

2.3 Requisitos 42

2.3.1 Requisitos funcionales 42

2.3.2 Requisitos no funcionales 43

2.4 Modelo de sistema 44

2.4.1 Actores del sistema 44
2.4.2 Diagrama de Casos de Uso del Sistema 45
2.4.3 Descripción textual de los Casos de Uso del Sistema 46
2.4.4 Patrones de Casos de Uso utilizados 48
2.5 Elementos de la arquitectura 49
2.5.1 Patrones arquitectónicos y de diseño 49
2.5.2 Patrones de Diseño utilizados 51
2.6 Modelo de Diseño 55
2.6.1 Diagrama de Clases del Diseño 55
2.7 Diagrama Entidad Relación 57
2.8 Conclusiones Parciales 58
Capítulo 3: Implementación y Validación de la Solución Propuesta 59
3.1 Modelo de Implementación 59
3.1.1 Representación de dependencias entre componentes de software 59
3.2 Modelo de Despliegue 60
3.3 Estándares de Codificación 61
3.4 El Proceso de Pruebas 63
3. 5 Tipos de Pruebas 64
3.5.1 Diseño de casos de prueba 64
3.5.2 Aceptación de la propuesta de solución 67
3. 6 Ejecución de las pruebas. Resultados 68
3. 7 Conclusiones Parciales 69
Conclusiones Generales 70
Recomendaciones 71
Referencias Bibliográficas 72
Anexos **¡Error! Marcador no definido.**

Índice de Tablas

Tabla. 1 Actores del negocio. 39
Tabla. 2 Trabajadores del negocio. 39
Tabla. 3 Descripción textual del CUN "Realizar Auditoría". 40
Tabla. 4 Actores del Sistema. 45
Tabla. 5 Descripción textual del Caso de Uso del Sistema: "Realizar Auditoria". 46
Tabla. 6 Descripción de los nodos correspondientes al diagrama de despliegue del sistema. 61
Tabla. 7 Secciones a probar en el Caso de Uso del Sistema: "Realizar Auditoría". 65
Tabla. 8 Descripción de las Variables. 66
Tabla. 9 Matriz de datos: Realizar Auditoría. 66

Índice de Figuras

Fig. 1 Logotipo de la aplicación ContarERP. 23
Fig. 2 Interfaz de Bienvenida de VERSAT Sarasola. 24
Fig. 3 Ciclo de vida de la metodología Open Up. 29
Fig. 4 Maqueta de la propuesta de solución. 37
Fig. 5 Diagrama de Casos de Uso del Negocio. 40
Fig. 6 Diagrama de Actividades del Caso de Uso del Negocio: "Realizar Autoría". 41
Fig. 7 Diagrama de Casos de Uso del Sistema. 46
Fig. 8 Patrón de Diseño Modelo-Vista-Controlador. 50
Fig. 9 Patrón de diseño Objeto de Acceso a Datos (DAO). 54
Fig. 10 Diagrama de Clases del Diseño de la Propuesta de Solución. 56
Fig. 11 Diagrama Entidad Relación de la Propuesta de Solución. 57
Fig. 12 Diagrama de componentes. Caso de Uso: Realizar Auditoría. 60
Fig. 13 Diagrama de Despliegue del Sistema. 60
Fig. 14 Fragmento de código de la propuesta de solución. 63
Fig. 15 Ejecución de las pruebas de caja negra sobre la aplicación. 69

Introducción

Desde los propios cimientos de las sociedades antiguas el ser humano ha tenido la necesidad de controlar sus pertenencias y las del grupo del cual forma parte, por lo que de alguna manera se tenían tipos de control para evitar desfalcos. Con el surgimiento de las relaciones económicas y su evolución en sistemas cada vez más complejos los mecanismos para preservar los bienes han enfrentado desafíos, desarrollándose y jugando un papel protagónico dentro de los más diversos tipos de organizaciones.

Se dice que el control interno es una herramienta surgida de la imperiosa necesidad de accionar proactivamente a los efectos de suprimir y/o disminuir significativamente la multitud de riesgos a las cuales se hayan afectadas los distintos tipos de organizaciones, sean estos privados o públicos, con o sin fines de lucro.

Debido a la importancia que ameritan los temas de control, en agosto de 2009 la Asamblea Nacional del Poder Popular en Cuba, crea la Contraloría General de la República, la que tiene entre sus funciones establecidas, según lo señalado en el Artículo 31 inciso l), normar, supervisar, evaluar los sistemas de control interno; manifestar las recomendaciones necesarias para su mejoramiento y perfeccionamiento continuo(Cuba, 2012). Los Activos Fijos Tangibles (en lo adelante AFT) representan las propiedades físicamente tangibles con que cuenta una empresa o entidad, con el fin de utilizarse por un período largo en sus operaciones regulares y que normalmente no se destinan a la venta (Jara, 2014).

La Universidad de las Ciencias Informáticas (en lo adelante UCI) es un centro cubano de altos estudios con la misión estratégica en el país de la formación de profesionales de alta calidad y la producción de software. El sistema de control interno constituye para la institución garantía de apego a las normas legales respecto a todas las actividades relevantes del centro, representando el mecanismo fundamental de salvaguarda de los recursos o bienes que utiliza de manera continua en el curso de sus operaciones. La preservación y control de los AFT se vuelve tarea primordial en aras de lograr indicadores de eficacia y productividad en muchos de los procesos sustantivos de la entidad, recomendándose como parte del mencionado sistema de control, la auditoría de al menos el 10% de los AFT de manera mensual.

En la Facultad de Ciencias y Tecnologías Computacionales (en lo adelante CITEC) perteneciente a la UCI esta actividad se incluye dentro de la gestión administrativa e involucra a todas las áreas docentes,

productivas y de servicio. Algunos inconvenientes hoy limitan la calidad y agilidad con la que se lleva a efecto la auditoría y control de los AFT conforme lo previsto en el sistema de control interno.

El proceso de auditoría mensual se realiza a partir del nivel que cubra la revisión, siendo el responsable del área o centro de costo dentro de la facultad el encargado de obtener a partir del Sistema de Gestión Universitaria (SGU en lo adelante) un listado impreso de los AFT bajo su responsabilidad. Un responsable designado o el propio jefe de área o centro de costo lleva a cabo un testeo manual de los medios, realizando una comparación entre cifras y estado de los activos, debiendo comprobarse su existencia y características. Este modus operandi torna el proceso engorroso e ineficiente con gastos significativos de recursos humanos y materiales, provocando la imposibilidad de cumplir con la cantidad de medios que representa el 10%. Lo anteriormente planteado se reafirma si se tiene en cuenta que la Facultad CITEC cuenta con más de 620 estaciones de trabajo destinadas a los procesos docente-productivos y una aproximación de 2408 AFT en total.

Por otra parte la forma en la que tiene lugar el proceso no facilita la generación de informes resultantes de las auditorías realizadas, los cuales son realizados de forma manual, lo que introduce una incertidumbre en el dato a partir de un posible error humano. Todo lo anterior limita la toma de decisiones y la calidad y disponibilidad de la información que se presenta, lo que provoca que se plantee el siguiente problema a resolver, el cual origina la investigación:

¿Cómo agilizar el proceso de Auditoría y Control de los Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales de la Universidad de las Ciencias Informáticas?

De acuerdo al problema planteado, esta investigación pretende como objetivo general **Desarrollar una aplicación informática para gestionar el proceso de auditoría y control de los Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales.**

Como objeto de estudio se propone **el proceso de auditoría y control de los activos fijos tangibles en Cuba** y como campo de acción enmarcado en dicho objeto **la informatización del proceso de auditoría y control de los Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales.**

Para guiar el proceso investigativo se plantean las siguientes **preguntas de investigación**:

1. ¿Cuáles son los referentes teóricos del proceso de auditoría y control de los activos fijos tangibles en la Facultad de Ciencias y Tecnologías Computacionales?
2. ¿Qué técnicas, tecnologías y herramientas de desarrollo resultan adecuadas para implementar la solución propuesta?
3. ¿Qué funcionalidades debe tener la aplicación para poder contribuir al proceso de auditoría y control de los activos fijos tangibles en la Facultad de Ciencias y Tecnologías Computacionales?
4. ¿Qué tipos de pruebas se pueden utilizar en la validación de la propuesta de solución y cómo se deben aplicar?

Con el fin de dar cumplimiento al objetivo general propuesto se definen una serie de **Tareas de la Investigación**:

1. Caracterización del proceso de auditoría y control de los AFT en Cuba y en la Facultad CITEC.
2. Valoración de las soluciones existentes que responden al problema de la investigación en alguna medida, sus limitaciones y fortalezas.
3. Caracterización de las principales herramientas, tecnologías, lenguajes y metodologías a utilizar para la construcción de la propuesta de solución.
4. Realización del análisis y el diseño de la solución propuesta.
5. Implementación de la solución propuesta.
6. Validación del sistema implementado.

En la presente investigación se utilizan los siguientes **métodos científicos**:

Dentro de los teóricos:

Histórico-Lógico: El método histórico estudia la trayectoria real de los fenómenos y acontecimientos en el transcurso de su historia. El método lógico investiga las leyes generales del funcionamiento y desarrollo de los fenómenos(Zayas, 1997).

Este método se utiliza para estudiar la evolución de los conceptos asociados a los AFT y los procesos de auditoría y control que tienen lugar sobre los mismos, permitiendo la definición de términos propios.

Analítico-Sintético: Permite la descomposición de un todo complejo en sus partes y cualidades. La síntesis, por su parte, establece la unión entre las partes, previamente analizadas y posibilita descubrir relaciones y características generales entre los elementos de la realidad(Zayas, 1997).

Este método se utiliza para la evaluación de soluciones que respondan al problema y permite realizar una valoración crítica y detallada de cada una de ellas. Se utiliza, además, para seleccionar las herramientas y tecnologías a utilizar durante el desarrollo del sistema.

Dentro de los empíricos:

Estos métodos nos permiten extraer de los fenómenos analizados las informaciones que se necesitan sobre ellos a través de observaciones, del uso de técnicas opináticas y la propia experimentación.

Se hace uso del método de la **Entrevista**, aplicado a directivos del área de Economía y Administración de la Facultad CITEC con el fin de comprender y determinar el proceso actual de auditoría y control de los AFT en el área en cuestión.

La **Observación** como método empírico de la investigación científica se emplea para formular y proponer mecanismos y medios que faciliten la realización del proceso de testeo a partir del seguimiento del mismo en diferentes ámbitos.

Como parte del proceso de pruebas en busca de la validación del sistema se pone de manifiesto el empleo del **Experimento** como medio de validar que la solución cumple con los requisitos tanto funcionales como no funcionales y se adecua a la solución del problema a resolver.

La investigación se estructura en 3 capítulos cuyos contenidos esenciales se describen a continuación:

Capítulo 1: En este capítulo se especifican y se explican los principales conceptos asociados a la auditoría y control. Se describe el proceso de auditoría y control de los activos fijos tangibles como objeto de estudio y se valoran y critican las soluciones actuales que de alguna manera ofrecen respuesta al problema en cuestión. Así mismo se definen, argumentan y valoran las principales herramientas, tecnologías, metodologías y/o lenguajes que se utilizan para la construcción de la solución.

Capítulo 2: En este capítulo se comienza la construcción de la solución según la metodología de desarrollo seleccionada, se definen el Modelo de Negocio, los Requisitos Funcionales y No funcionales y se describen los Casos de Uso del Sistema. Finalmente, se presentan los artefactos referidos a la etapa de diseño de la solución.

Capítulo 3: Este capítulo describe el proceso de implementación y pruebas de la solución, su lectura permitirá conocer un panorama general del proceso de pruebas, los tipos de prueba recomendados sobre la tipología de la solución propuesta así como la selección, diseño y aplicación de las mismas. Finalmente se muestran los resultados obtenidos a partir de la ejecución de las pruebas.

Capítulo 1: Fundamentación Teórica del Proceso de Auditoría y Control de los Activos Fijos Tangibles

En este capítulo se describen los principales aspectos que caracterizan la auditoría y control de AFT como parte del fundamento teórico; se abordan los elementos básicos referentes a las aplicaciones informáticas además de formalizarse los conceptos más significativos asociados al objeto de automatización o negocio: el proceso de auditoría y control de los activos fijos tangibles. Finalmente, se valoran las posibles soluciones similares y la descripción de las tendencias actuales en cuanto a metodologías, herramientas y tecnologías que se emplean en el desarrollo e implementación del sistema informático que se propone como solución a la problemática existente.

1.1 Conceptos asociados al dominio del problema

1.1.1 Activos Fijos Tangibles

La manera en que son tratados hoy en día los bienes utilizados en las empresas difiere de manera significativa en la forma que se manejaban hace algunos años. No fue sino el desarrollo de las ciencias contables y empresariales lo que conllevó al cambio en la variante con que eran denominados los bienes o servicios de las entidades. El término activo fijo surge para facilitar el manejo fundamentalmente documental sobre los bienes que posee una empresa, teniendo en cuenta que la pérdida, deterioro o el uso indebido del mismo afectan directamente la economía de la entidad.

Los activos fijos son bienes adquiridos con la finalidad de ser usados en la producción, brindar un servicio o de uso administrativo en la empresa(Jara, 2014). El término activo fijo expresa claramente que son bienes que permanecen fijos dentro de la organización. Caso contrario el activo circulante son bienes que constantemente están en movimiento dentro de las actividades de la empresa. Los activos fijos no pueden convertirse en líquido[1] a corto plazo, su costo incluye el total del valor de compra más todos los gastos necesarios para tener el activo en el lugar y condiciones que permitan su funcionamiento, pudiendo

[1] Activo líquido: son definidos como aquellos que pueden convertirse en corto plazo en dinero en efectivo sin perder valor y que siendo bienes sin tener postergación se los puede transformar en efectivo.

considerarse como un paquete de servicios potenciales que se van consumiendo gradualmente en el transcurso de varios años. Se clasifican en dos grande grupos, los tangibles e intangibles.

Los activos fijos intangibles son aquellos que se utilizan en la operación del negocio pero que no tiene sustancia física y no son corrientes. Como ejemplo están las patentes, los derechos del autor, las marcas registradas entre otros(Jara, 2014).

Los activos fijos tangibles hacen referencia a un bien físico, verificable, de los cuáles se espera que prestarán servicios según sea su naturaleza. Estos se clasifican a su vez en los siguientes grupos:

- Bienes no sujetos a depreciación: son aquellos que no disminuyen su valor con el paso del tiempo, en algunos casos aumentan su valor, ejemplo de estos son los terrenos.
- Bienes sujetos a depreciación: en este grupo se incluyen los que tienen deterioro a lo largo del tiempo, esto conlleva a que su valor disminuya periódicamente. Ejemplo de estos son los edificios, maquinarias, muebles, herramientas entre otros.
- Recursos naturales: estos se agotan a medida que se explotan, son recursos no renovables, algunos de estos son fondos mineros, bosques, pozos petroleros, etc.

El control de los activos fijos tangibles constituye tarea fundamental dentro de las organizaciones, debiendo implementarse de manera contable y física resultando en garantía para sus potencialidades de uso y el correcto funcionamiento de la entidad(Cornelio et al., 2016).

1.1.2 Auditoría y Control

Auditoría

Según la Real Academia de la lengua Española (RAE) una auditoría se define como la revisión sistemática de una actividad o de una situación para evaluar el cumplimiento de las reglas o criterios objetivos a que aquellas deben someterse. En el ámbito institucional una auditoría es un proceso sistemático para obtener y evaluar de manera objetiva, las evidencias relacionadas con informes sobre actividades económicas y otras situaciones que tienen una relación directa con las actividades que se desarrollan en una entidad. El fin del proceso consiste en determinar el grado de precisión del contenido informativo con las evidencias

que le dieron origen, así como determinar si dichos informes se han elaborado observando principios establecidos para el caso(Cornelio, Fonseca, Caballero and Hernández, 2016).

La auditoría constituye una herramienta de control y supervisión que contribuye a la creación de una cultura de la disciplina de la organización y permite descubrir fallas en las estructuras o vulnerabilidades existentes en la organización; pueden clasificarse en(Jara, 2014):

- Auditoría externa: es realizada por auditores totalmente ajenos a la empresa, esto permite que el auditor externo utilice su libre albedrío en la aplicación de los métodos, técnicas y herramientas con las cuales hará la evaluación de las actividades y operaciones de la empresa que audita.
- Auditoría interna: es realizada por un auditor que labora en la empresa donde se realiza la misma.

Control

Se entiende como control la comprobación o inspección de tareas con el fin de verificar su cumplimiento de forma eficiente y eficaz y tomar acciones correctivas cuando sea necesario. Por su parte el autocontrol es una obligación que tienen los directivos superiores, ejecutivos y funcionarios de los órganos, organismos, organizaciones y demás entidades, estas últimas con independencia del tipo de propiedad y forma de organización, de autoevaluar su gestión de manera permanente; elaborando un plan para corregir las fallas e insuficiencias, adoptar las medidas administrativas que correspondan y dar seguimiento al mismo en el órgano colegiado de dirección, comunicar sus resultados al nivel superior y rendir cuenta a los trabajadores(Cuba, 2012).

El control interno es llevado a cabo por las instituciones con un enfoque de mejoramiento continuo extendido a todas las actividades inherentes a la gestión, efectuado por la dirección y el resto del personal; se implementa mediante un sistema integrado de normas y procedimientos, que contribuyen a prever y limitar los riesgos internos y externos, proporciona una seguridad razonable al logro de los objetivos institucionales y una adecuada rendición de cuenta(Caballero, 2013).

Entre sus objetivos fundamentales destacan:

- Eliminar o reducir al mínimo posible las causas y condiciones que propician los riesgos internos y externos, así como los hechos de indisciplinas e ilegalidades, que continuados y en un clima de

impunidad, provocan manifestaciones de corrupción administrativa o la ocurrencia de presuntos hechos delictivos.

- Proteger los recursos de la organización, buscando su adecuada administración ante posibles riesgos que lo afecten.
- Velar porque todas las actividades y recursos de la organización estén dirigidos al cumplimiento de los objetivos de la entidad.

Auditoría y Control

Apoyándose en los conceptos anteriormente descritos se puede concluir que la auditoría y control es un proceso llevado a cabo en organizaciones, instituciones o entidades sistemáticamente, que consiste en obtener y evaluar objetivamente evidencias sobre las afirmaciones relativas a los actos o eventos de carácter económico – administrativo, con el fin de determinar el grado de correspondencia entre esas afirmaciones y los criterios establecidos, para luego comunicar los resultados a las personas interesadas.

Además, examina y evalúa la planificación, organización, dirección y control interno administrativo, la economía y eficiencia con que se han empleado los recursos humanos, materiales y financieros, así como el resultado de las operaciones previstas a fin de determinar si se han alcanzado las metas propuestas(Cuba, 2012).

1.1.3 Aplicación Informática

Una aplicación informática puede definirse como un tipo de software que permite al usuario realizar distintos tipos de trabajo. Son, aquellos programas que permiten la interacción entre el usuario y la computadora, brindando la oportunidad al usuario de elegir opciones y ejecutar acciones que el programa le ofrece.

Según(Pressman, 2010) el **software de computadora** es el producto que los ingenieros de software construyen y después mantienen en el largo plazo. Incluye los programas que se ejecutan dentro de una computadora de cualquier tamaño y arquitectura, el contenido que se presenta conforme los programas se ejecutan y los documentos, tanto físicos como virtuales, que engloban todas las formas de medios electrónicos.

Se puede decir que una aplicación es un tipo de programa informático diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de trabajos.

De acuerdo a lo planteado por Roger S. Pressman existen siete grandes categorías del software de computadora(Pressman, 2010):

- Software de sistemas.
- Software de aplicación.
- Software de ingeniería y ciencia.
- Software empotrado.
- Software de línea de producto.
- Aplicaciones web.
- Software de inteligencia artificial.

Cada sistema informático cumple objetivos y funciones acorde a esta clasificación, aunque la barrera entre unos y otros suele ser delgada atendiendo a la versatilidad y naturaleza cooperativa que en la actualidad poseen los productos de software, los cuales tienden a ser más complejos y abarcadores. Atendiendo esta categorización la propuesta de solución se propone enmarcarla en la categoría “software de aplicación”, decisión respaldada por las características del negocio, ampliamente descritas en el capítulo dos del presente documento. A continuación se reflejan algunos detalles de esta categoría de software.

Software de aplicación

El software de aplicación son aquellos programas aislados que resuelven una necesidad específica de negocio. Las aplicaciones en esta área procesan datos comerciales o técnicos en una forma que facilita las operaciones de negocios o la toma de decisiones administrativa o técnica(Pressman, 2010).

Las aplicaciones de escritorio como parte del software de aplicación (o *desktop*) son programas creados para ejecutarse en un ordenador de escritorio, sobre un sistema operativo de interfaz visual como Windows o Linux. Las mismas cuentan con una serie de características que las distinguen a la vez que aportan elementos a su favor(Sommerville, 2005).

- Poseen eficiencia, robustez, velocidad y capacidades gráficas de gran resolución.
- Ante la inexistencia de conexión de red pueden seguir brindando servicios.
- Poseen excelente experiencia de usuario.

- Cuentan con mecanismos para acceder a datos remotamente.
- Hacen uso de un buen aprovechamiento del entorno de ejecución al estar desarrolladas para un sistema y un hardware específico.
- Tienen un gran espectro de actuación.
- Baja latencia y retroalimentación instantánea.

1.2 Caracterización del Objeto de Estudio

1.2.1 El proceso de Auditoría y Control de los AFT en Cuba

Hoy en día resulta sumamente importante que todas las empresas establezcan un mecanismo de control para sus activos fijos. Los entes reguladores ofrecen información de cómo las empresas pueden administrar sus bienes. Las entidades financieras se han esforzado para establecer una única línea de administración, gestión y control en el activo fijo que conlleve a un registro claro, transparente y que, a su vez, permita realizar un seguimiento de los mismos(Contreras, 2013).

Algunas de las normas más significativas a nivel mundial que ponen foco en la gestión del activo fijo son las plasmadas en la Noma Internacional de Contabilidad (NIC), ejemplo de estas destacan(IASB, 2014):

- NIC-2 (Inventario).
- NIC-16 (Propiedad, Planta y Equipo).
- NIC-8 (Políticas contables, cambios en estimaciones contables y errores).
- NIF C-36 (Deterioro en el valor de los activos).
- NIIF-5 Activos no corrientes mantenidos para la venta y operaciones discontinuadas.
- NIC-38 Activos intangibles

En Cuba existe el Ministerio de finanzas y precios, el cual es el encargado de dirigir, ejecutar y controlar la aplicación de la Política Financiera, Tributaria, de Precios, de Auditoria y de Seguros, del Estado y del Gobierno, asesorándolos en esta política, dirigir y controlar la organización de la Administración Financiera del Estado(Precios, 2007). El 6 de junio de 2006 se crea La Resolución No. 148 el cual está compuesto por una serie de modelos, entre los que se encuentra el Modelo SC-1-07 - CONTROL DE ACTIVOS FIJOS TANGIBLES, cuyo objetivo fundamental es Mantener un control permanente de los activos fijos tangibles

en Contabilidad y servir a su vez como relación de los que se encuentran en poder de las áreas que controlan operativamente los mismos, siendo éstas responsables de su custodia y cuidado. Sirve además como base para el chequeo físico de estos bienes(Precios, 2007). Actualmente Cuba se rige por las normas planteadas en dicho documento para realizar el proceso de auditoría y control de los bienes de las empresas e instituciones.

1.2.2 El proceso de Auditoría y Control de los AFT en la UCI

La UCI es un centro docente-productor cubano con la misión de formar profesionales comprometidos con su Patria y altamente calificados en la rama de la Informática; producir aplicaciones y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación y servir de soporte a la industria cubana de la informática. La Universidad se organiza mediante una estructura que le permite cumplir con su misión fundamental docente-productiva que incluye: 6 facultades formadoras, 14 centros de desarrollo de software (algunos adscritos a las mencionadas facultades) y 41 direcciones generales, responsabilizadas estas últimas fundamentalmente de los servicios que presta y consume la entidad.

La Universidad de Ciencias Informáticas es considerada además una ciudad universitaria donde, para lograr su funcionamiento intervienen un grupo importante de recursos que van desde transporte y alimentación, hasta la base material para el estudio, el material de oficina y los propios equipos de cómputo. Constituye por ende el control de los medios en cuanto a su contabilización, debido uso y estado tarea meridiana dentro de los procesos sustantivos de la entidad. Los principales responsables del control de los medios puestos a disposición de estudiantes y trabajadores son los jefes de cada área administrativa, rectorados por los vicedecanatos de Administración y Economía en cada facultad y la vicerrectoría de Economía en la universidad.

Acorde a las normas establecidas por el país, cada mes debe auditarse el 10% de los AFT de un área determinada. El proceso se organiza a partir de una estructura compuesta por **centros de costo** y **áreas de responsabilidad**, siendo los primeros de mayor alcance, englobando varias áreas de responsabilidad y haciéndose corresponder comúnmente con las facultades, direcciones generales y centros de desarrollo independientes. Cada trabajador y estudiante perteneciente a un área de responsabilidad firma un compromiso ante el cuidado, conservación y preservación de los medios puestos a su disposición de cumplimiento obligatorio, siendo no obstante el jefe de área el principal responsable ante los AFT bajo su

encomienda. Se verificará la existencia del medio contabilizado y su estado físico, emitiéndose un informe a cada nivel con los resultados de la inspección, tributando las áreas de responsabilidad hacia los centros de costo, y estos a su vez hacia las facultades y direcciones, culminando en el nivel de la Universidad, no estando exenta esta ultima de inspecciones foráneas a nivel de ministerio. Las tareas relacionadas con la auditoria de los medios son asistidas en la universidad por el Sistema de Gestión Universitaria, el cual provee del listado de los medios a los responsables autorizados a cada nivel.

El proceso de auditoría y control en la Facultad de Ciencias y Tecnologías Computacionales

La Facultad CITEC de la Universidad de Ciencias Informáticas cuenta con alrededor de 620 estaciones de trabajo dedicadas a los procesos docentes productivos, desplegadas en toda la estructura de la propia facultad conformada por el decanato, cuatro vicedecanatos, cuatro departamentos docentes y tres centros productivos, las cuales cuentan además con el inmobiliario y otros recursos que aproximan un total de 2408 AFT.

En el Vicedecanato de Economía y Administración (en lo adelante VEA), fundamentalmente en la figura del vicedecano recae la responsabilidad de rectorar el proceso de auditoría y control con el objetivo de prevenir riesgos de sustracción, despilfarro, uso indebido u otras irregularidades respecto a los activos fijos tangibles. Cada jefe de área de responsabilidad está encargado de que al menos con una periodicidad mensual el 10% de sus activos sean auditados, echo que se extiende de igual modo a los centros de costo y la facultad en general, el responsable del área comúnmente cuenta con un asesor en el cual puede delegar el conteo y control de los AFT.

Comúnmente el proceso se desencadena a partir de la indicación del VEA de iniciar el proceso de auditoría y control mensual. Cada responsable de área obtiene del Sistema de Gestión Universitaria un listado impreso con los rótulos de los medios básicos y el estado físico en el que se encontraban en la última inspección, luego el propio responsable o el asesor deben comprobar mediante un testeo manual la coincidencia numérica en los medios y el listado registrando además el estado del AFT. Finalmente se debe obtener un informe que refleje como resultado de la auditoría si existen faltantes o sobrantes y el estado general de los medios, el mismo se debe archivar y debe constar como evidencia del control sobre los recursos asignados.

1.3 Análisis de soluciones similares

Respecto al proceso de auditoría y control de los Activos Fijos Tangibles por parte de las entidades, se han desarrollado soluciones informáticas con el propósito de aportar cualidades deseables a la mencionada actividad. Se considera oportuno el análisis de alguno de estos productos con el fin de evaluar sus debilidades, fortalezas y la manera en la que se adaptan al problema de la investigación, en busca de obtener elementos que contribuyan al desarrollo de la propuesta de solución. A continuación se hace referencia a los productos más relevantes:

ContarERP

Fig. 1 Logotipo de la aplicación ContarERP.

Solución que enfoca y garantiza la administración de Activos Fijos. Permite la administración y control de la propiedad, para los activos tangibles adquiridos, construidos o en proceso de construcción, asignando el responsable, su ubicación, estado y demás; realizar la depreciación de cada activo de manera automática. Entre sus principales características destaca(erp, 2014a):

- Libre definición de Grupos, Tipos, Ubicación, Estados, Características (Partes), Departamentos, Responsables y Centros de costos.
- Captura de Saldos Iniciales por activo, ingresando valor de compra, depreciación acumulada, ajuste a la depreciación acumulada, ajuste por inflación, fecha de compra y meses a depreciar.
- Permite cargar la foto del activo.
- Captura de información para el registro del avalúo técnico (valor, tercero y fecha).
- Asociación de características al activo para controlar sus partes.

- Manejo de hoja de vida del activo para documentar sus mejoras.
- Ventas de Activos Fijos.

El sistema se considera configurable y contempla la gestión y control en buena medida de los aspectos esenciales referentes a los AFT, resultando factible en entidades de gran tamaño como otras de menor envergadura. ContarERP es un software comercial licenciado y de servicio, propiedad de la empresa de su mismo nombre. El mismo se desarrolló bajo los estándares y las normas de control establecidas en Colombia mediante el empleo de tecnologías privativas fundamentalmente(erp, 2014b).

La solución que se describe no se considera adecuada para su empleo en el entorno del problema de la presente investigación, ya que su implantación supone una limitante económica, no se cuenta con acceso a su código fuente, la misma no permite la integración con otros sistemas y aplicaciones. Por otra parte enfoca su funcionamiento a las normativas colombianas del control de activos y su implantación sería contraria a las políticas de soberanía tecnológica que impulsa el país. No obstante algunas de sus funcionalidades serán consideradas como parte de la solución propuesta.

VERSAT Sarasola

Fig. 2 Interfaz de Bienvenida de VERSAT Sarasola.

La aplicación VERSAT Sarasola fue creada por un grupo villaclareño de especialistas económicos e informáticos pertenecientes a la Empresa de Servicios Técnicos Industriales del Grupo Azucarero nacional cubano. Cuenta con una serie de funcionalidades que permite la planificación y control de los recursos

humanos, materiales y financieros de cualquier entidad, tanto del sistema empresarial como presupuestado. Asimismo posee una versión educativa para su utilización en escuelas de comercio y en universidades. VERSAT Sarasola consta de 12 módulos o subsistemas de configuración, contabilidad y costos; finanzas, inventarios, activos fijos; nóminas, planificación, facturación, mensajería, auditoría y generador de reportes. Dentro de sus características fundamentales destacan(Clara, 2001):

- Herramienta para la planificación económica, el control y el análisis de gestión.
- Permite llevar el control y registro contable individual de todos los hechos económicos que se originan en las estructuras internas de las entidades, así como exponer el estado financiero y toda la información económica y contable en este universo.
- Se estructura en un grupo de subsistemas en los cuales se procesan y contabilizan los documentos primarios, donde se anotan los movimientos, los recursos materiales, laborales y financieros que se utilizan en una entidad.
- Rapidez y fiabilidad, a partir de la configuración del proceso de contabilización de los documentos primarios y de las propias posibilidades de trabajo contenidas en cada subsistema.

A pesar de las bondades de la aplicación se desestima su empleo como solución al problema planteado ya que la misma no permite acceso a su código fuente, está construida sobre Windows y solo para Windows desde Visual Estudio sobre la plataforma .Net, todas herramientas y tecnologías propietarias, la misma no proporciona recursos que permitan agilizar el proceso de auditoría de medios contra listado de activos.

CEDRUX

En el ámbito nacional destaca como software de propósito afín la plataforma CEDRUX, Sistema Integral de Gestión Empresarial, concebido en su proyección inicial como un paquete de soluciones integrales de gestión, basado en los principios de independencia tecnológica y en la inclusión de las funcionalidades generales presentes en los procesos de la economía cubana. El mismo incluye un módulo para la gestión y control de AFT el cual cuenta con soporte para los procesos fundamentales que ocurren respecto a los activos, acorde a las normas y legislaciones nacionales, entre los cuales cabe destacar(Virgen Damaris Quevedo Campins, 2010):

- Apertura
- Modificaciones

- Cierre de Apertura
- Alta Movimientos
- Baja
- Inventario
- Depreciación

El mismo emplea un conjunto de herramientas y tecnologías libres en su construcción. Sin embargo el mencionado módulo muestra limitadas posibilidades de comunicación con sistemas externos, no contempla el inventario físico dentro de los procesos a agilizar, limitando su funcionamiento a áreas de conectividad dada la naturaleza web del sistema, lo cual constituye una desventaja en el entorno que demanda la solución del problema de la presente investigación.

Akademos (Sistema de Gestión Universitaria)

La Universidad de las Ciencias Informáticas maneja actualmente parte del proceso de control de sus activos mediante la plataforma web "Akademos", desarrollado bajo el empleo de herramientas y tecnologías adecuadas, en consonancia a las políticas de soberanía tecnológica impulsadas por el país. El Módulo de Activos Fijos permite observar las existencias de los diferentes elementos por áreas, así como hacer ajustes respecto a su estado y existencia. Posee una adecuada estructura de acceso basada en roles y acorde a las características del centro, su funcionalidades se encuentran completamente atemperadas a las normas y legislaciones nacionales vigentes(Reyes, 2016).

A pesar de las bondades de "Akademos" el mismo no incluye el manejo de las auditorías, carece de elementos como la generación de reportes e informes enriquecidos y su acceso desde algunas áreas auditables de la universidad carentes de conexión se hace imposible. Por las razones anteriores no se estima el empleo de la solución como respuesta al problema de la presente investigación aunque se tienen en cuenta sus potencialidades y se valora el uso de alguna de sus funcionalidades dentro del *Sistema para la informatización del proceso de Auditoría y Control de los Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales*.

Conclusiones sobre el estudio de soluciones similares

Luego de realizado un estudio de las soluciones informáticas que resuelven problemas similares, se considera que ninguna de las analizadas satisface el problema que da origen a la presente investigación. Las mismas no brindan herramientas que permitan contribuir a la agilidad del proceso de auditoría física de los medios, aspecto fundamental dentro del propio proceso de control, tanto en unos como en otros hay que

verificar la existencia de los AFT de forma manual, revisando el rótulo de cada medio previamente escrito, contra la lista del local donde deben estar: además en su gran mayoría no se adaptan a las características específicas de la Facultad CITEC de la Universidad de las Ciencias Informáticas. Por otra parte no son capaces de generar informes y reportes, enriquecidos con gráficas y otros recursos que pueden apoyar la toma de decisiones. Apoyado en los planteamientos anteriores se decide la construcción del "*Sistema para la informatización del proceso de Auditoría y Control de los Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales*".

1.4 Metodología de Desarrollo de Software

En la actualidad el software de computadora es la tecnología individual más importante en el ámbito mundial. A medida que la importancia del software ha crecido, la comunidad del software ha intentado de manera continua desarrollar tecnologías que hagan más fácil, más rápida y menos cara la construcción y el mantenimiento de programas de computadora de alta calidad(Pressman, 2010).

Las metodologías de desarrollo de software son específicamente quienes permiten a los desarrolladores, a partir de un conjunto de técnicas, procedimientos y herramientas, elaborar el nuevo producto; detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla, o sea, describen paso a paso el proceso de conformación del producto informático deseado(Romero, 2009). Las metodologías de desarrollo se encuentran divididas en dos grandes grupos conocidos comúnmente como ágiles y tradicionales debiéndose adaptar a las necesidades de cada proyecto en cualquier caso.

Metodologías Tradicionales

Se focalizan en documentación, planificación y procesos, cumpliendo con un detallado plan y documentando exhaustivamente todo el proyecto, entre ellas podemos destacar el Proceso Unificado de Racional (RUP), adaptable para proyectos de largo plazo y gran dimensión con pobre soporte para cambios; Microsoft Solution Framework (MSF), que resulta un compendio de las mejores prácticas en cuanto a administración de proyectos se refiere, muy adaptable a formatos y tecnologías(Jacobson et al., 2007).

Metodologías Ágiles

Combinan una filosofía con un conjunto de lineamientos de desarrollo. La filosofía pone el énfasis en: la satisfacción del cliente y en la entrega rápida de software incremental, los equipos pequeños y muy

motivados para efectuar el proyecto, los métodos informales, los productos del trabajo con mínima ingeniería de software y la sencillez general en el desarrollo(Mike Beedle, 2001). Dentro de las propuestas ágiles destaca Open Up, metodología seleccionada para guiar el proceso de desarrollo de la presente investigación, detallándose a continuación sus características esenciales, las cuales refuerzan dicha decisión.

El Proceso Unificado Abierto: metodología utilizada

En función del estudio realizado y las características del equipo de desarrollo y de la solución propuesta en cuestión se determina el empleo de una metodología ágil, ya que el tiempo de desarrollo es limitado, el proyecto que se pretende es relativamente pequeño, así como el equipo responsable del desarrollo del mismo. Se necesita solo la documentación imprescindible para dar soporte al sistema y su proceso de construcción.

Dentro de las metodologías ágiles se optó por la selección del Proceso Unificado Abierto (del inglés Open Unified Process), basado en sus características distintivas: apropiada para proyectos pequeños y de bajos recursos, basada en casos de uso, centrada en la arquitectura, además permite disminuir las probabilidades de fracaso e incrementar las probabilidades de éxito. Detecta errores tempranos a través de un ciclo iterativo y evita la elaboración de documentación, diagramas e iteraciones innecesarias requeridas por otras metodologías. Posee un enfoque centrado al cliente y con iteraciones cortas ajustándose plenamente a las necesidades para el desarrollo de la propuesta de solución. A continuación se abunda con más profundidad en la metodología de desarrollo seleccionada para guiar el proceso de desarrollo del sistema.

El Proceso Unificado Abierto (del inglés Open Unified Process, Open UP) es un proceso que aplica propuestas de gestión ágil, tratando de ser manejable en relación con algunas metodologías tradicionales (específicamente RUP), manteniendo sus características esenciales: centrada en la arquitectura, dirigida por Casos de Uso y a través de un desarrollo iterativo e incremental.

No provee lineamientos para todos los elementos que se manejan en un proyecto, pero contiene el conjunto mínimo de práctica que ayuda a los equipos de desarrollo a ser más eficientes. Cuenta con los componentes básicos que pueden servir de base a procesos específicos y la mayoría de los elementos están declarados para fomentar el intercambio de información entre los equipos de desarrollo y mantener un entendimiento

compartido del proyecto, sus objetivos, alcance y avances. Es un proceso iterativo que es mínimo, completo y extensible y puede utilizarse tal cual o ampliarse para tratar una amplia variedad de tipos de proyectos.

La colaboración para alinear los intereses y desarrollar buenas prácticas colaborativas que generen un buen ambiente de trabajo se encuentra entre sus principios básicos entre los cuales se encuentran además el enfoque para reducir riesgos y articular la arquitectura, el balance para confortar las prioridades (necesidades y costo técnico) y la evolución para dividir el proyecto en iteraciones cortas obteniendo retroalimentación temprana. Se valora la colaboración y el aporte de las personas involucradas en el proyecto fundamentalmente sobre los entregables y la definición de las formalidades necesarias. A continuación mediante la Fig. 3(eclipse, 2010) se describen brevemente las cuatro fases fundamentales propuestas por la metodología(Up, 2010).

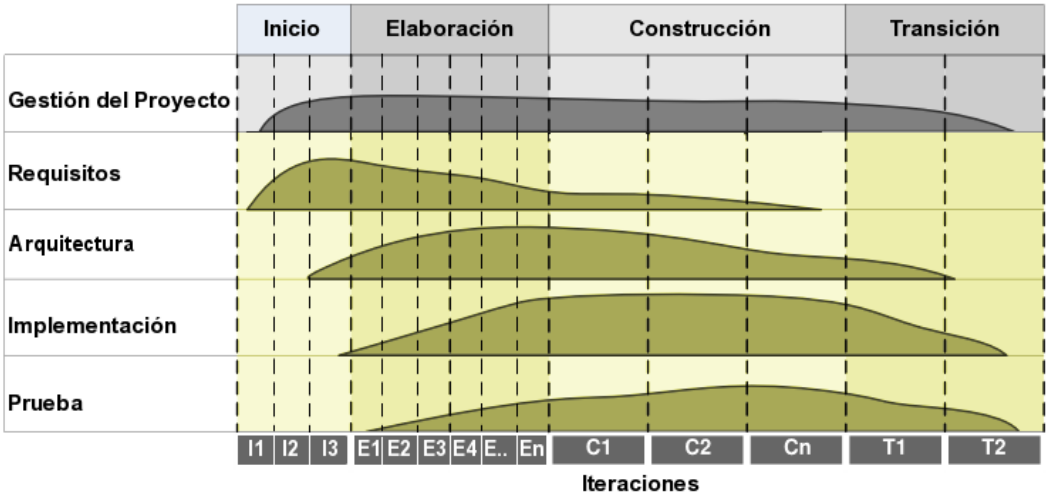

Fig. 3 Ciclo de vida de la metodología Open Up.

1. **Fase de inicio**: En esta fase, las necesidades de cada participante del proyecto son tomadas en cuenta y plasmadas en objetivos del proyecto. Se definen para el proyecto: el ámbito, los límites, el criterio de aceptación, los Casos de Uso críticos, una estimación inicial del coste y un boceto de la planificación.
2. **Fase de elaboración**: En esta fase se realizan tareas de análisis del dominio y definición de la arquitectura del sistema. Se debe elaborar un plan de proyecto, estableciendo, unos requisitos y una arquitectura estables, se especifican, además, las herramientas, la infraestructura a utilizar y el entorno de desarrollo.

3. **Fase de construcción**: Todos los componentes y funcionalidades del sistema que falten por implementar son realizados, probados e integrados en esta fase.
4. **Fase de transición**: Esta fase corresponde a la introducción del producto en la comunidad de usuarios. Consta de las sub-fases de pruebas de versiones beta, pilotaje y capacitación de los usuarios finales y de los encargados del mantenimiento del sistema.

Open UP se señala apropiado no solo para proyectos pequeños sino también de bajos recursos, permite disminuir las probabilidades de fracaso e incrementar las posibilidades de éxito siendo especialmente diseñada para pequeños equipos que mantienen un alto compromiso con la calidad(Foundation, 2016).

1.4.1 UML 2.0: lenguaje de soporte a la metodología utilizada

Lenguaje Unificado de Modelado (*Unified Modeling Language*) UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Desde el año 1995, UML es un estándar aprobado por la ISO como ISO/IEC 19501:2005 *Information technology*; está respaldado por el Grupo de Administración de Objetos (*Object Management Group*) OMG[2]. Este se puede aplicar en el desarrollo de software, entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software, como es el caso de Open UP, aunque no especifica en sí mismo qué metodología o proceso usar(Group, 2005).

Entre sus principales características se encuentran:

- Es independiente del proceso, para utilizarlo de forma óptima se debe usar en un proceso que sea dirigido por Casos de Uso, centrado en la arquitectura e iterativo e incremental.
- Permite modelar sistemas utilizando técnicas de programación orientadas a objetos (POO).
- Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, etc.).
- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego

[2] El ***Object Management Group*** (**OMG**) es un consorcio, formado en 1989, dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos, tales como UML, XMI, CORBA y BPMN.

desplegar los sistemas.

1.4.2 Visual Paradigm for UML 8.0: herramienta CASE empleada

CASE (*Computer Aided Software Engineering*) en su traducción al Español significa Ingeniería de Software Asistida por Computación, y son aquellas herramientas CASE las que propician el aumento de la productividad en el desarrollo de un proyecto y como herramientas que son, deben ser aplicadas a una metodología determinada.

Visual Paradigm for UML es la herramienta CASE seleccionada, la cual soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas, despliegue y contribuye a una rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Se caracteriza por el uso de un lenguaje estándar común al equipo de trabajo, que facilita la comunicación entre sus integrantes, es una herramienta fácil de instalar y actualizar, genera código para varios lenguajes de programación y exporta en varios formatos, es una tecnología libre y está disponible en varios idiomas.

Esta herramienta permite aumentar la calidad del software, a través de la mejora en el desarrollo y mantenimiento del mismo, de igual forma potencia la reutilización de componentes y estandarización de la documentación generada. Puede ser empleada desde múltiples plataformas(Paradigm, 2016).

1.5 Sistema Gestor de Bases de Datos

Un Sistema Gestor de Bases de Datos (en lo adelante SGBD) es un conjunto de programas, procedimientos y lenguajes que proporcionan las herramientas necesarias para trabajar con una Base de Datos. Incorpora una serie de funciones que permiten definir los registros, sus campos, sus relaciones, insertar, suprimir, modificar y consultar los datos. Con el objetivo de trabajar sobre la persistencia y posterior manejo de los datos asociados a la problemática actual, se define y argumenta el uso de PostgreSQL como Sistema Gestor de Bases de Datos en el siguiente acápite.

1.5.1 PostgreSQL 9.5: Sistema Gestor de Bases de Datos

PostgreSQL es un SGBD objeto-relacional, distribuido bajo licencia BSD[3] y con su código fuente disponible libremente. Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Ofrece control de concurrencia multi-versión, soportando casi todas las sintaxis SQL como subconsultas, transacciones y funciones definidas por el usuario; cuenta además con un amplio conjunto de enlaces con lenguajes de programación. Funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema(Guerrero, 2010).

Entre sus características se encuentran:

- Gran escalabilidad. Capaz de ajustarse al número de Unidades Centrales de Procesamiento (del inglés Central Processing Units, CPU) y a la cantidad de memoria que posee el sistema de forma óptima.
- Capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia Base de Datos.
- Manejo de grandes volúmenes de datos.

Los componentes más importantes en el funcionamiento de PostgreSQL se muestran a continuación:

- **Aplicación cliente:** Esta es la aplicación cliente que utiliza PostgreSQL como administrador de Bases de Datos. La conexión puede ocurrir vía TCP/IP ó sockets locales.
- **Demonio postmaster:** Este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket las conexiones entrantes de clientes. También es el encargado de crear los procesos hijos que se encargarán de autenticar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes.
- **Write-Ahead Log (WAL):** Componente del sistema encargado de asegurar la integridad de los datos.
- **Disco:** Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

[3] **Licencia BSD**: Licencia de software libre permisiva como la licencia de OpenSSL o la MIT License. Tiene menos restricciones en comparación con otras como la GPL estando muy cercana al dominio público. Permite el uso del código fuente en software no libre.

PGAdmin III 1.20.0 para el manejo de PostgreSQL

PgAdmin III es la más popular y completa plataforma de administración y desarrollo de código abierto para PostgreSQL, el Sistema Gestor de Bases de Datos de código abierto más utilizado hoy. La aplicación puede utilizarse en Linux, FreeBSD, Solaris, Mac OSX y Windows para administrar PostgreSQL 7.3 y superiores y funciona en cualquier plataforma. Está diseñada para responder a las necesidades de todos los usuarios, desde escribir simples consultas SQL hasta crear Bases de Datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita su administración.

La conexión con el servidor es posible realizarla a través de TCP/IP o Unix Domain Sockets (en plataformas Unix), y puede utilizar encriptado SSL para la seguridad. No se requieren controladores adicionales para comunicarse con el servidor de Base de Datos. Desarrollado por una comunidad de expertos de PostgreSQL en todo el mundo, está disponible en más de una docena de idiomas(pgAdmin, 2016).

1.6 Lenguaje de Programación utilizado

Un lenguaje de programación consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, es diseñado para describir el grupo de acciones consecutivas que un equipo debe ejecutar, lo que permite crear programas informáticos posibilitando al desarrollador comunicarse con los dispositivos de hardware y software existentes.

Java 8.0 como lenguaje de programación

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere"), lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra.

El lenguaje ha sido probado, ajustado, ampliado y probado por toda una comunidad de desarrolladores, arquitectos de aplicaciones y entusiastas. Su diseño se caracteriza por ser robusto, seguro, portable, independiente a la arquitectura, dinámico e interpretado permitiendo el desarrollo de aplicaciones portátiles de elevado rendimiento para el más amplio rango de plataformas informáticas posibles(TM, 2016).

Alguna de las ventajas que destacan ante el empleo del lenguaje son: su sencillez, con una curva de aprendizaje corta, lenguaje orientado a objetos, soporte para aplicaciones distribuidas, interpretado y compilado, con características robustas de seguridad y una documentación profunda y exhaustiva(TM, 2016).

Java FX 8 para el manejo de interfaces

JavaFX es un conjunto de paquetes de gráficos y medios que permite a los desarrolladores diseñar, crear, probar, depurar e implementar aplicaciones de cliente enriquecido que operan de forma consistente en diversas plataformas. Se encuentra escrito a manera de una API (del inglés *Aplication Programing Interface*) Java, el código de la aplicación JavaFX puede hacer referencia a las API desde cualquier biblioteca Java. Por ejemplo, las aplicaciones JavaFX pueden utilizar bibliotecas de API Java para acceder a las capacidades nativas del sistema y conectarse a aplicaciones de middleware[4] basadas en servidor.

El aspecto y la sensación de las aplicaciones JavaFX se pueden personalizar. Hojas de estilo en cascada (CSS) separan el aspecto y el estilo de la implementación para que los desarrolladores puedan concentrarse en la codificación. Los diseñadores gráficos pueden personalizar fácilmente la apariencia y el estilo de la aplicación a través del CSS. Es un producto de código abierto distribuido bajo licencia GPL. Con el propósito de lograr interfaces robustas y con alto grado de usabilidad se decide el empleo de la mencionada tecnología(Oracle, 2017).

Hibernate ORM 5.0.2

Hibernate ORM (del inglés *Object Relational Mapping*) es una herramienta de correlación objeto-relacional para el lenguaje de programación Java. Proporciona un marco para asignar un modelo de dominio orientado a objetos a una base de datos relacional. Hibernate resuelve los problemas de desajuste de impedancia objeto-relacional mediante la sustitución de accesos de base de datos directos y persistentes por funciones de alto nivel de manejo de objetos. Es un software libre que se distribuye bajo la GNU Lesser General Public License 2.1.

[4] Middleware es un software de computadora que proporciona servicios a aplicaciones de software más allá de las disponibles desde el sistema operativo. Se describe comúnmente como "software pegamento".

La característica principal de Hibernate es la asignación de clases Java a tablas de bases de datos y la asignación de tipos de datos Java a tipos de datos SQL. Hibernate también proporciona servicios de consulta y recuperación de datos. Genera llamadas SQL y libera al desarrollador de la manipulación manual y la conversión de objetos del conjunto de resultados. Fue diseñado para funcionar en un clúster de servidores de aplicaciones y ofrecer una arquitectura altamente escalable. Hibernate se adapta bien a cualquier entorno. Hibernate es bien conocida por su excelente estabilidad y calidad, probada por la aceptación y el uso por decenas de miles de desarrolladores Java, una tecnología altamente escalable y configurable. Por las características antes mencionadas se decide el empleo de Hibernate como tecnología ORM para el manejo de la persistencia de datos entre el gestor PostgreSQL y la aplicación(ORM, 2016).

1.7 Entorno Integrado de Desarrollo

Un Entorno de Desarrollo Integrado (del inglés *Integrated Development Environment*, IDE) es un programa compuesto por un conjunto de herramientas para un programador. Provee un marco de trabajo amigable para la mayoría de los lenguajes de programación, pudiendo utilizarse en el mismo uno o varios lenguajes de programación.

NetBeans IDE 8.0 como entorno integrado de desarrollo utilizado

En la implementación de la propuesta de solución se define utilizar el IDE NetBeans 8.0, herramienta de código abierto con una gran base de usuarios y una comunidad en constante crecimiento. Existe además un número importante de módulos para extenderlo. Es un producto libre y gratuito sin restricciones de uso. Está compuesta también por una base modular y extensible usada como una estructura de integración para crear grandes aplicaciones de escritorio.

Entre sus características están las administraciones de ventanas, almacenamiento, interfaces y configuraciones de usuario. Soporta el desarrollo de aplicaciones Java (J2SE, web, EJB y aplicaciones móviles), empresariales con Java EE 5, JavaFX 2.0, WebLogic 12c y CSS3. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones(NetBeans, 2016).

1.8 Conclusiones parciales

El proceso de auditoría y control de los activos fijos tangibles resulta fundamental para la Facultad de Ciencias y Tecnologías Computacionales en cuanto a los indicadores del control interno y preservación de los recursos refiere. Sin embargo, el proceso carece de agilidad, seguridad y eficacia, que garantice emplear poco tiempo y desarrollar las tareas asociadas de forma satisfactoria.

La utilización de las herramientas y tecnologías propuestas facilita y fortalece el trabajo, aportando rapidez y eficacia a lo largo del ciclo de vida del desarrollo y garantiza una mayor usabilidad de la aplicación. Las herramientas seleccionadas impulsan la soberanía tecnológica propuesta por el país y por el entorno de desarrollo donde se produce este resultado.

Capítulo 2: Descripción y Diseño de la Solución Propuesta

En el capítulo que se desarrolla a continuación se muestran las primeras tareas correspondientes al análisis de la propuesta de solución. Se detallan los respectivos modelos del negocio y sistema así como los Requisitos Funcionales y No Funcionales que deben conformar la solución que se persigue, se obtendrá el diagramado de los Casos de Uso del Sistema así como su descripción textual. Se obtendrán además los diagramas correspondientes a la representación del diseño, fundamentando la arquitectura seleccionada y los patrones de diseño utilizados. Al concluir este capítulo se contará con una comprensión de los problemas actuales del negocio y una idea pertinente del sistema a desarrollar.

2.1 Propuesta de solución

Se propone desarrollar un sistema capaz de agilizar el proceso de auditoría y control de los AFT en la Facultad CITEC. La solución se basa en un sistema de escritorio multiplataforma con comunicación mediante servicios al SGU. La aplicación debe de ser capaz de ante la selección de un área determinada a auditar obtener el listado de medios y servirse de un dispositivo o scanner de barras para apoyar el testeo de los medios, ofreciendo de manera automática los resultados de la auditoria a manera de informe, con gráficos y otros elementos que apoyen la toma de decisiones. La inexistencia de una amplia red inalámbrica en el entorno que se propone la solución descarta la posibilidad de una aplicación web, ya que la mayoría de los medios se encuentran alejados de puntos de acceso cableados a la red, y la movilidad inherente al proceso requiere el trabajo sin conexión. A continuación, en la Fig. 4, se muestra, de forma gráfica, una idea general de la solución propuesta:

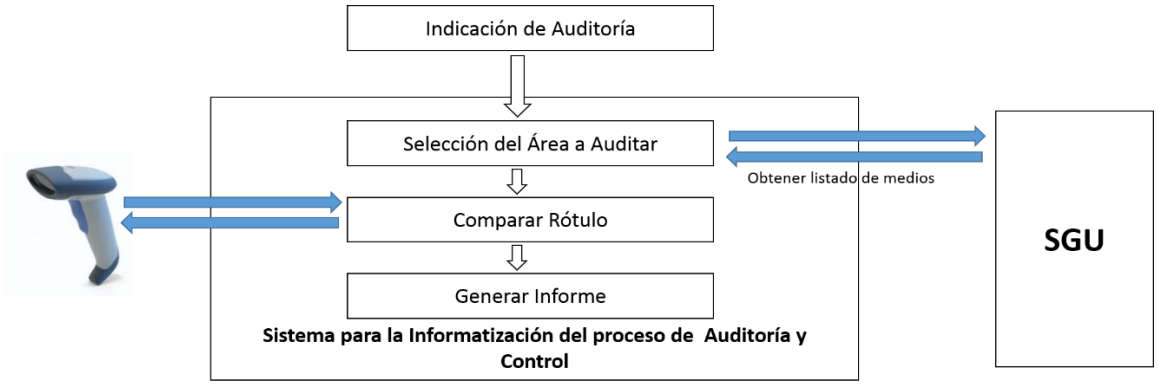

Fig. 4 Propuesta de solución.

2.2 Modelo de negocio

El primer paso ante la decisión de informatizar un proceso resulta la comprensión del mismo en su ambiente natural. Es por ello que resulta imprescindible acudir a recursos que permitan servir de punto de partida al diseño del sistema en cuestión. El modelo de negocio es una técnica para comprender los procesos de negocio de la organización objeto de estudio. Este modelo presenta un sistema desde la perspectiva de su uso, y esquematiza cómo proporciona valor a sus usuarios.

Según Ivar Jacobson el objetivo del modelo de negocio es describir los procesos, existentes u observados, con el propósito de comprenderlos. Se especifican aquí que procesos del negocio soportará el sistema y se identifican los objetos del dominio o del negocio implicados, este modelo establece las competencias que se requiere de cada proceso: sus trabajadores, sus responsabilidades y las operaciones que llevan a cabo(Jacobson, Booch and Rumbaug, 2007).

2.2.1 Reglas del negocio a considerar

En una organización, tanto los procesos como los datos que estos manejan, están restringidos por reglas del negocio que comúnmente conforman el conjunto de elementos del negocio que actúan como reguladores de la ejecución, supervisión, control y toma de decisiones de los diferentes procesos y actividades (Abran and Moore, 2004). A continuación se describen las principales reglas del negocio a considerar:

1. Pueden existir dos tipos de auditorías: internas y externas.
2. Las auditorías internas son realizadas por los responsables de áreas o centros de costos y/o sus asesores económicos.
3. Las auditorías externas son llevadas a cabo por un auditor ajeno a la organización que se desea controlar.
4. Dentro de las áreas o centros de costo se debe garantizar al menos una auditoría de carácter mensual.
5. El vicedecano de economía y administración puede auditar todos los centros de costo pertenecientes a la facultad, a su vez el responsable de un centro de costo solo puede auditar las áreas comprendidas en dicho centro.

6. Cada auditoría debe contemplar al menos el 10% de los activos fijos tangibles del centro de costo o área que está siendo controlada.
7. Al concluir la auditoría se genera un informe con los resultados de la misma incluyendo una evaluación final de aceptable o deficiente.

2.2.2 Actores y trabajadores del negocio

El modelo de Casos de Uso del Negocio es un modelo que describe los procesos de un negocio (Casos de Uso del negocio) y su interacción con elementos externos (actores), en esencia refleja las funciones que el negocio pretende realizar, así mismo un trabajador del negocio es una abstracción de una persona (o grupo de personas), una máquina o un sistema automatizado; que actúa en el negocio realizando una o varias actividades(Sommerville, 2005). A continuación se explicitan y justifican los actores y trabajadores que intervienen en el negocio a modelar.

Tabla. 1 Actores del negocio.

Actores del Negocio	Justificación
Responsable	Es el directivo que demanda o solicita la realización de una auditoría acorde al rango y las competencias del mismo dentro de la estructura organizacional de la entidad.

Tabla. 2 Trabajadores del negocio.

Trabajadores del Negocio	Justificación
Auditor	Es el responsable de llevar a hechos el proceso de conteo de los activos fijos tangibles dentro de la organización como parte de la auditoría demandada.

2.2.3 Diagrama de Casos de Uso del Negocio

Un Diagrama de Casos de Uso del Negocio (en lo adelante DCUN) representa gráficamente los procesos del negocio y su interacción con los actores del negocio (Abram and Moore, 2008) , mediante la Fig. 5 se muestra el DCUN correspondiente a la propuesta de solución.

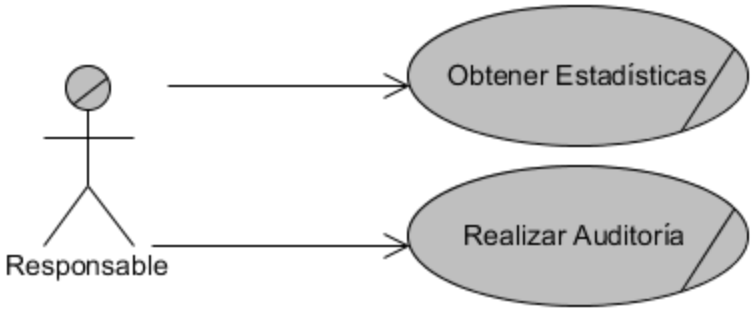

Fig. 5 Diagrama de Casos de Uso del Negocio.

2.2.4 Descripción textual del Caso de Uso del Negocio "Realizar Auditoría"

Tabla. 3 Descripción textual del CUN "Realizar Auditoría".

Nombre del Caso de Uso del Negocio:	Realizar Auditoría.
Actores del negocio:	Responsable (inicia).
Propósito:	Obtener información del estado de los activos fijos tangibles del área o centro de costo que está siendo auditado y emitir una evaluación final.
Resumen: El caso de uso inicia cuando el responsable demanda la realización de una auditoría a un área o centro de costo. Se realiza posteriormente un testeo manual de los activos fijos tangibles contra un listado previamente impreso, resultado de esto se obtiene un informe en el cual se evidencia el estado actual de los activos así como la evaluación final de la auditoría.	
Casos de Uso asociados:	Realizar Auditoría.
Flujo normal de los eventos	
Acción del actor:	**Respuesta del negocio:**
1 El responsable solicita una auditoría a un área o centro de costo determinado.	2 Se realiza un conteo manual de los activos fijos contra el listado previamente impreso. 2.1 Se verifica la coincidencia de los números de medio básico. 2.2 Se especifica el estado del medio. 3 Se emite un resultado final, dejando plasmado la evaluación final de la auditoría. 3.1 Se refleja la composición de equipo auditor.

4 El responsable obtiene la evaluación final de la auditoría.	
Prioridad:	Alta.
Mejoras:	Permitirá agilizar el proceso con que se lleva a cabo las auditorías. Se mitigan las posibles incertidumbres en los datos introducidas por un error humano. Permitirá cumplir con el objetivo planteado de auditar mensualmente el 10% de los activos fijos de las áreas o centros de costo.
Cursos alternos:	
2 Se informa que el área seleccionada no podrá ser auditada.	
Casos de Uso Incluidos:	No existe.

2.2.5 Diagrama de actividades del Caso de Uso del Negocio: "Realizar Auditoría"

Un diagrama de actividades describe un proceso que explora el orden de las tareas o actividades que logran los objetivos del negocio. Para un mejor entendimiento del Caso de Uso del Negocio anteriormente especificado se propone el siguiente diagrama de actividades ilustrado mediante la Fig. 6 donde se señalan en fondo gris las actividades objeto de informatización(Larman, 2005).

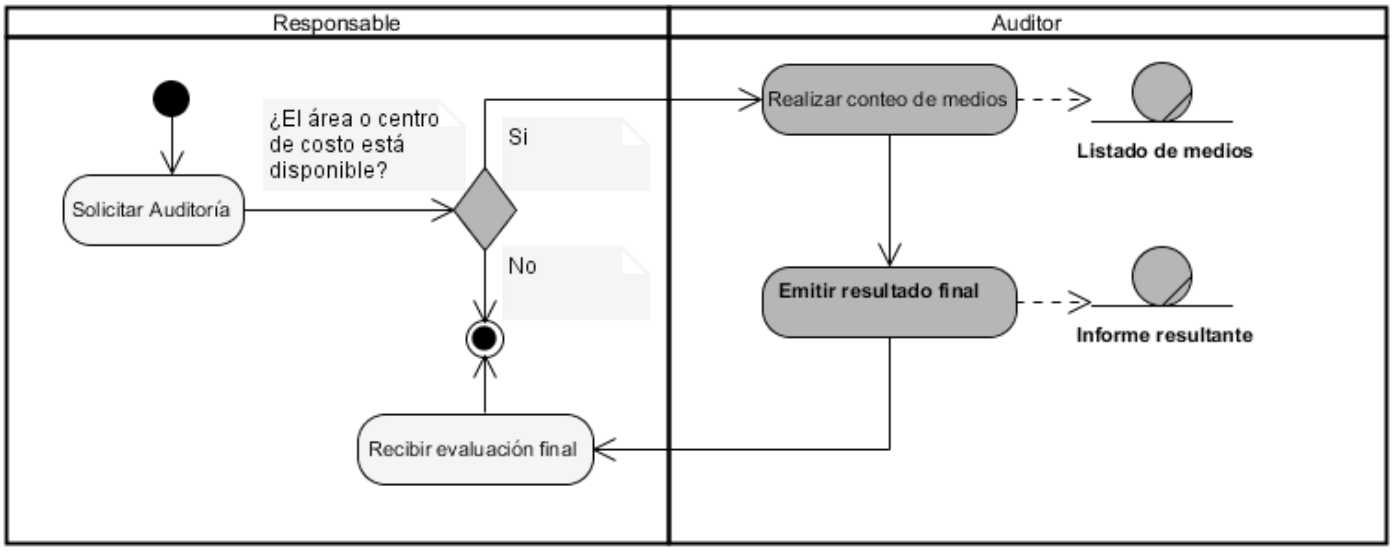

Fig. 6 Diagrama de Actividades del Caso de Uso del Negocio: "Realizar Auditoría".

2.3 Requisitos

Para (Sommerville, 2005) de forma concreta, los requisitos para un sistema son las descripciones de los servicios proporcionados por el sistema y sus limitaciones operacionales. Estos requisitos reflejan las necesidades de los clientes para un sistema que ayuda a resolver algunos problemas, tales como el control de un dispositivo, realizar un pedido o la búsqueda de información. Una visión más amplia del tema podría ser la brindada por (Pressman, 2008) al plantear que los requisitos de software son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema de desarrollo y las restricciones en las que debe operar. Los requisitos se dividen en Funcionales y No Funcionales y muestran las capacidades o condiciones que el sistema debe cumplir y las propiedades o cualidades que el producto debe tener.

2.3.1 Requisitos funcionales

Los requisitos funcionales (en lo adelante RF) describen lo que el sistema debe hacer. Según (Sommerville, 2005), los RF son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones específicas. Estos requisitos dependen del tipo de software que se desarrolle, de los posibles usuarios y del enfoque general tomado por la organización al redactarlos. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el producto no debe hacer.

El sistema debe ser capaz de:

RF1. Autenticar usuario.

RF2. Insertar nuevo usuario.

RF3. Actualizar usuario.

RF4. Mostrar usuarios.

RF5. Buscar usuario.

RF6. Eliminar usuario.

RF7. Insertar nuevo rol.

RF8. Actualizar rol.

RF20. Insertar nueva área de responsabilidad.

RF21. Actualizar área de responsabilidad.

RF22. Mostrar área de responsabilidad.

RF23. Buscar área de responsabilidad.

RF24. Eliminar área de responsabilidad.

RF25. Insertar nuevo centro de costo.

RF26. Actualizar centro de costo.

RF27. Mostrar centro de costo.

RF9. Mostrar rol.

RF10. Buscar rol.

RF11. Eliminar rol.

RF28. Buscar centro de costo.

RF29. Eliminar centro de costo.

RF30. Descargar documentación.

RF12. Obtener listado de medios a partir del SGU.

RF13. Realizar conteo de medios a partir del scanner de barras.

RF14. Generar informe de auditoría en formato pdf.

RF15. Generar informe de auditoría en formato excel.

RF16. Generar gráfico a partir de resultado de auditoría.

RF17. Archivar resultado de auditoría.

RF18. Graficar historial de auditorías.

RF19. Generar informe comparativo entre centros de costo.

2.3.2 Requisitos no funcionales

Por Requisitos No Funcionales (en lo adelante RNF) se entiende el conjunto de propiedades o cualidades que el producto debe tener, características que lo hacen más agradable, rápido o confiable. Luego de analizado el entorno de desarrollo y futuro despliegue de la solución propuesta, para el desarrollo de la aplicación se detallan a continuación los siguientes RNF:

Requisitos de usabilidad: Aunque el sistema está orientado a usuarios con un dominio en el manejo de computadoras el mismo se caracterizará por un diseño sencillo e intuitivo que garantiza la facilidad para el trabajo de los usuarios. Los botones tienen descripciones acordes a las funcionalidades que realizan.

Requisitos de confiabilidad: La información que se maneja estará protegida de acceso no autorizado y divulgación. El sistema garantiza la gestión de roles y su asignación a los usuarios, lo cual permite que cada usuario tenga privilegios establecidos de acuerdo a su rol. Se recomienda cada tres meses la realización por parte del administrador del sistema de copias de seguridad a la Base de Datos para garantizar la persistencia de la información ante posibles fallos.

Requisitos de eficiencia: La naturaleza de la aplicación provoca que no se atiendan peticiones concurrentes por lo que los tiempos de respuesta serán cortos no sobrepasando en ningún caso los 3 segundos.

Restricciones de diseño: El diseño debe ser sencillo, con pocas entradas, donde no sea necesario mucho entrenamiento para utilizar el sistema. Se deben emplear los estándares establecidos en la estrategia marcaria de la producción de la UCI.

Requisitos de interfaz

Interfaces de hardware: Las estaciones de trabajo deben contar como mínimo con 256MB de RAM, al menos 30GB de disco duro y un procesador Pentium 4 o superior a una frecuencia mínima de 512 MHz.

Interfaces de software: Las estaciones de trabajo deben contar con sistema operativo instalado Windows 7 o superior, Ubuntu 12.4 o superior, o cualquier distribución de Linux con soporte en los repositorios para Java 8. PostgreSQL 9.5 o superior. Máquina virtual para la ejecución de Java 8, JVM (*del inglés Java Virtual Machine*) en el caso de Windows y se recomienda para Linux JRE (*del inglés Java Runtime Enviroment*).

Portabilidad

El sistema debe desarrollarse de forma tal que cuente con la posibilidad de ser multiplataforma.

2.4 Modelo de sistema

El modelo de Casos de Uso permite describir los RF del sistema, dando lugar a un acuerdo entre el cliente y los desarrolladores de la aplicación. Proporciona una explicación clara y consistente de lo que debería hacer el software, de modo que el modelo se use a lo largo del proceso de desarrollo. Posibilita que se obtenga una base para realizar verificaciones del producto informático, siendo la entrada fundamental para el análisis, el diseño y las pruebas (Larman, 2005).

2.4.1 Actores del sistema

Un actor es una entidad externa al sistema representada por un ser humano, una máquina o un software que interactúa con el sistema. Representa un tipo particular de usuario del negocio más que un usuario

físico, debido a que varios usuarios físicos pueden realizar el mismo papel en relación al negocio (Schmuller, 2010). En la Tabla. 4 se muestran los actores del sistema y las funciones correspondientes a cada uno:

Tabla. 4 Actores del Sistema.

Actor	Descripción
Administrador	Es el súper usuario del sistema, con permisos para agregar, eliminar y asignar privilegios a los usuarios. Tiene acceso a todas las funcionalidades del sistema.
Usuario	Solo cuenta con los permisos de autenticación y descargar la documentación de una auditoría determinada.
Responsable	Puede realizar las auditorías, así como generar los reportes correspondientes a las mismas y graficar los resultados pertinentes. Puede realizar además las mismas funciones que el Usuario.
Responsable de Área	Es el encargado de solicitar una auditoría, archivar una auditoría y graficar comportamiento histórico. Por su rol en el sistema puede realizar las mismas funciones del Responsable y las del Usuario.
Vicedecano	Es el encargado de obtener las estadísticas generales, además de poder realizar las mismas funciones que realiza el Responsable del Área, así como las del Responsable y el Usuario.

2.4.2 Diagrama de Casos de Uso del Sistema

El diagrama de Casos de Uso del Sistema (en lo adelante DCUS), es el encargado de recoger el comportamiento de un sistema desde el punto de vista de los usuarios. Es utilizado para describir las especificidades de un software y documentar su comportamiento, mostrando la relación que existe entre el usuario final y las funcionalidades (Schmuller, 2010).

A continuación se muestra mediante la Fig. 7 el DCUS de la solución propuesta, cabe destacar que el caso de uso “Solicitar Auditoria” interactúa con el Sistema de Gestión Universitaria (SGU) referido en este documento en el epígrafe 1.3 Análisis de soluciones similares.

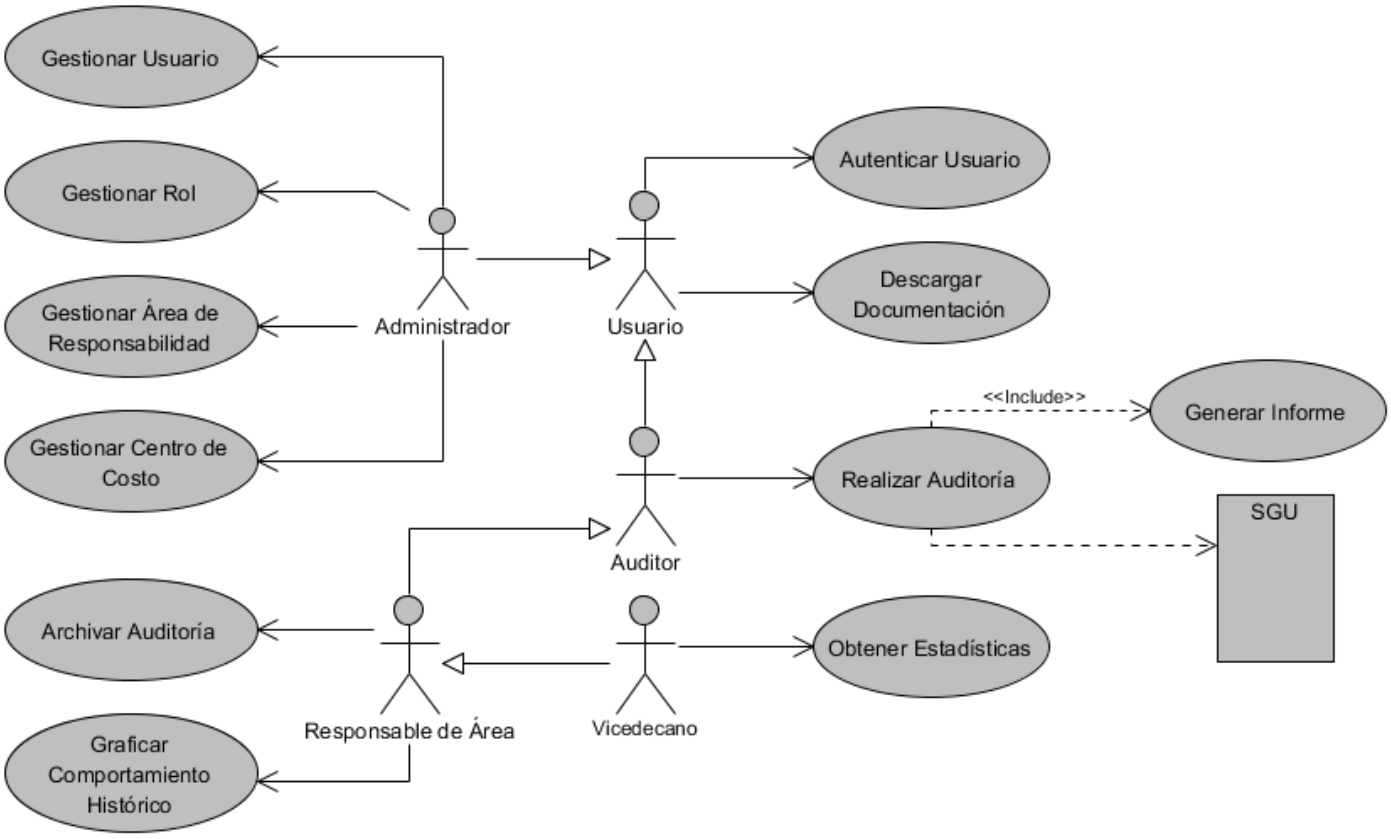

Fig. 7 Diagrama de Casos de Uso del Sistema.

2.4.3 Descripción textual de los Casos de Uso del Sistema

A continuación se especifica la descripción textual del Caso de Uso del Sistema: "Realizar Auditoría".

Tabla. 5 Descripción textual del Caso de Uso del Sistema: "Realizar Auditoria".

Caso de Uso:	Realizar Auditoría.
Actores:	Responsable, Vicedecano y Responsable de Área (inician).
Propósito:	Este Caso de Uso se lleva a cabo con el objetivo de realizar una auditoría de los activos fijos tangibles de un área, concluyendo con la obtención de un informe resultante de la misma.
Resumen:	Este Caso de Uso se inicia cuando el responsable de la auditoría selecciona la opción de descargar el listado de los AFT del área que desea auditar y culmina con la inspección de la totalidad de los medios contenidos en dicha área.
Precondiciones:	El actor debe estar autenticado en el sistema y debe tener permisos para realizar la auditoría acorde al nivel de responsabilidad del usuario en cuestión.
Referencias:	***RF** 12, **RF** 13, **RF** 14.*
Prioridad:	Crítico.
Flujo Normal de Eventos	

Acción del Actor:	Respuesta del Sistema:
1 El caso de uso inicia cuando el responsable selecciona o despliega la opción “Realizar auditoría”.	**2** El sistema muestra las áreas de responsabilidad de acuerdo al nivel del usuario.
3 EL usuario selecciona el área, o las áreas que desea auditar.	**4** El sistema obtiene el listado de Activos Fijos Tangibles de él o las áreas a auditar.
5 El responsable de la auditoría introduce en el sistema el identificador del AFT mediante un lector de código de barras.	**6** El sistema muestra una interfaz de color verde con un mensaje: “AFT registrado”, en caso contrario ver flujo alterno **6**, muestra una interfaz con los atributos del medio y la posibilidad de editarlos.
7 El usuario no edita ningún atributo del activo fijo tangible, caso contrario ver flujo alterno **7**.	**8** El sistema muestra un mensaje: “Continúe la auditoría o elija terminar”.
9 El usuario cliquea el botón terminar auditoría, caso contrario ver flujo alterno **9**.	**10** El sistema muestra un mensaje: “Auditoría concluida”.
Prototipo de Interfaz	

Flujos Alternos	
Acción de Actor:	**Respuesta del Sistema:**
	6 El sistema muestra un mensaje: “AFT no registrado o código no escaneado”.
7 El usuario edita alguno de los campos del activo.	**7.1** El sistema verifica que los datos introducidos sean válidos. Caso contrario ver flujo alterno **7.3.** **7.2** El sistema verifica que no existan campos vacíos. Caso contrario ver flujo alterno **7.4.**

	7.3 El sistema muestra un mensaje: “Existen datos no válidos”. **7.4** El sistema muestra un mensaje: “Existen campos vacíos”.
9 El usuario continúa con el proceso de auditoría, paso **5**.	
Postcondiciones:	Queda realizado un conteo y comparación de la existencia de los AFT de un área determinada.
Casos de Uso Incluidos:	Generar Informe.

2.4.4 Patrones de Casos de Uso utilizados

A continuación se describen los patrones de Casos de Uso que fueron aplicados en la construcción del diagrama de Casos de Uso del Sistema:

Concreta Inclusión

Este patrón concreta inclusión se evidencia al establecerse una relación de inclusión por particionamiento entre el caso de uso del sistema “Realizar Auditoría” y el Caso de Uso del Sistema “Generar Informe”. Una relación de inclusión es una relación desde un Caso de Uso base a un Caso de Uso de inclusión, que especifica cómo el comportamiento definido para el Caso de Uso de inclusión se inserta explícitamente dentro del comportamiento definido para el Caso de Uso base. Se utiliza para dividir partes de un flujo de trabajo de cuyos resultados, y no del método para obtenerlo, depende el Caso de Uso base. Se puede hacer esta partición si simplifica la comprensión del Caso de Uso base o si el comportamiento separado puede reutilizarse en otros Casos de Uso (SCHMULLER, 2010).

Múltiples actores rol común

Una relación de generalización de una clase hija de actor a otra clase padre de actor indica que el hijo hereda el rol que la clase padre puede jugar respecto a un Caso de Uso. Varios actores pueden jugar el mismo rol en un Caso de Uso particular, como se pone de manifiesto en el diagrama de Casos de Uso del Sistema con la generalización/especialización establecida entre el actor “Responsable de Àrea” al asumir el rol del actor “Responsable” (Schmuller, 2010).

CRUD Completo

Este patrón consiste en un caso de uso para administrar información, nos permite modelar las diferentes operaciones para administrar una entidad de información, tales como crear, leer, cambiar y eliminar o dar de baja. El mismo se manifiesta en los Casos de Uso: “Gestionar Rol”, “Gestionar Usuario”, “Gestionar Área de Responsabilidad” y “Gestionar Centro de Costo”(Schmuller, 2010).

2.5 Elementos de la arquitectura

Según (P. Clements et al., 2003) “la arquitectura de software de un sistema es la estructura o estructuras del sistema, lo cual abarca componentes de software, las propiedades visibles externamente de esos componentes, y las relaciones entre ellas”. La arquitectura de software permite representar de forma concreta la estructura y funcionamiento interno de un sistema.

Al diseñar una arquitectura de software se debe crear y representar componentes que interactúen entre ellos y tengan asignadas tareas específicas, además de organizarlos de forma tal que se logren los requerimientos establecidos. Se puede iniciar con patrones de soluciones ya probados, y utilizar modelos que han funcionado. Estas soluciones probadas se conocen como estilos arquitectónicos o patrones arquitectónicos, que van de lo general a lo particular.

2.5.1 Patrones arquitectónicos y de diseño

MVC como patrón arquitectónico utilizado

Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones a un alto nivel de abstracción. La propuesta de solución se construye a partir del patrón denominado **Modelo Vista Controlador** (MVC) el cual tiene la intención de particionar las aplicaciones interactivas en tres componentes independientes, tal como su nombre lo indica estos son: Modelo o *Model*, Vista o *View*, y Controlador o *Controler*(Daniel Pages Chacón, 2008).

El primero de estos elementos representa el modelo de datos de la aplicación, la estructura de los mismos y la lógica de acceso a ellos. El segundo se refiere a la capa de presentación donde serán mostrados los datos de una o varias maneras con sus correspondientes comportamientos visuales, y a través de la cual se interactúa con la aplicación para acceder a la información. Por su parte, el último de estos elementos se encarga de contener y procesar la lógica que debe seguirse en la aplicación. Aquí se procesan los eventos

manipulados por los usuarios, los cuales pueden desencadenar actualizaciones de los datos o una manipulación directa de estos en la vista. Controla la interacción entre la Vista y el Modelo y las secuencias de peticiones con sus respectivas respuestas(Daniel Pages Chacón, 2008).

Este patrón propone una comunicación entre estos tres elementos o partes del sistema a través de interfaces que son brindadas por cada uno, de manera que se mantenga un alto desacoplamiento entre ellos, pudiéndose realizar variaciones en cualquiera, sin que el resto de la aplicación se vea afectada. La Fig. 8(Dudney B, 2004) representa la distribución en paquetes correspondiente a la arquitectura propuesta en el Sistema para la Informatización del proceso de Auditoría y Control de los Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales(Daniel Pages Chacón, 2008).

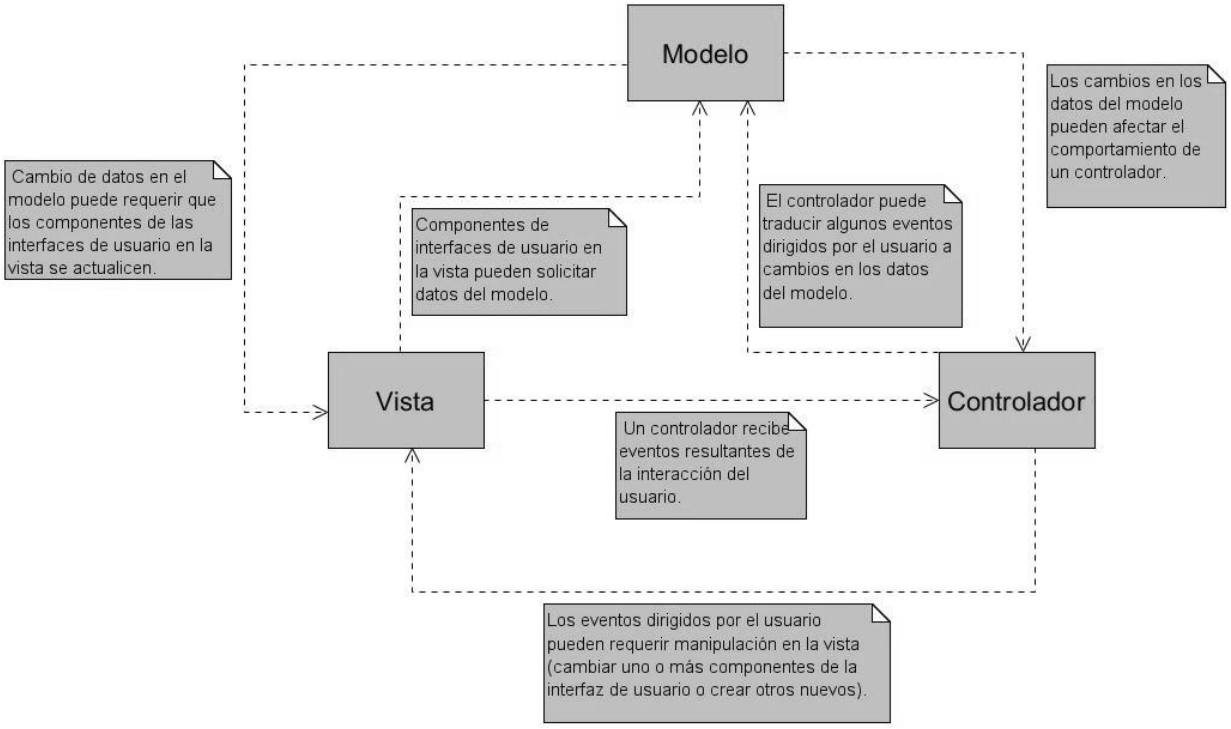

Fig. 8 Patrón de Diseño Modelo-Vista-Controlador.

Con MVC se hacen palpables varios beneficios tales como:

- Se hace más eficiente la construcción de aplicaciones que requieren diferentes representaciones de una misma información dado que permite simplemente construir vistas distintas que interactúen con el mismo controlador y la misma estructura o modelo de datos.

- De la misma forma pueden ser representadas diferentes interfaces condicionadas por distintos look and feel, a su vez establecidos por la necesidad de un desarrollo necesario para diferentes Sistemas Operativos (SO), o sea multiplataforma.
- Se hace palpable la reusabilidad de componentes de interfaz ya que estos no se encuentran integrados con ninguna información de lógica de negocio o de acceso a datos.
- El desacoplamiento con el modelo y el controlador permite que los cambios realizados en estas capas no afecten a las restantes, lo cual permite mayor eficiencia en el desarrollo y mantenimiento, dado que ya no se emplea tiempo ni esfuerzo en realizar cambios en toda la aplicación por el hecho de que sus partes se encuentren con un alto grado de acoplamiento.

Patrones de Diseño utilizados

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. Proporciona un esquema para refinar los subsistemas o componentes de un sistema de software, o las relaciones entre ellos. Son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Son soluciones basadas en la experiencia y que se ha demostrado que funcionan(Buschmann et al., 1996).

Patrones Generales de Software para Asignar Responsabilidades (GRASP)

Los GRASP (del inglés *General Responsibility Assignment Software Patterns*) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones(Larman, 2005).

Para la asignación de responsabilidades en la aplicación se utilizó el patrón **Experto (*Expert*)** el cual indica que la clase que cuenta con la información necesaria para cumplir la responsabilidad es la responsable de manejar la información(Larman, 2005). De este modo se obtendrá un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento). Sus ventajas se centran en que mantiene el encapsulamiento, los objetos utilizan su propia información para llevar a cabo sus tareas, se distribuye el comportamiento entre las clases que contienen la información requerida y son más fáciles de entender y mantener. La clase “Activo” como ejemplo, contiene la información referente a un AFT y la misma es la encargada de proporcionar esta información a los componentes que la demandan y facilitar su actualización sobre la Base de Datos.

El empleo de la programación orientada a objetos provoca la adopción del patrón **Creador (*Creator*)** aplicado en cualquier caso de agregación o composición de clases. Este patrón ayuda a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases. Una de las consecuencias de usar este patrón es la visibilidad entre la clase creada y la clase creadora. Una ventaja es el bajo acoplamiento, lo cual supone facilidad de mantenimiento y reutilización(Larman, 2005). Se manifiesta con claridad el patrón creador mediante la relación existente entre las clases "CentroCosto" y "Área", manteniendo las buenas prácticas del encapsulamiento además, al determinar la composición de un Centro de Costo por una o varias Áreas de Responsabilidad permitiendo la visibilidad en este propio sentido y responsabilizando a la clase contenedora de la creación de objetos de la clase contenida.

Para propiciar la reutilización de código se tuvo en cuenta el patrón **Bajo Acoplamiento (*Low Coupling*)** que indica una menor dependencia entre clases asegurando facilidad en las labores de mantenimiento y soporte(Larman, 2005). El diseño de la propuesta de solución propicia que las clases que se encuentran en la capa Modelo no presenten fuertes dependencias o relaciones directas con las clases contenidas dentro de la capa Controlador o Vista, brindando así la posibilidad de realizar transformaciones en estas independientemente al resto del sistema; así mismo, la manera en la que se maneja la relación entre las clases pertenecientes a la capa Controlador y Vista permiten agregar o retirar funcionalidades sin incurrir en cambios traumáticos para el sistema, propiciando la reutilización y el mantenimiento del mismo.

La propuesta de solución emplea además el patrón **Alta Cohesión (*High Cohesion*)**. Los conceptos de cohesión y acoplamiento están íntimamente relacionados. Un mayor grado de cohesión implica un menor grado de acoplamiento. Maximizar el nivel de cohesión intra-modular en todo el sistema resulta en una minimización del acoplamiento inter-modular. Este patrón plantea que la información que almacena una clase debe de ser coherente y debe estar (en la medida de lo posible) relacionada con la clase(Larman, 2005). El diseño de la aplicación tiene muy en cuenta este aspecto, tratando con independencia de conceptos y apego a la coherencia de la información cada una de sus clases. Un ejemplo del empleo del patrón Alta Cohesión se pone de manifiesto en la relación que establece la clase "ActivoService", la cual establece la comunicación con la clase "ActivoDao", manejando los datos provistos por "Activo", estas clases son las responsables en toda la aplicación del manejo de los datos referentes a los AFT. De manera análoga se comportan los restantes conceptos manejados en el sistema a través de un diseño similar.

Estos dos últimos patrones no deben tratarse de forma independiente pues poseen relaciones que mantienen el equilibrio entre clases, equilibrando las responsabilidades y garantizando que las clases sean creadas con un buen diseño donde los objetos sean capaces de interactuar.

Patrones GOF

La línea base en el tema de patrones de diseño la impone el catálogo "Design Patterns: Elements of Reusable Object-Oriented Software", en este libro se presenta un conjunto de 23 patrones de diseño identificados a partir del estudio y la experiencia del grupo llamado Banda de los Cuatro o The Gang of Four (GOF): Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides, quienes se dedicaron a analizar los problemas recurrentes en el desarrollo de software. A continuación se describen los patrones propuestos por la banda de los cuatro mayormente empleados en la aplicación(Larman, 2005).

Instancia única: Puesto en práctica con el fin de garantizar que una clase solo tenga una única instancia, proporcionando un punto de acceso local a la misma (Larman, 2005). El uso de este patrón viene dado por el acceso dentro de la aplicación del "*SGUService"*, instancia de la clase con igual nombre, la cual se encarga del acceso a los métodos relacionados con el manejo de datos del sistema.

Agente Remoto: Utilizado en el caso de que el sistema en cuestión deba comunicarse con un servicio externo (Larman, 2005). En este caso se propone crear una clase de software local que representa al componente externo y asignarle la responsabilidad de contactar con el componente real. Su utilización en el sistema tiene como objetivo establecer la comunicación con el servicio provisto por el Sistema de Gestión Universitaria (SGU) brindado por la Universidad de las Ciencias Informáticas, para la obtención de datos referentes a los AFT pertenecientes a un Área de responsabilidad o centro de costo determinado, así como los responsables previamente asignados a los mencionados mecanismos de agrupamiento y organización de los medios. Un ejemplo de su empleo lo constituyen las clases "*UsuarioService", "ÁreaService", "CentroCostoService", "ActivoService", "AuditoríaActivoService", "AuditoríaService"* las cuales actúan como agente remoto entre el servicio provisto por el SGU y el sistema propuesto, funcionando como proveedores de datos de acuerdo a su propia especialización y cohesión.

Composición: Combina objetos en estructuras de árbol para representar jerarquías de parte-todo. Permite que los clientes traten de manera uniforme a los objetos individuales y a los compuestos(Larman, 2005). Un ejemplo del mencionado patrón se manifiesta en la propuesta de solución en el empleo de una jerarquización

mediante relaciones de dependencia entre las clases “SGUService” como raíz del árbol y “ÁreaService”, “UsuarioService”, “CentroCostoService”, “ActivoService”, “AuditoríaActivoService”, y “AuditoríaService” como ramas primarias, así mismo se extiende la estructura hacia las clases “ÁreaDao”, “UsuarioDao”, “CentroCostoDao”, “ActivoDao”, “AuditoríaActivoDao” y “AuditoríaDao”, concluyendo la jerarquización con las clases que representan el nivel más bajo del modelo, permitiendo un tratamiento homogéneo de los datos independientemente de su proveedor.

Los **Objetos de Acceso a Datos** (**DAO**, *Data Access Object*) son un Patrón de los subordinados de Diseño Core J2EE y considerados una buena práctica, aunque ya no dentro de los propuestos por la Banda de los Cuatro se considera importante abundar en sus características dado el papel que desempeñan en el diseño de la propuesta de solución. La ventaja de usar objetos de acceso a datos es que cualquier objeto de negocio no requiere conocimiento directo del destino final de la información que manipula. Los Objetos de Acceso a Datos pueden usarse en Java para aislar a una aplicación de la tecnología de persistencia Java subyacente. Usando Objetos de Acceso de Datos significa que la tecnología subyacente puede ser actualizada o cambiada sin transformar otras partes de la aplicación(Román, 2010). La Fig. 9 muestra una representación general de un diseño DAO.

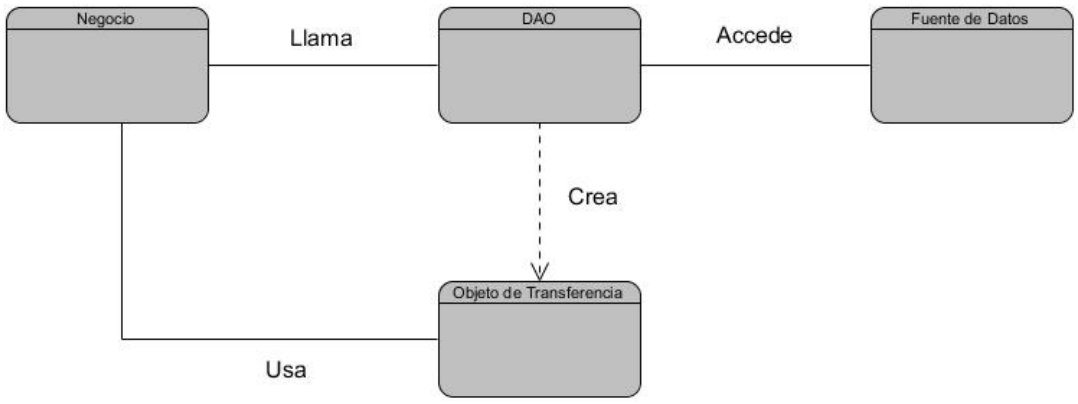

Fig. 9 Patrón de diseño Objeto de Acceso a Datos (DAO).

El mencionado patrón se evidencia con claridad en el Diagrama de Clases del Diseño propuesto para el Caso de Uso del Sistema “Realizar Auditoria”, explicitado a continuacion dentro del epigrafe 2.6 “Modelo de Diseño”.

2.6 Modelo de Diseño

El modelo del diseño describe la realización física de los Casos de Uso y se corresponde directamente con los elementos físicos del ambiente de implementación. Consiste en colaboraciones de clases, que pueden ser agregadas en paquetes y/o subsistemas (Jacobson, 2007).

2.6.1 Diagrama de Clases del Diseño

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación y permiten modelar la vista de diseño del sistema(Jacobson, Booch and Rumbaug, 2007). Mediante la Fig. 10 se muestra el diagrama de clases del diseño empaquetado según los componentes arquitectónicos, correspondiente al Caso de Uso: Realizar Auditoria:

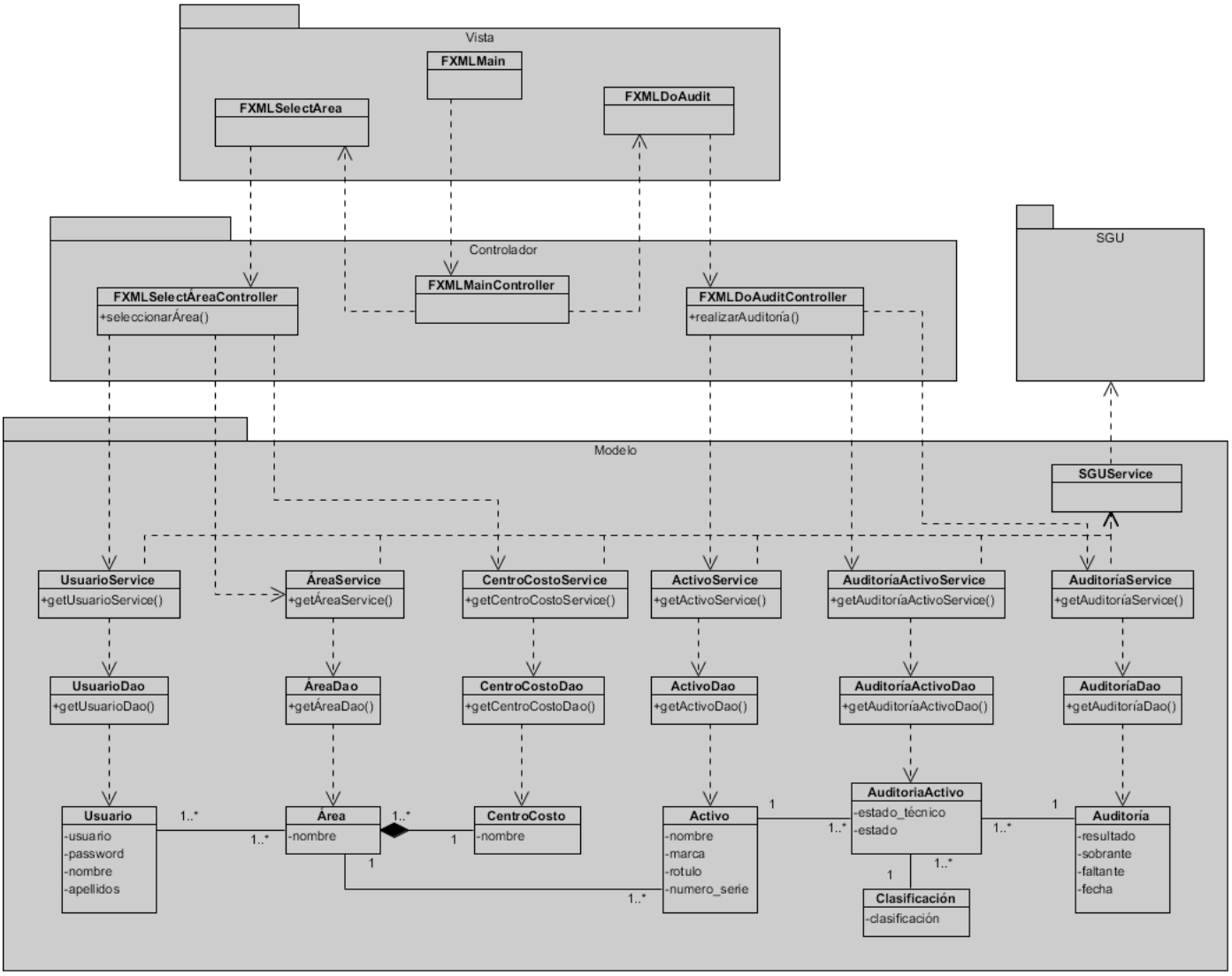

Fig. 10 Diagrama de Clases del Diseño de la Propuesta de Solución.

A partir del diagrama propuesto en la figura anterior se puede evidenciar el empleo de los patrones de diseño expuestos en el epígrafe anterior, persiguiendo las ventajas que proporciona su empleo. Por otra parte el patrón arquitectónico Modelo Vista Controlador rige la estructura de las clases y su distribución y agrupación en tres paquetes respectivos a las tres capas del mencionado esquema.

2.7 Diagrama Entidad Relación

Un diagrama Entidad-Relación (DER) tiene como objetivo representar mediante una notación gráfica los objetos de datos y sus relaciones. Este define todos los datos que se introducen, se almacenan, se transforman y se producen dentro de una aplicación(Pressman, 2010). Seguidamente en la Fig. 11 se muestra la estructura física de la Base de Datos del sistema:

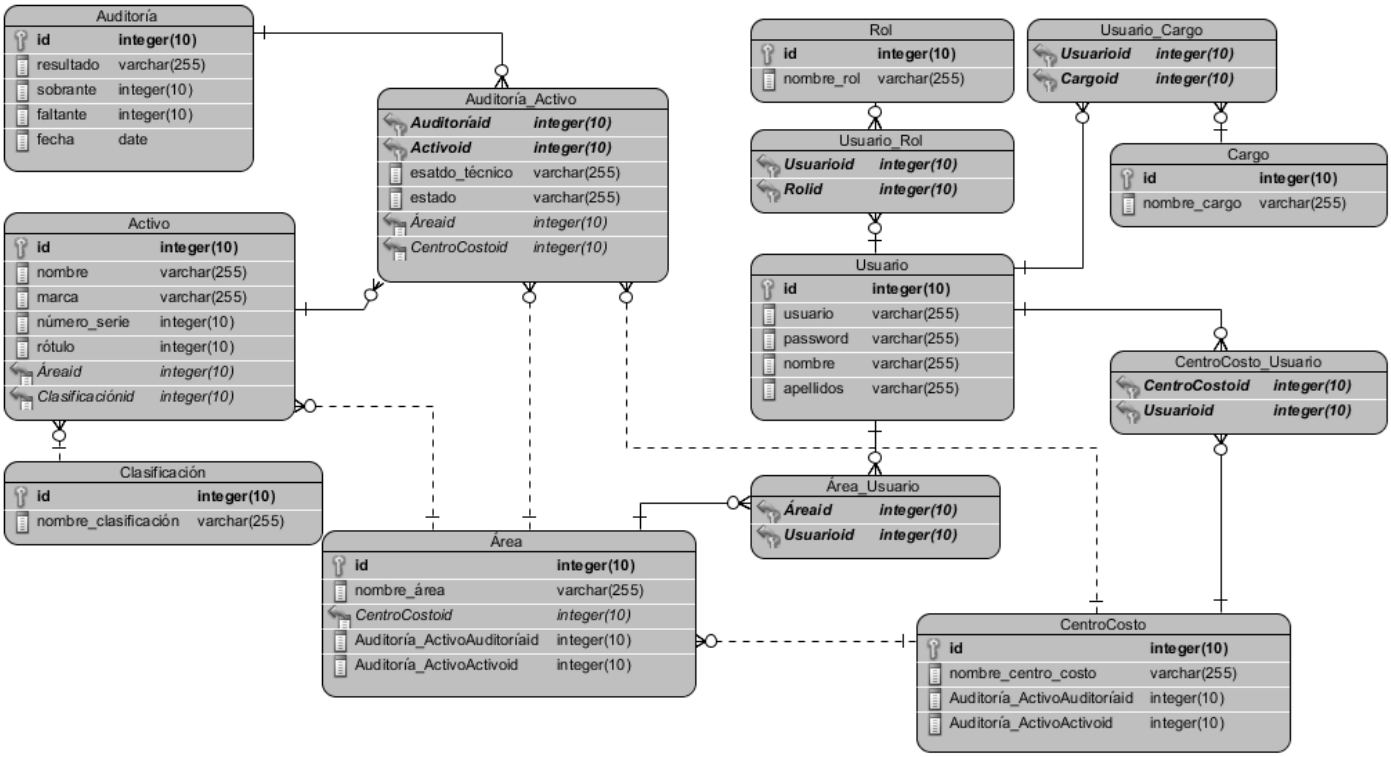

Fig. 11 Diagrama Entidad Relación de la Propuesta de Solución.

2.8 Conclusiones Parciales

El modelado del negocio, la obtención de los Requisitos Funcionales y No Funcionales, de conjunto con el modelado del sistema, propician el entendimiento entre el equipo de desarrollo y el cliente de la aplicación en función de lo que el sistema debe realizar y las características que debe poseer. Con la obtención de los artefactos fundamentales correspondientes al flujo de Diseño que dicta la metodología de desarrollo se garantizan las labores futuras de mantenimiento y soporte de la propuesta de solución, estableciendo una guía eficaz ante las labores de implementación.

Capítulo 3: Implementación y Validación de la Solución Propuesta

En el presente capítulo se abordan las fases correspondientes a la implementación, despliegue y validación de la propuesta de solución. Contar con un producto final que cumpla los requisitos de una manera lo suficientemente aceptable, que implique su aplicación satisfactoria, es una meta indispensable; es por ello que se hace necesaria la comprobación para asegurar que satisface su especificación y entrega las funcionalidades esperadas. Los principales aportes relacionados con los procesos sustantivos de la entidad en los que impacta el software que se construye son abordados al final del presente capitulo.

3.1 Modelo de Implementación

El modelo de implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes, como ficheros de código fuente, ejecutables, entre otros. Es otra forma de representar una vista del sistema, que muestra la organización y dependencia que existe entre los componentes físicos que se necesitan para ejecutar la aplicación. El modelo de implementación describe además cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros(Sommerville, 2005).

3.1.1 Representación de dependencias entre componentes de software

Un componente de software es una parte física de un sistema, pudiendo considerarse la personificación en software de una clase. La representación de dependencias entre componentes de software puede considerarse como representación tácita del modelo de implementación, en ella se pueden incluir componentes de código fuente, componentes del código binario, y componentes ejecutables (Jacobson, Booch and Rumbaug, 2007). A continuación se presenta el diagrama de componentes correspondiente al Caso de Uso: Realizar Auditoría, donde se han incluido los componentes fundamentales y sus dependencias,

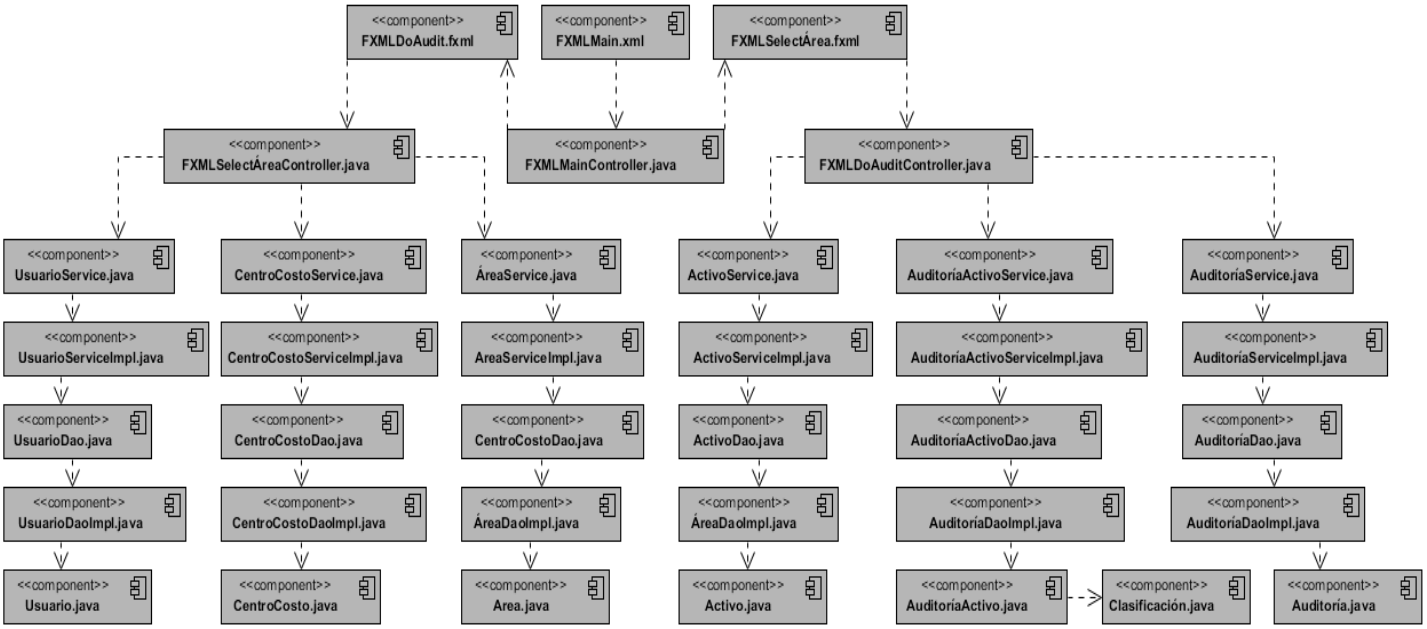

Fig. 12 Diagrama de componentes. Caso de Uso: Realizar Auditoría.

3.2 Modelo de Despliegue

El propósito del modelo de despliegue es capturar la configuración de los elementos de procesamiento y las conexiones entre estos elementos en el sistema. Permite el mapeo de procesos dentro de los nodos, asegurando la distribución del comportamiento a través de aquellos que son representados(Pressman, 2010). En la Fig. 13 se muestra una representación del diagrama de despliegue propuesto para el sistema y en la Tabla. 6 la descripción de sus nodos.

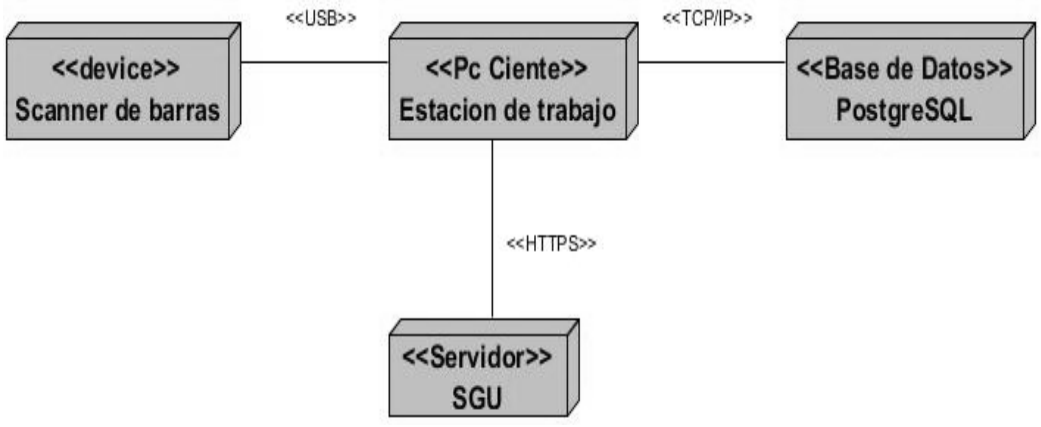

Fig. 13 Representación del Diagrama de Despliegue del Sistema.

Tabla. 6 Descripción de los nodos correspondientes al diagrama de despliegue del sistema.

Nodos	Descripción
Scanner de Barras	Nodo que representa el dispositivo que permite la fácil recepción y comparación de los rótulos en los AFT transmitiéndolos hacia la aplicación.
Estación de Trabajo Portátil	Nodo que representa la computadora o estación de trabajo donde será instalado el Sistema para la Informatización del Proceso de Auditoría y Control en la Facultad CITEC.
Servidor de Base de Datos	Nodo que representa el servidor donde se encuentra la Base de Datos que contiene los datos persistentes de la aplicación.
SGU	Sistema de Gestión Universitaria, aplicación web que almacena los datos de los AFT de acuerdo a las áreas de responsabilidad y centros de costo de la Facultad CITEC.

3.3 Estándares de Codificación

Los estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física del código. Las normas de codificación definidas por la línea de arquitectura están enfocadas a lograr una mejor comprensión y mantenimiento del código que responda a la complejidad dada por la cantidad de componentes y el alto nivel de integración que puede existir en el desarrollo del software(Quevedo, 2016).

Las realizaciones de revisiones frecuentes al código, la aplicación de forma continua de técnicas y estándares de codificación definidos, junto al empleo de buenas prácticas de programación, garantizan una alta calidad en el desarrollo de la solución además de proporcionar un código legible y reutilizable. Entre los estándares utilizados durante la implementación se encuentran:

Tamaño y organización de las líneas de código

La longitud de línea es aproximadamente de 80 caracteres. En caso de que una expresión ocupe más de este rango, podrá romper o dividirse en una nueva línea y deberán estar todas alineadas con respecto a la anterior, para mantener la legibilidad del código.

Declaraciones

Toda variable local tendrá que ser inicializada en el momento de su declaración, salvo que su valor inicial dependa de una llamada a otro método en determinado momento de ejecución de método. Las

declaraciones deben situarse al principio de cada bloque principal en el que se utilicen y nunca en el momento de su uso.

Llaves de apertura y cierre

Se utiliza siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales. Esto aumenta la legibilidad y disminuye la probabilidad de errores lógicos.

Nomenclaturas Utilizadas

PascalCase establece que los nombres de los identificadores, las variables, métodos y clases están compuestos por una o más palabras juntas. Con ella cada palabra inicia con letra mayúscula y el resto en minúscula. Todas las clases están nombradas con la utilización del estándar PascalCase, nombrándolas de acuerdo al propósito y la función que corresponda. Ejemplo: las clases “UsuarioDao”, “ÁreaDao”, “CentroCostoDao”, “ActivoService”, y “AuditoríaDao”.

CamelCase es similar a PascalCase con diferencia en la letra inicial del identificador no comienza con mayúscula. Esta notación se utilizó para el nombre de funciones, atributos y variables.

Nomenclatura de las funciones: El identificativo de todas las funciones o métodos se escribe con la primera palabra en minúscula de acuerdo a la función que realizan. Ejemplo: “getActivoDao()” y “getÁreaDao()”.

Nomenclatura de los atributos: El identificativo de los atributos se escribe de acuerdo a su objetivo con la primera letra en minúscula. Ejemplos: “estadoTécnico” y “númeroSerie”.

La siguiente figura contiene un fragmento de código fuente donde se evidencian los estándares de codificación antes descritos.

```
@FXML
private void updateSaveAction(ActionEvent event) {
    List<Rol> selected_roles = roles_table.getSelectionModel().getSelectedItems();
    List<Centro_Costo> selected_cost_centers = cost_centers_table.getSelectionModel().getSelectedItems();
    List<Area> selected_areas = areas_table.getSelectionModel().getSelectedItems();
    Set<Rol> roles_set = new HashSet<Rol>(selected_roles);
    Set<Centro_Costo> cost_centers_set = new HashSet<Centro_Costo>(selected_cost_centers);
    Set<Area> areas_set = new HashSet<Area>(selected_areas);
    user.setUsuario(username.getText());
    user.setPassword(password.getText());
    user.setRoles(roles_set);
    user.setCentros_costo(cost_centers_set);
    user.setAreas(areas_set);
    user_service.updateUsuario(user);
}
```

Fig. 14 Fragmento de código de la propuesta de solución.

3.4 Pruebas de software

El desarrollo de un producto de software implica una serie de actividades las cuales se encuentran lejos de estar exentas de errores humanos. La prueba del software es un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación(Pressman, 2010).

Según (Pressman, 2010) se establecen una serie de reglas que pudieran entenderse como los objetivos fundamentales del proceso de pruebas:

- La prueba es un proceso de ejecución de un programa con la intención de descubrir un error.
- Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.
- Una prueba tiene éxito si descubre un error no detectado hasta entonces.

Los procesos de prueba para las aplicaciones web comienzan con pruebas que ejercitan el contenido y la funcionalidad de la interfaz que es inmediatamente visible para los usuarios finales. Conforme se realizan las pruebas, se ejercitan los aspectos de la arquitectura de diseño y la navegación. Finalmente la atención se centra en las pruebas que ejercitan las capacidades tecnológicas referentes a la infraestructura de la aplicación y cuestiones de instalación e implementación(Pressman, 2010).

3.4.1 Tipos de Pruebas

Con el fin de llevar a cabo la evaluación de la propuesta de solución actual, atendiendo a las características y objetivo del sistema, se seleccionó la aplicación de dos tipos de prueba correspondientes a las pruebas a nivel de componentes.

Las pruebas a nivel de componentes, también llamadas pruebas de función, se enfocan sobre un conjunto de pruebas que intentan descubrir errores en la aplicación. Cada función es un módulo de software el cual se determina probar empleando el método de prueba de Caja Negra.

Las **pruebas de Caja Negra** se centran en los RF del software y forman parte de las pruebas de sistema. Estas se emplean cuando se conoce la función específica para la que se diseñó el producto; se aplican a la interfaz del software intentando demostrar que cada función es plenamente operacional, mientras se buscan los errores de cada función.

El mencionado método utiliza además técnicas para su realización, en la presente investigación se determina el uso de la técnica de Partición Equivalente, la que divide el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba. Un Caso de Prueba representa una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las que ha de probarse (Jacobson, Booch and Rumbaug, 2007).

Por otra parte las **pruebas de aceptación**, son llevadas a cabo por el cliente, básicamente constituyen pruebas funcionales, sobre el sistema completo, y buscan una cobertura de la especificación de requisitos. Estas pruebas no se realizan durante el desarrollo, pues sería impresentable al cliente; sino que se realizan sobre el producto terminado e integrado o pudiera ser una versión del producto o una iteración funcional pactada previamente con el cliente.

3.5.1 Diseño de casos de prueba

Según (Peña, 2006) “el conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo particular como, por ejemplo, ejercitar un camino concreto de un programa o verificar el cumplimiento de un determinado requisito” es a lo que podríamos llamar o determinar cómo caso de prueba.

A continuación se muestra el diseño del caso de prueba (en lo adelante DCP) correspondiente al Caso de Uso "Realizar Auditoría", acorde al método y técnica de evaluación descritos en el anterior epígrafe.

Diseño de Casos de Prueba del Caso de Uso: "Realizar Auditoría"

- **Descripción general**

Este Caso de Uso se lleva a cabo con el objetivo de realizar una auditoría de los activos fijos tangibles de un área, concluyendo con el testeo de los medios.

- **Condiciones de ejecución**

El usuario debe estar autenticado en el sistema y debe tener permisos para realizar una auditoría de acuerdo al nivel de responsabilidad del mismo.

- **Secciones a probar en el Caso de Uso: "Realizar Auditoría"**

Tabla. 7 Secciones a probar en el Caso de Uso del Sistema: "Realizar Auditoría".

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: "Flujo Normal de Eventos".	**EC 1.1:** El usuario inserta y selecciona datos correctos en los campos de entrada. Escanea los rótulos de los medios.	El sistema va comparando la existencia de los medios de acuerdo al listado obtenido desde el SGU, en caso de variar alguna de sus características las mismas son actualizadas.	▪ Clic en "Comenzar auditoría". ▪ Empleo del scanner en el testeo de rótulos. ▪ Inserta datos en los campos de edición de los AFT. ▪ Clic en el botón "Actualizar".
	EC 1.2: El usuario deja campos vacíos.	El sistema muestra un mensaje indicando que se deben especificar todos los campos para actualizar las características de un activo.	▪ Clic en "Comenzar auditoría". ▪ Empleo del scanner en el testeo de rótulos. ▪ Inserta datos en los campos. ▪ Clic en el botón "Actualizar".

	EC 1.3: El usuario introduce datos incorrectos.	El sistema muestra un mensaje indicando que existen datos incorrectos.	▪ Clic en "Comenzar auditoría". ▪ Empleo del scanner en el testeo de rótulos. ▪ Inserta datos en los campos. ▪ Clic en el botón "Actualizar".

- **Descripción de las Variables**

Tabla. 8 Descripción de las Variables.

No.	Nombre del Campo	Clasificación	Valor Nulo	Descripción
1	Nombre	Campo de texto	No	Debe ser un nombre propio válido. Solo letras.
2	Marca	Campo de texto	No	Puede ser cualquier campo, combinación de caracteres, letras números, signos, símbolos.
3	Número de Serie	Campo de texto	No	Debe ser una combinación de números únicamente.
4	Rótulo	Campo de texto	No	Puede ser cualquier campo, combinación de caracteres, letras números, signos, símbolos.
5	Área	Campo de texto	No	Debe ser el nombre de un Área de la Facultad CITEC válida. Solo letras y Números.
6	Clasificación	Lista Desplegable	No	Debe escogerse entre las opciones de las posibles clasificaciones.
7	Estado	Lista Desplegable	No	Debe escogerse entre las opciones de los posibles estados.

- **Matriz de Datos**

Tabla. 9 Matriz de datos: Realizar Auditoría.

Id. del escenario	Escenario	Nombre	Marca	Número de Serie	Rótulo	Área	Clasificación	Estado	Respuesta del sistema	Resultado de la prueba

EC 1.1	El usuario inserta y selecciona datos correctos en los campos de entrada. Escanea el código de barras del AFT correctamente.	V /(Monitor)	V/(Acer)	V/(1019282634)	V/(II29383)	V/(Dpto. Programación)	V/(Técnico)	V/(Bueno)	El sistema realiza el testeo del medio.	Satisfactorio
EC 1.2	El usuario deja campos vacíos.	V /(Monitor)	V/(Acer)	V/(1019282634)	V/(II29383)	V/(Dpto. Programación)	V/("")	V/("")	El sistema muestra un mensaje indicando que se deben especificar todos los campos para actualizar las características del medio.	Satisfactorio
EC 1.3	El usuario introduce datos incorrectos.	I /(Mo04040"nitor)	V/(Acer)	V/(1019282634)	V/(II29383)	I/(".$Rsrs)	V/(Técnico)	V/(Bueno)	El sistema muestra un mensaje indicando que existen datos incorrectos.	Satisfactorio

3.5.2 Aceptación de la propuesta de solución

Las pruebas de aceptación ponen en práctica una versión del sistema que podría ser entregado al cliente. Aquí, el equipo de pruebas se ocupa de validar que el sistema satisface sus requerimientos. Si la entrega es lo suficientemente buena, el cliente puede entonces aceptarla para su uso (Somerville, 2005).

La aplicación desarrollada fue sometida a un proceso de aceptación por parte del Vicedecanato de Economía y Administración (ver Anexo 2). Durante este proceso el colectivo del vicedecanato, ha reconocido que:

- La solución propuesta posee una interfaz agradable acorde a la identidad marcaria planteada por la UCI, resultando en botones y menús de fácil acceso. Puede ser operada con facilidad por personal con una cultura informática elemental, mostrando un alto grado de usabilidad.
- El funcionamiento de la aplicación es estable y consistente, con tiempos de respuesta no mayores a los 3 segundos para ninguna de sus operaciones.
- Facilita de manera sustancial el proceso de testeo de los AFT en cada una de las áreas auditadas, permitiendo agilidad en el proceso y confiabilidad en el resultado de las inspecciones.
- Permite la rápida obtención de informes y gráficos que ofrecen de manera interactiva y útil el resultado de las auditorías, resultando en recursos valiosos para la toma de decisiones y representando herramienta eficaz ante el control de los medios.

3. 6 Ejecución de las pruebas. Resultados

Como parte de la ejecución de las pruebas de Caja Negra, con el objetivo esencial de identificar en qué medida satisface la aplicación las funcionalidades implementadas, se realizó una primera iteración de pruebas, en la que fueron aplicados los diseños de casos de pruebas realizados a partir de las descripciones textuales de los Casos de Uso del Sistema. En esta primera iteración se identificaron un total de 7 no conformidades, distribuidas conforme se muestra en la Fig. 15. Una vez corregidas las no conformidades anteriores, se ejecutan nuevamente los diseños elaborados obteniéndose un total de tres no conformidades, correspondientes a los Casos de Uso “Realizar Auditoría” y “Gestionar Usuario”. Luego de corregidas estas últimas no conformidades se decide la realización de una tercera iteración de los diseños anteriormente aplicados obteniéndose resultados satisfactorios, por lo que se determina la no realización de una nueva iteración.

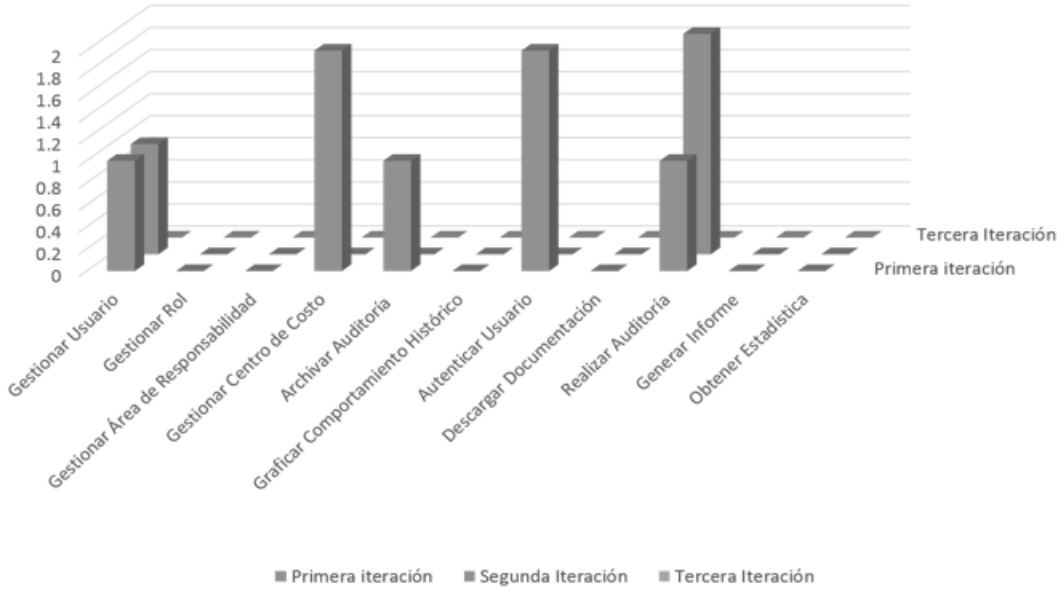

Fig. 15 Ejecución de las pruebas de caja negra sobre la aplicación.

Las no conformidades detectadas estuvieron relacionadas con el funcionamiento de las interfaces del usuario. Generalmente asociadas a mensajes de información no personalizados o respuestas no deseadas para los valores de entrada definidos. La realización de las pruebas permitió identificar los errores que no habían sido detectados hasta el momento.

3. 7 Conclusiones Parciales

Mediante el desarrollo del proceso de pruebas se pudieron identificar y resolver los errores en la implementación aumentando la calidad del producto final obtenido, se comprobó que el mismo satisface las funcionalidades requeridas y cumple con los Requisitos No Funcionales definidos.

Conclusiones Generales

1. El estado del arte referido informatización del proceso de auditoría y control permite afirmar que las soluciones que existen, que de alguna manera tributan a la investigación, no resuelven la problemática planteada, evidenciándose la necesidad de esta investigación.
2. Las herramientas, tecnologías y lenguajes propuestos para la construcción del sistema, se corresponden con las políticas de soberanía tecnológica que impulsa la UCI y el país.
3. Todos los RF que se definieron fueron debidamente implementados, igualmente, se incluyeron en la propuesta las exigencias de todos los Requisitos No Funcionales detectados.
4. La característica de ser multiplataforma amplía las posibilidades de utilización y la gama de usuarios que puede utilizar la solución.
5. Los artefactos generados durante el proceso de desarrollo del software permitirán escalar la solución en el futuro y facilitar las labores de mantenimiento y soporte.
6. La utilización de estilos y patrones promueve buenas prácticas en el desarrollo de la solución al proporcionar uniformidad en la implementación, siendo más entendible y escalable en el tiempo.
7. Los Diseños de Casos de Prueba desarrollados, como parte de las pruebas de Caja Negra, permitieron validar los requisitos de la aplicación con las funcionalidades implementadas.
8. La utilización de la herramienta aportará agilidad al proceso de auditoría y control de los activos fijos tangibles en la facultad de Ciencias y Tecnologías Computacionales de la Universidad de Ciencias Informáticas.

Recomendaciones

Se recomienda la puesta en práctica de iniciativas de informatización del proceso de auditoría y control en la totalidad de las facultades de la UCI, así como en otras entidades cubanas, con el fin de contribuir a lo dispuesto por la Contraloría General de la República, en consonancia con los Lineamientos de la Política del Partido y la Revolución respecto al control y autocontrol.

Referencias Bibliográficas

ABRAM, A. AND J. W. MOORE *SWEBOOK*. Edtion ed. California: IEEE Computer Society, 2008.

ABRAN, A. AND J. W. MOORE *Swebok. Guide to the Software Engineering Body of Knowledge*. Edtion ed. Estados Unidos de América: IEEE Computer Society Professional Practices Committee, 2004. ISBN 0-7695-2330-7.

BUSCHMANN, F., R. MEUNIER AND H. ROHNERT *Pattern-Oriented Software Architecture. A System of Patterns.* Edtion ed. Inglaterra: John Wiley and Sons, Inglaterra, 1996.

CABALLERO, Y. G. Sistema para Auditoría y Control de Activos Fijos Tangibles. Universidad de las Ciencias Informáticas, 2013.

CLARA, T. S. M. F. V. Versat Sarasola. 2001.

CONTRERAS, M. A. B. Administración y control interno de activo fijo. Facultad de Contaduría y Administración., 2013.

CORNELIO, O. M., B. B. FONSECA, Y. G. CABALLERO AND R. C. J. HERNÁNDEZ. Sistema para la auditoría y control de los Activos Fijos Tangibles. 2016, pp. 6.

CUBA, C. G. D. L. R. D. Sistema de Control Interno. 2012, pp. 84.

DANIEL PAGES CHACÓN, J. P. M. P. Principales variantes tecnológicas del entorno Java para el desarrollo de aplicaciones en la capa cliente 2008, 10.

DUDNEY B, L. J., WILLIS B, MATTINGLY L. Mastering JavaServer Faces. In. 14 Convención Científica de Ingeniería y Arquitectura: Willey Publishing Inc, 2004, vol. 460x345.

ECLIPSE. Ciclo de vida de un proyecto según OpenUP. In.: eclipse, 2010.

ERP, C. 2014a.

ERP, C. In., 2014b.

FOUNDATION, E. Eclipse Process Framework Project. In., 2016.

GROUP, O. M. Visual Modelling: past, present and future. In., 2005, p. 6.

GUERRERO, R. M. Sobre PostgreSQL. In., 2010.

IASB. Normas Internacionales de Contabilidad. 2014, pp. 14.

JACOBSON, I., G. BOOCH AND J. RUMBAUG. Rational Unified Proces. 2007.

JARA, S. W. A. Activos Fijos. 2014, pp. 10.

LARMAN, C. *UML y Patrones*. Edtion ed. California: Prentice Hall, 2005. 520 p.

MIKE BEEDLE, A. V. B., ALISTAIR COCKBURN, MARTIN FOWLER, JIM HIGHSMITH, ANDREW HUNT, RON JEFFRIES, JON KERN, BRIAN MARICK, ROBERT C. MARTIN, KEN SCHWABER, JEFF SUTHERLAND, DAVE THOMAS Manifesto for Agile Software Development 2001 2001, 1.

NETBEANS. Welcome to the NetBeans. In., 2016, vol. 2016.

ORACLE. Getting Started with JavaFX. In., 2017, vol. 2017.

ORM, H. Hibernate ORM. Idiomatic persistence for Java and relational databases. In., 2016, vol. 2017.

P. CLEMENTS, L BASSAN AND R KAZZMAN *Software Architecture in Practice*. edited by A. WESLEY. Edtion ed. EEUU: Adison Wesley, 2003. 546 p.

PARADIGM, V. What is Visual Paradigm? In., 2016, p. 1.

PEÑA, J. M. F. Pruebas de Software. In. Ciudad México, 2006.

PGADMIN pgAdmin 27/10/2016 2016.

PRECIOS, M. D. F. Y. Resolución No.10. 2007, pp. 8.

PRESSMAN, R. S. *Ingeniería del Software*. Edtion ed., 2008. 765 p.

PRESSMAN, R. S. *Ingeniería del software Un enfoque práctico*. Edtion ed., 2010. 805 p. ISBN 978-607-15-0314-5.

QUEVEDO, V. D. CEDRUX. Solución para sistemas de control logístico. 15 Convención Científica de Ingeniería y Arquitectura. Instituto Superior Politécnico José Antonio Hecheverría, 24/07/2016 2016, 12.

REYES, E. A. Componente de Activos del Sistema de Gestión Universitaria de la Universidad de las Ciencias Informáticas. Serie Científica de la Universidad de las Ciencias Informáticas, 2016, 9, 16.

ROMÁN, A. Patrón DAO (Data Access Object). In., 2010.

ROMERO, H. Metodologías de Desarrollo. 2009, pp. 47. Available from Internet:<**Error! Hyperlink reference not valid.**.

SCHMULLER, J. Aprendiendo UML en 24 horas [online]. 2010.

SOMMERVILLE, I. *Ingeniería del Software*. Edtion ed. Madrid: Pearson Addison Wesley, 2005. 712 p. ISBN 84-7829-074-5.

TM, J. Conozca más sobre la tecnología Java. In., 2016, vol. 2016.

UP, O. What is OpenUP? . 2010, 2016]. Available from Internet:<http://epf.eclipse.org/wikis/openup/>.

VIRGEN DAMARIS QUEVEDO CAMPINS, M. M. C., LESTER ANTONIO GONZÁLEZ GUTIERREZ, ALIEN FERNÁNDEZ FUENTES. CEDRUX. Solución para sistemas de control logístico. 2010.

ZAYAS, A. *LAS METODOLOGÍAS DE LA INVESTIGACIÓN CIENTÍFICA*. Edtion ed.: Pueblo y Educación, 1997.