Facultad de Ciencias y Tecnologías Computacionales

Sistema informático para la auditoria de 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: Fernando Diosmedes Córdova Pons
Tutor(es): As., Ing. Yadira García García
As., Ing. Pedro Arango Astorga

La Habana, noviembre del 2022

"Año 64 de la Revolución"

PENSAMIENTO

“Quien no sea capaz de luchar por otros, no será nunca suficientemente capaz de luchar por sí mismo.”

Fidel Castro

DECLARACIÓN DE AUTORÍA

El autor del trabajo de diploma con título “Sistema informático para la auditoria de Activos Fijos Tangibles en la Facultad de Ciencias y Tecnologías Computacionales, concede a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declara como único autor de su contenido. Para que así conste firma la presente a los ___ días del mes de noviembre del año 2022.

Fernando Diosmedes Córdova Pons

_____

Firma del Autor

As., Ing. Yadira García García

_____

Firma del Tutor

As., Ing. Pedro Arango Astorga

_____

Firma del Tutor

DATOS DE CONTACTO

Yadira García García, se graduada de ingeniería en ciencias informática en la Universidad de las Ciencias Informáticas (UCI). Actualmente se desempeña como profesora del departamento de Informática de la Facultad 3.

Pedro Arango Astorga se graduado de ingeniería en ciencias informáticas en la Universidad de las Ciencias Informáticas (UCI). Se desempeña actualmente como Jefe de departamento de Tecnología y como profesor de departamento de Informática de la Facultad 3.

AGRADECIMIENTOS

Hoy se cumple un sueño que en su principio estaba lejos y lleno de dudas, pero el camino fue recorrido y satisfactoriamente puedo decir que a pesar de las caídas y tropezones llegué a mi meta: ser un profesional.

Agradezco a mi familia en especial a mis padres, Miriam y Fernando por el apoyo incondicional que siempre me brindaron, por siempre recordarme que es el empeño y la dedicación lo que conlleva al éxito, por estar presente en las buenas y en las malas, gracias padres por ser mi mayor fuente de inspiración y superación. A mi hermana Iriam por confiar en mí y ser la primera en demostrarme que los sueños se cumplen y que en el esfuerzo está la recompensa.

A mi novia Isbel, por brindarme todo su amor, tiempo, compresión y estar siempre presente, dándome toda la fuerza que necesitaba para poder terminar lo que ya había comenzado.

A mis tutores Yadira y Pedro, por su constante apoyo, paciencia, dedicación, colaboración y conocimiento en todo el recorrido de la tesis, les agradezco de corazón.

A Álvaro y a Cristian por su ayuda incondicional en el trascurso de la tesis.

A todos mis profesores por contribuir en mi formación como mejor ser humano y profesional.

A todos aquellos amigos de aquí y de allá que brindaron su mano para levantarme y poder seguir adelante. A todos los que en estos cinco años de carrera la vida puso en mi camino y me demostraron su buen corazón: Emilio, Sindy, Rey, Daniel A., Raúl, Nayalis, Maricarmen…

A todas estas lindas personas muchas gracias, agradecido para toda la vida.

DEDICATORIA

Esta tesis está dedicada a mi familia, en especial a mis padres amados, por todos sus sabios consejos y estar siempre para mí.

A mis abuelos, por siempre desear que me gradúe y confiar en mí.

A mi hermana porque no imagino que sería la vida sin ella.

A mi novia por acompañarme y darme ese gran apoyo en toda esta dura trayectoria.

RESUMEN

El control interno es el proceso integrado a las operaciones con un enfoque de mejoramiento continuo, extendido a todas las actividades inherentes a la gestión, efectuado por la dirección de una entidad y el resto del personal. Este representa el mecanismo fundamental para la preservación y control de los bienes y recursos de una organización. Los activos fijos tangibles son aquellos bienes materiales que una entidad utiliza continuamente en dichas operaciones, de forma permanente.

Actualmente en la Facultad de Ciencias y Tecnologías Computacionales (CITEC) el proceso de auditoría de los activos fijos tangibles se realiza de forma manual por parte del personal encargado en cada área, almacenando los datos en formato duro, lo que resulta muy engorroso a la hora de poder acceder a la información, ya sea para realizar una búsqueda, modificar algún dato, obtener reportes o realizar revisiones, afectando así la calidad y agilidad del proceso.

El presente trabajo de diploma tiene como objetivo desarrollar un sistema informático que permita una mejor gestión del proceso de auditoría en dicha facultad. La herramienta propuesta cuenta con una aplicación para dispositivos móviles con sistema operativo Android, una aplicación de escritorio multiplataforma y un servicio API-REST encargada de la sincronización entre ambas aplicaciones. La solución desarrollada contribuye a la gestión del proceso de auditoría garantizando la movilidad inherente al proceso, permitirá además la generación de informes resultantes de la autoría, así como los modelos de movimientos y las actas de responsabilidad material por cada área.

PALABRAS CLAVES

activos fijos tangibles, auditoría, dispositivos móviles, gestión, sistema informático

ABSTRACT

Internal control is the process integrated into operations with a focus on continuous improvement, extended to all activities inherent to management, demonstrated by the management of an entity and the rest of the staff. This represents the fundamental mechanism for the possession and control of the assets and resources of an organization. Tangible fixed assets are those material goods that an entity uses continuously in said operations, permanently.

Currently, at the Faculty of Computational Sciences and Technologies (CITEC), the audit process of tangible fixed assets is carried out manually by the personnel in charge of each area, storing the data in hard format, which is very cumbersome at the time to be able to access the information, either to carry out a search, modify some data, obtain reports or carry out revisions, thus lowering the quality and agility of the process.

The objective of this diploma work is to develop a computer system that allows a better management of the audit process in said faculty. The proposed tool has an application for mobile devices with the Android operating system, a multiplatform desktop application, and an API-REST service that traces between both applications. The solution developed will contribute to the audit process, guaranteeing the mobility inherent to the process, it will also allow the generation of the reports resulting from the authorship, as well as the models of movements and the acts of material responsibility for each area.

KEYWORDS

Tangible fixed assets, audit, mobile devices, management, computer system.

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

OPINIÓN DEL(OS) TUTOR(ES)

<Contenido de la opinión de los tutores>

AVAL DEL CLIENTE

<Contenido del aval del cliente sobre la solución desarrollada>

INTRODUCCIÓN

La Resolución Económica del V Congreso del Partido Comunista de Cuba se consigna: "…En las nuevas condiciones en que opera la economía, con un mayor grado de descentralización y más vinculados a las exigencias de la competencia internacional, el control oportuno y eficaz de la actividad económica es esencial para la dirección a cualquier nivel…". Con el fin de regular las actividades llevadas a cabo en las organizaciones, así como el uso eficiente y eficaz de los recursos se deben implementar pautas que garanticen un control que sea interno al mecanismo de gestión y no dependa únicamente de comprobaciones externas. A finales del siglo XIX, como consecuencia del importante crecimiento operado dentro de las empresas, se demostró la importancia de implantar sistemas de control interno (Plasencia Asorey 2010).

El control interno es el proceso integrado a las operaciones 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 (Contraloría General de la República 2021).

El sistema de control interno constituye para toda institución garantía de apego a las normas legales respecto a las actividades relevantes de la entidad, representando el mecanismo fundamental de proteger los recursos o bienes que emplea de manera continua en el curso de sus operaciones (Hernández González et al. 2019). En este sentido la preservación y control de los bienes y recursos se vuelve tarea primordial en aras de lograr indicadores de eficacia y productividad. Los Activos Fijos Tangibles (en lo adelante AFT) o Medios Básicos (MB), no son más que los bienes materiales que una entidad utiliza continuamente en sus operaciones, de forma permanente, los cuales sufren una depreciación durante su vida útil (Dueñas, Pina y Maldonado 2021).

La Universidad de las Ciencias Informáticas (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. Como parte del proceso de control interno en la institución se establece la verificación de al menos el 10 % de los AFT mensualmente, aunque de igual forma cada área puede realizar un control y seguimiento de sus medios siempre que sea necesario. Durante dicho proceso se comparan las existencias físicas con los registros contables para verificar su coincidencia y prevenir riesgos de sustracción, despilfarro, uso indebido u otras irregularidades (Cornelio, Fonseca y Caballeros 2016).

En la Facultad de Ciencias y Tecnologías Computacionales (CITEC) perteneciente a la UCI esta actividad se incluye dentro de la gestión administrativa e involucra todas las áreas docentes, productivas y de servicio. Algunos inconvenientes hoy limitan la calidad y agilidad con la que se lleva a efecto el control de los AFT conforme lo previsto en el sistema de control interno. La facultad cuenta con más de 620 estaciones de trabajo destinadas a los procesos docente-productivos y una aproximación de 2500 AFT en total.

A través de entrevistas realizadas a especialistas y jefes de áreas de la facultad, y en correspondencia con lo planteado por disímiles autores (Cornelio, Fonseca y Caballeros 2016), (Puig Díaz y González Caballero 2019),(García Argüello y Ramírez Pérez 2020) se relacionan a continuación algunos problemas asociados con el control de los AFT en distintas instituciones, de la cual no queda excepta la Facultad CITEC, :

- Extravíos de AFT debido a un procedimiento equivocado en el movimiento de área.
- Inexistencia en algunas áreas de las actas de responsabilidad material actualizadas.
- Traslados de lugar sin movimientos registrados formalmente.
- Gastos significativos de recursos humanos y materiales teniendo en cuenta que la auditoría resulta un proceso dilatado y sujeto a errores.
- No se evidencia la utilización de etiquetas para los AFT con un código identificador (preferiblemente código de barras), que a través de un sistema informático con un software para la adquisición de datos realice las auditorías.
- Algunos de los sistemas existentes para el control de los AFT son aplicaciones de escritorio, lo que representa un desaprovechamiento en tiempo y recursos en despliegue al tener que ser instalado en diferentes computadoras, siendo este proceso muy dinámico y móvil entre las diferentes estructuras edificadas.

- Los sistemas existentes, no se adaptan a la dinámica del proceso de auditoría, pues estos deben utilizarse desde una computadora y esta tarea exige de una elevada movilidad.

Teniendo en cuenta las deficiencias anteriormente mencionadas, se define el siguiente **problema a resolver**: ¿cómo gestionar el proceso de auditoría de activos fijos tangibles en la Facultad CITEC?

Para dar solución al problema anteriormente planteado se define como **objetivo general**: desarrollar un sistema informático para la gestión del proceso de auditoría en la Facultad CITEC.

Se determina el siguiente **objeto de estudio**: sistemas informáticos para el proceso de auditoría de activos fijos tangibles.

Delimitando como **campo de acción**: sistemas informáticos para la gestión del proceso de auditoría de activos fijos tangibles en la facultad CITEC.

Para dar cumplimiento al objetivo general propuesto se definen los siguientes **objetivos específicos**:

- Elaborar el marco teórico de la investigación.
- Realizar el análisis y diseño del sistema informático para la gestión del proceso de auditoría de activos fijos tangibles.
- Realizar la implementación de las funcionalidades del sistema informático para la gestión del proceso de auditoría de activos fijos tangibles.
- Realizar la validación y verificación del sistema informático para la gestión del proceso de auditoría de activos fijos tangibles.

Como **idea a defender** de la investigación se plantea: con el desarrollo de un sistema informático se garantizará la gestión del proceso de auditoría en la facultad CITEC.

Con el propósito de dar cumplimiento a los objetivos planteados se definen las siguientes **tareas de investigación:**

1. Caracterizar el proceso de auditoría de los AFT en la Facultad CITEC.
2. Realizar la revisión bibliográfica sobre soluciones existentes para el proceso de auditoría de AFT.
3. Realizar un estudio de tecnologías para el desarrollo de aplicaciones móviles y servicio API-REST.
4. Seleccionar los lenguajes, metodologías, herramientas y tecnología para la implementación del sistema informático.
5. Realizar las historias de usuarios de la propuesta de solución.
6. Realizar el diseño de las tarjetas CRC (Clase-responsabilidad-Colaboración) de la propuesta de solución.
7. Realizar las tareas de ingeniería de la propuesta de solución.
8. Realizar el diseño de los casos de pruebas de aceptación de la propuesta de solución.
9. Implementar la propuesta de solución.
10. Realizar las pruebas correspondientes para la validación y verificación del sistema informático desarrollado.

Con el propósito de resolver el problema y lograr el objetivo planteado, se emplearon los siguientes **métodos de investigación:**

Métodos teóricos

- **Inductivo-deductivo:** se utilizó para el razonamiento de la información consultada y llegar así a la obtención de un grupo de conocimientos particulares y generales. Permitió determinar las características del proceso de auditoría, así como los sistemas existentes para ello.
- **Análisis histórico-lógico:** permitió el estudio crítico de soluciones anteriores referidas en la bibliografía científica, para extraer aspectos positivos de ellos, así como el uso de trabajos como punto de referencia para la definición de requisitos preliminares.
- **Analítico-sintético:** permitió, fundamentalmente, realizar el estudio teórico de la investigación, haciendo posible la selección de los elementos más importantes en el proceso de desarrollo del sistema informático.

- **Modelación:** permitió un mejor entendimiento del proceso a informatizar. Contribuyó junto a la metodología de desarrollo seleccionada, modelar los artefactos esenciales para el desarrollo del sistema informático

Métodos empíricos:

- **Entrevista:** se utilizó en el intercambio con el cliente para adquirir información sobre el negocio y los requisitos que se deben cumplir en el desarrollo del sistema. Se realizaron entrevistas a directivos del área de administración de la facultad CITEC para un mejor entendimiento del proceso de auditoría de AFT, así como poder identificar deficiencias que ocurren en el proceso.

El documento estará estructurado en tres capítulos organizados de la siguiente forma:

Capítulo 1. Fundamentación teórica-metodológica: Se definen los elementos teóricos necesarios para el desarrollo del sistema informático para la gestión del proceso de auditoría de AFT en la facultad CITEC, que se emplearán durante la investigación. Se realiza un análisis de las soluciones existentes a nivel nacional e internacional para la auditoría de AFT. Se selecciona la metodología de desarrollo de software, así como las herramientas y tecnologías que se deben utilizar para la implementación de la propuesta de solución. Además, se describen los patrones de diseño y arquitectónico a tener en cuenta para la implementación de las aplicaciones que componen la propuesta de solución.

Capítulo 2. Propuesta de solución: describe la propuesta de solución, junto a la modelación del proceso de auditoría de activos fijos tangibles que se lleva a cabo en la facultad CITEC. Se evidencia la especificación de las funcionalidades a implementar mediante las historias de usuario. Se muestra la arquitectura, las tarjetas CRC y el modelo de datos, así como el uso de patrones y estándares de codificación.

Capítulo 3. Validación de la solución propuesta: se define la estrategia de pruebas a partir de la metodología de desarrollo, con el objetivo de garantizar la validación de la solución propuesta. Se analizan los niveles de pruebas ejecutados y los resultados alcanzados.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA-METODOLÓGICA

El presente capítulo recoge los elementos teóricos que fundamentan el desarrollo de un sistema informático para la gestión del proceso de auditoría de la Facultad CITEC. Se describe el proceso de auditoría actualmente en la facultad CITEC. Seguidamente, se reflejan las características esenciales de las diferentes soluciones informáticas existentes en el mercado. Finalmente, se exponen los elementos más relevantes de la metodología, lenguajes y herramientas definidas para el desarrollo; reflejándose el uso de patrones y pruebas para garantizar la efectividad del proceso de desarrollo y calidad del producto final.

1.1 Conceptos fundamentales

Activos Fijos

Corresponde a las inversiones de capital permanente necesarios para el desarrollo habitual de las empresas, por ejemplo: propiedades, plantas, terrenos, maquinarias, mobiliarios, equipo de transporte, etc. Se incluye la compra de activo fijo nuevo o usado, construido por cuenta propia, ventas, retiros y traslados, los mismos pueden ser activos fijos tangibles e intangibles. 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 no pueden convertirse en líquido 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 (Pérez Porto, J., Merino, M. 2022) .

Activos fijos tangibles

Representan propiedades físicamente tangibles que han de utilizarse en las operaciones regulares de la empresa, por ejemplo: terrenos, edificios, maquinarias y equipos, vehículos, y mobiliarios, entre otros (Ignacio David Araya Gil 2019)

Tangibles, son además las inversiones de capital permanente necesarios para el desarrollo habitual de las empresas, descontado la reserva para depreciación, por ejemplo: propiedades,

plantas, terrenos, maquinarias, mobiliarios, equipos de transporte(Ignacio David Araya Gil 2019).

Auditoría

El proceso de auditoría, en la teoría y la práctica son actividades administrativas de revisión de documentos y de las tareas ejecutadas dentro del sistema productivo de la organización, para la verificación de cumplimiento de la normativa y los procedimientos internos dentro de sus rutinas diarias. En la mayoría de los casos, las empresas pequeñas y medianas no cuentan con un documento de instrucciones que es fundamental para el proceso de auditoría. En este sentido, el auditor debe aplicar ciertas técnicas, como entrevistas con cada uno de los jefes departamentales, observación directa de las actividades y los procesos de ejecución y, por último, la revisión de todos los registros disponibles para garantizar el cumplimiento de los sistemas de calidad, legislación gubernamental y fiscal, así como de las normas de seguridad internas (Enríquez y Nogueira 2021).

Aplicación Informática

Aplicación informática es un programa informático diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de tareas. Esto lo diferencia principalmente de otros tipos de programas, como los sistemas operativos (que hacen funcionar la computadora), las utilidades (que realizan tareas de mantenimiento o de uso general), y las herramientas de desarrollo de software (para crear programas informáticos). Suele resultar una solución informática para la automatización de ciertas tareas complicadas, como pueden ser la contabilidad, la redacción de documentos, o la gestión de un almacén. Algunos ejemplos de programas de aplicación son los procesadores de textos, hojas de cálculo, y base de datos (Chirimelli 2017).

Ciertas aplicaciones desarrolladas a medida suelen ofrecer una gran potencia ya que están exclusivamente diseñadas para resolver un problema específico. Otros, llamados paquetes integrados de software, ofrecen menos potencia pero a cambio incluyen varias aplicaciones, como un programa procesador de textos, de hoja de cálculo y de base de datos (Chirimelli 2017).

Según (Roger S. Pressman 2015) 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.

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

- 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 Atendiendo esta categorización la propuesta de solución se propone enmarcarla en la categoría “Sistema Informático”, decisión respaldada por las características del negocio, ampliamente descritas en el capítulo uno del presente documento.

Sistema Informático

Por sistema informático (SI) se entiende un sistema automatizado de almacenamiento, procesamiento y recuperación de datos, que aprovecha las herramientas de la computación y la electrónica para llevar a cabo su serie compleja de procesos y operaciones. En otras palabras, un sistema informático es un computador de alguna índole. Los sistemas informáticos son tipos de sistemas de información, o sea, sistemas que se organizan en torno al manejo de datos de diversa naturaleza, aunque no todos los sistemas de información sean informáticos. Esto es, no todos son digitales, ni automatizados, ni electrónicos (Sistema Informático 2022).

Los sistemas informáticos ocupan en el mundo contemporáneo un lugar clave para la organización humana de sus procesos productivos y de otras naturalezas. Es una herramienta poderosa para el intercambio de información y la construcción de redes informáticas que superan la dificultad de las distancias (Sistema Informático 2022).

1.2 Proceso de auditoría en la Facultad de Ciencias y Tecnologías Computacionales

El proceso de auditoría 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, las cuales corresponden comúnmente con facultades, direcciones generales y centros de desarrollo independientes. En cada área el responsable de la misma o auditor verifica 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 universidad. Las actividades relacionadas con la auditoría de los activos fijos tangibles son asistidas por el Sistema de Gestión Universitaria, el cual provee del listado de los medios a los responsables autorizados en cada una de las áreas.

La Facultad CITEC cuenta aproximadamente con 620 estaciones de trabajo dedicadas a los procesos docentes productivos, desplegadas en toda la estructura de la propia facultad, las cuales contienen inmobiliarios y recursos que aproximan un total de 2500 AFT. El proceso de auditoría es rectorado por el Vicedecanato de Economía y Administración (en lo adelante VEA), 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.

Comúnmente el proceso se realiza a partir de la indicación del VEA, orientado por la Dirección de Contabilidad y Finanzas, de iniciar el proceso de auditoría mensual. Cada responsable de área obtiene del Sistema de Gestión Universitaria un listado impreso correspondiente al inventario con los rótulos de los medios básicos y el estado físico en el que se encontraban en la última inspección. Luego el auditor debe 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 la existencia o no de faltantes o

sobrantes y el estado general de los medios. Este informe debe ser archivado y constar como evidencia de la auditoría llevada a cabo sobre los recursos asignados.

1.3 Análisis de soluciones existentes

Versat-Sarasola

Es el primer sistema integral de gestión de contabilidad certificado por los ministerios de Finanzas y Precios y de Informática y las Comunicaciones, desarrollado para la gestión económica eficaz y fiable. Actualmente es utilizado en Cuba en alrededor de 200 entidades de varias provincias. Este sistema integrado cuenta con un conjunto de 12 módulos entre los que se encuentran: configuración y seguridad, contabilidad general y de gastos, costos y procesos, análisis económico empresarial, control de activos fijos, finanzas y caja, planificación y presupuestos, control de inventarios, pago de salario, paquete de gestión, contratación, facturación. Tiene como desventaja que los datos introducidos en el sistema pueden ser modificados o eliminados. A pesar de las funcionalidades de la aplicación se desestima su empleo como solución a la problemática planteada ya que la misma es privativa y no permite acceso a su código fuente, además no proporciona recursos que permitan dinamizar el proceso de auditoría de medios contra listado de activos (Díaz y Caballero 2019).

Assets

El sistema de gestión integral (ASSETS) es un sistema multiusuario empleado en la Universidad de las Ciencias Informáticas, el cual se encuentra dividido en módulos económicos que trabajan en conjunto para el control de las actividades económica, financiera y contable sobre los medios materiales y financieros. Permiten registrar los activos en uso y obtener información acerca de cada uno de ellos. Presenta inconsistencia en la integridad de los datos, imposibilita ver reportes por diferentes conceptos, no es concurrente e inconvenientemente solo pueden interactuar con la aplicación un grupo muy pequeño de usuarios, donde se tiene que pagar una licencia de código por cada uno de ellos, impidiendo a los usuarios responsables de los AFT de las diferentes áreas de responsabilidad tener acceso al mismo. Al ser un sistema propietario bajo licencia no garantiza la soberanía tecnológica para Cuba pues recibe soporte de

una compañía propietaria que impide la colaboración de nuestro país con nuevas funcionalidades (Gómez y Cabaleiro 2021). De igual forma no se tiene en cuenta el uso del Assets ya que a pesar de contar con varios módulos bien definidos asociados al inventario de activos fijos tangibles, el mismo no permite realizar auditorías.

ConDor

El sistema contable ConDor cuenta con una suite integrada por siete módulos: Contabilidad General, Activos Fijos, Inventario, Nómina/prenómina, Disponibilidad Financiera, Condexce y Efectos. Actualmente este software se encuentra implementado para el sistema operativo Windows en cualquiera de sus versiones. El sistema presenta como característica que cada módulo tiene una base de datos diferente, lo que trae redundancia en la información. Cuenta con el módulo "Control de Activos Fijos", en el cual se automatiza los procedimientos establecidos para el control interno y la contabilización de los activos fijos tangibles e intangibles; de acuerdo a las normativas dictadas por el Ministerio de Finanzas y Precios, (MFP), y a las disposiciones emitidas por la Dirección de Contabilidad y Precios. Este sistema presenta como característica que está implementado en un entorno monousuario, significando esto una significativa desventaja. Además, cada uno de sus módulos tienes bases de datos diferentes, trayendo consigo redundancia en la información (Manuel Contreras 2015).

ContarERP

El sistema ContarERP es una 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; realiza la depreciación de cada activo de manera automática. Entre sus principales características destaca:

- 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 tanto 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 (ContarERP Software 2022).

Esta solución no se considera adecuada para su empleo en el entorno de la problemática ya que su implantación supone una limitante, pues al ser un sistema privativo no se cuenta con acceso a su código fuente y no permite la integración con otros sistemas y aplicaciones. Por otra parte, dirige su funcionamiento a las normativas colombianas del control de activos y su implantación además sería contraria a las políticas de soberanía tecnológica impulsadas por el país (ContarERP Software 2022).

Valoración sobre el estudio de soluciones existentes

Luego de realizado el análisis de las soluciones existentes descritas anteriormente, se puede plantear que en su generalidad no satisfacen completamente la necesidad que se plantea en la problemática. El Sistema de Gestión Integral Assets empleado en la universidad contiene varios módulos bien específicos, pero no es adaptable a la necesidad presente en la facultad CITEC, teniendo en cuenta que no permite la realización de auditorías. Si bien estas aplicaciones realizan la gestión de los AFT de manera eficiente, la forma en que los controla no es la más adaptable para el responsable de realizar las auditorías. Tanto en unos como en otros hay que verificar la existencia de los AFT manualmente, revisando el rótulo de cada medio, contra el listado del área al que corresponde. Estos sistemas no se adaptan a la dinámica del proceso de auditoría, pues deben utilizarse desde una computadora y esta tarea exige de una elevada movilidad.

Teniendo en cuenta los elementos anteriormente planteados se propone el desarrollo de un sistema informático que contribuya a la gestión del proceso de auditoría de activos fijos tangibles en la Facultad de Ciencias y Tecnologías Computacionales.

1.4 Metodología de desarrollo de software

En la actualidad la rapidez y el dinamismo en la industria del software han hecho replantear los cimientos sobre los que se sustenta el desarrollo de software tradicional. Actualmente existen una gran cantidad de metodologías para el desarrollo de software, separadas en dos grandes grupos; las metodologías tradicionales o pesadas y las metodologías agiles. Las metodologías tradicionales se basan en las buenas prácticas dentro de la ingeniería del software, siguiendo un marco de disciplina estricto y un riguroso proceso de aplicación. Las metodologías ágiles, en cambio, representan una solución a los problemas que requieren una respuesta rápida en un ambiente flexible y con cambios constantes, haciendo un uso menos estricto de documentación y métodos formales (Mory 2021).

1.4.1 Metodologías tradicionales

Las metodologías tradicionales son denominadas, a veces, de forma despectiva, como metodologías pesadas. Centran su atención en llevar una documentación exhaustiva de todo el proyecto, la planificación y control del mismo, en especificaciones precisas de requisitos y modelado y en cumplir con un plan de trabajo, definido todo esto, en la fase inicial del desarrollo del proyecto. Hacen una definición minuciosa de los procesos y tareas que se realizarán durante el ciclo de vida del software, generando una extensa documentación asociada a cada elemento del proceso de desarrollo Estas metodologías tradicionales imponen una disciplina rigurosa de trabajo sobre el proceso de desarrollo del software, con el fin de conseguir un software más eficiente. Para ello, se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto software (Mory 2021).

Entre las metodologías tradicionales más popular y usada está el Proceso Unificado de Desarrollo (Rational Unified Process, RUP):

RUP consiste en una estructura de trabajo de proceso con el objetivo en el producto y haciendo uso del Lenguaje Unificado de Modelado (Unified Modeling Language, UML), cuando se habla de programación orientada a objetos. El UML compone un lenguaje para definir una secuencia de artefactos y ayudar en la ejecución de las tareas del sistema a desarrollar, a través de diferentes tipos de diagramas. Todas las técnicas y prácticas utilizadas en el modelo RUP están probadas en la industria del software y la gestión de proyectos. Aunque RUP se utiliza para proyectos complejos y con equipos extensos, permite realizar actividades y artefactos de acuerdo con la elección del equipo y se puede adaptar para agilizar el proceso. Es una metodología dirigida por los casos de uso, centrada en la arquitectura y además iterativa e incremental (Celio Gil Aros 2020) .

1.4.2 Metodologías ágiles

Las metodologías ágiles dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque está mostrando su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo, pero manteniendo una alta calidad. Proporcionan una serie de pautas y principios junto a técnicas pragmáticas que hacen que la entrega del proyecto sea menos complicada y más satisfactoria tanto para los clientes como para los equipos de trabajo, evitando de esta manera los caminos burocráticos de las metodologías tradicionales, generando poca documentación y no haciendo uso de métodos formales (Mory 2021).

Algunas de las metodologías ágiles son:

- Extreme Programming XP.
- SCRUM.
- OpenUp

eXtreme Programming

La programación extrema o eXtreme Programming (XP) es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck, autor del primer libro sobre la materia, Extreme Programming Explained: Embrace Change (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales

principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad.

Características:

- Se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad.
- Se aplica de manera dinámica durante el ciclo de vida del software.
- Es capaz de adaptarse a los cambios de requisitos.
- Los individuos e interacciones son más importantes que los procesos y herramientas.

OpenUP

OpenUP es una metodología ágil de desarrollo de software, basada en RUP (Rational Unified Process), que contiene el conjunto mínimo de prácticas que ayudan a un equipo de desarrollo de software a realizar un producto de alta calidad, de una forma eficiente. Esta metodología fue propuesta por el grupo de empresas conformado por: IBM Corp, Telelogic AB, Armstrong Process Group Inc., Number Six Software Inc. y Xansa; quienes la donaron a la Fundación Eclipse en el año 2007, que la ha publicado bajo licencia libre. OpenUP, es un proceso unificado, iterativo e incremental, que se centra en el desarrollo colaborativo de software para generar sistemas de calidad (Roger S. Pressman 2015).

Scrum

Scrum es una metodología ágil para el desarrollo y el mantenimiento de productos complejos. Ken Schwaber y Jeff Sutherland desarrollaron Scrum; desde entonces ha sido usado para gestionar el desarrollo de productos complejos desde principios de los años 90. Scrum no es un proceso o una técnica para construir productos; en lugar de eso, es una metodología dentro de la cual se pueden emplear varias técnicas y procesos. Muestra la eficacia relativa de las prácticas de gestión de producto y las prácticas de desarrollo (Mory 2021).

Luego de realizado un análisis de las metodologías ágiles XP, SCRUM y OpenUp, y la metodología tradicional RUP, teniendo en cuenta la respuesta a los cambios, el trabajo con el cliente, los planes de entrega de las iteraciones, el trabajo en equipo y el enfoque, se selecciona **XP** como metodología para guiar el proceso de desarrollo del sistema informático.

En XP todos los requisitos son expresados como escenarios (historias de usuarios), los cuales son implementados directamente como una serie de tareas. Esta metodología resulta apropiada para proyectos con requisitos imprecisos y cambiantes, en los que existe un alto riesgo técnico. Los programadores trabajan en parejas y se desarrollan pruebas para cada una de las tareas. Todas las pruebas deben ser ejecutadas satisfactoriamente cuando el nuevo código es integrado al sistema(Sommerville, Ian 2011). La evaluación de las metodologías y la necesidad de una variante flexible a los cambios, que genere los artefactos mínimos para la comunicación con el cliente, permitió identificar a la Programación Extrema (XP) como la alternativa más acertada.

1.5 Lenguajes y herramientas de desarrollo

En el presente epígrafe se abordan los elementos fundamentales asociados a los lenguajes y herramientas seleccionados para el desarrollo del sistema informático.

1.5.1 Lenguaje Unificado de Modelado

El **Lenguaje Unificado de Modelado** (en inglés Unified Model Language, UML), es un lenguaje de modelado visual usado para especificar, visualizar, construir y documentar artefactos de un software. Permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma fácil de comprender y comunicar a otras personas.(SCHMULLER 2011) . Un lenguaje de modelado provee un vocabulario y conjunto de reglas centradas en la representación conceptual y física de un sistema (SCHMULLER 2011). La elección de UML se sustenta en su facilidad para especificar, documentar, visualizar y comprender los artefactos generados en las diferentes fases del desarrollo de software. Así como sus ventajas para la comunicación entre el cliente y el equipo de desarrollo.

1.5.2 Herramienta CASE de modelado

Para (Sommerville, Ian 2011) CASE (Computer Aided Software Engineering) es el nombre con el que se identifica la herramienta utilizada para apoyar las actividades del proceso de software. Según Kendall y Kendall estas herramientas facilitan la interacción entre los miembros de un equipo al hacer que la diagramación sea un proceso iterativo y dinámico. Los analistas de sistemas se apoyan en estas herramientas desde el principio hasta el fin del ciclo

de vida, para incrementar la productividad, comunicarse de manera eficiente con los usuarios y desarrolladores, e integrar el trabajo que desempeñan en el sistema.

Para el modelado se utilizará **Visual Paradigm For UML 8.0** por ser una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Agiliza la construcción de aplicaciones con calidad y a un menor coste. Posibilita la generación de bases de datos, transformación de diagramas de Entidad-Relación en tablas de base de datos, así como ingeniería inversa de bases de datos (Baquero Hernández et al. 2016).

1.5.3 Lenguajes de programación

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; permitiendo al desarrollador comunicarse con los dispositivos de hardware y software existentes(android 2022).

Java

Java es un lenguaje seguro y completo que satisface en gran medida las necesidades de los dispositivos móviles tanto en software como en hardware. Las aplicaciones que se desarrollan deben tener un alto nivel de seguridad y Java posee esta capacidad que le permite, a las aplicaciones estar protegidas contra hackers. Los códigos de los programas orientados a objetos deben tener facilidad en el desarrollo y la posibilidad de reutilizarlos, características que Java para Android cumple muy bien. Java para Android puede lograr de trabajar con extensiones dinámicas para descargar códigos y añadirlos a nuevas aplicaciones, así como la seguridad durante el acceso a la red, le regalan a Java características que lo ubican entre los favoritos por los programadores. El hardware de cada dispositivo móvil tiene ciertas características que por mucho que tengan similitudes siempre es necesario llenar ciertas necesidades que los diferencian, el reto es encontrar un conjunto de bibliotecas donde se puedan desarrollar programas lo independientemente posible del soporte físico (Java 2021).

JavaFX

Es una plataforma que permite a los desarrolladores crear e implementar fácilmente aplicaciones de Internet enriquecidas que se comportan de la misma forma en distintas plataformas. Amplía la potencia de Java permitiendo a los desarrolladores utilizar cualquier biblioteca de Java en aplicaciones JavaFX. Los desarrolladores pueden ampliar sus capacidades en Java y utilizar la tecnología de presentación que FX proporciona para crear experiencias visuales que resulten atractivas. Es una tecnología de software que, combinada con Java, permite crear y desplegar aplicaciones con un aspecto vanguardista y contenidos avanzados, audio y vídeo(Tokio School 2021).

Node.js

Es un entorno en tiempo de ejecución multiplataforma para la capa del servidor (en el lado del servidor) basado en JavaScript. Es un entorno controlado por eventos diseñado para crear aplicaciones escalables, permitiéndote establecer y gestionar múltiples conexiones al mismo tiempo. Gracias a esta característica, no tienes que preocuparte con el bloqueo de procesos, pues no hay bloqueos. Node.js presenta un bucle de eventos como una construcción en tiempo de ejecución en lugar de una biblioteca. Este bucle de eventos es invisible para el usuario.
Otra característica especial de Node.js es que está diseñado para simplificar la comunicación. No tiene subprocesos, pero te permite aprovechar múltiples núcleos en su entorno y compartir sockets entre procesos(Node.js 2021).

1.5.4 Entorno de Desarrollo Integrado (IDE)

Un IDE (Integrated Development Environment) es un programa compuesto por un conjunto de herramientas de programación. Puede dedicarse a un solo lenguaje de programación o a varios. Consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (veracode 2020).

Android Studio

Es el entorno de desarrollo integrado oficial para la plataforma Android. Es un sistema de compilación flexible basado en Gradle. Un emulador rápido y cargado de funciones. Un entorno unificado donde puedes desarrollar para todos los dispositivos Android. Aplicación de cambios

para insertar cambios de código y recursos a la App en ejecución sin reiniciarla. Admite los mismos lenguajes de programación de IntelliJ y CLion, como Java y C++ y más con extensiones, como Go; y Android Studio 3.0 o posterior es compatible con Kotlin y todas las características de lenguaje Java 7 y un subconjunto de características de lenguaje Java 8 que varían según la versión de la plataforma (android 2022).

Visual Studio Code (desarrollo de API-REST)

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sin número de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación.

Según una encuesta realizada por Stack Overflow a más de 80,000 desarrolladores en mayo del 2021, Visual Studio Code es el entorno de desarrollo más usado y con mucha diferencia, un 71.06%. Incluye una terminal con todas las funciones, la cual se inicia fácilmente en el directorio de trabajo. La terminal integrada puede utilizar cualquier Shell instalado en el equipo, como PowerShell, Bash o cualquier otro. Contar con una terminal en el propio editor es de gran utilidad para ejecutar diferentes comandos necesarios cuando se está desarrollando (VStudioCode 2021).

Apache NetBeans (desarrollo aplicación de escritorio)

Apache NetBeans es un IDE (Entorno de Desarrollo Integrado, por sus siglas en inglés) de código abierto que proporciona editores de código, asistentes y plantillas. Puede ayudar a los desarrolladores a crear aplicaciones en Java, PHP y otros lenguajes de programación. Apache NetBeans se puede instalar en sistemas operativos Windows, Mac OSX, Linux y BSD.

La herramienta Java Editor del IDE proporciona asistencia de código inteligente, opciones de personalización y capacidades de navegación. Para maximizar la productividad, los desarrolladores pueden utilizar una amplia gama de herramientas y características útiles, como

la finalización de código inteligente, plantillas de código, color y resaltado semánticos y la gestión de importación (apache 2021). Para el desarrollo de la aplicación de escritorio se hizo uso de este IDE en su versión 11.0

Funcionalidades totales de Apache NetBeans en su versión 11

- Depuración.
- Desarrollo de aplicaciones web.
- Desarrollo de aplicaciones web/móviles.
- Desarrollo de código.
- Desarrollo de sitios web.
- Desarrollo de software.
- Desarrollo móvil.
- Edición de código.

1.5.5 Sistema Gestor de Bases de Datos (SGBD)

Las valoraciones de varios autores apuntan que un SGBD (Data Base Management System, DBMS) permite a los usuarios procesar, describir, administrar y recuperar de forma rápida y eficiente los datos almacenados en una base de datos; garantizando la seguridad de los mismos mediante el acceso limitando al personal autorizado. Su objetivo fundamental es suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos; de forma que no sea necesario conocer cómo están estos almacenados en la computadora, o el método de acceso empleado (Ayuware S.L 2021).

Dentro del sistema de gestor de base de datos se utilizó **SQLite** que es un motor de base de datos SQL transaccional de código abierto, ligero, autónomo, de configuración simple y sin servidor, que se caracteriza por almacenar información persistente de forma sencilla, SQLite gracias a sus características se diferencia de otros gestores de bases de datos, proporcionando grandes ventajas, una de ellas es que guarda la base de datos en un solo archivo. Como desventaja es más simple y no admite un gran volumen de información. Es más restringido con respecto a los formatos de archivos aceptados (OpenWebinars S.L 2022).

1.6 Patrones para el desarrollo de software

Un patrón es una descripción de un problema y su solución, que recibe un nombre y que puede emplearse en otros contextos. Muchos patrones ofrecen orientación sobre cómo asignar las responsabilidades a los objetos ante determinada categoría de problemas. Para la arquitectura del sistema se utilizarán 3 patrones arquitectónicos y un grupo de patrones de diseño. Los patrones arquitectónicos son:

- Arquitectura de microservicios para el Api-Rest
- Arquitectura Modelo Vista Controlador para la aplicación de escritorio
- Arquitecturas móvil para la aplicación móvil del sistema

1.5.6 Estrategia de pruebas

La gestión de la calidad es una actividad que se aplica a lo largo del proceso de software. El cumplimiento de la especificación del programa y la satisfacción de las expectativas del cliente deben estar en la mira del equipo de desarrollo cada momento, a lo que usualmente suele llamarse verificación y validación(Roger S. Pressman 2015). Las pruebas de software constituyen el último bastión en el aseguramiento de la calidad, son un conjunto de actividades planeadas con anticipación y realizadas de forma sistemática, con el fin de descubrir resultados distintos al esperado; de ahí la necesidad de contar con una estrategia de pruebas desde los primeros momentos del proceso de software(Sommerville, Ian 2011).

Reconocidos autores como en Pressman y Sommerville tienden a coincidir en la definición de los siguientes niveles de pruebas, adaptables a las condiciones del desarrollo en cuestión:

- **Pruebas unitarias:** se concentran en el diseño y comportamiento de cada componente del software, desde la perspectiva de su implementación.
- **Pruebas de integración:** prueban la correcta relación entre los componentes del software.
- **Pruebas de aceptación:** evalúan el cumplimiento de los requisitos pactados con el cliente.

- **Pruebas de sistema:** se ejecutan en un ambiente similar al ambiente operacional real, probando el software como un todo de conjunto con el resto de los elementos del sistema.

En la metodología XP las pruebas son tan importantes como la programación, fortalecen la confianza del cliente en el equipo de desarrollo y dentro de este. Los máximos exponentes de esta metodología, proponen dos principios para aumentar la rentabilidad de las pruebas: el chequeo doble y el incremento del costo de los defectos. El chequeo doble consiste en la aplicación de dos conjuntos de pruebas: uno escrito desde la perspectiva de los programadores y orientado a probar exhaustivamente cada uno de los componentes del sistema, y el otro escrito desde la perspectiva de los clientes o usuarios finales, para probar el funcionamiento del sistema como un todo. Lo cual es coherente con los niveles de pruebas de unidad y aceptación respectivamente(Roger S. Pressman 2015).

1.7 Conclusiones del capítulo

El estudio de la bibliografía y análisis de las soluciones informáticas existentes para el proceso de auditoría de AFT, suscitó una elevada comprensión del estado del arte en el área del conocimiento en cuestión, aportando los elementos teóricos que fundamentan la investigación. Por lo que al finalizar el presente capítulo se arribó a las siguientes conclusiones parciales:

- Se enunciaron los principales términos asociados al dominio de la investigación, lo que permitió adquirir un mayor conocimiento sobre el objeto de estudio de la investigación.
- Las soluciones existentes, no se adaptan a la dinámica del proceso de auditoría, teniendo en cuenta los medios que se emplean y la elevada movilidad que exige la tarea al auditar cada una de las áreas de responsabilidad material en la facultad CITEC.
- En la práctica en las auditorías realizadas se han detectado traslados de AFT que no han sido registrados sus movimientos de área.
- Inexistencia del uso de identificadores para cada AFT, como puede ser el código de barra, que en conjunto con un software facilite el proceso de auditoría en la facultad CITEC.
- La metodología de software que guía el proceso de desarrollo es XP, dada la interacción con el cliente y el poco tiempo de duración.

- Realizada una revisión de los principales elementos teóricos para el desarrollo del trabajo, se decide desarrollar un sistema informático que constará de una aplicación desarrollada para dispositivos móviles con sistema operativo Android y una aplicación de escritorio multiplataforma que permitirá importar los inventarios proporcionados por la Dirección de Contabilidad y Finanzas.
- Para la concepción del nuevo sistema se selecciona la metodología XP, que guiará el proceso de desarrollo, el lenguaje de programación (*lenguaje de programación a usar*) haciendo uso del IDE Android Studio y (*nombre del SGBD a usar*) como gestor de base de datos.

CAPÍTULO 2: PROPUESTA DE SOLUCIÓN

El presente capítulo recoge las principales características de la propuesta de solución. Se analizan las técnicas de captura de requisitos, identificando las funcionalidades del sistema a desarrollar, así como las características del mismo mediante los requisitos no funcionales, se muestra el resultado de la planificación, se presenta el plan de iteraciones para el desarrollo del sistema y las historias de usuarios. Se describe la arquitectura, los modelos de diseño y de datos y los patrones aplicados como parte del diseño. Finalmente, en la fase de desarrollo se presentan las tareas de ingeniería por funcionalidad a implementar y los estándares de codificación tomados en consideración.

2.1 Propuesta de solución

El sistema a desarrollar como propuesta de solución al problema de investigación constituye una herramienta fundamental para el proceso de auditoría de los AFT en la facultad CITEC. Esta propuesta contará con una aplicación desarrollada para dispositivos móviles con sistema operativo Android garantizando la movilidad inherente al proceso de auditoría. El uso que se propone de la cámara incorporada en estos dispositivos permitirá realizar la lectura de los códigos de barra que identificarán a los AFT que se desean auditar, así como el código QR de cada área de responsabilidad material. Al mismo tiempo se contará con una aplicación de escritorio multiplataforma que permitirá importar los inventarios proporcionados por la Dirección de Contabilidad y Finanzas y centralizar las auditorías realizadas por auditores diferentes en áreas de responsabilidad diversas.

La Figura 1 representa la propuesta de solución anteriormente descrita.

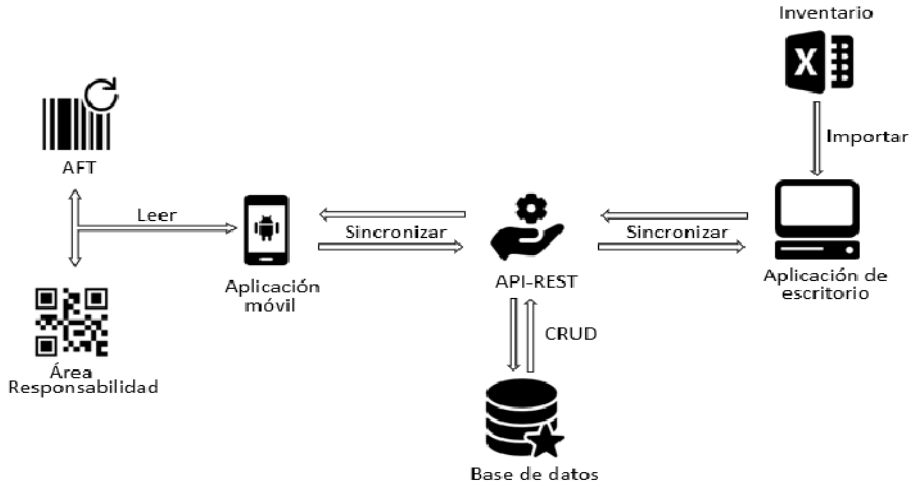


Figura 1. Propuesta de solución

Una vez finalizado el proceso de centralización de la información se podrá generar un informe resultante del proceso de auditoría, así como los modelos de movimiento y las actas de responsabilidad material por cada área. Para la comunicación entre las aplicaciones descritas con anterioridad (aplicación para móvil y aplicación de escritorio), se implementará un servicio API-REST que tiene como objetivo ejecutar el proceso de sincronización entre ambas aplicaciones, además de la inserción, modificación y consulta de la información en una base de datos relacional encargada de la persistencia de los datos.

2.2 Modelo de negocio

El proceso de auditoría de activos fijos tangibles (AFT) es solicitado por la Dirección de Contabilidad y Finanzas; para la realización del mismo es proporcionado el inventario, en el cual se encuentran el código, la descripción y el área de responsabilidad de cada uno de los AFT. El proceso de auditoría es llevado a cabo por un auditor, el cual comprueba la existencia física de los AFT de forma manual, realizando una comparación entre el inventario proporcionado y el rótulo escrito en cada medio, verificando, además el estado y características del AFT.

A continuación, haciendo uso de la notación BPMN, se muestra el diagrama de procesos que representa la descripción dada anteriormente.

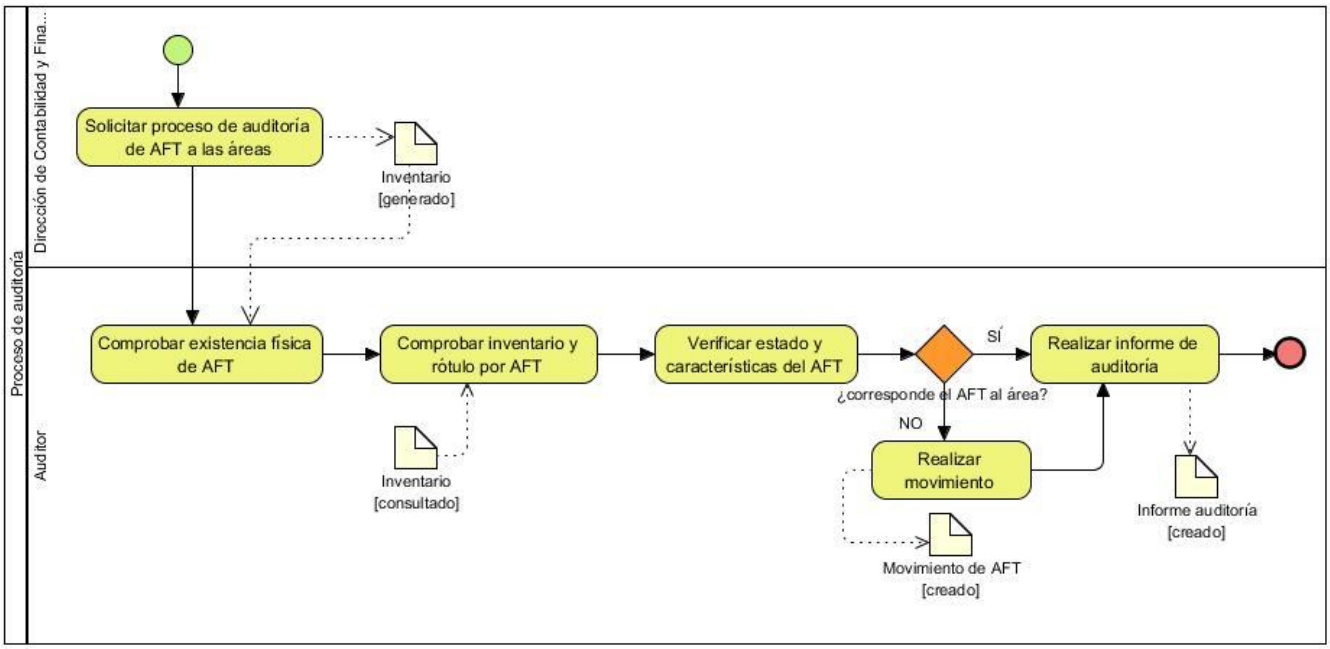


Figura 2. Proceso de auditoría

2.3 Modelo de dominio

Un modelo del dominio muestra clases conceptuales significativas en un dominio del problema; es el artefacto más importante que se crea durante el análisis. También se les denomina modelos conceptuales, modelo de objetos del dominio y modelos de objetos de análisis. Utilizando la notación UML, un modelo del dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación (Craig Larman 1999). Pueden mostrar:

- Objetos del dominio o clases conceptuales
- Asociaciones entre las clases conceptuales
- Atributos de las clases conceptuales

En la siguiente figura se representa los conceptos relevantes y sus relaciones en la descripción de la solución que se propone.

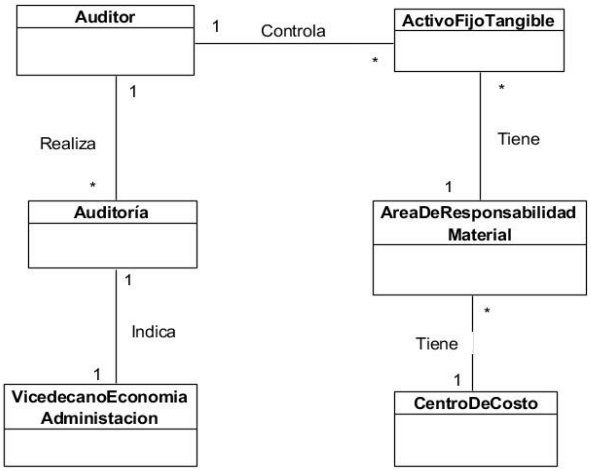


Figura 3. Modelo de dominio

2.4 Ingeniería de requisitos

La ingeniería de requisitos constituye un elemento fundamental en todo proceso de desarrollo de software que tenga como finalidad la obtención de sistemas informáticos que se ajusten a las necesidades reales de los clientes. Es un proceso centrado en el cliente y sus necesidades, con el objetivo de establecer los servicios que el sistema deberá proveer y las restricciones bajo las cuales deberá operar y ser desarrollado (Sommerville, Ian 2011).

2.4.1 Obtención de requisitos

La captura de requisitos es la etapa de mayor interacción con el usuario, en ella el equipo de desarrollo busca comprender las necesidades que debe cubrir el sistema. El propósito de la captura de requisitos es ganar conocimientos relevantes del problema con el objetivo de producir una especificación formal del software. La ingeniería de requisitos tiene en cuenta técnicas que permiten realizar este proceso de forma eficiente y precisa. Como parte del análisis previo al diseño de la propuesta de solución, se emplearon diferentes técnicas de captura de requisitos (Sommerville, 2011).

La entrevista es una técnica muy aceptada dentro de la ingeniería de requisitos. Su aplicación permite obtener una amplia visión de las necesidades del usuario, proporcionando una mejor

comprensión de los objetivos de la solución buscada(SG Buzz 2018). Las entrevistas realizadas, arrojaron información valiosa acerca de las necesidades y perspectivas de utilización del sistema, resaltando la importancia de desarrollar una aplicación sencilla, amigable y fácil de utilizar.

La tormenta de ideas es un proceso interactivo no estructurado de grupo, donde las opiniones individuales se analizan y estudian en colectivo. Es útil en la generación de una amplia variedad de puntos de vista sobre el problema, así como su formulación de formas diferentes, superando muchas de las barreras de comunicación en la captura de requisitos.(SG Buzz 2018).

2.4.2 Requisitos Funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. Son considerados características requeridas del sistema que expresa una capacidad de acción de su misma funcionalidad; generalmente expresan una declaración en forma verbal. Además son enunciados acerca de servicios que el sistema debe proveer(Sommerville, Ian 2011).

En el análisis realizado, se identificaron los siguientes requisitos funcionales:

RF-1. Realizar auditoría
RF-2. Cargar Excel con información de activo fijo tangible
RF-3. Insertar información de activo fijo tangible
RF-4. Modificar información de activo fijo tangible
RF-5. Buscar información de activo fijo tangible
RF-6. Insertar área de responsabilidad material
RF-7. Modificar área de responsabilidad material
RF-8. Eliminar área de responsabilidad material
RF-9. Mostrar información de área de responsabilidad material
RF-10. Listar áreas de responsabilidad material
RF-11. Listar auditorías
RF-12. Listar auditoría por área de responsabilidad material
RF-13. Buscar área de responsabilidad material
RF-14. Configurar información del auditor

RF-15. Escanear código de barra de activo fijo tangible
RF-16. Escanear código QR de área de responsabilidad material
RF-17. Sincronizar aplicaciones
RF-18. Generar informe resultante de auditoría
RF-19. Generar movimiento de activo fijo tangible
RF-20. Generar acta de responsabilidad material por área

2.4.3 Requisitos No Funcionales

Los requisitos no funcionales son limitaciones sobre servicios o funciones que ofrece el sistema. Los requisitos no funcionales se suelen aplicar al sistema como un todo, más que a características o a servicios individuales del sistema (Sommerville, Ian 2011).

A continuación, se enuncian los requisitos no funcionales identificados en la propuesta de solución:

Software

RNF 1 El sistema debe de funcionar sobre cualquier distribución de sistema operativo superior a Windows Seven en el caso de la aplicación de escritorio y funcionar en dispositivo Android con versión superior a la 5.0.

Hardware

RNF 2 El sistema para su correcta ejecución requiere de 1Gg de memoria RAM

RNF 3 El sistema requiere de un teléfono móvil Androide con más de 2 gb de RAM.

Restricciones de diseño e implementación.

RNF 4 El sistema utilizará como gestor de base de datos para el almacenamiento persistente de información PostgreSQL.

RNF 5 El sistema deberá ser implementado en Kotlin como lenguaje de programación.

RNF 6 El sistema será modelado utilizando como herramienta Visual Paradigm For UML 8.0 para los artefactos ingenieriles.

Apariencia o Interfaz externa.

RNF 7 El sistema poseerá un diseño manteniendo una letra visible y legible.

RNF 8 El sistema contendrá con un diseño gráfico que no debe ser complejo, con colores amigables en la gama de colores azul y blanco que no afecten la visión de los usuarios y totalmente libre de anuncios que puedan distraer a las personas que usan el sistema.

Rendimiento

RNF 9 El sistema deberá ejecutar las operaciones en 5s.

2.5 Fase de Planificación

En la fase de planificación se efectúa un diálogo entre las partes involucradas en el proyecto, incluyendo al cliente y a los programadores. Primeramente, se comienza recopilando las historias de usuario (HU), las cuales sustituyen a las descripciones de casos de uso de una metodología tradicional. El tiempo de desarrollo de cada una de las HU queda evidenciado mediante el plan de iteraciones, siendo ambos los artefactos generados en esta fase.

2.5.1 Historias de usuarios

El artefacto que genera XP para describir el funcionamiento y características de la aplicación son las Historias de Usuario (HU). Estas son el equivalente a las especificaciones de casos de uso, empleados en otras metodologías. Deben poder ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias.

Las HU correspondientes al sistema fueron creadas especificando la prioridad de cada una de ellas. Están representadas mediante tablas con los siguientes elementos:

- **Número:** Número de la HU, incremental en el tiempo.
- **Nombre:** Nombre que identifica la HU.
- **Referencia:** Es el conjunto de HU de las cuales depende la actual.
- **Prioridad:** Esta característica es dada por el cliente con los valores: alta, media o baja en dependencia de la importancia y orden de la implementación.
    - ✓ Baja: Se le otorga a las HU que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura, y no tienen que ver directamente con el sistema en desarrollo.

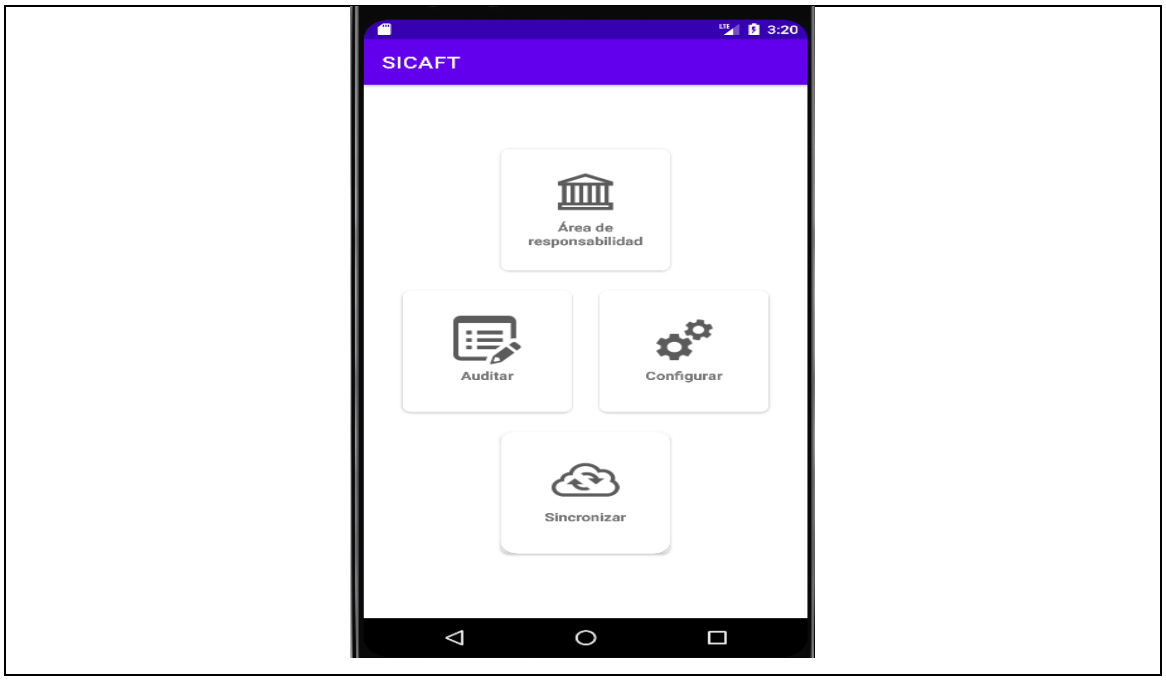
- ✓ Media: Se le otorga a las HU que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación sobre el sistema que se esté desarrollando.
- ✓ Alta: Se le otorga a las HU que resultan funcionalidades fundamentales en el desarrollo del sistema, las que el cliente define como principales para el control del sistema.

- **Iteración Asignada:** Número de la iteración en la cual se desarrollará la HU.
- **Puntos Estimados:** Tiempo estimado en semanas que se le asignará.
- **Descripción:** Breve descripción del proceso que define la historia.
- **Observaciones:** Alguna acotación importante a señalar sobre la historia.

A continuación, se presenta la historia de usuario correspondiente a uno de los requisitos funcionales identificados.

Tabla 1. Historia de usuario "Realizar auditoría"

HISTORIA DE USUARIO	
Número: HU 1	**Nombre del requisito:** Realizar auditoría
Programador: Fernando Córdova	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 4 días
Riesgo en Desarrollo: Medio	**Tiempo Real:** 4 días
Descripción: Permite al usuario escanear el código QR para registrar el área donde se realizará la auditoria y luego se escanean los códigos de barras de cada AFT para ir registrándolos en el sistema.	
Observaciones: El caso de uso se inicia cuando el auditor una vez registrado en la aplicación móvil selecciona el botón auditar, a partir de ahí haciendo uso de la cámara del móvil empieza a realizar la auditoria del área una vez escaneado el código QR de la misma.	
Prototipo elemental de interfaz gráfica de usuario:	



2.5.2 Estimación del esfuerzo por HU

Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto(Analytics 2018), lo que se conoce como estimación de esfuerzo por puntos.

Tabla 2. Estimación del esfuerzo por HU

No.	Nombre de historia de usuario	Puntos de estimación (días)
1	Realizar auditoría	4
2	Cargar Excel con información de activo fijo tangible	1
3	Insertar información de activo fijo tangible	2
3	Modificar información de activo fijo tangible	2
4	Buscar información de activo fijo tangible	1
5	Insertar área de responsabilidad material	1
6	Modificar área de responsabilidad material	1
7	Eliminar área de responsabilidad material	1
8	Mostrar información de área de responsabilidad material	3
9	Listar áreas de responsabilidad material	2
10	Listar auditorías	3

11	Listar auditoría por área de responsabilidad material	1
12	Buscar área de responsabilidad material	2
13	Configurar información del auditor	1
14	Escanear código de barra de activo fijo tangible	5
15	Escanear código QR de área de responsabilidad material	5
16	Sincronizar aplicaciones	1
17	Generar informe resultante de auditoría	1
18	Generar movimiento de activo fijo tangible	1
19	Generar acta de responsabilidad	1

Al realizar la estimación de esfuerzo por puntos, el tiempo de desarrollo obtenido fue de 29 días. Los tiempos estimados de las HU variaron de 1 a 5 días.

2.5.3 Plan de iteraciones

Luego de estimar el tiempo de desarrollo de las HU, se procede a realizar el plan de iteraciones para así implementar por un orden lógico cada una de las funcionalidades propuestas.

- **Iteración 1:** En esta iteración se desarrolla las HU 1, 2, 3, 4, 5 se realiza la auditoria y se carga la información de activos y áreas en la facultad. Como resultado se obtendrá la aplicación en un 30 % de su implementación.
- **Iteración 2:** Se desarrolla las HU 6, 7, 8, 9, 10 que son las encargadas de listar las áreas de responsabilidad y las auditorías realizadas. Como resultado se obtendrá la aplicación en un 50 % de su implementación.
- **Iteración 3:** Se desarrolla las HU 11, 12, 13, 14, 15, 16, 17, 18, 19 que son las encargadas de configurar la información del auditor, escanea código QR y códigos de barras, además de generar el informe de la auditoria, generar el movimiento de AFT y genera las actas de responsabilidad material cada AFT. Como resultado se obtendrá la aplicación al 100% de su implementación.

Una vez identificados los requisitos y descritas las historias de usuario correspondientes, el equipo de desarrollo realiza una estimación del esfuerzo que supone requerirá la implementación de cada una, mientras el cliente define la prioridad en función del valor para el negocio.

Finalmente, se llega a un acuerdo sobre el orden de implementación y el contenido de las entregas, apostando por enfrentar las historias de más valor y riesgo lo antes posible (Pressman, 2007).

El ciclo de desarrollo de software guiado por XP se caracteriza por ser iterativo e incremental. Los niveles de prioridad para el negocio y el esfuerzo estimado de implementación para cada una de las historias de usuario especificadas, hicieron posible la planificación de la implementación del sistema, quedando definidas cuáles historias de usuario serían implementadas en cada iteración.

Tabla 3. Plan de iteraciones

No. Iteración	Historia de usuario	Prioridad	Esfuerzo estimado(días)	Esfuerzo estimado(semanas)
1	Realizar auditoría	Alta	4	1
	Cargar Excel con información de activo fijo tangible	Alta	1	1.5
	Insertar información de activo fijo tangible	Alta	2	
	Modificar información de activo fijo tangible	Alta	2	
	Buscar información de activo fijo tangible	Alta	1	
2	Insertar área de responsabilidad material	Media	1	1
	Modificar área de responsabilidad material	Media	1	
	Mostrar información de área de responsabilidad material	Media	1	
	Listar áreas de responsabilidad material	Media	3	2
	Listar auditorías	Media	2	
3	Listar auditoría por área de responsabilidad material	Baja	3	
	Eliminar área de responsabilidad material	Baja	1	
	Buscar área de responsabilidad material	Baja	2	2
	Configurar información del auditor	Baja	1	
	Escanear código de barra de activo fijo tangible	Baja	5	

	Escanear código QR de área de responsabilidad material	Baja	5	
	Sincronizar aplicaciones	Baja	1	1
	Generar informe resultante de auditoría	Baja	1	
	Generar movimiento de activo fijo tangible	Baja	1	

2.6 Fase de Diseño

En la fase de diseño se confeccionan las tarjetas Clase Responsabilidad Colaborador (CRC) para crear diseños de clases orientados a responsabilidades, haciendo uso de patrones de diseño. Se selecciona la arquitectura adecuada para el desarrollo de la herramienta.

2.6.1 Arquitectura de la solución

Para que un software sea desarrollado con calidad es necesario establecer desde el inicio una arquitectura que describa sus principios fundamentales, garantizando robustez y escalabilidad. La arquitectura de software define la estructura del sistema, constituida por componentes con funciones específicas que interactúan entre sí (evelou 2019).

2.6.1.1 Arquitectura de la aplicación móvil del sistema

Android es un sistema operativo creado para ser independiente de cualquier tipo de arquitectura de hardware en los dispositivos móviles. Esta característica hace que sea tan atractivo ante los fabricantes y desarrolladores.

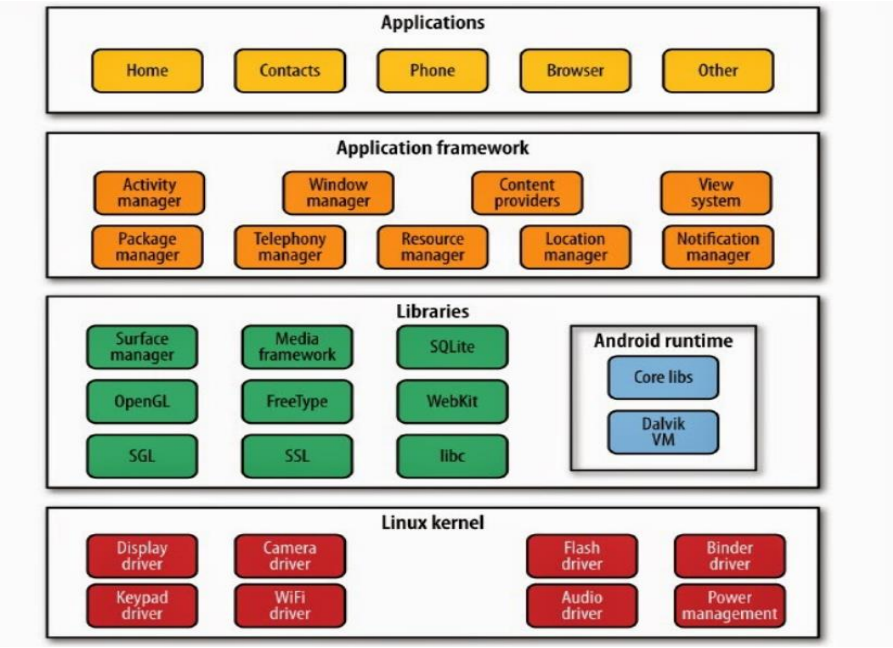


Figura 4. Modelo de capas en la arquitectura de Android

La figura anterior ejemplifica que la estructura de Android se encuentra construida sobre el Kernel de Linux. Luego hay una capa de librerías relacionadas con una estructura administradora en tiempo de ejecución. En el siguiente nivel se encuentra un framework de apoyo para construcción de aplicaciones y posteriormente la capa de aplicaciones (evelou 2019)

2.6.1.2 Arquitectura del servicio API-REST

Arquitectura de microservicios

Los microservicios no son solo un tipo de arquitectura, sino también un modo de abordar la escritura del software. Con ellos, las aplicaciones se dividen en sus elementos más pequeños, que son independientes entre sí. Cada uno de dichos elementos o procesos es un microservicio. El objetivo de usar una arquitectura de microservicios es distribuir un software de calidad con mayor rapidez. Se pueden desarrollar múltiples microservicios de forma simultánea, sin necesidad de volver a diseñar o implementar toda la aplicación después de realizar cambios, ya que los servicios se implementan de forma independiente.

Gracias a ello, varios desarrolladores pueden trabajar en sus servicios individuales al mismo tiempo, en lugar de actualizar toda la aplicación, lo que reduce el tiempo de desarrollo y permite lanzar características nuevas con mayor frecuencia. Junto con los equipos de DevOps y API,

los microservicios en contenedores constituyen la base de las aplicaciones nativas(decide4AI 2019).

2.6.1.3 Arquitectura aplicación de escritorio

Arquitectura Modelo Vista Controlador (MVC)

Para el diseño arquitectónico de la aplicación se hará uso del patrón MVC, el cual garantiza la organización del código fuente de la aplicación, dividiéndola en tres componentes fundamentales: el modelado del dominio, la presentación y las clases controladoras. El Modelo administra el comportamiento y los datos del dominio de aplicación. La Vista define las interfaces de usuario. Posibilita la interacción del usuario y muestra la información obtenida por el modelo del medio persistente. El Controlador responde a eventos que se traducen en solicitudes de servicio para el modelo o la vista, gestionando la interacción entre el usuario y los datos (MVC 2020). La siguiente figura ejemplifica el funcionamiento de este patrón arquitectónico.

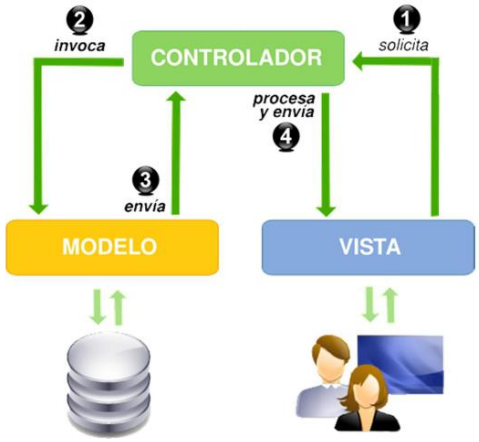


Figura 5. Arquitectura Modelo Vista Controlador

MVC divide una aplicación interactiva en 3 áreas: procesamiento, salida y entrada. Para esto, utiliza las siguientes abstracciones:

- Modelo (Model): Encapsula los datos y las funcionalidades. El modelo es independiente de cualquier representación de salida y/o comportamiento de entrada.
- Vista (View): Muestra la información al usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador.

- Controlador (Controller): Reciben las entradas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas. Los eventos son traducidos a solicitudes de servicio ("service requests") para el modelo o la vista (MVC 2020).

2.6.2 Tarjetas Clase-Responsabilidad-Colaborador (CRC)

Las tarjetas CRC se dividen en tres secciones: nombre de la clase, las responsabilidades (lo que la clase sabe o hace) y los colaboradores (otras clases con las que trabaja en conjunto para llevar a cabo sus funcionalidades) (Pressman, 2007). A continuación, se evidencia la tarjeta CRC correspondiente a la clase Auditar.MainActivity.kt

Tabla 4. Tarjeta CRC "Realizar auditoría"

Tarjetas CRC	
Clase: Auditar.MainActivity.kt	**Colaborador**
Responsabilidad	Auditar. MainActivity.kt Auditor.java Activity_auditores.xml
Es la clase encargada de: • Realizar las auditorias por áreas. • Hacer las consultas necesarias a la base de datos.	

2.6.3 Patrones de diseño

Los patrones de diseño constituyen soluciones genéricas probadas a problemas comunes del desarrollo de software, propician flexibilidad, elegancia y reutilización. Para un mejor diseño de la solución, se hace uso de los patrones que a continuación se abordan.

2.6.3.1 Patrones GRASP

Como patrones generales de asignación de responsabilidades (General Responsibility Assignment Software Patterns, GRASP) se emplearon los siguientes:

Experto: La clase *Activo* es la que contiene la información necesaria para que otra clase controladora instancie un objeto con sus atributos: nombre, descripción, rótulo, área, así como los métodos de obtención y establecimiento de información en estas entidades de tipo activo.

Controlador: La clase *Principal.java* es la encargada de toda la lógica del negocio, la que tiene la responsabilidad de instanciar objetos de tipo activo y gestionarlos en la base de datos, así como manejar debidamente su información.

Creador: La clase *AdapterListaArea* es la encargada de crear o construir listas para registrar y modificar las informaciones por área de los activos respectivamente.

Alta cohesión – Bajo Acoplamiento: en la modelación realizada, se logró obtener un mayor grado de cohesión con un menor grado de acoplamiento. De esta forma se tiene menor dependencia y se especifican los propósitos de cada objeto en el sistema.

2.6.3.2 Patrones GoF

De los patrones popularizados por la autoproclamada “Pandilla de los Cuatro” Gang of Four (GoF) se hizo uso de:

Factory Method (método de fabricación): centraliza en una clase constructora la creación de objetos de un subtipo de un tipo determinado, ocultando al usuario la casuística, es decir, la diversidad de casos particulares que se pueden prever, para elegir el subtipo que crear. Parte del principio de que las subclases determinan la clase a implementar. El uso de este patrón se evidencia en la clase *Activo*.

Builder (constructor virtual): abstrae el proceso de creación de un objeto complejo, centralizando dicho proceso en un único punto, la clase entidad *Activo*.

Abstract Factory (fábrica abstracta): permite trabajar con objetos de distintas familias de manera que las familias no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando. La creación de interfaces gráficas de distintos tipos (ventana, menú, botón, etc.), su uso se muestra en la clase *ActivityPrincipal.java*.

2.6.4 Modelo de datos

Un modelo de datos es un conjunto de herramientas conceptuales para describir la representación de la información en términos de datos. Los modelos de datos comprenden aspectos relacionados con: estructuras y tipos de datos, operaciones y restricciones (SCHMULLER 2011).

El modelo datos puede ser usado como una base para una vista unificada de los datos, adoptando el enfoque más natural del mundo real que consiste en entidades y relaciones. En la siguiente figura se muestra el modelo de datos correspondiente a la solución, donde se representan los datos, sus atributos y tipos, así como las relaciones.

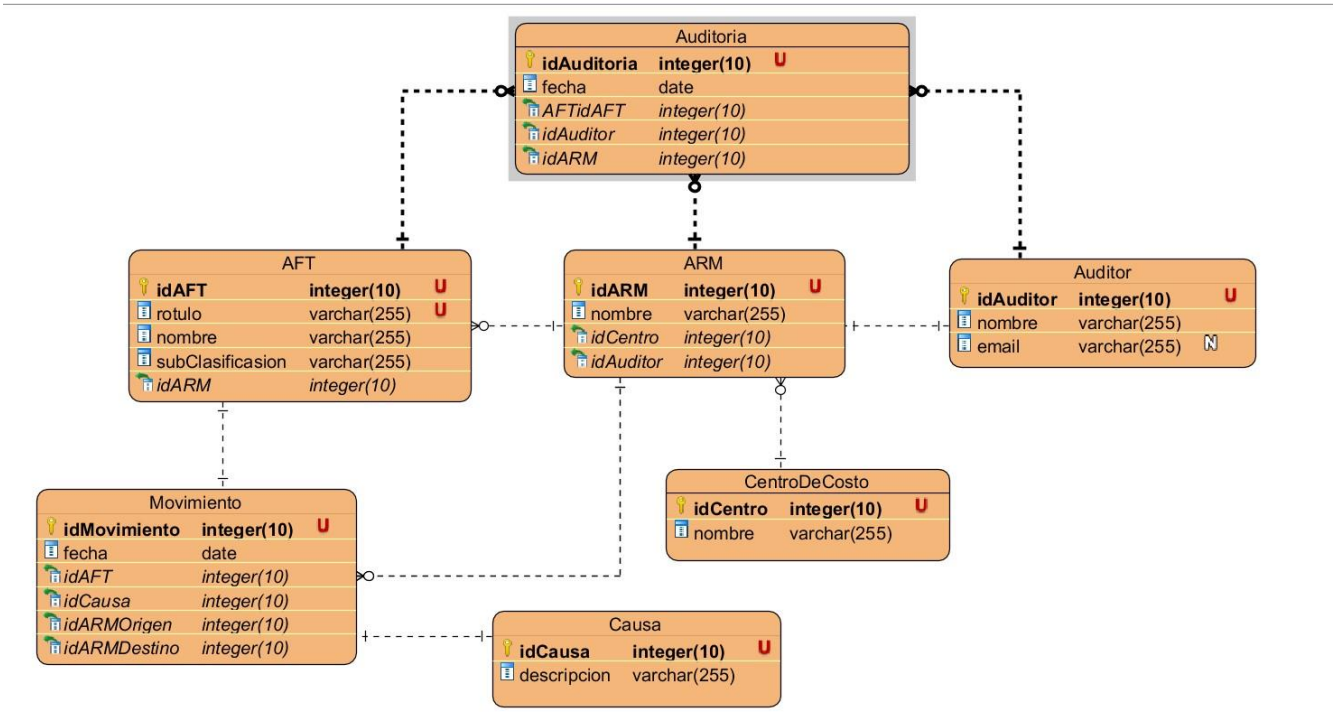


Figura 6. Modelo de datos correspondiente a la solución

2.7 Fase de Desarrollo

En la fase de desarrollo se planifican y ejecutan las tareas de ingeniería. Durante esta se codifica todo el sistema diseñado y se obtiene como resultado el sistema propuesto.

2.7.1 Tareas de ingeniería por iteraciones

Las tareas de ingeniería son actividades que se derivan de las HU para su implementación. A continuación, se muestran las tareas a desarrollar para la implementación, por cada HU.

Tabla 5. Tareas de Ingeniera por iteraciones

Iteración	Historia de usuario	Tareas de ingeniería
	Realizar auditoría	Implementar una funcionalidad que permita a un auditor empezar a realizar auditoria en el sistema.

1	Cargar Excel con información de activo fijo tangible	Implementar una funcionalidad que permita cargar Excel con información de activo fijo tangible en el sistema.
	Insertar información de activo fijo tangible	Implementar una funcionalidad que permita insertar información de activo fijo tangible en el sistema.
	Modificar información de activo fijo tangible	Implementar una funcionalidad que permita modificar información de activo fijo tangible en el sistema.
	Buscar información de activo fijo tangible	Implementar una funcionalidad que permita buscar información de activo fijo tangible en el sistema
2	Insertar área de responsabilidad material	Implementar una funcionalidad que permita insertar el área de responsabilidad material en el sistema
	Modificar área de responsabilidad material	Implementar una funcionalidad que permita modificar el área de responsabilidad material en el sistema
	Eliminar área de responsabilidad material	Implementar una funcionalidad que permita eliminar el área de responsabilidad material en el sistema
	Mostrar información de área de responsabilidad material	Implementar una funcionalidad que permita mostrar información de área de responsabilidad material en el sistema
	Listar áreas de responsabilidad material	Implementar una funcionalidad que permita listar áreas de responsabilidad material en el sistema
3	Listar auditorías	Implementar una funcionalidad que permita listar auditorias en el sistema
	Listar auditoría por área de responsabilidad material	Implementar una funcionalidad que permita listar auditoría por área de responsabilidad material en el sistema
	Buscar área de responsabilidad material	Implementar una funcionalidad que permita buscar área de responsabilidad material en el sistema
	Configurar información del auditor	Implementar una funcionalidad que permita configurar información del auditor en el sistema
	Escanear código de barra de activo fijo tangible	Implementar una funcionalidad que permita escanear código de barra de activo fijo tangible en el sistema
	Escanear código QR de área de responsabilidad material	Implementar una funcionalidad que permita escanear código QR de área de responsabilidad material en el sistema
	Sincronizar aplicaciones	Implementar una funcionalidad que permita sincronizar aplicaciones en el sistema
	Generar informe resultante de auditoría	Implementar una funcionalidad que permita generar informe resultante de auditoría en el sistema
	Generar movimiento de activo fijo tangible	Implementar una funcionalidad que permita generar movimiento de activo fijo tangible en el sistema

2.7.2 Tareas de ingeniería detalladas

A continuación, se muestra la tarea de ingeniería detallada correspondientes a la historia de usuario “Realizar auditoría”.

Tabla 6. Tarea de ingeniería detallada de la HU "Realizar auditoría"

TAREA DE INGENIERÍA	
Número tarea: 1	**Historia de usuario:** Realizar auditoría
Programador: Fernando Córdova	**Nombre Tarea:** Implementar una funcionalidad que permita a un auditor empezar a realizar auditoria en el sistema
Tipo de tarea: Desarrollo **(Desarrollo, Corrección, Mejora)**	**Tiempo Estimado:** 4 días
Fecha inicio: 05/10/2022	**Fecha fin:** 09/10/2022
Descripción: Permite al usuario escanear el código QR para registrar el área donde se realizará la auditoria y luego se escanean los códigos de barras de cada AFT para ir registrándolos en el sistema.	

2.7.3 Estándares de codificación

En función de lograr estandarización en el código del sistema, se definieron un conjunto de pautas a seguir durante la implementación.

Definición de clases: Las clases se encuentran en archivos independientes que solo contendrán el código de esta. Se utiliza el estilo de codificación “*UpperCamelCase*”, el cual establece que los nombres inician con letra mayúscula y si poseen más de una palabra, la primera letra de estas deberá ser mayúsculas también.

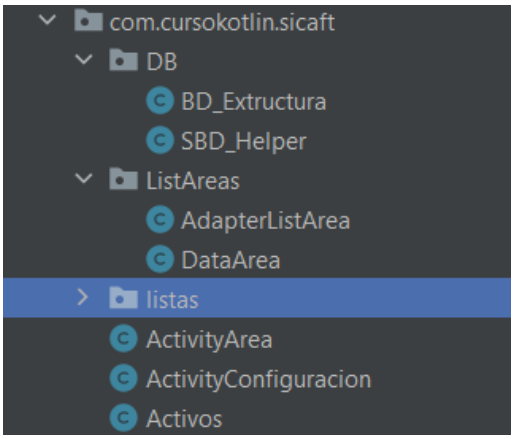


Figura 7. Definición de clases

Definición de variables: Los nombres de las variables deben ser descriptivos y concisos. No se usan abreviaciones. Se utiliza el estilo de codificación "*lowerCamelCase*", el cual establece que los nombres inician con letra minúscula y cada nueva palabra debe iniciar con mayúscula.

```
val btn: Button? = null
val floatingActionButton: FloatingActionButton? = null

fun onClickListener() {
    btn!!.setOnClickListener(this)
    floatingActionButton!!.setOnClickListener(this)
}
```

Figura 8. Definición de variables

2.8 Conclusiones del capítulo

Al finalizar el presente capítulo se arribó a las siguientes conclusiones parciales:

- La confección del modelo de datos y el modelado del proceso de negocio permitieron definir los elementos esenciales para el desarrollo de la propuesta de solución, así como la representación visual del dominio del problema.
- La especificación de los requisitos mediante historias de usuario contribuyó a garantizar la planificación de su implementación en función del esfuerzo necesario y la prioridad para el negocio.
- El empleo de los patrones arquitectónicos permitió obtener una visión general del sistema y una mayor adaptabilidad del mismo.

- Se obtuvieron las tarjetas CRC correspondientes a cada una de las clases y se evidenció como los patrones de diseño pueden contribuir a la obtención de un producto final de mayor calidad.
- Finalmente, se definieron las tareas de ingeniería y los estándares de codificación que servirán de guía para alcanzar un producto no solo funcional, sino de código legible y escalable.

CAPÍTULO 3: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

Una vez detallada la propuesta de solución se procedió a su ejecución. El presente capítulo precisa cada uno de los pasos que desde ese momento tuvo lugar en el afán de continuar y reforzar el aseguramiento de la calidad, iniciado desde la misma planeación del proceso de software. Se ofrece un análisis de cada uno de los niveles de pruebas ejecutados, los términos en que se realizaron las pruebas y los resultados alcanzados.

3.1 Estrategia de pruebas

Una estrategia de prueba de software proporciona una guía que describe los pasos que deben realizarse como parte de la prueba. Debe ser lo suficientemente flexible para promover un uso personalizado de la prueba(Roger S. Pressman 2015). Al probar el software se verifican los resultados de la prueba que se opera para buscar errores, anomalías o información de atributos no funcionales del programa(Sommerville, Ian 2011).

El proceso de prueba tiene dos metas distintas:

- Demostrar al desarrollador y al cliente que el software cumple con los requerimientos.
- Encontrar situaciones donde el comportamiento del software sea incorrecto, indeseable o no esté de acuerdo con su especificación.

Una estrategia de prueba está compuesta por niveles, tipos, métodos y técnicas de pruebas, así como los casos de prueba(Sommerville, Ian 2011).

La metodología XP propone un modelo en el que, lo primero que se escribe son los test que el sistema debe pasar. Luego, el desarrollo debe ser el mínimo necesario para pasar las pruebas previamente definidas. Las pruebas a las que se refiere esta práctica, son las pruebas unitarias, realizadas por los desarrolladores. La definición condiciona el desarrollo del sistema. XP divide las pruebas de software o de sistema en dos grupos, las pruebas unitarias y las pruebas de aceptación(Sommerville, Ian 2011).

3.2 Pruebas de unidad

El nivel de pruebas de unidad se enfoca en cada componente de forma individual o clases de objetos, lo que garantiza que funcione debidamente como unidad, de ahí su nombre. Estas pruebas deben enfocarse en comprobar la funcionalidad de objetos o métodos(Sommerville, Ian 2011).

Las pruebas unitarias son una de las piedras angulares de XP. Todos los módulos deben de pasar las pruebas unitarias antes de ser liberados o publicados. Las pruebas deben ser definidas antes de realizar el código(Sommerville, Ian 2011).

3.2.1 Pruebas automatizadas

Las pruebas automatizadas se han convertido en una de las técnicas más adoptadas en el proceso de pruebas de software, ayuda a acelerar el lanzamiento al mercado al tiempo que proporciona amplio margen para que los profesionales de las pruebas trabajen en la búsqueda de casos de prueba críticos. Las pruebas automáticas no solo mejoran la confiabilidad de su aplicación, sino que también aceleran el proceso de prueba, mejoran la cobertura de prueba y brindan confianza a los evaluadores para una prueba de mejor calidad del producto(Briceño 2020).

JUnit constituye un marco de trabajo de código abierto para la creación de pruebas automatizadas, es una instancia de la arquitectura xUnit para marcos de trabajo enfocados en pruebas de unidad. De conjunto con sus complementos para las principales herramientas de desarrollo (Junit 2020).

Testing en Android – Test Unitarios

Los test permiten verificar el código para asegurar el correcto funcionamiento de este. Con ello se puede controlar las partes complejas de la App y asegurar que cuando se realice una nueva mejora no se estropee nada de lo anterior, proporcionando un código sólido y profesional (kotlinAndroid 2022). Mediante el uso de los test se obtiene:

- Alertas sobre problemas o fallos.
- Detección rápida a problemas mientras se trabaja en futuros desarrollos.

- Simplificar los refactor (mejoras de código) ya que se puede optimizar funciones y asegurar su funcionamiento.
- Velocidad en el desarrollo y minimizar la deuda técnica.

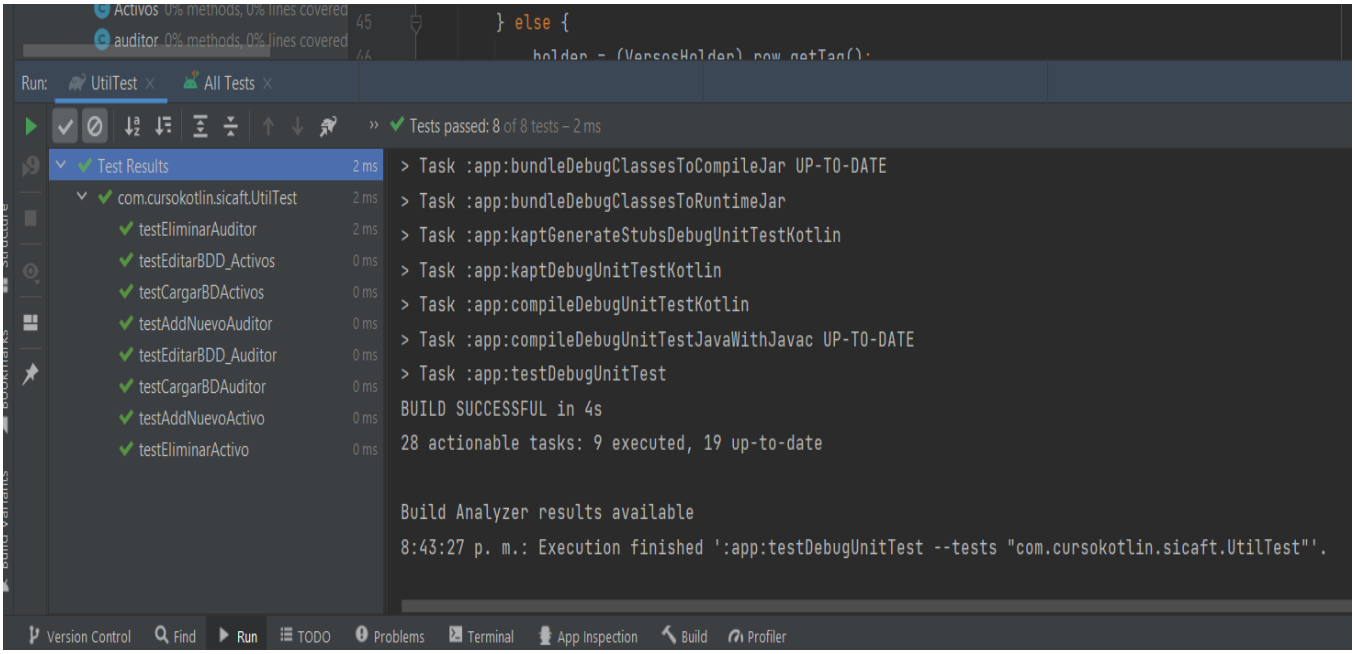


Figura 6. Resultados del Run testing

La utilización de JUnit hizo posible la ejecución controlada de las principales funcionalidades por clase, permitiendo su evaluación acorde al comportamiento esperado. Para lograr un 100 % de efectividad en cada prueba, todos los errores se fueron corrigiendo en la medida en que se fueron revelando. La automatización de las pruebas facilitó su repetición luego de cada cambio importante en el código fuente, garantizando la efectividad de la integración entre los componentes.

3.3 Pruebas de sistema

Esta prueba tiene como objetivo verificar que se han integrado adecuadamente todos los elementos del sistema y que realizan las operaciones apropiadas funcionando como un todo (Sommerville, Ian 2011).

3.3.1 Método de prueba de caja negra

Las pruebas de caja negra se centran en los requisitos funcionales del sistema. Con estas pruebas se intentan encontrar funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en acceso a base de datos externas y errores de rendimiento. Se centran en qué hace el software y no en cómo lo hace(Roger S. Pressman 2015).

Según (Roger S. Pressman 2015) existen varias técnicas para realizar este tipo de pruebas, se seleccionó la técnica partición equivalente, la cual permite comprobar los valores válidos e inválidos de las entradas existentes en la aplicación.

- Partición equivalente: método que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.

La figura a continuación muestra el resultado de la aplicación de esta prueba.

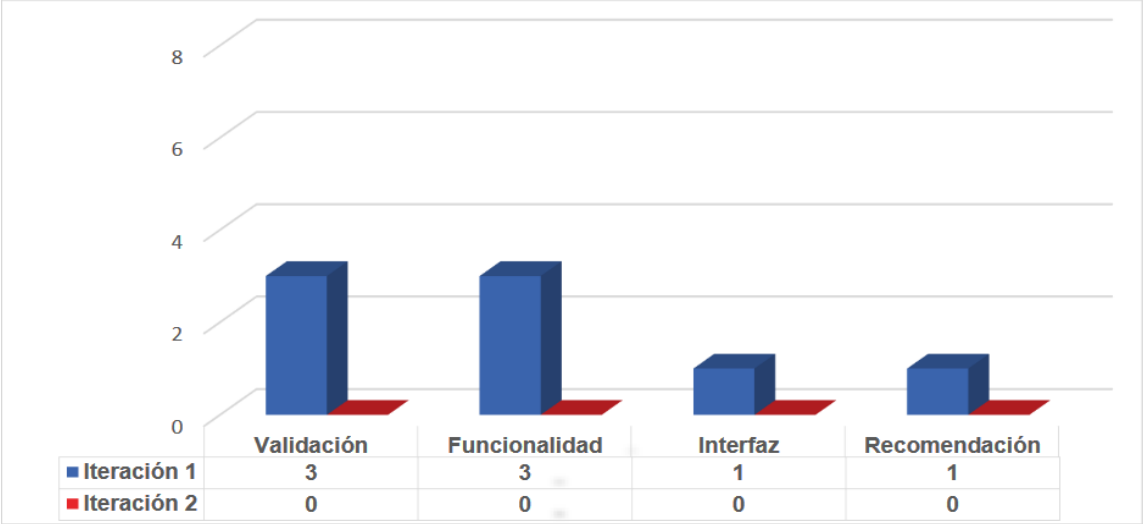


Figura 9. Representación de las no conformidades detectadas

Análisis de los resultados

Se realizaron dos iteraciones y un caso de prueba para cada funcionalidad. En la primera iteración se detectaron 8 no conformidades, siendo las 8 de aplicación, desglosadas en 3 de validación, 3 de funcionalidad, 1 de interfaz y 1 de recomendación. En una segunda iteración estas no conformidades fueron solucionadas obteniéndose resultados satisfactorios.

3.4 Pruebas de aceptación

Las pruebas de aceptación son creadas en base a las historias de usuarios, en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o varios escenarios para comprobar que una historia de usuario ha sido correctamente implementada (Roger S. Pressman 2015).

Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. De esta manera, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución.

Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Teniendo en cuenta que la responsabilidad es grupal, se recomienda sean publicados los resultados de las pruebas de aceptación una vez ejecutadas, de forma tal que todo el equipo esté al tanto de dicha información (Roger S. Pressman 2015). A continuación, se muestra el Casos de Prueba de Aceptación (CPA en lo adelante) de las historias de usuario Realizar Auditoria.

Tabla 7. CPA correspondiente a la Historia de usuario "Realizar auditoría"

CASO DE PRUEBA DE ACEPTACIÓN		
Código: CPA-1	**Historia de usuario:** HU-1 Realizar Auditoria	
Funcionalidad que se prueba: Realizar una auditoria		
Condiciones de ejecución: El caso de uso se inicia cuando el auditor una vez registrado selecciona el botón auditar a partir de ahí con la cámara del móvil empieza a realizar la auditoria del área		
Acción	**Datos de entradas**	**Resultado esperado**
El auditor selecciona el botón auditar	Abre la cámara del móvil	El sistema a partir de ahí empieza el usuario a realizar la auditoria del área seleccionada
Evaluación de la prueba: Satisfactoria		

Seguidamente se evidencian los resultados de la aplicación de los casos de prueba de aceptación.

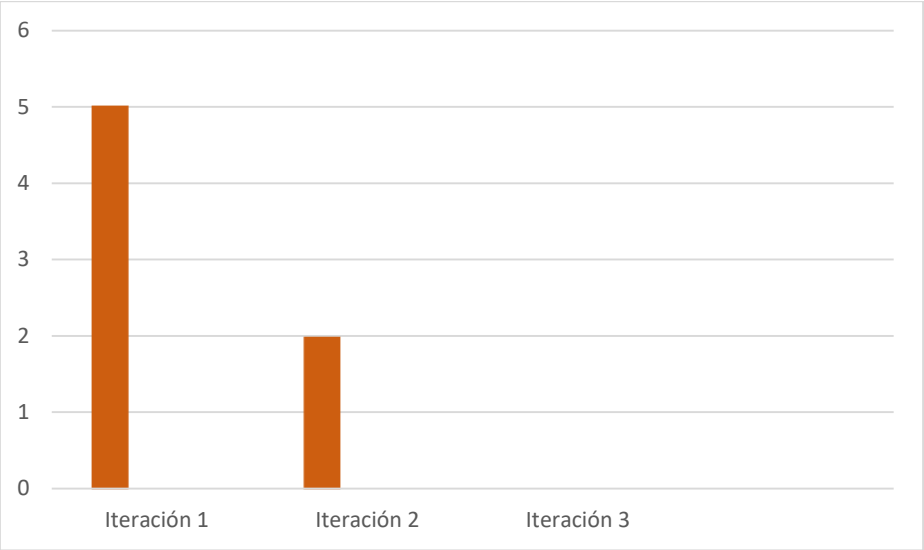


Figura 10. Resultados de la aplicación de los casos de prueba de aceptación

Análisis de los resultados

Para validar que el resultado obtenido por el sistema corresponde con el resultado esperado por el cliente se diseñaron los casos de prueba de aceptación en correspondencia con el número de historias de usuarios. En una primera iteración en la ejecución de los casos de prueba se detectaron un total de 5 no conformidades por parte del cliente, en una segunda fueron identificadas 2 nuevas no conformidades, las cuales fueron finalmente corregidas en una tercera iteración, obteniendo así un resultado exitoso.

3.5 Conclusiones del capítulo

Al finalizar el presente capítulo se arribó a las siguientes conclusiones parciales:

- Definir una estrategia de pruebas contribuyó a un temprano descubrimiento de errores en el sistema.
- Luego de cada cambio significativo en el sistema, se realizaron pruebas de unidad automatizadas a las principales funcionalidades por clases, exigiéndose un 100 % de efectividad para cada una.
- Las pruebas de aceptación permitieron validar la solución, y verificar que se cumplieron los requisitos identificados por el cliente y el equipo de desarrollo.

CONCLUSIONES

Finalizada la presente investigación se puede concluir que se desarrollaron todas las tareas a fin de cumplir los objetivos propuestos, resaltando que:

- El estudio de la bibliografía y análisis de los sistemas existentes para el control y auditoría de activos fijos tangibles demostró la necesidad del desarrollo de un nuevo sistema informático para llevar a cabo la gestión del proceso de auditoría en la Facultad de Ciencias y Tecnologías Computacionales.
- Durante la planificación y el diseño de la solución se obtuvieron los artefactos propuestos por la metodología de desarrollo seleccionada, lo que permitió facilitar la implementación de las funcionalidades definidas.
- La implementación de la propuesta de solución contribuye a la gestión del proceso de auditoría de activos fijos tangibles en la Facultad de Ciencias y Tecnologías Computacionales.
- Las pruebas de software realizadas permitieron garantizar un correcto funcionamiento de la herramienta, así como el cumplimiento de las necesidades y requisitos del cliente.

RECOMENDACIONES

Integrar la solución propuesta con el Sistema de Gestión Administrativa Kainos, para la obtención del inventario de los activos fijos tangibles de cada área de responsabilidad material.

REFERENCIAS BIBLIOGRÁFICAS

ANALYTICS, 2018. Estimación del esfuerzo en proyectos de software utilizando técnicas de inteligencia artificial. [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992014000400001.

ANDROID, 2022. Android Studio. En: Page Version ID: 145450228, *Wikipedia, la enciclopedia libre* [en línea]. [Consulta: 18 octubre 2022]. Disponible en: https://es.wikipedia.org/w/index.php?title=Android_Studio&oldid=145450228.

APACHE, 2021. Apache NetBeans: precios, funciones y opiniones | GetApp España 2022. [en línea]. [Consulta: 21 noviembre 2022]. Disponible en: https://www.getapp.es/software/124747/seed-apache-netbeans.

AYUWARE S.L, 2021. ➤ ¿Qué es un gestor de base de datos (SGBD) y qué funciones tiene? [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://www.ayuware.es/blog/que-es-un-sgbd/.

BAQUERO HERNÁNDEZ, L., PEÑA, D., RODRIGUEZ VALDÉS, O. y MAR CORNELIO, O., 2016. Extensión de la herramienta Visual Paradigm for UML para la evaluación y corrección de Diagramas de Casos de Uso. *Serie Científica de la Universidad de las Ciencias Informáticas*, vol. 9, pp. 1-14.

BRICEÑO, G., 2020. Pruebas Automatizadas: tipos y conceptos erróneos. *Club de Tecnología* [en línea]. [Consulta: 15 noviembre 2022]. Disponible en: https://www.clubdetecnologia.net/blog/2020/pruebas-automatizadas-tipos-y-conceptos-erroneos/.

CELIO GIL AROS, 2020. *ARTÍCULO DE INVESTIGACIÓN CIENTÍFICA y TECNOLÓGICA RuP: METODOLOGÍA EN LOS SISTEMAS Y APLICACIONES BASADAS EN LA WEB*. 2020. S.l.: s.n.

CHÁVEZ ORELLANA 2013, 2019. CHÁVEZ ORELLANA 2013. *OpenWebinars.net* [en línea]. [Consulta: 14 junio 2022]. Disponible en: https://openwebinars.net/blog/por-que-usar-php-y-mysql/.

CHIRIMELLI, M., 2017. ¿ Qué es una aplicación informática? Conoce los detalles. *Android Informa* [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: http://androidinforma.com/una-aplicacion-informatica/.

CONTARERP SOFTWARE, 2022. ContarERP Software de Gestión Empresarial ¿Qué es ContarERP®? *ContarERP Software de Gestión Empresarial* [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://contarerp.com.co/.

CONTRALORÍA GENERAL DE LA REPÚBLICA, 2021. Sistema de control interno. *Contraloría General de la República* [en línea]. [Consulta: 20 octubre 2022]. Disponible en: https://www.contraloria.gob.cu/sistema-de-control-interno.

CORNELIO, O.M., FONSECA, B.B. y CABALLEROS, Y.G., 2016. Sistema para la auditoría y control de los Activos Fijos Tangibles. *Serie Científica de la Universidad de las Ciencias Informáticas* [en línea], vol. 9, no. 6. [Consulta: 20 octubre 2022]. ISSN 2306-2495. Disponible en: https://publicaciones.uci.cu/index.php/serie/article/view/420.

CRAIG LARMAN, 1999. *UML Y Patrones. Una introducion al analisis y diseño orientado a objetos y al proceso unificado*. Segunda. S.l.: s.n. ISBN 970-17-0261-1.

DECIDE4AI, 2019. Arquitectura de microservicios: qué es, ventajas y desventajas. *Decide* [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://decidesoluciones.es/arquitectura-de-microservicios/.

DÍAZ, P.M.P. y CABALLERO, Y.G., 2019. Estado de las herramientas para la auditoría y control de los Activos Fijos Tangibles. *Serie Científica de la Universidad de las Ciencias Informáticas*, vol. 12, no. 4, pp. 43-50. ISSN 2306-2495.

DUEÑAS, J.L.H., PINA, D.D. y MALDONADO, E.A., 2021. Control de activos fijos tangibles mediante el uso de herramientas informáticas novedosas. *Serie Científica de la Universidad de las Ciencias Informáticas*, vol. 14, no. 1, pp. 108-115. ISSN 2306-2495.

ENRÍQUEZ, A.M. y NOGUEIRA, Y.E.M., 2021. Fundamentos teórico-conceptuales de la auditoría de procesos. , pp. 20.

EVELOU, 2019. Aprendiendo Sobre La Arquitectura De Android. [en línea]. [Consulta: 12 noviembre 2022]. Disponible en: https://www.develou.com/aprendiendo-la-arquitectura-de-android/.

GARCÍA ARGÜELLO Y RAMÍREZ PÉREZ (2020), 2020. Resolución 141 de 2007 de Ministerio de la Informática y las Comunicaciones. En: Last Modified: 2022-06-07T09:04-04:00, *Gaceta Oficial* [en línea]. [Consulta: 27 junio 2022]. Disponible en: https://www.gacetaoficial.gob.cu/es/resolucion-141-de-2007-de-ministerio-de-la-informatica-y-las-comunicaciones.

GÓMEZ, E. y CABALEIRO, R., 2021. Los «heritage assets» en los sistemas contables de las entidades públicas. *Empresa global y mercados locales, Vol. 1, 2007-01-01 (Ponencias), ISBN 978-84-7356-500-4, pag. 75*,

HERNÁNDEZ GONZÁLEZ, B., RAMÍREZ RAMÍREZ, T., MAR CORNELIO, O., HERNÁNDEZ GONZÁLEZ, B., RAMÍREZ RAMÍREZ, T. y MAR CORNELIO, O., 2019. Sistema para la auditoría y control de los activos fijos tangibles. *Revista Universidad y Sociedad*, vol. 11, no. 1, pp. 128-134. ISSN 2218-3620.

IGNACIO DAVID ARAYA GIL, 2019. ACTIVO FIJO TANGIBLE - PDF Descargar libre. [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://docplayer.es/10105617-Activo-fijo-tangible.html.

JAVA, 2021. Importancia de Java para Android - Cursos de Java. [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://www.buscaminegocio.com/cursos-de-java/java-para-dispositivos-moviles.html.

JUNIT, 2020. JUnit 5. [en línea]. [Consulta: 12 noviembre 2022]. Disponible en: https://junit.org/junit5/.

KOTLINANDROID, 2022. Testing en Android – Test unitarios. *Curso Kotlin Para ANDROID* [en línea]. [Consulta: 12 noviembre 2022]. Disponible en: https://cursokotlin.com/testing-en-android-test-unitarios/.

MANUEL CONTRERAS, 2015. El Sistema Contable ConDor cuenta con una suite integrada por siete módulos: Contabilidad General, Activos Fijos, Inventario, Nómina/prenómina, - PDF Free Download. [en línea]. [Consulta: 1 julio 2022]. Disponible en: https://docplayer.es/18039798-El-sistema-contable-condor-cuenta-con-una-suite-integrada-por-siete-modulos-contabilidad-general-activos-fijos-inventario-nomina-prenomina.html.

MORY, F.J., 2021. Metodologías Ágiles vs Tradicionales. *Revista Empresarial & Laboral* [en línea]. [Consulta: 1 julio 2022]. Disponible en: https://revistaempresarial.com/tecnologia/metodologias-agiles-vs-tradicionales/.

MVC, 2020. Utilización del Patrón Modelo Vista Controlador (MVC) en el diseño de software educativos. [en línea]. [Consulta: 12 noviembre 2022]. Disponible en: https://www.monografias.com/trabajos43/patron-modelo-vista/patron-modelo-vista.

NODE.JS, 2021. Acerca. *Node.js* [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://nodejs.org/es/about/.

OPENWEBINARS S.L, 2022. SQLite para Android: La herramienta definitiva | OpenWebinars. [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://openwebinars.net/blog/sqlite-para-android-la-herramienta-definitiva/.

PÉREZ PORTO, J., MERINO, M., 2022. Definición de activo fijo - Qué es, Significado y Concepto. *Definición.de* [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://definicion.de/activo-fijo/.

PLASENCIA ASOREY, C., 2010. El Sistema de Control Interno: garantía del logro de los objetivos. *MEDISAN*, vol. 14, no. 5, pp. 0-0. ISSN 1029-3019.

PRESSMAN, R., 2007. *Ingeniería de Software. Un enfoque práctico.* 2007. S.l.: s.n.

PUIG DÍAZ Y GONZÁLEZ CABALLERO (2019), 2011. Zimbra: Bandeja de entrada (98). [en línea]. [Consulta: 20 octubre 2022]. Disponible en: https://correo.estudiantes.uci.cu/#1.

ROGER S. PRESSMAN, 2015. *Ingieneria del software un enfoque practico*. 7ma edicion. S.l.: FreeLibro. ISBN 978-607-15-0314-5.

SCHMULLER, 2011. SCHMULLER, J. Aprendiendo UML en 24 horas. Edtion ed. Naucalpan de Juárez: Pearson Educación, 2000. ISBN 968444463X. - Yahoo Search Results. [en línea]. [Consulta: 11 noviembre 2022]. Disponible en: https://us.search.yahoo.com/search?fr=yhs-invalid&p=SCHMULLER%2C+J.+Aprendiendo+UML+en+24+horas.+Edtion+ed.+Naucalpan+de+Ju%C3%A1rez%3A+Pearson+Educaci%C3%B3n%2C+2000.+ISBN+968444463X.

SG BUZZ, 2018. Obtención de Requerimientos. Técnicas y Estrategia. *SG Buzz* [en línea]. [Consulta: 11 noviembre 2022]. Disponible en: https://sg.com.mx/revista/17/obtencion-requerimientos-tecnicas-y-estrategia.

SISTEMA INFORMÁTICO, 2022. Sistema Informático: qué es, tipos, características y ejemplos. *Enciclopedia Humanidades* [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://humanidades.com/sistema-informatico/.

SOMMERVILLE, IAN, 2011. *Ingeniería del software*. S.l.: Pearson Educación. ISBN 978-84-7829-074-1.

SOMMERVILLE, S., 2011. *Sommerville,* [en línea]. 2011. S.l.: s.n. [Consulta: 9 agosto 2022]. Disponible en: https://www.qualitydevs.com/.

TOKIO SCHOOL, 2021. ¿Qué es JavaFX y para qué se utiliza? - Tokio School. [en línea]. [Consulta: 14 noviembre 2022]. Disponible en: https://www.tokioschool.com/noticias/que-es-javafx-usos/.

VERACODE, 2020. What is IDE or Integrated Development Environments? *Veracode* [en línea]. [Consulta: 11 noviembre 2022]. Disponible en: https://www.veracode.com/security/integrated-development-environment.

VSTUDIOCODE, 2021. Qué es Visual Studio Code y qué ventajas ofrece | OpenWebinars. [en línea]. [Consulta: 21 noviembre 2022]. Disponible en: https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/.

Referencias bibliográficas

ANEXOS

Anexo 1. Entrevista realizada a responsables del control y auditoría de activos fijos tangibles en la Facultad CITEC.

Entrevista

(Realizada a responsables del control y auditoría de activos fijos tangibles en la Facultad CITEC)

Objetivo: conocer cómo se desarrolla el proceso de control y auditoria en la Facultad CITEC.

Compañero o compañera.

Se necesita su colaboración para la realización de esta investigación.

Datos generales

- Nombre y Apellidos
- Rol que desempeña

Aspectos a entrevistar

1. ¿Cómo se realiza el control y auditoría de los activos fijos tangibles en la facultad?
2. ¿Cómo es el proceso de auditoría cuando es solicitado por la Dirección de Contabilidad y Finanzas?
3. ¿Cuántos centros de costo, área de responsabilidad material y activos fijos tangibles existen actualmente en la facultad?
4. ¿Cómo se realiza el proceso de movimiento de un activo si este se encuentra fuera del área que le corresponde?
5. ¿Cuáles son los informes que se generan como parte del resultado de una auditoria?