Universidad Nacional Autónoma de Nicaragua

UNAN-MANAGUA

Facultad de Ciencias e Ingenierías

Departamento de Computación

"Automatización del proceso de auditoría a las tecnologías de la información y las comunicaciones"

SEMINARIO MONOGRÁFICO

Para obtener el título de:

LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN

Presentan:

Br. Miurel Yaneisy Gómez Bravo.

Br. Elvin Antonio Mendoza.

Br. Francisco Rigoberto Pérez López.

Docente:

MSc. Alejandro Antonio Ruiz Segovia.

Managua, Nicaragua, Noviembre, 2010

Tema

Automatización del proceso de auditoría a las tecnologías de la información y las comunicaciones

Subtema

Automatización de los procedimientos de auditoría de sistemas relativos a la extracción y análisis de datos.

Dedicatoria

Dedicamos esta investigación primeramente a nuestro DIOS que nos da el aliento de vida cada día para cumplir nuestras metas y a todas las personas que nos brindaron su apoyo incondicional para la elaboración de este proyecto, nuestros padres, docentes y amigos.

Agradecimiento

Los autores citados en la parte bibliográfica de este trabajo contribuyeron en la creación y finalización de esta investigación, y por consiguiente es nuestra obligación agradecerles, sin embargo debemos un especial agradecimiento a quien nos ha prestado una especial contribución en forma de revisión y corrección, así como numerosas sugerencias y consejos que nos brindo para mejorar nuestro trabajo, por lo que deseamos expresar nuestro reconocimiento y agradecimiento especial al Lic. Alejandro Antonio Ruiz Segovia.

Índice

Resumen

Este trabajo aborda los procedimientos de auditoría informática relacionados con la extracción y análisis de datos con el fin de desarrollar una herramienta que automatice estos procedimientos y que además agilice la tarea del auditor informático.

El primer capítulo de este documento muestra los objetivos propuestos en esta investigación.

El segundo capítulo trata de la base conceptual de la investigación: la definición de sistema, sistema informático y sistema de información, daremos a conocer la diferencia que existe entre sistema informático y sistema de información, las actividades básicas del sistema de información y los tipos de sistemas de información dentro del área de auditoría informática, también presentaremos la definición de la auditoría informática, objetivo fundamental, los tipos de auditoría, así como también sus fases, normas, técnicas y procedimientos, luego definiremos cada una de las herramientas que se utilizaron tanto para el análisis como para el diseño de nuestro proyecto, también presentaremos una pequeña introducción a lo que son los programas de Extracción y Análisis de datos.

Los resultados obtenidos de esta investigación se presentan en el capítulo 3, donde se incluyen la estrategia de prueba para el análisis de datos generados en los sistemas de información, las técnicas y procedimientos de auditoría aplicadas a la herramienta, los objetivos de control para la información y tecnologías relacionadas (Cobit 4.0), también daremos a conocer las herramientas CASE utilizadas para el diseño de la herramienta.

En el capítulo 4 se expondrán los criterios de factibilidad técnica, económica y operacional, relacionadas con el desarrollo e implementación de nuestra herramienta.

Introducción

Los sistemas informáticos son una herramienta que facilita el trabajo relacionado con procesar, ordenar, manipular y controlar datos con gran velocidad y precisión, por lo que cada día son más organizaciones las que los implementan para el mejoramiento de la eficiencia de su operatividad. A medida que se utilizan este tipo de sistemas incrementan el volumen de datos que almacenan, por lo que para la acertada toma de decisiones es necesaria la evaluación de la información producida.

Las funciones de auditoría no solo se basan en evaluar la eficacia, eficiencia y economía de una organización, todo esto con referencia al área contable, sino que también se encarga de evaluar el área informática y la información que los sistemas producen mediante la aplicación de técnicas informáticas con el fin de mejorar la confiabilidad y seguridad implementadas. La auditoría informática también permite evaluar el funcionamiento de centros de Procesamiento de datos (CPD), hardware y software.

La realización de los procedimientos de auditoría (procedimientos manuales) y sus resultados demandan mucho tiempo, por lo que desarrollaremos una herramienta informática útil y necesaria que permita al auditor analizar los datos almacenados en menor tiempo y con mayor confiabilidad.

Esta herramienta además de fácil de usar, le permitirá al usuario leer y comparar datos de la empresa, siendo los archivos de origen de solo lectura, ya que se creará una copia para que el auditor pueda trabajar en ella, de tal forma que luego pueda comparar el resultado de las operaciones realizadas y verificar que no existan variaciones en algunos de estos archivos. Lo mencionado anteriormente ayudará a alcanzar el objetivo de la auditoria, el cual consiste en emitir recomendaciones a la alta gerencia para mejorar el control interno en tecnología informática.

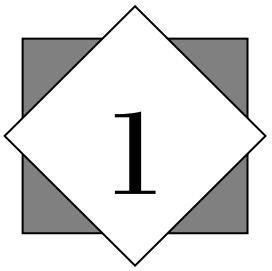
Justificación

En la actualidad existen herramientas informáticas que permiten el análisis y la extracción de datos desde otros programas, con la certeza que el archivo original no sufrirá ningún cambio, ya que inicialmente se crea una copia de este para trabajar en el, algunas de estas herramientas que utilizan este tipo de metodología son: ACL 9.0, IDEA 7.0, etc., sin embargo debido a su costo elevado se hace difícil la obtención legal de estas, de igual forma existen herramientas como Microsoft Excel, que se distribuye en paquetes de Office de Microsoft y que ayudan al análisis de datos, pero que carece de un control de seguridad y de bitácora de las tareas realizadas sobre la información a evaluar por el auditor, problema que se trata de solucionar con el desarrollo de este proyecto el cual será funcional para una empresa u organización.

Al llevar a cabo el diseño de una herramienta que posea características similares a programas de extracción y análisis de datos que están actualmente en el mercado, y que a la vez incorpore nuevas características, como el control de acceso al sistema y bitácora de acciones por usuarios, pretendemos proporcionarles otra opción a los usuarios de este tipo de sistemas (en su mayoría auditores), de modo que si antes se valían de diferentes herramientas para concluir su trabajo, con la implementación de esta novedosa herramienta podrá realizar diferentes funciones desde un mismo entorno de trabajo tales como: buscar, filtrar y comparar datos, extraerlos y presentarlos en un formato con el que se puedan efectuar revisiones con mayor rapidez y confiabilidad, de forma tal que se incremente la eficiencia de la función de auditoría.

Es importante señalar que se incluye como parte de esta herramienta un módulo de seguridad que garantizará el acceso a la información, solo a personal autorizado mediante la solicitud de nombre de usuario y contraseña.

OBJETIVOS



1.1.- Objetivo General

Desarrollar una Herramienta para la automatización de los procedimientos de auditoría de sistemas relativos a la extracción y análisis de datos.

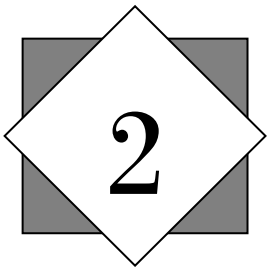
1.2.- Objetivos Específicos

1. Definir una estrategia de prueba para el análisis de la calidad de los datos generados en los sistemas informáticos basado en los criterios de información de COBIT 4.0 y las normativas de la Contraloría General de la República de Nicaragua.

2. Automatizar la estrategia de prueba, mediante el desarrollo de un sistema informático dirigido a disminuir el tiempo de aplicación de las pruebas de auditoría y emitir en menor tiempo los resultados de los análisis realizados.

3. Implementar un nivel de seguridad para el sistema, que garantice la confidencialidad de los proyectos realizados por cada uno de los usuarios del sistema, así como mantener un registro de los accesos y las acciones de los usuarios sobre el mismo.

MARCO TEORICO



1. Sistemas

1.1- Definición de Sistema.

Según Ian Sommerville **sistema** es: "*una colección de componentes interrelacionados que trabajan conjuntamente para cumplir algún objetivo*".[1]

1.2.- Definición de Sistema informático.

Un sistema informático es un conjunto de partes que funcionan relacionándose entre sí con un objetivo preciso. Sus partes son: hardware, software y las personas que lo usan. Por ejemplo, una computadora, sus dispositivos periféricos y la persona que la maneja, pueden constituir un sistema informático.

Un sistema informático puede formar parte de un sistema de información; en este último la información, uso y acceso a la misma, no necesariamente está informatizada. Por ejemplo, el sistema de archivo de libros de una biblioteca y su actividad en general es un sistema de información. Si dentro del sistema de información hay computadoras que ayudan en la tarea de organizar la biblioteca, entonces ese es un sistema informático.

1.3.- Sistema de información (SI).

1.3.1.- Definición de sistema de información: Es un conjunto de componentes interrelacionados que permiten capturar, procesar, almacenar y distribuir la información para apoyar la toma de decisiones y el control de una institución.

[1] *Sommerville, I.," Ingeniería del Software", Edo. Addison-Wesley. 6ta. Edición. 2002.*

1.3.2.- Actividades básicas de un sistema de información:

1. **Entrada de Información:** Es el proceso mediante el cual el sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos. Este último se denomina interfaces automáticas.
2. **Almacenamiento de información:** El almacenamiento es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos.
3. **Procesamiento de Información:** Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecidas. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada para la toma de decisiones, lo que hace posible, entre otras cosas, que un tomador de decisiones genere una proyección financiera a partir de los datos que contiene un estado de resultados o un balance general de un año base.
4. **Salida de Información:** La salida es la capacidad de un SI para sacar la información procesada o bien datos de entrada al exterior.

1.3.3.-Tipos de sistemas de información que existen dentro de la auditoría informática de sistemas son tres: SI informacionales, SI operacionales y SI de comunicación.

Los SI de comunicación tienen como único objetivo poner en contacto a todos los miembros de la organización, sirven de apoyo en las actividades de comunicación

que se presentan en la organización, en cambio el SI informacional sirve de apoyo en la toma de decisiones de una organización. Con el fin de cumplir este objetivo se debe de disponer de un almacén de datos y de una herramienta que permita acceder a dichos datos. A las herramientas de acceso se les llama herramientas de soporte o apoyo en la toma de decisiones y son sistemas informáticos (Sistemas Operacionales) especializados en extraer información de un almacén de datos para ofrecérsela al usuario, en otra palabras un SI operacional trata todas aquellas actividades que tienen que ver con la recopilación, selección y manipulación de la información.

1.4.- Diferencia entre sistema informático y sistema de información.

- En un sistema informático se utilizan computadoras para almacenar, procesar y/o acceder a información.
- En un sistema de información se pueden utilizar computadoras, pero no es necesario. El acceso a la información puede ser físico (por ejemplo, una persona se encarga de buscar en un archivador).
- Tanto el sistema informático como el sistema de información, incluyen a las personas que acceden o producen información dentro del sistema. Las personas tienen que capacitarse para entender el funcionamiento y procedimientos que soporta sistema.
- Ambos sistemas tienen un propósito. Por ejemplo, gestionar el acceso y distribución de libros una biblioteca, administrar la entrada/salida de mercadería, personal y otros recursos de un comercio, etc.

2.-Auditoria Informática

2.1.-Definición de auditoría informática:

La auditoría en informática es la revisión y la evaluación de los controles, sistemas, procedimientos de informática; de los equipos de cómputo, su utilización, eficiencia y seguridad, de la organización que participan en el procesamiento de la información, a fin de que por medio del señalamiento de cursos alternativos se logre una utilización más eficiente y segura de la información que servirá para una adecuada toma de decisiones.

2.1.1.- Auditoría de los sistemas de información: ISACA[2] la define como cualquier auditoría que abarca la revisión y evaluación de todos los aspectos (o de cualquier porción de ellos) de los sistemas automáticos de procesamiento de la información, incluidos los procedimientos no automáticos relacionados con ellos y las interfaces correspondientes.

2.2.-Alcance de la auditoría informática:

La auditoría en informática deberá comprender no sólo la evaluación de los equipos de cómputo, de un sistema o procedimiento específico, sino que además habrá de evaluar los sistemas de información en general desde sus entradas, procedimientos, controles, archivos, seguridad y obtención de información. La auditoría en informática es de vital importancia para el buen desempeño de los sistemas de información, ya que proporciona los controles necesarios para que los sistemas sean confiables y con un buen nivel de seguridad. Además debe evaluar todo (informática, organización de centros de información, hardware y software).

2.3.- Objetivo fundamental de la auditoría informática.

El objetivo fundamental de la auditoria informática es la operatividad, la cual consiste en que para efectuar una auditoría es necesario que el área auditada

[2] *Information System Audit and Control Association*

funcione con normalidad, no es aceptable detener la jornada de trabajo de esta para que el auditor pueda realizar su labor.

2.4.- Fases de la auditoría informática según manual de la Contraloría General de la república de Nicaragua (CGR).

i. Planeación de la auditoría: Su objetivo es obtener un conocimiento de la Entidad u Organismo sobre cómo operan los sistemas de información que se han de incluir en la revisión y el ambiente de control en cuanto a las fuentes de información, evaluación de riesgos inherente al control e identificación de controles claves, para definir el enfoque de auditoría que se aplicará, determinar los procedimientos de auditoría específicos a realizar, seleccionar el personal, determinar tiempo y costo y preparar los programas de auditoría respectivos.

ii. Ejecución del plan de auditoría: Su objetivo es la aplicación de las técnicas de auditoría asistidas por computador, procedimientos de pruebas y evaluación de controles para los riesgos de aplicación y del Centro de Cómputo.

iii. Conclusión y preparación de informe

Su objetivo es obtener una conclusión general sobre la auditoría realizada por lo que el informe debe contener:

1. Objetivos de auditoría cubiertos;
2. Naturaleza y alcance de los procedimientos aplicados;
3. Hallazgos detectados y recomendaciones de auditoría ;
4. Comentarios de los Funcionarios de la Entidad u Organismo;
5. Conclusión.

2.5.- Normas de auditoría informática.

Definición: Son los requisitos mínimos de calidad relativos a la personalidad del auditor, al trabajo que desempeña y a la información que rinde como resultado de este trabajo.

Las normas de auditoría se clasifican en:

a) **Normas personales:** Son cualidades que el auditor debe tener para ejercer sin dolor y ser imparcial a la hora de dar sus sugerencias, todo esto basado en sus conocimientos profesionales y entrenamiento técnico.
b) **Normas de ejecución del trabajo:** Son la planificación de los métodos y procedimientos, así como los papeles de trabajo a aplicar dentro de la auditoría.
c) **Normas de información:** También conocidas como informe o dictamen, en el cual se detalla todo lo evaluado por el auditor.

2.6.- Técnicas de auditoría informática.

2.6.1.-Definición de técnicas de auditoría informática: Según el Instituto Mexicano de Contadores Públicos (IMCP) en su libro *Normas y procedimientos de auditoría* las técnicas se clasifican generalmente con base en la acción que se va a efectuar, estas acciones pueden ser oculares, verbales, por escrito, por revisión del contenido de documentos y por examen físico.

2.6.2.-Clasificación de las Técnicas de auditoría informática: Siguiendo la clasificación anterior las técnicas de auditoría se agrupan específicamente de la siguiente manera:

- **Inspección**. De documentos, procedimientos, activos, para verificar su existencia, más que para determinar la forma en que se están utilizando.
- **Observación.** De procesos o acciones.
- **Investigación o indagación.**

- **Confirmación.** Ratificar datos.
- **Cálculo.** Precisión matemática
- **Revisión Analítica.** Investigación de fluctuaciones o partidas poco usuales.

2.6.3.-Definición de las Técnicas de auditoría Asistidas por Computador (TAAC's). La norma SAP 1009 las define como programas de computador y datos que el auditor usa como parte de los procedimientos de auditoría para procesar datos de significancia en un sistema de información.

2.6.4.-Normas emitidas para el uso de las TAAC's.

El desarrollo de una auditoría se basa en la aplicación de normas, técnicas y procedimientos de auditoría. Las Normas Internacionales de auditoría emitidas por IFAC (International Federation of Accountants) en la NIA (Norma Internacional de auditoría o International Standards on Auditing, ISA), establecen que para contemplar el uso de las técnicas manuales de auditoría los auditores se pueden apoyar en otras técnicas asistidas por PC.

La norma SAP 1009 (Statement of Auditing Practice) denominada Computer Assisted Audit Techniques (CAATs) o Técnicas de auditoría Asistidas por Computador (TAACs), plantea la importancia del uso de TAACs en auditoría s en un entorno de sistemas de información por computadora.

NOTA: Es importante resaltar que las TAACs se apoyan en programas de computador, los cuales deben ser desarrollados usando las metodologías de ingeniería de software. Es responsabilidad del auditor garantizar el control sobre estos programas, sería deficiente si confiara la recolección de evidencia a una herramienta o técnica que funciona errónea o deficientemente. Por lo anterior es su deber participar en todas las fases del desarrollo (determinación de requerimientos, diseño, construcción, prueba e implantación). Debe el auditor

mantener bajo su vigilancia estas aplicaciones para evitar que sufran modificaciones no autorizadas.

2.7.- Procedimientos de auditoría:

2.7.1.- Definición de procedimientos de auditoría: Al conjunto de técnicas de investigación aplicables a un grupo de hechos o circunstancias que nos sirven para fundamentar la opinión del auditor dentro de una auditoría, se les dan el nombre de procedimientos de auditoría en informática.

La norma SAP 1009 describe los procedimientos de auditoría en que pueden ser usados las TAAC's:

1. Pruebas de detalles de transacciones y balances (Recálculos de intereses, extracción de ventas por encima de cierto valor, etc.)
2. Procedimientos analíticos, por ejemplo identificación de inconsistencias o fluctuaciones significativas.
3. Pruebas de controles generales, tales como configuraciones en sistemas operativos, procedimientos de acceso al sistema, comparación de códigos y versiones,
4. Programas de muestreo para extraer datos.
5. Pruebas de control en aplicaciones.
6. Recálculos.

2.8.- Documentación de las TAAC's

La Norma Internacional especifica que el estándar de papeles de trabajo es consistente con el de la auditoría como un todo (incluido en NIA 9, Documentación). Puede ser conveniente mantener los papeles técnicos que se refieren al uso de la TAAC separados de los otros papeles de trabajo de la auditoría. A continuación detallaremos como la documentación de las TAACs cubren todas las fases de auditoría (planeación, ejecución, conclusión):

a. Planeación:

- Objetivos de la TAAC.
- TAAC a utilizar.
- Controles que se van a implementar.
- Personal involucrado, cronograma y costos.

b. Ejecución:

- Procedimiento de preparación y prueba de la TAAC.
- Detalles de las pruebas realizadas.
- Detalles de datos de entrada, procesamiento y datos de salida.

c. Conclusión y preparación del informe:

- Datos de salida proporcionados.
- Descripción del trabajo de análisis sobre salidas.
- Conclusión de la auditoría.
- Recomendaciones.

Si es el caso, se incluyen sugerencias para mejorar la TAAC.

3.-Ingeniería del software

3.1.- Unified Modeling Language (UML 2.0 según Booch, Jacobson, Rumbaugh)

El Lenguaje Unificado de Modelado (Unified Modeling Language, UML) es un lenguaje estándar para escribir planos de software. UML puede utilizarse para visualizar, especificar, construir y documentar los artefactos de un sistema que involucre una gran cantidad de software. El vocabulario y las reglas de un lenguaje como UML indican cómo crear y leer modelos bien formados, pero no dicen qué modelos se deben crear ni cuando se deberían crear. Un proceso bien definido guiará a sus usuarios al decidir qué artefactos producir, qué actividades y qué personal se emplea para crearlos y gestionarlos, y cómo usar esos artefactos para medir y controlar el proyecto de forma global.

Una organización de software que trabaje bien, produce toda clase de artefactos además de código ejecutable. Estos artefactos incluyen (aunque no se limitan a):

- Requisitos.
- Arquitectura.
- Diseño.
- Código fuente.
- Planificación de proyectos.
- Pruebas.
- Prototipos.
- Versiones.

Tales artefactos no son tan sólo los entregables de un proyecto, también son críticos para el control, la medición y comunicación que requiere un sistema durante su desarrollo y después su despliegue. UML cubre la documentación de la arquitectura de un sistema y todos sus detalles. UML también proporciona un lenguaje para expresar requisitos y pruebas. Por último UML proporciona un

lenguaje para modelar las actividades de planificación de proyectos y gestión de versiones.

3.1.1.- Diagramas Estructurales

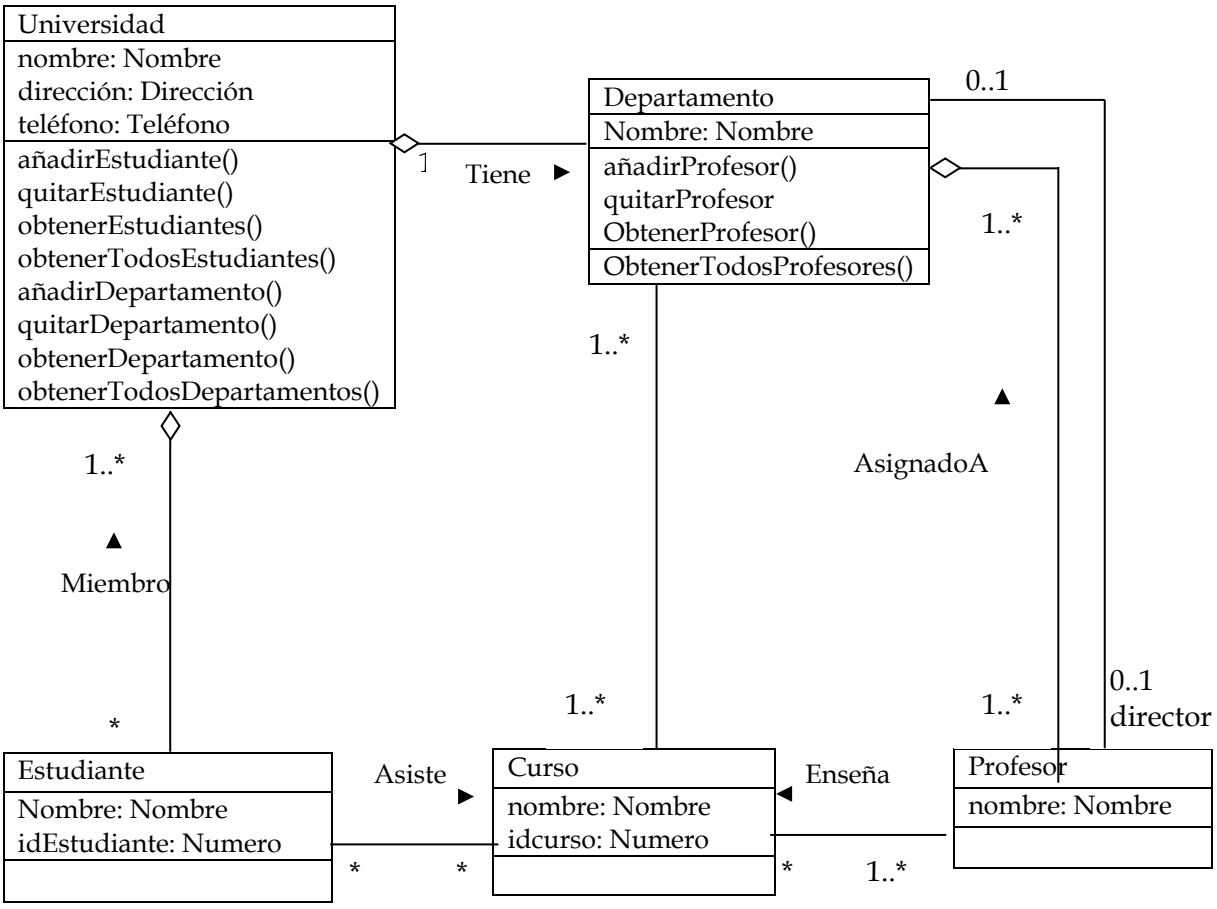
Los diagramas estructurales de UML existen para visualizar, especificar, construir y documentar los aspectos estáticos de un sistema. Los aspectos estáticos de un sistema software incluyen la existencia y ubicación de clases, interfaces, colaboraciones, componentes y nodos. Los diagramas estructurales de UML se organizan en líneas generales alrededor de los principales grupos de elementos que aparecen al modelar un sistema:

1. Diagrama de clases. Clases, interfaces y colaboraciones.
2. Diagrama de componentes. Componentes
3. Diagrama de estructura compuesta. Estructura interna.
4. Diagrama de objetos. Objetos
5. Diagrama de despliegue. Nodos
6. Diagrama de artefactos. Artefactos

1. Diagramas de clases. Un diagrama de clases presenta un conjunto de clases, interfaces y colaboraciones, y las relaciones entre ellas. Los diagramas de clases se utilizan para describir la vista de diseño estática de un sistema.

La siguiente figura muestra un conjunto de clases extraídas de un sistema de información de una universidad. Esta figura muestra las clases a un nivel suficientemente detallado para construir una base de datos física. Comenzando por la parte inferior izquierda de este diagrama, se encuentran las clases *Estudiante, Curso y Profesor*. Hay una asociación entre *Estudiante y Curso*, que especifica que los estudiantes asisten a los cursos.

Además cada estudiante puede asistir a cualquier número de cursos y cada curso puede tener cualquier número de estudiantes. Este diagrama muestra los atributos de las seis clases.



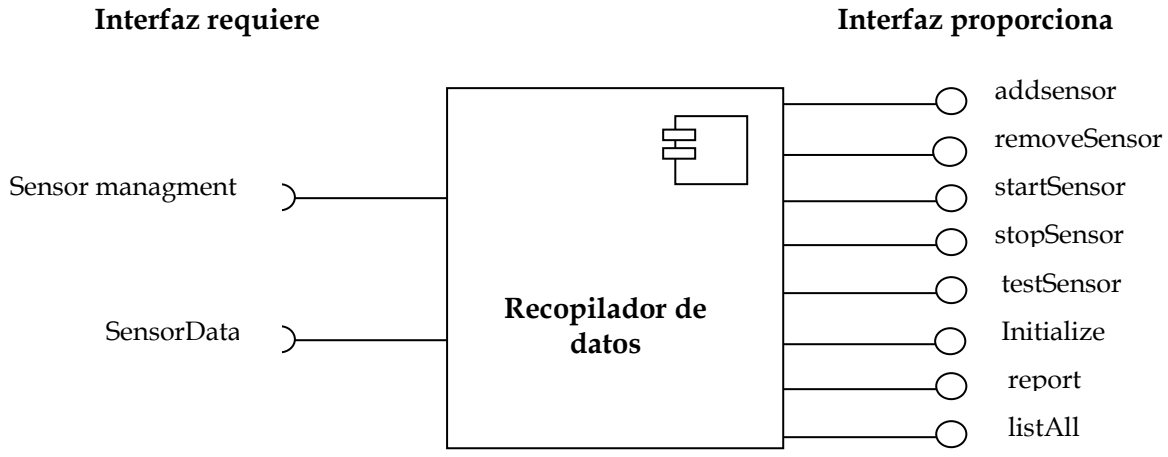
Ejemplo de Diagrama de clases

Dos de estas clases (Universidad y Departamento) muestran varias operaciones para manipular sus partes. Estas operaciones se incluyen porque son importantes para mantener la integridad de los datos (añadir o eliminar un Departamento).

2. Diagramas de componentes. Un diagrama de componentes muestra las partes internas, los conectores y los puertos que implementan un componente.

Cuando se instancia el componente, también se instancian las copias de sus partes internas.

El siguiente es un ejemplo de un modelo de componente de recopilación de datos. La Figura muestra un modelo de un componente que ha sido diseñado, para recopilar y comparar informaci6n de un vector de sensores. Se ejecuta de forma autónoma para recoger datos durante un periodo de tiempo, y proporciona bajo demanda la comparación de los datos a otro componente. La interfaz proporciona e incluye métodos para añadir, eliminar, iniciar, parar y probar los sensores. También incluye métodos para informes (report y IistAll) que informan sobre los datos recopilados y la conftguraci6n de los sensores. Aunque no se ha mostrado esto aquí, estos métodos naturalmente tienen asociados parámetros que especifican la localizaci6n de los sensores y otra información.

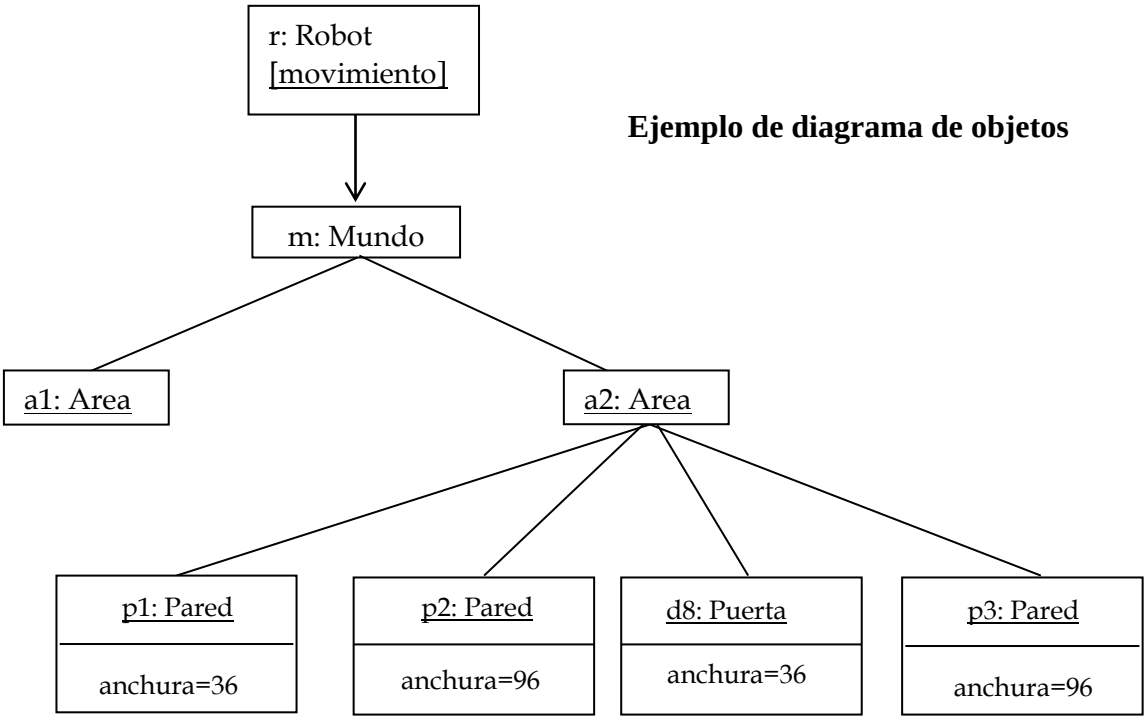


El componente de recopilación de datos requiere que los sensores proporcionen una interfaz de gestión y una interfaz de datos. Estos tienen parámetros que especifican el funcionamiento y los datos que hay que recopilar. Se ha diseñado de forma deliberada la interfaz requerida para que no incluya operaciones específicas tales como Test. La interfaz permite al componente de recopilación de datos utilizarse con sensores con diferentes interfaces. Un componente adaptador se utiliza como una interfaz entre el recopilador de datos y la interfaz hardware

específico del sensor.

3. **Diagrama de estructura compuesta.** Un diagrama de estructura compuesta muestra la estructura interna de una clase o una colaboración. La diferencia entre componentes y estructura compuesta es mínima.

4. **Diagrama de objetos.** Un diagrama de objetos representa un conjunto de objetos y sus relaciones. Se utilizan para describir estructuras de datos, instantáneas estáticas de las instancias de los elementos existentes en los diagramas de clases. Los diagramas de objetos abarcan la vista de diseño estática o la vista de procesos estática de un sistema al igual que los diagramas de clases, pero desde la perspectiva de casos reales o prototípicos.



Ejemplo de diagrama de objetos

La figura anterior representa un conjunto de objetos extraídos de la implementación de un robot autónomo. Esta figura se centra en algunos de los objetos implicados en el mecanismo utilizado por el robot para calcular un modelo del mundo en el que se mueve.

Hay muchos más objetos implicados en un sistema en ejecución, pero este diagrama se centra sólo en aquellas abstracciones implicadas directamente en la creación de esta vista del mundo. Como se indica en la figura, un objeto representa al propio robot (*r*, una instancia de *Robot*), y r se encuentra actualmente en el estado *movimiento*. Este objeto tiene un enlace con m, una instancia de *Mundo*, que representa una abstracción del modelo del mundo del robot.

En ese instante, *m* esta enlazado a dos instancias de *Área*. Una de ellas (*a2*) se muestra con sus propios enlaces a tres objetos *Pared* y un objeto *Puerta*. Cada una de estas paredes esta etiquetada con su anchura actual, y cada una se muestra enlazada a sus paredes vecinas. Como sugiere este diagrama de objetos, el robot ha reconocido el área que lo contiene, que tiene paredes en tres lados y una puerta en el cuarto.

5. **Diagrama de artefactos.** Un diagrama de artefactos muestra un conjunto de artefactos y sus relaciones con otros artefactos y con las clases a las que implementan.
6. **Diagramas de despliegue.** Un diagrama de despliegue muestra un conjunto de nodos y sus relaciones. Los diagramas de despliegue se utilizan para describir la vista de despliegue estática de una arquitectura**.**

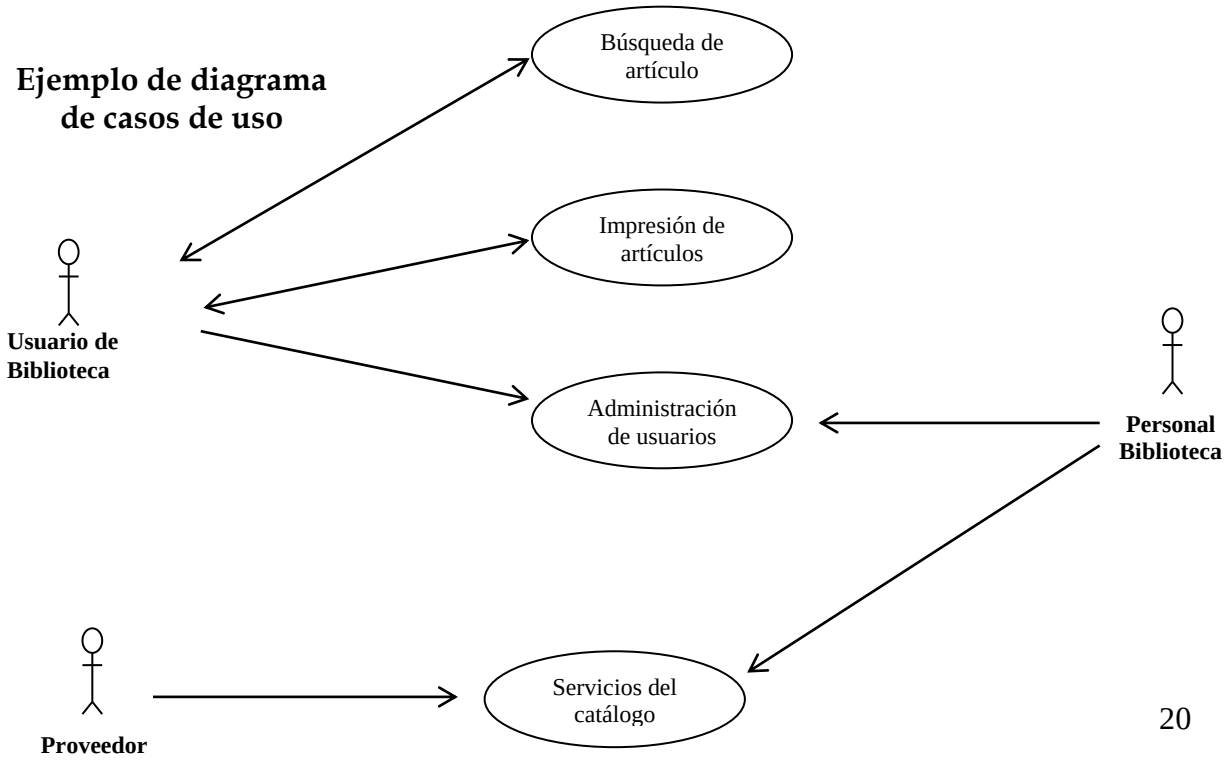
3.1.2.- Diagramas de Comportamiento

Los diagramas de comportamiento de UML se emplean para visualizar, especificar, construir y documentas los aspectos dinámicos de un sistema. Los aspectos dinámicos de un sistema software involucran cosas tales como el flujo de mensajes a lo largo del tiempo y el movimiento físico de componentes en una red. Los diagramas de comportamiento de UML se organizan en líneas generales alrededor de las formas principales en que se pueden modelar la dinámica de un sistema:

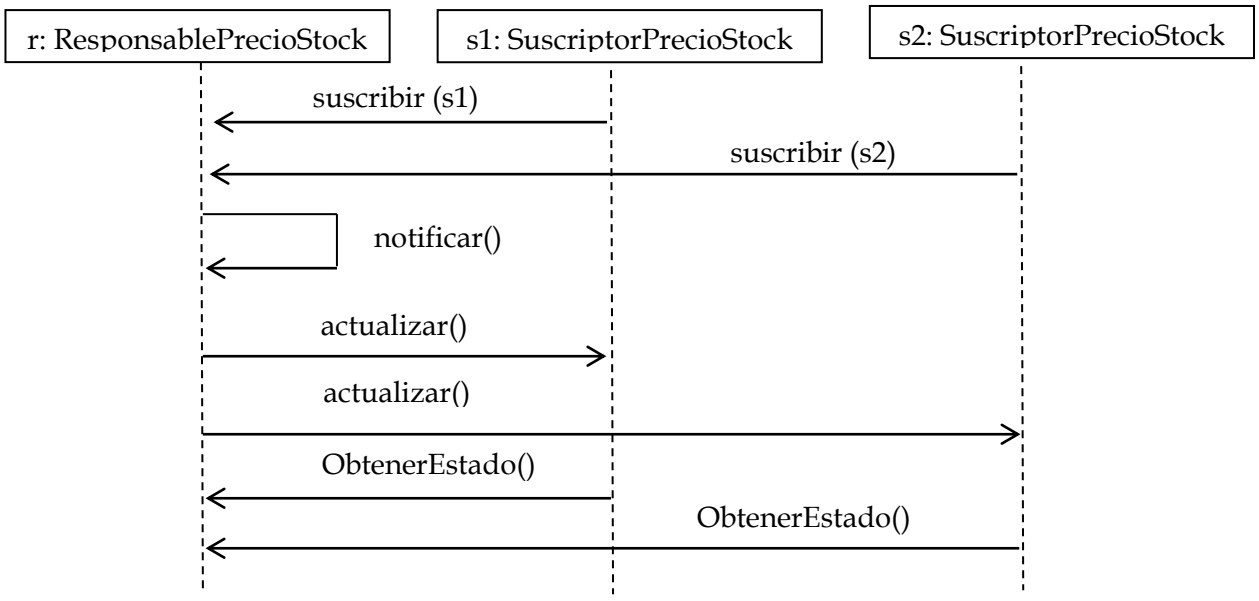
1. Diagramas de casos de uso. Organiza los comportamientos del sistema

2. Diagramas de secuencia. Centrados en la ordenación temporal de los mensajes.
3. Diagramas de comunicación. Centrados en la organización estructural de los objetos que envían y reciben mensajes
4. Diagramas de estados. Centrados en el estado cambiante de un sistema dirigido por eventos.
5. Diagramas de actividades. Centrados en el flujo de control de actividades.

1. Diagramas de casos de uso. Un diagrama de casos de uso representa un conjunto de casos de uso y actores (un tipo especial de clases) y sus relaciones. Los diagramas de casos de uso son especialmente importantes para organizar y modelar el comportamiento de un sistema. La siguiente figura ilustra la esencia de la notación para los casos de uso. Los actores en el proceso se representan como figuras delineadas, y cada clase de interacción se representa con una elipse con su nombre. El conjunto de casos de uso representa todas las posibles interacciones a representar en los requerimientos del sistema. Esta figura muestra un ejemplo de un de sistema biblioteca.



2. Diagramas de secuencia. Un diagrama de secuencia es un diagrama de interacción que resalta la ordenación temporal de los mensajes. Un diagrama de secuencia presenta un conjunto de roles y los mensajes enviados y recibidos por las instancias que interpretan los roles. Los diagramas de secuencia se utilizan para describir la vista dinámica de un sistema.



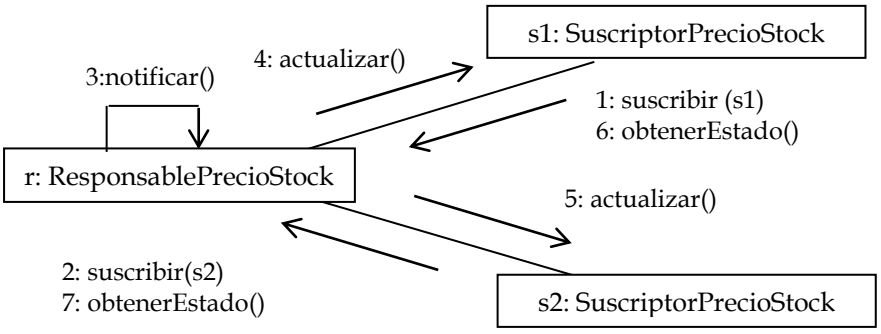
Ejemplo de Diagrama de secuencia

Por ejemplo, la figura anterior representa un conjunto de objetos que interactúan en el contexto de un mecanismo de publicación y suscripción. Esta figura incluye tres roles: r (un *ResponsablePrecioStock*), s1 y s2 (ambos instancias de *SuscriptorPrecioStock*). Esta figura es un ejemplo de un diagrama de secuencia, que resalta la ordenación temporal de los mensajes.

3. Diagramas de comunicación. Un diagrama de comunicación es un diagrama de interacción que resalta la organización estructural de los objetos que envían y reciben mensajes.

El siguiente ejemplo es semánticamente equivalente a la figura anterior (Ejemplo de diagrama de secuencia), pero ha sido dibujada como un diagrama de

comunicación, el cual destaca la organización estructural de los objetos. Esta figura representa el mismo flujo de control, pero también muestra los enlaces entre esos objetos.



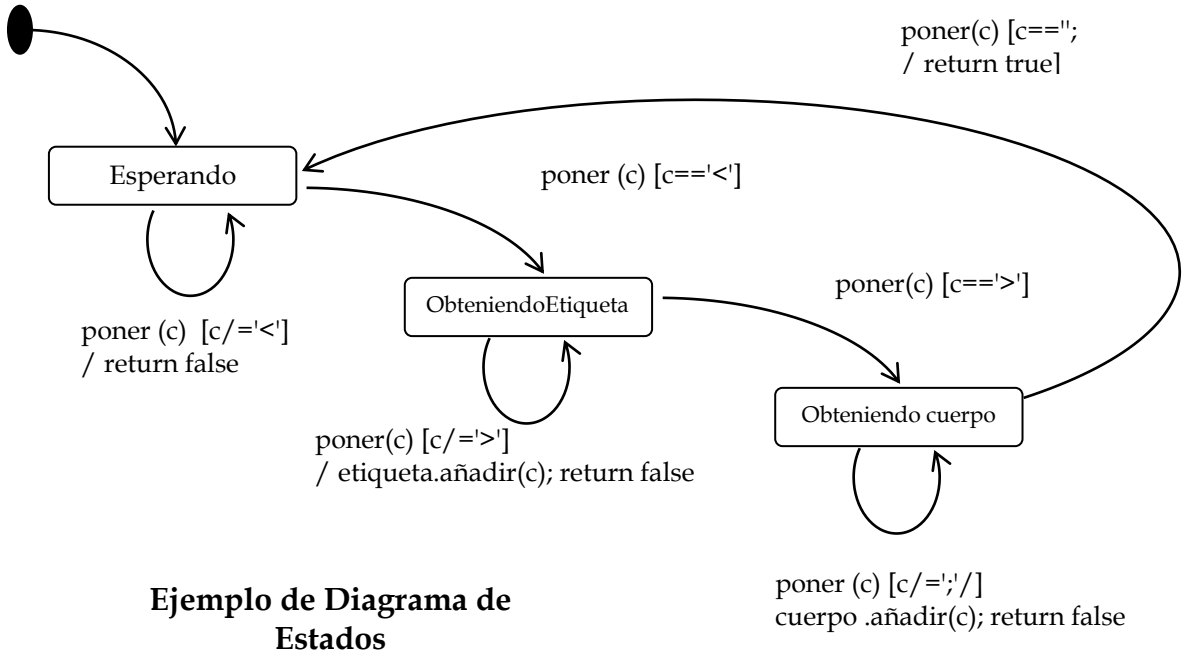
Ejemplo de diagrama de comunicación

4. Diagramas de estados. Un diagrama de estados representa una maquina de estados, constituida por estados, transiciones, eventos y actividades. Los diagramas de estado se utilizan para describir la vista dinámica de un sistema. Son especialmente importantes para modelar el comportamiento de una interfaz, una clase o una colaboración. Los diagramas de estados resaltan el comportamiento dirigido por eventos de un objeto, lo que es especialmente útil al modelar sistemas reactivos.

Por ejemplo, la siguiente figura muestra el diagrama de estados para analizar un lenguaje libre de contexto muy sencillo, como el que puede haber en un sistema que genera o interpreta mensajes en XML. En este caso, la maquina se ha diseñado para analizar una secuencia de caracteres que coincida con la siguiente sintaxis:

mensaje : `<´ cadena `>´ cadena `;´

La primera cadena de caracteres representa una etiqueta; la segunda representa el cuerpo del mensaje. Dada una secuencia de caracteres, sólo se pueden aceptar los mensajes bien formados que sigan esta sintaxis.

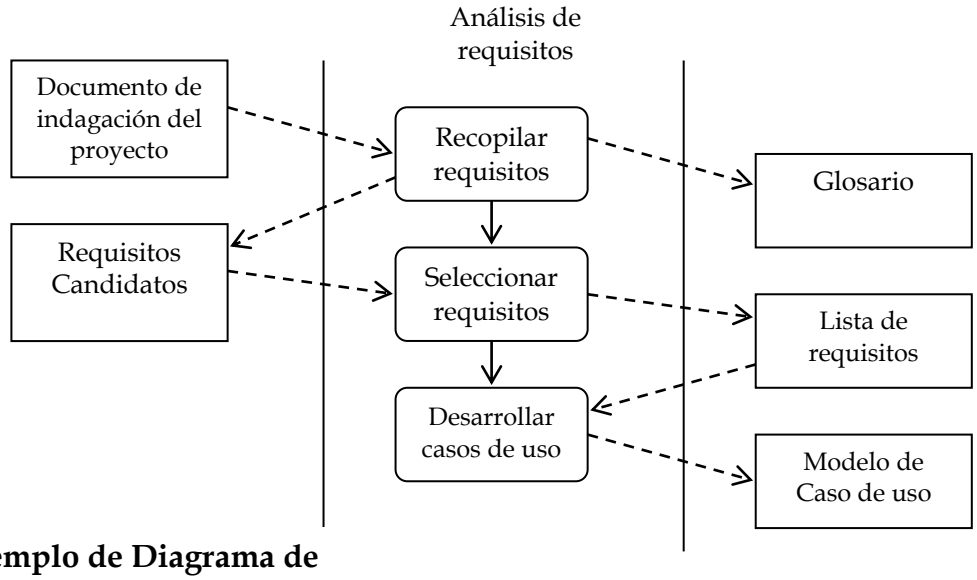


Ejemplo de Diagrama de Estados

Como se muestra en la figura, sólo hay tres estados estables en esta máquina de estados: *Esperando*, *ObteniendoEtiqueta* y *ObteniendoCuerpo.* Esta máquina de estados se ha diseñado con las acciones asociadas a las transiciones, de hecho sólo hay un evento de interés, la invocación de *poner* con el parámetro actual c (un carácter). Mientras está *Esperando,* esta máquina rechaza cualquier carácter que no suponga el comienzo de una etiqueta (como se especifica en la condición de guarda). Cuando se recibe el carácter de inicio de una palabra, el estado del objeto cambia a *ObteniendoEtiqueta*. Mientras está en este estado, la máquina guarda cualquier carácter que no suponga el final de la etiqueta (como se especifica en la condición guarda). Cuando se recibe el final de la etiqueta, el estado del objeto cambia a *ObteniendoCuerpo.* Mientras en este estado, la máquina guarda cualquier carácter que no suponga el final del cuerpo de un mensaje (como se especifica en la condición guarda). Cuando se recibe el final del mensaje, el estado del objeto

cambia a *Esperando,* y se devuelve un valor indicando que el mensaje ha sido analizado (y la máquina está preparada para recibir otro mensaje).Nótese que este diagrama especifica una máquina que funciona continuamente, no hay estado final.

5. Diagramas de actividades. Un diagrama de actividades muestra el flujo paso a paso en una computación. Una actividad muestra un conjunto de acciones, el flujo secuencial o ramificado de acción en acción, y los valores que son producidos o consumidos por las acciones. Los diagramas de actividades se utilizan para ilustra la vista dinámica de un sistema. Además, estos diagramas son especialmente importantes para modelar la función de un sistema, así como para resaltar el flujo de control en la ejecución de un comportamiento. El siguiente ejemplo muestra el diagrama de actividad para captura y modelado de requisitos:

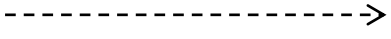


Ejemplo de Diagrama de Actividades

3.1.3.- Relaciones en UML: Hay cuatro tipos de relaciones en UML:

1. Dependencia. Es una relación semántica entre dos elementos, en la cual un cambio a un elemento (el elemento independiente) puede afectar a la

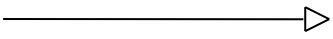
semántica del otro elemento (el elemento dependiente). Gráficamente una dependencia se representa como una línea discontinua.



Dependencias

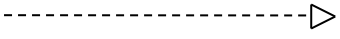
2. Asociación. Es una relación estructural entre clases que describe un conjunto de enlaces, los cuales son conexiones entre objetos que son instancias de clases. La agregación es un tipo especial de asociación, que representa una relación estructural entre un todo y sus partes. Gráficamente una asociación se representa como una línea continua.

3. Generalización. Es una relación de especialización/generalización en la cual el elemento especializado (el hijo) se basa en la especificación del elemento generalizado (el padre). El hijo comparte la estructura y el comportamiento del padre. Gráficamente una relación de realización se representa como una línea continua con una punta de flecha vacía apuntando al padre.



Generalizaciones

4. Realización. Es una relación semántica entre clasificadores, en donde un clasificador especifica un contrato que otro clasificador garantiza que cumplirá. Se pueden encontrar relaciones de realización en dos sitios: entre interfaces y las clases y componentes que las realizan, y entre los casos de uso y las colaboraciones que los realizan. Gráficamente, una relación de realización se representa como una mezcla entre una generalización y una relación de dependencia.



Realizaciones

3.2.- Modelo de proceso unificado de desarrollo de software.

3.2.1.- Definición de Proceso Unificado: es un proceso de software genérico que puede ser utilizado para una gran cantidad de tipos de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de competencia y diferentes tamaños de proyectos. Provee un enfoque disciplinado en la asignación de tareas y responsabilidades dentro de una organización de desarrollo. Su meta es asegurar la producción de software de muy alta calidad que satisfaga las necesidades de los usuarios finales, dentro de un calendario y presupuesto predecible.

El Proceso Unificado usa el Lenguaje de Modelado Unificado (UML) en la preparación de todos los planos del sistema. De hecho, UML es una parte integral del Proceso Unificado (fueron desarrollados a la par).

Los aspectos distintivos del Proceso Unificado están capturados en tres conceptos clave: iterativo e incremental, dirigido por casos de uso, centrado en la arquitectura (esto es lo que hace único al Proceso Unificado).

3.2.2.- Conceptos clave del proceso unificado

- **Iterativo e incremental**

El Proceso Unificado es un marco de desarrollo compuesto de cuatro fases: inicio, elaboración, construcción, transición. Cada una de ellas es, a su vez, dividida en una serie de iteraciones que ofrecen como resultado un incremento del producto desarrollado, que añade o mejora las funcionalidades del sistema en desarrollo. Es decir, un "incremento" no implica necesariamente una ampliación de dicho sistema.

Durante cada una de estas iteraciones se realizarán a su vez las actividades definidas en el ciclo de vida clásico: requisitos, análisis, diseño, implementación, prueba e implantación. Aunque todas las iteraciones suelen incluir trabajo en casi todas estas actividades, el grado de esfuerzo dentro de cada una de ellas varía a lo

largo del proyecto. Por ejemplo, en la fase de inicio se centrarán más en la definición de requisitos y en el análisis, y durante la de construcción quedarán relegadas en favor de la implementación y las pruebas.

Si una iteración cumple sus metas, publicando una nueva versión del producto que implemente ciertos casos de uso, el desarrollo continúa con la siguiente. Cuando no las cumple, los desarrolladores deben revisar sus decisiones previas y probar un nuevo enfoque.

- **Dirigido por los casos de uso**

Un sistema software se crea para servir a sus usuarios por lo que, para construir un sistema exitoso, se debe conocer qué es lo que quieren y necesitan. El término "usuario" no se refiere solamente a los usuarios humanos sino también a otros sistemas, es decir, representa a algo o alguien que interactúa con el sistema a desarrollar.

En el Proceso Unificado, los casos de uso se utilizan para capturar los requisitos funcionales y para definir los objetivos de las iteraciones. En cada una, los desarrolladores identifican y especifican los casos de uso relevantes, crean el diseño usando la arquitectura como guía, implementan el diseño en componentes y verifican que los componentes satisfacen los casos de uso.

- **Centrado en la arquitectura**

El concepto de arquitectura del software involucra los aspectos estáticos y dinámicos más significativos del sistema, y actúa como vista del diseño, dando una perspectiva completa y describiendo los elementos más importantes. La arquitectura surge de los propios casos de uso, sin embargo, también está influenciada por muchos otros factores, como la plataforma en la que se ejecutará, el uso de estándares, la existencia de sistemas heredados (aunque éste no sea el caso que nos ocupa) o los requisitos no funcionales.

Puesto que la arquitectura y los casos de uso están relacionados, por una parte, los casos de uso deben, cuando son realizados, acomodarse en la arquitectura, y ésta debe ser lo bastante flexible para realizar todos los casos de uso, hoy y en el futuro. De palabras de los propios creados del Proceso Unificado, es un problema semejante al del "huevo y la gallina". En la realidad, arquitectura y casos de uso deben evolucionar en paralelo.

3.3.- Herramientas CASE.

3.3.1.- Rational Rose: Es la herramienta CASE que comercializan los desarrolladores de UML y que soporta de forma completa la especificación del UML 1.1.Esta herramienta propone la utilización de cuatro tipos de modelo para realizar un diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software.

a. Desarrollo Iterativo.

Rational Rose utiliza un proceso de desarrollo iterativo controlado (controlled iterative process development), donde el desarrollo se lleva a cabo en una secuencia de iteraciones. Cada iteración comienza con una primera aproximación del análisis, diseño e implementación para identificar los riesgos del diseño, los cuales se utilizan para conducir la iteración, primero se identifican los riesgos y después se prueba la aplicación para que éstos se hagan mínimos. Cuando la implementación pasa todas las pruebas que se determinan en el proceso, ésta se revisa y se añaden los elementos modificados al modelo de análisis y diseño. Una vez que la actualización del modelo se ha modificado, se realiza la siguiente iteración.

b. Trabajo en Grupo.

Rose permite que haya varias personas trabajando a la vez en el proceso iterativo controlado, para ello posibilita que cada desarrollador opere en un espacio de trabajo privado que contiene el modelo completo y tenga un control exclusivo sobre la propagación de los cambios en ese espacio de trabajo.

También es posible descomponer el modelo en unidades controladas e integrarlas con un sistema para realizar el control de proyectos que permite mantener la integridad de dichas unidades.

c. Generador de Código

Se puede generar código en distintos lenguajes de programación a partir de un diseño en UML.

d. Ingeniería Inversa

Rational Rose proporciona mecanismos para realizar la denominada Ingeniería Inversa, es decir, a partir del código de un programa, se puede obtener información sobre su diseño.

3.3.2.- Herramienta CASE para el diseño de base de datos:

- **ERwin:** Es una herramienta de diseño de base de datos. Brinda productividad en diseño, generación, y mantenimiento de aplicaciones. Desde un modelo lógico de los requerimientos de información, hasta el modelo físico perfeccionado para las características específicas de la base de datos diseñada, ERwin permite visualizar la estructura, los elementos importantes, y optimizar el diseño de la base de datos.

ERwin hace fácil el diseño de una base de datos. Los diseñadores de bases de datos sólo apuntan y pulsan un botón para crear un gráfico del modelo Entidad-Relación

de todos sus requerimientos de datos y capturar las reglas de negocio en un modelo lógico, mostrando todas las entidades, atributos, relaciones, y llaves importantes.

Más que una herramienta de dibujo, ERwin automatiza el proceso de diseño de una manera inteligente. Por ejemplo, ERwin habilita la creación de un diccionario de atributos reusables, asegurando la consistencia de nombres y definiciones para su base de datos.

Se mantienen las vistas de la base de datos como componentes integrados al modelo, permitiendo que los cambios en las tablas sean reflejados automáticamente en las vistas definidas. La migración automática garantiza la integridad referencial de la base de datos.

ERwin establece una conexión entre una base de datos diseñada y una base de datos, permitiendo transferencia entre ambas y la aplicación de ingeniería reversa. Usando esta conexión, Edwin genera automáticamente tablas, vistas, índices, reglas de integridad referencial (llaves primarias, llaves foráneas), valores por defecto y restricciones de campos y dominios.

3.4.- Normalización

La normalización es la transformación de vistas de usuarios complejas y almacenes de datos a un conjunto de estructuras de datos establecidos menor, más simples y más fáciles de mantener.

3.4.1.- Pasos para la normalización

- Una relación esta en **Primera forma normal (1FN)** si todos los campos en cada registro contienen un solo valor tomado de los dominios respectivos; otra definición seria la descomposición de todos los grupos de datos en registro bidimensionales (relación uno a uno); incluye la eliminación de todos los grupos repetidos y la identificación de clave primaria.

- **Segunda forma normal (2FN)** se alcanza cuando un registro esta en 1FN y cada campo depende totalmente de la llave de registro; para obtener la 2FN cada campo del registro que no dependa de la llave primaria debe sustituirse y utilizarse para forma una relación aparte, o también eliminando las relaciones en las que los datos no dependan completamente de la llave primaria, las dependencias parciales son eliminadas y puestas en otra relación.

- **La tercera forma normal (3FN)** se alcanza cuando una relación esta en (2FN) y ningún atributo no clave en la relación es funcionalmente dependiente de un atributo no clave. Esta forma se logra cuando se eliminan todas las relaciones que contengan dependencias transitivas. La dependencia transitiva es aquella en la cual los atributos que no son llaves, son dependientes de otros atributos que tampoco son llaves.

En la mayoría de los casos la normalización se completa cuando todas las relaciones derivada están en la " 3FN " , por lo tanto la 1FN, 2FN, 3FN son consideradas las más comunes dentro del proceso de normalización además existe otras dos tipos más de normalización la cuarta, y la quinta forma normal.

3.5.- El Modelo Entidad-Relación

Con este modelo se definen las entidades, las cuales son identificables y de importancia para los usuarios. Todas las entidades de cierto tipo forman una clase de entidad. Una entidad particular se denomina una ocurrencia. Las entidades tienen atributos que describen sus características y uno o más atributos identifican una entidad.

Las relaciones son asociaciones entre entidades. El modelo E-R define de manera física las relaciones; cada relación tiene un nombre; y hay clases de relaciones al

igual que ocurrencias de relaciones. El grado de relación es la cantidad de entidades que participan en ella. La mayoría de las relaciones son binarias. Los tres tipos de relaciones binarias son 1:1, 1:N, y N:M.

En los diagramas entidad-relación, las entidades se muestran en rectángulos y las relaciones en diamantes. La cardinalidad máxima de la relación aparece dentro del diamante. La cardinalidad mínima se indica a través de una línea transversal o un ovalo. Las relaciones que conectan las ocurrencias de entidades de la misma clase son recursivas. En un diagrama E-R, los atributos pueden mostrarse en elipses o en una tabla separada.

Una entidad débil es aquella cuya existencia depende de otra. Las entidades débiles se muestran en rectángulos con esquinas redondeadas y las relaciones de las que dependen se indican por medio de un diamante con esquinas redondeadas.

3.6.- Métodos de prueba del software

La prueba del software es un elemento crítico para la garantía de la calidad y representa la revisión final de las especificaciones del diseño y de la clasificación.

3.6.1.-Estrategias de prueba

1. **Prueba de unidad**

La prueba de unidad centra el proceso de verificación en la menor unidad del diseño del software: el componente software o módulo. Durante la prueba de unidad, la comprobación selectiva de los caminos de ejecución es una tarea esencial. Se deben diseñar casos de prueba para detectar errores debidos a cálculos incorrectos, comparaciones incorrectas o flujos de control inapropiados. Las pruebas del camino básico y de bucles son técnicas muy efectivas para descubrir una gran cantidad de errores en los caminos.

2. **Prueba de integración**

La prueba de integración es una técnica sistemática para construir la estructura del programa mientras que, al mismo tiempo, se llevan a cabo pruebas para detectar errores asociados con la interacción. El objetivo es tomar los módulos probados mediante la prueba de unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño.

3. **Prueba de validación**

Tras la culminación de la prueba de integración, el software está completamente ensamblado como un paquete, se han encontrado y corregido los errores de interfaz y puede comenzar una serie final de pruebas del software: La prueba de validación.

- **Criterios de la prueba de validación**

a. **Revisión de la configuración:** Asegura que todas las configuraciones del software se han desarrollado apropiadamente, se han catalogado y están suficientemente detalladas para soportar las fases de mantenimiento.

b. **Pruebas alpha y beta**

La prueba alpha se lleva a cabo en el lugar de desarrollo por un cliente, con el desarrollador como observador del usuario registrando todos los errores (entorno controlado).

La prueba beta se lleva a cabo por los usuarios finales en los lugares de trabajo y el desarrollador normalmente se encuentra presente (entorno no controlado), el usuario registra todos los errores y los informa al desarrollador.

4. Prueba de sistema

Está constituida por una serie de pruebas cuyo propósito primordial es ejercitar profundamente el sistema basado en computadora, la prueba de sistema se clasifica en:

- **Prueba de recuperación:** es una prueba que consiste en forzar el fallo del software de muchas formas y verifica que la recuperación se lleva a cabo apropiadamente. Si la recuperación es automática (llevada a cabo por el propio sistema) hay que evaluar la corrección de la inicialización, de los mecanismos de recuperación del estado del sistema, de la recuperación de datos y del proceso de re arranque. Si la recuperación requiere la intervención humana, hay que evaluar los tiempos medios de reparación (TMR) para determinar si están dentro de unos límites aceptables.

- **Prueba de seguridad:** este tipo de prueba verifica que los mecanismos de protección incorporados en el sistema lo protegerán de accesos no autorizados.

- **Pruebas de resistencia:** están diseñadas para enfrentar a los programas con situaciones anormales, haciendo que el sistema demande recursos en cantidad, frecuencia y volúmenes fuera de lo común con el fin de determinar la potencia a la que pueda funcionar el sistema.

- **Prueba de rendimiento:** Esta diseñada para probar el rendimiento del software en tiempo de ejecución dentro del contexto de un sistema integrado.

3.7.- Teoría sobre factibilidad.

3.7.1.- Definición de factibilidad: Factibilidad se refiere a la disponibilidad de los recursos necesarios para llevar a cabo los objetivos o metas señalados. Los recursos son tratados en relación con tres áreas de factibilidad.

3.7.2.- Áreas de factibilidad

1. **Factibilidad técnica:** El analista debe determinar si existe tecnología disponible para implementar el sistema. El proyecto debe considerar si los recursos técnicos actuales son suficientes o deben complementarse es decir determinar si la organización tiene la capacidad de satisfacer las necesidades para el desarrollo del sistema.

2. **Factibilidad económica:** Los recursos básicos a considerar son: el tiempo del analista de sistema, el costo de estudio de sistemas, costo del tiempo de los empleados para el estudio, costo estimado de hardware y costo del paquete de software/desarrollo del software.

3. **Factibilidad operacional:** Durante ésta etapa se identifican todas aquellas actividades que son necesarias para lograr el objetivo, se evalúa y determina todo lo necesario para llevarlo a cabo. La factibilidad operacional depende de los recursos humanos disponibles para el proyecto.

3.8.- Base de datos

3.8.1.- Definición de base de datos: Una base de datos es una colección de datos estructurados según un modelo que refleje las relaciones y restricciones existentes en el mundo real. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de éstas, y su definición y descripción han de ser únicas estando almacenadas junto a los mismos. Los tratamientos que sufran estos datos tendrán que conservar la integridad y seguridad de éstos.

Los componentes de una aplicación de una base de datos son la base de datos, los sistemas de administración de la base de datos (DBMS) y los programas de aplicación. Algunas veces los programas de aplicación están separados por completo del DBMS; en otras ocasiones, algunas partes sustanciales de la aplicación son proporcionadas mediante características y funciones del DBMS.

Las características y funciones del DBMS se agrupan en tres subsistemas. El subsistema de herramientas de diseño define la base de datos y la estructura de las aplicaciones o componentes de la aplicación. Las funciones del subsistema de tiempo de ejecución son materializar las formas, reportes y consultas, leyendo o escribiendo los datos de la base de datos. El motor DBMS es el intermediario entre los otros dos subsistemas y el sistema operativo. Recibe peticiones establecidas en términos de tablas, filas y columnas, y las traduce a solicitudes de lectura y escritura.

Un esquema es una descripción de la estructura de una base de datos e incluye descripciones de las tablas, las relaciones, los dominios y las reglas de negocios. Las filas de una tabla pueden hacer referencia a las filas de otra tabla. Las funciones del DBMS se usan para crear estructuras de tablas, así como para relaciones y generar formas, reportes, consultas y menús.

En la actualidad hay muchos sistemas gestores de bases de datos para almacenar grandes bases de datos. Los más comunes son: Oracle, SQL–Server, Informix, MySQL, postgree SQL, Access y aún se encuentra en sistemas viejos Visual Fox Pro. Cada uno posee sus propias características.

3.9.- Lenguaje de programación Visual Basic Profesional 6.0

3.9.1.- ¿Qué es Visual Basic?

La palabra "Visual" hace referencia al método que se utiliza para crear la interfaz gráfica de usuario (GUI), la palabra "Basic" hace referencia al lenguaje BASIC (Beginners All-Purpose Symbolic Instruction Code). Visual Basic ha evolucionado a partir del lenguaje BASIC original y ahora en lugar de escribir numerosas líneas de código para describir la apariencia y la ubicación de los elementos de la interfaz, simplemente podemos agregar objetos prefabricados en su lugar dentro de la pantalla, que ahorran mucho tiempo de programación y que sobrepasan el concepto de la programación convencional. Es un lenguaje guiado por eventos, y centrado en un motor de formularios poderoso que facilita el rápido desarrollo de aplicaciones gráficas.

3.9.2.- Ediciones de Visual Basic

Visual Basic se encuentra disponible en diferentes versiones, que se describen a continuación:

- **Visual Basic Edición Empresarial:** Creado para entornos de programación en equipo y computación cliente/servidor donde las aplicaciones distribuyen el procesamiento y la información entre varias computadoras.
- **Visual Basic Edición Profesional:** Dirigido a programadores profesionales que desean sacar el máximo provecho al entorno de programación de Visual Basic. Esta edición incluye un conjunto completo de herramientas y asistentes que le ayudan a empaquetar y distribuir las aplicaciones.

- **Visual Basic Edición de Aprendizaje:** Lo fundamental con el complemento de las herramientas de programación estándar y todo lo necesario para comenzar a programar.
- **Visual Basic Working Model:** Es el entorno de programación básico. No permite la generación de ejecutables, pero es un buen medio para darse una idea del entorno, sus capacidades y prestaciones, u decidir si se adquiere o no alguna de las otras versiones.

3.10.- Sistema de encriptación: "Criptosistema Hill "

El tipo de sistema de encriptación utilizado para el módulo de seguridad será "Criptosistema Hill", a continuación daremos a conocer con mayor exactitud este tipo de sistema de encriptación.

Este sistema está basado en el álgebra lineal y ha sido importante en la historia de la criptografía. Fue Inventado por Lester S. Hill en 1929, y fue el primer sistema criptográfico poli alfabético que era práctico para trabajar con más de tres símbolos simultáneamente.

Este sistema es poli alfabético pues puede darse que un mismo carácter en un mensaje a enviar se encripte en dos caracteres distintos en el mensaje encriptado.

Suponiendo que trabajamos con un alfabeto de 26 caracteres, las letras se numeran en orden alfabético de forma tal que A=0, B=1, ..., Z=25

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Se elije un entero "d" que determina bloques de "d" elementos que son tratados como un vector de "d" dimensiones, se elije de forma aleatoria una matriz de (d × d) elementos los cuales serán la clave a utilizar.

Los elementos de la matriz de (d × d) serán enteros entre 0 y 25, además la matriz M debe ser invertible en $\mathbb{Z}_{26}^{n}$.

Para la encriptación, el texto es dividido en bloques de "d" elementos los cuales se multiplican por la matriz (d × d).

Todas las operaciones aritméticas se realizan en la forma módulo 26, es decir que 26=0, 27=1, 28=2 etc.

Dado un mensaje a encriptar debemos tomar bloques del mensaje de "d" caracteres y aplicar:

$M \times P_i = C$, donde C es el código cifrado para el mensaje P_i

Ejemplo:

Si tomamos la matriz $A = \begin{pmatrix} 5 & 17 & 20 \\ 9 & 23 & 3 \\ 2 & 11 & 13 \end{pmatrix}$ como matriz de claves.

Para encriptar el mensaje "CODIGO" debemos encriptar los seis caracteres de "CODIGO" en bloques de 3 caracteres cada uno, el primer bloque

$$P_1 = \text{"COD"} = \begin{pmatrix} 2 \\ 14 \\ 3 \end{pmatrix} \qquad P_2 = \text{"IGO"} = \begin{pmatrix} 6 \\ 8 \\ 14 \end{pmatrix}$$

$$A \cdot P_1 = \begin{pmatrix} 5 & 17 & 20 \\ 9 & 23 & 3 \\ 2 & 11 & 13 \end{pmatrix} \begin{pmatrix} 2 \\ 14 \\ 3 \end{pmatrix} = \begin{pmatrix} 308 \\ 349 \\ 197 \end{pmatrix} = \begin{pmatrix} 22 \\ 11 \\ 15 \end{pmatrix} (\mathrm{mod}\, 26)$$

El primer bloque "COD" se codificara como "WLP"

$$A \cdot P_2 = \begin{pmatrix} 5 & 17 & 20 \\ 9 & 23 & 3 \\ 2 & 11 & 13 \end{pmatrix} \begin{pmatrix} 8 \\ 6 \\ 14 \end{pmatrix} = \begin{pmatrix} 422 \\ 252 \\ 264 \end{pmatrix} = \begin{pmatrix} 6 \\ 18 \\ 4 \end{pmatrix} (\mathrm{mod}\, 26)$$

El segundo bloque "IGO" se codificara como "GSE", luego 'CODIGO' encriptado equivale a 'WLPGSE'.

NOTA: Observar que las dos "O" se codificaran de forma diferente.

Para desencriptar el método es idéntico al anterior pero usando la matriz inversa de la usada para encriptar.

Cálculo de la matriz inversa

Antes que nada debemos verificar que la matriz elegida sea invertible en modulo 26. Hay una forma relativamente sencilla de averiguar esto a través del cálculo del determinante. Si el determinante de la matriz es 0 o tiene factores comunes con el módulo (en el caso de 26 los factores son 2 y 13), entonces la matriz no puede utilizarse. Al ser 2 uno de los factores de 26 muchas matrices no podrán utilizarse (no servirán todas en las que su determinante sea 0, un múltiplo de 2 o un múltiplo de 13)

Para ver si es invertible calculo el determinante de A

$$\begin{vmatrix} 5 & 17 & 20 \\ 9 & 23 & 3 \\ 2 & 11 & 13 \end{vmatrix}$$

5 (23 ·13 – 3 ·11) – 17 (9 ·13 – 3 ·2) + 20 (9 ·11 – 23 ·2) =

1215 – 1734 + 1060 = 503

503 = 9 mod 26

La matriz A es invertible en modulo 26 ya que 26 y 9 son coprimos.

Para hallar la inversa de la matriz modulo 26, utilizamos la fórmula $A^{-1}=C^{T}\cdot\left(\det(A)\right)^{-1}$

Donde CT es la matriz de cofactores de A transpuesta

Hay que tener en cuenta que $\left(\det(A)\right)^{-1}$ debe realizarse en modulo 26 por lo tanto para el ejemplo la inversa de 9 (mod 26) es 3 (mod 26) ya que:

9 (mod 26) ·3 (mod 26) = 27 mod 26 = 1 (mod 26)

Por lo tanto 3 es la inversa multiplicativa de 9 en modulo 26.

Para calcular C hay que calcular los cofactores de A

$$C_{11}=+\begin{vmatrix}23 & 3\\ 11 & 13\end{vmatrix} \qquad C_{12}=-\begin{vmatrix}9 & 3\\ 2 & 13\end{vmatrix} \qquad C_{13}=+\begin{vmatrix}9 & 23\\ 2 & 11\end{vmatrix}$$

$$C_{21}=-\begin{vmatrix}17 & 20\\ 11 & 13\end{vmatrix} \qquad C_{22}=+\begin{vmatrix}5 & 20\\ 2 & 13\end{vmatrix} \qquad C_{23}=-\begin{vmatrix}5 & 17\\ 2 & 11\end{vmatrix}$$

$$C_{31}=+\begin{vmatrix}17 & 20\\ 23 & 3\end{vmatrix} \qquad C_{23}=-\begin{vmatrix}5 & 20\\ 9 & 3\end{vmatrix} \qquad C_{33}=+\begin{vmatrix}5 & 17\\ 9 & 23\end{vmatrix}$$

$$C=\begin{pmatrix}266 & -111 & 53\\ -1 & 25 & -21\\ -409 & 165 & -38\end{pmatrix} \qquad C^{T}=\begin{pmatrix}266 & -1 & -409\\ -111 & 25 & 165\\ 53 & -21 & -38\end{pmatrix}$$

Ahora aplicamos la formula de la inversa

$$A^{-1} = C^{T} \cdot \left(\det(A)\right)^{-1} = \begin{pmatrix} 266 & -1 & -409 \\ -111 & 25 & 165 \\ 53 & -21 & -38 \end{pmatrix} \cdot 3$$

$$A^{-1} = \begin{pmatrix} 798 & -3 & -1227 \\ -333 & 75 & 495 \\ 159 & -63 & -114 \end{pmatrix} \qquad A^{-1} = \begin{pmatrix} 18 & 23 & 21 \\ 5 & 23 & 1 \\ 3 & 15 & 16 \end{pmatrix} (\mathrm{mod}\, 26)$$

Esta última es la matriz que utilizamos para desencriptar.

4.- Extracción y análisis de datos

4.1.- Introducción a los programas de Extracción y Análisis de datos:

Los programas de extracción y análisis de datos tienen como funciones específicas investigar el contenido de archivos, tablas y bases de datos generando informes con los resultados que el auditor podrá incorporar a sus papeles de trabajo.

Este tipo de programas están desarrollados sobre lenguajes de interrogación de archivos y bases de datos sobre los denominados lenguajes de cuarta generación. Frente al usuario, estos programas suelen aparecer con una interfaz de tipo menú, o en sus versiones más avanzadas y ya prácticamente generalizadas con una interfaz gráfica.

Además de las funciones específicas de cualquier lenguaje de interrogación (selección de registros, obtención de totales, resumen), cuentan con rutinas específicas para los trabajos de auditoría, tales como análisis de integridad, análisis financiero, muestreo, etc.

4.2.- Importancia del Análisis de datos.

La principal importancia que radica en el análisis de datos es la aplicación de diferentes tipos de técnicas y procedimientos de auditoría. El conjunto de datos o información son de tal importancia que es necesario verificarlos y comprobarlos. La verificación del análisis de datos se puede ejecutar de distintas formas, a continuación expondremos los utilizados por nuestro proyecto:

4.3.- Verificación del análisis de datos

1. Datos de prueba

Se emplea para verificar que los procedimientos de control incluidos los programas de una aplicación funcionen correctamente. La ejecución de esta prueba dentro de la herramienta consistirá en la preparación de una serie de transacciones

que contienen tanto datos correctos como datos erróneos predeterminados para la prueba.

Uso de los datos de prueba:

- Evaluación de controles específicos.
- Verificación de validaciones.
- Prueba de perfiles de acceso.
- Prueba a transacciones seleccionadas.

2. Datos de prueba integrados

Técnica muy similar a la anterior, con la diferencia de que en ésta se debe crear una entidad para procesar los datos exportados de la base original y verificar que la información es la correcta una vez comparada con la información contenida en el sistema original.

Análisis de bitácoras: La importancia de las bitácoras es la de recuperar información ante incidentes de seguridad, detección de comportamiento inusual, información para resolver problemas y evidencia legal.

4.4.- Aspectos presentados por las Normas nacionales de auditoría para la extracción y análisis de datos.

Las normas técnicas de control interno (NTCI) según manual de la contraloría general de la república establecen los siguientes aspectos:

- **Ingresos de datos**

La máxima autoridad de cada entidad u organismo, o por su delegación los directores y jefes de la unidad administrativa serán responsables de asegurar que los sistemas automatizados tengan controles de validación de los datos al ser ingresados para procesamientos por lo que es necesario establecer alguna medida de control como las siguientes:

a) Controles de adición y validación.
b) Controles de lote;
c) Doble digitación de campos críticos;

- **Controles y procedimientos operativos.**

1- manuales de operación y controles operativos diarios.
2- Supervisión de usuarios privilegiados.
3- Control sobre software sensitivo.
4- Controles sobre el desarrollo de sistema.
5- Políticas.
6- Procedimientos y lineamientos de seguridad.
7- Funciones de administración de los empleados es seguridad.
8- Entrenamientos de los empleados en seguridad.

4.5.- Riesgos de la extracción y análisis de datos.

Los servicios de auditoría comprenden la evaluación objetiva de las evidencias, efectuada por los auditores, para proporcionar una conclusión independiente que permita calificar el cumplimiento de las políticas, reglamentaciones, normas, u otros requerimientos; respecto a un sistema, proceso, subproceso, actividad, tarea u otro asunto de la organización.

Es necesario en este sentido tener en cuenta los siguientes aspectos, los cuales se pretenden tener en cuenta en el diseño del sistema:

- La evaluación de las amenazas o causas de los riesgos.
- Los controles utilizados para minimizar las amenazas o riesgos.
- La evaluación de los elementos del análisis de riesgos.

Generalmente se habla de Riesgo y conceptos de Riesgo en la evolución de los Sistemas de Control Interno, en los cuales se asumen tres tipos de Riesgo:

4.5.1.- Tipos de riesgos

1. **Riesgo de Control:** Que es aquel que existe y que se propicia por falta de control de las actividades de la empresa y puede generar deficiencias del Sistema de Control Interno.
2. **Riesgo de Detección**: Es aquel que se asume por parte de los auditores que en su revisión no detecten deficiencias en el Sistema de Control Interno.
3. **Riesgo Inherente:** Son aquellos que se presentan inherentes a las características del Sistema de Control Interno.

Entre una gran diversidad de situaciones, es posible mencionar las siguientes:

- Omisión deliberada de registros de transacciones.
- Falsificación de registros y documentos.
- Proporcionar al auditor información falsa.

5.- Cobit 4.0

5.1.- Objetivos de Control para la información y Tecnologías Relacionadas (Cobit 4.0).

Cobit es una herramienta para gobernar la Tecnología de Informática. La Misión de COBIT es:

Investigar, desarrollar, publicar y promover un conjunto de objetivos de control en tecnología de información con autoridad, actualizados, de caracteres internacionales y aceptados generalmente para el uso cotidiano de gerentes de empresas y auditores.

5.2.- Criterios de información de Cobit 4.0

Para satisfacer los objetivos del negocio, la información necesita adaptarse a ciertos criterios de control, los cuales son referidos en COBIT como requerimientos de información del negocio. Con base en los requerimientos de calidad, fiduciarios y de seguridad, se definieron los siguientes siete **criterios de información:**

a. **La efectividad:** tiene que ver con que la información sea relevante y pertinente a los procesos del negocio, y se proporcione de una manera oportuna, correcta, consistente y utilizable.

b. **La eficiencia** consiste en que la información sea generada optimizando los recursos (más productivo y económico).

c. **La confidencialidad** se refiere a la protección de información sensitiva contra revelación no autorizada.

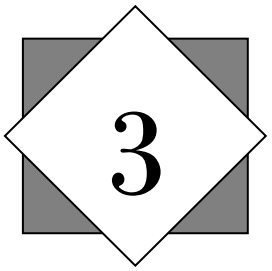
d. **La integridad** está relacionada con la precisión y completitud de la información, así como con su validez de acuerdo a los valores y expectativas del negocio.

e. **La disponibilidad** se refiere a que la información esté disponible cuando sea requerida por los procesos del negocio en cualquier momento. También concierne con la protección de los recursos y las capacidades necesarias asociadas,

f. **El cumplimiento** tiene que ver con acatar aquellas leyes, reglamentos y acuerdos contractuales a los cuales está sujeto el proceso de negocios, es decir, criterios de negocios impuestos externamente, así como políticas internas.

g. **La confiabilidad** significa proporcionar la información apropiada para que la gerencia administre la entidad y ejercite sus responsabilidades fiduciarias y de gobierno.

DISEÑO METODOLOGICO



1. Tipo de investigación.

Según la aplicabilidad del conocimiento obtenido se clasificó como una investigación aplicativa, ya que se pretende solucionar problemas con referencia a los procedimientos manuales de auditoria mediante la automatización de estos, aplicando normas, técnicas y procedimientos de auditoria informatica.

2. Diseño de investigación.

El tipo de diseño de investigación condiseramos que es de campo ya que los datos provienen de consultas hechas directamente al auditor además de recomendaciones hechas por el mismo, también se hizo uso de fuentes bibliográficas a partir de las cuales elaboramos el marco teórico.

3. Lenguaje utilizado para el desarrollo de la herramienta.

El lenguaje en el que se desarrollara el software es una lenguaje de cuarta generacion (4GL): Microsoft Visual Basic 6.0 Edición profesional. Algunas de las caracteristicas por las que seleccionamos este lenguaje son:

- Es un lenguaje de programación basado en objetos.
- Permite el diseño de interfaces gráficas con simplicidad.
- Uso de la capacidad de Visual Basic para crear nuevos controles ActiveX, de modo que pueda aumentar las opciones de programación y las herraminetas con que sus usuarios interactuen.
- Fácil portabilidad del código hacia diferentes sistemas operativos de Microsoft.

4. **Gestor de base de datos utilizado:**

SQL-Server 2000

SQL Server es un conjunto de objetos eficientemente almacenados. Los objetos donde se almacena la información se denominan tablas, y éstas a su vez están compuestas de filas y columnas. Algunas de las características por las que elegimos SQL Server 2000 como gestor de base de datos son las siguientes:

- contiene herramientas de desarrollo integradas como Visual Studio 6.0 o .NET
- Incorpora un modelo de objetos totalmente programable (SQL-DMO) con el que podemos desarrollar cualquier aplicación que manipule componentes de SQL Server, es decir, crear bases de datos, tablas, DTS, backups, etc., todo lo que se puede hacer desde el administrador del SQL Server y podemos hacerlo no solo en Visual C++ sino también en Visual Basic, ASP y por supuesto en .NET.

5. **Estrategias de prueba para el análisis de datos generados en los sistemas se información.**

En la auditoría de sistemas informáticos se han desarrollado muchas técnicas para el análisis y extracción de datos y es por ello que una herramienta que apoye esta tarea beneficiará en gran manera al auditor a generar informes con mayor rapidez. Se necesita que esta herramienta permita Importar datos desde varios orígenes específicos: Microsoft Access 97-2003, Microsoft Excel 97-2003, SQL SERVER 2000, Archivos de Texto, al ambiente de la herramienta de software, con el propósito de extraerla y analizarla mediante procedimientos preprogramados.

La herramienta deberá contar con la capacidad de realizar algunos controles,

1. *controles de exactitud*, (validación de campos numéricos, validación de exceso en un campo, conteo de registros, cifras de control de valores, etc.).
2. co*ntroles de totalidad,* (validación de campo en blanco, conteo de registro, cifras control de valores, validación secuencia de registros, etc.).

3. *Controles de redundancia* (Sirven para asegurar que los registros son procesados una sola vez; como por ejemplo: sello de cancelación de lotes, verificación de secuencia de registros, archivo de suspenso, cifras control, etc.).

Para llevar a cabo estos controles se deberán de realizar las pruebas respectivas como las siguientes:

a. **Prueba de comparaciones, cálculos, registros y acumulaciones.**

Las transacciones de prueba pueden ser preparadas y sometidas a cálculos, imputaciones y acumulaciones. Si los resultados reales concuerdan con lo previsto existirá una razonable seguridad de que las funciones de procesamiento computarizadas funcionan adecuadamente.

b. **Prueba de totales de control.**

Se puede utilizar recuentos de registros y otros totales de control generados durante el procesamiento, para detectar transacciones no autorizadas, faltantes, duplicadas o erróneas. Se pueden usar transacciones de pruebas ingresadas y procesadas como un lote separado para verificar que los totales de control generados por el sistema sean correctos.

c. **Prueba o ejecución de cálculos.**

Los programas de recuperación y análisis pueden ser utilizados para probar los cálculos, ya sea rehaciendo los mismos, o mediante comprobaciones globales de razonabilidad.

Otras funcionalidades que deberá contener la herramienta son:

a. A menudo la forma en que la información es almacenada o presentada no es conveniente para la ejecución de los procedimientos de auditoría , por lo tanto el

auditor puede *resumir y reordenar* la información de la manera más conveniente para ser analizada por él mismo, tomando esto en cuenta la herramienta deberá tener la capacidad de extraer un conjunto de registro utilizando las siguientes modalidades: Procedimientos de muestreo y Definiendo el criterio de selección de los registros o filtrados de información.

b. La herramienta deberá exportar los registros seleccionados a diferentes ambientes para otros tipos de análisis y/o presentaciones, como Excel y Word 97-2003.

c. Además la herramienta deberá permitir ubicar rápidamente tablas y campos específicos utilizando *herramientas de búsqueda* a través de la interfaz, que le permita al auditor buscar por descripción, nombre de tabla y por módulo.

d. Crear funciones analíticas de comparación de datos, cálculo de razones, identificación de fluctuaciones en los registros almacenados cuando llevan numeraciones consecutivas. Además de las propias técnicas de análisis y extracción de datos de la auditoría, la herramienta deberá de guardar un historial de las operaciones realizadas por los usuarios de esta herramienta.

3. Cada usuario deberá de registrarse debidamente con un login y un password para iniciar sesión en el sistema, que además siguiendo con lo que establece el inciso a. de la norma 11.6.1"Control de acceso a las aplicaciones y la información" de la NTP-ISO/IEC 17799, se deberá controlar el acceso de los usuarios a la información y las funciones del sistema de aplicación de acuerdo con la política de control de accesos.

Por lo expuesto anteriormente la bitácora deberá registrar información acerca de eventos relacionados con el sistema, algunos de los cuales pueden ser:

- Fecha y hora.
- Direcciones IP origen y destino.
- Dirección IP que genera la bitácora.
- Usuarios.

- Errores.

6. Técnicas y procedimientos de auditoria aplicadas a la herramienta:

Siguiendo las clasificaciones de las técnicas y procedimientos mencionadas anteriormente (capítulo 2) nuestra investigación, destinada al diseño de una herramienta para la extracción y análisis de datos se apoyara de las siguientes técnicas de auditoría : a) tiene la capacidad de realizar operaciones correspondientes a la confirmación de datos, b) inspección de registros, calculo de operaciones y c) la declaración de datos, así como su base estará enfocada en los siguientes procedimientos de auditoría : 1) prueba y detalles de transacciones, 2) procedimientos analíticos y 3) pruebas de recálculos. Brindándole al auditor de esta manera una mejor visión de lo que está sucediendo dentro de la empresa y poder emitir su criterio de la forma más acertada y eficiente.

7. Objetivos de Control para la información y Tecnologías Relacionadas (Cobit 4.0).

- **Criterios de información de Cobit 4.0 aplicadas a la información generada por la herramienta.**

a. Efectividad: Este criterio es aplicado en la herramienta a implementar, ya que los datos proporcionados por el cliente serán evaluados y la información final generada por la herramienta será el indicador de que los datos cumplan con este criterio.

b. Eficiencia. Este criterio se aplicará de la siguiente manera: la información generada por la herramienta será proporcionada en menor tiempo y con mejor calidad, gracias a la automatización de los procedimientos de auditoría.

c. Confidencialidad. Al auditor se le proporcionará acceso a los datos de la organización para su debida evaluación. El auditor no podrá por ningún motivo

revelar información privada de la organización. La información generada por la herramienta estará disponible únicamente para las personas que la necesitan utilizar y que tienen sus respectivos nombres de usuarios y claves de acceso.

d. La integridad: la herramienta proporcionará exactamente la información solicitada por el auditor.

e. Disponibilidad. Este criterio será aplicado de la siguiente manera: Los trabajos realizados por el auditor se almacenaran en una base de datos lo que asegura que la información estará disponible para una futura consulta.

f. Cumplimiento: La información generada por la herramienta estará sujeta a técnicas y procedimientos de auditoría con relación a la extracción y análisis de datos.

g. Confiabilidad: La información que será generada por la herramienta no podrá incurrir en ningún tipo de errores.

A continuación definiremos el tipo de sistema de información de la herramienta:

7. Tipo de sistema de informacion:

El tipo de sistema al que irá enfocado nuestro trabajo es el operacional, ya que la herramienta servirá para recopilar información de forma correcta, sin que se produzcan errores, redundancias o inconsistencias.

8. Herramientas CASE utilizadas.

- **Rational Rose:**

La herramienta CASE utilizada para realizar el diseño de la herramienta es RATIONAL ROSE 2001, con esta herramienta se crearon los diagramas de clases y de casos de uso para la herramienta "HEAD" y para su módulo de seguridad (ver

anexos 1, 2, 3, 4, 5, 6), esta herramienta fue elegida ya que proporciona diferentes modelos para el diseño de sistemas.

- **ERwin:**

All Fussion ERwin Data Modeler 7.1.0 es la herramienta CASE que se utilizó para el diseño de la base de datos por ser una de las herramientas q soporta base de datos como SQL Server, Oracle, Sybase, DB2 e Informix, una de las principales ventajas de esta herramienta es que el mismo modelo puede ser usado para generar múltiples bases de datos, o convertir una aplicación de una plataforma de base de datos a otra, también genera automáticamente las tablas y miles de líneas de procedimiento almacenado y triggers para los principales tipos de base de datos. Ver anexos 7y 8.

9. Analisis de riesgos en la extracción y análisis de datos:

La herramienta no podrá tener control con respecto a todos los tipos de riesgos que existen en la extracción y análisis de datos ya que el riesgo de detección existe debido a deficiencias o irregularidades no detectadas, falta de informacion o falsa informacion facilitada al auditor, con referencia al riesgo de control la herramienta contará con un módulo de seguridad en el que existirá un usuario administrador que tendrá la función de proporcionar al personal autorizado (auditores) su nombre de usuario y contraseña, ademas de registrar en una bitácora todas las acciones realizadas por los diferentes usuarios, sin embargo la herramienta no podrá asegurar que el personal autorizado no revele sus datos de usuario a terceras personas (riesgo inherente) por lo tanto la herramienta juega un papel muy importante con relación al riesgo de control ya que esta llevará un control en la administración de usuarios y bitácora de acciones pero no se podran evaluar los otros dos tipos de riesgos.

10. Sistema para la gestion de contraseñas.

Para la gestion de contraseñas de usuario se utilizó un modelo de sistema criptográfico conocido como "Criptosistema Hill", elegido por ser un sistema práctico y polialfabético que no encripta un carácter con el mismo simbolo, lo que lo convierte en un sistema de encriptacion confiable.

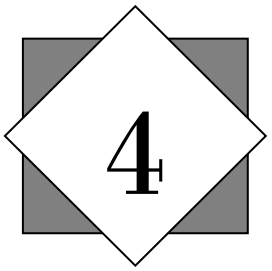
11. Pruebas realizadas al sistema:

Inicialmente se realizó la **prueba de unidad** comprobando que se ejecuta al menos una vez todos los caminos posibles de cada módulo, también se evaluaron las instrucciones lógicas así como también se comprobó que no existan ciclos infinitos, seguimos con la **pprueba de integración** la cual se centró en la verificación de parámetros correctos entre módulos evaluando cantidad y tipos de datos definidos para cada módulo, la siguiente prueba a la que fue sometida la herramienta fue la **prueba de validación** la cual consistió en detectar y corregir errores encontrados por usuarios en los siguientes tipos de entorno:

1. Entorno controlado o prueba alfa: la que consistió en la utilización de la herramienta bajo la supervisión del desarrollador, este a la vez registro los errores encontrados con lo cual fue posible mejorar la herramienta.
2. Entorno no controlado o prueba Beta: esta prueba consistió en que el usuario utilizara la herramienta sin ninguna supervisión y generara un informe sobre los errores encontrados para que fueran rectificados por el desarrollador.

También se realizó la **prueba de seguridad** la cual consistió en tratar de ingresar al sistema utilizando login correctos con password incorrectos y viceversa, prueba en la cual no se tuvo ningún éxito con lo cual la herramienta se considera segura.

ESTUDIO DE FACTIBILIDAD



1. Factibilidad técnica:

La tecnología necesaria para implementar la herramienta para la extracción y análisis de datos debe prestar los siguientes requerimientos mínimos:

- Sistemas operativos: Windows 2000, Windows 2003, Windows XP o superior.
- Espacio libre en disco: 32 Megabytes.
- Cantidad de memoria: 32 Megabytes.
- Características del procesador: Intel Pentium III o superior y compatibles, con velocidad de 800 MHz.
- Características del adaptador de pantalla: Tarjeta de video VGA con controlador de video con resoluciones de 800x600 pixeles con 32 bits de colores.

2. Factibilidad económica*:*

A continuación detallaremos los costos del proceso investigativo:

1. Costos de Hardware*

Concepto	**p/u. US $**	**Cantidad**	**Total US $**
Procesador Intel core 2 Duo 2.66 GHZ	186.29	1	186.29
Memoria Ram de 1 GB 2.3 GHZ	64.00	1	64.00
Multi DVD/RW LG	35.19	1	35.19
Disco Duro de 160 GB 7200 RPM SATA	58.71	1	58.71
UPS Tripp Lite PRO 550 VA	61.11	1	61.11
Ratón óptico USB	5.35	1	5.35
Teclado	6.22	1	6.22
Impresora HP deskject 1560	40.25	1	40.25
		Total US $	**457.08**

* Fuente: DATATEX, COMTECH, Impresiones "SAN JOSE"

2. **Costos de Software:***

Software	**Costo US $**
Microsoft Windows XP Professional	215.23
SQL Server 2000	914.25
Total US $	1129.48

3. **Otros***

Papelería/Otros	**p/u. C$**	**Cantidad**	**Total C$**
Cartucho HP 21	428.49	1	428.49
Resma de papel	120.00	2	240.00
Encolochados y empastado	1200.00	-	1200.00
		Total C$	**1868.49**

3. Factibilidad operacional:

La aplicación incluye la ayuda que describe el uso de la interfaz de usuario y el uso de las características de la herramienta.

Entre los usuarios finales de esta herramienta se encuentran:

- Auditores informáticos.
- Estudiantes de carreras de licenciatura, ingeniería en computación, informática y también se incluyen estudiantes de contabilidad y auditoría que deseen ampliar sus conocimientos al área de auditoría informática.
- Profesionales del área administrativa.

Por lo mencionado anteriormente se garantiza que el desarrollo de esta herramienta es factible.

* Fuente: DATATEX, COMTECH, Impresiones "SAN JOSE"

RESULTADOS

A continuación se presenta parte de los algoritmos utilizados para la creación de las pruebas que realiza la herramienta de extracción y análisis de datos "HEAD".

- **Algoritmo PRUEBA DE SECUENCIA** sobre los valores de un campo especificado de una tabla de datos.

Objetivo: En esta prueba el usuario selecciona un (1) campo de tipo Entero, el cual será analizado para probar su respectiva secuencia en los valores que almacena., antes de iniciar la prueba los registros de la Tabla se ordenarán por el campo seleccionado de forma ASCENDENTE.

Variables en uso

(i): especifica al valor Anterior del campo en la Prueba

(Pivote) : especifica al valor Siguiente del campo en la Prueba.

(c): contador de las anomalías en la tabla.

(Campo seleccionado): es el campo por el cual se hará la prueba.

INICIO

Paso 1. Abrir Tabla

Paso 2. Ordenar los registros de forma ASCENDENTE por el campo seleccionado

Paso 3: Apuntar al primer registro de la Tabla

Paso 4: Hacer i = valor del Primer campo seleccionado (Primer Registro)

Paso 5: Apuntar al siguiente registro de la Tabla

Paso 6: While Not (EOF)

Paso 5:1. Hacer Pivote = valor del campo del registro actual

Paso 5.2. If ((Pivote – i) <> 1) Then Print 'Campos no Secuenciales'

Paso 5.3. c = c + 1

Paso 5.4. i = Pivote /*Valor Anterior = Valor Siguiente

Paso 5.5: Apuntar al siguiente registro de la Tabla

Paso 7: End While

Paso 8: If (c = 0) Then

Print 'prueba de secuencia completada exitosamente, sin problemas encontrados.'

Paso 9: Else If (c > 0) Then

Print 'prueba de secuencia completada con: ', c , 'problemas encontrados.'

Paso 10. End If

FIN

- **Algoritmo PRUEBA DE DUPLICADOS** sobre los valores de un campo especificado de una tabla de datos.

Objetivo: En esta prueba el usuario selecciona un (1) campo de tipo Entero, el cual será analizado para probar su respectiva secuencia en los valores que almacena., antes de iniciar la prueba los registros de la Tabla se ordenarán por el campo seleccionado de forma ASCENDENTE.

Variables en uso

(i): especifica al valor Anterior del campo en la Prueba

(Pivote) : especifica al valor Siguiente del campo en la Prueba.

(c): contador de las anomalías en la tabla.

(Campo seleccionado): es el campo por el cual se hará la prueba.

INICIO

Paso 1. Abrir Tabla

Paso 2. Ordenar los registros de forma ASCENDENTE por el campo seleccionado

Paso 3: Apuntar al primer registro de la Tabla

Paso 4: Hacer i = valor del Primer campo seleccionado (Primer Registro)

Paso 5: Apuntar al siguiente registro de la Tabla

Paso 6: While Not (EOF)

Paso 5:1. Hacer Pivote = valor del campo del registro actual

Paso 5.2. If (Pivote = i) Then Print 'Campos Duplicados'

Paso 5.3. c = c + 1

Paso 5.4. i = Pivote /*Valor Anterior = Valor Siguiente

Paso 5.5: Apuntar al siguiente registro de la Tabla

Paso 7: End While

Paso 8: If (c = 0) Then

Print 'prueba de duplicados completada exitosamente, sin problemas encontrados.'

Paso 9: Else If (c > 0) Then

Print ‘prueba de duplicados completada con: ‘ c ‘problemas encontrados.’

Paso 10. End If

FIN

CONCLUSIONES

El punto de partida del proceso investigativo fue la recopilación de información mediante la utilización de fichas bibliográficas y mediante consultas realizadas al usuario final, lo cual hizo posible la representación teórica de la herramienta para la extracción y análisis de datos.

La información generada por la herramienta se adapta a los requerimientos de información contemplados en COBIT 4.O (Objetivos de Control para la Información y tecnologías relacionadas), además de cumplir con las normas para la extracción y análisis de datos establecidas por la CGR (Contraloría General de la República de Nicaragua).

Esta herramienta ya que automatiza los procedimientos de auditoría referentes a la extracción y análisis de datos disminuye el tiempo de evaluación y respuesta de los trabajos realizados, posee una bitácora de acciones para cada usuario y también cumple con el inciso 11.5.2 "identificación y autenticación de usuario" de la Norma Técnica Peruana ISO/IEC 17799, lo cual con asegura que los proyectos realizados no puedan ser modificados o visualizados por personal no autorizado de tal forma que garantiza la privacidad de los proyectos para cada usuario.

RECOMENDACIONES

- No se debe instalar la aplicación en computadoras q no cumplan con los requerimientos establecidos en la parte de factibilidad técnica antes mencionada.
- Completar algunas partes del proyecto que ayudarán a complementar la funcionalidad del mismo, como por ejemplo, la notificación de vencimiento de contraseña del usuario, generación de un Histograma, y Prueba de comparación de campos.

BIBLIOGRAFIA

[Kroenke, 1996]

Kroenke, D., "**Procesamiento de bases de datos, Fundamentos, Diseño e instrumentación**", Edo. Prentice Hall. 5ta. Edición.1996.

[Burch, 1989]

Burch J., Grudnitski G. "**Information systems: Theory and Practice**", John Wiley & Sons. Fifth Edition. 1989.

[Booch, Rumbaugh, Jacobson, 2005]

Booch, Grady; Rumbaugh, James; Jacobson, Ivar.," **El lenguaje unificado de modelado**", Edo. Addison-Wesley. 2a. Edición. 2005.

[IT Governance Institute, 2005]

IT Governance Institute, COBIT 4.0, El "Trabajo", 2005

[De Marco, 1978]

De Marco T." Structures analysis and system specification". Prentice Hall, 1978.

[EQV. ISO/IEC, 2007]

EQV. ISO/IEC 17799-2005. **"Information technology code of practice for information security managment"**. 2nd. Edition. 2007.

[Gil, 1994]

Gil, I., **"Sistemas de información para la gestión empresarial"**. Servicios de publicaciones, Universidad Politécnica de Valencia, 1994.

[Laudon, 1994]

Laudon K., Laudon J. **"Management information systems, Organization and Technology"**. Third Edition. McMillan Colleege Publishing Company, 1994.

[Muños, 1998]

Muños. C., **"Como elaborar y asesorar una investigación de tesis"**, Edo. Prentice Hall. 1ra Edición.1998.

[Piura, 1994]

Piura, J., **"Introducción a la Metodología de la Investigación Científica"**, Edo. El Amanecer. Material Inédito. 1998.

[Pressman, 2002]

Pressman, R., **"Ingeniería del software: Un enfoque practico"**, Edo. McGraw Hill. 5ta. edición. 2002.

[Sommerville, 2002]

Sommerville, I.," Ingeniería del Software", Edo. Addison-Wesley. 6ta. Edición. 2002.

[Sommerville, 2005]

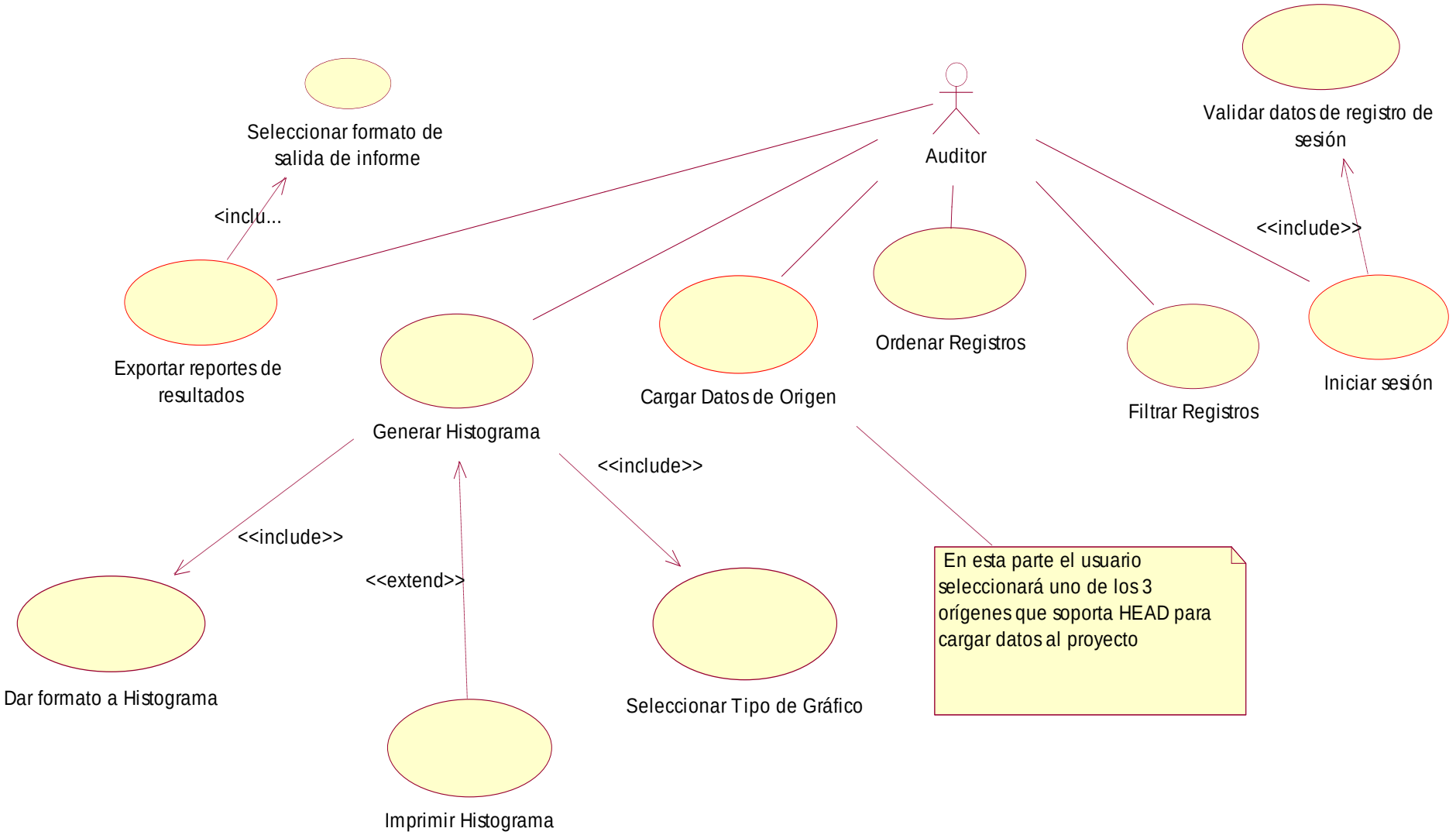
Somerville, I.**"Ingeniería del software"**, Edo. Pearson Educación S.A, 7ma. Edición. 2005.

Bibliografía Internet

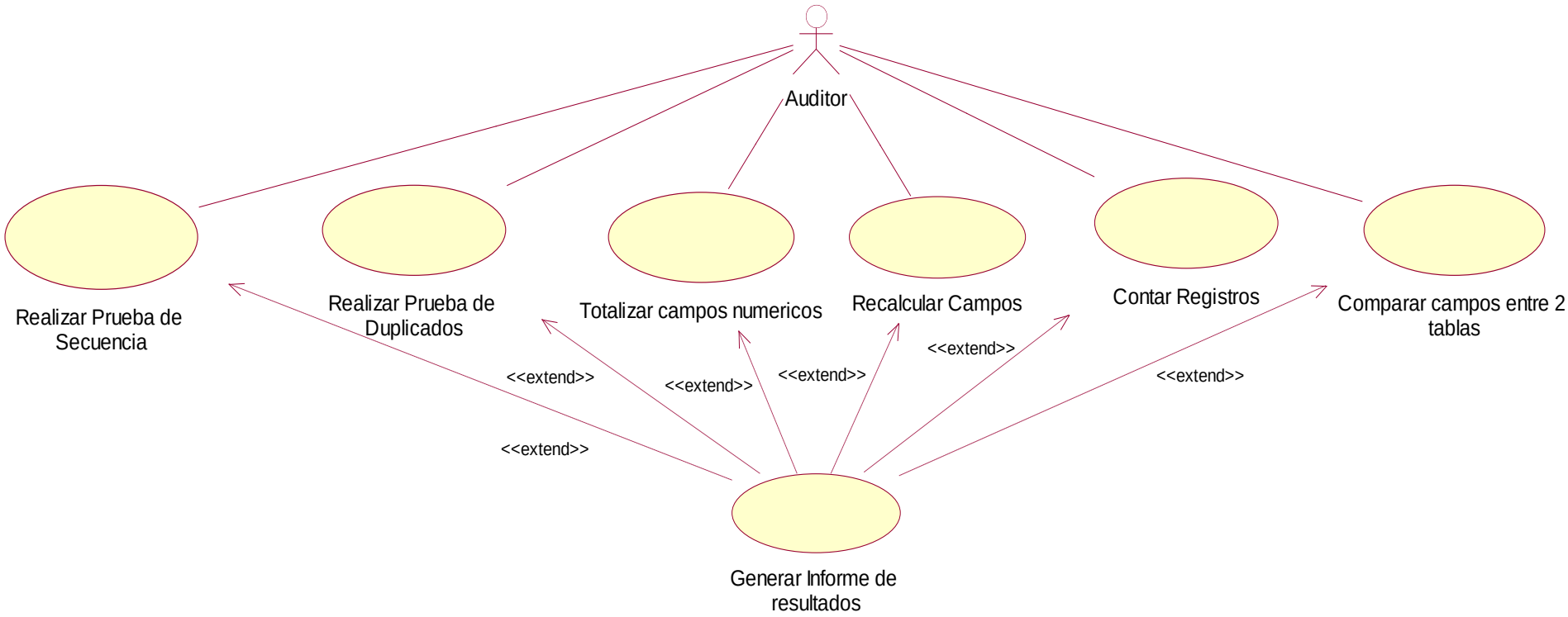
Contraloría General de la República, "**Manual de Auditoria Informática", en** *http://www.cgr.gob.ni/cgr/index.php?option=com_docman&task=cat_view&gid=173&Itemid=101*

ANEXOS

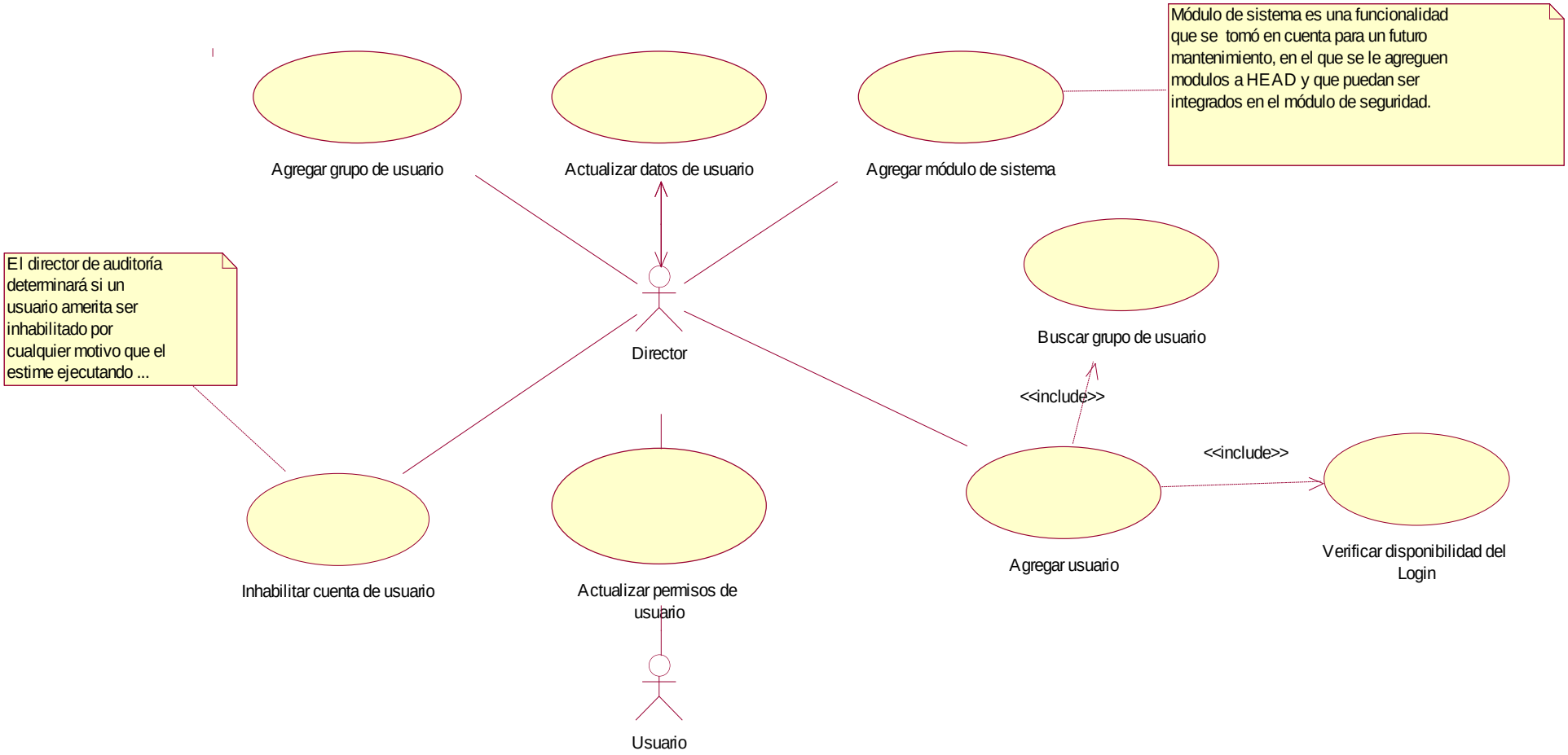
ANEXO 1: Diagrama de casos de uso de herramienta "HEAD".**PARTE 1**.



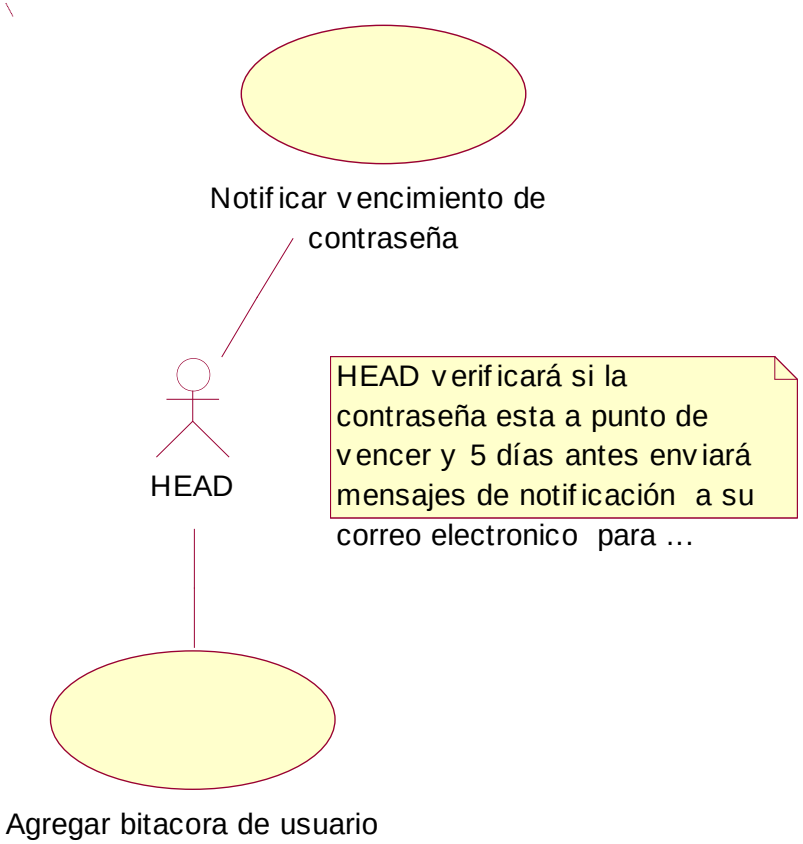
ANEXO 2: Diagrama de casos de uso de la herramienta "HEAD".**PARTE 2.**



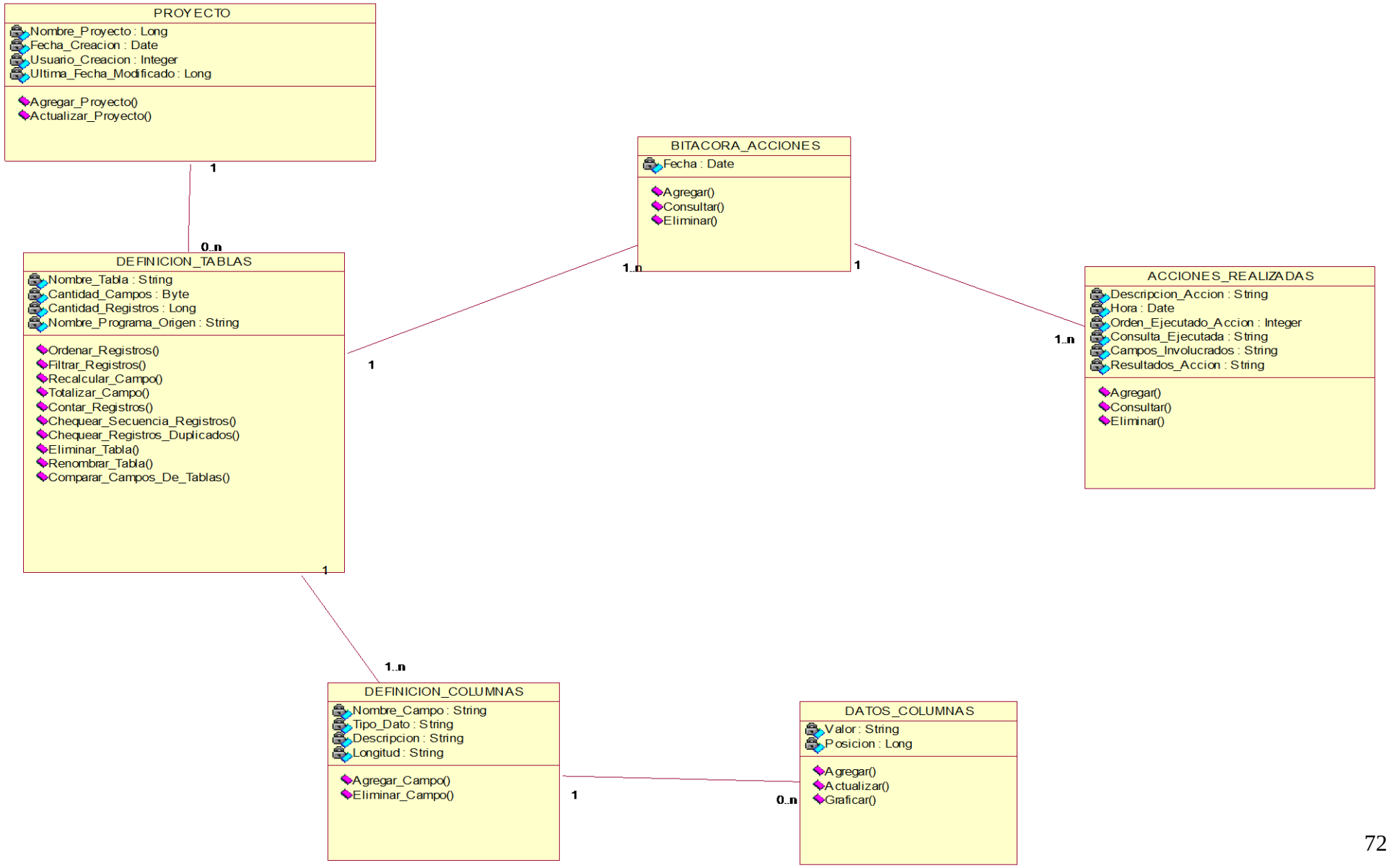
ANEXO 3: Diagrama de casos de uso de módulo de seguridad de la herramienta "HEAD".**PARTE 1**.



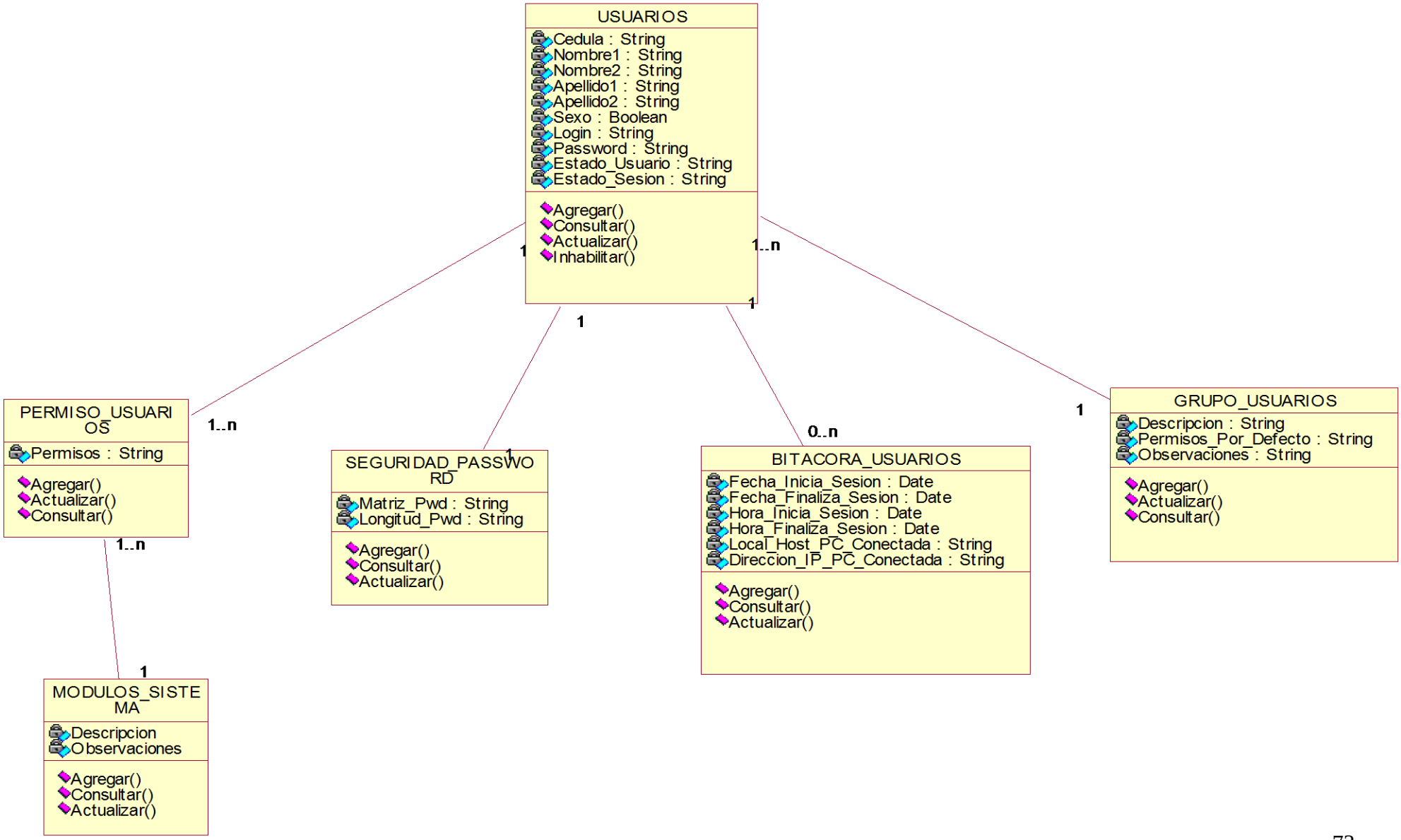
ANEXO 4: Diagrama de casos de uso de módulo de seguridad de la herramienta "HEAD"**. PARTE 2.**



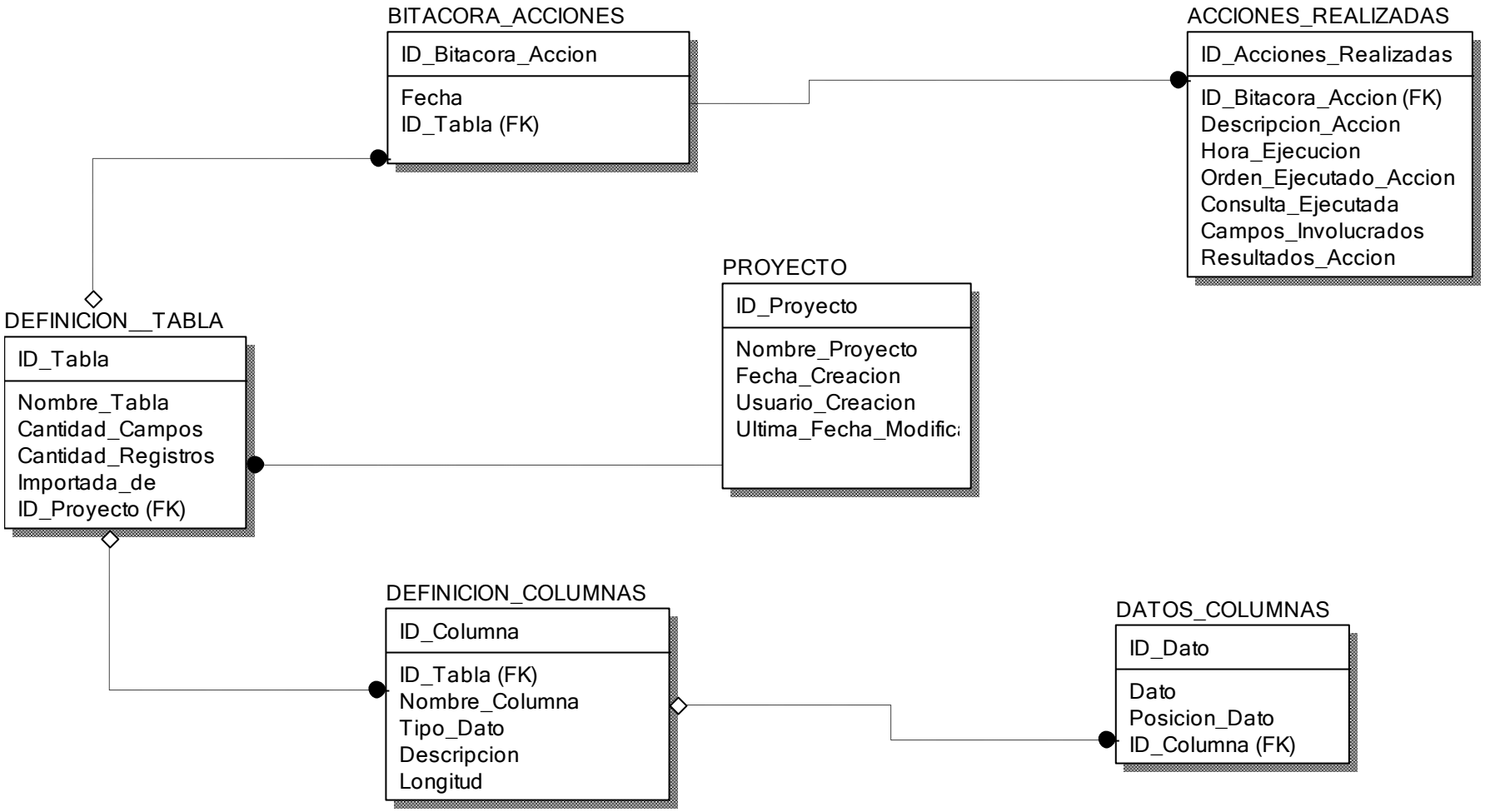
ANEXO 5: Diagrama de clases de la herramienta "HEAD".**PARTE 1.**



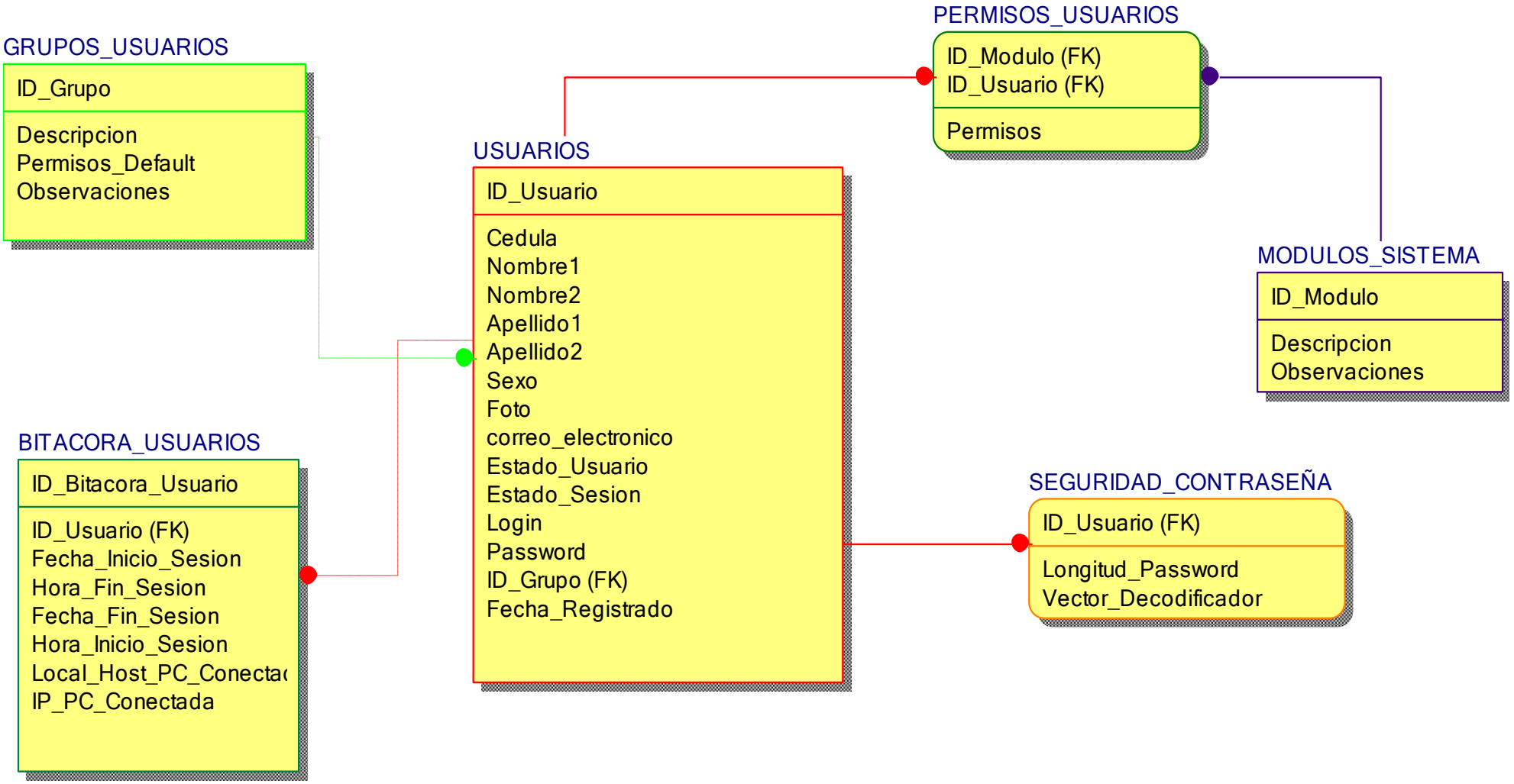
ANEXO 6: Diagrama de clases del módulo de seguridad de la herramienta "HEAD".



ANEXO 7: Diagrama relacional de la herramienta "HEAD"..



ANEXO 8: Diagrama relacional del módulo de seguridad de la herramienta "HEAD".



ANEXO 9: Descripción de actores y casos de uso de "HEAD".

Actor	Descripción
Director	El director trabaja en el área de auditoria y es el responsable de todos los auditores, es el que lleva control general sobre las auditorias realizadas por cada uno de los auditores
Usuario	Es el usuario del sistema, es el auditor que se apoya con la herramienta de software para realizar sus controles referentes a la extracción y análisis de datos
Sistema	La herramienta como tal realiza operaciones en trasfondo del usuario como revisar el vencimiento de las cuentas, registrar la bitácora de acceso, entre otras.

Caso de uso	Descripción
Iniciar sesión	Los usuarios al ingresar al sistema deberán pasar un registro de sesión, que los autoriza a usar el sistema, en este se valida un login y un password correctos.
Validar datos de registro de sesión	Los usuarios después de haber proporcionados los datos de login y password para ingresar al sistema, éste validará los datos introducidos
Cargar datos de origen	Esta parte permitirá a los usuarios seleccionar el origen de los datos que almacenara en el proyecto para realizarle las diferentes operaciones de auditoria señaladas en el proyecto
Registrar bitácora de acciones	Cuando un usuario realiza una operación (acción) en el sistema como cargar datos, o realizar algún tipo de prueba de los controles de auditoria que se implementa, esta acción será registrada en el sistema

Caso de uso	Descripción
Generar informe de resultados	El usuario una vez que ha realizado diversas pruebas el sistema presentará este resultado en un informe que podrá ser exportado a diferentes formatos como: Excel, PDF, Word, etc.
Contar registros	Una de las pruebas que realiza el sistema como parte de los controles que lleva el auditor es la del conteo de registros, que totalizara los registros cargados a una tabla.
Totalizar campos numéricos	Otra prueba que podrá realizar el usuario ("auditor") del sistema es la de totalizar campos numéricos, para cotejar posiblemente el dato con documentos físicos.
Recalcular campos	Con el objetivo de que el usuario pueda revisar los datos de un campo previamente calculado, podrá crear un nuevo campo para recalcularlo y cotejar los datos de los dos campos involucrados en la prueba.
Generar Histograma	El usuario podrá generar un Histograma en base a un campo numérico para presentar de una mejor manera "gráfica", la información.
Realizar prueba de secuencia	Otra prueba que implementa la herramienta de software es la revisión de la secuencia en campos que son de tipo numérico y que deberían estar libres de fluctuaciones.
Comparar campos entre tablas	Otra manera de evaluar la integridad de los datos es la revisión de los mismo, cuando éstos son almacenados en tablas diferentes una prueba de comparación entre estos campos podrá arrojar información útil para el auditor.
Filtrar Registros	Los usuarios podrán filtrar el conjunto de registros proporcionando criterios de filtrado al sistema para que este los ejecute sobre una tabla.
Realizar prueba de Duplicados	Para verificar la integridad de los datos almacenados en una Base de datos se implementa la prueba de revisar registros duplicados en alguna tabla.

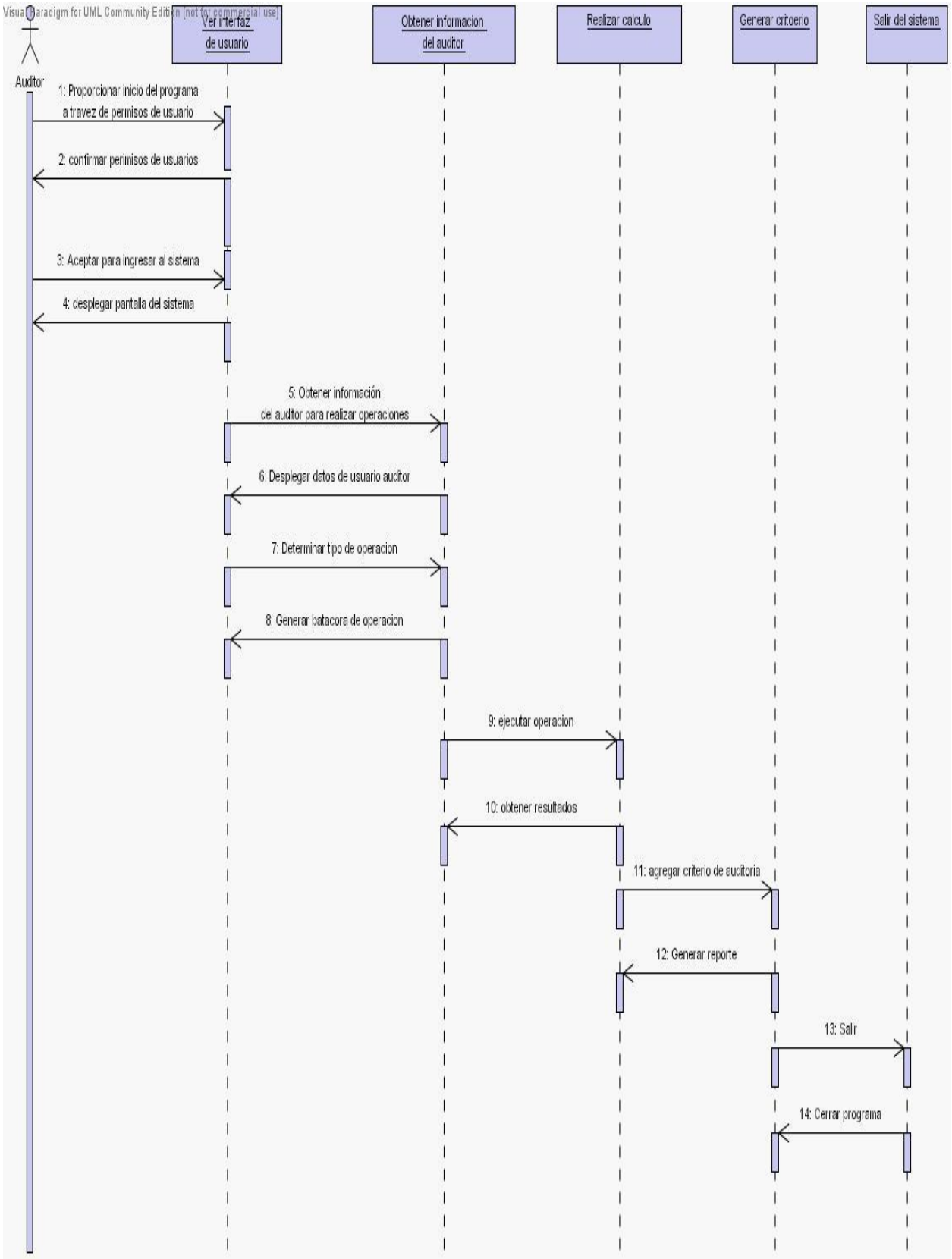
Ordenar registros	En ocasiones el usuario deseará presentar la información de una manera más ordenada los registros en la tabla, la herramienta facilitará esta función.

ANEXO 10: Descripción de actores y casos de uso del módulo de seguridad de "HEAD".

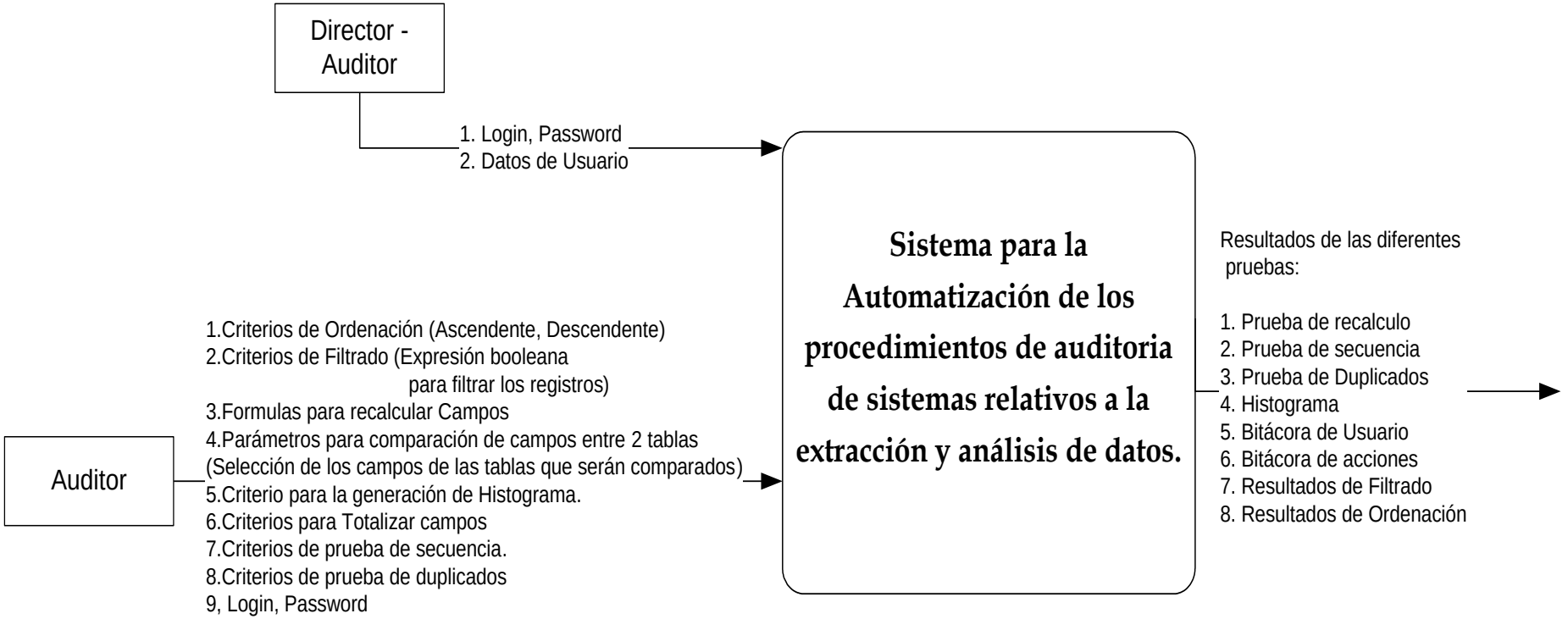
Caso de uso	Descripción
Agregar grupo de usuario	Los usuarios en el sistema tienen roles y con esto permisos a las funciones programadas de la herramienta, por lo que los usuarios están agrupados para realizar sus tareas permitidas.
Agregar modulo de sistema	El sistema posteriormente puede ser objeto de una mejora o agregado que incremente sus capacidades funcionales y es por esto que permite se agregue un modulo al sistema y posteriormente ajustar permisos de acceso a los usuarios de dicho modulo
Agregar usuario	La herramienta permite la opción de agregar un nuevo usuario al sistema, datos de cedula, nombre, login, password, etc. Serán registrados como parte del registro del usuario.
Inhabilitar cuenta de usuario	Un usuario de la empresa puede tener varias situaciones laborales como para no permitírsele el acceso al sistema, situaciones como despido, suspensión o traslado a otra área, son razones suficientes para inhabilitar la cuenta del mismo y que ya no pueda acceder al sistema.
Actualizar datos de usuario	En muchas ocasiones, y por razones de seguridad sobre las operaciones que el usuario realiza es obvia la necesidad de actualizar los datos principalmente la contraseña.
Notificar vencimiento de contraseña	HEAD verificará si la contraseña esta a punto de vencer y 5 días antes enviará mensajes de notificación a su correo electrónico para su actualización
Agregar bitácora de usuario	La herramienta automáticamente registrara la fecha, hora de acceso, usuario, entre otros datos que sirvan para efectos de llevar un historial sobre el ingreso al sistema.

Nota: A este módulo únicamente tendrá acceso el usuario administrador del sistema, que en este caso sería el Director del área en la cual se llevan a cabo las auditorías.

ANEXO 11: Diagrama de Secuencia de "HEAD".



ANEXO 12: Diagrama de nivel 0 de "HEAD"



ANEXO 13: Diccionario de Datos de "HEAD" Primera parte

#	Nombre de la Tabla	Nombre del Campo	Tipo de Datos	Longitud
1	**tbl_ACCIONES_REALIZADAS**			
		ID_Accion_Realizada	int	4
		Orden_Ejecutado_Accion	smallint	2
		Descripcion_Accion	varchar	300
		Consulta_Ejecutada	Varchar	100
		Campos_Involucrados	Varchar	100
		Resultados_Accion	Varchar	800
		Hora_Ejecucion	smalldatetime	4
		ID_Bitacora_Accion	int	4
2	**tbl_BITACORA_ACCIONES**			
		ID_Bitacora_Accion	int	4
		Fecha	smalldatetime	4
		ID_Tabla	int	4
3	**tbl_CONFIGURACION**			
		ID_Usuario	int	4
		Color_Esquema	tinyint	1
		Limite_Proyectos_Recientes	tinyint	1
		Encabezado_Reporte	varchar	50
		Encabezado_Tipo_Fuente	varchar	50
		Encabezado_Tamano_Fuente	tinyint	1
		Pie_Reporte	varchar	50
		Pie_Tamano_Fuente	tinyint	1
		Pie_Tipo_Fuente	varchar	50
		Imagen_Reporte	image	16
4	**tbl_DATOS_COLUMNAS**			
		ID_Dato	int	4
		Dato	varchar	100
		Posicion_Dato	int	4
		ID_Columna	int	4

ANEXO 14: Diccionario de Datos de "HEAD" - Segunda parte

#	Nombre de la Tabla	Nombre del Campo	Tipo de Datos	Longitud
5	**tbl_DEFINICION_COLUMNAS**			
		ID_Columna	int	4
		Nombre_Columna	varchar	50
		Tipo_Dato	varchar	20
		Descripcion	varchar	50
		Longitud	varchar	5
		ID_Tabla	int	4
6	**tbl_DEFINICION_TABLAS**			
		ID_Tablas	int	4
		Nombre_Tabla	varchar	50
		Cantidad_Campos	tinyint	1
		Cantidad_Registros	int	4
		Importada_de	varchar	50
		ID_Proyecto	int	4
7	**tbl_PROYECTOS**			
		ID_Proyecto	int	4
		Nombre_Proyecto	varchar	50
		Fecha_Creacion	smalldatetime	4
		Usuario_Creacion	int	4
		Ultima_Fecha_Modificado	smalldatetime	4

ANEXO 15: Diccionario de Datos del modulo de seguridad de "HEAD" Primera parte

#	Nombre de la Tabla	Nombre del Campo	Tipo de Datos	Longitud
1	**tbl_BITACORA_USUARIOS**			
		ID_Bitacora	int	4
		Fecha_Inicio_Sesion	varchar	20
		Fecha_Fin_Sesion	varchar	20
		Hora_Inicio_Sesion	varchar	20
		Hora_Fin_Sesion	varchar	20
		Nombre_Equipo	varchar	50
		Direccion_IP_PC	varchar	15
		ID_Usuario	int	4
		Estado_Sesion	bit	1
2	**tbl_GRUPOS_USUARIOS**			
		ID_Grupo	tinyint	1
		Descripcion	varchar	20
		Permisos_Default	varchar	4
		Observaiones	varchar	100
3	**tbl_MODULOS_SISTEMA**			
		ID_Modulo	tinyint	1
		Descripcion	varchar	20
		Observacion	varchar	100
4	**tbl_PERMISOS_USUARIOS**			
		ID_Usuario	int	4
		ID_Modulo	tinyint	1
		Permisos	varchar	4
5	**tbl_SEGURIDAD_CONTRASENIA**			
		Id_Usuario	int	4
		Longitud_Password	tinyint	1
		Vector_Decodificador	varchar	50

ANEXO 16: Diccionario de Datos del modulo de seguridad de "HEAD" - Segunda parte

6 **tbl_USUARIOS**

ID_Usuario	int	4
Cedula	varchar	14
Nombre1	varchar	25
Nombre2	varchar	25
Apellido1	varchar	25
Apellido2	varchar	25
Sexo	bit	1
Foto	Image	16
Correo_electronico	varchar	100
Estado_Usuario	varchar	20
Estado_Sesion	bit	1
Login	varchar	20
Passwrd	varchar	20
ID_Grupo	tinyint	1
Fecha_Registrado	smalldatetime	4