MODELADO CONCEPTUAL DEL SOFTWARE DE APOYO PARA LA AUDITORIA DE SISTEMAS Y TECNOLOGÍAS DE LA INFORMACIÓN "SISAUDITE"

CONCEPTUAL MODELLING OF THE SOFTWARE OF SUPPORT FOR THE AUDIT OF SYSTEMS AND TECHNOLOGIES OF THE INFORMATION "SISAUDITE"

AUTOR
EDWIN DURAN BLANDON
Ingeniero de Sistemas, Especialista en Tecnologías Avanzadas para el Desarrollo de Software.
Universidad Cooperativa de Colombia Seccional Barrancabermeja - Instituto Universitario de la Paz.
Docente
Facultad de Ingeniería de Sistemas
edubla01@yahoo.es
COLOMBIA

INSTITUCIÓN
Universidad Cooperativa de Colombia Seccional Barrancabermreja
UCC Barrancabermeja
Calle 57 Carreras 24 y 27 B. Galán, Segundo Piso
Barrancabermeja - Colombia
e-mail: Comunicacionesbca@correoucc.edu.co
COLOMBIA

Recepción: Junio 10 de 2009

Aceptación: Septiembre 16 de 2009

Temática: Gestión Tecnológica

Artículo Tipo: Artículo de Investigación científica y Tecnológica

RESUMEN

Este trabajo presenta un modelado conceptual para la construcción de un software de apoyo a la auditoria de sistemas y tecnologías de la información acorde al estándar COBIT (Control Objectives for Information and Related Technology) de Information System Audit and Control Association (2006), utilizando como base la metodología de sistemas blandos propuesta por Checkland (1994) y la propuesta de investigación del Dr. Martínez (2004), así como el uso del Proceso unificado de Desarrollo de Software de Jacobson, I., Booch, G., Rumbaugh, J. (1999) para el desarrollo de la herramienta software.

PALABRAS CLAVES

Auditoria de sistemas
COBIT
Ingeniería del software

ABSTRACT

This work presents conceptual modelling for the construction of a software of support to the audit of systems and technologies of the information to the

standard COBIT (Control Objectives for Information and Related Technology) ISACA (2006), using like base the Soft System Methodology proposed by Checkland (1994) and the offer of investigation(research) of the Dr. Martinez (2004), as well as the use of the Process unified of Development of Software Jacobson, I., Booch, G., Rumbaugh, J. (1999) for the development of the tool software.

KEYWORDS

Systems of Audit
COBIT
Software of the Engineering

INTRODUCCION

La propagación del uso de los sistemas y las tecnologías de la información (S&TI) en las organizaciones grandes, medianas y pequeñas ha logrado que las empresas sistematicen sus procesos y mejoren la calidad de los servicios que prestan a sus clientes pero también ha conseguido que se vean cada vez más expuestas a riesgos y a la necesidad de plantear mecanismos de control que le permitan asegurar la continuidad del servicio. La auditoria de Sistemas y tecnologías de la Información como una posible solución a este problema es un tema que toma bastante auge en las últimas décadas, ya que ha permitido evaluar continuamente la adquisición, planeación, desarrollo y monitoreo de los sistemas que apoyan y sostiene la información.

El objetivo final que tiene el auditor de S&TI es dar recomendaciones a la alta gerencia para mejorar o lograr un adecuado control interno en ambientes de tecnología informática con el fin de lograr mayor eficiencia operacional y administrativa. Por ende debe basar su trabajo en métodos de auditoria que le permitan obtener los resultados esperados.

Uno de los métodos utilizados para la realización de la auditoria de S&TI es el método normativo, Gómez (2000), el cual se basa en estándares de auditoria que se comparan con el sistema auditado para determinar cuáles son las recomendaciones pertinentes. Parte de de los estándares utilizados internacional y nacionalmente por los auditores de S&TI es el estándar COBIT (Control Objectives for Information and Related Technology) el cual se basa en objetivos de control que permitan:

- Usuarios. Garantizar la seguridad y el control de los servicios prestados por los S&TI
- Gerencia. Realizar un balance entre los riesgos y las inversiones en control de los S&TI
- Auditores. Emitir juicios asertivos respecto de los controles asociados a los riesgos en los S&TI.

De acuerdo con los objetivos de control COBIT, el auditor crea listas de verificación que le permite planificar el desarrollo de la auditoria y garantizar el adecuado levantamiento de las evidencias que le habiliten para entregar y apoyar el juicio de auditoria.

Sin embargo, dada la importancia y confidencialidad de la auditoria, las empresas encargadas de realizar esta labor en Colombia y el mundo guardan celosamente las listas de verificación que les permiten garantizar que, como mínimo el trabajo realizado abordará los requisitos definidos en el alcance de la auditoria. Lo anterior deja sin herramientas de trabajo que apoyen el proceso de auditoría a los auditores que no pertenezcan a las grandes multinacionales y ocasiona a su paso los siguientes inconvenientes tanto para la empresa auditada como para el auditor:

- Las preguntas pueden ser intimidantes para el auditado.
- El enfoque de la lista de verificación puede ser muy estrecho en su alcance para identificar áreas de problema específicas.
- Un auditor sin experiencia puede no tener la capacidad para comunicar claramente lo que está buscando, si depende significativamente en una lista de verificación para orientar sus interrogaciones.
- Las listas de verificación preparadas con deficiencia pueden lentificar una auditoria debido a la duplicación y la repetición.
- Las listas de verificación genéricas, que no reflejan el sistema de gestión específico de la organización, pueden no agregar ningún valor y pueden interferir con la auditoria.

Por otro lado se encuentra el inconveniente percatado por los docentes del área de sistemas y tecnologías de la información bajo un estudio sobre los perfiles y habilidades con los que contaban los estudiantes que se inscribían en la Facultad de Ingeniería para realizar prácticas profesionales en las empresas de Barrancabermeja. En este estudio se dejó ver la dificultad que presentan los estudiantes de una carrera como ingeniería de sistemas al momento de empezar a hacer sus prácticas en las empresas, ya que al contar únicamente con el conocimiento teórico sobre la utilización de técnicas para el desarrollo de la auditoria como COBIT, no pueden garantizar que su desempeño sea el adecuado y por ende emiten juicios tímidos sobre el comportamiento de una transacción o sobre la utilización adecuada de un control en la organización.

Teniendo en cuenta este enfoque, y si se tiene en cuenta que muchos de los estudiantes se desempeñaran al graduarse

como auditores externos o internos de una empresa, se hace necesario contar con una herramienta que le permita tanto a futuros ingenieros como a profesionales del área de auditoría de sistemas y tecnologías de la información apoyar su desempeño laboral.

Por todo lo anterior, el propósito fundamental de este proyecto de investigación es Desarrollar el Modelado Conceptual y el prototipo del software de apoyo y automatización de las listas de verificación utilizadas en la fase de recopilación de la información del método normativo de auditoría y bajo el estándar Cobit 4.0 (control objetives information and related tecnology), utilizando la metodología de los Sistemas Blandos de Peter Checkland y la metodología de proceso unificado de desarrollo de software con el fin de apoyar a auditores y gerentes en la evaluación de los sistemas y tecnologías de la información asociados a su empresa.

1. ESTADO DEL ARTE

A nivel nacional e internacional los auditores de sistemas de información cuentan con el apoyo de las herramientas CAAT's (Técnicas de Auditoria Asistidas por Computador). Dichas herramientas son de vital importancia para realizar auditorías a sistemas de información e incluyen diversos tipos de herramientas y técnicas, tales como software de auditoría generalizado (ACL, IDEA, etc.), software utilitario (SAS, Excel, Access), datos de prueba, software de monitoreo, y sistemas de auditoría expertos.

Los CAATs se pueden usar en la ejecución de diferentes procedimientos de auditoría, incluyendo:

- Pruebas de detalles de transacciones y balances
- Procedimientos de revisión analíticos
- Pruebas de cumplimiento de controles generales de SI
- Pruebas de cumplimiento de controles de aplicación
- Pruebas de penetración

Los CAATs pueden producir una gran porción de la evidencia de auditoría desarrollada en auditorias de SI, y como resultado, el auditor de SI debe planearlos cuidadosa y profesionalmente.

Las Técnicas de Auditoria Asistidas por Computador (CAATs – Computer Assisted Audit Techniques) son herramientas importantes en la ejecución de las auditorias.

Por otro lado existe también el software generalizado de auditoría que le permite a un auditor acceder a los datos de producción de forma incorporada a los sistemas de aplicación de la organización. Con software de auditoría incorporado, el auditor de SI debe estar involucrado en el diseño de los sistemas y las técnicas que tendrán que ser desarrolladas y mantenidas dentro de los programas/ sistemas de la organización. Además la comunidad de auditores pertenecientes a ISACA pueden acceder al sistema experto de auditoría "MEYCOR COBIT CSA el cual le permite al auditor de sistemas y tecnologías de la información diagnosticar el status de la organización sujeta a medición a nivel de seguridad, calidad, eficacia y eficiencia en Tecnología de la Información (TI) de acuerdo al marco mundial COBIT.

Con respecto a la metodología utilizada en este proyecto para el modelado conceptual, se tuvo como referente la propuesta de investigación del Dr. Martínez (2004), en la cual plantea una metodología para el diseño de sistemas de información basada en el estudio de sistemas blandos.

Para el desarrollo de la herramienta software se aplicaron elementos de la ingeniería del software, la cual según Lewis G. (1994), es un enfoque sistemático del desarrollo, operación, mantenimiento y retiro del software; que provee muchos modelos de procesos de software para el desarrollo. Los modelos más utilizados son: el Cascada También denominado "clásico" Pressman (2006), Bajo este modelo, los procedimientos de la metodología se ordenan en pasos o etapas, las cuales deberán ser seguidas bajo un enfoque secuencial de análisis, diseño y desarrollo. Creado a partir del modelo convencional de "línea de producción" de la ingeniería clásica, este modelo es el más aplicado en el desarrollo de Software; el modelo por prototipos según Pressman (2006), son modelos (no necesariamente productos de software) que permiten estudiar y probar aspectos específicos del producto final (en este caso el producto de software), bajo este modelo, se planifica la aplicación de las diferentes herramientas, para producir elementos de pruebas específicas (interfaz de usuario, mantenedores, procesos) que deberán ser presentados al usuario y confirmados por éste; en el modelo espiral según Sommerville (1988), se pretende optimizar los tiempos y reducir la incertidumbre del proyecto, así, la idea es partir produciendo una pequeña parte del sistema (pero completamente funcional) y una vez completada, se procede a crear una segunda parte, acoplada a la primera, de manera de que en cada iteración, se obtiene una versión aumentada del sistema, y el proceso concluye cuando se considera que el sistema ha alcanzado un nivel de maduración tal, que permite que el trabajo para el que fue creado, sea realizado sin mayores inconvenientes; existe también la Metodología de Proceso Unificado (PU) de Jacobson, I., Booch, G., Rumbaugh, J. (1999), que utiliza el Lenguaje Unificado de Modelado (UML) para preparar todos los esquemas del sistema; utilizando los casos de uso para definir las necesidades de usuario, además utiliza la arquitectura software para describir el sistema en construcción y finalmente es iterativo e incremental.

2. APLICACIÓN DE LA METODOLOGIA DE CHECKLAND.

El desarrollo del proyecto de investigación involucro un híbrido entre dos metodologías que van relacionadas con la solución del problema. Como marco de trabajo se utilizó la Metodología para sistemas blandos (SSM) de Peter Checkland (SSM) (1994) la cual permitió la generación de los modelos conceptuales; para el desarrollo de la herramienta software se utilizó la Metodología de Proceso Unificado de desarrollo de Software de Jacobson, Booch, Rumbaugh (1999).

La SSM está conformada por 7 etapas, cuyo orden puede variar de acuerdo a las características de lo que se quiere estudiar. Aquí se construyó una imagen lo más clara posible del problema, y no se trata de representarla mediante sistemas cuantitativos:

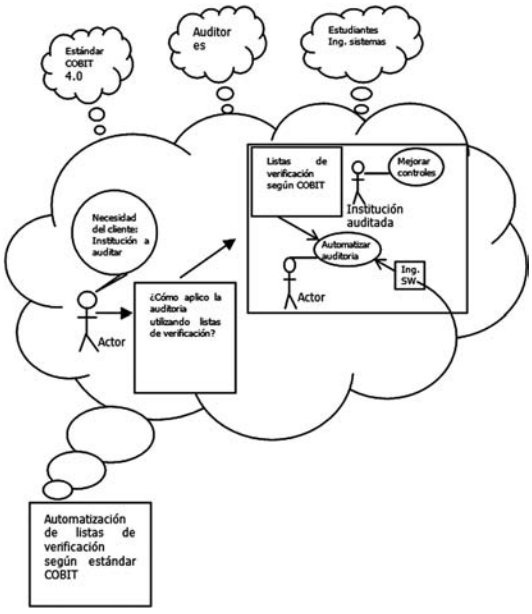
1. Investigar el problema no estructurado: En esta etapa se investigó el problema en sí logrando describirlo y formularlo teniendo una visión del estado actual de la situación en estudio. Se partió del estudio del estándar COBIT el cual posee unos objetivos de control para los cuales el auditor crea listas de verificación que le permite planificar el desarrollo de la auditoria y garantizar el adecuado levantamiento de las evidencias que le habiliten para entregar y apoyar el juicio de auditoría. Pero, ¿Por qué las lista de verificación utilizadas en auditoria de TI no son de fácil acceso? ¿Cómo garantizar el uso de listas de auditoría adecuadas derivadas del estándar COBIT por parte de auditores con poca experiencia? ¿Cómo apoyar tecnológicamente la automatización de las listas de verificación en el proceso de auditoría de TI?

2. Expresar la situación del problema: Las listas de verificación son confidenciales para las empresas y personas expertas que realizan auditorias, lo cual origina que auditores inexpertos no tengan como base dichas listas para diseñar las propias (realidad).

   En el pasado el resultado de una buena auditoria dependía de la experticia de quien la realizaba y no habían muchos estándares que utilizar (pasado), hoy día existen estándares que permiten guiar un proceso de auditoría y dan un marco de trabajo para la misma (presente), y la automatización de las listas de verificación basadas en estándares como COBIT apoyarían a un auditor inexperto.

   A continuación se muestra un Rich Picture ilustrando los aspectos del sistema que son relevantes a la definición del problema.

FIGURA 1. Rich Picture del primer nivel de análisis. Fuente: Autor.



3. Seleccionar una visión de la situación y producir una definición raíz: El propósito de la definición de la raíz es expresar la función central del sistema, esta raíz se expresa como un proceso de transformación que toma una entidad como entrada de información, cambia o transforma a esa entidad, y produce una nueva forma de entidad.

Se elaboraron definiciones según los diferentes weltanschauung involucrados con el fin de identificar que procesos de transformación son prioritarios llevar a la realidad. La construcción de estas definiciones se fundamentó en seis factores agrupados bajo el nombre CATWOE:

- Cliente: Empresas a auditar que poseen Tecnologías de la información y comunicación
- Agente: Auditores internos y externos
- Proceso de transformación: listas de verificación automatizadas según estándar COBIT con elementos de la ingeniería de software.
- Weltanschauung: Utilizar los objetivos de control del estándar COBIT, Opinión de auditores, Opinión de estudiantes de ingeniería de sistemas que han tenido experiencias con auditorias.
- Dueño: Institución propietaria de la auditoria: Universidad Cooperativa de Colombia
- Apremios ambientales: estándares, leyes, tipos de auditores, tipos de empresas

Del anterior análisis se estableció que la definición raíz del sistema es la automatización de la listas de verificación

según el estándar COBIT y el aporte de auditores o estudiantes de ingeniería de sistemas que realizan auditorias, con el fin de apoyar a auditores y gerentes en la evaluación de los sistemas y tecnologías de la información asociados a su empresa. La automatización de las listas de verificación requiere del desarrollo de alguna herramienta software que almacene dichas listas en una base de datos y permita la tabulación de los resultados obtenidos.

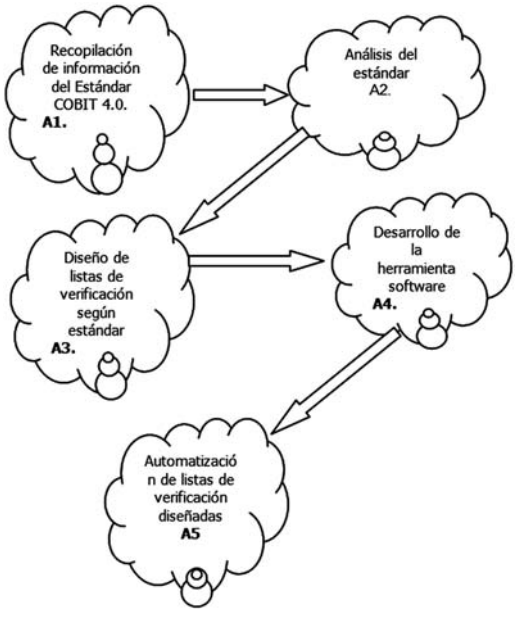
4. Confección y verificación de modelos conceptuales: Partiendo de la definición de la raíz, se elaboró el modelo conceptual que representa las actividades necesarias para lograr la perspectiva descrita en el paso anterior (paso 3).

Para lograr lo anterior primero se creó una tabla en la cual se describen dichas actividades y luego procedió a diseñar un Rich Picture que las representa como se muestra en la figura 2.

TABLA 1. Actividades que conforman el Modelo Conceptual.

Actividad	Descripción
A1	Recopilación de información del Estándar COBIT 4.0
A2	Análisis del estándar
A3	Diseño de listas de verificación según estándar
A4	Desarrollo de la herramienta software
A5	Automatización de listas de verificación diseñadas

FIGURA 2. Rich Picture que representa el modelo conceptual. Fuente: Autor.



5. Comparación de los modelos conceptuales con la realidad, es decir etapa 4 con la etapa 2: En esta etapa los modelos construidos en al etapa 4 (elaboración de modelos conceptuales, a través de una malla "PERT") se compararon con la expresión real del mundo, de la etapa 2 (diagrama, figura 1), se detectaron las diferencias y similitudes entre los modelos conceptuales y lo que existe en la actualidad del sistema.

Del anterior análisis comparativo de concluye que hay que puntualizar en la actividad 4 (A4), desglosando a un mayor nivel de detalle dicha actividad puesto que para obtener la herramienta software que permita la automatización de las listas de verificación no necesariamente se deben aplicar elementos de la ingeniería de software y en el modelo conceptual de la realidad (etapa 2) se plantea que para la automatización de las lista que permitan obtener buenos resultados en el proceso de auditoría se deben utilizar principios de la ingeniería de software como por lo menos una metodología de desarrollo de software que guie el proceso.

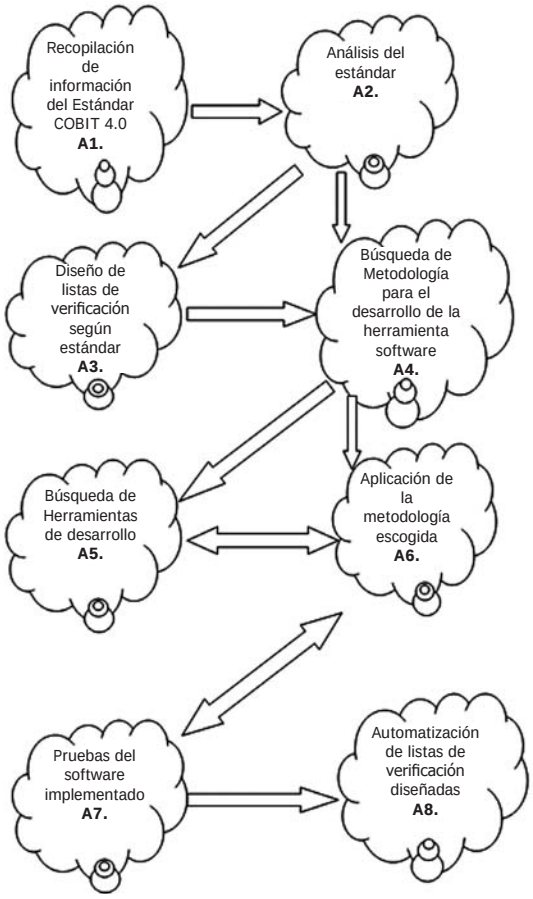
6. Diseño de cambios deseables, viables y factibles: Se detectaron los cambios que eran posible llevar a cabo en la realidad y en la etapa siguiente. Estos cambios se detectaron de las diferencias emergidas entre la situación actual, y los modelos conceptuales, se propusieron cambios tendientes a superarlas, dichos cambios involucraron a los diversos actores del proyecto.

La actividad 4 fue desglosada de forma más específica dando como resultado que aparecieran cuatro actividades como se muestra en la tabla 2.

TABLA 2. Actividades de mejora que conforman el Modelo Conceptual.

Actividades	Descripción
A1	Recopilación de información del Estándar COBIT 4.0
A2	Análisis del estándar
A3	Diseño de listas de verificación según estándar
A4	Búsqueda de Metodología para el desarrollo de la herramienta software
A5	Búsqueda de herramientas de desarrollo
A6	Aplicación de la metodología escogida
A7	Pruebas del software implementado
A8	Automatización de listas de verificación diseñadas

FIGURA 3. Rich Picture con las acciones de mejora. Fuente: Autor.



7. Acciones para mejorar la situación del problema: Es decir la implantación de cambios, que fueron detectados en la etapa 6. El siguiente Rich Picture representa las modificaciones realizadas en el paso anterior, donde son claros los cambios que sufrió la actividad 4.

Para la solución del problema se decidió darle un nivel mayor de detalle a la actividad A4, A5 y A6 que son las que permiten llegar a la actividad 8 (A8). Para lograr lo anterior se estableció utilizar la metodología de Proceso Unificado de Desarrollo de software con la cual se obtendría la herramienta software.

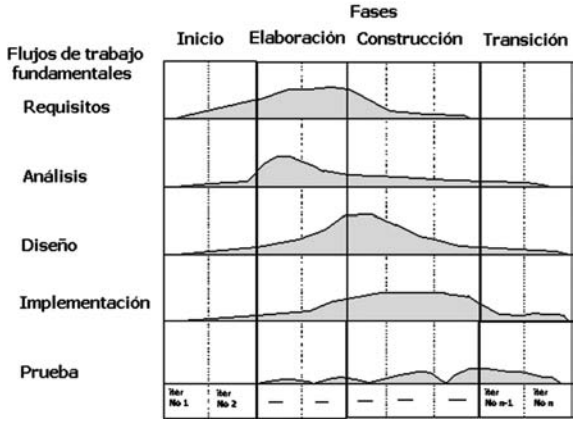
3. APLICACIÓN DEL PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE

La ingeniería del Software ofrece las metodologías, herramientas y técnicas para desarrollar software. Estas metodologías son llamadas también modelos de procesos de software, Pressman (2006), los cuales dan las pautas para obtener un software y sus productos asociados como son la documentación, código fuente ,etc.

Dentro del mundo de las metodologías y procesos de desarrollo de software se encuentra el Proceso Unificado de desarrollo de Software, la cual se seleccionó y utilizó por parte de los participantes del proyectos por tener alguna experiencia en su utilización; y se aplicó para desglosar la actividad A6 y poder llegar a la actividad A8 de la fase 7 del modelado conceptual anteriormente descrito.

La Metodología de Proceso Unificado (PU) de Jacobson, Booch, Rumbaugh (1999) utiliza el Lenguaje Unificado de Modelado (UML) para preparar todos los esquemas del sistema; utilizando los casos de uso para definir las necesidades de usuario, además utiliza la arquitectura software para describir el sistema en construcción y finalmente es iterativo e incremental, que estudia paso a paso el incremento y mejoras que se presentan en la construcción de sistema software.

FIGURA 4. Metodología de Proceso Unificado de Desarrollo de Software. Fuente: Jacobson, I., Booch, G., Rumbaugh, J. (1999).

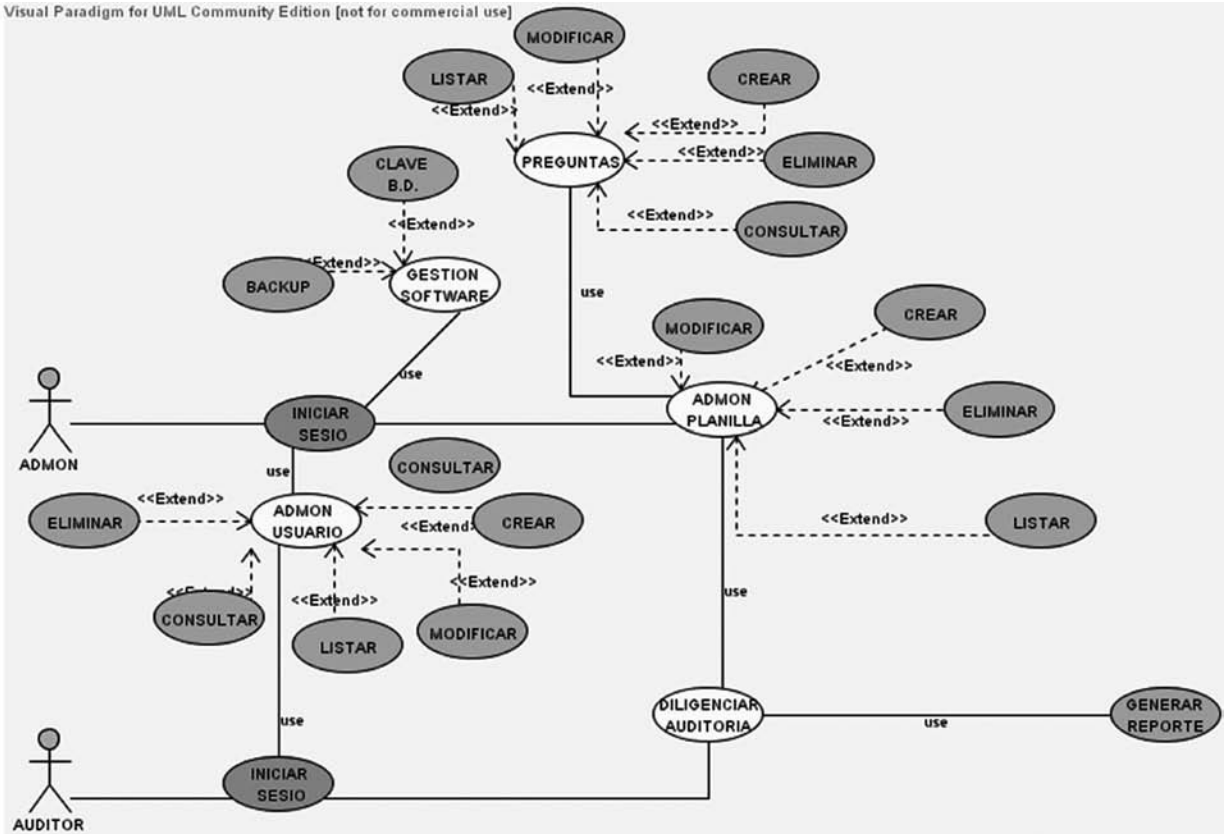


De acuerdo a la metodología se empezó con la fase de inicio en la cual se hizo el levantamiento de requisitos, definiendo los requerimientos funcionales y no funcionales, analizando y diseñando los casos de uso del sistema.

En esta fase se revisaron las listas de verificación realizadas por los estudiantes de sistemas de los semestres anteriores, para obtener evidencias de las listas de verificación que ya se habían venido trabajando en la asignatura Auditoria de Sistemas, para tenerlas como base para realizar las listas de verificación que se montarán en el software. Se hizo un estudio sobre el estándar Cobit para poder generar las listas de la verificación.

Con respecto a esta fase se creó el diagrama de casos de uso que especifica las funcionalidades del sistema, donde se definen los actores que participan. El siguiente diagrama de casos de uso muestra las funcionalidades del software:

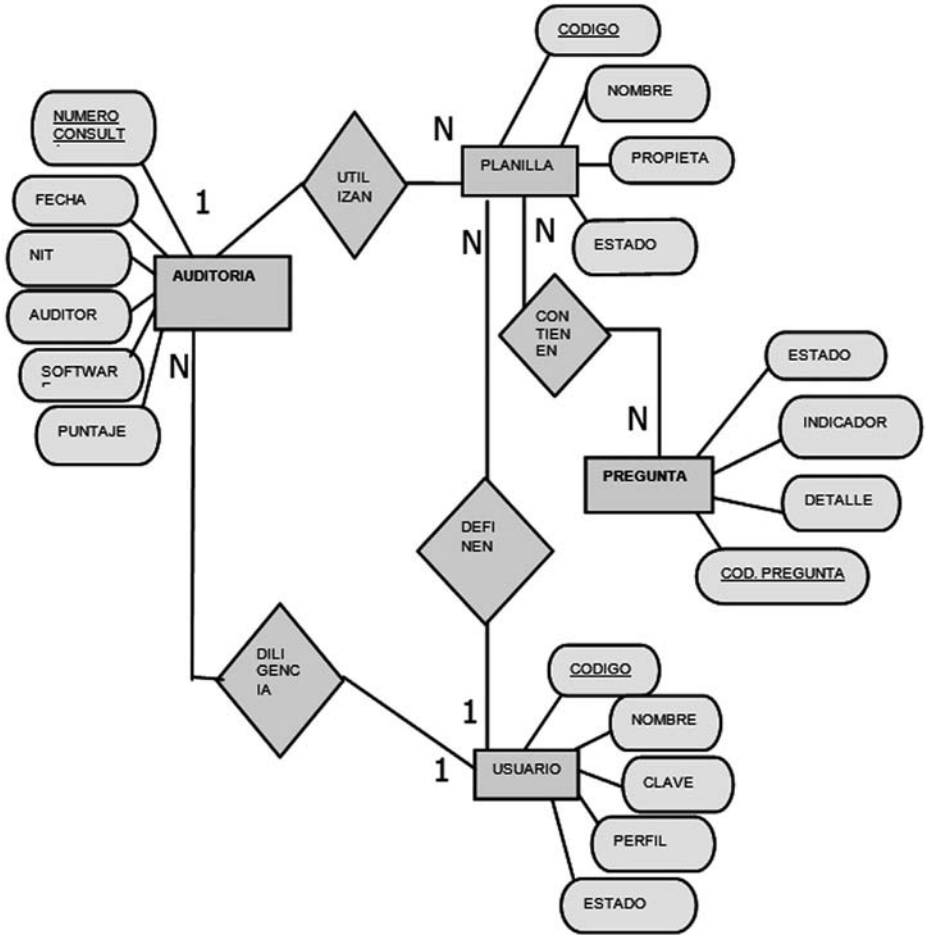
FIGURA 5. Diagrama de Casos de uso general del sistema. Fuente: Autor.



Luego en la fase de elaboración se analizó y diseño la arquitectura del sistema, la cual fue concebida como tres capas, se definieron las herramientas de programación necesarias para implementar el aplicativo, se elaboró el diseño conceptual y lógico la base de datos que soportaría la herramienta software, así como los diagramas de actividades que describían a fondo los casos de uso definidos anteriormente y los diagramas de secuencia que mostraban la interacción entre las clases del sistema.

A continuación se presenta el diagrama entidad relación o modelo conceptual de la base de datos que soporta el software.

Figura 6. Fuente: Autor. Modelo Conceptual Base de Datos del sistema.



Este modelo conceptual de la base de datos dio las pautas para el modelo lógico y el modelo físico de la base de datos, los cuales se basan en el modelo relacional.

Los diagramas de actividades describieron en detalle los casos de uso, mostrando como era el flujo de eventos y el flujo alterno. Por otra parte los diagramas de secuencia describían como era la interacción entre las clases de interfaz, control y entidad.

Posteriormente en la fase de construcción se determinó que la herramienta tendría dos incrementos relacionados con los módulos del auditor y del administrador. A cada incremento se le aplicó un ciclo de vida terminado con las pruebas de cada uno y su funcionamiento.

Los productos de esta fase fueron el montaje de la base de datos en Mysql 5.0, que corresponde al diseño físico de la misma; también se construyeron las páginas dinámicas que correspondían con los casos de uso utilizando los lenguajes PHP, HTML y JavaScript. Se utilizó el servidor Web Apache con el cual se hizo el montaje a nivel local de la aplicación. Cada página fue probada de acuerdo a casos de prueba diseñados para cada caso de uso. Cada incremento arrojó un modulo de la aplicación funcional, donde se correspondía con un ciclo de vida clásico.

Finalmente en la fase de transición se realizaron pruebas de funcionamiento con la aplicación de una auditoria a un aplicativo específico para medir su funcionalidad. Esta fase arrojó el manual de usuario y el manual de instalación.

4. CONCLUSIONES

La Metodología de Sistemas Blandos desarrollada por Checkland se puede aplicar en Ingeniería de Software

en lo concerniente al modelado conceptual del software, ayudando a entender y mejorar los sistemas en donde existe actividad humana, ya que con el análisis realizado se tiene la certeza de considerar las variables relevantes del problema que se verán reflejadas en el modelo para la obtención de una solución más cercana a la realidad, ya que no se ve al proceso de manera asilada sino como un proceso que involucra el entorno. Contó el desarrollo de siete etapas, que fueron descritas y desarrolladas anteriormente permitiendo la construcción de una imagen de la situación del problema, que no requirió su representación mediante sistemas cuantitativos. Esta metodología esta siendo usada actualmente en el desarrollo de trabajos relacionados con la Ingeniería de Sistemas alrededor del mundo teniendo magníficos resultados.

La utilización de la metodología de sistemas blandos propuesta por Checkland con la Metodología de Proceso Unificado de Desarrollo de Software permitió establecer un camino para la elaboración de este trabajo acorde a las necesidades de los involucrados en la situación en estudio, guiados por modelos conceptuales que garantizaron y brindaron las pautas o actividades a realizar para obtener una solución al problema planteado, introduciendo el Proceso Unificado como parte fundamental del marco de trabajo para brindar una solución informática que se ajusta a las necesidades o requerimientos funcionales del sistema a desarrollar.

5. REFERENCIAS BIBLIOGRAFICAS

[1] Checkland, Peter. (1994). Metodología de los Sistemas Blandos. 1a. ed., Noriega Editores, México.

[2] Gómez, Luís Carlos. (2000). Auditoria de Sistemas de Información. Editorial UIS.

[3] Information System Audit and Control Association. (ISACA). (2006). Control Objectives for Information and Related Technology 4.0. (COBIT 4.0).

[4] Ivar Jacobson, Grady Booch, James Rumbaugh. (1999). **"**El Proceso Unificado de Desarrollo Software**",** Addison Wesley.

[5] Lewis G. 1994. "What is Software Engineering?" DataPro (4015). Feb 1994. pp. 1-10.

[6] Martínez, Andrés. (2004). Una Metodología para el diseño de sistemas de información, basada en el estudio de sistemas blandos. Revista Espacios. ISSN 0798-1015. Caracas.

[7] Pressman, Roger. (2006). Ingeniería del Software. Un Enfoque Práctico. Sexta Edición. McGraw-Hill.

[8] Sommerville, Ian (1988). Ingeniería de Software. Sistemas Técnicos de Edición.