Universidad de Matanzas

Sede "Camilo Cienfuegos"

Facultad de Ciencias Técnicas

Departamento de Informática

Sistema Automatizado para la Gestión Ambiental Empresarial.

Módulo de Auditoría Interna basado en la NC ISO 14001:2015.

Trabajo de Diploma en Opción al Título de Ingeniero Informático

Autor: Eliza García Tápanes

Tutor: MSc. Liana Romero Lovio

Matanzas, 2018

" El peligro radica en que nuestro poder para dañar o destruir el medio ambiente, o al prójimo, aumenta a mucha mayor velocidad que nuestra sabiduría en el uso de ese poder."

Hawking, Stephen

Dedicatoria

Este trabajo está dedicado a todas esas personas que me han ayudado a ser quien soy hoy en día y la profesional que seré al concluir esta tesis. No puedo decir que el camino ha sido fácil, cuando se desea algo en la vida no te puedes sentar a esperar lograrlo, sino que debes luchar cada día. Gracias a todos por haber estado en mí camino.

Agradecimientos

En primer lugar quiero agradecerle a mis padres que me han dado la vida .Me han enseñado a ser quien soy. Apoyo; es una palabra pequeña para describir el estar presente en cada momento. Mi papá ha sido mi guía, mi referente como persona y amigo, un padre que hace hasta lo imposible por mí que nunca me ha abandonado aunque el camino fuera tormentoso y siempre que he caído me ha levantado. Y mi madre ella es única; me enseño la disciplina, la abnegación, la puntualidad, el dar más de lo que tengo si alguien lo necesita y superarme cada día como estudiante

A mi familia más cercana ,a mis vecinos, los compañeros de trabajo de la EMPAI,particularmente a la Gestora del programa de auditoría Idali a la cual admiro como profesional y amiga de muchos años .A todos ellos por ser incondicionales conmigo y también a los que están lejos por estar siempre pendientes de mí y mis estudios.

A Anelis porque me guio a una persona muy especial que cuando pensé que no podía encontrar refugio en nadie la conocí y desde ese momento siempre ha estado ahí presente; mi tutora Liana. A la cual quiero agradecer por su tiempo, su paciencia, compartir sus conocimientos conmigo y por formar parte fundamental de mi trabajo de diploma.

A mi tata Yoselaine por ser más que una amiga una hermana a pesar de la distancia siempre la tengo presente.

A mis amigos desde mis inicios de la carrera en la UCI, un grupo de varones donde me hicieron sentir la hermana pequeña de todos. A mi grupo en la Universidad de Matanzas en el cual cada persona ha dejado una huella en mí, porque aunque somos muy distintos aprendimos uno de los otros.

A mi mejor amiga en el transcurso de esta carrera Eni con la cual he compartido momentos inolvidables y espero que siga formando parte de mi vida para siempre. Y compañeras de grupo que siempre han estado presente como Iseli, Lisdanis, Eliani y Chintia cuando he necesitado una mano amiga.

A mi esposo el cual conocí en el transcurso de mi carrera y me volvió a llenar la vida de ilusión ,de amor y sobre todo del saber que tienes alguien al lado que te va a seguir en el camino te tomes y va a estar para apoyarte en cada paso que des y a toda su familia.

A Eduardo, una persona como quedan pocas en el mundo .Un magnifico compañero y amigo, por el cual doy gracias a la vida porque sin su apoyo en cada circunstancia, este sueño no hubiera sido posible.

A todos los profesores que he conocido a lo largo de este tiempo y que han aportado un granito de arena a mi crecimiento como estudiante. Especialmente a Liz por estar siempre que la he necesitado y ser un gran apoyo en esta etapa de mi carrera.

A la empresa XETID por darme la oportunidad de vivir la experiencia de integrar un equipo de desarrollo de software durante estos años .De conocer grandes profesionales y amigos .En especial a Yaqueline por su apoyo durante la investigación del proceso.A Genri por estar siempre pendiente por si se presentaba algún problema y estar dispuesto a solucionarlo.A Dayana, Yela,Belkis y Ernesto por compartir sus conocimientos de Ingeniería de Software. Y excelentes programadores como Orlando, Yadier, Gedri, Roider, Yasiel y Maidoli que me dedicaron su preciado tiempo .A los directivos Jasinto y Maikel .A Diego por animarme a entrar en este gran proyecto . A los que fueron encargados de enseñarme lo que fui capaz de hacer en este trabajo de diploma el Yuro y Juan Luis.

A todos muchas gracias

Resumen

En la actualidad nuestro país está inmerso en un proyecto llamado Life(Vida) con el que surge la necesidad de desarrollar una herramienta informática que mantenga el control de la Gestión Ambiental basándose en las normativas del CITMA y en la NC ISO 14001:2015 y cumpliendo con todos sus requerimientos en las empresas cubanas. Es por ello que surge la idea de crear un sistema que pueda suplir esas necesidades. El objetivo fundamental de esta investigación es el desarrollo del módulo Auditoría Interna en SAPGAE. El cual permitirá que las empresas lleven registros del Programa de auditorías de la empresas así como la gestión de las mismas, búsquedas avanzadas reportes y gráficas, así como el tratamiento de sus no conformidades; no de forma independiente ;sino obteniendo resultados de otras funcionalidades para verificar el cumplimiento de los requisitos legales y reglamentarios .La metodología que se utiliza para fundamentar el desarrollo de la investigación es PRODESOFT, abordando las definiciones asociadas al objeto de estudio y al campo de acción. . El sistema es desarrollado en el marco de trabajo Zeolides. Se empleó PHP como Lenguaje de Programación y PostgreSQL como Gestor de Bases de Datos.

Summary

Currently our country is immersed in a project called Life with which arises the need to develop a computer tool that maintains control of Environmental Management based on the regulations of the CITMA and the ISO 14001: 2015 and fulfilling with all your requirements in Cuban companies. That is why the idea of creating a system that can supply those needs arises. The fundamental objective of this research is the development of the Internal Audit module in SAPGAE. Which will allow companies to keep records of the audit program of the companies as well as their management, advanced searches, reports and graphics, as well as the treatment of their non-conformities; not independently, but obtaining results from other functionalities to verify compliance with legal and regulatory requirements. The methodology used to support the development of research is PRODESOFT, addressing the definitions associated with the object of study and the field of action . . The system is developed in the Zeolides framework. PHP was used as Programming Language and PostgreSQL as Database Manager.

Índice

Índice de figuras

Índice de tablas

Índice de anexos

Introducción

La informática por su gran desarrollo, ha venido transformando rápidamente las sociedades actuales. Las nuevas tecnologías nos permiten avanzar a pasos agigantados, ya que son aplicadas en los distintos ámbitos del quehacer cotidiano del hombre, al punto que han revolucionado la sociedad de una manera sorprendente y su desarrollo ha contribuido a la modernización y a un manejo totalmente diferente de la información.

El avance tecnológico que se ha alcanzado en los últimos tiempos, además del mayor aprovechamiento de las tecnologías de la información y las comunicaciones, ha propiciado un auge en la aparición de software de gestión en el contexto nacional. Estos cambios han alcanzado mayor auge gracias a la rápida expansión de Internet y las aplicaciones web (RAMOS 2018).

Por otra parte, el entorno económico de las empresas está cambiando rápidamente debido a las grandes alteraciones que se están produciendo en el mercado. Ante este nuevo panorama, al que todos nos referimos como globalización, las empresas se están adaptando y definiendo estrategias que fomenten la innovación, el desarrollo tecnológico y la internacionalización con el fin de incrementar su competitividad e integración en el mercado mundial (BAKAIKOA *et. al.* 2004).

En Cuba, a pesar de las carencias tecnológicas existentes, varias instituciones están actualizando sus servicios y formas de gestión, con el objetivo de aumentar sus ingresos y simplificar la burocracia existente. Muestra de ello es la gestión del medio ambiente, el cual constituye en la actualidad un tema crucial para el éxito de cualquier negocio o proyecto social.

En la medida que crece la preocupación por mantener y mejorar la calidad del medio ambiente y proteger la salud humana, diversas organizaciones están volcando su atención hacia los impactos potenciales de sus actividades, productos y servicios. El desempeño ambiental de una organización es de creciente importancia para las partes interesadas tanto internas como externas, por lo que el logro de un desempeño ambiental razonable, requiere de un compromiso de la organización con un enfoque sistemático y un mejoramiento continuo de su Sistema de Gestión Ambiental (ESCOBAR 2003).

La implementación de un Sistema de Gestión Medioambiental (SGA) en la empresa constituye un marco apropiado para gestionar los impactos que se producen en el medio ambiente. Además de minimizar los impactos negativos sobre el medio ambiente, un SGA puede reducir los costos, mejorar la eficiencia y dar una ventaja competitiva a las empresas (GARCÍA 2012).

Los SGA constituyen una fuerte herramienta de gestión, para aquellas entidades que pretenden ejercer de manera sostenible sus producciones. Cuba ha evolucionado en este ámbito, demostrado por el

número de certificaciones obtenido en los últimos años. El sector empresarial cubano aplica como mecanismo para la mejora continua de estos sistemas, las auditorías ambientales internas. La gestión de este tipo de auditorías se torna un poco engorrosa, principalmente en la programación y seguimiento del proceso (SANTANA 2014).

Los estándares más utilizados para lograr estos propósitos son las normativas, en particular la NC ISO 14001:2015 que es la encargada de los requisitos para el uso de los Sistemas de Gestión Ambiental y que sustituye a la NC ISO 14001:2004. En la actualidad varias empresas de nuestro país están tomando medidas para adaptar sus Sistemas Ambientales a esta nueva norma basada en riesgos y ciclo de vida, además especifica las directrices para cualquier organización y pueden ser aplicadas en cualquier momento. Esta norma internacional ayuda a una organización a lograr los resultados previstos de su sistema de gestión ambiental, con lo que aporta valor al medio ambiente, a la propia organización y a sus partes interesadas.

Las organizaciones deben llevar a cabo auditorías internas a intervalos planificados para proporcionar información acerca de si el sistema de gestión ambiental cumple con los requisitos propios de la organización y los requisitos de la norma internacional. Además, se debe establecer e implementar un programa de auditoría interna y conservar información documentada para lo cual se deben tener en cuenta determinados aspectos en su implementación como los referidos en la NC ISO 19011:2011 Directrices para las auditorías de los Sistemas de Gestión (SANTANA 2014).

En Cuba se realizan auditorías internas para mantener la calidad de su Sistema de Gestión Ambiental u optar por la certificación de la norma. Pero en la práctica muchas de estas empresas no conocen el procedimiento como lo establecen las normativas o realizan un proceso de baja calidad .Existen deficiencias tales como: el control de los auditores capacitados para realizar estas auditorías, la visualización de las evidencias y las gestión de los hallazgos respecto a los criterios de auditoría, el seguimiento de las no conformidades existente así como su tratamiento a través de las acciones correctivas, el gran número de documentación generada por cada programa de auditoría, su planificación e informe de auditoría interna, los cuales son de gran importancia cuando se realiza un control externo y que en ocasiones sufren pérdidas así como la evidencia de que se cumple con los requisitos establecidos.

Actualmente, la Empresa de Tecnologías de la Información para la Defensa (XETID), conjuntamente con el Centro de Investigación y Desarrollo (CIDES), se encuentran inmersos en el desarrollo del Sistema Automatizado para el apoyo a la Gestión Ambiental Empresarial (SAPGAE), con el

asesoramiento del Ministerio de Ciencia Tecnología y Medio Ambiente (CITMA) y de los investigadores de la Universidad de Matanzas.

Como parte del desarrollo de este sistema automatizado y para dar cumplimiento a la NC ISO 14001:2015, se hace necesario el desarrollo de un módulo de auditoría interna que permita a las empresas conocer su desempeño respecto al medio ambiente y que constituya un aval para que las mismas sean certificadas por la Oficina Nacional de Normalización y brinden evidencia a las auditorías externas que son realizadas por el CITMA.

A partir de la situación problémica anterior se plantea el siguiente problema científico: ¿Cómo facilitar la gestión de auditorías internas ambientales en las empresas cubanas?

La solución a este problema se inserta en el siguiente objeto de estudio: La gestión de auditorías internas ambientales, lo que propicia penetrar en un campo de acción centrado en: Sistema automatizado para la gestión de auditorías internas ambientales en las empresas cubanas.

Para dar respuesta al problema científico esta investigación tiene como objetivo general: Desarrollar un módulo Auditoría Interna para SAPGAE que facilite la gestión de auditorías internas ambientales en las empresas cubanas.

Para dar cumplimiento al objetivo que de solución al problema planteado, se plantean los siguientes objetivos específicos:

1. Analizar los fundamentos teóricos metodológicos relacionados con el desarrollo de un sistema automatizado para la gestión de auditorías internas ambientales en las empresas cubanas.
2. Seleccionar las tecnologías y herramientas de desarrollo que serán utilizadas en la implementación de la propuesta de solución.
3. Determinar las funcionalidades que debe tener la propuesta de solución.
4. Validar la propuesta de solución.

La investigación se sustenta en la siguiente hipótesis: El desarrollo del módulo Auditoría Interna para el sistema SAPGAE facilitará la gestión de auditorías internas ambientales en las empresas cubanas.

De esta forma se pretende obtener una herramienta automatizada que facilite la gestión de auditorías internas ambientales en las empresas cubanas. La misma estará basada en el punto 9 de la NC 14001:2015, utilizada para la validación de sistema y las normativas y metodologías establecidas por el CITMA. Permitirá detectar el estado ambiental en la institución, analizar las causas de los problemas, identificar los procesos que requieren acciones de carácter correctivo y mejora el desempeño de la

empresa. Además, facilitará las búsquedas avanzadas de información y la organización de la misma logrando mantener de forma segura la integridad, extracción, manipulación y persistencia de los datos.

En el desarrollo de la investigación se utilizarán diferentes métodos, los cuales serán seleccionados, elaborados y aplicados sobre la base del método materialista dialéctico. Dentro de los métodos teóricos a ser utilizados están:

Análisis-síntesis: Este se utilizó durante el desarrollo de la fundamentación de la investigación, principalmente para la caracterización del proceso de gestión de auditorías internas y el estudio de las diferentes herramientas y metodologías para el desarrollo del sistema.

Inducción-deducción: Se empleó para inducir las características generales del proceso y a partir de ellas obtener deducciones y conclusiones en la confección del documento y el producto final.

Histórico-lógico: Permitió estudiar no solo el desarrollo y las tendencias actuales de los Sistemas de Gestión Ambiental, sino también la evolución y desarrollo de herramientas automatizadas aplicadas a la gestión de auditorías internas ambientales en el mundo y en Cuba, en particular.

A nivel empírico se utilizarán:

Entrevista: Se empleó como vía fundamental para la identificación de los requerimientos del sistema, lo que posibilitó conocer cómo se realizan las auditorías internas desde el inicio el proceso en el programa de auditoría.

Observación: Se aplicó con el objetivo de captar información de forma visual, es decir, mediante observaciones realizadas a otro software ya existente vinculados con el tema.Se pudieron tomar elementos que sirvieron de apoyo para el desarrollo de esta herramienta.

El presente trabajo está compuesto por una Introducción, tres Capítulos, Conclusiones, Recomendaciones, Referencias Bibliográficas, que apoyadas en 17 figuras y 9 tablas, constituyen la totalidad del documento presentado como trabajo de diploma. A continuación se presenta una síntesis de cada uno de los capítulos.

En el **Capítulo 1** ´´Marco teórico referencial´´ se presenta una exposición detallada de los referentes teóricos que argumentan la propuesta y permiten un acercamiento al objeto de estudio. Además se comenta sobre el estado del arte, las tendencias y tecnologías actuales que serán usadas para el desarrollo de la herramienta informática propuesta.

En el **Capítulo 2** ´´Análisis y diseño de la solución propuesta´´ se argumenta la solución que se propone al problema de investigación mediante su descripción. Se presenta el empleo de la

metodología de desarrollo de software PRODESOFT, y se realiza un estudio tanto de factibilidad, como de los beneficios tangibles e intangibles para la realización de la herramienta informática.

En el **Capítulo 3** ´´Validación de la solución propuesta´´ se describe el software y se le realizan las pruebas con el objetivo de entregarle al cliente un producto totalmente funcional, que cumpla con todos los requisitos demandados por el mismo y que satisfaga sus necesidades.

Capítulo 1. Marco teórico referencial

1.1. Introducción.

En este capítulo se lleva a cabo una descripción teórica de las temáticas que sirven de apoyo a la investigación a desarrollar. Se describe las auditorías internas ambiental en el sistema de gestión ambiental, así como los antecedentes necesarios su comprensión, las dificultades que se presentan y la propuesta de solución. Se definen las principales herramientas tecnológicas, lenguajes de programación y metodología para su desarrollo.

1.2. Principales conceptos asociados al desarrollo de la investigación.

La autora considera que existen ciertos conceptos básicos de vital importancia para la comprensión de la investigación. Estos proporcionan una mayor información para el entendimiento de las relaciones existentes entre disímiles aspectos que permite verlo como un todo y no de manera individual.

Sistema de Gestión Ambiental: Para lograr los resultados previstos, incluida la mejora de su desempeño ambiental, la organización debe establecer, implementar, mantener y mejorar continuamente un sistema de gestión ambiental, que incluya los procesos necesarios y sus interacciones, de acuerdo con los requisitos de la NC ISO14001:2015 Sistema de Gestión Ambiental-Requisitos con orientación para su uso (NC 2015).

Programa de auditoría interna: Detalles acordados para un conjunto de una o más auditorías planificadas para un período de tiempo determinado y dirigida a un propósito específico (NC 2015).

Auditoría: Proceso sistemático, independiente y documentado para obtener las evidencias de auditoría y evaluarlas de manera objetiva con el fin de determinar el grado en el que se cumplen los criterios de auditoría. Una auditoría interna la realiza la propia organización o una parte externa en su nombre (NC 2015).

Equipo auditor: Uno o más auditores que llevan a cabo auditorías, con el apoyo si es necesario de expertos técnicos. A un auditor del equipo se le designa como líder del mismo. Puede incluir también auditores en formación (NC 2012).

Experto técnico: Es la persona que aporta conocimientos o experiencias específicas al equipo. El conocimiento o experiencia específico son las relaciones con la organización, el proceso o actividad a auditar (NC 2012).

El observador: Es la persona q acompaña al equipo pero que no audita (NC 2012).

Criterios: Los criterios de auditoría son el conjunto de políticas, procedimientos o requisitos usados como referencia contra el cual se compara la evidencia de auditoría. Si los criterios de auditoría son requisitos legales incluyendo los reglamentarios, el término cumple y no cumple se utiliza a menudo en los hallazgos de auditoría (NC 2012).

Requisito: Necesidad o expectativa establecida, generalmente implícita u obligatoria. Los requisitos diferentes de los legales se convierten en obligatorios cuando la organización decide cumplirlos (NC 2012).

Evidencia: Registro, declaraciones de hecho o cualquier otra información que es pertinente para los criterios de auditoría y que es verificable (NC 2012).

Hallazgos: Es el resultado de la evaluación de la evidencia de auditoría recopilada frente a los criterios de auditoría. Pueden indicar conformidad o no conformidad. Pueden conducir a la identificación de oportunidades de mejora o el registro de buenas prácticas (NC 2012).

No conformidad: Incumplimiento de un requisito (NC 2012).

Conformidad: Cumplimiento de un requisito (NC 2012).

1.3. Descripción del objeto de estudio.

El proceso de auditorías internas en las empresas cubanas comienza con el programa de auditorías internas. Este programa establece, según la NC ISO14001:2015 Sistema de gestión ambiental-Requisitos con orientación para su uso, que la organización debe establecer, implementar y mantener uno o varios programas de auditoría interna, donde se deben definir los criterios de auditoría, el alcance para cada auditoría, seleccionar los auditores y llevar a cabo auditorías para asegurarse de la objetividad y la imparcialidad del proceso de auditoría. La organización debe conservar información documentada como evidencia de la implantación del programa de auditoría y los resultados de esta (NC, 2015). No obstante, en algunas empresas no se establece este programa por lo que se incumple con las normativas vigentes, además de no tener evidencia del mismo y de no establecer el alcance, los procedimientos, métodos, riesgos a tener en cuenta y recursos previos a la hora de realizar las auditorías internas. En este programa el gestor del programa de auditoría define quien va a conformar el equipo de auditores y que cargo va a desempeñar. Con esta información desarrolla un cronograma con las auditorías internas a realizar en donde define las auditorías que se van a realizar, en qué proceso, según qué criterios, cuál va a ser el equipo auditor que se le va a asignar de los anteriormente creados en el programa y el mes en que se va a desarrollar la auditoría. También

designa a las personas encargadas de su revisión y aprobación para posteriormente obtener un informe del programa que servirá de apoyo a los auditores líderes en cada equipo y será informado a la dirección de la entidad.

En varias ocasiones los gestores en las empresas no informan de este programa a las personas involucradas o sufre pérdida este documento, por lo cual las auditorías no son realizadas ni organizadas.

Por otra parte y según NC (2012), el informe que se obtiene de las auditorías internas debería proporcionar un registro completo, preciso y claro de la auditoría y debería incluir o hacer referencia a los objetivos y el alcance de estas, particularmente:

- ✓ La identificación de los procesos auditados
- ✓ El cliente
- ✓ La identificación del equipo auditor
- ✓ Los participantes del auditado en la auditoría
- ✓ La fecha y ubicación donde se realizaron las actividades
- ✓ Los criterios
- ✓ Los hallazgos de auditoría y las evidencias relacionadas
- ✓ Las conclusiones de la auditoría

Estos informes sufren deficiencias ya que los auditores no registran los criterios en los que se basaron y la evidencia en la cuales fueron detectados los hallazgos de auditorías así como el tratamiento de las no conformidades encontradas.

1.4. Análisis de soluciones existentes.

En diversas empresas de nuestro país hay una gran carencia organizativa respecto a las normativas de gestión ambiental establecidas por el CITMA y la Oficina Nacional de Normalización. La gestión de las auditorías internas ambientales actualmente se realiza, en la mayoría de las empresas, con resultados muy limitados e inconcluyentes, presentando desventajas tales como la inexistencia de búsqueda avanzada, falta de organización, seguridad y precisión.

Hasta el momento de esta investigación no existía en el país una herramienta informática que gestionara las auditorías internas vinculadas a un SGA. Los estudios ambientales actualmente se realizan mediante pequeños sistemas con resultados muy limitados o en el marco internacional, fuera del alcance de nuestro país. Algunos de los sistemas analizados son:

- ***eco2biz***

Desarrollado en Perú para las empresas mineras. Cuenta con un módulo que permite registrar, gestionar y consultar oportunamente todas las evaluaciones (Auditoría, Inspección o Fiscalización) que las autoridades o la empresa considera evaluar. Dentro de sus principales funcionalidades se encuentran:

- ✓ Registros de empresas auditoras y supervisoras.
- ✓ Registro de supervisores de las empresas auditoras.
- ✓ Registro de la norma y requisitos de la norma.
- ✓ Registro de la evaluación: auditoría, inspección, fiscalización.
- ✓ Genera el cronograma de verificación de la fecha de evaluación.
- ✓ Registro de resultado de las evaluaciones realizadas.
- ✓ Registro de acciones y observaciones.

- ***EcoGestor***

Con este software, en una misma herramienta la empresa dispone de:

- ✓ Documentación: Política, manuales, procedimientos, instrucciones técnicas y todo tipo de documentos asociados al sistema para administrar, distribuir y firmar según los condicionantes que se requieran. Se evita el manejo de gran cantidad de papel, pasando todos los archivos a soporte informático.
- ✓ Legislación: Consultores expertos actualizan la normativa de aplicación en la organización y sus obligaciones legales, incluye también un servicio de alertas al correo electrónico, personalizable para recibir tanto novedades legislativas como vencimientos de tareas legales.
- ✓ Auditoría: Ágil checklist que permite revisar de forma exhaustiva el cumplimiento legal del centro y la auditoría interna del sistema de gestión, ganando en precisión y con un importante ahorro de tiempo.
- ✓ No conformidades: Permite realizar la gestión de incidencias de forma automatizada, incluyendo su registro, gestión y cierre final.
- ✓ Gestión: Módulo específico con los puntos exigidos por los sistemas de gestión ambientales.

- ***eGAM Medio Ambiente***

Es un software diseñado para ayudar a cualquier empresa u organización a implantar, mantener y certificar su sistema de gestión ambiental de acuerdo a lo exigido en la Norma ISO 14001, que permite despreocuparse de las evidencias y procedimientos, eliminando la burocracia, aumentando la productividad y reduciendo el costo. El software eGAM Medio Ambiente está construido sobre la plataforma tecnológica eGAMbpm, que automatiza los procedimientos requeridos por la Norma ISO 14001 de manera que cuando se necesita ejecutar un proceso, el procedimiento correspondiente se carga instantáneamente en el sistema, distribuyendo automáticamente tareas, instrucciones a responsables y plazos, de forma que la propia ejecución de las tareas autogenera las evidencias necesarias del sistema de gestión.

- ***SE Audit***

SE Audit es un software planificado para ayudar a las organizaciones a administrar la amplia gama de actividades, datos y procesos relacionados con las auditorías en un ambiente único y amplio. Proporciona la flexibilidad de soportar todos los tipos de auditoría, incluyendo auditorías internas, operacionales, de TI, de proveedores, de riesgos/controles y auditorías de calidad. El sistema también ofrece funcionalidades para administrar el ciclo de vida completo de una auditoría incluyendo planificación, programación, preparación, desarrollo de planes de auditoría estándar y checklists, colecta de datos, ejecución, informes y monitoreo. SE Audit contempla todos los controles requeridos para atender los requisitos de reglamentos internacionales relacionados con responsabilidad social, gobernanza, calidad, medio ambiente, salud y seguridad. Entre ellos están: ISO 26000, ISO 9001, ISO 14001, OHSAS 18001, ISO 19011, ISO 13485 y FDA.

- ***Audita***

Está creado especialmente para cubrir integralmente la informatización de todas las actividades que el área de auditoría debe realizar para cumplir con su misión. Este producto se complementa con Audita2 creado para que los sectores auditados interactúen con el área de auditoría y el comité de auditoría. El producto fue diseñado para cubrir regulaciones vigentes tales como: planificación por riesgos y seguimiento de observaciones por parte del Comité, de las entidades financieras argentinas, así como del Banco Central de la República Argentina.

- ***AUDITORI***

Herramienta informática desarrollada que contribuye a la gestión de auditorías internas, a partir de los resultados generados mediante la modelación del proceso en el Grupo Empresarial de la

Construcción en Granma. En la misma se realiza el registro del programa de auditoría, inicio de la auditoría individual, registro y listado del equipo auditor, edición de los objetivos y alcance de la auditoría, registro de los requisitos de confidencialidad y conservación, registro de la documentación, registro y listado del cronograma de reuniones, registro y listado de las tareas asignadas, registro de hallazgos y control de la planificación realizada.

Todas estas herramientas constituyen excelentes opciones, sin embargo, la mayoría de ellas no se encuentran fácilmente accesibles para las empresas cubanas y si bien este grupo cumple con características importantes, no obedecen a la condición imprescindibles de que la gestión debe ser siempre desde la perspectiva de las empresas cubanas. Por otra parte, el sistema AUDITORI, creado en Cuba, se realizó en el 2014, por lo que se basó en las normativas de la ISO 14001:2004 y no en la NC ISO 14001: 2015. SE Audit, Audita y AUDITORI, a pesar de abarcar el proceso de auditoría, no están integrados a un Sistema de Gestión Ambiental, el cual fue uno de los principales requisitos planteados por el cliente. Otros como EGAM Medio Ambiente y EcoGestor no controlan aspectos de la auditoría interna de forma centralizada.

A partir de las insuficiencias detectadas y la poca accesibilidad que presenta el sector empresarial cubano a las tecnologías de la información, la presente investigación propone desarrollar un módulo para la gestión de las auditorías internas para el sistema SAPGAE, que contribuya a la gestión de auditorías de SGA basados en la NC ISO 14001: 2015.

1.5. Fundamentación de la metodología utilizada.

PRODESOF (Proceso de Desarrollo y Gestión de Proyectos de Software): Según Proceso de Desarrollo y Gestión de Proyectos de Software (2012), es una combinación de las características de algunas de las metodologías de desarrollo de software más utilizadas en el mundo, que más se ajustan a las condiciones de trabajo del centro, resultando ser robusta, flexible, ágil y fácil de asimilar por todos los implicados en un proyecto. Su proceso es iterativo e incremental, es decir, un enfoque en el que el ciclo de vida está compuesto por iteraciones, estas son pequeños procesos compuestos de varias actividades cuyo objetivo es entregar una parte del sistema parcialmente completo, probado, integrado y estable. En cada iteración se obtiene como resultado un incremento. Es basado en componentes, lo que permite alcanzar un mayor nivel de reutilización de software, aún en contextos distintos a aquellos para los que fue diseñado. Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes ensamblados. Dado que un componente puede ser construido y luego mejorado continuamente, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo. El ciclo de vida del proceso de desarrollo

está compuesto por 5 fases: inicio, modelación, construcción, explotación experimental y despliegue como se muestra a continuación:

Figura 1. Ciclo de vida del proceso de desarrollo.

Durante la fase de inicio se logra una visión preliminar de la problemática a resolver y se definen los recursos relevantes para la ejecución del proyecto. En la modelación se capturan las partes esenciales del sistema, donde se identifican los procesos y se aceptan los requisitos funcionales del sistema, obteniéndose la línea base de la arquitectura y una estrategia de construcción de la aplicación aprobada por los implicados en el proyecto. En la construcción se aclaran los requisitos restantes y se completa el desarrollo del sistema sobre una base estable de la arquitectura. Las fases anteriores sólo proveen una arquitectura básica que es refinada, en la fase de construcción, de manera incremental conforme se construye el producto. En esta fase todas las características, componentes, y requisitos deben ser integrados, implementados, y probados en su totalidad, obteniendo una versión liberada del producto. La explotación experimental convierte la versión liberada del producto en una solución estable, donde se eliminan los errores que surgen durante las pruebas y se obtiene una certificación funcional y de seguridad del producto. En la fase de despliegue se instala y configura el sistema para un ambiente de producción real, se capacita al personal que usará la aplicación y se continúa dando soporte durante la explotación del sistema, culminando de ser preciso con transferencias tecnológicas (PROCESO DE DESARROLLO Y GESTIÓN DE PROYECTOS DE SOFTWARE 2012).

En el anexo # 1 se muestra una comparación entre PRODESOF y las metodologías ágiles y robustas, de forma general.

1.6. Fundamentación de las herramientas empleadas.

Luego de un estudio de las tecnologías actuales en el mercado para el desarrollo de software, se decidió dadas las ventajas y el conocimiento del personal de desarrollo de este trabajo utilizar las herramientas que se caracterizan a continuación.

1.6.1. Herramienta de modelado.

Visual Parading: Es una de las herramientas CASE más utilizadas para el modelado de software. Esta herramienta cuenta con los medios necesarios para extender sus funcionalidades, pues da soporte a las extensiones de aplicación. Provee de forma libre una interfaz de programación, que permite a los desarrolladores implementar y reutilizar clases e interfaces y desarrollar funciones agregadas que son útiles para el desarrollo de software (QUINTERO *et al*. 2012). Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite modelar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación.

1.6.2. Lenguaje de modelado.

UML: El Lenguaje de Modelado Unificado (UML- Unified Modeling Language) es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables (GARCÍA *et al*. 2004).

1.6.3. Lenguajes de programación.

CSS3: Es un lenguaje de estilo utilizado en la presentación de documentos HTML. Sirve para organizar la presentación y aspecto de una página web. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento con la de su presentación. La nueva versión CSS3 tiene a su favor muchas mejoras que disminuyen el trabajo innecesario de los desarrolladores, como por ejemplo alguna animación que se desee hacer a algún elemento HTML se hace con propiedades css, lo que antes era más complicado y debía ser implementarlo con JavaScript (GAUCHAT 2012).

JavaScript: Es un lenguaje de programación interpretado del lado del cliente, se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Implementado

como parte de un navegador web, permite mejoras en la interfaz de usuario y páginas web dinámicas. JavaScript fue desarrollado por la empresa La principal ventaja de este lenguaje es que permite rapidez y eficiencia en los sistemas ya que sirve para validar datos introducidos por el cliente sin necesidad de llegar al servidor a hacer la petición, para mostrar mensajes en el navegador. Cuenta con funcionalidades tan poderosas como las de otro lenguaje y posee una enorme cantidad de librerías que pueden facilitar el trabajo del desarrollador (GAUCHAT 2012).

PHP: Se empleó este lenguaje debido a que es un lenguaje de programación de uso general de script del lado del servidor, originalmente diseñado para el desarrollo web de contenido dinámico. Es además un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor (Qué es PHP?, 2013). Su característica más notable es la posibilidad de embeberse junto al código HTML. Presenta además las siguientes características:

- ✓ Es multiplataforma.
- ✓ Es compatible con casi la totalidad de los servidores Web.
- ✓ Es gratis.
- ✓ Es fácil de aprender y se ejecuta eficientemente en el servidor.
- ✓ Tiene cientos de funciones.

1.6.4. Lenguaje de consultas.

SQL: Es un lenguaje de consulta estructurado que brinda la posibilidad de realizar consultas con el objetivo de recuperar información de las bases de datos de manera sencilla. SQL es a la vez un lenguaje sencillo de comprender y una herramienta completa para la administración de los datos.

1.6.5. Gestor de Base de Datos.

PostgreSQL: Es un sistema de gestión de bases de datos relacional orientado a objetos y es libre, publicado bajo la licencia BSD que permite el uso de licencia de software no libre. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa o persona, sino que es dirigido por una comunidad de desarrolladores que trabaja de forma desinteresada y libre apoyados por organizaciones comerciales. Entre las bondades que brinda están:

- ✓ Ideal para tecnologías web.
- ✓ Fácil de administrar.
- ✓ Su sintaxis SQL es estándar y fácil de comprender.
- ✓ Multiplataforma.

- ✓ Capacidades de replicación de datos.
- ✓ Soporte empresarial disponible.
- ✓ Diseñado para ambientes de alto volumen.

1.6.6. Framework.

Doctrine: Doctrine es un mapeador de objetos relacionales (ORM- Object Relational Mapper) para PHP. Se sitúa en la parte superior de una potente capa de abstracción de bases de datos (DBAL- Database Abstraction Layer). Uno de sus elementos claves es la opción para escribir consultas a bases de datos en un lenguaje SQL orientado a objeto llamado DQL (Doctrine Query Language) (GALLARDO 2010).

Ext JS: Es una biblioteca o conjunto de librerías de JavaScript para el desarrollo de aplicaciones web interactivas. Permite realizar completas interfaces de usuario, fáciles de usar, muy parecidas a las conocidas aplicaciones de escritorio. Esto permite a los desarrolladores web concentrarse en la funcionalidad de las aplicaciones en vez de en las advertencias técnicas (MAYORI 2013).

Zend Framework: Está desarrollado en PHP con el afán de hacer el trabajo de los programadores más simple y amistoso. Se basa en la programación orientada a objetos pensando en hacer aplicaciones más seguras, modernas y robustas. Utiliza el patrón Modelo Vista Controlador (MVC). Un punto importante es que brinda un buen estándar de codificación para proyectos. Cuenta con soporte para internalización y localización de aplicaciones, además de una completa documentación y test de alta calidad (SOUTO 2010).

ZendExt: Es una extensión del framework Zend realizada por la UCID con el fin de adaptar algunos componentes del mismo a las características y particularidades del desarrollo de software en dicho centro. Las modificaciones llevadas a cabo se centran en los procesos de autenticación, autorización y gestión de las sesiones de usuarios. Además, se trabajó en la configuración, gestión de trazas, tratamiento de excepciones, transacción, notificaciones y validación (PROCESO DE DESARROLLO Y GESTION DE PROYECTOS DE SOFTWARE 2012).

Zeolides: Es un conjunto de librerías, herramientas, tecnologías y componentes de software integrados en un marco de trabajo para desarrollar aplicaciones de múltiples propósitos, gran tamaño y grandes volúmenes de datos, que permiten el desarrollo ágil, basado en componentes, centrado en los requerimientos del usuario, las interfaces de usuario y la lógica del negocio de las aplicaciones que con el mismo se desarrollen (HURTADO s.f.).

1.6.7. Herramientas de desarrollo.

PHP Storm: Es un editor de texto para programar en PHP: Resalta los nombres de las funciones y clases, identifica las variables, encuentra posibles errores, hace refactoring de PHP y tiene cientos de combinaciones de teclas que permiten programar sin casi tocar el mouse. Además, soporta JavaScript y HTML como los mejores editores. Otras características relevantes son que ayudan a depurar y a probar unidades.

Servidor Web Apache: Un servidor web es un programa informático que da la posibilidad de alquilar un espacio en un servidor para alojar nuestro sitio. La principal función de un servidor web es almacenar los archivos de un sitio y emitirlos por Internet para poder ser visitado por los usuarios. Cuando un usuario entra en una página de Internet su navegador se comunica con el servidor enviando y recibiendo datos que determinan qué es lo que ve en la pantalla. En el sistema se utilizó el servidor Apache que es muy robusto y estable, por tales motivos es uno de los servidores web más usado en la actualidad.

Wamp Server: El uso de un Wamp permite servir páginas web a internet, además de poder gestionar datos en ellas. Al mismo tiempo, proporciona lenguajes de programación para desarrollar aplicaciones web. El mismo también implementa el protocolo HTTP, el cual se encarga de transferir hipertextos, páginas web o páginas HTML, textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música.

1.6.8. Navegador Web.

Un navegador web es un software que permite el acceso a Internet, interpretando la información de archivos y sitios web para que éstos puedan ser leídos. Además, permite visitar páginas web y hacer actividades en ella, es decir, se puede enlazar un sitio con otro, imprimir, enviar y recibir correos, ver videos, fotos, imágenes, música y descargar ficheros y programas. Para la realización del sistema se utilizó el Mozilla Firefox y Google Chrome que cuentan con muchas ventajas tales como:

- ✓ Brindan la posibilidad de restaurar una sesión si se ha cerrado el navegador.
- ✓ Cuentan con corrector ortográfico y sugerencias de búsqueda integrada.
- ✓ Incluyen protección antiphishing que no permite enviar código desde el navegador a su consola, actualizaciones automáticas, protección contra programas espías y la posibilidad de limpiar la información privada.
- ✓ Son libres y se le pueden instalar muchos plugins.

- ✓ Para los desarrolladores web son una herramienta imprescindible ya que de una manera u otra facilita el trabajo a la hora de detectar algún error de diseño o de lógica.

1.6.9. Patrones utilizados para el desarrollo del sistema.

Durante el desarrollo del sistema se utilizaron varios patrones de diseño con el objetivo de cumplir con los principios o reglas de diseño. Un patrón es una descripción del problema y la esencia de su solución, que se puede reutilizar en casos distintos. Estos ayudan a estandarizar el código, lo que hace que el diseño sea más comprensible para otros programadores.

El diseño de la solución propuesta para el módulo Auditoría Interna del SAPGAE se rige por el empleo de diferentes patrones implementados en el marco de trabajo Zeolides.

1.6.9.1. Patrones de Acceso a Datos.

Estos patrones brindan diferentes posibilidades de estrategias de acceso a las bases de datos. El acceso a los datos es un elemento importante y muy delicado en los proyectos. El uso de estos reduce el tiempo de desarrollo y mejora el mantenimiento de la capa de datos después de estar desplegado (HURTADO s.f.).

Patrón *Active Record* (Uso de Doctrine_Record): Se puede entender como un objeto que no transporta solamente datos sino también el comportamiento, es decir deposita la lógica sobre su persistencia.

Patrón *Query Object* (Uso de Doctrine_Query): Un objeto que representa una consulta de base de datos que será utilizado para los SQL dinámicos.

Patrón *Repository* (Uso de Doctrine_Record y Doctrine_Table): Un repositorio realiza las tareas de intermediario entre las capas de modelo de dominio y mapeo de datos, y actúa de forma similar a una colección en memoria de objetos del dominio. Conceptualmente un repositorio encapsula a un conjunto de objetos almacenados en la base de datos y las operaciones que sobre ellos pueden realizarse, y proveen de una forma más cercana a la orientación, a objetos de la vista de la capa de persistencia.

Unidad de Trabajo (Uso de Doctrine_Collection): Grafo de objetos creados o manipulados, que almacena coherentemente los datos.

Patrón Data Mapper (ORM): Tiene como objetivo separar las estructuras de los objetos, las estructuras de los modelos relacionales y realizar la transferencia de datos entre ambos. Cada tabla en la base de datos se representa en una clase y un objeto de esa clase representa una fila en la tabla. Se utiliza para evitar tipos de ataques como pueden ser las inyecciones SQL y facilita el mantenimiento

del código debido a la correcta ordenación de la capa de datos, permitiendo que el mantenimiento del código sea mucho más sencillo. La aplicación gana en abstracción y reutilización de código.

Patrón *Foreign Key Mapping*: Establece las relaciones entre objetos.

1.6.9.2. Patrones de diseño seguro.

***Validador-Interceptor**:* Garantiza que los datos provenientes de cualquier entidad externa cumplan con un conjunto de reglas de sintaxis, tipo de dato, longitud y negocio

***Escape de las salidas**:* Evita que caracteres o información de control presentes en los datos, tengan un significado especial para el intérprete hacia el que van dirigidos.

***Punto único acceso**:* Define un punto único de acceso hacia la aplicación donde sea posible aplicar diferentes medidas de seguridad, con el objetivo de ejercer un mejor control de las comunicaciones hacia y desde la misma.

Web SSO (Web *Single Sign On*): Tiene como objetivo garantizar que los usuarios de cualquier organización solo tengan que autenticarse una única vez para acceder a los diferentes servicios de la misma evitándoles memorizar varias cuentas de usuarios y contraseñas.

Role Based Access Control (RBAC): Es un enfoque basado en el concepto de rol para implementar políticas de control de acceso y su intención es lograr un control de acceso más granular, donde las asignaciones usuario-rol y permiso-rol permiten que un usuario pertenezca a varios roles simultáneamente y que un rol posea múltiples usuarios.

1.6.9.3. Patrones de diseño y estilos arquitectónicos de Zeolides.

Abstract Factory (Fábrica Abstracta): Permite trabajar con objetos de distintas familias de manera que estas no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando. El problema a solucionar por este patrón es el de crear diferentes familias de objetos, como por ejemplo la creación de interfaces gráficas de distintos tipos (ventana, menú, botón, etc.).

Singleton (Instancia Única): Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Restringe la instanciación de una clase o valor de un tipo a un solo objeto.

Observer (Observador): Define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él.

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.

Front-Controller**:** Obliga a que todas las peticiones hechas a la aplicación pasen por un controlador. El controlador proporciona un punto de entrada único que controla y gestiona las peticiones web realizadas por los clientes.

Facade (Fachada): Provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema.

Comando**:** Este patrón permite solicitar una operación a un objeto sin conocer realmente el contenido de esta operación, ni el receptor real de la misma. Para ello se encapsula la petición como un objeto, con lo que además se facilita la parametrización de los métodos.

1.6.10. Arquitectura de Software.

La arquitectura de software define la estructura a grandes rasgos y la organización fundamental de un sistema que se representa mediante componentes, definiendo las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución.

Zeolides usa un estilo arquitectónico híbrido, combina el estilo arquitectónico N-Capas y el MVC, tomando lo mejor de los dos, divide la funcionalidad de una aplicación en 3 componentes fundamentales: modelo, vista y controlador y a su vez divide la arquitectura de la aplicación en capas: presentación, negocio, acceso a datos y datos. Otro elemento de su estilo arquitectónico es la incorporación de requerimientos no funcionales a las aplicaciones sin necesidad de modificar el código de la aplicación, utilizando los principios de la programación orientada a aspectos (AOP). Estos requerimientos se denominan aspectos arquitectónicos. Además, se tuvo en cuenta el uso de inversión de control (IoC) para separar las responsabilidades de una aplicación en componentes, integrando estos a través de contratos bien definidos sin necesidad de instanciar directamente sus clases o conocer sus implementaciones, esto nos permite sentar las bases para el desarrollo de software basado en componentes (HURTADO, s.f.).

1.7. Conclusiones.

El desarrollo del capítulo dio a conocer las bases teóricas sobre las cuales se sustenta la propuesta de esta investigación. Se analizó como se efectúa el proceso de auditoría y se

corroboró la necesidad de diseñar el sistema para solucionar los problemas existentes, al tomar como base que el software analizado no satisfacen las necesidades demandadas. Además se definieron las tecnologías y herramientas que se aplicarán en el diseño e implementación del software.

Capítulo 2. Análisis y diseño de la solución propuesta

2.1. Introducción.

En este capítulo se detallan los elementos esenciales del sistema en el modelo de dominio. Se describen las características que deberá cumplir el sistema según la metodología de desarrollo de software PRODESOFT, correspondientes a la fase de modelación y se mencionan los requisitos funcionales y no funcionales.

2.2. Modelo de dominio.

Un modelo de dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción. La tarea de construcción del modelo de dominio se presenta como uno o más diagramas de clases y contiene, no conceptos propios de un sistema de software sino de la propia realidad física. Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema (MILIÁN 2011). La figura 2 muestra el modelo de dominio del módulo de Auditoría interna.

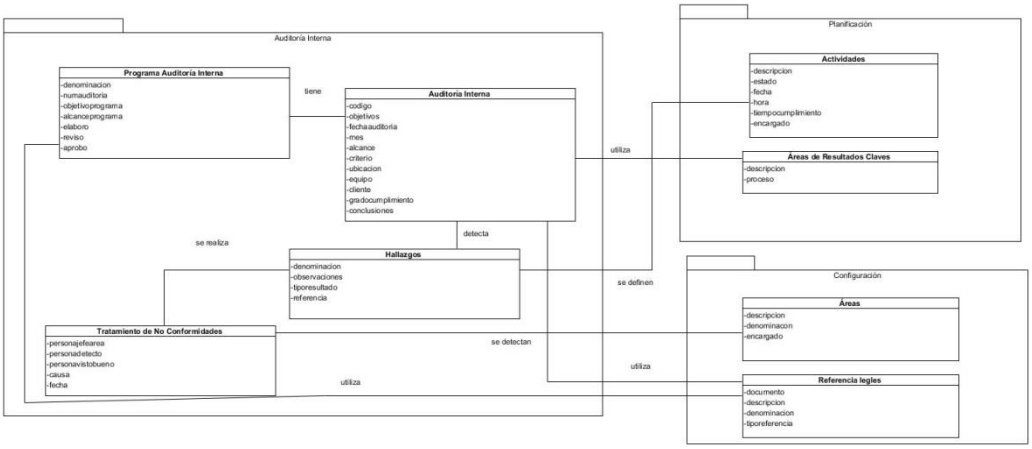


Figura 2. Modelo de dominio del módulo de Auditoría Interna.

2.2.1. Reglas del dominio.

En este epígrafe se recogen, a través de las reglas de dominio, las condiciones necesarias para que el sistema pueda funcionar correctamente y se conviertan en futuras validaciones del sistema que se pretende realizar.

R1. El programa de auditoría específica los detalles acordados para un conjunto de auditorías.

R2. Cada programa de auditoría interna debe contener el control de las auditorías internas ambientales que se van a desarrollar en él.

R3. Cada equipo auditor puede tener varios auditores que desempeñan determinados roles en equipos distintos.

R4. En un equipo pueden existir auditores que desempeñan el mismo rol excepto el auditor líder que es único.

R5. Una auditoría interna se les realiza a diversos procesos en la empresa.

R6. En los procesos se detectan hallazgos basándose en los criterios de auditorías.

R7. Los hallazgos pueden indicar conformidad o no conformidad y conducir a la identificación de oportunidades de mejora y buenas prácticas.

R8. Los criterios de auditoría contienen un conjunto de políticas, objetivos, procedimientos, normas, requisitos legales, requisitos de gestión, código de conducta sectoriales u otro acuerdo planificado aplicable.

2.3. Definición de los requisitos.

Los requisitos para un sistema son la descripción de los servicios proporcionados y sus restricciones operativas. Estos requisitos reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema. El proceso de descubrir, analizar, documentar y verificar las funcionalidades del software se denomina Ingeniería de Requisitos (SOMMERVILLE 2005). Los requisitos que se obtienen en este proceso pueden ser funcionales o no funcionales.

2.3.1. Requisitos funcionales.

Para describir las funcionalidades del sistema se definen los siguientes requisitos funcionales:

Módulo Auditoría Interna

- ✓ **Programa auditoría interna**

  R1. Adicionar programa

  R2. Modificar programa

  R3. Eliminar programa

  R4. Buscar programa

  R5. Obtener reporte del programa de auditorías internas

R6. Detalles del programa

- ✓ **Equipos auditores**
  - R1. Adicionar equipo
  - R2. Eliminar equipo
  - R3. Adicionar persona
  - R4. Eliminar persona
- ✓ **Gestionar auditoría interna**
  - R1. Adicionar auditoría interna
  - R2. Modificar auditoría interna
  - R3. Buscar auditoría interna
  - R4. Resultados
  - R5. Imprimir auditoría interna
  - R6. Graficar auditoría
- ✓ **Gestionar Procesos**
  - R1. Adicionar proceso
  - R2. Eliminar proceso
  - R3. Buscar proceso
- ✓ **Gestionar hallazgos**
  - R1. Adicionar hallazgos
  - R2. Modificar hallazgos
  - R3. Eliminar hallazgos
  - R4. Buscar hallazgos
- ✓ **Gestionar medidas**
  - R1. Adicionar medidas
  - R2. Eliminar medidas
- ✓ **Tratamiento de no conformidades**
  - R1. Detalles de no conformidad
  - R2. Buscar no conformidad
  - ✓ R3. Obtener reporte del tratamiento de no conformidad

- ✓ **Seguimiento de no conformidades**

  R1. Adicionar seguimiento de no conformidades

  R2. Modificar seguimiento de no conformidades

  R3. Eliminar seguimiento de no conformidades

Se realizaron especificaciones para cada uno de los requisitos del software (ERS), las que se encuentran documentadas en el expediente del proyecto. A continuación, se muestran las especificaciones de la funcionalidad Gestionar auditoría interna, la que permite gestionar las auditorías, posibilitando listar, adicionar, modificar, eliminar, imprimir, asociar resultados, graficar y buscar. Además, se puede consultar la ayuda detallada de la misma. En las tablas siguientes se describen las especificaciones de los requisitos correspondientes a la funcionalidad Gestionar auditoría interna. Se muestra además en la figura 3 la interfaz de usuario (IU) de esta funcionalidad.

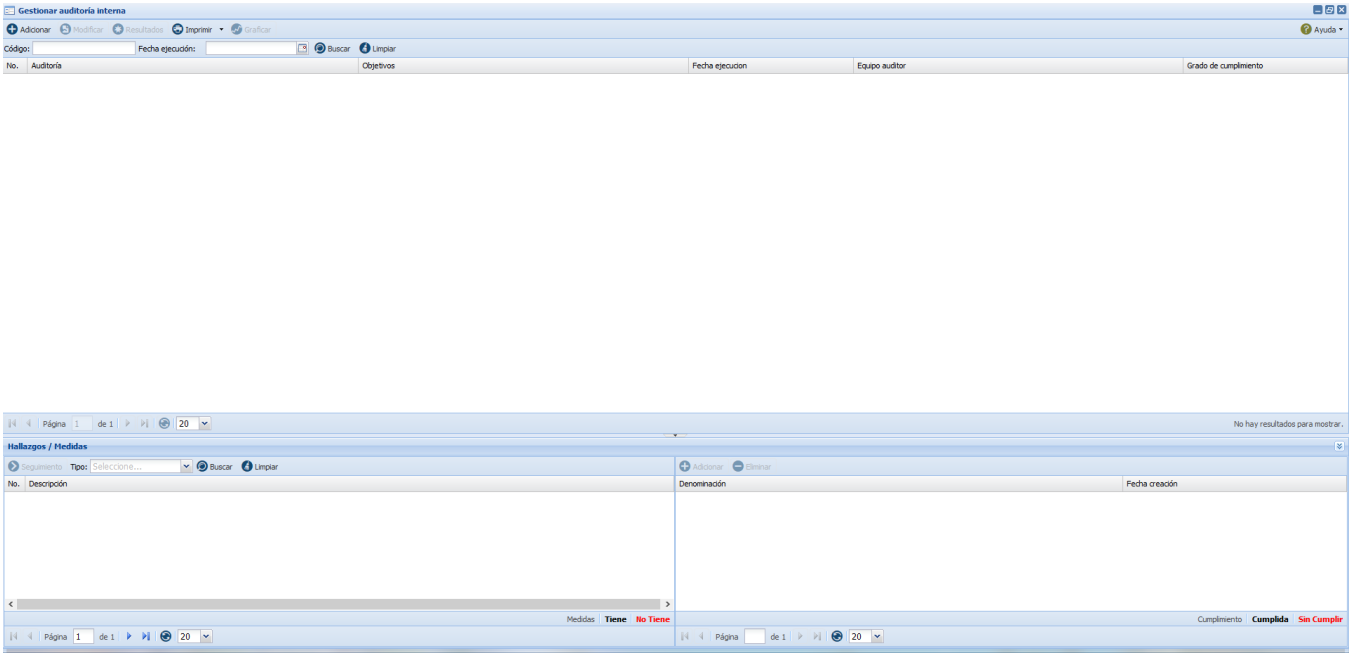


Figura 3. Prototipo de IU Gestionar auditoría interna.

Tabla 1. Especificación del requisito Adicionar auditoría interna.

Conceptos tratados	**Conceptos**	**Atributos**
	Auditoría interna	código, objetivos, equipo, alcance, procesos, criterios, fecha de ejecución, cliente, ubicación
Precondiciones	**Precondiciones**	**Pre-requisito**
	El usuario se ha autenticado.	Autenticar usuario.
Descripción	1. Se accede mediante el menú **Inicio/ SAPGAE Auditoría Interna/ Gestionar auditoría interna.** 2. Se muestra la interfaz Gestionar auditoría interna (Figura 3). 2.1. Si se selecciona la opción **Ayuda**, el sistema muestra la ayuda detallada de la funcionalidad. 3. Se selecciona la opción **Adicionar**. 4. Se muestra la interfaz Adicionar auditorías internas (Figura 4) con los siguientes campos: • Código: Se define el código de la auditoría. • Objetivos: Se introduce el/los objetivo(s) de la auditoría interna. • Equipo: Se gestiona el equipo auditor que va a realizar la auditoría. • Alcance: Se introduce el alcance de la auditoría interna. • Proceso: Se seleccionan los procesos asociados a la auditoría o para adicionar uno nuevo (Ver ERS Gestionar Proceso que se encuentra en el Expediente del Proyecto SAPGAE, Auditoría Interna). • Criterios: Se gestiona el/los criterios que se van a utilizar para realizar la auditoría. • Fecha de ejecución: Se selecciona la fecha de ejecución. • Cliente: Se introduce el cliente al que se le realiza la auditoría. • Ubicación: Se introduce la ubicación donde se realiza la auditoría. 4.1. Si se marca el checkbox Auditoría interna programada (Figura 5): 4.1.1. Se selecciona el programa donde se encuentra la auditoría que se	

	desea gestionar. 4.1.2. Se selecciona el código de la auditoría y los campos se cargarán automáticamente con los datos ya definidos en el Programa y se debe adicionar la fecha de ejecución, cliente y ubicación. 5. Si se selecciona la opción Aceptar Se validan los datos, se guardan y se muestra un mensaje de información: "La auditoría ha sido adicionada satisfactoriamente" y se cierra la interfaz. 5.1. Si se selecciona la opción **Aplicar**, el sistema valida los datos, se guardan, se limpian los campos y se muestra un mensaje de información: *"La auditoría ha sido adicionada satisfactoriamente"* y se mantiene la interfaz permitiendo adicionar una nueva auditoría. 5.2. Si se selecciona la opción Cancelar, el sistema no adiciona la auditoria y se cierra la ventana actual. 6. Se muestran los datos adicionados en el grid.
Validaciones	• El sistema valida los datos según lo descrito en el documento Modelo Conceptual. • Si se introducen datos incorrectos o se dejan campos obligatorios vacíos, se muestra un mensaje de error de color rojo en cada uno de estos. • Si se introduce una auditoría que ya existe, el sistema muestra un mensaje de error: "*La auditoría que quiere adicionar ya existe.*" • Los botones **Aceptar** y **Aplicar** se habilitarán solo cuando se hayan llenado todos los campos obligatorios.
Complejidad	Alta
Prioridad	Alta
Post-condiciones	Se ha adicionado una auditoría
Post-requisito	No procede

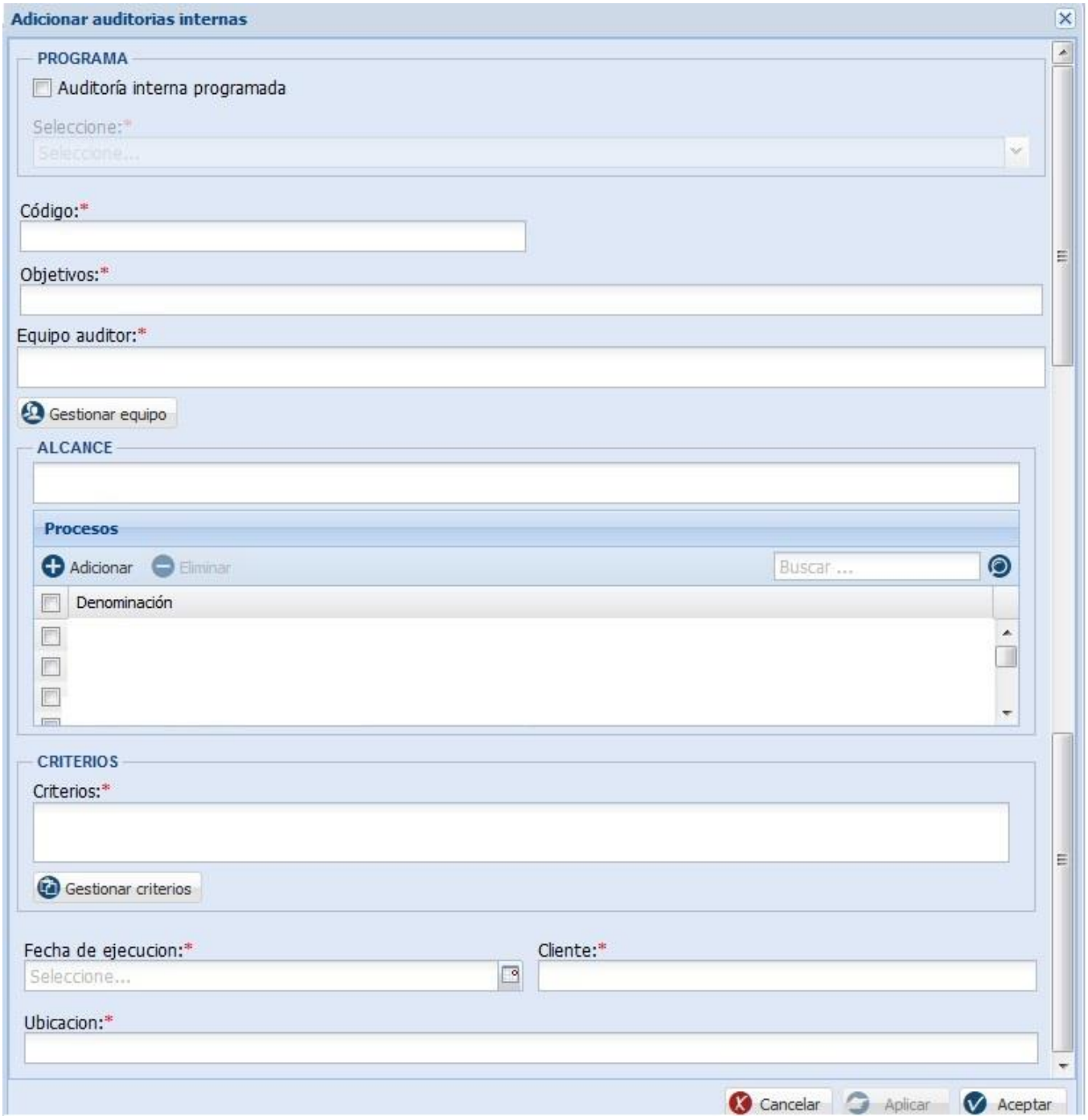


Figura 4. Prototipo de IU Adicionar auditorías internas.

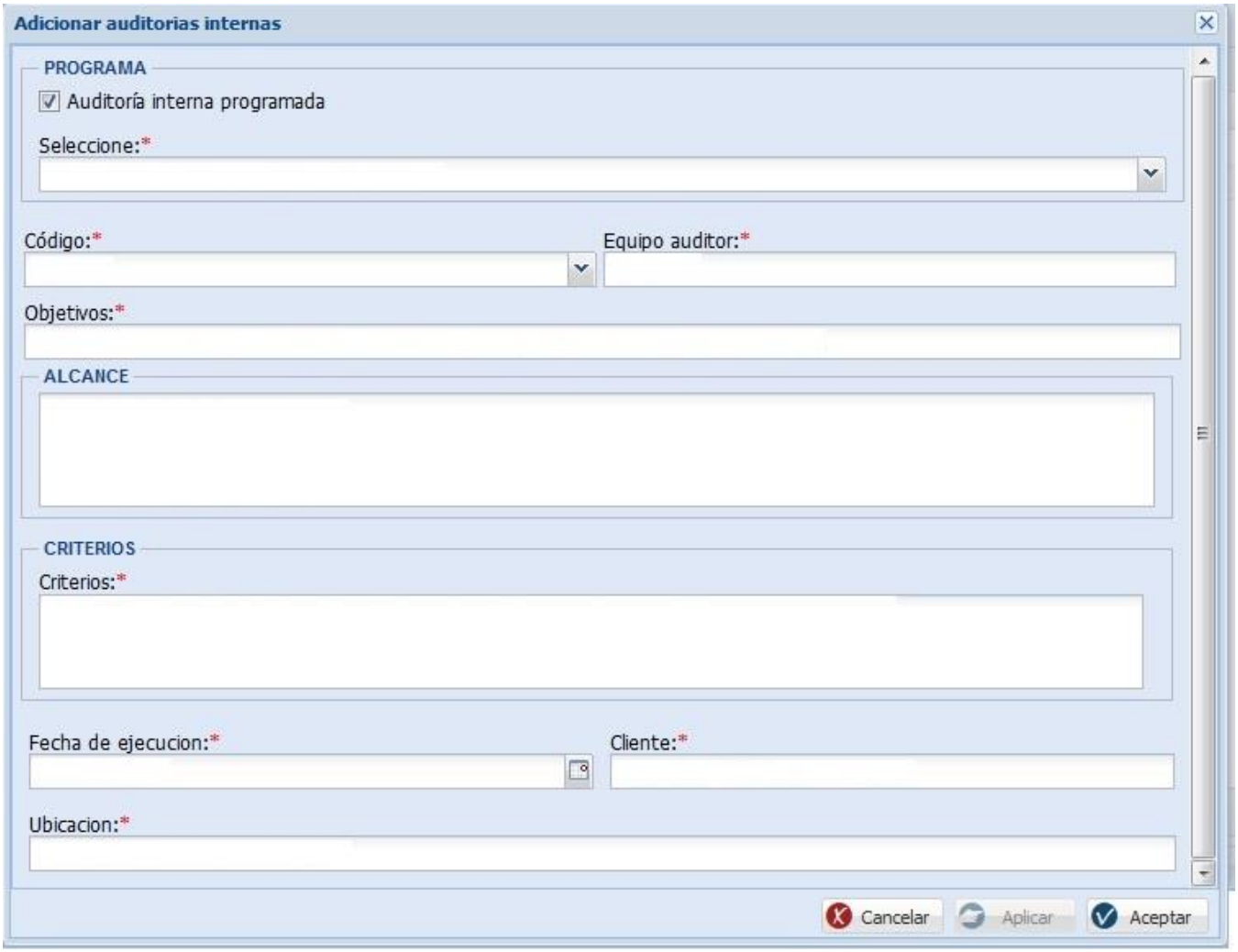


Figura 5. Prototipo de IU Adicionar auditoría programada.

Tabla 2. Especificación del requisito Modificar auditoría interna.

Conceptos tratados	**Conceptos**	**Atributos**
	Auditoría interna	código, objetivos, equipo, alcance, procesos, criterios, fecha de ejecución, cliente, ubicación
Precondiciones	**Precondiciones**	**Pre-requisito**
	Autenticar usuario.	El usuario se ha autenticado.
	Se ha adicionado una auditoría.	Adicionar la auditoría.

Descripción	1. Se accede mediante el menú **SAPGAE Auditoría Interna/ Gestionar auditoría interna.** 2. Se muestra la interfaz Gestionar auditoría interna (Figura 3). 3. Se listan las auditorías registradas hasta el momento. 3.1 Si se selecciona la opción **Ayuda**, el sistema muestra la ayuda detallada de la funcionalidad. 4. Se selecciona la auditoría que desea modificar. 5. Se selecciona la opción **Modificar**. 6. Se muestra la interfaz Modificar auditoría (Figura 6). 7. Se modifican los datos que desee sean permitidos. 8. Se selecciona la opción **Aceptar.** 8.1 Si se selecciona la opción Cancelar, el sistema no modifica los datos de la entidad y se cierra la ventana actual. 9. Se validan los datos, se guardan y se muestra un mensaje de información: *"La auditoría ha sido modificada correctamente"* y se cierra la interfaz. 10. Se muestran los datos actualizados en el grid.
Validaciones	• El botón **Modificar** se habilitará cuando se seleccione una auditoría en el grid. • El sistema valida los datos según lo descrito en el documento Modelo Conceptual. • Si se introducen datos incorrectos o se dejan campos obligatorios vacíos, se muestra un mensaje de error de color rojo en cada uno de estos.
Complejidad	Alta
Prioridad	Alta
Post-condiciones	Se han modificado los datos de la auditoría
Post-requisito	No procede

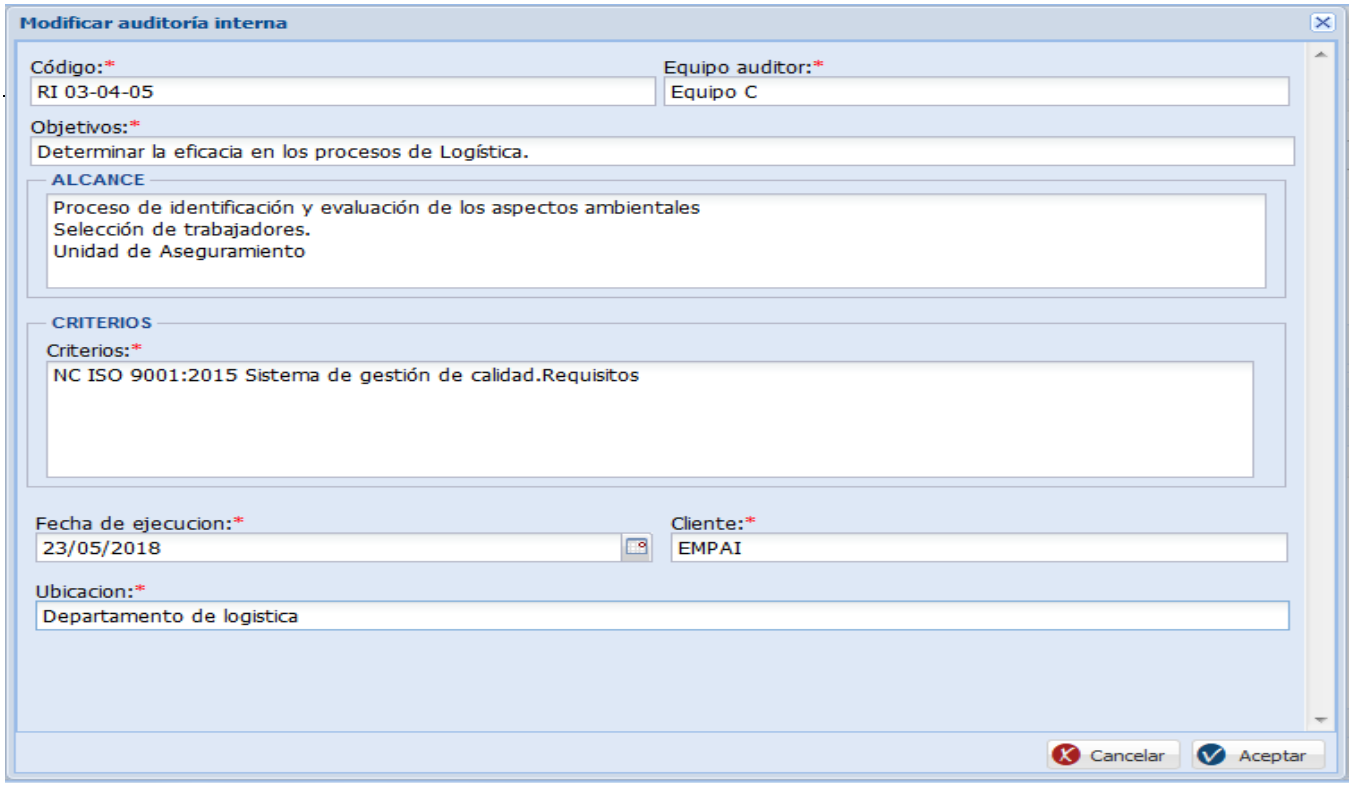


Figura 6. Prototipo de IU Modificar auditoría interna.

Tabla 3. Especificación del requisito Buscar auditoría interna.

Conceptos tratados	**Conceptos**	**Atributos**
	Auditoría interna	código, objetivos, equipo, alcance, procesos, criterios, fecha de ejecución, cliente, ubicación
Precondiciones	**Precondiciones**	**Pre-requisito**
	Autenticar usuario.	El usuario se ha autenticado.
	Se ha adicionado al menos una auditoría.	Adicionar auditoría.
Descripción	1. Se accede mediante el menú **SAPGAE Auditoría Interna/ Gestionar auditoría interna.** 2. Se muestra la interfaz Gestionar auditoría interna (Figura 3). 3. Si se selecciona la opción **Ayuda**, el sistema muestra la ayuda detallada de la funcionalidad.	

	4. Se especifican los criterios de búsqueda. 5. Si se presiona el botón **Buscar,** se mostrarán los datos que coincidan con el parámetro de búsqueda. 6. Si se selecciona la opción Limpiar, el sistema limpia los campos. 7. Se muestra en el grid el resultado de la búsqueda en caso de que exista.
Validaciones	• Si se dejan los campos vacíos y se presiona el botón Buscar, se listarán todas las auditorías registrados en el sistema.
Complejidad	Baja
Prioridad	Alta
Post-condiciones	Se ha realizado la búsqueda.
Post-requisito	No procede

Tabla 4. Especificación del requisito Resultados.

Conceptos tratados	**Conceptos**	**Atributos**
	Auditoría interna y Resultados	proceso, hallazgos, conclusiones, grado de cumplimiento
Precondiciones	**Precondiciones**	**Pre-requisito**
	Autenticar usuario.	El usuario se ha autenticado.
	Se ha adicionado al menos una auditoría.	Adicionar auditoría.
Descripción	1. Se accede mediante el menú **SAPGAE Auditoría Interna/ Gestionar auditoría interna** 2. Se muestra la interfaz Gestionar auditoría interna (Figura 3). 3. Se muestra el listado de auditorías registradas hasta el momento. 3.1. Si se selecciona la opción Ayuda, el sistema muestra la ayuda	

	detallada de la funcionalidad. 4. Se selecciona la auditoría a la que se desea asignarle los resultados. 5. Se selecciona la opción **Resultados**. 6. Se muestra la interfaz Resultados (Figura 7) con el listado de hallazgos que ya han sido adicionados a través de la funcionalidad Gestionar hallazgos (Los hallazgos que se muestran en rojo son los que no tiene medidas y los que están en negro son los que tienen). 6.1. Si desea gestionar otro hallazgo ver ERS Gestionar hallazgos que se encuentra en el Expediente del Proyecto SAPGAE, Auditoría interna. 7. Se selecciona el proceso. 8. Se selecciona en el grid el hallazgo que se desea asignar. 9. Se selecciona el botón con el icono poner botón más.  10. El sistema muestra después de haber adicionado los datos necesarios en la ventana de Hallazgo (Ver ERF Hallazgo en el expediente SAPGAE) el hallazgo en el grid de Seleccionados y muestra el mensaje de información *"El hallazgo ha sido adicionado satisfactoriamente".* 10.1. Si desea eliminar el hallazgo del grid de Seleccionados, seleccionar el botón con el icono poner botón menos.  10.2. El sistema elimina el hallazgo del grid y muestra el mensaje de información *"El hallazgo ha sido eliminado satisfactoriamente".* 11. Se introducen los datos de las conclusiones y el grado de cumplimiento. 12. Se presiona el botón Cerrar. 13. Se muestra el grado de cumplimiento en el grid de auditoría y los hallazgos en el panel Hallazgos/Medidas.
Validaciones	• Los procesos cargados en el combobox son los que fueron seleccionados cuando se adicionó en la auditoría seleccionada.
Complejidad	Alta
Prioridad	Alta

Post-condiciones	Se ha asociado los resultados a la auditoría.
Post-requisito	No procede

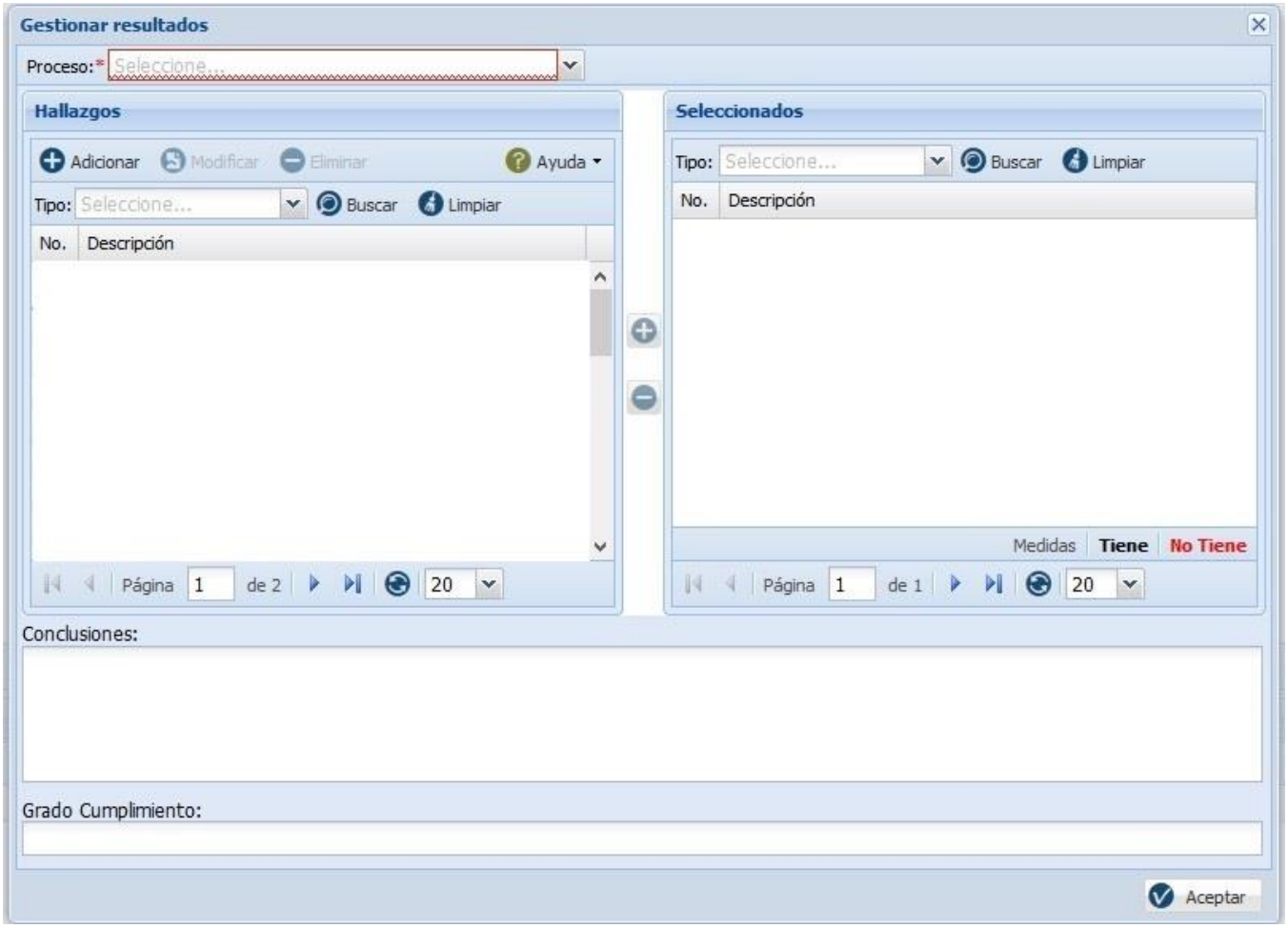


Figura 7. Prototipo de IU Resultados.

Tabla 5. Especificación del requisito Graficar auditoría.

Conceptos tratados	**Conceptos**	**Atributos**
	Auditoría interna	procesos, resultados
Precondiciones	**Precondiciones**	**Pre-requisito**
	Autenticar usuario.	El usuario se ha autenticado.

	Se ha adicionado al menos una auditoría.	Adicionar auditoría.
Descripción	1. Se accede mediante el menú **SAPGAE Auditoría Interna/ Gestionar auditoría interna.** 2. Se muestra la interfaz Gestionar auditoría interna (Figura 3). 3. Se listan las auditorías registradas hasta el momento. 3.1 Si se selecciona la opción **Ayuda**, el sistema muestra la ayuda detallada de la funcionalidad. 4. Se selecciona la auditoría de la cual se desea graficar sus procesos respecto a sus hallazgos. 5. Se selecciona la opción **Graficar**. 6. El sistema muestra una gráfica que relaciona los procesos con los hallazgos (no conformidad y oportunidad de mejora) obtenidos en esa auditoría (Figura 8).	
Validaciones	• El botón Graficar se habilitará cuando se seleccione una auditoría en el grid.	
Complejidad	Alta	
Prioridad	Alta	
Post-condiciones	Se ha realizado la búsqueda.	
Post-requisito	No procede	

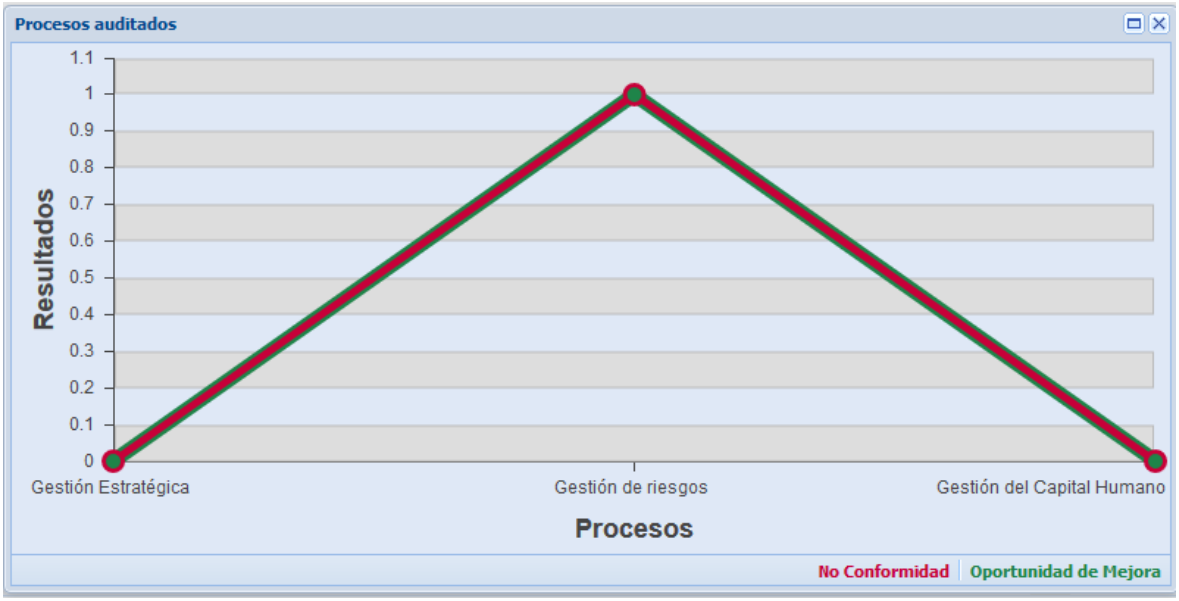


Figura 8. Prototipo de IU Graficar.

Tabla 6. Especificación del requisito Imprimir auditoría interna.

Conceptos tratados	**Conceptos**	**Atributos**
	Auditoría interna	
Precondiciones	**Precondiciones**	**Pre-requisito**
	Autenticar usuario.	El usuario se ha autenticado.
	Se ha adicionado al menos una auditoría.	Adicionar auditoría.
Descripción	1. Se accede mediante el menú **SAPGAE Auditoría Interna/ Gestionar auditoría interna** 2. Se muestra la interfaz Gestionar auditoría (Figura 3). 3. Se listan las auditorías registradas hasta el momento. 3.1 Si se selecciona la opción **Ayuda**, el sistema muestra la ayuda detallada de la funcionalidad. 4. Se selecciona la opción **Imprimir**.	

	4.1. Se despliega un listado de los diferentes tipos de reportes que se pueden obtener. 4.1.1. Si se selecciona el Informe de auditoría interna el sistema muestra un reporte en formato pdf, solo con los datos de la auditoría (Ver Anexo # 2). 4.1.2. Si se selecciona el Resumen el sistema muestra un reporte en formato pdf, solo con los datos de la auditoría (Ver Anexo # 3). 4.1.3. Si se selecciona el Tratamiento de no conformidades el sistema muestra un reporte en formato pdf, solo con los datos de la auditoría (Ver Anexo # 4).
Validaciones	
Complejidad	Media
Prioridad	Alta
Post-condiciones	Se ha realizado la búsqueda.
Post-requisito	No procede

2.3.2. Requisitos no funcionales.

- ✓ Usabilidad: El sistema podrá ser usado por personas con conocimientos básicos en el manejo de computadoras. Se emplearán máscaras de progreso para indicar el estado de los procesos que por su complejidad requieran de un tiempo de procesamiento apreciable por los usuarios. El software tendrá siempre visible la opción de Ayuda, lo que posibilitará un mejor aprovechamiento por parte de los usuarios de sus funcionalidades.
- ✓ Confiabilidad: Deben establecerse los mecanismos necesarios para el restablecimiento del sistema ante fallos de comunicación u otros. Deben montarse sistemas de respaldo eléctrico en los locales de los servidores para mantener la vitalidad de los servicios.
- ✓ Rendimiento: Teniendo en cuenta que el producto se debe diseñar sobre una arquitectura cliente-servidor, los tiempos de respuestas del sistema deben ser rápidos, al igual que la velocidad de procesamiento de la información para lograr respuestas rápidas del mismo.
- ✓ Soporte: Se utilizará el servidor Web Apache para el trabajo con PHP, se debe utilizar un sistema gestor de base de datos que soporte grandes volúmenes de información, por lo que

se utilizará PostgreSQL. Debe elaborarse un paquete de instalación que abarque verificación de componentes ya instalados y la instalación de los nuevos.

- ✓ Restricciones de diseño: El producto de software final debe diseñarse sobre una arquitectura cliente-servidor. Se deben emplear los estándares establecidos (diseño de interfaces, base de datos y codificación). Emplear PHP como lenguaje del lado del servidor y del lado del cliente JavaScript, HTML y CSS.
- ✓ Requerimiento de Ayuda y Documentación: El sistema debe contar con una ayuda general en la página principal, que guíe al usuario a trabajar en el sistema. También estará disponible en cada una de las interfaces, de manera que los usuarios tengan conocimiento de las funcionalidades del mismo y logren hacer un mejor uso de ellas.
- ✓ Interfaz: El sistema debe contar con una interfaz fácil de usar, sencilla, amigable, permitiendo que los usuarios sean capaces de interactuar con la aplicación aun teniendo conocimientos básicos de informática. Será diseñada para adaptarse a la resolución del usuario, utilizando colores refrescantes, agradables y se emplearán imágenes identificadas con el negocio donde se utilizará el sistema.
- ✓ Portabilidad: El sistema será un sistema multiplataforma por lo que deberá ser compatible con los sistemas operativos Windows y Linux.
- ✓ Políticos Culturales: El sistema solo podrá ser usado en territorio cubano. Sólo debe contener palabras en idioma español. Debe respetar los términos usados en la entidad.
- ✓ Legales: El sistema se debe ajustar y regir por las leyes, decretos, resoluciones y manuales establecidos, que norman los procesos que serán automatizados.
- ✓ Seguridad: El usuario debe autenticarse antes de entrar al sistema.
- ✓ Confiabilidad: La información que se maneje en el sistema estará protegida de acceso no autorizado y divulgación, a partir de los diferentes roles de los usuarios que empleen el sistema.
- ✓ Integridad: La información manejada por el sistema será protegida contra corrupción y estados inconsistentes, de igual manera el origen y autoridad de los datos.
- ✓ Disponibilidad: La información se encontrará disponible en todo momento para aquellos usuarios autorizados a acceder al sistema.
- ✓ Software: El cliente deberá tener instalado el navegador Mozilla Firefox con el sistema operativo Windows XP o superior y un servidor Web Apache v2.0 o superior con módulo PHP5 disponible, debe estar configurado con las extensiones PDO, PDO_pgsql, pgsql y soap, teniendo como gestor de base de datos PostgreSQL.

- ✓ Hardware: El cliente debe tener tarjeta de red instalada en la computadora, impresora y Procesador Pentium III a 133GHz con 256 Mb de memoria RAM o superior. El servidor constará con al menos 40 Gb de espacio libre en disco duro y un procesador Pentium IV a 3.0 GHz y 1 Gb de memoria RAM o superior.
- ✓ Aplicación de Estándares: Se utilizarán los estándares de codificación, de diseño para la base de datos y mecanismos de diseño definidos por la entidad.

2.4. Diseño de la Base de datos.

El diseño de la base de datos es de crucial importancia pues tiene como propósito asegurar que los datos persistentes sean almacenados consistente y eficientemente. En el diseño de la base de datos los modelos de datos tienen un papel significativo, pues se encargan de proveer una vista de las entidades lógicas de datos y sus relaciones, si los modelos no son definidos apropiadamente, se puede tener muchos problemas en el momento de ejecutar consultas a la base de datos. El modelo físico de la base de datos del módulo Auditoría Interna del SAPGAE se muestra en la figura 9.

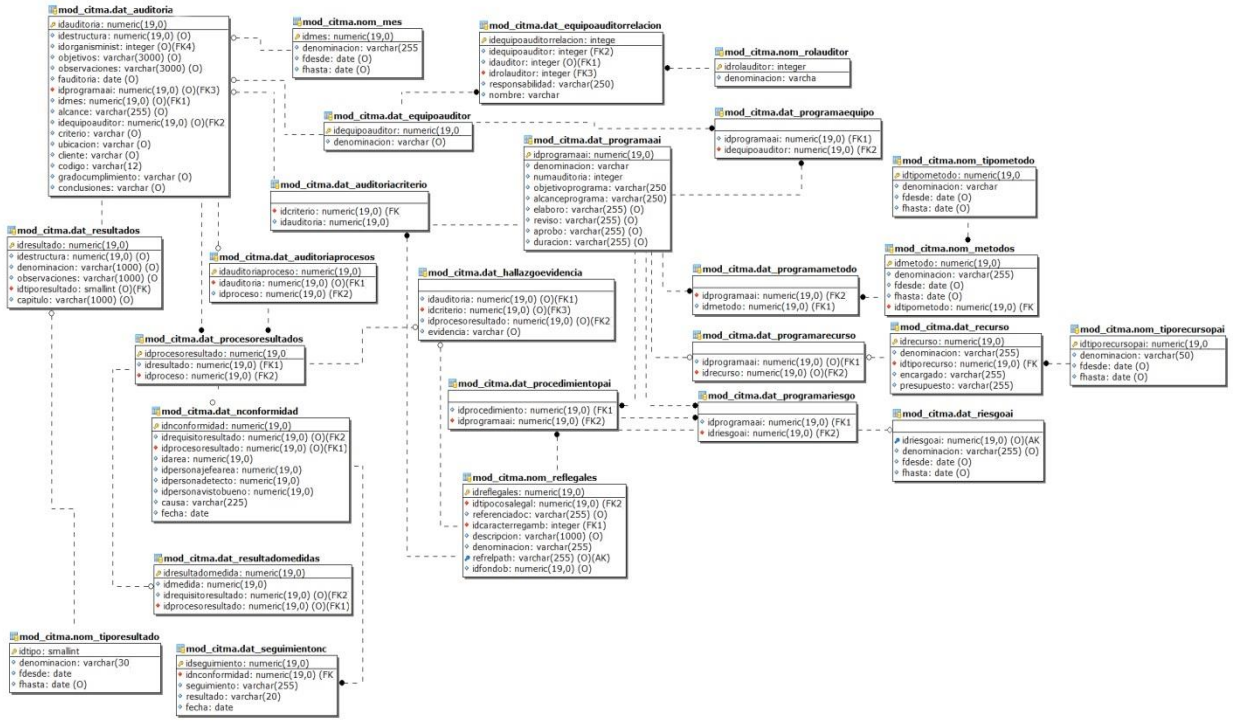


Figura 9. Modelo físico de la base de datos.

2.5. Diagrama de componentes.

Un componente es la parte modular de un sistema, desplegable y reemplazable que encapsula implementación y un conjunto de interfaces, además proporciona la realización de los mismos. Visto de otro modo un diagrama de componentes puede ser un tipo especial de diagrama de clases que se centra en los componentes físicos del sistema. El marco de trabajo Zeolides está basado en componentes, estos pueden ser horizontales y verticales.

Los componentes horizontales o de gestión común facilitan los servicios que deben realizar la mayoría de las aplicaciones, ejemplo de estos son el Portal de aplicaciones, el Subsistema de seguridad y dentro el Subsistema de gestión y monitoreo de trazas, además del Gestor de ayuda de sistemas y los componentes verticales, los que se construyen para el desarrollo del sistema. En la figura 10 se muestra el diagrama de componentes específico del sistema y en particular del módulo Auditoría interna.

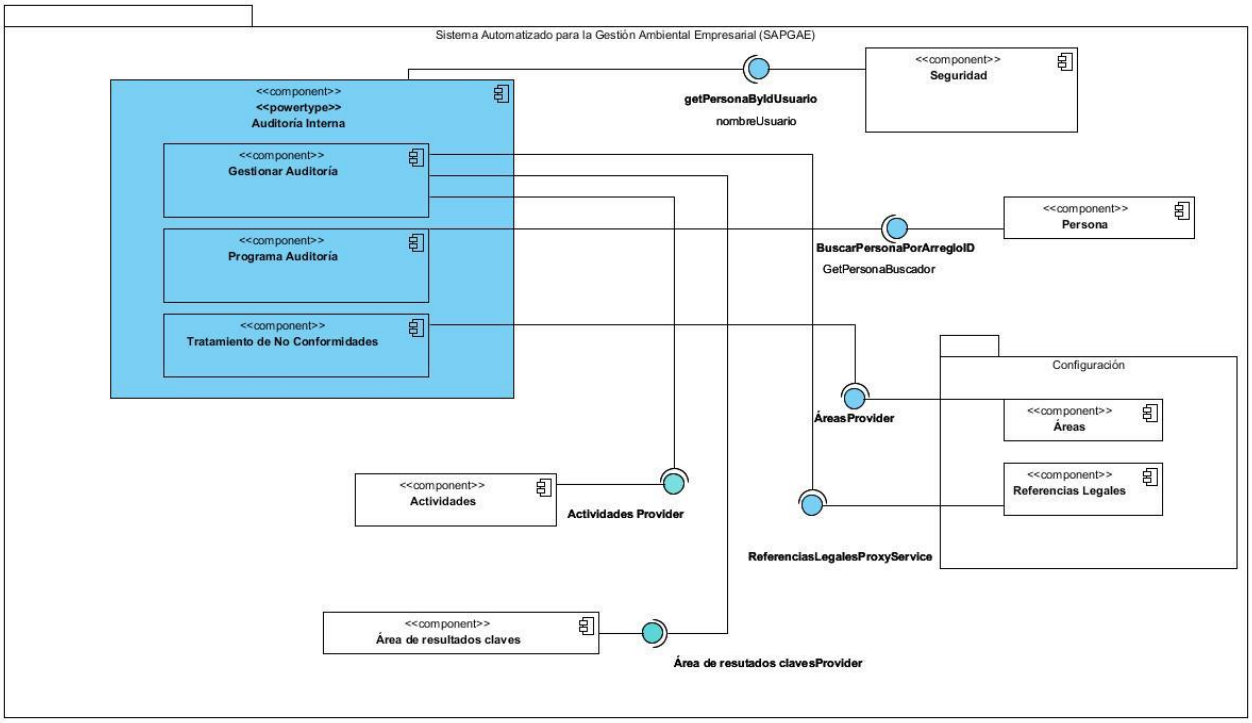


Figura 10. Diagrama de componentes específicos.

2.6. Mecanismo de diseño.

El mecanismo de diseño es un artefacto que agrupa un conjunto de clases de diseño, colaboraciones, e incluso subsistemas del modelo de diseño que llevan a cabo requisitos comunes como persistencia, distribución, seguridad, y funcionamiento. Con el objetivo de minimizar el trabajo y hacerlo de forma más eficiente se definió un diagrama de clases genérico que agrupa todas las funcionalidades y responsabilidades comunes de las clases, el cual se puede observar en la figura 11.

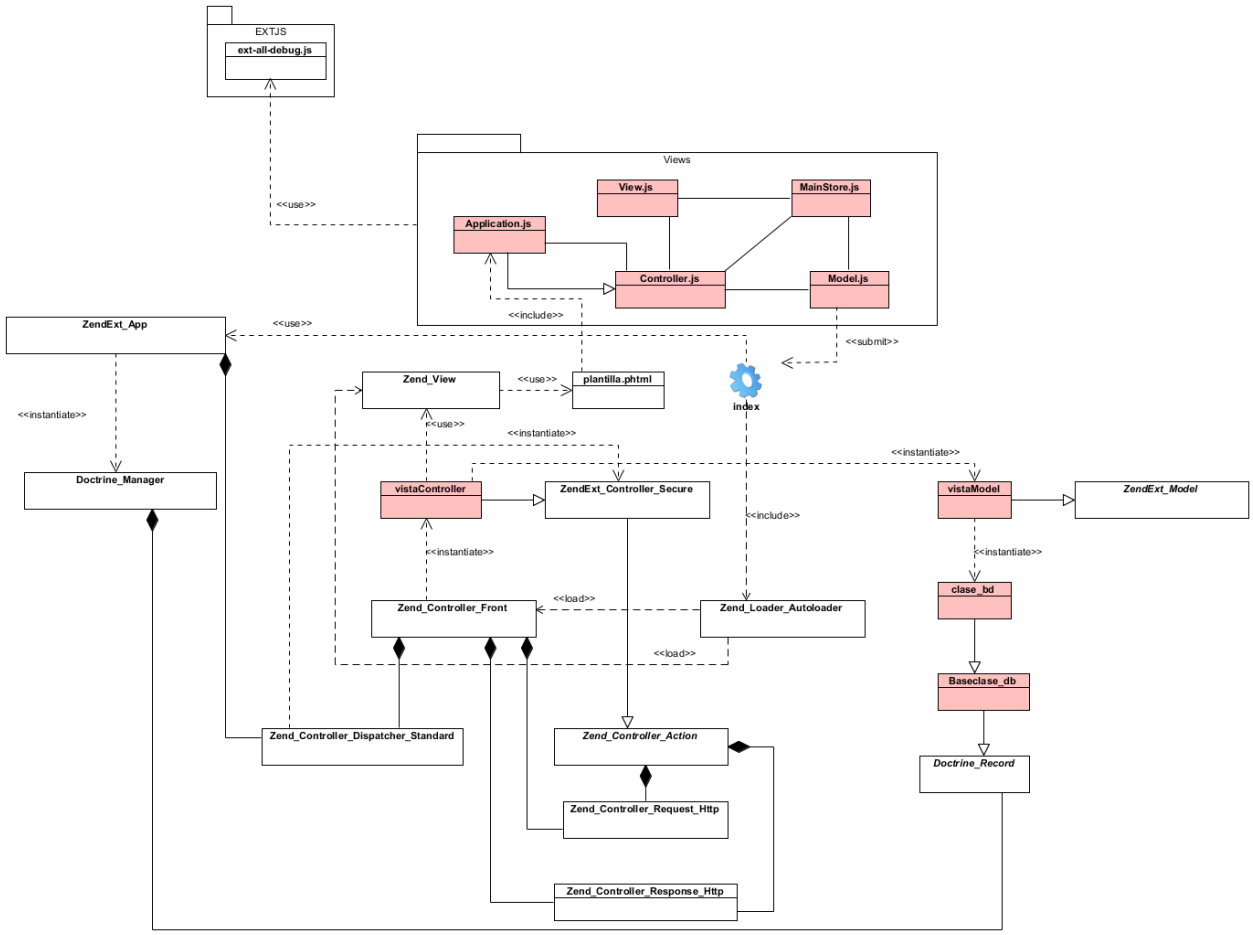


Figura 11. Diagrama de mecanismo de diseño.

2.7. Diagrama de clases del diseño web.

Los diagramas de clases de diseño web muestran un conjunto de clases e interfaces y sus relaciones, describen el diseño de los sistemas orientados a objetos ya que capturan la estructura lógica del mismo. Es un modelo que representa lo que existe y qué atributos y comportamiento tiene. La estructura de clases del sistema se especifica con relaciones entre clases e interfaces. En la figura 12 se muestra el diagrama de clases de diseño web de la clase Auditoría interna.

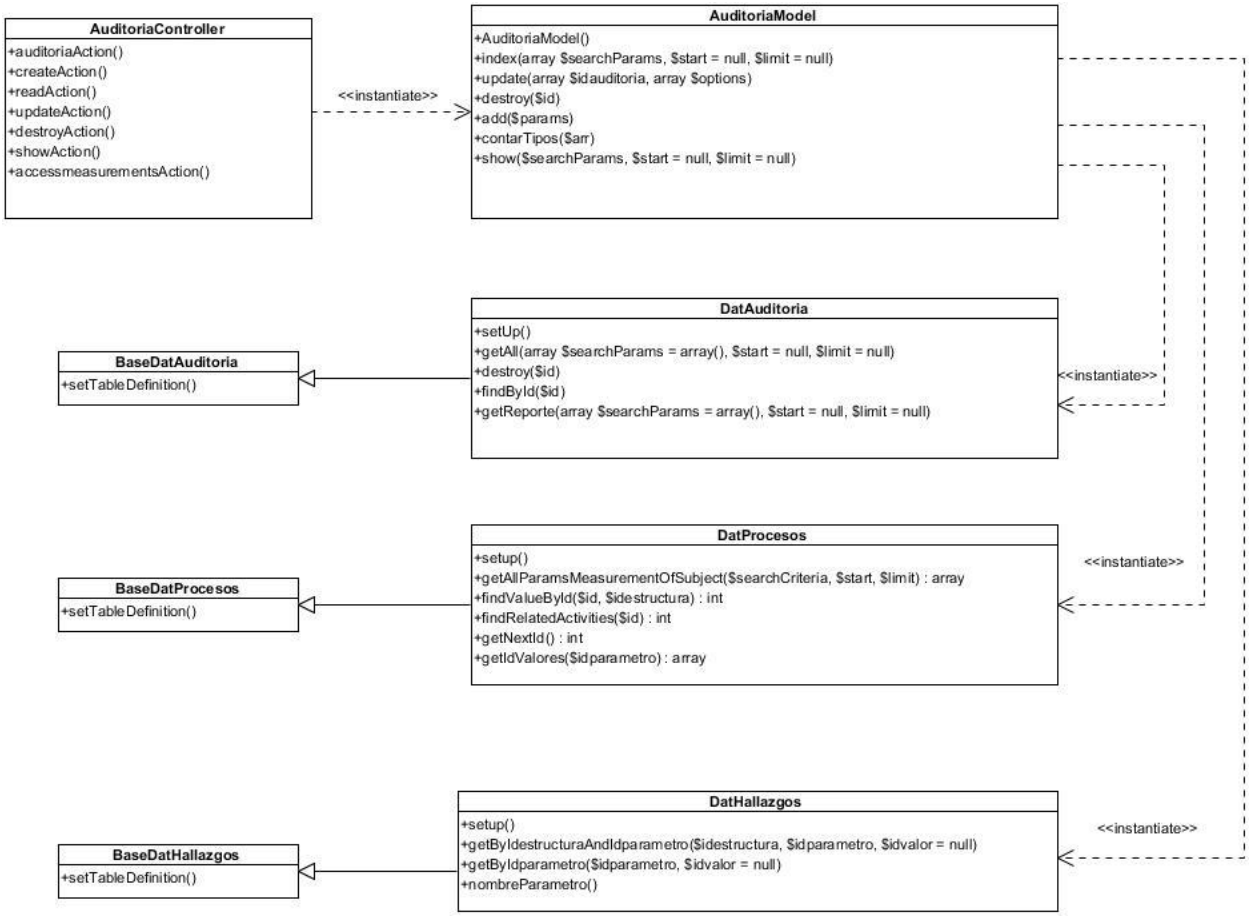


Figura 12. Diagrama de clases de diseño web de la clase Auditoría interna.

2.8. Diagrama de secuencia.

El diagrama de secuencia es una descripción de cada una de las funciones de la interfaz del sistema donde se muestran detalles de la implementación, incluyendo los objetos y clases que se usan para implementar el escenario y mensajes intercambiados entre los mismos. Se realizó un diagrama de secuencia por cada acción de los requisitos funcionales. En la figura 13 se muestra el diagrama de secuencia de la función Adicionar auditoría interna.

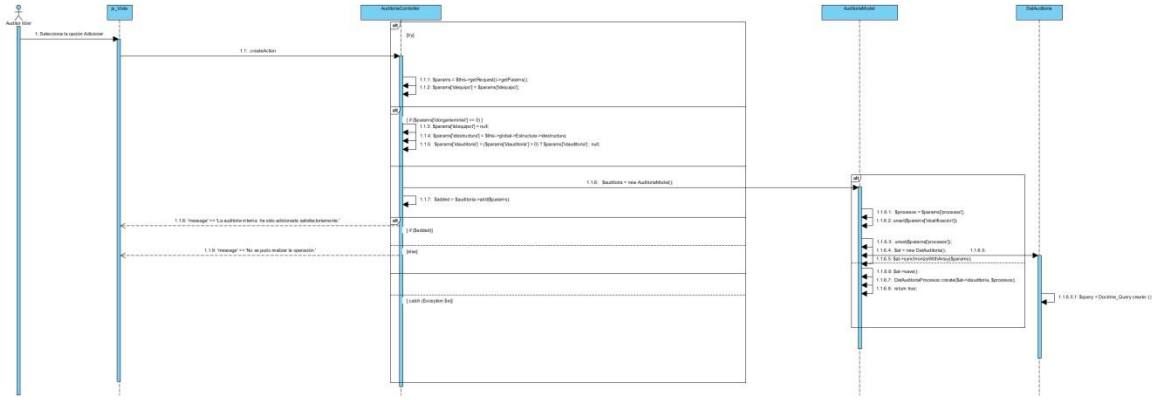


Figura 13. Diagrama de secuencia de la acción Adicionar auditoría interna.

2.9. Diagrama de despliegue.

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados, es decir se sitúa el software en el hardware que lo contiene. Cada hardware se representa como un nodo y un nodo se representa como un cubo. Un nodo es un elemento donde se ejecutan los componentes que representan el despliegue físico de estos.

En la figura 14 se muestra el diagrama de componentes del módulo de Auditoría interna.

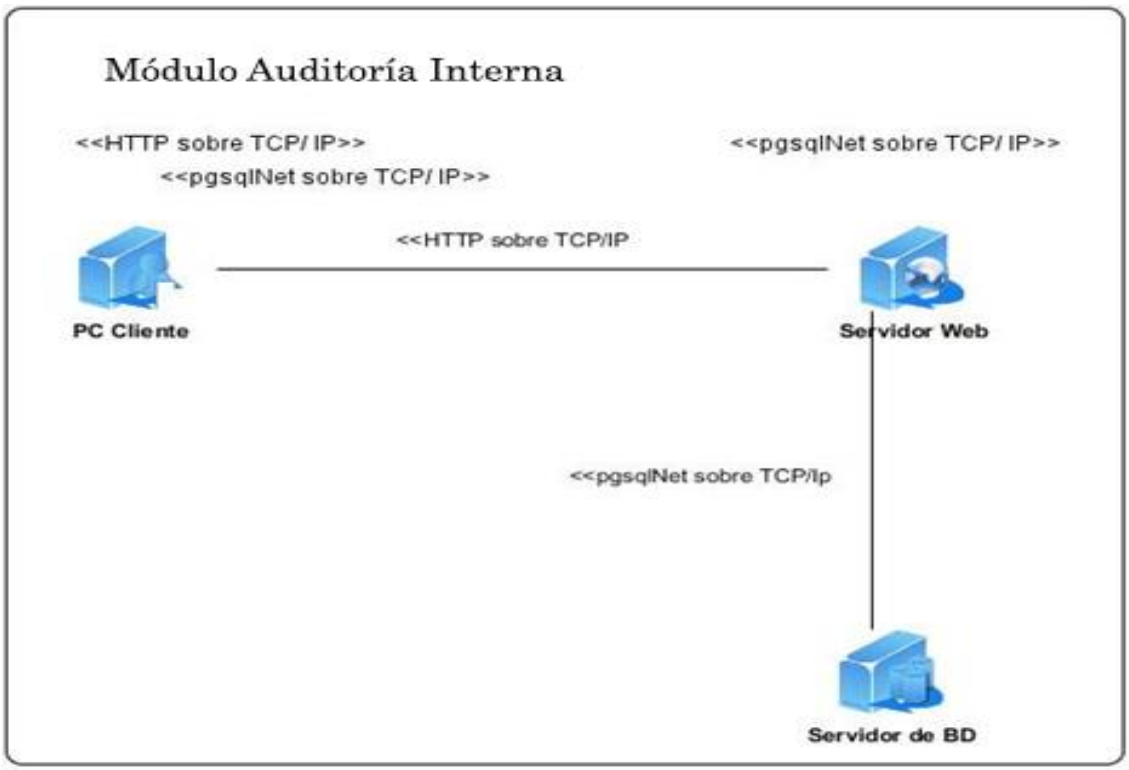


Figura 14. Diagrama de despliegue del módulo de Auditoría interna.

2.10. Tratamiento de errores.

El tratamiento de errores es un mecanismo a tener en cuenta cuando se implementa un software. Un error es un suceso o evento en tiempo de ejecución que puede causar que un método fracase, y que interrumpa la ejecución normal de un programa.

Uno de los tratamientos de errores en el módulo de Auditoría Interna se realizó a la hora de programar del lado del cliente, pues se utilizó como lenguaje de programación JavaScript, el cual posibilita validar los elementos antes de que el usuario los envíe al servidor. Permite comprobar que el formato de un campo sea el definido y se comprueba que no sobrepasa la longitud, número de líneas o tamaño de la entrada de datos. De esta forma se reduce la cantidad de transacciones que se efectúan a través del protocolo http y las posibilidades de que se genere un error durante la inserción de datos. También otros tipos de errores que puedan ocurrir en la solución tales como errores de negocio son tratados a través de excepciones, pues se le muestran mensajes personalizados al usuario para que sean más entendibles.

2.11. Seguridad.

Para el desarrollo de este software se utilizó como componente horizontal del marco de trabajo de Zeolides, el Sistema de Gestión Integral de Seguridad QuarSeg, desarrollado en la Universidad de Ciencias Informáticas. Este es responsable de garantizar los procesos de autenticación, autorización y auditorías de manera horizontal a los subsistemas que son desarrollados sobre la plataforma de software y de igual forma, mediante la implementación de WSS (Web Services Security) ofrece sus servicios a otras plataformas o sistemas externos que lo requieran.

En la figura 15 se representan, a través de un modelo conceptual, los principales conceptos o clases que forman las relaciones que se establecen entre los mismos.

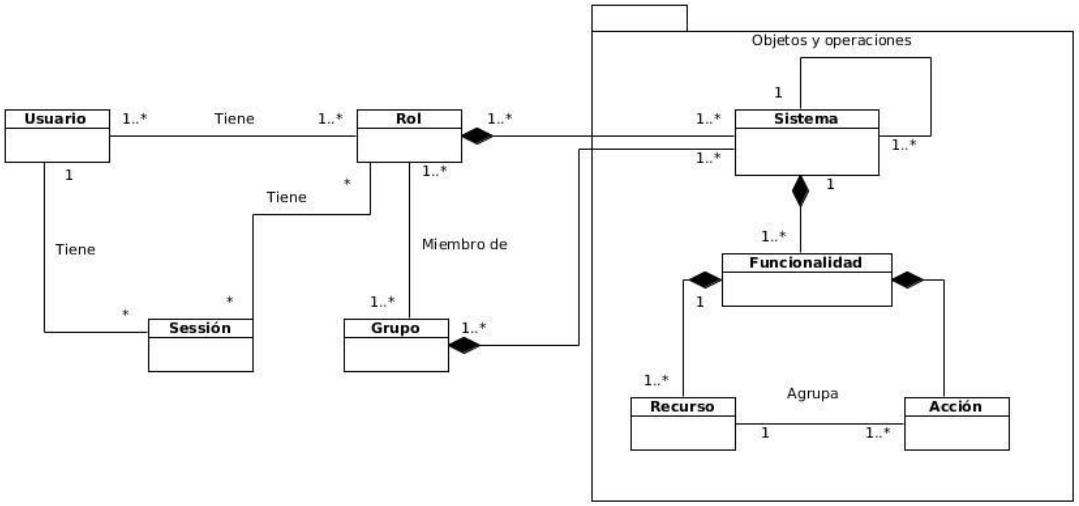


Figura 15. Modelo conceptual de seguridad.

Los conceptos del modelo conceptual de seguridad son:

- **Usuario:** Usuarios del sistema.
- **Rol:** Entidad que encapsula un conjunto de privilegios que serán asignados posteriormente a los usuarios.
- **Sesión:** Representa la asociación de los privilegios adquiridos por los usuarios mediante la asignación de roles.
- **Grupo:** Al igual que los roles encapsulan un conjunto de privilegios. La principal diferencia que se establece es que estos solo se asocian directamente a los roles (garantizando así la herencia de roles).
- **Sistema:** Se encargan de gestionar la estructura de los objetos y la forma en que estos serán mostrados a usuarios y administradores.
- **Funcionalidad:** Se encarga de gestionar las URL asociadas a los objetos registrados anteriormente.
- **Acción:** Listado de acciones que se pueden ejecutar en una funcionalidad. Básicamente en la Programación Orientada a Objetos (POO) estas podrían representar métodos de una clase.
- **Recurso:** Agrupación de un conjunto de acciones.

A su vez las relaciones entre las clases son las siguientes:

- **Usuario- Rol:** Establece que un usuario puede tener asignado varios roles y de igual forma un rol determinado puede ser asociado a otros usuarios.

- **Rol- Grupo:** Establece que un rol puede ser miembro de uno o más grupos y de igual forma un grupo puede ser asociado a diferentes roles, garantizando así la herencia de roles basada en la asignación de privilegios mediante la creación de grupos de roles.
- **Rol- Sistema/Funcionalidad/Acción/recurso y Grupo- Sistema/Funcionalidad/Acción/recurso:** Estas relaciones son las que establecen el nivel de privilegios concedidos a roles y grupos por separados, garantiza la separación de responsabilidades y permite mayor granularidad para establecer los niveles de acceso a las aplicaciones.
- **Usuario- Rol- Sesión:** Encapsula los privilegios adquiridos por un usuario en un intervalo de tiempo determinado. Comúnmente este tipo de asociación engloba todas las operaciones que un usuario puede realizar en su sesión de la aplicación.

Los privilegios asignados a los roles se derivan principalmente de las responsabilidades que tengan, a su vez los usuarios solo están autorizados a realizar las operaciones definidas según el o los roles que tenga asociado. Y finalmente los roles y privilegios solo serán administrados por los administradores y no por usuarios.

2.12. Interfaz.

El sistema a desarrollar proporcionará al usuario una interfaz amigable y fácil de entender. Existirá un lenguaje común entre el sistema y el usuario, evitando textos incomprensibles o mensajes crípticos. Para el desarrollo de las interfaces se tendrá en cuenta la propuesta del estándar para el desarrollo de interfaces de usuario presentada por el XETID, que constituirá una pauta a cumplir por todas las aplicaciones web de tipo gestión. La información que sea reflejada en la aplicación web estará exenta de errores gramaticales, ortográficos y tipográficos. Los iconos que se mostrarán se corresponderán con las tareas a realizar. El usuario siempre tendrá a la mano la posibilidad de "Ayuda" de la interfaz con la que interactuar.

2.13. Concepción de la ayuda.

El sistema informático fue realizado con el objetivo de que el usuario pudiera acceder a él con facilidad y navegar sin dificultad, lo cual indica que la aplicación posee facilidades de uso. No obstante, todas las aplicaciones web deben propiciarle al cliente una ayuda donde se explique cómo utilizarla. En la aplicación cada interfaz tendrá un botón Ayuda, al cual podrá acudir el usuario fácilmente en caso de no saber cómo realizar alguna acción, facilitándole la gestión.

2.14. Análisis del costo y el esfuerzo.

Para el análisis del costo y beneficios del sistema se utilizó la metodología de Prodesoft, donde desde el inicio se estima de forma empírica la duración de la implementación de cada uno de los requisitos, basado en la experiencia del programador en el trabajo con el lenguaje de programación, el entorno de desarrollo, el conocimiento sobre el tema de investigación y las técnicas de programación necesarias para resolver el problema. Para esto es necesario conocer el tiempo de desarrollo de cada requisito y la cantidad de trabajadores que participan para estimar si resulta beneficioso o no su desarrollo.

Como se muestra en la tabla 7 se obtiene un total de 25 semanas de trabajo en el desarrollo de la solución, que serían aproximadamente 7 meses.

Tabla 7. Análisis del costo de cada funcionalidad.

No	***Nombre de la funcionalidad***	***Prioridad***	***Complejidad***	***Tiempo de Análisis (semanas)***	***Tiempo de Desarrollo (semanas)***
1	Programa auditoría	Alta	Alta	4	4
2	Gestionar auditoría interna	Alta	Alta	4	5
3	Gestionar hallazgo	Alta	Media	1	1
4	Tratamiento de No Conformidades	Alta	Media	3	3

Para la estimación del costo del módulo se utilizó la fórmula:

$Costo = CT * SM * TD$, la cual es un submodelo matemático de base empírica desarrollado por Boehm donde:

CT: Cantidad de trabajadores

SM: Salario mensual

TD: Tiempo de desarrollo en semanas

Tomando en consideración que el salario básico promedio de un trabajador es de $365.00, el cálculo quedó de la siguiente forma:

$$Costo = CT * SM * TD$$

$$= 1 * 365 * 25$$

$$= 9{,}125$$

El costo de desarrollo del sistema fue $ 9,125.00 aproximadamente, lo que en materia económica constituye una cifra relativamente pequeña de dinero con respecto a los beneficios que se obtiene con este módulo tanto tangible como intangible. El mismo permitirá al gestor del Programa de Auditoría Interna realizar rápidamente y con los datos de manera centralizada la programación de las auditorías internas ambientales, así como la gestión de las mismas y el tratamiento de sus no conformidades y reportar al CITMA los hallazgos detectados basados en un criterio y con constancia de su evidencia. Este módulo proveerá a las empresas de una herramienta eficiente que elimina el trabajo manual, disminuye los costos y el tiempo de realización del proceso. La seguridad y protección de los datos se corresponda con el nivel requerido por el cliente y cumple con las expectativas especificadas en los requerimientos funcionales del cliente.

2.15. Conclusiones.

Luego del análisis y estudio de este capítulo se pudo observar claramente cómo se realiza el proceso de auditoría interna. Se logró comprender los conceptos asociados al dominio del problema, lo que se refleja en el artefacto modelo conceptual. Se obtuvo la lista de requisitos especificados que constituyen la base para el desarrollo de los modelos de diseño. Se realizaron detalladamente los diagramas de secuencia que guiarán el proceso de implementación de forma más fácil y los diagramas de componentes que facilitarán el conocimiento de las relaciones de dependencias entre los componentes y las interfaces que estos soporten. El estudio de la relación beneficio- costo arrojó un estimado del proyecto y se demostró la factibilidad del desarrollo del mismo.

Capítulo 3. Validación de la solución propuesta

3.1. Introducción.

En este capítulo se describe el software desarrollado como resultado de la solución propuesta. Se definen los métodos de pruebas a utilizar y se realizan los mismos, con el objetivo de comprobar su calidad y el cumplimiento de los requisitos establecidos desde un inicio, lo que constituye uno de los pasos más importantes en el diseño e implementación de un sistema.

3.2 Descripción del software.

El Sistema Automatizado para la Gestión Ambiental Empresarial (SAPGAE) del cual se muestra su pantalla principal en la figura 16, se desarrolla con el objetivo de crear una plataforma tecnológica que provea una eficiente gestión, control y visibilidad de las actividades que realizan las empresas hoy en día, en el cumplimiento oportuno de sus compromisos con el medio ambiente y la nueva estrategia ambiental del país. Uno de los elementos fundamentales para mantener una gestión ambiental adecuada y responsable es el cumplimiento de la empresa respecto a los requisitos y normas ambientales vigentes en nuestro país. El sistema cuenta con varios módulos, en esta investigación se desarrolló el módulo Auditoría Interna al cual se puede acceder como se muestra en la figura 17, posibilitándole a las empresas identificar y encausar las no conformidades detectada respecto al medio ambiente y darles el tratamiento correspondiente. El módulo cuenta con 4 funcionalidades, las que se describen a continuación:

Programa de auditoría: Permite adicionar, modificar, buscar, ver detalles, eliminar los programas de auditorías, además permite, crear los equipos, asignarle las auditorías a realizar, definir quiénes son los encargados de la revisión y aprobación del programa e imprimir el programa seleccionado el cual se puede observar en el Anexo 5.

Gestionar auditoría interna: Permite adicionar, modificar y buscar las auditorías internas. Además, permite asignarle los resultados según los hallazgos detectados en cada auditoría e imprimir el informe de auditoría, el control de las mismas y el tratamiento de sus no conformidades la cual se puede observar en la figura 3 correspondiente a la ERS Prototipo de IU Gestionar auditoría, que se encuentra en el Capítulo 2 de este documento**.**

Gestionar hallazgos: Permite adicionar, modificar, buscar y eliminar los hallazgos, además de ser asociados a una auditoría después de haber seleccionado un proceso en la funcionalidad Auditoría interna visible en el Anexo 6.

Tratamiento de No Conformidades: Permite buscar, imprimir, visualizar los detalles y darle seguimiento a la no conformidad detectada en las auditorías internas, ver Anexo 7.

Desde cada funcionalidad se puede consultar la ayuda detallada de la misma.

Figura 16. Ventana principal del sistema.

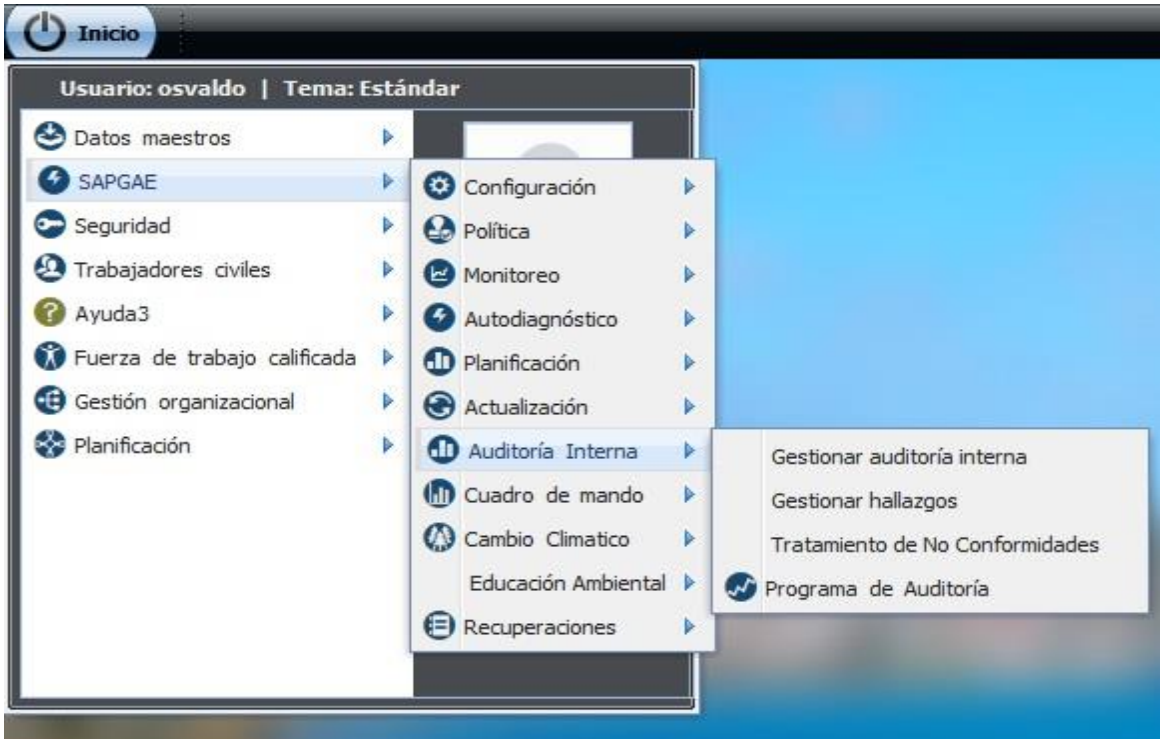


Figura 17. Menú principal para acceder al módulo Auditoría Interna y sus funcionalidades.

3.3. Realización de las pruebas.

Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de software. Es un proceso de ejecución de un programa con la intención de descubrir errores. El principal objetivo del procedimiento de prueba es servir como guía y evaluar o valorar la calidad del producto a través de: buscar y documentar errores, validar el cumplimiento de requerimientos y el desempeño, dar una indicación de calidad (VALDIVIA 2005).

3.3.1. Métodos empleados para la realización de las pruebas.

Para la realización de las pruebas del software se tuvieron en cuenta varios métodos definidos dentro de las normativas para la liberación del software en el XETID.

Pruebas de caja negra: Son pruebas que se llevan a cabo sobre la interfaz del software. El objetivo es demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y se produce un resultado correcto y que la integridad de la información externa se mantiene (VALDIVIA 2005).

Pruebas de aceptación: Son aquellas pruebas diseñadas por el propio equipo de desarrollo en base a los requisitos funcionales especificados en la fase de análisis para cubrir todo ese espectro, y ejecutadas por el propio usuario final, no por todos evidentemente, pero sí por una cantidad de usuarios finales significativo que den validez y conformidad al producto que se les está entregado en base a lo que se acordó inicialmente (SANZ, 2010).

Pruebas unitarias y de integración: Las pruebas unitarias consisten en realizar pruebas a pequeños fragmentos de código. Cada prueba tiene que ser lo más independiente posible de las otras y encargarse de una acción específica. En POO se puede afirmar que estas unidades son los métodos o las funciones que están definidos. El objetivo de las pruebas unitarias es el aislamiento de partes del código y la demostración de que estas partes no contienen errores (SANZ, 2010).

Una vez que se han ejecutado las pruebas unitarias se llevan a cabo las pruebas de integración las cuales consisten en realizar pruebas para verificar que un gran conjunto de partes del software funcionan juntos (SANZ, 2010).

Pruebas funcionales: Están basadas en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software, son pruebas específicas, concretas y exhaustivas para probar y validar que el software hace lo que debe y sobre todo, lo que se ha especificado. Está compuesta por varias fases: Planificación, Preparación, Ejecución y Defectos (SANZ, 2010).

En las tablas 8 y 9 se muestran las clases de equivalencia utilizadas en las pruebas funcionales realizadas a la funcionalidad Adicionar Auditoría interna y el resumen de casos de pruebas respectivamente.

Tabla 8. Clases de equivalencia.

Condición de Entrada	Clases Válidas	Clase Inválidas
Código	1. Se introduce un código en el Textfield.	2. No se introduce el código en el Textfield.
Mes	3. Se selecciona una de las opciones de Combo box.	4. No se selecciona una de las opciones de Combo box.

Objetivos	5. Se introduce el objetivo en el Textarea.	6. No se introduce el objetivo en el Textarea.
Alcance	7. Se introduce el alcance en el Textarea.	8. No se introduce el alcance en el Textarea.
Fecha de ejecución	9. Se selecciona la fecha de ejecución en el Datefield.	10. No selecciona la fecha de ejecución en el Datefield.
Cliente	11. Se introduce el cliente en el Textfield.	12. No se introduce el cliente en el Textfield.
Ubicación	13. Se introduce la ubicación en el Textfield.	14. No se la ubicación en el Textfield.

Tabla 9. Resumen de casos de pruebas.

Clases de Equiva-lencia	Textfield Código	Combo box Mes	Textarea Objetivos	Textarea Alcance	Datefield Fecha de ejecución	Textfield Cliente	Textfield Ubicación	Botón Aceptar	Resultado esperado (Mensaje)
1,3,5,7, 9,11,13	Se introduce un código	Se selecciona un mes	Se introduce l objetivo	Se introduce el alcance	Se selecciona la fecha de ejecución	Se introduce el cliente	Se introduce la ubicación	Habilitado	*La auditoría sido adiciona correctamente*
2,3,5,7, 9,11,13	No se Introduce un código	Se selecciona un mes	Se introduce el objetivo	Se introduce el alcance	Se selecciona la fecha de ejecución	Se introduce el cliente	Se introduce la ubicación	Deshabilitado	_____
1,4,5,7, 9,11,13	Se Introduce un código	No se selecciona el mes	Se introduce el objetivo	Se introduce el alcance	Se selecciona la fecha de ejecución	Se introduce el cliente	Se introduce la ubicación	Deshabilitado	_____
1,3,6,7, 9,11,13	Se Introduce un código	Se selecciona el mes	No se Introduce el objetivo	Se introduce el alcance	Se selecciona la fecha de ejecución	Se introduce el cliente	Se introduce la ubicación	Deshabilitado	_____
1,3,5,8, 9,11,13	Se Introduce un código	Se selecciona el mes	Se introduce el objetivo	No se introduce el alcance	Se selecciona la fecha de ejecución	Se introduce el cliente	Se introduce la ubicación	Deshabilitado	_____

1,3,5,7, 10,11,13	Se Introduce un código	Se selecciona el mes	Se introduce el objetivo	Se introduce el alcance	No se selecciona la fecha de ejecución	Se introduce el cliente	Se introduce la ubicación	Deshabilitado	_____
1,3,5,7, 9,12,13	Se Introduce un código	Se selecciona el mes	Se introduce el objetivo	Se introduce el alcance	Se selecciona la fecha de ejecución	No se introduce el cliente	Se introduce la ubicación	Deshabilitado	_____
1,3,5,7, 9,11,14	Se Introduce un código	Se selecciona el mes	Se introduce el objetivo	Se introduce el alcance	Se selecciona la fecha de ejecución	Se introduce el cliente	No se introduce la ubicación	Deshabilitado	_____

3.3.2. Diseño de Casos de Pruebas (DCP)

Los casos de prueba pretenden demostrar que las funciones del software son operativas, que no existen errores de interfaz o de rendimiento a cada funcionalidad del módulo. A continuación se muestra el DCP de la funcionalidad Gestionar auditoría.

Condiciones de ejecución:

- El usuario ha sido autenticado.
- El usuario selecciona la opción Auditoría Interna/Gestionar auditoría.

Nombre del requisito	**Descripción general**	**Escenarios de pruebas**	**Flujo del escenario**
Gestionar auditoría.	Este requisito permite gestionar las auditorías programadas y no programadas de la empresa, posibilitando adicionar, modificar, asociar resultados, buscar, imprimir y graficar.	EP1 Adicionar auditoría.	El sistema carga la interfaz Gestionar auditoría: • Se oprime el botón **Adicionar**. • Se muestra la interfaz Adicionar auditoría. • Se llenan los campos .Los que son de carácter obligatorio se deben introducir o aparecen en color rojo. • Se selecciona la opción **Aceptar**. • Si se selecciona la opción **Aplicar**, el sistema valida los datos, se guardan, se limpian los campos y se muestra un mensaje de

			información: *"La auditoría ha sido adicionada satisfactoriamente"* y se mantiene la interfaz permitiendo adicionar una nueva auditoría. • Si se selecciona la opción **Cancelar**, el sistema no adiciona la auditoría y se cierra la ventana actual. • Se muestra un mensaje de información: *"La auditoría ha sido adicionada satisfactoriamente"* y se cierra la interfaz. • Se muestran los datos adicionados en el grid.
		EP2 Modificar auditoría.	El sistema carga la interfaz Gestionar autodiagnóstico: • Se selecciona del listado el autodiagnóstico que se desea modificar. • Se habilita el botón **Modificar.** • Se oprime el botón **Modificar**. • Se muestra la interfaz Modificar auditoría. • Se introducen los datos que sean permitidos modificar. • En caso de que la información sea incompleta o introdujo error en algún campo, el sistema marca los campos seleccionados de color rojo y permite corregirlos. • Si se oprime el botón **Cancelar** el sistema no realiza ninguna operación y cierra la interfaz. • Se presiona el botón **Aceptar**. • El sistema valida los datos, los guarda, muestra el mensaje: *"La auditoría ha sido modificada satisfactoriamente."*, se muestran

			los datos en el listado de auditorías y se regresa a la interfaz Gestionar auditoría.
		EP3 Resultados.	El sistema carga la interfaz Gestionar auditoría: • Se selecciona en el listado la auditoría a la que se desea asociar los resultados. • Se habilita el botón **Resultados**. • Se oprime el botón **Resultados**. • Se selecciona el proceso. • Se selecciona en el grid el hallazgo que se desea asignar. • Se selecciona el botón con el icono  . • El sistema muestra el hallazgo en el grid de Seleccionados después de haber adicionado los datos necesarios en la ventana de Hallazgo (Ver ERF Hallazgo en el expediente SAPGAE) y muestra el mensaje de información *"El hallazgo ha sido adicionado satisfactoriamente".* • Se introducen los datos de las conclusiones y el grado de cumplimiento. • Se presiona el botón **Guardar**. • El sistema muestra el mensaje de información *"La auditoría ha sido modificada satisfactoriamente".* • Si se oprime el botón **Cancelar** el sistema no realiza ninguna operación y cierra la interfaz.

		EP4. Imprimir.	El sistema carga la interfaz Gestionar auditoría: • Se selecciona del listado de auditorías la que se desea imprimir, la cual debe de tener grado de cumplimiento. • Se habilita el botón **Imprimir.** • Se despliega un listado de los reportes que se pueden obtener. • Si selecciona **Informe de auditoría interna.** • Se muestra una nueva ventana con toda la información el informe de la auditoría realizada en los pasos anteriores. • Si selecciona **Resumen.** • Se muestra una nueva ventana con el resumen de las auditorías internas realizadas.
		EP5. Buscar auditoría.	El sistema carga la interfaz Gestionar auditoría: • Se introduce el código, la fecha de ejecución o ambos por los que se desea buscar. • Se marca el botón **Buscar** para realizar la búsqueda. • Se muestra en el grid el resultado de la búsqueda en caso de que exista.

3.4. Análisis de los resultados obtenidos.

Después de desarrollar todo un proceso de pruebas se lograron resultados satisfactorios, pues tras la detección de los diferentes errores que se obtuvieron, se solucionaron varios problemas que impedían el cumplimiento de los requisitos funcionales del módulo en cuestión. Las primeras pruebas fueron planeadas y ejecutadas en funcionalidades individuales del sistema y a medida que se avanzó se desplazaron a módulos integrados, hasta que finalmente se completó y logró obtener un software cuyas funciones se encuentran en correspondencia con las especificaciones acordadas y que además

cumple con los requerimientos de seguridad. El desarrollo de la aplicación cumple las expectativas trazadas al inicio del proyecto y satisface al cliente en su totalidad.

3.5. Conclusiones.

La realización de pruebas le ofreció al desarrollador conocer los errores del sistema y así poder erradicarlos. Se obtuvo la documentación de pruebas y el manual de usuario del módulo que brinda una detallada explicación sobre su funcionamiento. Por otra parte las pruebas de aceptación realizadas por el cliente posibilitaron pasar a las siguientes fases sin errores, agilizando el desarrollo de la aplicación.

Conclusiones

Al finalizar el desarrollo del presente trabajo se concluye que se cumplieron los objetivos trazados y se arriba a las siguientes conclusiones:

1. Se elaboró el marco teórico referencial requerido, el que expresa el estado actual del arte y de la práctica en la gestión del proceso de auditorías internas ambientales.
2. Se determinaron, mediante la metodología PRODESOFT, los requisitos funcionales como requerimiento previo al diseño del software SAPGAE, para realizar la propuesta de solución.
3. Se implementó el módulo Auditoría Interna que permitirá la gestión de este proceso en las empresas cubanas.
4. Se verificó la correcta ejecución de las funcionalidades del módulo Auditoría Interna, a través de los diferentes tipos de pruebas que fueron realizadas.
5. Se valida de esta forma la hipótesis propuesta, comprobándose que el módulo de Auditoría Interna contribuye a elevar la seguridad y la eficiencia de la gestión de este proceso en diversas empresas de nuestro país.

Recomendaciones

Desde el punto de vista del alcance del presente trabajo y teniendo en cuenta el tiempo para el desarrollo del mismo, se proponen las siguientes recomendaciones:

1. Validar los resultados del módulo Auditoría Interna a partir del criterio de especialistas.
2. Realizar los estudios de adaptación en empresas pilotos de nuestra provincia con el objetivo de comprobar su rendimiento en el entorno real.
3. Emprender el monitoreo de los impactos del módulo de Auditoría Interna, sobre la mejora del proceso de gestión en las empresas cubanas.

Referencias Bibliográficas

BAKAIKOA, B. *et.al*. Redes e innovación cooperativa, Revista de Economía Pública, Social y Cooperativa, en CIRIEC-España, nº 49, pp. 263-294, agosto 2004.

ESCOBAR, S. C. Realidad de los Sistemas de Gestión Ambiental, 2003.

GALLARDO, I. 2010. Utilizando Doctrine como ORM en php. Obtenido de: Utilizando Doctrine como ORM en php: http://web.ontuts.com/tutoriales/utilizando-doctrine-como-orm-en-php/.

GARCÍA, D. Metodología para la gestión ambiental para sitios agrícolas con probables riesgos a la salud por presencia de contaminación. Tesis en opción al grado científico de Doctor en Ciencias Técnicas. Instec. La Habana, 2012.

GARCÍA, J. J. *ET AL.* UML: el lenguaje estándar para el modelado de software. Novática: Revista de la Asociación de Técnicos de Informática, 2004, no 168, p. 4-5.

GAUCHAT, J. D. El gran libro de HTML5, CSS3 y Javascript. 1era edición, 2012, Ediciones Técnicas. ISBN eBook: 978-84-267-1782-5 2012. DL: SE-7867-2011.

HURTADO, E. (s.f.). Manual de usuario. Marco de trabajo para el desarrollo de aplicaciones web. Versión 2.0.

MAYORI, M. 2013. Fundamentos de Ext JS. Obtenido de: Fundamentos de Ext JS: https://www.udemy.com/ext-js-fundamentos/.

MILIÁN, C. Sistema Integrado de Gestión Estadística: Componente para la captación de encuestas económicas. Tesis para optar por el Título de Ingeniero en Ciencias Informáticas, Universidad de las Ciencias Informáticas, La Habana, 2011.

NC. Directrices para la auditoría de los sistemas de gestión (ISO 19011: 2011, IDT). 2da edición, mayo, 2012. ICS: 03.120.10; 13.020.10.

NC. Sistema de Gestión Ambiental- Requisitos con Orientación para su uso [ISO 14001:2015, (Traducción certificada), IDT]. 2da edición, septiembre, 2015. ICS: 13.020.10.

Proceso de Desarrollo y Gestión de Proyectos de Software. La Habana, 2012.

Qué es PHP?, 2013. [Disponible en: http://www.phpya.com.ar/temarios/descripcion.php?cod=23

QUINTERO, J. B. *et al.* Un estudio comparativo de herramientas para el modelado con UML. Revista Universidad EAFIT, 2012, vol. 41, no 137, p. 60-76.

RAMOS, H. & RODRÍGUEZ, L. R. Subsistema de Gestión Comercial para el Sistema de Gestión Empresarial DISTRA. Universidad & Ciencia. Vol. 7, No. 1, diciembre-marzo (2018). ISSN: 2227-2690 RNPS: 2450.

SALINAS CARO, P., & HISTCHFELD K, N. (s.f.). Tutorial UML. Obtenido de Tutorial UML: http://users.dcc.uchile.cl/~psalinas/uml/introduccion.html

SANTANA, Y. Herramienta informática para la gestión de auditorías de sistemas de gestión ambiental basados en ISO 14001. Tesis en opción al título de Máster en Informática para la Gestión Medioambiental, Universidad Central "Marta Abreu", Las Villas, 2014.

SANZ. Pruebas de aceptación en Sistemas Navegables. Revista Española de Innovación, Calidad e Ingeniería del Software, 2010, 47-55, DDLF.

SOMMERVILLE, I. *et al.* 2005. Ingeniería del software. Madrid, Pearson Educación, 687p.

SOUTO, R. 2010. Maestros del web. Obtenido de: maestros del web: http://www.maestrosdelweb.com/guia-zend/

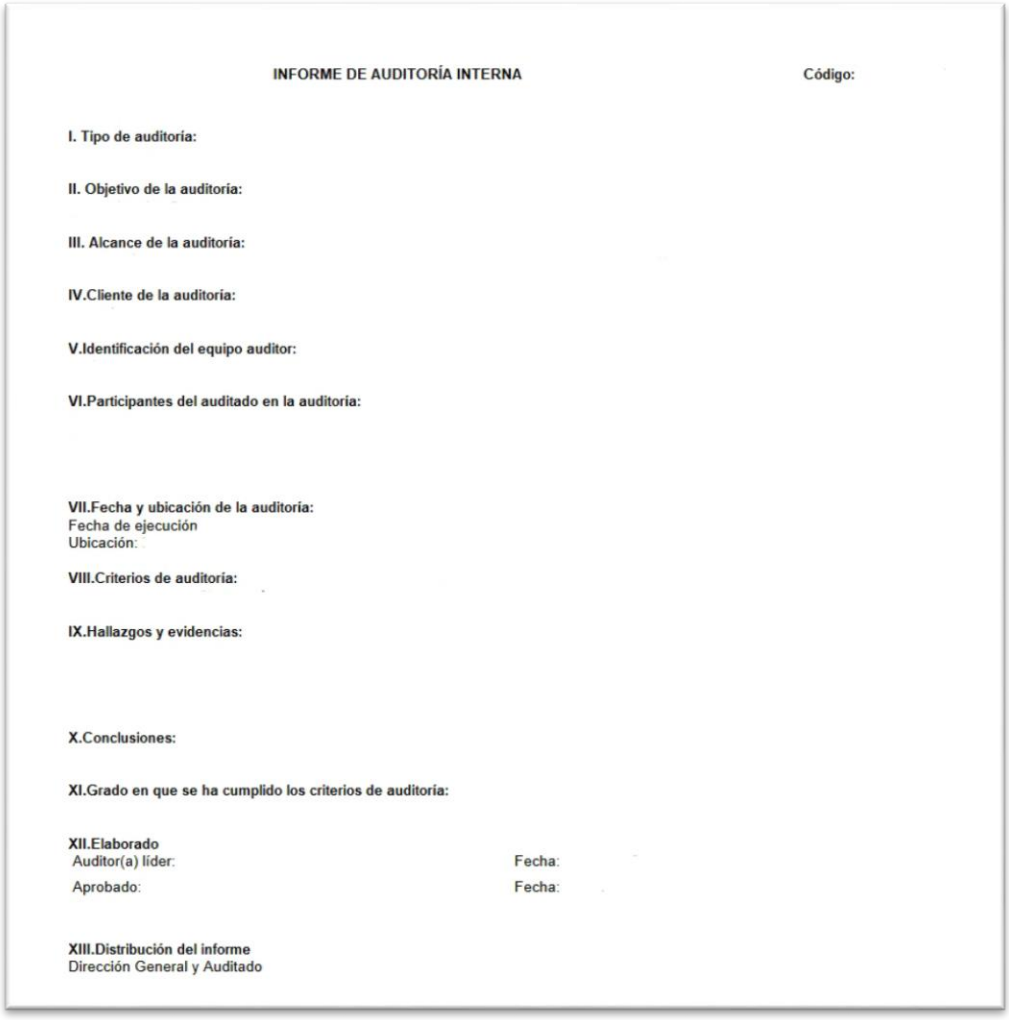
VALDIVIA, D. R. & VALDIVIA, E.G. Estándares de calidad para pruebas de software. Tesis para optar por el título profesional de Ingeniero de Sistemas. Universidad Nacional Mayor de San Marcos. Perú, 2005.

Anexos

Anexo # 1.Comparación entre PRODESOF y las metodologías ágiles y robustas.

Aspectos	Robusta	Ágil	PRODESOF
Requisitos	Requieren los requisitos detallados desde el inicio del proyecto y no pueden cambiar.	Los requisitos son muy cambiantes y se requiere de un feedback sobre un resultado obtenido para determinar si es lo requerido o no.	En su inicio se realiza la captura de requisitos del sistema, durante la modelación se aceptan, en la construcción se aclaran y en la explotación están estables a un 95% hasta que llegan a despliegue con un 100% de su cumplimiento.
Cambios	Hacer un cambio al alcance requiere de un proceso formal de control de cambios.	El cambio es bienvenido en cualquier momento del proyecto.	Es controlando el cambio de los elementos a lo largo de su ciclo de vida, registrando y reportando su estado, las solicitudes de cambio y verificando estén completos y sean los correctos. Se aplica en todas las fases del ciclo de vida del proyecto.
Tiempo	Existe un compromiso respecto al tiempo de entrega del proyecto (no siempre se cumple esta meta).	Existe incertidumbre respecto al tiempo de entrega de todo el producto. Lo cierto es que máximo cada 2 meses (máximo un mes en Scrum) hay entrega de producto de valor para el cliente.	Se planifica para que exista puntualidad en la conclusión del proyecto. Estas actividades constituyen el resultado del trabajo que debe realizarse para entregar, en el tiempo acordado, un producto con las funciones y características especificadas.
Documentación	Atención exhaustiva a la documentación.	Solo se genera la documentación que genera valor al cliente y al proyecto.	Se lleva el control de la documentación con las especificaciones del funcionamiento establecido como norma y principio.
Iteraciones	Pocas iteraciones que generan gran volumen de información y software para la construcción del producto.	Utilización de múltiples iteraciones de desarrollo para aprender y evolucionar el producto.	En cada iteración se entrega una parte de la funcionalidad empezando por el modelo/requisitos hasta las pruebas/despliegue y en la siguiente iteración se amplía la utilidad del producto a desarrollar.
Riesgos	Los riesgos son asumidos por el proveedor.	Voluntad del cliente para compartir la responsabilidad en las decisiones y riesgos.	Se realiza una gestión de riesgos que se compone de las actividades de Planificación, Identificación y Evaluación de riesgos, así como la Planificación de la respuesta y el Seguimiento y control de los mismos; teniendo como principal objetivo disminuir la probabilidad y el impacto de los eventos adversos para el desarrollo de los proyectos.

Anexo 2. Prototipo de IU Informe de auditoría interna.



INFORME DE AUDITORÍA INTERNA Código:

I. Tipo de auditoría:

II. Objetivo de la auditoría:

III. Alcance de la auditoría:

IV.Cliente de la auditoría:

V.Identificación del equipo auditor:

VI.Participantes del auditado en la auditoría:

VII.Fecha y ubicación de la auditoría:
Fecha de ejecución
Ubicación:

VIII.Criterios de auditoría:

IX.Hallazgos y evidencias:

X.Conclusiones:

XI.Grado en que se ha cumplido los criterios de auditoría:

XII.Elaborado
Auditor(a) líder: Fecha:
Aprobado: Fecha:

XIII.Distribución del informe
Dirección General y Auditado

Anexo 3. Prototipo del Informe del Resumen auditoría interna.

CONTROL DE AUDITORIAS INTERNAS

No	Fecha de ejecución	Código	Objetivo	Equipo	Proceso	Cantidad				Conclusiones
						NC	C	OM	BP	

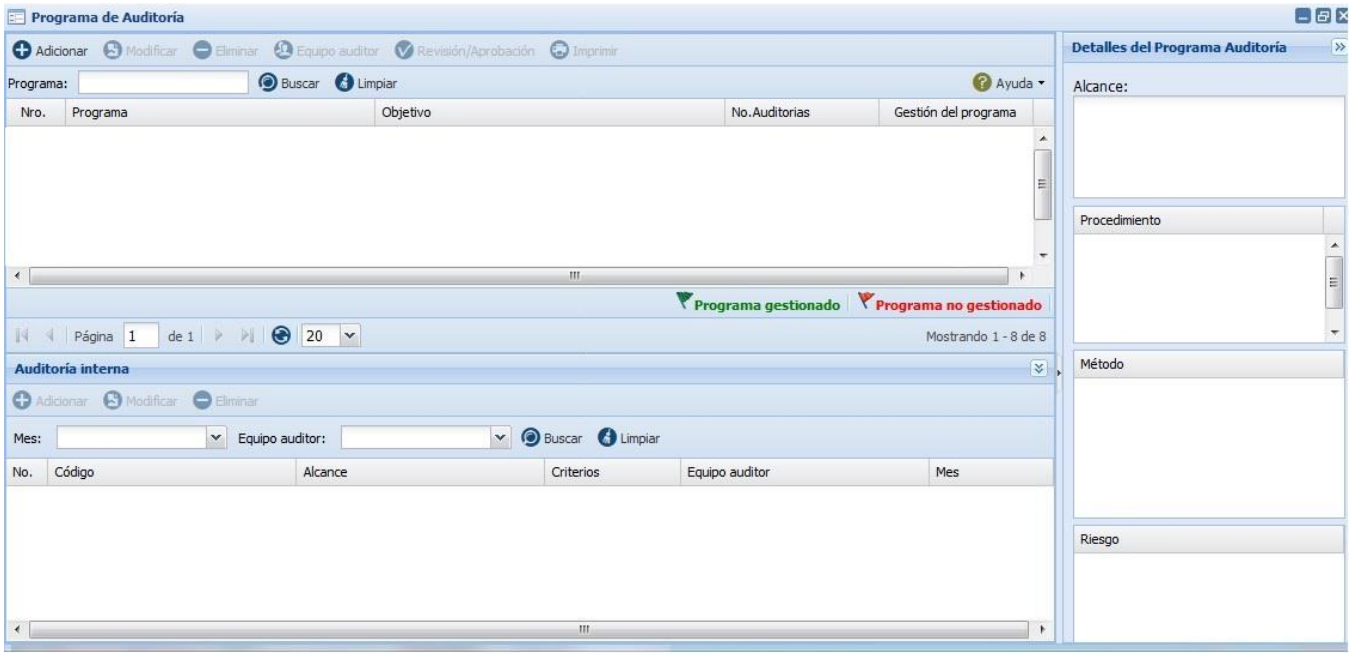
Anexo 4. IU del Informe de Tratamiento de no conformidades.

TRATAMIENTO DE NO CONFORMIDADES

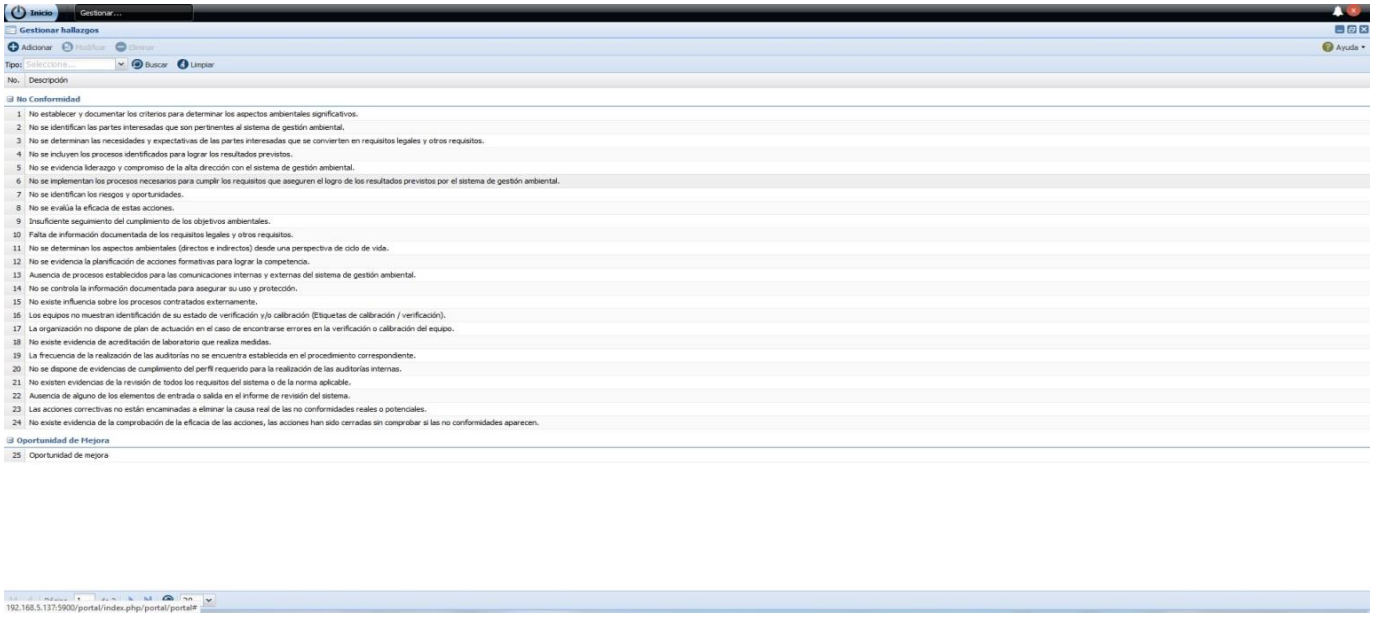
Como resultado de:

No-Conformidades						
No	Referencia	Descripción	Detectó	Área	Fecha	Jefe de área
Causa						
No	Descripción		Fecha	Jefe de área	Visto bueno	
Acción correctiva						
No	Descripción		Fecha	Jefe de área	Visto bueno	
Seguimiento						
No	Seguimiento		Resultado		Fecha	Área

Anexo 5. IU de la funcionalidad Programa de auditoría.



Anexo 6. IU de la funcionalidad Gestionar Hallazgo.



Anexo 7. IU de la funcionalidad Tratamiento de no conformidades.