UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

"DESARROLLO DEL PORTAL WEB DEL DEPARTAMENTO DE SEGURIDAD EN CÓMPUTO DE LA FACULTAD DE INGENIERÍA"

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERA EN COMPUTACIÓN

P R E S E N T A

BEATRIZ LÓPEZ MARTÍNEZ

DIRECTOR DE TESIS:

ING. RAFAEL SANDOVAL VÁZQUEZ

México, D.F. 2008

Ciudad Universitaria

ÍNDICE

PREFACIO

Podemos entender la seguridad informática como una característica de cualquier sistema (informático o no) que nos indica que ese sistema está libre de peligro, daño o riesgo. Se entiende como peligro o daño todo aquello que pueda afectar su funcionamiento directo o los resultados que se obtienen del mismo. Cabe señalar que el concepto de seguridad en la informática es utópico porque no existe un sistema cien por ciento seguro. Para que un sistema se pueda definir como seguro debemos de dotar de tres características al mismo: integridad, confidencialidad y disponibilidad.

Dependiendo de las fuentes de amenazas, la seguridad puede dividirse en seguridad lógica y seguridad física. Recordemos que un incidente de seguridad informática es un evento que concreta una amenaza al explotar una vulnerabilidad y que afecta a una o más propiedades de la información.

De acuerdo a los crecientes riesgos, ha habido la necesidad de crear organismos, a nivel mundial, capaces de atender y dar respuesta a los distintos incidentes, tal es el caso de los equipos de Respuesta a Incidentes de Seguridad en Cómputo (RISC o CERT, por sus siglas en inglés, *Computer Emergency Response Team*).

El equipo de respuesta a incidentes de seguridad en cómputo de la Universidad Nacional Autónoma de México (UNAM), nació en 1988 como respuesta al primer 'gusano' de la Internet. Debido al impacto económico y la potencialidad de propagación de estos gusanos y otras formas de ataque virtual, el gobierno de Estados Unidos decidió crear o apoyar CERT's en el mundo con el fin de estudiar, catalogar y combatir los problemas y riesgos de la Internet. El esquema ha crecido de tal forma que hoy existe una red de 170 CERT's en el mundo.

El hecho de que en México exista únicamente un CERT ha desembocado en la preocupación de crear otro u otros organismos para que apoye los sistemas informáticos financieros y uno más para los gubernamentales, así como de difundir masivamente las políticas de seguridad que emanen de sus estudios.

Derivado de dichas necesidades surge el Departamento de Seguridad en Cómputo de la Facultad de Ingeniería, el 2 de Septiembre de 2004, el secretario general de la Facultad de Ingeniería, el ingeniero Gonzalo López de Haro, instaló oficialmente el departamento, el cual quedó adscrito a la Unidad de Servicios de Cómputo Académico (UNICA); y designó como su responsable al ingeniero Rafael Sandoval Vázquez.

AGRADECIMIENTOS

Dedico esta tesis a mis abuelos, en especial a la memoria de Roberta Cruz, gracias por tu compañía en el desarrollo de este trabajo.

A mis padres, Guillermina Martínez por su ejemplo de fortaleza, por su amor, dedicación y protección presente en cada etapa de mi vida. A Ricardo S. López por su valioso ejemplo de persistencia y dedicación, de honestidad, respeto y por toda la formación integral que siempre me brindo. A ambos gracias por su invaluable amor y apoyo.

A mis hermanos porque ahora compartimos la dicha de ver reflejado el esfuerzo y comprobamos la importancia de la educación, inculcada por nuestros padres. A mis sobrinos por ser el impulso de querer mejorar el presente.

Le agradezco mucho al Ing. Rafael Sandoval Vázquez por todo el apoyo brindado en el desarrollo y conclusión de este proyecto, por su paciencia y asesoría, pero sobre todo por su admirable ejemplo a seguir en el ámbito profesional, académico y personal.

A la Universidad Nacional Autónoma de México y a la Facultad de Ingeniería por las oportunidades ofrecidas en general.

A toda UNICA también gracias por permitirme formar parte de sus filas, y por el muy significativo conocimiento enseñado.

A todos los que directa e indirectamente me apoyaron, gracias.

INTRODUCCIÓN

La paradoja que enfrenta al hombre con sus propias limitaciones y posibilidades está representada en el avance de la tecnología informática, probablemente más que en ningún otro aspecto del desarrollo cultural. Pareciera que este creciente avance conlleva fragilidad y que las medidas de seguridad son el acicate para los hackers y los errores humanos. De tal suerte que existen mil formas de perder el hilo en la red.

La inseguridad informática crece con el avance tecnológico. Esta hipótesis de apariencia paradójica es resultado de la capacidad humana para hacer de algo muy bueno, algo muy malo. Claro, no todo proviene de la mente criminal, también del descuido, del tan humano error, de la falta de previsión y de la baja capacitación, entre otros factores.

La alta velocidad con la cual se procesan las transacciones, los sistemas de administración de las bases de datos, las redes de telecomunicaciones globales, el procesamiento distribuido de datos, la comunicación sobre Internet, y muchos otros factores, han causado que en toda organización, sin excepción alguna, la información y los datos en los cuales se apoya se tornen cada día más y más importantes. Por lo que las estrategias de administración, las políticas de seguridad, la segregación de las funciones, el impacto de los fallos, los accesos no autorizados, la revelación de la información, la continuidad del normal procesamiento de los datos, la adecuación de los sistemas de información, y otros aspectos que surgen de la aplicación de innovadoras tecnologías, han pasado a tener un impacto mucho mayor dentro de las instituciones que el de hace unos años; de ahí la necesidad de contar con un adecuado marco de control.

Por lo expuesto, para muchas organizaciones, la información y la tecnología que las soporta, han pasado a representar a ser activos más valiosos. Bajo esta situación, éstas han comenzado a reconocer los beneficios potenciales que las herramientas tecnológicas les pueden proporcionar. Pero sin embargo, también han comprendido la importancia de conocer y administrar los riesgos asociados con la implementación de las nuevas tecnologías.

Esto nos enseña que los cambios en la forma de uso de la tecnología en la sociedad han abierto la puerta para nuevas oportunidades de intrusión. El área de seguridad en cómputo está consciente de esto, por lo que ha desarrollado procedimientos para actuar ante un incidente, el cual representa un primer paso para el plan de seguridad informática con el que contará la Facultad de Ingeniería.

De la misma forma resulta de vital importancia el desarrollo de herramientas que a la vanguardia de la tecnología faciliten y ayuden a la implementación de dichos procedimientos. Es aquí donde juega un papel fundamental la ingeniería de software como instrumento a los ingenieros en cómputo para producir sistemas informáticos acorde a las necesidades del cliente.

El software se ha convertido en el elemento clave de la evolución de los sistemas y productos informáticos. En los pasados 50 años, ha pasado de ser una resolución de problemas especializada y una herramienta de análisis de información, a ser una industria por sí misma. Se compone de programas, datos y documentos. Cada uno de estos elementos compone una configuración que se crea como parte de la ingeniería del software.

La ingeniería del software intenta proporcionar un marco de trabajo para construir software con mayor calidad.

El presente trabajo recopila estos elementos de la ingeniería del software para la construcción del portal Web del Departamento de Seguridad en Cómputo, el cual representa una herramienta en línea para la atención de incidentes, fomento a la cultura de la prevención, y un acercamiento del departamento a la comunidad en general de la facultad.

Así, el capítulo 1 denominado "marco histórico", analiza el impacto y transición evolutiva del Departamento de Seguridad en Cómputo de la Facultad de Ingeniería, exponiendo las ventajas y necesidades de su creación. También se describen la misión, visión, constitución y los servicios que ofrece el departamento; además de los resultados obtenidos posterior a su creación.

El capítulo 2, "marco teórico" define el marco de referencia para la formulación correcta del sistema a desarrollar, a través de la ingeniería del software, la cual es una guía en el análisis, desarrollo, construcción e implementación del sistema. Pretende además identificar las herramientas propias y auxiliares de la ingeniería del software, en cada una de las fases del trabajo a desarrollar para su correcta formulación y satisfactoria conclusión.

En el capítulo 3 llamado "análisis y diseño del sistema" describe el estudio de la fase de análisis y diseño, apoyándose de la metodología *racional unified process*. Se establecen y definen las herramientas de desarrollo a utilizar (sistema operativo, tipo servidor y lenguaje de programación), se registran los requerimientos del cliente identificando actores y roles como base para la creación del diseño de software, y posteriormente se especifica el modelado de datos y diseño del sistema, para finalmente disponer de la arquitectura óptima del sistema para proceder a la construcción final.

"Desarrollo del sistema" es el capítulo 4 donde se especifica a nivel conceptual la arquitectura de cada interfaz involucrada en las necesidades del cliente a partir de los requerimientos recopilados.

El capítulo 5 denominado "requerimientos de implementación" obtiene una visión específica de los requerimientos de hardware y software necesarios para una implementación óptima del sistema. Asimismo se realiza una estimación del costo total del sistema hasta la implementación.

Finalmente se establecen las conclusiones obtenidas y se hacen las recomendaciones pertinentes.

CAPÍTULO 1

MARCO HISTÓRICO

1.1 Objetivo

Analizar el impacto y transición evolutiva del Departamento de Seguridad en Cómputo de la Facultad de Ingeniería (DSC-FI), desde el planteamiento del proyecto hasta su actual situación, englobando con ello cada una de los beneficios y ventajas de su creación.

También se describen la misión, visión, constitución y los servicios que ofrece el departamento; además de los resultados obtenidos posterior a su creación.

1.2 Historia del Departamento de Seguridad en Cómputo

El nacimiento de dicho proyecto nace como inquietud de dos jóvenes que aún en ciernes en el área de la seguridad en cómputo, en marzo de 2003, y siendo integrantes del Departamento de Redes y Operación de Servidores (DROS) adscrito a la Unidad de Servicios de Cómputo Académico (UNICA) de la Facultad de Ingeniería de la UNAM; a cargo del ingeniero Noé Cruz Marín; el ingeniero Rafael Sandoval Vázquez y el ingeniero Luis Fernando Fuentes Serrano asistieron a las líneas de capacitación de seguridad en cómputo organizadas por el Departamento de Seguridad en Cómputo de la Dirección General de Servicios de Cómputo Académico (DGSCA) de la UNAM.

El ingeniero Sandoval Vázquez participó como becario en la línea de especialización: técnicas de intrusión, ataques y manejo de incidentes; mientras que el ingeniero Fuentes Serrano de igual forma participó como becario en la línea de especialización: administración y seguridad en UNIX.

Al regreso de estas líneas de especialización los ingenieros Sandoval y, motivados y consientes de la necesidad de involucrar a UNICA y a la Facultad de Ingeniería en los temas relativos a la seguridad informática sostuvieron varias conversaciones con el ingeniero Noé Cruz Marín, jefe del DROS, planteándole la necesidad de conformar un área que se encargará exclusivamente de la seguridad de redes y sistemas de dicha unidad.

Hasta este entonces en la Facultad de Ingeniería no se habían planteado interrogantes relativas a la seguridad informática, y las acciones de seguridad que se realizaban eran meramente reactivas, es decir hasta tener los problemas, regularmente catastróficos se buscaban las soluciones.

El ingeniero Cruz, convencido de la necesidad de contar con un equipo encargado de atender los incidentes de seguridad, formalizó en el mes de agosto de 2003 la creación del área de seguridad en cómputo, lo cual motivo grandes cambios hacia el interior del DROS y de UNICA, designando como responsable al ingeniero Sandoval y como su colaborador al ingeniero Fuentes.

Es de esta manera que en la Facultad de Ingeniería aparecía la primera área enfocada única y exclusivamente a la atención de problemas de seguridad en cómputo.

El ingeniero Sandoval y el ingeniero Cruz motivados por los buenos resultados de esta área a pocos meses de su creación y después de iniciado el proyecto de seguridad perimetral en cómputo para UNICA, decidieron iniciar las gestiones para crear un centro único y especializado en la atención de incidentes de seguridad en cómputo para la Facultad de Ingeniería. De esta forma se dio el acercamiento con el jefe de UNICA, el ingeniero Enrique Barranco Vite.

Durante el primer semestre de 2003, el área de seguridad en cómputo formalizaba sus actividades y procedimientos de servicios y de atención a incidentes, era preparada la documentación que sustentaría la creación del departamento y a la vez crecían los recursos humanos y recursos de cómputo, para esta área.

Todos los integrantes fueron previamente seleccionados del plan de becarios en cómputo de UNICA, el cual prepara de manera integral en materia de cómputo a estudiantes de la Facultad de Ingeniería, y del cual solo los mejores son seleccionados.

Los ingenieros Cruz, Barranco y Sandoval hacía el segundo tercio del año de 2003 buscaron afanosamente la creación del DSC-FI; de esta manera el acercamiento con el ingeniero Gonzalo López de Haro, secretario general de la Facultad de Ingeniería, se dio.

Para este tiempo el área de seguridad en cómputo del DROS requería de un proyecto que motivará la formalización del departamento es así que el ingeniero Sandoval propuso el sistema de seguridad en cómputo perimetral para la Facultad de Ingeniería, el cual consistía a grandes rasgos en fortalecer la seguridad en cómputo de la facultad colocando dispositivos de filtrado y sistemas detectores de intrusos robustos y de bajo costo en las acometidas principales de cada uno de los segmentos de red de la facultad, manteniéndolo con un monitoreo estricto y continuo, por parte del área de seguridad en cómputo, que permitiera detectar oportunamente los problemas de seguridad y resolverlos antes de que estos trascendieran.

Aunado a esto, proponía el modelo de seguridad en cómputo para la Facultad de Ingeniería, cuyos objetivos principales eran el de fortalecer y difundir las políticas de seguridad vigentes, disminuir la cantidad y gravedad de los problemas de seguridad y difundir la cultura de la seguridad en cómputo a la comunidad en general.

El 2 de Septiembre de 2004, el secretario general de la Facultad de Ingeniería, el ingeniero Gonzalo López de Haro, instaló oficialmente el Departamento de Seguridad en Cómputo de la Facultad de Ingeniería, el cual quedó adscrito a UNICA; y designó como su responsable al ingeniero Rafael Sandoval Vázquez.

El camino desde la creación del área de seguridad en cómputo hasta estos días no ha sido fácil, principalmente por la falta de espacios de difusión, así como la dificultad de concientizar algunas áreas en participar e involucrarse seriamente con la seguridad en cómputo.

Fueron muchas las pláticas, discusiones y adversidades que la gente del DSC-FI a tenido que sobre llevar; sin embargo, día a día se fortalece y el reconocimiento por su trabajo hacia dentro y fuera de la facultad crece.

1.2.1 Departamento de Seguridad en Cómputo

El surgimiento del Departamento de Seguridad en Cómputo es respuesta a la necesidad de dar atención inmediata a los crecientes problemas que genera la evolución de los sistemas computacionales y su interacción, específicamente a través de Internet. Por ello se requiere disponer de una herramienta en línea, portal Web, que permita controlar y administrar las distintas tareas del departamento y generar con ello información actualizada para prevenir y solucionar los distintos problemas relacionados a la seguridad en cómputo de los usuarios de la Facultad de Ingeniería.

Para ello se creará un portal Web, como parte de la innovación de sistemas y procesos de la ingeniería, que despliegue todas estas opciones en línea y con ello se atiendan de manera más rápida y eficiente dichos problemas, así como mostrar información y herramientas que ayuden a su prevención.

1.2.2 Misión

Proveer un único y confiable punto de contacto para los administradores de redes y servidores de la Facultad de Ingeniería de la UNAM para tratar los problemas de seguridad e incidentes, y a su vez ser el centro más confiable para la recolección y diseminación de información relacionada con las amenazas sobre las redes computacionales, vulnerabilidades y respuesta a incidentes de la Facultad de Ingeniería.

1.2.3 Constitución

Es el grupo de usuarios, servidores, redes y organizaciones las cuales tendrán el beneficio de la misión del Departamento de Seguridad en Cómputo, esto quiere decir que estas entidades o grupos serán los beneficiados por el trabajo que desempeñe el DSC-FI.

Actualmente la Facultad de Ingeniería cuenta con cuatro segmentos de red en el campus de ciudad universitaria y uno en la división de educación continua y a distancia (Palacio de Minería), cada una con su propia acometida de red.

1.2.4 Servicios

Los servicios están listados en orden decreciente conforme a su importancia:

1. Amenazas e incidentes de seguridad en cómputo

- Amenaza de la seguridad física de seres humanos.
- Ataques a *root*, al sistema o al manejo de la información de cualquier elemento del que se compone la red y que involucre sistemas críticos, multiusuario y en producción.
- Ataque a la confidencialidad de la información, de las cuentas de usuario, del software, de los sistemas y de la administración en sistemas multiusuario, críticos y en producción.
- Ataques de denegación de servicios o incidentes relacionados.
- Ataques de cualquier equipo definidos en la constitución hacia equipos externos.
- Ataques a gran escala de cualquiera de este tipo: ataques de *sniffeo*, ingeniería social, *crackeo* de contraseñas, etc.
- Cuentas individuales comprometidas en sistemas críticos, en producción y sistemas multiusuario.
- Amenazas, hostigamiento y cualquier otro tipo de ofensas que involucren a cuentas de usuario individuales.
- Computadoras personales comprometidas.
- Violación a las políticas de cómputo vigentes.

- Denegación de servicio a cuentas individuales.

2. Servicios comunes del Departamento de Seguridad en Cómputo

- Manejo de incidentes
  - Proporcionar mecanismos de reporte de incidentes.
  - Entender la amenaza, naturaleza y alcance de los ataques.
  - Identificar nuevos tipos de métodos de ataque.
  - Proporcionar soporte técnico para respuesta de incidentes de seguridad.
  - Facilitar la comunicación entre los sitios y equipos de respuesta.
- Alertas y anuncios
  - Analizar y desarrollar avisos y anuncios.
  - Retransmitir avisos de otros equipos.
- Manejo de vulnerabilidades
  - Analizar y verificar el problema.
  - Coordinarse con otros equipos de respuesta a incidentes y otros expertos de confianza.
  - Mantener la información de las vulnerabilidades en un lugar seguro.
- Sistemas detectores de intrusos
  - Revisar las alertas de los sistemas de detección de intrusiones.
  - Actualizar y mantener las firmas de los sistemas detectores de intrusos.
  - Revisar y monitorear el ambiente de red existente para establecer una línea base de actividad de la red, con objeto de tener un punto de comparación en contra de potenciales anomalías.
  - Mantener registros para usar durante la investigación y recuperación de actividades.
- Educación y capacitación
  - Crear una cultura de seguridad con cursos de capacitación.
  - Generar documentación que permita a todos los usuarios estar enterados de los avances y actividades actuales de la seguridad en cómputo.
  - Formación de nuevos elementos para la seguridad en los sistemas de cómputo.
  - Capacitación de los elementos del equipo de forma organizada y constante.
- Auditorias
  - Para determinar que tan seguro es un sistema actualmente o que necesita para incrementar su seguridad.
  - Para garantizar que el sistema cumple con los estándares y políticas correspondientes.
  - Para realizar el seguimiento a un incidente de seguridad para determinar qué alteraciones ocurrieron, como ingresaron al sistema, etc.
- Colaboración y Coordinación
  - Colaboración y coordinación con otros equipos de respuesta a incidentes en cómputo y áreas de seguridad.

Todos los servicios y sus prioridades que el DSC-FI a descrito en esta sección se reevaluarán continuamente y conforme se avance en la investigación y desarrollo de nuevas tecnologías de seguridad así como a los nuevos métodos de ataque por la red.

1.2.5 Situación actual

A más de tres años de la creación del DSC-FI, los resultados han sido satisfactorios, los incidentes han disminuido drásticamente, un 80% en las áreas participantes, los esquemas de seguridad planteados operan correctamente y han permitido aumentar la disponibilidad de los servicios y el mantenimiento y actualización es permanente, la capacitación ha ido en aumento y cada vez más personal se ve beneficiado por esto, las políticas de seguridad en cómputo de la facultad han sido fortalecidas ya que ahora un departamento se encarga de hacerlas cumplir, se ha dado asesorías

inter-institucionales, se ha participado en congresos de seguridad, se ha promovido la cultura de la seguridad informática, se ha participado protagónicamente en la revisión del plan de estudios de la carrera de Ingeniería en Cómputo con la creación de tres nuevas materias enfocadas a la seguridad, la infraestructura en equipo de computo del DSC-FI se ha fortalecido, así como su motivación y espíritu de servicio para con su institución.

Sin embargo faltan muchos objetivos por alcanzar, y tanto autoridades como los responsables directos de los sistemas de seguridad en cómputo tendrán que aportar lo necesario para conseguirlos, como son recursos económicos, recursos humanos, de infraestructura, de capacitación y de retroalimentación como corresponda a cada uno de ellos para consolidar a la facultad como pionera y exitosa en el ámbito de la seguridad informática no sólo dentro de la UNAM si no a nivel nacional.

1.3 Unidad de Servicios de Cómputo Académico

Historia UNICA

Surge en el año de 1994 cuando se decide seccionar el centro de cálculo de acuerdo a sus objetivos y funciones, con la finalidad de proporcionar una mayor eficiencia en el desempeño del personal. Con base a esto se crean dos unidades para desempeñar el trabajo que realizaba el centro de cálculo. UNICA y la unidad de Servicios de Cálculo Administrativo (USECAD), son las dos unidades creadas para llevar a cabo las tareas Académicas y Administrativas de la Facultad de Ingeniería.

UNICA se esfuerza siempre para estar a la vanguardia de la tecnología en el área de cómputo. Las funciones que desempeña son:

- Mantener el liderazgo en cuanto a tópicos en cómputo.
- Continuar proporcionando recursos de cómputo de calidad a la comunidad de la facultad.
- Impulsar a nivel de la facultad la creación de una política de cómputo definida.
- Lograr la capacitación cada vez más completa y actualizada para la formación de recursos humanos.
- Aplicar todos los conocimientos y las herramientas de cómputo con los que cuenta la unidad para realizar las actividades de forma más eficiente y segura.

La excelencia de UNICA se debe a que una de las principales actividades de la unidad es brindar el mejor servicio a los usuarios para ayudar en su formación como futuros profesionistas. El hecho de ser parte de UNICA nos ha brindado una visión de la problemática que enfrenta el profesionista en el mundo real.

Política

Está basada en cumplir con los requerimientos de nuestros clientes en el área de cómputo teniendo como meta elevar la calidad de nuestros productos y servicios, para ello nos comprometemos en un proceso de mejora continua.

Misión

Proporcionar eficaz y eficientemente en el ámbito institucional, los servicios de cómputo y el apoyo en actividades relacionadas que coadyuven al proceso integral de formación académica en la Facultad de Ingeniería.

Visión

La proyección al año 2010 es continuar siendo una unidad líder en la prestación de servicios de cómputo de vanguardia a la Facultad de Ingeniería, al entorno universitario y a la sociedad en general, mediante los siguientes puntos:

- Contando con la organización, administración y los recursos adecuados.
- Siendo líderes en la formación, capacitación y difusión de la cultura informática.
- Manteniéndonos vigentes con las herramientas y convenios adecuados para el desarrollo y la investigación informática.
- Disponiendo de una infraestructura de red de cómputo moderna y tecnología de punta, brindando servicios de calidad y alta disponibilidad en tecnologías de la información y comunicación.
- Manteniendo los servicios y procesos de atención sistematizados y actuales en apoyo a los eventos de seguridad informática.
- Configurando la infraestructura adecuada y mecanismos para la actualización continúa del equipo de cómputo.

Objetivos

UNICA es la encargada de proporcionar a nivel institucional, los servicios de apoyo en cómputo que los alumnos de la Facultad de Ingeniería requieren para la realización y cumplimiento eficaz de sus tareas sustantivas.

Formar recursos humanos de calidad, tanto en el área de cómputo como en el desempeño de la vida profesional.

Apoyar a la secretaría general en las actividades que involucren institucionalmente a la Facultad de Ingeniería.

Estructura administrativa

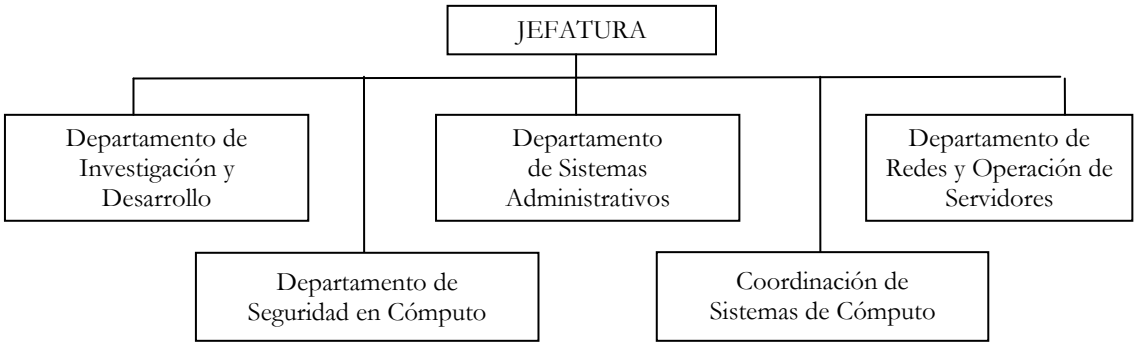


Figura 1.1 Estructura administrativa de UNICA.

CAPÍTULO 2

MARCO TEÓRICO

2.1 Objetivo

Definir el marco de referencia teórico para la formulación correcta del problema a desarrollar, el portal del DSC-FI, definiendo la rama de estudio que engloba, ingeniería del software, la cual representará una guía en el análisis, desarrollo, construcción e implementación de dicho sistema.

Se pretende además identificar las herramientas propias y auxiliares de la ingeniería del software, en cada una de las fases del trabajo a desarrollar para su correcta formulación y satisfactoria conclusión.

2.2 Ingeniería del software

El software de computadora se ha convertido en el *alma mater* de la computación, es quien conduce a la toma de decisiones comerciales, sirve de base para la investigación científica moderna y de resolución de problemas de ingeniería, es el factor clave que diferencia los productos y servicios modernos, está inmerso en sistemas de todo tipo: de transportes, médicos, de telecomunicaciones, militares, marítimos, procesos industriales, entretenimientos, productos de oficina, entre otros. El software es casi ineludible en un mundo moderno, a medida que se avanza en el siglo XXI, el software conduce a nuevos avances, desde la educación elemental hasta la ingeniería genética.

La ingeniería del software es una disciplina o área de la informática o ciencias de la computación, que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo. Hoy en día es cada vez más frecuente la consideración de la ingeniería del software como una nueva área de la ingeniería, y el ingeniero del software comienza a ser una profesión implantada en el mundo laboral internacional, con derechos, deberes y responsabilidades que cumplir, junto a una, ya, reconocida consideración social en el mundo empresarial .

El término *ingeniería de software* fue usado ocasionalmente al final de la década de los años 50 e inicios de los años 60. El término fue popularizado por la conferencia sobre ingeniería de software de la OTAN (Organización del Tratado del Atlántico Norte) en 1968 que tuvo lugar en Garmish, Alemania, y ha sido ampliamente utilizado desde entonces.

2.2.1 Definición de ingeniería del software

Para una definición más precisa de la ingeniería del software, se mencionan las siguientes de autores acreditados que comenzaron en su momento a utilizar dicho término, y finalmente considerando la definición de organismos internacionales profesionales.

"Ingeniería del software trata del establecimiento de los principios y métodos de la ingeniería a fin de obtener software rentable que sea fiable y trabaje eficientemente en máquinas reales" [1].

"Ingeniería del software es la aplicación práctica del conocimiento científico en el diseño y construcción de programas de computadora y la documentación asociada y requerida para desarrollar, operar (funcionar) y mantenerlos. Se conoce también como desarrollo de software o producción de software" [3].

"Ingeniería del software es el estudio de los principios y metodologías para desarrollo y mantenimiento de sistemas de software" [17].

"La aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software; es decir, la aplicación de ingeniería al software" [7].

Para el desarrollo de este trabajo, se considero la siguiente definición:

"Ingeniería del software es la aplicación práctica, disciplinada y cuantificable del conocimiento científico en el diseño y construcción de programas de computadora así como en la documentación asociada y requerida para desarrollar, operar (funcionar) y mantenerlos".

2.2.2 El proceso del software

Proporciona un marco de trabajo para la tecnología de ingeniería del software. Representa el marco de trabajo de las tareas que se requieren para construir software de alta calidad, y es una tecnología multicapa, apoyada sobre un compromiso de organización de calidad.

En la siguiente figura 2.1 se observa que el fundamento de la ingeniería del software esta en el proceso. Dicho proceso es la capa encargada de unir a las capas de tecnología y que permite un desarrollo racional y oportuno de la ingeniería del software.

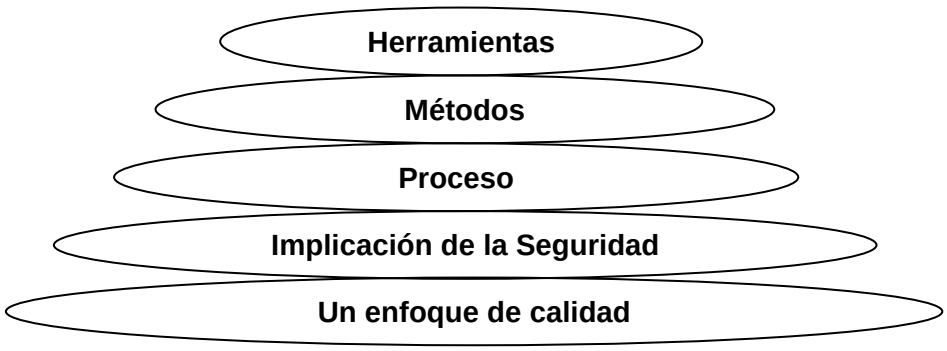


Figura 2.1 Capas de la ingeniería del software.

El proceso define un marco de trabajo para un conjunto de Áreas Clave de Proceso (ACPs) que se deben establecer para la tecnología de la ingeniería del software. Las ACPs forman la base del

control de gestión de proyectos del software y establecen el contexto en el que se aplican los métodos técnicos, se obtienen productos del trabajo (modelos, datos, informes), se asegura la calidad y el cambio se gestiona adecuadamente.

El proceso del software se puede caracterizar por un marco de trabajo común, definiendo un pequeño número de actividades del marco de trabajo de que son aplicables a todos los proyectos del software, con independencia de su tamaño o complejidad. Un número de conjuntos de tareas – cada uno es una colección de tareas de trabajo de ingeniería del software, hitos de proyectos, productos de trabajo y puntos de garantía de calidad- que permiten que las actividades del marco de trabajo se adapten a las características del proyecto del software y a los requisitos del equipo del proyecto. Finalmente, las actividades de protección – tales como garantía de calidad del software, gestión de configuración del software y medición- abarcan el modelo de procesos. Las actividades de protección son independientes de cualquier actividad del marco de trabajo y aparecen durante todo el proceso, la figura 2.2 ilustra este marco de trabajo del proceso del software.

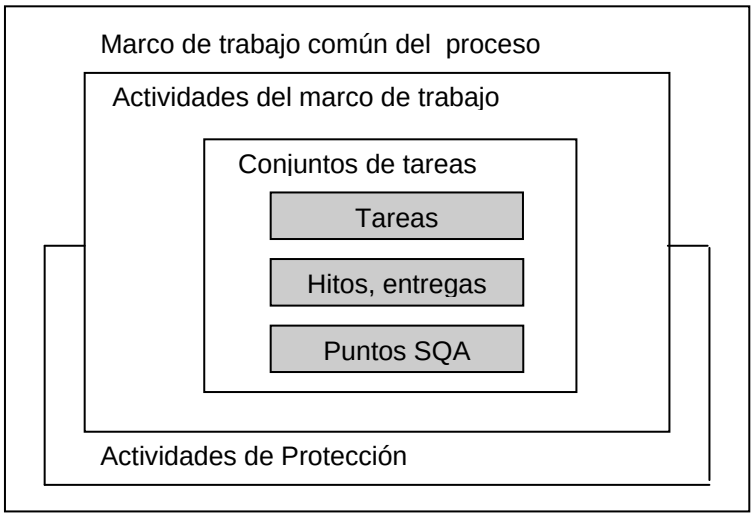


Figura 2.2 El proceso del software.

El Software Engineering Institute (SEI de la Carnegie Mellon University) ha desarrollado un modelo completo que se basa en un conjunto de funciones de ingeniería del software que deberían estar presentes conforme organizaciones alcanzan diferentes niveles de madurez del proceso. Para determinar el estado actual de madurez del proceso de una organización, el SEI utiliza un cuestionario de evaluación y un esquema de cinco grados.

El esquema de grados determina la conformidad con un Modelo de Capacidad de Madurez (MMC) que define las actividades clave que se requieren en los diferentes niveles de madurez del proceso.

Los niveles se definen en la siguiente tabla 2.1:

Nivel	Definición
1. Inicial	El proceso del software se caracteriza según el caso, y ocasionalmente incluso de forma caótica. Se definen pocos procesos, y el éxito depende del esfuerzo individual.
2. Repetible	Se establecen los procesos de gestión del proyecto para hacer seguimiento del costo, de la planificación y de la funcionalidad. Para repetir éxitos anteriores en proyectos con aplicaciones similares se aplica la disciplina necesaria para el proceso.
	El proceso del software de las actividades de gestión y de ingeniería se documenta, se estandariza y se integra dentro de un proceso de software de toda una organización. Todos los proyectos utilizan una

3. Definido	versión documentada y aprobada del proceso de la organización para el desarrollo y mantenimiento del software. En este nivel se incluyen todas las características definidas para el segundo nivel.
4. Gestionado	Se recopilan medidas detalladas del proceso del software y de la calidad del producto. Mediante la utilización de medidas detalladas, se comprenden y se controlan cuantitativamente tanto los productos como los procesos del software. En este nivel se incluyen todas las características definidas en el tercer nivel.
5. Optimización	Mediante una retroalimentación cuantitativa del proceso, ideas y tecnologías innovadoras se posibilita una mejora del proceso. En este nivel se incluyen todas las características del cuarto nivel.

Tabla. 2.1 Niveles de madurez del proceso.

Los cinco niveles definidos por el SEI se obtienen como consecuencia de evaluar las respuestas del cuestionario de evaluación basado en el MMC. Los resultados del cuestionario se refinan en un único grado numérico que proporciona una indicación de la madurez del proceso de una organización.

El SEI ha asociado las ACPs a cada uno de los niveles de madurez. Las ACPs describen esas funciones de la ingeniería del software que se deben presentar para satisfacer una buena práctica a un nivel en particular. Cada ACP se describe identificando estas características:

- Objetivos: representan los objetivos globales que deben alcanzar la ACP.
- Compromisos: requisitos (impuestos en la organización) que se deben cumplir para lograr los objetivos y que proporcionan una prueba del intento por ajustarse a los mismos.
- Capacidades: aquellos elementos que deben encontrarse (organizacional y técnicamente) para permitir que la organización cumpla los objetivos.
- Actividades: las tareas específicas que se requieren para lograr la función ACP.
- Métodos para supervisar la implementación: la manera en que las actividades son supervisadas conforme se aplican.
- Métodos para verificar la implementación: la forma en que se pueden verificar la práctica adecuada para la ACP.

Se definen 18 ACPs en el modelo de madurez y se distribuyen en niveles diferentes del proceso. De acuerdo a cada nivel de proceso, se agrupan en la tabla 2.2 como sigue:

Nivel	ACPs
Repetible	• Gestión de configuración del software. • Garantía de calidad del software. • Gestión de subcontratación del software. • Seguimiento y supervisión del proyecto de software. • Planificación del proyecto de software. • Gestión de requisitos.
Definido	• Revisiones periódicas. • Coordinación entre grupos. • Ingeniería de productos de software. • Gestión de integración del software. • Programa de formación. • Definición del proceso de la organización.

	• Enfoque del proceso de la organización.
Gestionado	• Gestión de la calidad del software. • Gestión cuantitativa del proceso.
Optimización	• Gestión de cambios del proceso. • Prevención de defectos.

Tabla. 2.2 Áreas claves del proceso.

Cada una de las ACPs se define con un conjunto de 'prácticas clave' que contribuyen a cumplir estos objetivos. Las prácticas clave son normas, procedimientos y actividades que deben ocurrir antes de que se haya instituido completamente un área clave de proceso. El SEI define a los indicadores clave como "aquellas prácticas clave o componentes de prácticas clave que ofrecen una visión mejor para lograr los objetivos de un área calve de proceso". Las cuestiones de valoración se diseñan para averiguar la existencia (o falta) de un indicador clave.

2.2.3 Modelos de proceso del software

Al enfrentar problemas reales de cualquier índole, se debe pasar por una serie de fases que contribuyan secuencialmente a la solución del mismo.

En la ingeniería del software sucede lo mismo, se debe definir una estrategia de desarrollo que acompañe al proceso, métodos, capas de herramientas y fases descritas en el proceso del software. Dicha estrategia se conoce como "modelo de proceso o paradigma de ingeniería del software", en la cual se selecciona un modelo de proceso según la naturaleza del proyecto y de la aplicación, los métodos y herramientas a utilizar, así como los controles y entregas requeridos.

De acuerdo con L.B.S. Raccoon, creador del modelo referido al "caos y el ciclo de vida caótico en el desarrollo de software", describe en un documento sobre la naturaleza del proceso del software, que todo el desarrollo del software se puede caracterizar como un bucle de resolución de problemas (ver tabla 2.3) en el que se encuentran cuatro etapas distintas:

- Estado actual
- Definición del problema
- Desarrollo técnico
- Integración de soluciones

Paradigma de la Ingeniería del Software	
Estado actual	Representa el estado actual del suceso(s).
Definición del problema	Identifica el problema específico a resolverse.
Desarrollo técnico	Resuelve el problema a través de la aplicación de alguna tecnología.
Integración de soluciones	Ofrece los resultados (documentos, programas, datos) a los que solicitan la solución en primer lugar.

Tabla. 2.3 Descripción del bucle de resolución de problemas de Raccoon.

Las interferencias generadas entre las etapas se muestran en las figuras 2.3 y 2.4, las cuales representan una situación que sugiere que independientemente del modelo de proceso que se

seleccione para un proyecto de software, todas las etapas coexisten simultáneamente en algún nivel de detalle. Y dada la naturaleza recursiva descrita, las cuatro etapas se aplican igualmente al análisis de una aplicación completa y a la generación de un pequeño segmento de código.

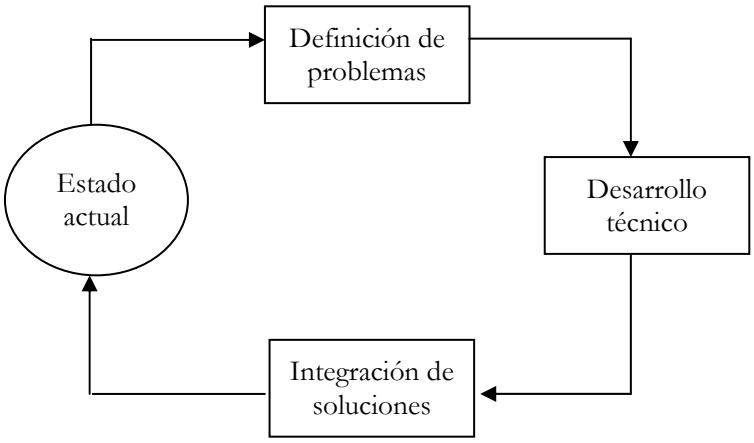


Figura 2.3 Fases de un bucle de resolución de problemas.

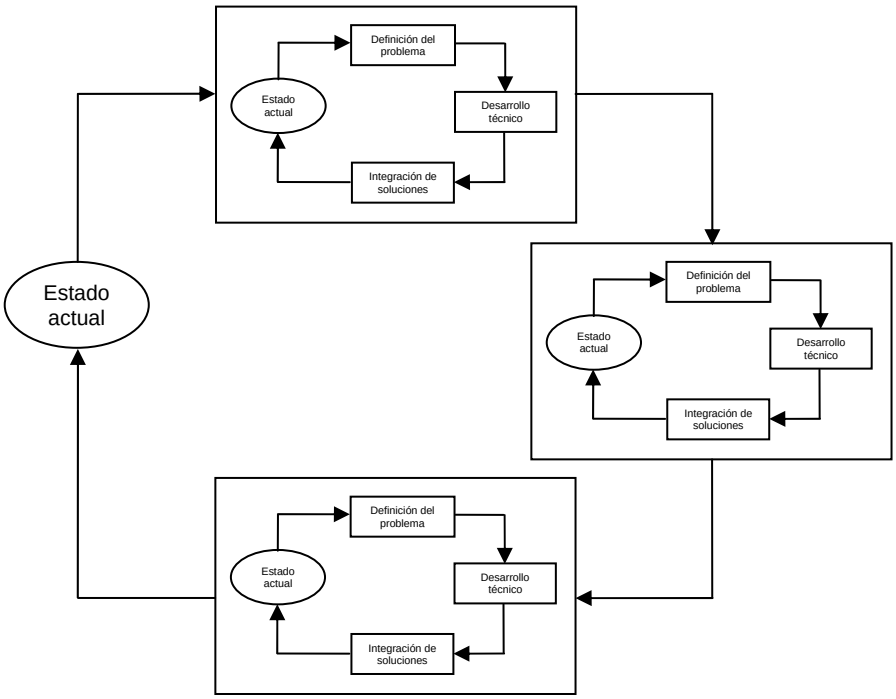


Figura 2.4 Fases dentro de las fases del bucle de resolución de problemas.

Raccoon, sugiere en su modelo del caos el desarrollo del software como una extensión desde el usuario hasta el desarrollador y la tecnología. Durante el progreso del trabajo hacia un sistema completo, las etapas descritas antes, se aplican recursivamente a las necesidades del usuario y a la especificación técnica del desarrollador del software.

Los modelos de procesos para la ingeniería del software, representan un intento de ordenar una actividad inherentemente caótica, y su importancia radica en la forma en que ayuden al control y a la coordinación de un proyecto de software real.

Modelo lineal secuencial

Llamado también "ciclo de vida básico" o "modelo en cascada", este modelo sugiere un enfoque sistemático, secuencial, para el desarrollo del software que comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento.

La figura 2.5 representa el modelo lineal secuencial para la ingeniería de software, modelado según el ciclo de ingeniería convencional.

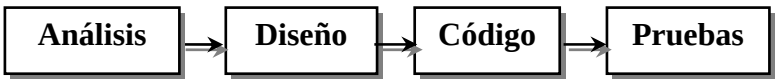


Figura 2.5 Modelo secuencial.

Comprende las siguientes actividades:

Ingeniería y modelado de sistemas/información.- El inicio del trabajo esta basado en establecer los requisitos de todos los elementos del sistema y asignando al software algún subgrupo de estos requisitos.

Análisis de los requisitos del software.- El analista del software debe comprender el dominio de información de software, así como la función requerida, comportamiento, rendimiento e interconexión.

Diseño.- Se centra en cuatro atributos distintos de programa: estructura de datos, arquitectura de software, representaciones de interfaz y detalle procedimental (algoritmo). El proceso del diseño traduce requisitos en una representación del software donde se pueda evaluar su calidad antes de que comience la codificación.

Generación de código.- El diseño se debe traducir en una forma legible por la máquina. Al realizar el diseño de una forma detallada, la generación de código se realiza mecánicamente.

Pruebas.- Se asegura que todas las sentencias se han comprobado, se detectan errores y se asegura que la entrada definida produce resultados reales de acuerdo con los resultados requeridos.

Mantenimiento.- Se producirán cambios al encontrarse errores, para que el software se adapte a los cambios de su entorno externo, o porque el cliente requiere mejoras funcionales o de rendimiento.

Este modelo proporciona una plantilla en la que se encuentran métodos para análisis, diseño, codificación, pruebas y mantenimiento.

Modelo de construcción de prototipos

Suele ayudar en la identificación de los requisitos detallados de entrada, proceso o salida para el software, con el propósito de asegurar la eficacia de un algoritmo, la capacidad de adaptación de un sistema operativo, o la forma en que debería tomarse la interacción hombre-máquina, observe la figura 2.6.

Comienza con la recolección de requisitos. Dando paso a un "diseño rápido", que se centra en una representación de esos aspectos del software que serán visibles para el usuario/cliente. El diseño rápido lleva a la construcción de un prototipo. El prototipo lo evalúa el cliente/ usuario y se utiliza para refinar los requisitos del software a desarrollar. La iteración ocurre cuando el prototipo se pone

a punto para satisfacer las necesidades del cliente, permitiendo al mismo tiempo que el desarrollador comprenda mejor lo que se necesita hacer.

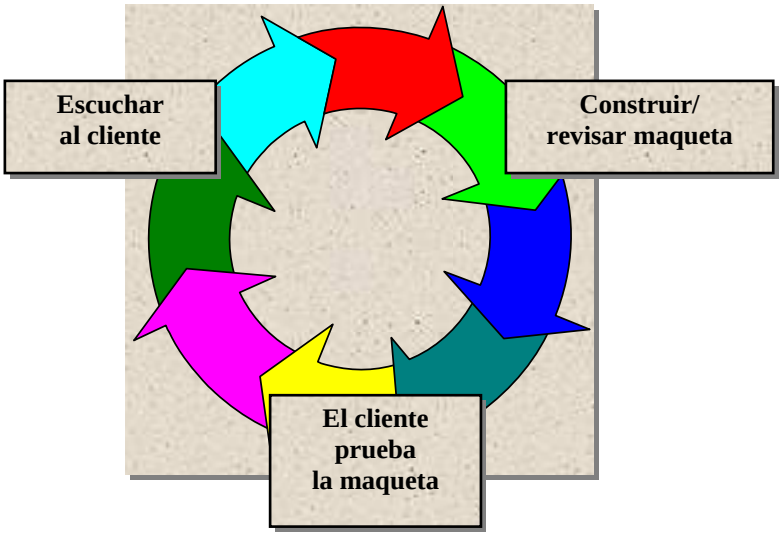


Figura 2.6 El paradigma de construcción de prototipos.

Sin embargo, la construcción de prototipos también puede ser problemática por las siguientes razones:

1. La gestión de desarrollo del software es muy lenta.
2. Se puede utilizar un sistema operativo o lenguaje de programación inadecuado, lo que hace surgir problemas y esto se resiente en la calidad.

La construcción de prototipos puede ser un paradigma efectivo para la ingeniería del software, aunque pueden surgir algunos problemas. La clave es definir las reglas del juego al comienzo; es decir, el cliente y el desarrollador se deben poner de acuerdo en que el prototipo se construya para servir como un mecanismo de definición de requisitos.

Modelo de desarrollo rápido de aplicaciones

DRA por sus siglás, es un modelo de proceso del desarrollo del software lineal secuencial que enfatiza un ciclo de desarrollo extremadamente corto. Es una adaptación a "alta velocidad" del modelo lineal secuencial en el que se logra el desarrollo rápido utilizando una construcción basada en componentes. Comprende las siguientes fases mostradas en la figura 2.7:

Modelado de Gestión.- El flujo de información entre las funciones de gestión debe responder a las siguientes preguntas: ¿qué información conduce el proceso de gestión?, ¿quién genera y a dónde va la información?, ¿quién la procesa?.

Modelado de datos.- El flujo de información se refina como un conjunto de objetos de datos necesarios para apoyar la empresa. Se definen las características (llamadas atributos) de cada uno de los objetos y las relaciones entre estos objetos.

Modelado del proceso.- Los objetos de datos definidos en la fase de modelado de datos quedan transformados para lograr el flujo de información necesario para implementar una función de gestión. Las descripciones del proceso se crean para añadir, modificar, suprimir, o recuperar un objeto de datos.

*Generación de aplicaciones.- U*tilización de técnicas de cuarta generación, tratadas en los siguientes párrafos. El proceso DRA trabaja para volver a utilizar componentes de programas ya existentes o a crear componentes reutilizables (cuando sea necesario). Utilizando herramientas para facilitar la construcción del software.

Pruebas y entrega.- La reutilización, permite comprobar muchos de los componentes de los programas. Esto reduce el tiempo de pruebas. Sin embargo, se deben probar todos los componentes nuevos y se deben ejercitar todas las interfaces a fondo.

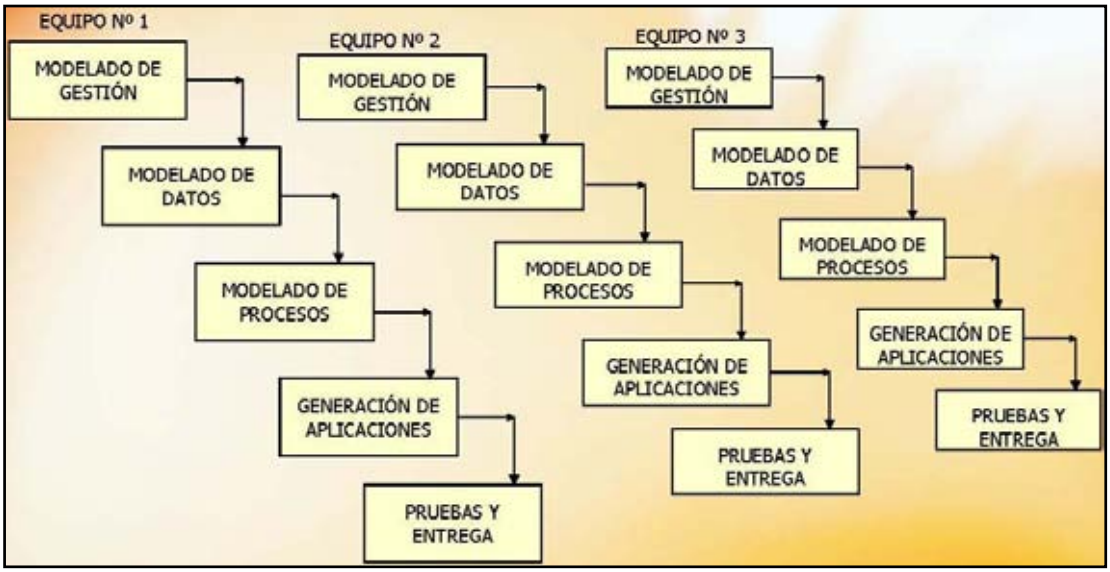


Figura 2.7 Modelo DRA.

Modelos evolutivos

De acuerdo con Gilb T., precursor del desarrollo evolutivo, menciona en su libro "Principles of Software Engineering Management", que el software al igual que todos los sistemas complejos, evoluciona con el tiempo.

Los modelos evolutivos son iterativos, se caracterizan por la forma en que permiten a los ingenieros del software desarrollar versiones cada vez más completas del software.

Modelo incremental

Combina elementos del modelo lineal secuencial (aplicados repetidamente) con la filosofía interactiva de construcción de prototipos. Como se ilustra en la figura. 2.8, el modelo incremental aplica secuencias lineales de forma escalonada mientras progresa el tiempo en el calendario. Cada secuencia lineal produce un "incremento" del software. Y se debe considerar que el flujo del proceso de cualquier incremento puede incorporar el paradigma de construcción de prototipos.

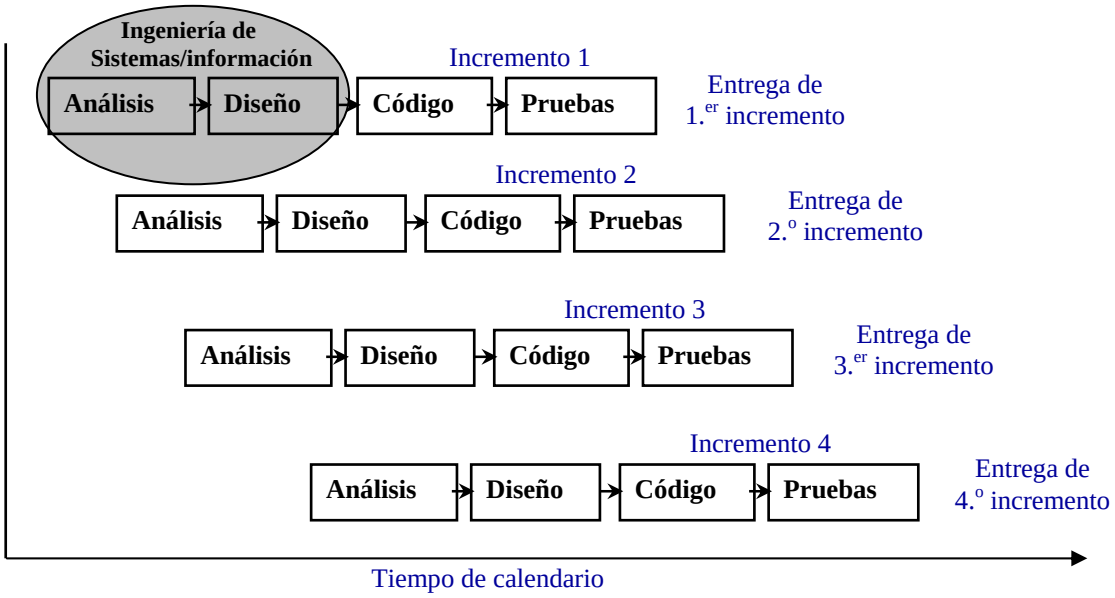


Figura 2.8 El modelo incremental.

Es decir, el modelo incremental entrega el software en partes pequeñas, pero utilizables, llamadas "incrementos", y en general, cada incremento se construye sobre aquél que ya ha sido entregado.

El modelo de proceso incremental, es iterativo por naturaleza y se centra en la entrega de un producto operacional con cada incremento. Los primeros incrementos son versiones "incompletas" del producto final, pero proporcionan al usuario la funcionalidad que precisa y también una plataforma para la evaluación.

Modelo espiral

Modelo de proceso de software evolutivo que conjuga la naturaleza iterativa de construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial. Proporciona el potencial para el desarrollo rápido de versiones incrementales. Durante las primeras iteraciones, la versión incremental podría ser un modelo en papel o un prototipo. Durante las últimas iteraciones, se producen versiones cada vez más completas del sistema diseñado.

Se divide en un número de actividades de marco de trabajo, también llamadas *regiones de tareas*. La figura 2.9, representa un modelo en espiral que contiene seis regiones de tareas: comunicación con el cliente, planificación, análisis de riesgos, ingeniería, construcción y acción, evaluación del cliente

Representa un enfoque realista del desarrollo de sistemas de software a gran escala. Como el software evoluciona, a medida que progresa el proceso, el desarrollador y el cliente comprenden y reaccionan mejor ante riesgos en cada uno de los niveles evolutivos.

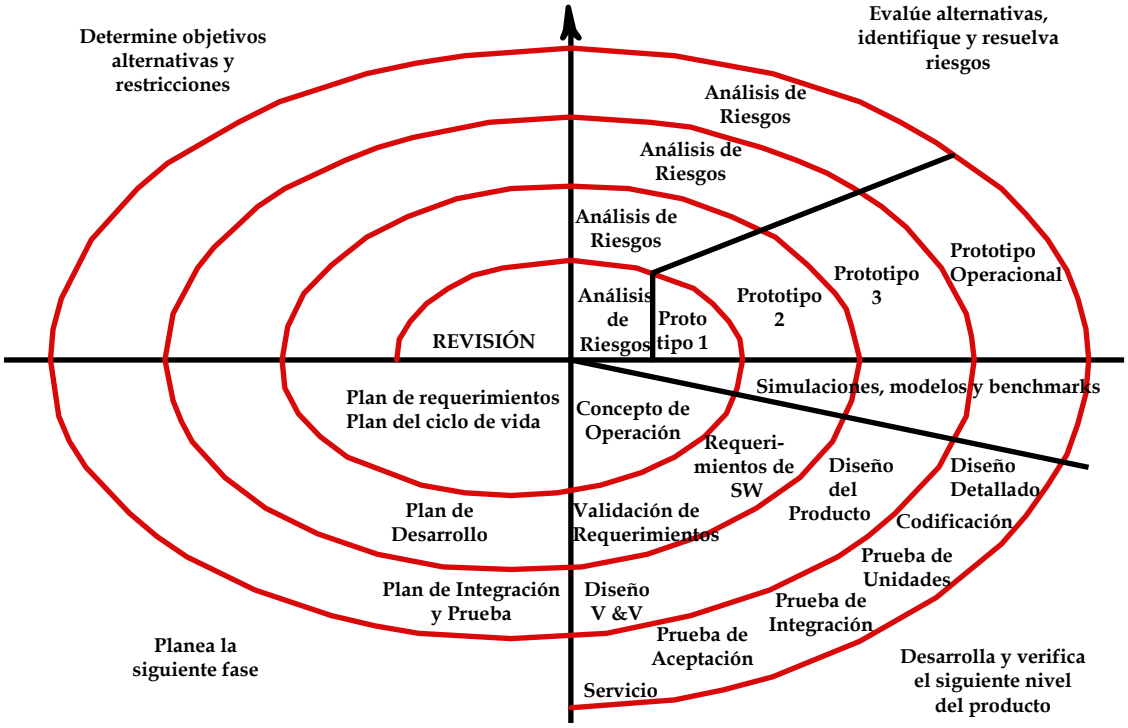


Figura 2.9 Modelo espiral.

Como desventaja, requiere de una considerable habilidad para la evaluación del riesgo, y cuenta con esta habilidad para el éxito.

Modelo espiral WINWIN

Es un modelo basado en la teoría de gestión de sistemas, se basa en el principio de que el proyecto es exitoso si y solo si todos los implicados resultan ganadores. Se enfoca en los intereses, separando a las personas del problema.

El modelo en espiral WINWIN (Ganar & Ganar, traducción de sus siglas en inglés) define un conjunto de actividades de negociación al principio de cada paso alrededor de la espiral. Definiendo las siguientes actividades:

- Identificación del sistema o subsistemas clave de los directivos.
- Determinación de las "condiciones de victoria" de los directivos.
- Negociación de las condiciones de "victoria" de los directivos para reunirlas en un conjunto de condiciones "victoria-victoria" para todos los afectados.

Además del énfasis realizado en la negociación inicial, el modelo en espiral WINWIN mostrado en la figura 2.10 introduce tres hitos en el proceso, llamados *puntos de fijación*, que ayudan a establecer la completitud de un ciclo alrededor de la espiral y proporcionan hitos de decisión antes de continuar el proyecto de software.

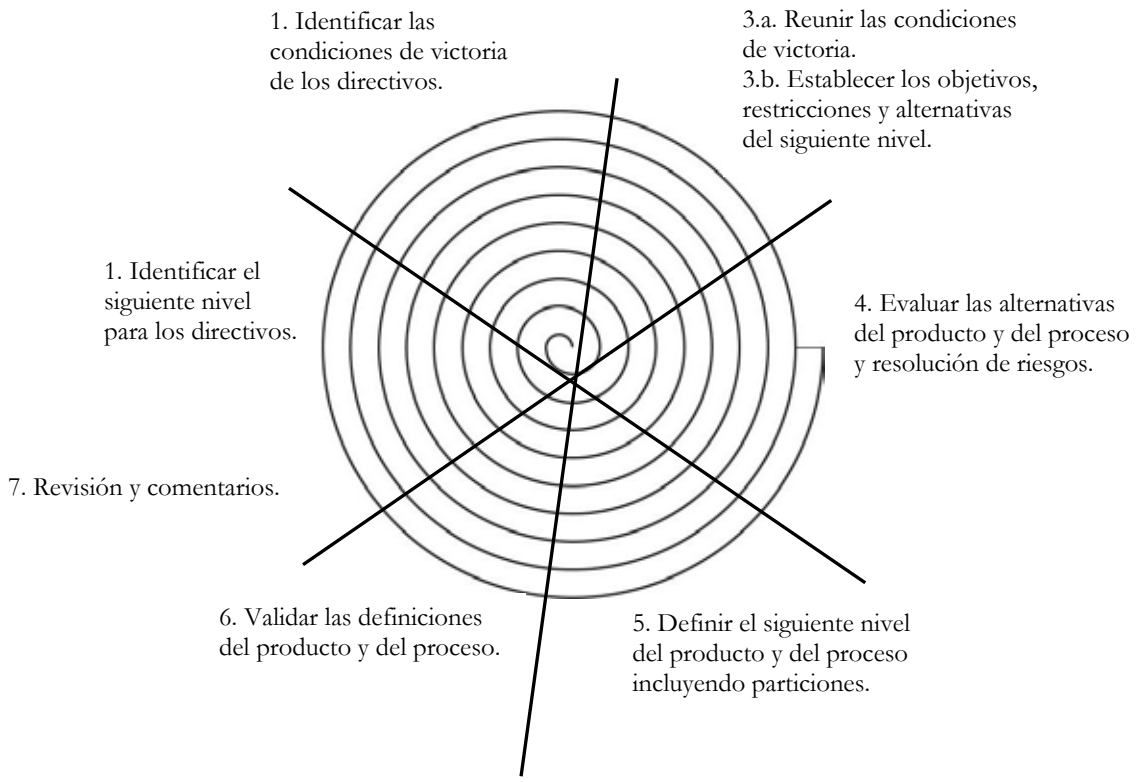


Figura 2.10 Modelo WinWin.

Modelo de desarrollo concurrente

Se puede representar en forma de esquema como una serie de actividades técnicas importantes, tareas y estados asociados a ellas.

La figura 2.11 proporciona una representación esquemática de una actividad dentro del modelo de proceso concurrente. La actividad –análisis- se puede encontrar en uno de los estados destacado anteriormente en cualquier momento dado. Todas las actividades existen concurrentemente, pero residen en estados diferentes.

El modelo se utiliza a menudo como el paradigma de desarrollo de aplicaciones cliente/servidor. Cuando se aplica a cliente/servidor, define actividades en dos dimensiones: de sistemas y de componentes. Los aspectos del nivel de sistemas se afrontan mediante tres actividades: diseño, ensamblaje y uso.

En realidad, el modelo de proceso concurrente es aplicable a todo tipo de desarrollo de software y proporciona una imagen exacta del estado actual de un proyecto.

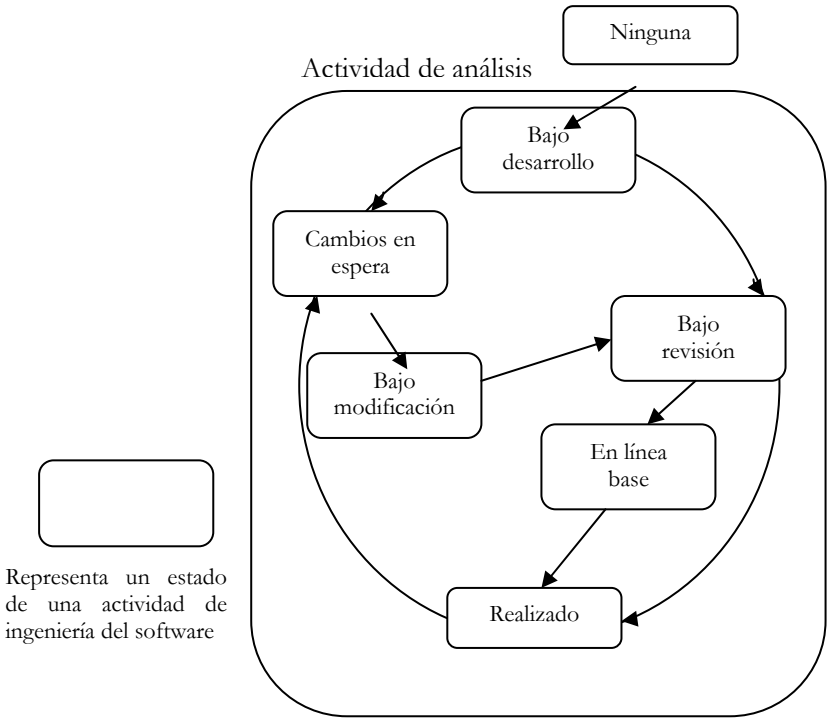


Figura 2.11 Modelo de desarrollo concurrente.

Desarrollo basado en componentes

Las tecnologías de objetos proporcionan el marco de trabajo técnico para un modelo de proceso basado en componentes para la ingeniería del software. El paradigma orientado a objetos enfatiza la creación de clases que encapsulan tanto los datos como los algoritmos que se utilizan para manejar los datos. Si se diseñan y se implementan adecuadamente, las clases orientadas a objetos son reutilizables por las diferentes aplicaciones y arquitecturas de sistemas basados en computadora.

El modelo de desarrollo basado en componentes, como se ilustra en la figura 2.12, incorpora muchas de las características del modelo en espiral. Es evolutivo por naturaleza, y exige un enfoque iterativo para la creación del software.

El proceso unificado de desarrollo de software representa un número de modelos de desarrollo basados en componentes que han sido propuestos en la industria. Utilizando el Lenguaje de Modelado Unificado (UML por sus siglas en inglés *Unified Modeling Language*), el proceso unificado define los componentes que se utilizarán para construir el sistema y las interfaces que conectarán los componentes.

Utilizando una combinación del desarrollo incremental e iterativo, el proceso unificado define la función del sistema aplicando un enfoque basado en escenarios. Entonces acopla la función con un marco de trabajo arquitectónico que identifica la forma que tomará el software.

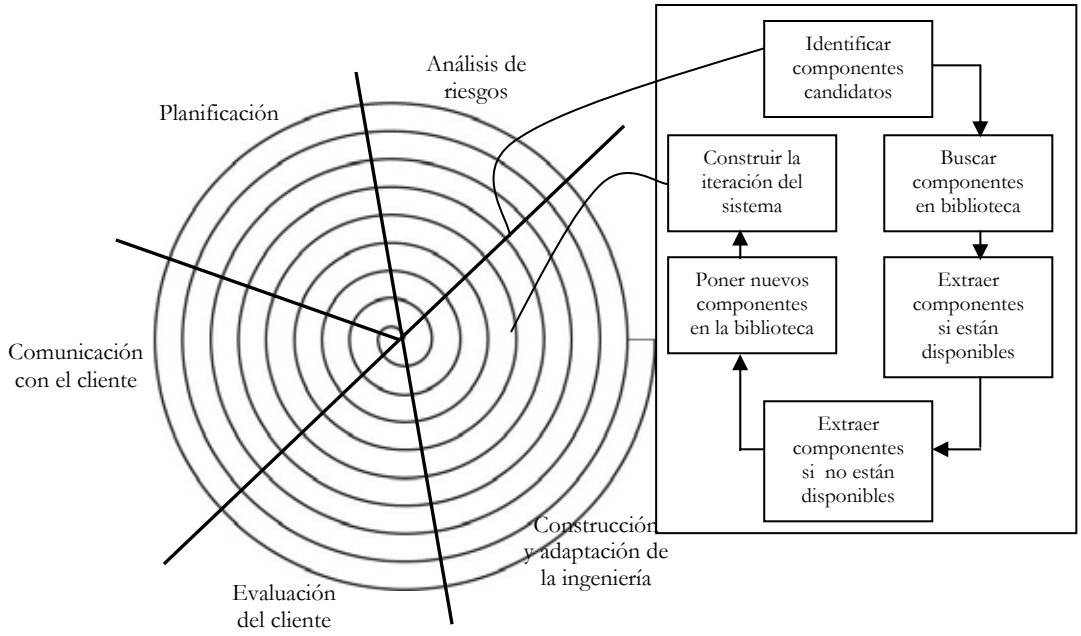


Figura 2.12 Modelo basado en componentes.

Modelo de métodos formales

Comprende un conjunto de actividades que conducen a la especificación matemática del software de computadora, además permiten especificar, desarrollar y verificar un sistema basado en computadora aplicando una notación rigurosa y matemática.

Cuando se utilizan métodos formales durante el diseño, sirven como base para la verificación de programas y por consiguiente permiten descubrir y corregir errores que no se pudieron detectar de otra manera.

Es posible que el enfoque a través de métodos formales sea más recomendable para software que se debe construir con mucha seguridad, y para detección de errores a tiempo que eviten pérdidas económicas.

Técnicas de cuarta generación

El término Técnicas de Cuarta Generación (T4G) abarca un amplio espectro de herramientas de software que tienen algo en común: facilitan la especificación de algunas características del software de alto nivel, la herramienta genera automáticamente el código fuente basándose en la especificación del técnico. Este paradigma para la ingeniería del software se orienta hacia la posibilidad de especificar el software usando formas de lenguaje especializado o notaciones gráficas que describa el problema que hay que resolver en términos que los entienda el cliente.

En resumen, el estado actual de los enfoques de estas técnicas, se describe en los siguientes puntos:

1. El uso de T4G es un enfoque viable para muchas de las diferentes áreas de aplicación. Junto con las herramientas de ingeniería de software asistida por computadora y los generadores de código, además ofrece una solución fiable a muchos problemas del software.

2. Los datos recogidos en compañías que usan estas técnicas parecen indicar que el tiempo requerido para producir software se reduce mucho para aplicaciones pequeñas y de tamaño mediano medio, y que la cantidad de análisis y diseño para las aplicaciones pequeñas también se reduce.

3. Sin embargo, para grandes trabajos de desarrollo de software exige el mismo o más tiempo de análisis, diseño y prueba (actividades de ingeniería del software), para lograr un ahorro sustancial de tiempo que puede conseguirse mediante la eliminación de la codificación.

Al combinarse estás técnicas con enfoques de ensamblaje de componentes, dicho paradigma se puede convertir en el enfoque dominante hacia el desarrollo del software.

2.3 Lenguajes de programación

Son sistemas de comunicación. Un lenguaje de programación consiste en todos los símbolos, caracteres y reglas de uso que permiten a las personas "comunicarse" con las computadoras. Los lenguajes de programación deben tener instrucciones que pertenecen a las categorías ya familiares de entrada/salida, cálculo /manipulación de textos, lógica/comparación y almacenamiento /recuperación.

Tienen un conjunto de instrucciones que permiten realizar dichas operaciones, y además poseen una marcada diferencia en los símbolos, caracteres y sintaxis de los lenguajes de bajo, medio y alto nivel

2.3.1 Tipos y características fundamentales

Lenguaje de bajo nivel

También denominado de bajo nivel consta de cadenas de números binarios (ceros y unos) y es el único que interpretan directamente los procesadores, cada instrucción se corresponde con un código máquina equivalente. Las instrucciones preparadas en cualquier lenguaje de máquina tienen por lo menos dos partes. La primera es el comando u operación, que dice a la computadora cuál es la función que va a realizar. Las computadoras tienen un código de operación para cada una de sus funciones. La segunda parte de la instrucción es el operando, que indica a la computadora donde hallar o almacenar los datos y otras instrucciones que se van a manipular; el número de operandos de una instrucción varía en las distintas computadoras.

Generalmente, en la codificación de los programas se empleaba el sistema hexadecimal para simplificar el trabajo de escritura.

Lenguaje de nivel intermedio

Permiten un manejo abstracto (independiente de la máquina), pero sin perder mucho del poder y eficiencia que tienen los lenguajes de bajo nivel.

El lenguaje ensamblador es el primer intento de sustituir el lenguaje máquina por otro más similar a los utilizados por las personas. En este lenguaje, cada instrucción equivale a una instrucción en lenguaje máquina, utilizando para su escritura palabras mnemotécnicas en lugar de cadenas de bits.
A principios de la década de los años 50, y con el fin de facilitar la labor de los programadores, se desarrollaron códigos mnemotécnicos para las operaciones y direcciones simbólicas. Uno de los primeros pasos para mejorar el proceso de preparación de programas fue sustituir los códigos de operación numéricos del lenguaje de máquina por símbolos alfabéticos, que conforman un *lenguaje mnemotécnico*. Las computadoras actuales tienen códigos mnemotécnicos aunque, naturalmente, los

símbolos que se usan varían en las diferentes marcas y modelos. La computadora sigue utilizando el lenguaje de máquina para procesar los datos, pero los programas ensambladores traducen antes los símbolos de código de operación especificados a sus equivalentes en lenguaje de máquina.

Una desventaja importante de estos lenguajes es que tienen una orientación a la máquina. Es decir, están diseñados para la marca y modelo específico de procesador que se utiliza.

Lenguajes de alto nivel

Están dirigidos a solucionar problemas mediante el uso de estructuras dinámicas de datos y son independientes de la arquitectura del ordenador, entre sus principales características.

El desarrollo de las técnicas mnemotécnicas y las macroinstrucciones condujo, a su vez, al desarrollo de lenguajes de alto nivel que a menudo están orientados hacia una clase determinada de problemas de proceso.

A diferencia de los programas de ensamble, los programas en este lenguaje se pueden utilizar con diferentes marcas de computadoras sin tener que hacer modificaciones considerables.

Esto permite reducir sustancialmente el costo de la reprogramación cuando se adquiere equipo nuevo. Otras ventajas de los lenguajes de alto nivel son: fáciles de aprender, se pueden escribir más rápidamente, y permiten mejor documentación.

2.3.2 Lenguajes estructurados y orientados a objetos

Lenguajes estructurados

La programación estructurada es especialmente útil, cuando se necesitan realizar correcciones o modificaciones después de haber concluido un programa o aplicación. Al utilizarse, es mucho más sencillo entender la codificación del programa, que se habrá hecho en diferentes secciones.

Dicha programación se basa en una metodología de desarrollo de programas llamada refinamiento sucesivo: se plantea una operación como un todo y se divide en segmentos más sencillos o de menor complejidad. Una vez terminado todos los segmentos del programa, se procede a unificar las aplicaciones realizadas por el *pool* de programadores. Al utilizarse adecuadamente, esta integración debe ser sencilla y no presentar problemas al integrar la misma, y de presentar algún problema, será rápidamente detectable para su corrección.

La representación grafica de la programación estructurada se realiza a través de diagramas de flujo, el cual representa el programa con sus entradas, procesos y salidas.

Estos lenguajes proponen segregar los procesos en estructuras lo más simple posibles, las cuales se conocen como secuencia, selección e interacción. Ellas están disponibles en todos los lenguajes modernos de programación imperativa en forma de sentencias. Combinando esquemas sencillos se pueden llegar a construir sistemas amplios y complejos pero de fácil entendimiento.

Lenguajes orientados a objetos

El paradigma Orientado a Objetos (OO) se basa en el concepto de objeto. Un **objeto** es aquello que tiene estado (propiedades más valores), comportamiento (acciones y reacciones a mensajes) e identidad (propiedad que lo distingue de los demás objetos). La estructura y comportamiento de objetos similares están definidos en su clase común; los términos instancia y objeto son intercambiables. Una **clase** es un conjunto de objetos que comparten una estructura y comportamiento común.

La diferencia entre un objeto y una clase es que un objeto es una entidad concreta que existe en tiempo y espacio, mientras que una clase representa una abstracción, la "esencia" de un objeto. De aquí que un objeto no es una clase, sin embargo, una clase puede ser un objeto.

Las propiedades del objeto son claves:

- **Encapsulación**. En el proceso de ocultar todos los detalles de un objeto que no contribuyen a sus características esenciales.
- **Abstracción**. Es una descripción simplificada o especificación de un sistema que enfatiza algunos de los detalles o propiedades del sistema, mientras suprime otros.
- **Modularidad**. Es la propiedad de un sistema que ha sido descompuesto en un conjunto de módulos coherentes e independientes.
- **Jerarquía o herencia**. Es el orden de las abstracciones organizado por niveles.
- **Tipificación**. Es la definición precisa de un objeto de tal forma que objetos de diferentes tipos no puedan ser intercambiados o puedan intercambiarse de manera muy restringida.
- **Concurrencia**. Es la propiedad que distingue un objeto que está activo de uno que no lo está.
- **Persistencia**. Es la propiedad de un objeto a través de la cual su existencia trasciende el tiempo (es decir, el objeto continua existiendo después de que su creador ha dejado de existir) y/o el espacio (es decir, la localización del objeto se mueve del espacio de dirección en que fue creado).

Las relaciones entre objetos definen el comportamiento del sistema. Se dice que los objetos actúan entre sí mediante **mensajes**, es decir, acciones que pide el objeto transmisor que ejecute el objeto receptor.

Actualmente las metodologías más importantes de análisis y diseño de sistemas han confluido en lo que se es el UML, bajo el respaldo del *Object Management Group*.

2.3.3 Programación con Visual Studio .NET©

.NET es una plataforma de software que conecta información, sistemas, personas y dispositivos. Desarrollado con base en los estándares de servicios Web XML, .NET permite que los sistemas y aplicaciones, ya sea nuevos o existentes, conecten sus datos y transacciones independientemente del sistema operativo, tipo de computadora o dispositivo móvil que se utilice, o del lenguaje de programación empleados para crearlo.

.NET Framework

.NET Framework es el modelo de programación de la plataforma .NET para crear, implementar y ejecutar aplicaciones Web, aplicaciones de cliente inteligente y servicios Web XML. Incluye motor de ejecución común (CLR por sus siglas en inglés, *Common Language Runtime*), y bibliotecas de clases, como se ilustra en su marco de trabajo, figura 2.13.

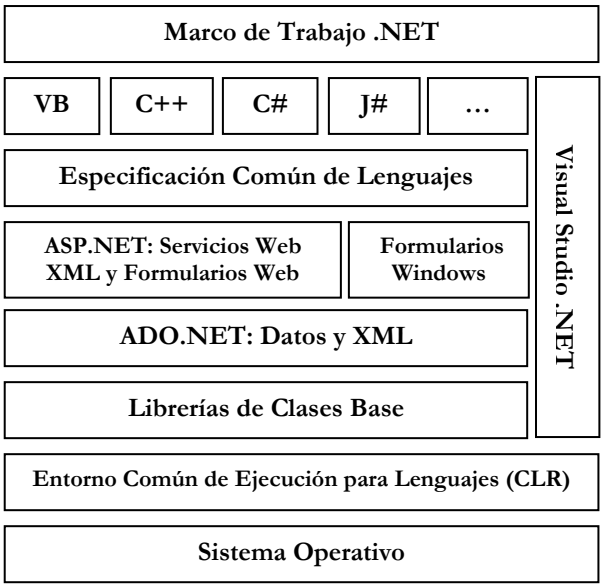


Figura 2.13 Marco de trabajo de .NET Framework.

ASP .NET

ASP.NET proporciona un enfoque exclusivo para el desarrollo de aplicaciones Web. Se basa en el entorno de trabajo .NET de Microsoft y este entorno se basa a su vez en el motor de ejecución. Por lo tanto, las aplicaciones ASP.NET se benefician de todas las ventajas del motor CLR.

Entre sus principales características destacan las siguientes: compatibilidad para varios lenguajes de programación, desarrollo entre lenguajes, separación de la lógica del contenido y de la aplicación, autenticación segura de usuario, nueva arquitectura de procesamiento de servidor, funciones de depuración y seguimiento mejoradas, mayor control sobre la configuración de la aplicación, implementación más sencilla de la aplicación y funciones de caché mejorada.

2.4 Bases de datos

Es un programa residente en memoria, que se encarga de gestionar todo el tratamiento de entrada, salida, protección y elaboración de la información de interés del usuario.

Una definición más formal es la siguiente, es una colección de datos integrados con redundancia controlada y con una estructura que refleja las interrelaciones y restricciones existentes en el mundo real; los datos son compartidos por diferentes usuarios y aplicaciones, se mantienen independientes de estas, y su definición y descripción, únicas para cada tipo de datos esta almacenada junto con los mismos datos. Los procesos de actualización y recuperación son comunes y bien determinados, siendo capaces de conservar la integridad, seguridad y confidencialidad del conjunto.

2.4.1 Conceptos

Datos

Un dato es la unidad mínima de información; son hechos sin valuar, un valor sin significado. Un dato puede representar hechos, ideas o conceptos que pueden ser reunidos y representados electrónicamente en forma digital.

Los datos son hechos aislados y en bruto, que deben ser procesados por varias operaciones para obtener resultados relacionados con la evaluación e identificación de personas, eventos y objetos, es decir, obtener la información. La figura 2.14 muestra la función de cómo responde un dato dentro de una base de datos.

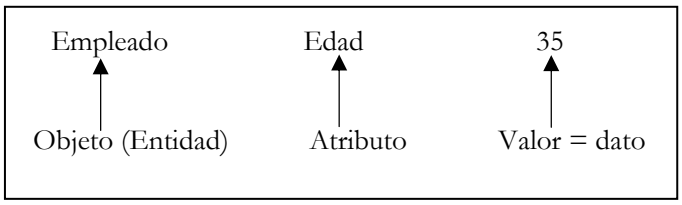


Figura 2.14 Representación de un dato dentro de una base de datos.

Información

Es un conjunto de datos interrelacionados entre sí que tienen un significado del cual se pueden obtener conocimientos para una futura toma de decisiones. Se obtiene asociando los hechos en un contexto determinado, es decir, la adición o el procesamiento de los datos proporcionan un conocimiento o entendimiento de ciertos factores.

Las operaciones que se le pueden aplicar a los datos son de dos tipos:

- Lógicas. Seleccionar, ordenar, verificar, calcular, etc.
- Técnicas. Clasificación, almacenamiento, destrucción, reproducción y distribución.

Tipos de Bases de Datos

Desde el punto de vista de la organización lógica, se dividen en:

a) Jerárquicas (Progress)

   Es un tipo de sistema gestor de bases de datos que almacenan la información en una estructura jerárquica que enlaza los registros en forma de estructura de árbol (similar a un árbol visto al revés), en donde un *nodo padre* de información puede tener varios nodos *hijo*.

b) Relacionales (Oracle, Access, SyBase, SQL)

   Es una base de datos basada en un modelo relacional. El término se refiere a una colección específica de datos pero a menudo es usado como sinónimo del software usado para gestionar esa colección de datos. Ese software se conoce como sistema gestor de base de datos relacional o RDBMS (*Relational Database Management System*).

Desde el punto de vista de número de usuarios:

a) Monousuario (dBase, Acces)
b) Multiusuario cliente/servidor (Oracle, SQL)

Funciones de las bases de datos

1. Permitir la introducción de datos por parte de los usuarios (o programadores)
2. Salida de datos
3. Almacenamiento de datos
4. Protección de datos (seguridad)
5. Elaboración de datos

Básicamente, la comunicación del usuario-programador con la base de datos se hace a través de un lenguaje de consultas denominado SQL: lenguaje estructurado de consultas (*Structured Query Laguage*, por sus siglas en inglés), mismo que permite las funciones descritas en la figura 2.15.

Figura 2.15 Funciones de bases de datos.

Componentes de bases de datos

Una base de datos consta de varios componentes, los cuales se describen a continuación:

Motor: El programa ejecutable que debe estar en memoria para manejar la base de datos. Cuando este programa se ejecuta se dice que la base de datos está levantada (startup), en caso contrario se dice que la base de datos está abajo (shutdown).

Servicio de red: Es un programa que se encarga de establecer las conexiones y transmitir datos entre cliente y servidor o servidor y servidor.

Listener (escuchador): Es un programa residente en memoria que se encarga de recibir las llamadas que llegan a la base de datos desde la red, y de pasárselas a esta. Una base de datos que no tenga un listener cargado, no podrá recibir llamadas remotas. El listener se comunica con el servicio de red.

Utilidades: Programas de utilidad como pueden ser: interpretes de consultas, programas de administración de bases de datos, programas de copia de seguridad, monitores de rendimiento.

La figura 2.16 ilustra los componentes citados, en una base de datos.

A todo el conjunto de la base de datos se le denomina sistema de gestión de bases de datos relacionales (RDBMS por sus siglas en inglés, *Relational DataBase Manager System*).

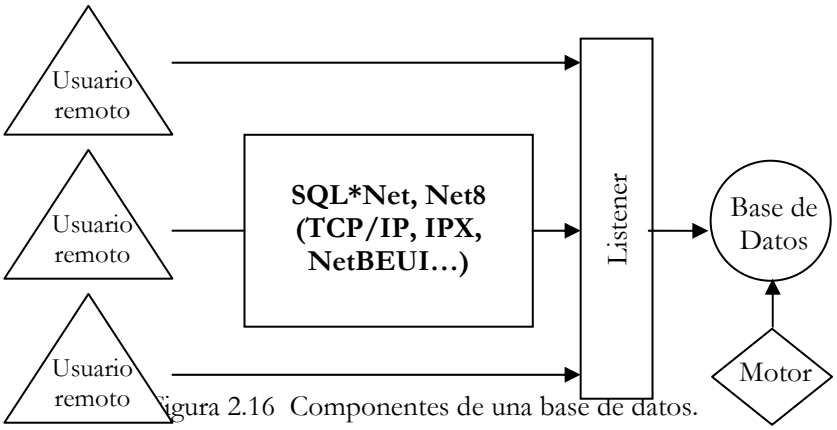


Figura 2.16 Componentes de una base de datos.

2.4.2 Modelo entidad-relación

Es un concepto de modelado para bases de datos, mediante el cual se pretende visualizar los objetos que pertenecen a la base de datos como **entidades** las cuales tienes unos atributos y se vinculan mediante r**elaciones**.

Dicho modelo fue creado por el prestigiado doctor ciencias de la computación y matemáticas de Harvard, Peter Chen en 1976. Es el modelo conceptual más utilizado para el diseño conceptual de bases de datos, ha sido la base para diversas metodologías sobre análisis y diseño de sistemas, herramientas de ingeniería de software asistida por computador y repositorios de sistemas.

El modelo entidad-relación está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas, los cuales se muestran en la figura 2.17.

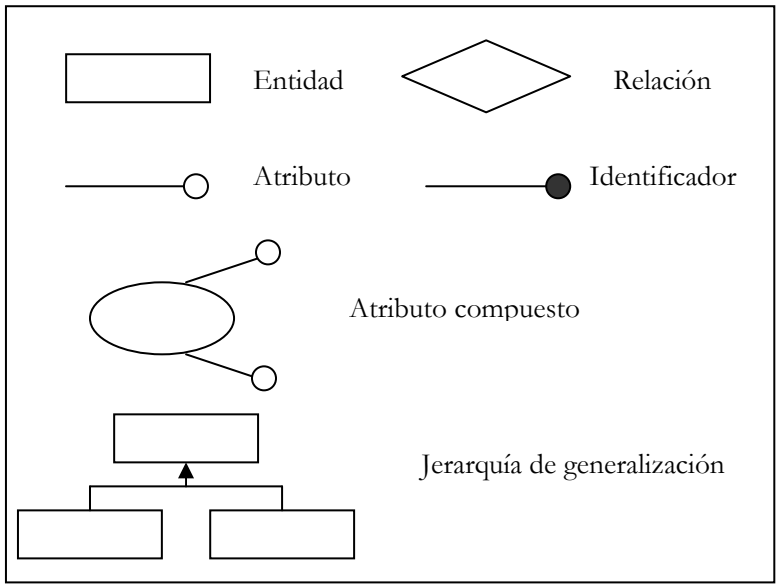


Figura 2.17 Elementos del modelo entidad-relación

El primer paso en el diseño de una base de datos es la producción del esquema conceptual. Normalmente, se construyen varios esquemas conceptuales, cada uno para representar las distintas visiones que los usuarios tienen de la información.

A los esquemas conceptuales correspondientes a cada vista de usuario se les denomina "esquemas conceptuales locales". Cada esquema se compone de entidades, relaciones, atributos, dominios de atributos e identificadores. También tendrá una documentación, que se irá produciendo durante su desarrollo. Las tareas a realizar en el diseño conceptual son las siguientes:

1. Identificar las entidades.
2. Identificar las relaciones.
3. Identificar los atributos y asociarlos a entidades y relaciones.
4. Determinar los dominios de los atributos.
5. Determinar los identificadores.
6. Determinar las jerarquías de generalización (si las hay).
7. Dibujar el diagrama entidad-relación.
8. Revisar el esquema conceptual local con el usuario.

2.4.3 Modelo relacional

Es un modelo de datos para la gestión de una base de datos, basado en la lógica de predicado y en la teoría de conjuntos. Este modelo considera la base de datos como una colección de relaciones. De manera simple, una relación representa una tabla, en que cada fila representa una colección de valores que describen una entidad del mundo real. Cada fila se denomina tupla o registro y cada columna campo.

Ventajas:

1. Garantiza herramientas para evitar la duplicidad de registros, a través de campos claves o llaves.
2. Garantiza la integridad referencial: así al eliminar un registro elimina todos los registros relacionados dependientes.
3. Favorece la normalización por ser más comprensible y aplicable.
4. Se utiliza a nivel conceptual.

En 1970, Edgar Frank Codd, reconocido científico informático inglés, propuso el modelo relacional. Las ventajas de este modelo y su enfoque matemático centraron los esfuerzos de la industria dando lugar a los sistemas gestores de bases de datos relacionales. Estos últimos han reemplazado a las bases de datos jerárquicas hoy día, pero no completamente, debido a que el rendimiento de las bases de datos jerárquicas sigue sin ser superado por las bases de datos relacionales. Además estos sectores sufren un gran volumen de transacciones.

El lenguaje más común para construir las consultas a bases de datos relacionales es SQL, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

Las bases de datos relacionales pasan por un proceso al que se le conoce como normalización de una base de datos, la cual es entendida como el proceso necesario para que una base de datos sea utilizada de manera óptima.

Formas normales

El análisis de un sistema de bases de datos consta de dos fases principales:

- Análisis conceptual (o lógico, o relacional)

  Es un análisis abstracto de aquellas entidades que formarán la base de datos, así como las relaciones que establecen unas con otras y las restricciones que se aplican a cada una de ellas. El resultado de esta fase de análisis se ve reflejado en algo llamado Modelo Conceptual o lógico, que es una descripción de las entidades, atributos, relaciones y restricciones que compondrán la base de datos.

- Análisis físico

  Consta de un análisis específico teniendo en cuenta qué base de datos se va a utilizar y en qué arquitectura se va a implementar la base de datos.

Las formas normales son tres reglas que se deben tener en cuenta dentro del análisis conceptual utilizando concretamente entidad/relación.

El proceso de aplicar las tres formas normales se llama *normalización*. Un diseño de base de datos que no cumpla la primera forma normal no será correcto. Cuantas más formas normales cumpla el diseño de base de datos, significará que la base de datos está más correctamente analizada.

Una vez que se tiene el esquema relacional, se revisa que no haya ninguna relación no normalizada, esto es, con grupos repetitivos. Se eliminan todos los grupos repetitivos de esta relación, obteniendo un conjunto de relaciones en Primera Forma Normal (1FN). Se eliminan las dependencias funcionales parciales, para obtener relaciones en Segunda Forma Normal (2FN). Finalmente, se eliminan las dependencias transitivas, creando relaciones en Tercera Forma Normal (3FN).

Primera forma normal

Una relación normalizada es una relación que tiene sólo valores elementales (o simples) en la intersección de cada renglón y columna. Así, una relación normalizada no tiene grupos repetitivos.

Para normalizar una relación que contienen un solo grupo repetitivo, se elimina el grupo repetitivo y se forman dos nuevas relaciones.

Segunda forma normal

Para eliminar anomalías de la 1FN, se deben eliminar las dependencias parciales. Una relación está en 2FN, si esta en 1FN y se han eliminado dependencias parciales.

Tercera forma normal

Una relación esta en 3FN si está en 2FN y no tienen dependencias transitivas. Esto es cada atributo no llave depende totalmente de la llave primaria y no hay dependencias transitivas, esta última ocurre cuando un atributo o llave depende de uno o más atributos no llave.

2.5 Software libre

Nace de la mano del propio software en la década de los años 60. Las gigantescas máquinas de entonces, hacían uso de programas cuyo código fuente estaba a la vista de todos y se podía distribuir libremente. Esto provocó que existiera una pequeña comunidad de científicos y programadores que intercambiara código, y asimismo informará de errores e ideas. El software entonces sólo representaba un valor añadido a las costosas computadoras y se solía distribuir gratuitamente por los fabricantes.

A mitad de la década de los años 80, Richard Stallman, célebre científico norteamericano cuya mayor influencia radica en el establecimiento de un marco de referencia moral, político y legal opuesto al desarrollo y distribución del software privado, formalizó las ideas básicas del movimiento de software libre que está revolucionando la industria del software. Dicho software, tal y como se conoce hoy, dio sus primeros pasos con un manifiesto en favor de la libertad de expresión y un proyecto conocido hoy mundialmente, el proyecto GNU (acrónimo recursivo que significa GNU No es UNIX).

Definición

El software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. Específicamente, el proyecto GNU, lo refiere a cuatro libertades de los usuarios del software, explicados en la siguiente tabla 2.4:

Nivel de libertad	Descripción
Libertad 0	La libertad de usar el programa, con cualquier propósito
Libertad 1	La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades. *
Libertad 2	La libertad de distribuir copias, con lo que puedes ayudar a alguien.
Libertad 3	La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. *
*	El acceso al código fuente es una condición previa para esto.

Tabla 2.4 Libertades de los usuarios de software, según GNU.

En los últimos años se ha venido escuchando cada vez más los términos software libre y, más recientemente software de fuentes abiertas (*Open Source Software,* en inglés).

Estos términos se refieren al modelo de desarrollo y de distribución del software desarrollado cooperativamente.

Actualmente, la mayoría del software libre se produce por equipos internacionales que cooperan a través de la libre asociación. Los equipos están típicamente compuestos por individuos con una amplia variedad de motivaciones.

2.6 Servidores Web

Un servidor Web sirve contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que se comunican a través del protocolo HTTP (*HyperText Transfer Protocol*). Se pueden utilizar varias tecnologías en el servidor para aumentar su potencia más allá de su capacidad de entregar páginas HTML; éstas incluyen scripts CGI (*Common Gateway Interface*), seguridad SSL (*Secure Socketr Layer*) y páginas activas del servidor (ASP por sus siglas en inglés, *Active Server Pages*), entre otras.

Servicios Web

Se utilizan para integrar diferentes aplicaciones que acceden a los datos a través de Internet. Para ello, se invocan métodos de servicios Web en Internet a los que las aplicaciones desarrolladas en diferentes plataformas pueden acceder. En otras palabras, un servicio Web es un componente reutilizable (como un método) al que puede recurrir cualquier aplicación Web que se ejecute e Internet.

2.6.1 Servidor Web Apache

Es un software de código abierto que funciona sobre cualquier plataforma. Se distribuye prácticamente con todas las implementaciones de Linux. Es un software que está estructurado en módulos. La configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo.

Es un servidor HTTP de código abierto para plataformas UNIX, Windows y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Su nombre se debe a que originalmente Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA (por sus siglas en inglés, *Nacional Center for Supercomputing Applications*). Era, en inglés, *a patchy server* (un servidor *parcheado*).

Apache presenta entre otras características mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido, pero es criticado por la falta de una interfaz gráfica que ayude en su configuración.

2.6.2 Servicio de Información de Internet (IIS)

Es un conjunto de servicios basados en Internet, para maquinas con Windows. Originalmente se p roporcionaba como opcional en Windows NT, pero posteriormente fue integrado a *Windows 2000©* y *Windows Server 2003©.*

Son servicios de software que admiten la creación, configuración y administración de sitios Web, además de otras funciones de Internet. Los servicios de Microsoft Internet Information Server incluyen el Protocolo de transferencia de noticias a través de la red (NNTP), el Protocolo de transferencia de archivos (FTP) y el Protocolo simple de transferencia de correo (SMTP). Los Servicios de I*nternet Information Server* también se denominan IIS.

Características de IIS 6.0

- *Integración con Servicios de Internet Information Server 6.0.- P*ermiten compartir fácilmente documentos e información a través de la red intranet de una compañía o de Internet. Con IIS, es posible distribuir aplicaciones escalables y confiables basadas en Web e incorporar aplicaciones y datos existentes al Web. Otras características de IIS 6.0 son: confiabilidad y escalabilidad, seguridad y capacidad de administración, y mejor desarrollo y compatibilidad internacional.

- *ASP.- S*on un entorno de secuencias de comandos del lado del servidor que permite crear aplicaciones de servidor Web dinámicas e interactivas.

2.7 Sistema operativo Windows Server 2003©

Es la versión de Windows para servidores lanzada por Microsoft en el año 2003. Basado en el núcleo de Microsoft *Windows XP*, añadiendo una serie de servicios, y bloqueo de algunas características. Es una plataforma productiva construida hasta la fecha para todo tipo de aplicaciones, redes y servicios Web.

Las versiones y descripción de cada una de ellas son las siguientes:

- **Web edition** Diseñado para los servicios y el hospedaje Web.
- **Standard edition** Ofrece un gran número de servicios útiles para organizaciones de cualquier tamaño.
- **Enterprise edition** Para organizaciones de mayor tamaño que la s*tandard edition*.
- **Datacenter edition** Para organizaciones que requieran bases de datos más escalables y un procesamiento de transacciones de gran volumen.

La tabla 2.5 describe las características físicas generales de las versiones mencionadas.

Windows Server 2003©				
	Web Edition (Solo 32 bits)	Standard Edition (Solo 32 bits)	Enterpise Edition (32 y 64 bits)	Datacenter Edition (32 y 64 bits)
No. máximo procesadores soportados	2	4	8	64
Memoria máxima	2GB	4GB	32GB (32bit) 64GB (64bit)	32 GB(32bit) 64GB(64bit)
Requerimientos del sistema recomendados	Procesador a 550 Mhz 256 MB RAM 1.5GB DD	Procesador a 550 Mhz 256 MB RAM 1.5 GB DD	Procesador a 550 Mhz 256 MB RAM 1.5 – 2.0 GB DD	Procesador a 550 Mhz 1 GBRAM 1.5-2.0 GB DD

Tabla 2.5 Características de las actuales versiones de Windows 2003.

Las funcionalidades de cada versión se describen en la tabla 2.6.

Windows Server 2003©				
	Web Edition (Solo 32 bits)	Standard Edition (Solo 32 bits)	Enterpise Edition (32 y 64 bits)	Datacenter Edition (32 y 64 bits)
Servicios de Directorio Activo	Si	Si	Si incluido metadirectorio	Si incluido metadirectorio
Servicio de Archivos	Limitado	Si	Si	Si
Servicio de Impresión	No	Si	Si	Si
Clustering	No	No	8 Nodos	8 Nodos
Servicios de Balanceo de Carga	Si	Si	Si	Si
Servicios IIS	Si, servidor Web	Si	Si	Si
Servicios de Fax	No	Si	Si	Si
Cortafuegos básico	No	Si	Si	No
Servicios de	Administración	Servidor,	Servidor,	Servidor,

Terminal	Remota	administración remota	administración remota, session directory	administración remota, session directory
Límite VPN	1	1000 conexiones concurrentes	Ilimitada	Ilimitada
Windows System Resource Manager	No disponible	No disponible	Si	Si

Tabla 2.6 Funcionalidades de las actuales versiones de Windows 2003.

El marco de seguridad de Windows Server 2003©

La figura 2.18 ilustra el marco de seguridad en la plataforma Windows Server 2003, la cual se agrupa en tres rubros: por diseño, de forma predeterminada, y en la implementación.

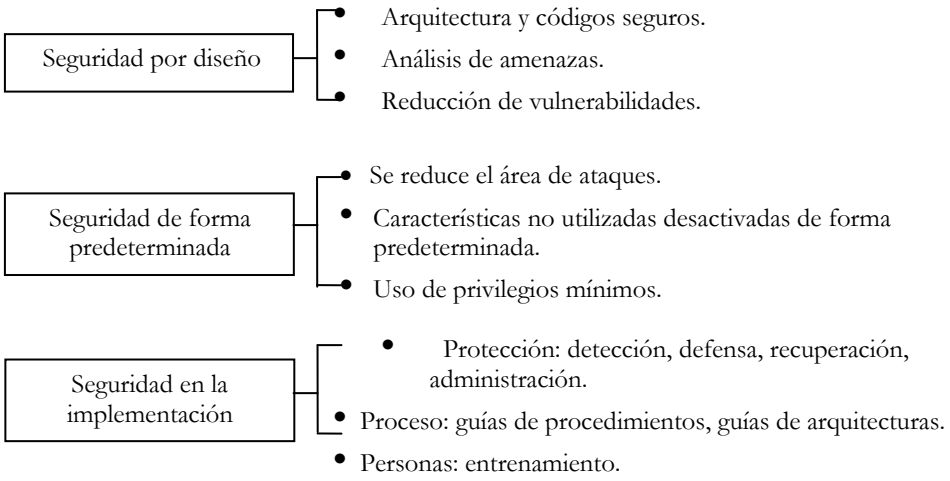


Figura 2.18 Marco de Seguridad de Windows Server 2003©.

2.8 Seguridad en el desarrollo

Entre los retos a los que hay que enfrentarse al implementar la seguridad se incluyen el sopesar las ventajas que tiene el atacante sobre los defensores y cómo afecta el grado de seguridad a los usuarios, así como la transmisión del mensaje de que la seguridad aporta valor empresarial.

- Atacantes frente a defensores: el defensor debe proteger todos los puntos y vigilar continuamente.
- Seguridad frente a facilidad de uso: los desarrolladores de la seguridad deben esforzarse por hacer que el sistema sea lo más seguro posible, pero no imposible de utilizar.
- La seguridad como idea tardía: implementar la seguridad desde el principio es esencial para garantizar una solución robusta ya que cualquier cambio a última hora en el ciclo de desarrollo del producto, es mucho más costoso que si se diseñara la seguridad en primer lugar

El papel del desarrollador en la seguridad de las aplicaciones se basa en estos principales puntos:

- Trabajar con los arquitectos de soluciones y los administradores de sistemas para garantizar la seguridad de las aplicaciones.
- Contribuir a la seguridad adoptando buenas prácticas de desarrollo para lograr aplicaciones seguras.

- Conocer dónde se producen las vulnerabilidades de la seguridad y cómo evitarlas, y utilizar técnicas de programación seguras.

La seguridad a lo largo del ciclo de vida del proyecto debe tenerse en cuenta en todas las etapas: diseño, desarrollo, implementación. Y en todas las capas: red, host y aplicación.

El modelado de amenazas es un análisis de una aplicación basado en la seguridad, que ayuda a un grupo de producto a conocer los puntos más vulnerables del producto, evalúa las amenazas a una aplicación, trata de reducir los riesgos generales de seguridad, busca activos, descubre los puntos vulnerables, identifica amenazas y ayuda a constituir la base de las especificaciones del diseño de seguridad. Representa una parte crucial del proceso de diseño, reduce el costo de proteger una aplicación y ofrece un proceso lógico y eficiente.

La tabla 2.7 establece recomendaciones para mejorar la seguridad y las ventajas que representan.

Recomendación	Ventaja
Adoptar el modelo de amenazas	• Identifica las vulnerabilidades de seguridad • Aumenta el conocimiento de la arquitectura
Entrenar al equipo de desarrollo	• Evita defectos comunes de seguridad • Correcta aplicación de las tecnologías de seguridad.
Revisión de código	• Protege código que tiene acceso a la red • Se ejecuta de forma predeterminada • Utiliza protocolos no autenticados • Se ejecuta con privilegios elevados
Usar herramientas	• Pruebas más coherentes para detectar Vulnerabilidades
Usar soluciones de infraestructura	• Más seguro con SSL/TLS e IPSec
Usar soluciones de componentes	• Robusto con CAPICOM y el espacio de nombres .NET Cryptography
Migrar código administrado	• Evita vulnerabilidades comunes

Tabla 2.7 Recomendaciones para mejorar la seguridad.

Es necesario utilizar y aplicar, durante el desarrollo, las distintas tecnologías de seguridad tales como cifrado, hashing, firmas digitales, certificados digitales, comunicación segura, autenticación, autorización, servidores de seguridad, auditoria, services pack, actualizaciones.

Finalmente hay que considerar lo siguiente:

Aprender de los errores

- ¿Cómo se produjo el error de seguridad?, documéntelo.
- ¿Se ha cometido el mismo error en otra parte del código?
- ¿Cómo se habría evitado?
- ¿Qué se debe cambiar para evitar repetir esta clase de error?
- ¿Necesita actualizar el material de cursos o las herramientas de análisis?

2.9 Redes

En la década de los años 70 se comenzaron a introducir sistemas informáticos que podían ser interconectados de diversas maneras. Actualmente las bases de la arquitectura de sistemas distribuidos están en plena evolución. La información ya no se concibe monopolizada sino accesible para empresas públicas, privadas, organismos de investigación, etc.

La presente era de la información depende por igual de las computadoras y de las redes que las comunican, y con el éxito de Internet, se enfrenta la posibilidad de una red global distribuida que cubre el planeta completo.

Un sistema distribuido es aquel en el que los componentes localizados en computadoras, conectados en red, comunican y coordinan sus acciones únicamente mediante el paso de mensajes.

Distribución no significa descentralización, por ello se entiende dispersión sin integración de las partes que los componen para formar un sólo conjunto.

Las principales ventajas del uso de sistemas distribuidos son:

- Reducción de costos.
- Mejora en tiempo de respuesta => elimina largas colas de espera.
- Recursos compartidos: equipos, datos, programas.

2.9.1 Arquitectura cliente -servidor

Una arquitectura es un conjunto de reglas, definiciones, términos y modelos que se emplean para producir un producto. La arquitectura cliente – servidor agrupa conjuntos de elementos que efectúan procesos distribuidos y cómputo cooperativo.

Entre sus principales características se tiene el mejor aprovechamiento de la potencia del cómputo, reduce el trabajo en la red, opera bajo sistemas abiertos y permite el uso de interfaces gráficas variadas y versátiles.

El cliente (conjunto de software y hardware que invoca los servicios de uno o varios servidores), mantiene y procesa todo el dialogo con el usuario, maneja las pantallas, interpreta los menús y comandos, permite la entrada de datos y su validación y realiza el procesamiento de ayudas, etc.

El servidor (conjunto de hardware y software que responde a los requerimientos del cliente), accede, almacena, actualiza y organiza los datos, administra los recursos compartidos y ejecuta toda la lógica para procesar una transacción.

En general esta arquitectura permitirá compartir cualquier tipo de recurso (datos, noticias, resultados, software, video, etc.) en un entorno distribuido, a pesar las diferencias de hardware que pudieran existir, aprovechando la el medio de comunicación más explotado en la actualidad en los medios informáticos como lo es Internet

CAPÍTULO 3

ANÁLISIS Y DISEÑO DEL SISTEMA

3.1 Objetivo

El presente capítulo describe el estudio de la fase de análisis y diseño para la construcción del sistema requerido por el DSC-FI, apoyándose de la metodología *Racional Unified Process*. Se establecen y definen las herramientas de desarrollo a utilizar (sistema operativo, tipo servidor y lenguaje de programación), se registran los requerimientos del cliente identificando actores y roles como base para la creación del diseño de software, y posteriormente se especifica el modelado de datos y diseño del sistema.

El objetivo de este capítulo es disponer de la arquitectura óptima del sistema para proceder a la construcción final.

3.2 Estructura del modelo de análisis

El modelo de análisis debe lograr tres objetivos primarios:

- Describir lo que requiere el cliente
- Establecer una base para la creación de un diseño de software
- Definir un conjunto de requisitos que se pueda validar una vez que se construya el software.

Para lograr esto, el análisis estructurado (actividad de construcción de modelos) debe tomar la forma que se muestra en la figura 3.1, donde cada componente se describe en los siguientes párrafos.

El *diccionario de datos* (DD) es el almacén que contiene definiciones de todos los objetos de datos consumidos y producidos por el software. El *diagrama entidad-relación* (DER) representa las relaciones entre los objetos de datos. Este último es la notación usada para realizar la actividad del modelado de datos. Los atributos de cada objeto de datos señalado en el DER se pueden representar mediante la descripción de objetos de datos.

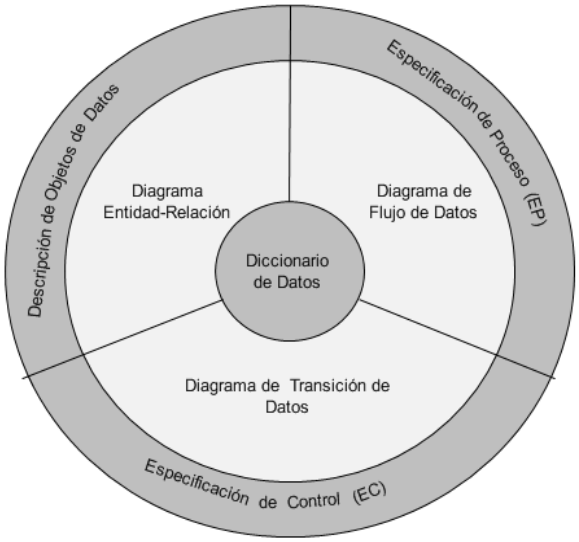


Figura 3.1 Estructura del modelo de análisis.

El *diagrama de flujo de datos* (DFD) sirve para proporcionar una indicación de cómo se transforman los datos a medida que se avanza en el sistema, y para representar las funciones que transforman el flujo de datos. Este diagrama proporciona información adicional que se usa durante el análisis del dominio de información y sirve como base para el modelado de función. En una especificación de proceso se encuentra una descripción de cada función presentada en este diagrama.

El *diagrama de transición de estados* (DTE) indica cómo se comporta el sistema como consecuencia de sucesos externos, a través de la representación de los diferentes modos de comportamiento (estados) del sistema y la manera en que se realizan las transacciones de estado a estado. Este modelo sirve como base del modelado de comportamiento.

Una vez analizados y especificados los requerimientos del software, el diseño del software es la primera actividad técnica de las siguientes: diseño, generación de código y pruebas, que se requieren para construir y verificar el software.

Todos los elementos recopilados en el modelo de análisis proporcionan la información necesaria para crear los siguientes modelos de diseño requeridos para una especificación completa de diseño, los cuales son: diseño de datos, diseño arquitectónico, diseño de la interfaz y diseño a nivel de componentes.

El *diseño de datos* transforma el modelo del dominio de información que se crea durante el análisis en las estructuras de datos que se necesitarán para implementar el software. Este diseño es alimentado de los objetos de datos y las relaciones del diagrama entidad-relación y el contenido del diccionario de datos.

El d*iseño arquitectónico* define la relación entre los elementos estructurales principales del software, los patrones de diseño que se pueden utilizar para lograr los requisitos que se han definido para el sistema, y las restricciones que afectan a la manera en que se pueden aplicar los patrones de diseño arquitectónico. Este modelo es derivado de la especificación del sistema, del modelo de análisis y de la interacción del subsistema definido en el modelo de análisis.

El d*iseño de la interfaz* describe la manera de comunicarse con el software, con sistemas que interoperan dentro de él y con las personas que lo utilizan. Una interfaz implica un flujo de información y tipo específico de comportamiento. Los diagramas de flujo de control y de datos son fuente para este modelo.

El d*iseño a nivel de componentes* transforma los elementos estructurales de la arquitectura del software en una descripción procedimental de los componentes. La información se obtiene de la especificación de procesos, de control y del diagrama de transición de estados.

El diseño es de suma importancia al considerarse el lugar que fomentará la calidad, y es la única forma de convertir exactamente los requisitos de un cliente en un producto o sistema de software finalizado.

La relación entre el análisis y diseño se muestran en la figura 3.2.

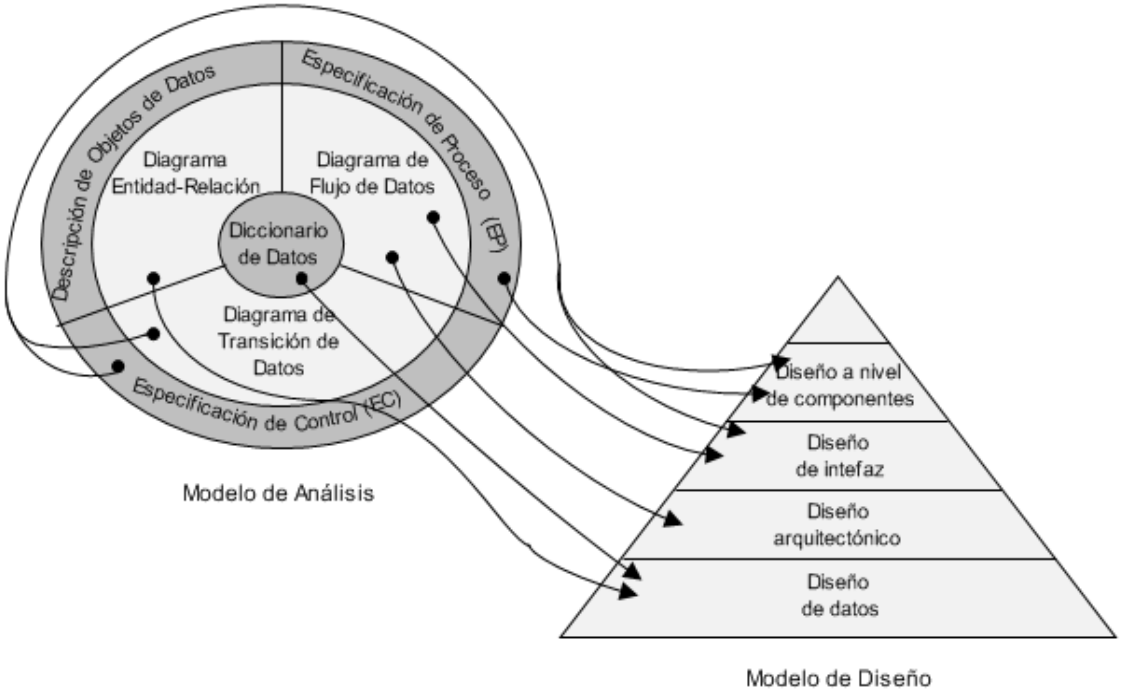


Figura 3.2 Conversión del modelo de análisis a diseño de software.

3.3 Identificación de requerimientos

La Facultad de Ingeniería a través del DSC-FI requiere de la oportuna y eficaz asistencia de una herramienta Web que centralice la información e históricos de los incidentes de cómputo.

La recopilación de los requerimientos está comprendida dentro de la fase inicial de la metodología *RUP* y su seguimiento está ilustrado en la figura 3.3.

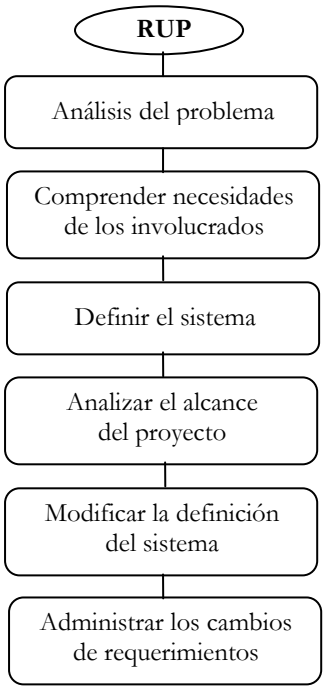


Figura 3.3 Recopilación de requerimientos en RUP.

Esta primera fase -análisis de requerimientos de la metodología RUP, se da a través de la comunicación directa entablada con el ingeniero Rafael Sandoval Vázquez, encargado del DSC-FI, la aplicación de cuestionarios, diagramas de flujo y finalmente el asentamiento de las solicitudes en un documento que es revisado por las partes involucradas.

3.3.1 Estudio del problema

El DSC–FI es el encargado de proporcionar atención oportuna, eficaz y adecuada a los incidentes de seguridad registrados en las zonas comprendidas como A, B, C y P. Estas zonas se describen como sigue:

A – Edificio principal, conjunto norte
B – Edificios: DIE, DIMEI
C – Conjunto sur
P – Edificios de postgrado

Los tipos de incidentes registrados son vulnerabilidades, virus, gusanos, spam, troyanos, phising, entre otros, registrados dentro del ámbito de acción, es decir, en redes y sistemas de la Facultad de Ingeniería.

El procedimiento actual a seguir consiste en los siguientes pasos:

a) El responsable de cómputo del área, el usuario final, alguna entidad externa a la Facultad de Ingeniería o en su caso el propio DSC-FI, detectan una falla en su sistema, de manera tal que imposibilita el desempeño adecuado de las actividades cotidianas.

b) Se solicita la atención del DSC-FI a través de: llamada telefónica, correo electrónico o un oficio dirigido directamente al responsable del departamento, la información que deberá ser proporcionada es:

- Nombre del área y departamento involucrados.
- Nombre completo del encargado de la computadora afectada.
- Breve descripción del incidente de seguridad y elementos afectados.
- Breve descripción de intentos de solución
- Dirección IP y MAC de dicha computadora.

c) El reporte se registra en el historial de solicitudes a fin de dar una clasificación y seguimiento; inmediatamente destina al personal disponible y capacitado para su atención y solución.

d) El personal capacitado acude personalmente al lugar del incidente, verifica los datos del usuario, de la computadora y analiza la posible causa.

e) Hecho un diagnóstico a la computadora afectada, se procede a la solución del problema con las herramientas adecuadas, las cuales son llevadas por el personal de forma externa a fin de evitar la proliferación del incidente a otras computadoras.

f) Después de haber atendido el incidente, el personal levanta un reporte de atención, mismo que el responsable de la computadora afectada debe firmar de conformidad.

g) Si posterior a esto, se siguen presentando irregularidades en la computadora mencionada, el escenario es trasladado al laboratorio del DSC-FI a fin de obtener soluciones alternas. Con todo esto, el problema se sigue atendiendo con el número de oficio original y su estatus es de seguimiento.

h) Finalmente al resolver el problema, se documenta y se genera un expediente de atención, el cual forma parte del historial de atención a incidentes.

i) Con toda la información recopilada el DSC-FI genera periódicamente estadísticas de registro y atención de incidentes, así como su clasificación y periodicidad de ocurrencia, a fin de prevenir e informar a la comunidad de la Facultad de Ingeniería.

3.3.2 Necesidades de los involucrados

Las necesidades de los involucrados en este caso fueron identificadas por las tareas y/o actividades que son llevadas a cabo por cada uno de los integrantes del departamento así como de los solicitantes de su servicio.

Se requiere de una herramienta que permita a través de un sistema cliente-servidor, la consulta, registro, seguimiento, solución, prevención y corrección de los distintos tipos de incidentes de seguridad en cómputo que tienen lugar en redes y sistemas.

Cada vez que se presente la ocurrencia de un incidente de seguridad, el usuario podrá ingresar desde cualquier computadora a la página del departamento, y registrar a detalle el incidente, proporcionando la información antes señalada.

De la misma forma el equipo de trabajo, requiere de una herramienta en línea que le permita la gestión de incidentes y el seguimiento y atención de los mismos.

Finalmente toda la comunidad de la Facultad de Ingeniería, podrá disponer de un sitio donde consultar las últimas noticias referentes a la seguridad en cómputo, y conocer al departamento, es decir, sus funciones y servicios.

3.3.3 Definición del sistema

Las secciones en general serán *presentación*, *administración*, *registro* (inventario, incidente), *reportes* e información propia del área identificada como *somos*, donde:

Presentación.- Mostrará información reciente sobre los ultimas vulnerabilidades, riesgos y noticias y herramientas para atención de incidentes, las estadísticas más relevantes sobre la atención y solución de los mismos. Asimismo desplegará ligas visibles de redireccionamiento a las otras secciones del portal.
Administración.- Se darán de alta noticias, se administrará a los usuarios, es un área de acceso restringido mediante usuario y contraseña.
Registro.- Esta sección deberá comprender los formularios para registro de incidentes e inventario de recursos físicos y lógicos.
Reportes.- Aquí estarán localizadas las opciones para generar los tipos de reportes que se requieran para el DSC-FI.
Somos.- En esta parte se localizará la información referente al departamento, y la forma de contactar directamente al personal del mismo.

3.3.4 Usuarios

Dadas las necesidades de gestión de la herramienta se identifican los siguientes usuarios:

- Administrador
  Este tipo de usuario es el de mayores privilegios dentro del portal, puede hacer modificaciones a la base de datos, solicitar reportes que requiera por un periodo determinado, incrementar o disminuir privilegios a otros usuarios.

  Plasmado en el análisis de requerimientos como la persona responsable de atender, dar seguimiento y solución a los incidentes de cómputo registrados en la Facultad de Ingeniería, así como coordinar y asignar las actividades, recursos, proyectos y tareas involucradas con el Departamento de Seguridad en Cómputo.

- Operador
  Este usuario es el encargado de registrar reportes de incidentes, subir noticias, registrar equipo de inventario y dar seguimiento y solución de los incidentes a su cargo.

  Se identifico dentro de los requerimientos a este usuario, como la persona designada para atender incidentes específicos, asesorar e informar sobre medidas preventivas y correctivas de seguridad en cómputo a otros usuarios dentro de la comunidad de la Facultad de Ingeniería, así como desarrollar actividades de investigación.

- Usuario final
  Este usuario es el encargado de registrar los incidentes detectados en su computadora o sistemas involucrados, de descargar las últimas actualizaciones, parches o herramientas de prevención, de solicitar o hacer comentarios o sugerencias al departamento, de leer y mantenerse informado de las alertas y noticias emitidas por el departamento, también puede consultar las estadísticas e información disponible en el portal.

  Identificado en la etapa de análisis como la persona que forma parte de la comunidad de la Facultad de Ingeniería, dentro de la cobertura del departamento y que requiere de atención a incidentes de seguridad.

Con esta clasificación, el sistema debe permitir la realización de cada una de estas actividades de acuerdo al rol de cada usuario

3.4 Metodología de desarrollo

3.4.1 Historia del Proceso Unificado Racional

El Proceso Unificado Racional (RUP por sus siglas en inglés, *Rational Unified Process*) es un producto comercial desarrollado y comercializado por Rational Software, una compañía de IBM. A continuación se describe en resumen, la historia de la metodología, características principales y estructura del proceso.

El antecedente más importante se ubica en 1967 con la metodología Ericsson (*Ericsson Approach*) elaborada por Ivar Jacobson, quien hizo una aproximación de desarrollo basada en componentes, e introdujo el concepto de caso de uso. Entre los años de 1987 a 1995 Jacobson fundó la compañía *Objectory* AB y lanza el proceso de desarrollo *Objectory* (abreviación de *Object Factory*).

Posteriormente en 1995 *Rational Software Corporation* adquiere *Objectory AB* y entre 1995 y 1997 se desarrolla *Rational Objectory Process* (ROP) a partir de *Objectory* 3.8 y del Enfoque Rational (*Rational Approach*) adoptando Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, *Unified Modeling Language*©).
Desde ese entonces y a la cabeza de Grady Booch, Ivar Jacobson y James Rumbaugh, Rational Software desarrolló e incorporó diversos elementos para expandir RUP, destacándose especialmente el flujo de trabajo conocido como modelado del negocio. En junio del 1998 se lanza *Proceso Unificado Racional.*

3.4.2 Características generales del Proceso Unificado Racional

Los autores destacan que el proceso de software propuesto por RUP tiene tres características esenciales:

- Casos de uso
- Centralización en la arquitectura
- Iterativo e incremental

3.4.2.1 Proceso dirigido por casos de uso

Los casos de uso son una técnica de captura de requisitos que obliga a pensar en términos de importancia para el usuario y no sólo en términos de funciones que serían bueno contemplar. Se definen como un fragmento de funcionalidad del sistema que proporciona al usuario un valor añadido y representan los requisitos funcionales del sistema.

En RUP son una herramienta para especificar los requisitos del sistema y asimismo guían su diseño, implementación y prueba. Además constituyen un elemento integrador y una guía del trabajo como se muestra en la figura 3.4.

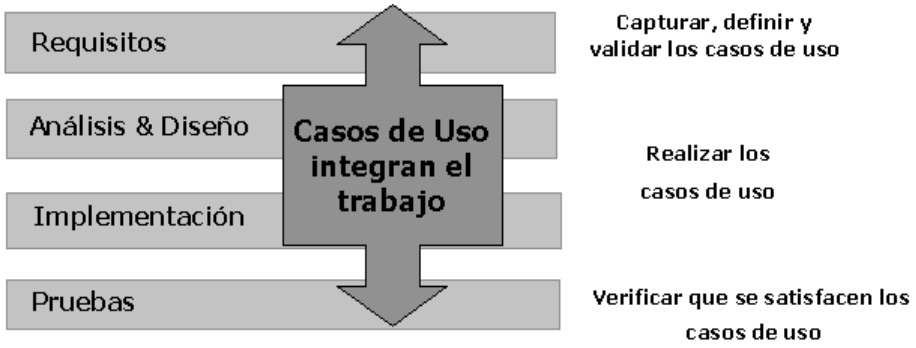


Figura 3.4 Los casos de uso integran el trabajo.

Estos, inician el proceso de desarrollo y proporcionan un hilo conductor, permitiendo establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo.

Basándose en los casos de uso se crean los modelos de análisis y diseño, luego la implementación que los lleva a cabo, y se verifica que efectivamente el producto implemente adecuadamente cada uno de ellos. Todos los modelos deben estar sincronizados con el modelo de casos de uso. Este proceso lo ilustra la figura 3.5.

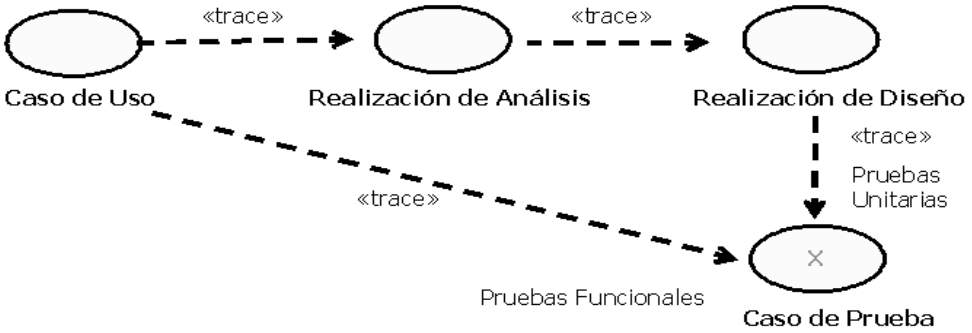
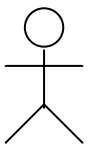


Figura 3.5 Trazabilidad a partir de los casos de uso.

Los diagramas de caso de uso representan de forma esquemática la interrelación existente entre los usuarios de un sistema y las tareas en las que intervienen. Tienen como elementos principales tres iconos que representan a varios conceptos dentro de un sistema, descritos en la tabla 3.1.

Concepto	Icono	Semántica
Actor	 Nombre	El icono representa a un actor específico dentro del sistema en estudio, cada actor debe ser descrito utilizando un *nombre* con el cual identificar su rol dentro del sistema.
Caso de uso	 Nombre del Caso de Uso	Representa las formas en que un actor puede utilizar un sistema, su aplicación supone que toda la funcionalidad requerida del sistema sea descrita a través de ellos.

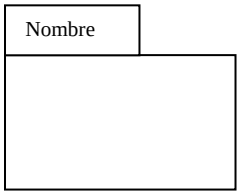
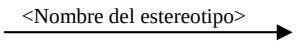
Paquete		El paquete permite agrupar a todos los *use case* contenidos dentro de un sistema, definiendo sus límites y alcance. En cierto sentido, un paquete contiene la representación de los elementos de la interfaz de usuario de un sistema.
Enlace		Un enlace representa a la intervención que tiene un actor en el cumplimiento de un caso de uso.

Tabla 3.1 Conceptos de casos de uso.

3.4.2.2 Proceso centrado en la arquitectura

La arquitectura de un sistema es la organización o estructura de sus partes más relevantes, lo que permite tener una visión común entre todos los involucrados (desarrolladores y usuarios) y una perspectiva clara del sistema completo, necesaria para controlar el desarrollo.

La arquitectura involucra los aspectos estáticos y dinámicos más significativos del sistema, está relacionada con la toma de decisiones que indican cómo tiene que ser construido el sistema y ayuda a determinar en qué orden. Además la definición de la arquitectura debe tomar en consideración elementos de calidad del sistema, rendimiento, reutilización y capacidad de evolución por lo que debe ser flexible durante todo el proceso de desarrollo. La arquitectura se ve influenciada por la plataforma software, sistema operativo, gestor de bases de datos, protocolos, consideraciones de desarrollo como sistemas heredados. Muchas de estas restricciones constituyen requisitos no funcionales del sistema.

En el caso de RUP además de utilizar los casos de uso para guiar el proceso se presta especial atención al establecimiento temprano de una buena arquitectura que no se vea fuertemente impactada ante cambios posteriores durante la construcción y el mantenimiento.

Cada producto tiene tanto una función como una forma. La función corresponde a la funcionalidad reflejada en los casos de uso y la forma la proporciona la arquitectura. Existe una interacción entre los casos de uso y la arquitectura, estos deben encajar en la arquitectura cuando se llevan a cabo y la arquitectura debe permitir el desarrollo de todos los requeridos, actualmente y en el futuro. Esto provoca que tanto arquitectura como cada caso deban evolucionar en paralelo durante todo el proceso de desarrollo de software.

En la figura 3.6 se ilustra la evolución de la arquitectura durante las fases de RUP. Se tiene una arquitectura más robusta en las fases finales del proyecto. En las fases iniciales lo que se hace es ir consolidando la arquitectura por medio de líneas base y se va modificando dependiendo de las necesidades del proyecto.

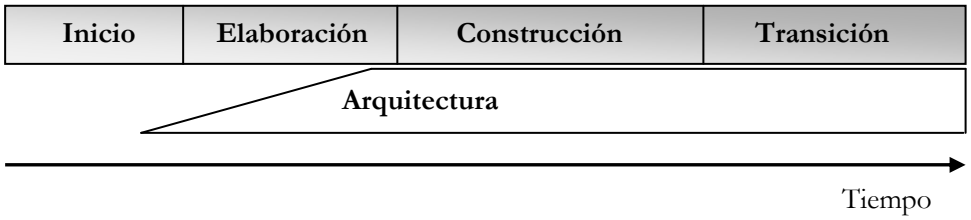


Figura 3.6 Evolución de la arquitectura del sistema.

3.4.2.3 Proceso iterativo e incremental

El equilibrio correcto entre los casos de uso y la arquitectura es algo muy parecido al equilibrio de la forma y la función en el desarrollo del producto, lo cual se consigue con el tiempo. Para esto, la estrategia que se propone en RUP es tener un proceso iterativo e incremental en donde el trabajo se divide en partes más pequeñas o mini proyectos. Permitiendo que el equilibrio entre casos de uso y arquitectura se vaya logrando durante cada mini proyecto, así durante todo el proceso de desarrollo. Cada mini proyecto se puede ver como una iteración (un recorrido más o menos completo a lo largo de todos los flujos de trabajo fundamentales) del cual se obtiene un incremento que produce un crecimiento en el producto.

Una iteración puede realizarse por medio de una cascada como se muestra en la figura 3.7. Se pasa por los flujos fundamentales (requisitos, análisis, diseño, implementación y pruebas), también existe una planificación de la iteración, un análisis de la iteración y algunas actividades específicas de la iteración. Al finalizar se realiza una integración de los resultados con lo obtenido de las iteraciones anteriores.

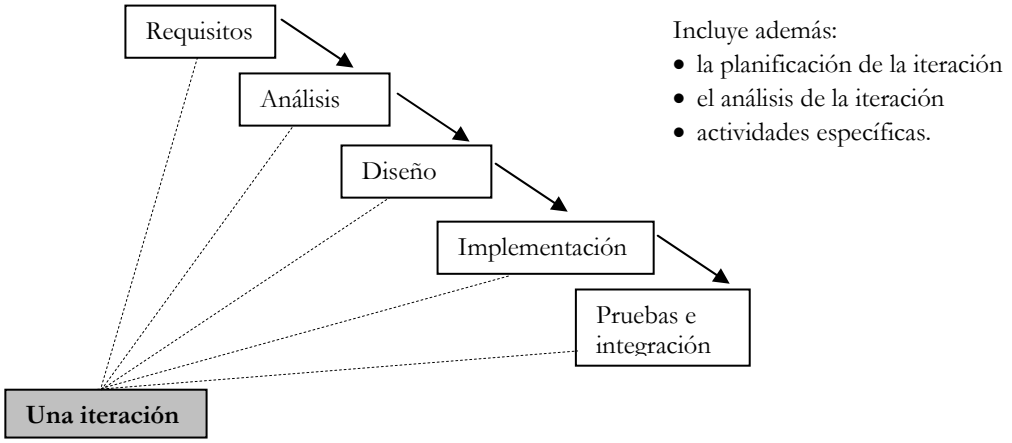


Figura 3.7 Una iteración RUP.

Toda la retroalimentación de la iteración pasada permite reajustar los objetivos para las siguientes iteraciones. Se continúa con esta dinámica hasta que se haya finalizado por completo con la versión actual del producto.

RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en los distintas actividades. En la figura 3.8 se muestra cómo varía el esfuerzo asociado a las disciplinas según la fase en la que se encuentre el proyecto RUP. Las fases son:

- *Inicio.* El objetivo en esta etapa es determinar la visión del proyecto.
- *Elaboración.* En esta etapa el objetivo es determinar la arquitectura óptima.
- *Construcción.* En esta etapa el objetivo es llevar a obtener la capacidad operacional inicial.
- *Transmisión.* El objetivo es llegar a obtener la liberación del proyecto.

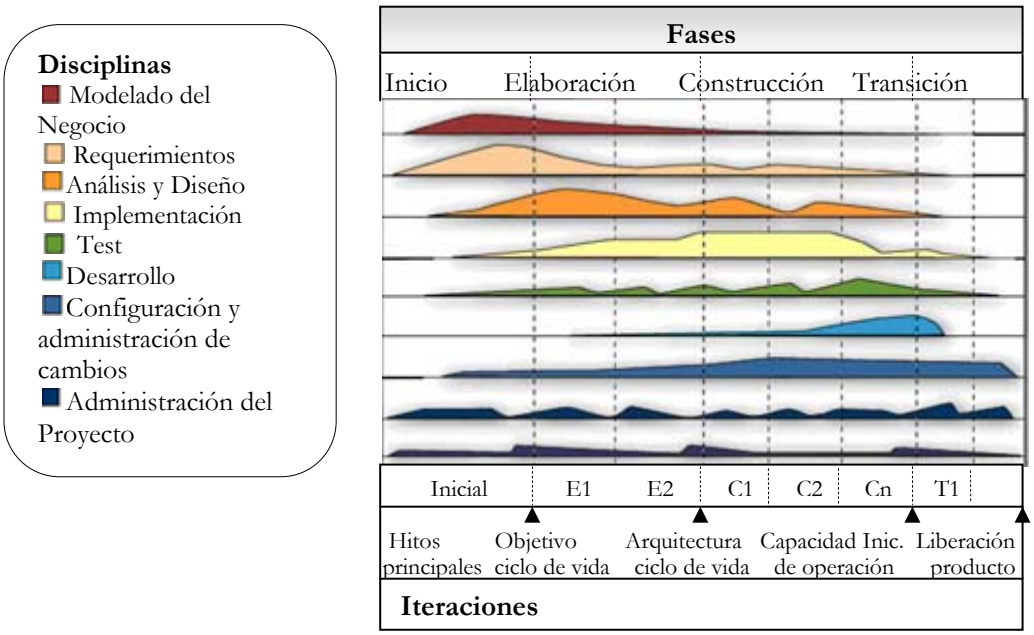


Figura 3.8 Fases que comprende RUP.

Las primeras iteraciones (en las fases de inicio y elaboración) se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y al establecimiento de una línea base de la arquitectura.

En la fase de construcción, se lleva a cabo la construcción del producto por medio de una serie de iteraciones. Para cada iteración se seleccionan algunos casos de uso, se refina su análisis y diseño y se procede a su implementación y pruebas. Se realiza una pequeña cascada para cada ciclo. Se realizan tantas iteraciones hasta que se termine la implementación de la nueva versión del producto.

En la fase de transición se pretende garantizar que se tiene un producto preparado para su entrega a la comunidad de usuarios.

El ciclo de vida que se desarrolla por cada iteración, como muestra la figura 3.9, es llevada bajo dos disciplinas:

Disciplinas de Desarrollo

- Ingeniería de negocios: Entendiendo las necesidades del negocio.
- Requerimientos: Trasladando las necesidades del negocio a un sistema automatizado.
- Análisis y diseño: Trasladando los requerimientos dentro de la arquitectura de software.
- Implementación: Creando software que se ajuste a la arquitectura y que tenga el comportamiento deseado.
- Pruebas: Asegurándose que el comportamiento requerido es el correcto y que todo los solicitado está presente.

Disciplinas de Soporte

- Configuración y administración del cambio: guardando todas las versiones del proyecto.
- Administrando el proyecto: administrando horarios y recursos.
- Ambiente: administrando el ambiente de desarrollo.
- Distribución: hacer todo lo necesario para la salida del proyecto

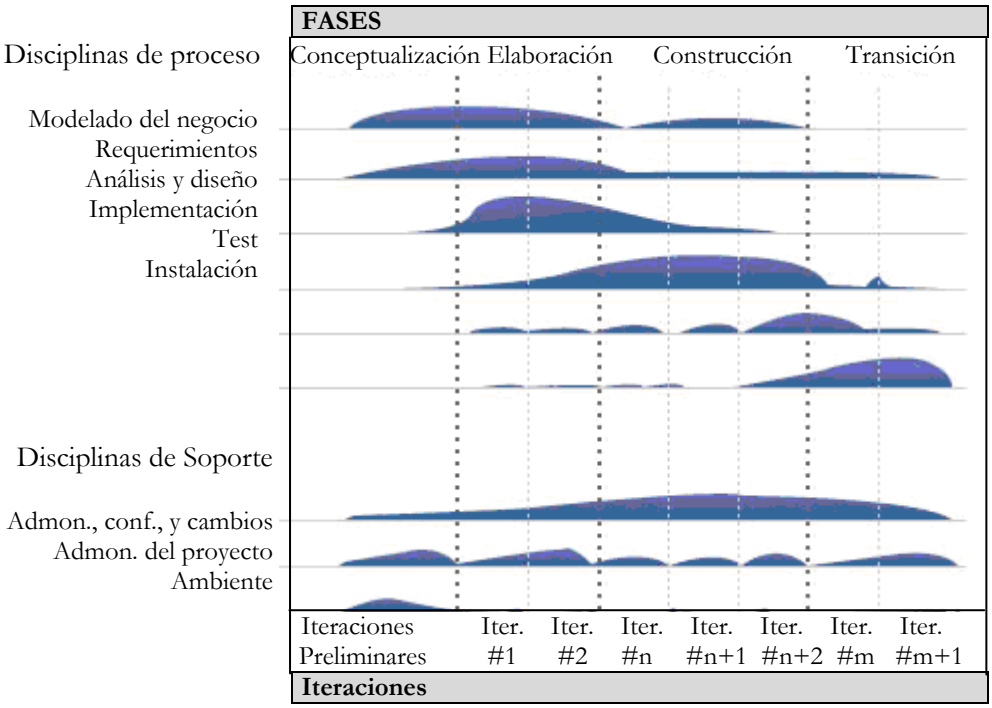


Figura 3.9 Disciplinas comprendidas en RUP.

Los elementos del RUP son:

- *Actividades*. Son los procesos que se llegan a determinar en cada iteración.
- *Trabajadores*. Son las personas o entes involucrados en cada proceso.
- *Artefactos*. Un artefacto puede ser un documento, un modelo, o un elemento de modelo.

Resumiendo, el ciclo de vida de RUP como se conoce al trazado de las actividades de desarrollo en el tiempo, está dividido en cuatro fases: inicial, elaboración, construcción y transición, que corresponden a los cuatro hitos principales de RUP: proyecto, arquitectura, versión β y liberación del producto.

3.5 Herramientas de desarrollo

Para el desarrollo de este portal, se hará uso de las siguientes herramientas:

- *Visual Basic .Net con Crystal Reports©*
- *Manejador de Base de Datos SQL Server 2000©*

Visual Basic .Net© con *Crystal Reports©*, a través de esta herramienta basada en la plataforma .NET, se llevará a cabo el desarrollo de todo el sistema, creación de pantallas, formularios, presentación y transacciones con la base de datos. Considerando que una de las necesidades identificadas es la creación de reportes estadísticos, también es necesario el uso de una herramienta capaz de crear

reportes desde datos almacenados en una base de datos, para ello se hará uso de *Crystal Reports*© ya que es el *reporteador* integrado en la plataforma .NET.

El almacenamiento y manipulación de datos recopilados a través del sistema en cuestión, se llevará a cabo a través del manejador de base de datos SQL Server 2000, desde el cual se procederá a la implementación del modelo entidad-relación recopilado y creado para este sistema.

Para la creación de los casos de uso que son requeridos en la metodología a desarrollar, se hará uso de UML.

La metodología a seguir es *Proceso Unificado Racional*, la cual se describió en la sección 3.3.

Todas las herramientas mencionadas, están disponibles para ser implementadas bajo plataforma operativa *Microsoft Windows*.

3.5.1 Sistema Operativo

Windows es la plataforma de sistema operativo determinado principalmente por el origen de las herramientas necesarias para el desarrollo del sistema mencionado.

La aplicación final desprendida de este proyecto se localiza montada bajo plataforma del servidor, y sistema operativo *Windows Server 2003 Standard Edition*©.

Windows Server 2003© es la pieza fundamental de *Microsoft Windows Server System*©. Esta basado en los fundamentos de *Windows 2000 Server*©, donde Microsoft intenta mejorar la fiabilidad, escalabilidad, rendimiento y facilidad de uso y administración, con el propósito de que en conjunto reduzcan el costo total de propiedad de las infraestructuras informáticas en toda clase de instalaciones.

Se describe como un sistema operativo:

- Multitarea.- Admite uno o varios usuarios simultáneamente.
- Multiusuario.- Varios usuarios pueden acceder simultáneamente a la misma computadora desde otras tantas terminales conectadas directamente al mismo.
- Extensible.- Permite aumentar y cambiar a medida que cambien los requisitos del mercado, es decir, fácilmente mantenible.
- Portable.- Permite la portabilidad de aplicaciones entre distintas computadoras.
- Fiable y Robusto.- Protección contra errores externos e internos.
- Compatibilidad.- Interfaz de usuario y de las aplicaciones (API) es compatible con las de otros sistemas operativos de Microsoft.
- Multiproceso y de fácil ampliación.- Las aplicaciones se pueden ejecutar en sistemas monoprocesador como multiprocesador
- Procesamiento distribuido.- Soporta facilidades de comunicaciones, distribución de tareas entre distintas computadoras de la red. Soporta computadoras locales como remotos.
- Sistema Abierto.- Cumple con las normas POSIX, estándar de sistemas abiertos basado en el sistema operativo UNIX.
- Rendimiento.- Garantiza la mayor rapidez y eficiencia posible.

Es un sistema operativo orientado a las comunicaciones, empleando una estructura cliente-servidor, con multiproceso simétrico, y diseñado orientado a objetos.

Sus características más importantes son:

1. Herencia
2. Permisos
3. Cuotas
4. Cifrado
5. Compresión de archivos
6. Permite montar dispositivos de almacenamiento sobre sistemas de archivos de otros dispositivos al estilo Unix.

A un alto nivel, proporciona compatibilidad con:

- Características de red avanzadas, como el servicio de autenticación de Internet (IAS), el puente de red y la conexión compartida a Internet (ICS).
- Multiproceso simétrico (SMP) de dos vías.
- Cuatro gigabytes (GB) de memoria RAM.

Contiene importantes herramientas de administración automática:

- Servicios de actualización de software (SUS) de Microsoft
- Asistentes de configuración de servidores para facilitar la automatización de la implementación
- Consola de administración de directivas de grupo (GPMC), permite que un mayor número de organizaciones utilicen mejor active directory y se beneficien de sus eficaces funciones de administración
- *Internet Information Services* 6.0 (IIS 6.0)
- Infraestructura de claves públicas (PKI)
- Kerberos
- Active directory es más rápido y robusto a través de conexiones de red área extensa (WAN)

Físicamente *Windows Server 2003, Edición Standard Edition*© soporta hasta 4 procesadores, hasta 4 GB de RAM e incluye servidor Web y servidor de terminales.

Una de las actividades del DSC-FI es la investigación, desarrollo y análisis de esquemas de seguridad en cómputo bajo distintas plataformas, entre ellas Windows. Así este proyecto permanecerá integrado no solo como herramienta auxiliar en la atención de incidentes sino como ambiente para análisis y estudio.

3.5.2 Servidores Web

Es un programa que implementa el protocolo HTTP, dicho protocolo está diseñado para transferir lo que llamamos hipertextos y páginas Web.

La plataforma *Windows Server 2003 Server Standard Edition*©, ofrece la posibilidad de funcionar como servidor Web y de Aplicaciones Web. La familia *Windows Server 2003*© dispone de IIS, un componente importante de la familia Windows, ya que con esta herramienta se puede habilitar el servicio de servidor Web en esta plataforma.

IIS es un conjunto de servicios para servidores usando *Microsoft Windows*©, especialmente usado en servidores Web. Proporciona una plataforma para desarrollar e implementar rápidamente servicios y aplicaciones Web usando ASP.NET, el cual forma parte del Framework .NET, mismo que analizaremos en la siguiente sección dedicada a Lenguajes de Programación.
Como servidor Web, posee las siguientes características principales:

- IIS 6.0 proporciona una arquitectura robusta y potente para crear y publicar aplicaciones Web.
- Lenguaje ASP.NET para desarrollar e implementar rápidamente servicios y aplicaciones Web.
- Escalabilidad por medio de Network Load Balancing (Equilibrio de carga de red).
- Interfaz de usuario Web basada en un navegador, que es fácil de usar.
- Los servicios de terminal en modo de administración permiten la flexibilidad de usar la administración de sobremesa y *Microsoft Management Console* (MMC) para control de ajuste preciso.
- La autoría y versiones distribuidas Web-based Distributed Authoring and Versioning (WebDAV) permiten a los usuarios publicar, gestionar y compartir información fácilmente a través de la Web.
- Asistentes adicionales para hacer que sea más fácil para los administradores configurar y gestionar autenticación segura y seguridad *Secure Sockets Layer* (SSL).

Una aplicación para Internet se puede describir en términos de relaciones *cliente –servidor*, donde el *cliente* es un explorador y el *servidor* es un servidor Web. Existen dos tipos de aplicaciones Web: s*ervicios Web XML* y f*ormularios Web*. Ambas se ejecutan en un servidor Web configurado como *Microsoft Internet Information Server©*.

Para crear una aplicación Web es necesario tener instalado .NET Framework SDK. Un editor de texto le permitirá escribir la aplicación, que podrá compilar y ejecutar desde la línea de órdenes.

3.5.3 Lenguajes de programación

Visual Studio .Net 2005©

La plataforma .NET es un amplio conjunto de bibliotecas de desarrollo que pueden ser utilizadas por otras aplicaciones para acelerar enormemente el desarrollo, ilustrada en la figura 3.10.

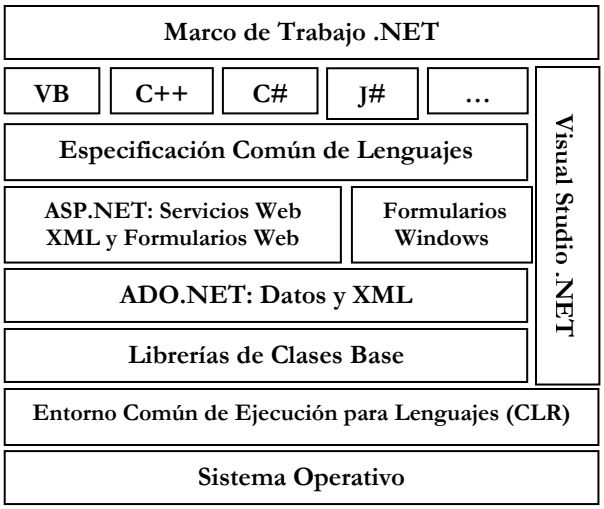


Figura 3.10 Arquitectura de la plataforma .NET.

La principal aportación de ASP.NET al mundo de la programación es que ha llevado a la Web el paradigma de la programación orientada a eventos propia de aplicaciones de escritorio, ofreciendo:

- Separación entre diseño y lógica.
- Componentes de interfaz de usuario, tanto estándar como de terceras empresas o propios.
- Diseñadores gráficos.

- Eventos.
- Estado.
- Enlazado a datos desde la interfaz.

ASP.NET facilita la generación de aplicaciones de las siguientes formas:

- Proporciona una abstracción de la interacción cliente-servidor Web tradicional, que permite programar aplicaciones utilizando herramientas de diseño rápido (RAD) y la programación orientada a objetos.
- Elimina los detalles de implementación relacionados con la separación de las partes cliente y servidor, presentando un modelo unificado que responde a los eventos de los clientes, en el código que se ejecuta en el servidor.
- Mantiene automáticamente el estado de la página, y de los controles que contiene, durante el ciclo de vida de la misma.

El proceso de una página Web (un formulario Web) ocurre en varias fases, que podemos resumir en: Iniciación, Carga del estado de vista, validación y Control de eventos. Durante estas fases se producen eventos, como *Init*, *Load* o *Unload*. El evento *Init* es el primero en el ciclo de vida de una página y se produce cuando la página es iniciada; para responder a este evento hay que sobrescribir el método *OnInit* heredado de la clase base. Una vez iniciada la página, reproduce el evento *Load* y cuando la página se descarga se produce el evento *Unload*.

3.6 Identificación de actores y roles

Una de las partes en donde el análisis actual enfoca su menor atención es la determinación de requerimientos, debido a ello es una de las etapas que mayor cantidad de errores introduce en un proyecto. Por tanto, resulta imprescindible mejorar los mecanismos en su mayoría informales y poco documentados de capturar las necesidades de un sistema.

Con base en la metodología RUP, se puede señalar que el objetivo de la etapa de determinación de requerimientos es identificar y documentar lo que es realmente necesario en un sistema, sin ambigüedades, para minimizar los riesgos de construir productos no solicitados.

Un *actor* es una entidad externa al sistema que de alguna forma participa en la historia del caso de uso. Un actor generalmente estimula al sistema con alguna entrada o recibe alguna respuesta por parte del mismo. En esta técnica los actores se denotan generalmente por el rol que cumplen, pero pueden ser cualquier clase de sistemas. Las clases de actores incluyen:

- Roles que la gente cumple
- Sistemas de cómputo
- Dispositivos eléctricos o mecánicos

Pueden existir dos tipos de actores, "actores participantes", que son aquellos que intervienen de alguna forma en el funcionamiento del caso de uso. Los "actores iniciadores" son aquellos que generan el estímulo inicial de un caso de uso, aunque pueden existir ocasiones en las que el estímulo no sea dado por un actor específico.

La identificación de actores y roles en un sistema se hace para:

- Delimitar el sistema de su ambiente externo
- Delinear qué y quién (actores) interactuarán con el sistema y qué funcionalidad es la que se espera de él.

- Capturar y definir un glosario de términos, necesarias para crear descripciones detalladas de la funcionalidad del sistema, permitiendo que el texto generado sea comprendido sin ambigüedades.

Para la identificación de actores, uno de los elementos a tener en cuenta es que los actores se comunican con el sistema a través de mensajes, así la clave para identificarlos correctamente, consiste en definir cuáles usuarios utilizarán el sistema y cuáles sistemas interactuarán con él. Cada categoría de usuarios o sistemas definidos será representada como un actor, denotando una separación entre las responsabilidades de los actores y las responsabilidades del sistema. Esta separación ayuda a definir los límites del mismo.

Posteriormente se analiza cuáles son las formas en las que cada actor actúa con el sistema, ya sea realizando solicitudes o enviando información al mismo, es decir se identifican los casos de uso, sección 3.7.1.

Actores y roles del sistema

El siguiente listado de actores y roles es resultado de la obtención a través del análisis de requerimientos, de las actividades y tareas de las personas que estarán interactuando directamente con el sistema.

Responsable del DSC-FI

Este actor dentro del sistema es fundamental para la coordinación de actividades, solución de problemas referentes a la seguridad en cómputo, generación de puntos de mejora, desarrollo e investigación de nuevos esquemas de seguridad, administración de recursos lógicos y físicos, y sobretodo representación del departamento ante eventualidades de emergencia.

Específicamente sus tareas se engloban como sigue:

- Monitorear el segmento correspondiente de red de la Facultad de Ingeniería.
- Priorizar y canalizar adecuadamente la atención de incidentes para una pronta y eficaz solución.
- Administrar y proveer los recursos lógicos, físicos y humanos del departamento.
- Solucionar incidentes de seguridad en caso de contingencia y/o emergencia.
- Capacitar y/o guiar al personal a su cargo sobre los esquemas de investigación y desarrollo de las últimas tendencias de la seguridad en cómputo.
- Obtener reportes estadísticos que generen indicadores cuantitativos para prevenir incidentes, detectar vulnerabilidades, y aportar mejoras al departamento.

Dentro del sistema son:

- Atender, dar seguimiento y solución a incidentes de seguridad en cómputo
- Verificar estatus de los incidentes.
- Generar oficios preventivos y correctivos.
- Obtener reportes gráficos del registro y la atención de incidentes.
- Obtener reportes estadísticos de los registros y la atención de incidentes.
- Registrar noticias y/o alertas.
- Dar o cambiar privilegios a usuarios del sistema.
- Acceso a la Base de Datos.

Operadores del DSC-FI

Son personal capacitado previamente a su ingreso, cuyas habilidades a desenvolver son el análisis, investigación y desarrollo de proyectos relacionados a la seguridad en cómputo. Sus tareas involucradas con el sistema son:

- Monitorear el segmento correspondiente, de la red Internet de la Facultad de Ingeniería.
- Dar atención, seguimiento y solución a un incidente de seguridad en cómputo.
- Registrar noticias y/o alertas nuevas.
- Verificar el buen funcionamiento del sistema.
- Modificaciones al sistema.

Personal contratado de la Facultad de Ingeniería

Se hace referencia a estos actores como el personal contratado directamente por la Facultad de Ingeniería, entre ellos personal de confianza, secretarias, profesores, investigadores, estudiantes, becarios, jefes de área, y que dispongan de un equipo de cómputo en dichas áreas.

Su interacción con el sistema está dada como sigue:

- Registrar incidentes de seguridad en cómputo.
- Mantenerse informado de las últimas vulnerabilidades y/o amenazas, así como de herramientas preventivas y correctivas, que le permitan estar mejor protegido.
- Consultar información referente al DSC-FI.

Comunidad de la Facultad de Ingeniería

Se refiere a toda la comunidad de la Facultad de Ingeniería, los cuales no pertenecen directamente a un área pero que ocupan o reciben los beneficios de la infraestructura de la red de datos.

Interactúan con el sistema como sigue:

- Consultar información de noticias y/o alertas, herramientas, y sobre el departamento.

Red de datos de UNICA

Es la red que albergará al portal Web del DSC-FI, la cual es administrada por DROS.

Sistema de detección de amenazas del DSC-FI

Representa el conjunto de sistemas, herramientas lógicas y físicas, así como el personal capacitado para detectar vulnerabilidades y amenazas dentro del segmento de red bajo resguardo directo del departamento.

Hasta este paso se ha detectado a los principales actores y los roles que desempeñan dentro del sistema a crear, más adelante se definirá gráficamente sus relaciones con los procesos y objetos del sistema.

3.7 Modelado de datos

Responde a una serie de preguntas específicas para cualquier aplicación de procesamiento de datos, tales como ¿cuáles son los objetos de datos primarios que va a procesar el sistema?, ¿cuál es la composición de cada objeto de datos y qué atributos describe el objeto?, ¿dónde residen los objetos? y ¿cuál es la relación entre los objetos y los procesos que los transforman?

Dadas las preguntas anteriores, surgen los métodos de modelado de datos para su respuesta. Los métodos de modelado de datos, hacen uso del DER descrito posteriormente, para identificar objetos de datos y sus relaciones mediante una notación gráfica. En el contexto del análisis estructurado, el DER define todos los datos que se introducen, se almacenan, se transforman y se producen dentro de una aplicación.

Sin embargo el modelado de datos estudia los datos independientemente del procesamiento que los transforma. Este se compone de tres piezas de información interrelacionadas: el objeto de datos, los atributos que describen el objeto de datos y la relación que conecta objetos de datos entre sí. La figura 3.11 ejemplifica la relación del usuario final y la computadora que posee donde se lleva a cabo el incidente de seguridad.

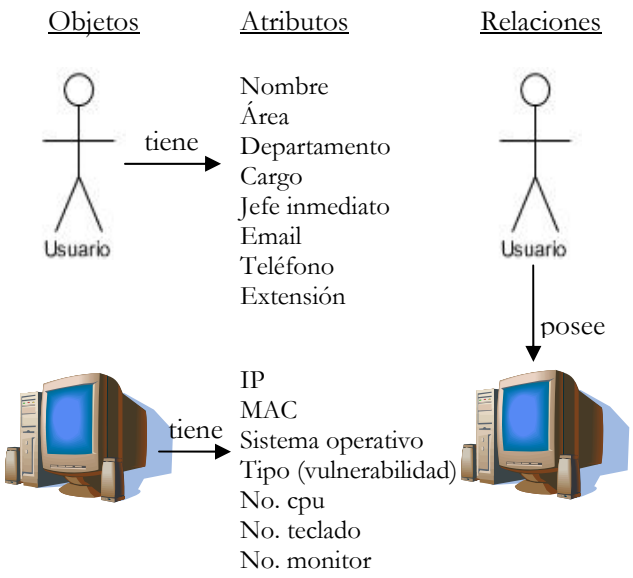


Figura 3.11 Ejemplo de objetos de datos, atributos y relaciones.

La técnica de diagrama de flujo de datos, es una representación gráfica que permite definir entradas, procedimientos y salidas de la información en la organización bajo estudio, permitiendo así comprender los procedimientos existentes con la finalidad de optimizarlos y reflejándolos en el sistema propuesto. Las figuras 3.12 y 3.13 ejemplifican los diagramas de flujos de datos para la publicación de noticias y registro de incidentes, respectivamente. El capítulo 4 hace uso de los diagramas de flujos de datos para la construcción de interfaz correspondiente, asimismo el anexo 1 los despliega gráficamente y el anexo 2 incluye parte de su transformación a código.

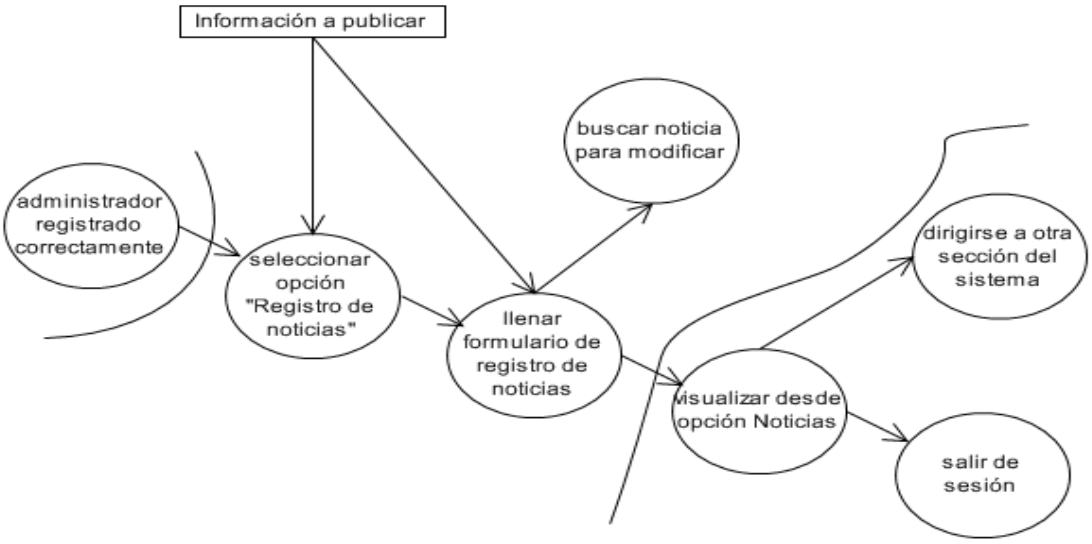


Figura 3.12 Ejemplo de diagrama de flujos para publicación de noticias en el sistema.

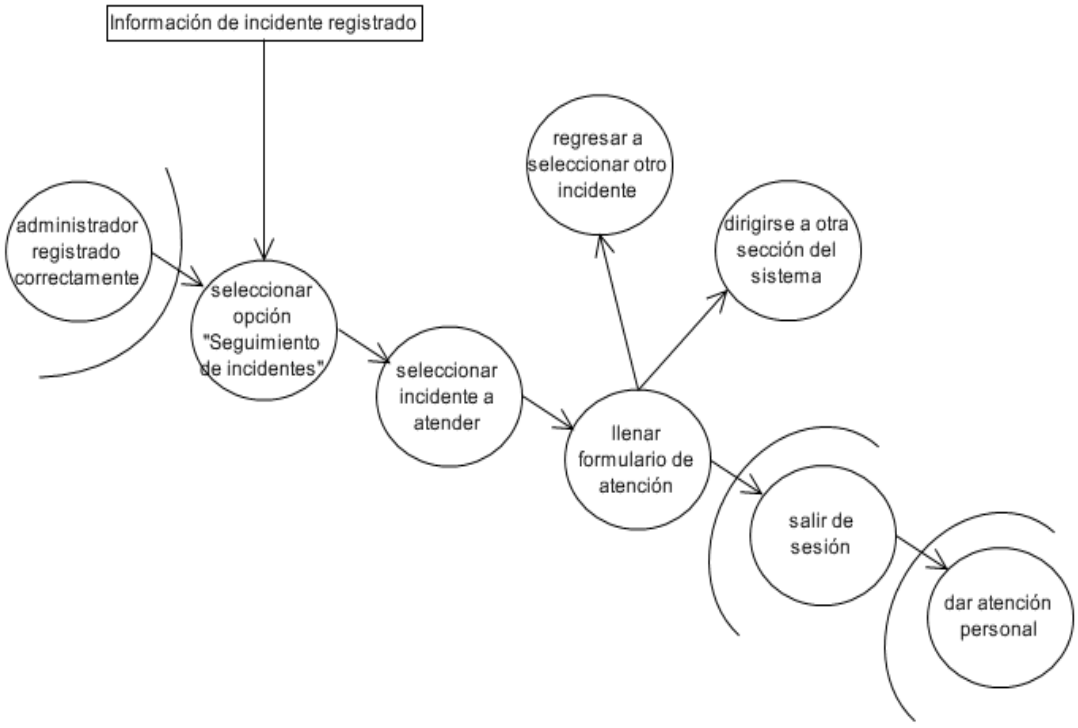


Figura 3.13 Ejemplo de diagrama de flujos para registro de incidentes en el sistema.

Los sistemas de datos y flujo de control son la base de representación de la transformación de datos y control. Al mismo tiempo, estos métodos son usados para crear un modelo funcional del software y proveerse de un mecanismo para dividir funciones. Posteriormente se debe crear un modelo de comportamiento usando el diagrama de transición de estados y un modelo de contenido de los datos con un diccionario de datos. Las especificaciones de los procesos y del control proporcionan una elaboración adicional de los detalles.

3.7.1 Diagrama de la Base de Datos

La pareja objeto-relación es la piedra angular del modelo de datos. Estas parejas se pueden representar gráficamente mediante el DER. Se identifica un conjunto de componentes primarios: objetos de datos, atributos, relaciones e indicadores de tipo. El propósito primario del DER es representar objetos de datos y sus relaciones.

El modelo de datos y el diagrama entidad-relación proporcionan una notación concisa para examinar dentro del contexto de una aplicación de procesamiento de datos. En esta parte se utiliza el enfoque de modelado de datos para el diseño de la base de datos.

Las figura 3.14, 3.15 y 3.16 muestran el diagrama entidad – relación, conceptualizado a partir de los requerimientos del cliente para la creación del sistema.

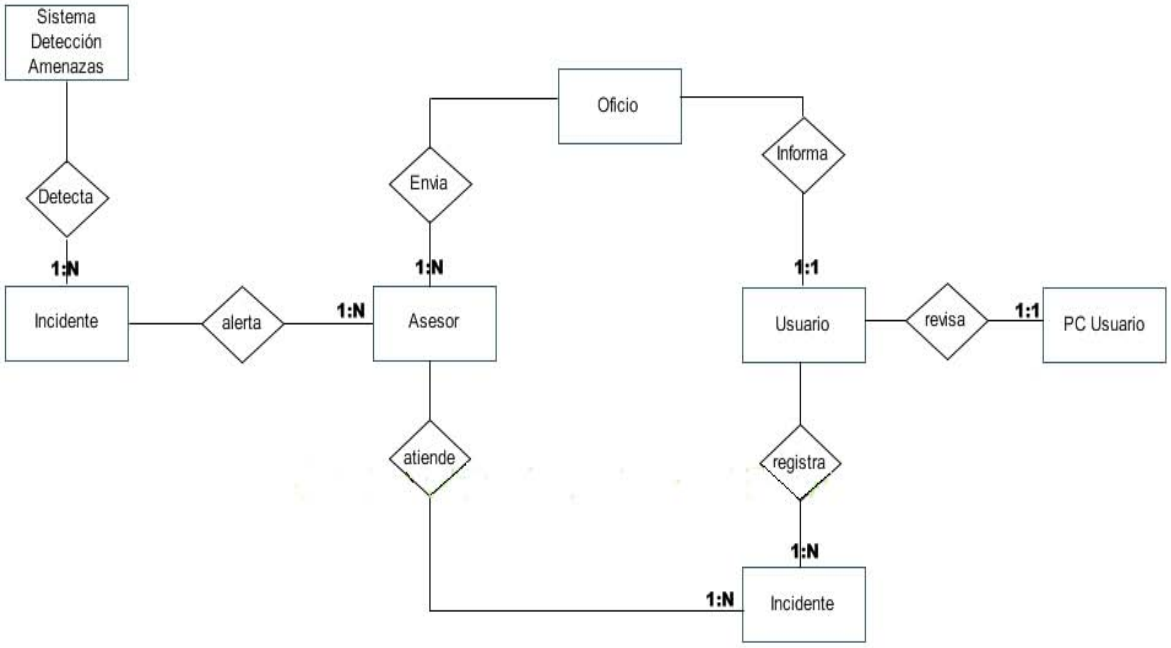


Figura 3.14 Diagrama entidad – relación de atención a incidente.

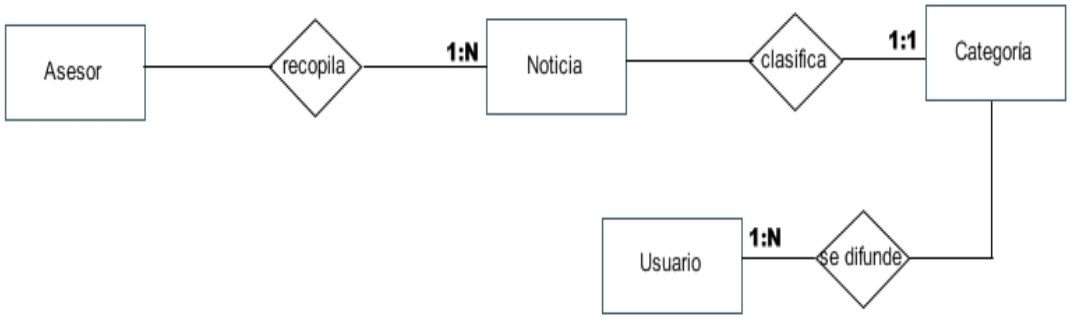


Figura 3.15 Diagrama entidad – relación de difusión de información.

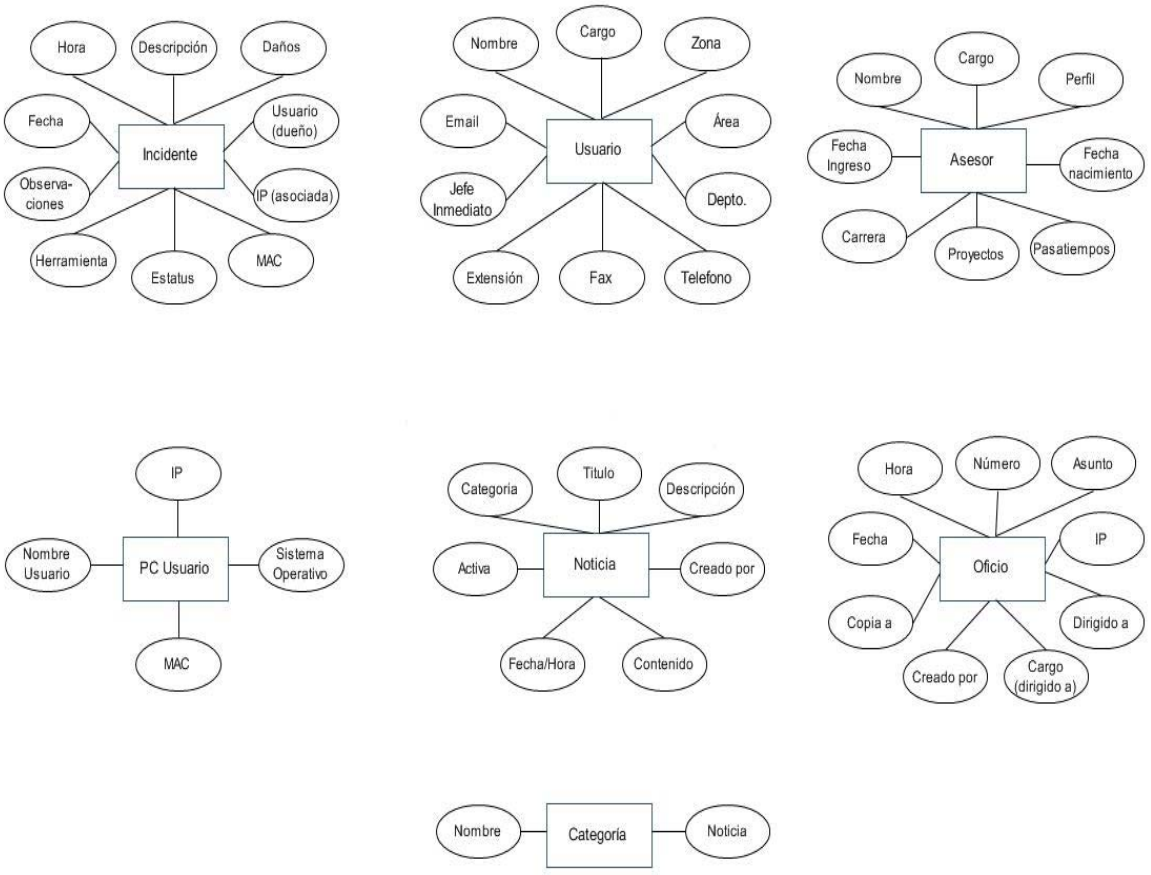


Figura 3.16 Diagrama entidad – relación de atributos de entidades.

3.7.2 Mapa del Sitio

Uno de los objetivos del diseño es producir un modelo o representación de una entidad que será constituida posteriormente.

Partiendo de los requerimientos recopilados en la etapa de análisis, se desplegará a través de la descripción del mapa del sitio la estructura del sistema a construir, es decir, se hará una especificación del diseño. La especificación está compuesta de los modelos de diseño que describen los datos, la arquitectura, interfaces y componentes.

Los mapas de sitio sirven como métodos para desplazarse por un sitio Web. Son una lista de vínculos a los archivos HTML del proyecto Web. A continuación se describe el mapa de sitio del portal desarrollado.

Inicio	Acerca del DSC-FI	Servicios	Atención a Incidentes
	Historia	Noticias/Boletines	Reporte de Incidente
	Proyectos	Análisis de Riesgos	Seguimiento Incidente
	Integrantes	Desarrollo Esquemas	Histórico
		Auditorias	Estadísticas
		Asesorías	
		Análisis Forense	

Capacitación	Descargas	Somos
Cursos	Manuales/Tutoriales	
	Software	

Donde:

Inicio

Liga del portal que permitirá desplazarse a la página de inicio del sistema, la cual incluye la información más relevante y de presentación accesible al usuario.

Acerca del DSC-FI

En esta sección el usuario podrá visualizar información referente a la historia del DSC-FI, los proyectos desarrollados y una breve descripción que le permitirá conocer a sus integrantes.

Historia
Desplegará la información de cómo, cuándo, por qué se crea el departamento.
Proyectos
Mostrará información de los proyectos más relevantes que lleva a cabo el departamento.
Integrantes
Se desplegarán datos de los integrantes del área, a fin de darlos a conocer y parte de su trayectoria.

Servicios

Esta liga del menú principal nos dirigirá a las secciones de noticias y la descripción de proyectos e información relacionada en con el departamento referente a análisis de riesgos, desarrollo de esquemas, auditorias, asesorías y análisis forense.

Noticias/Boletines
Mostrará las noticias referentes a las últimas amenazas, a fin de mantener informada a la comunidad de la Facultad de Ingeniería y prevenir así incidentes.
Análisis de Riesgos
Esta sección mostrará información referente a proyectos relacionados con el análisis de riesgos dentro del departamento y su vinculación con otras áreas.
Desarrollo de Esquemas
Información referente al desarrollo de esquemas en el área.
Auditorias
Información referente al desarrollo, planeación y ejecución de auditorías de seguridad en cómputo a las distintas áreas usuarias.
Asesorías
Temas de seguridad en cómputo, referentes a las dudas más frecuentes de la comunidad de la facultad.
Análisis Forense

Información referente al desarrollo de proyectos llevados a cabo o en vinculación con el departamento, referente a la rama del análisis forense de la seguridad en cómputo.

Atención de Incidentes

Dado que una parte esencial del área es atender y solucionar distintos tipos de incidentes de seguridad en cómputo, esta liga redirecciona las secciones de captura de incidente, seguimiento al incidente, un almacén de consulta de los incidentes registrados llamado histórico y una sección dedicada a la explotación estadística de los datos capturados.

Reporte de Incidente
Esta liga desplegará un formato de captura que le permitirá al usuario reportar uno o más incidentes desde una página Web, donde la única condición será pertenecer al segmento de red usuaria.

Seguimiento de Incidente
Mostrará información exclusiva a los integrantes del departamento que les permitirá llevar un control de seguimiento, solución y asignación de los incidentes reportados por los usuarios.

Histórico
Sección de consulta de los incidentes registrados, seguimiento y oficios generados. Será exclusiva a usuarios con los mayores privilegios dentro del área.

Estadísticas
Esta parte del sistema permitirá generar oficios preventivos y correctivos sobre incidentes de seguridad, así como reportes numéricos y gráficos en un período establecido de los incidentes registrados.

Capacitación

Liga que redireccionará a la información referente con los cursos internos y externos.

Cursos
Desplegado de cursos en cómputo y seguridad en cómputo ofrecidos por el departamento y UNICA.

Descargas

Esta liga permitirá dirigirse a las secciones de descarga de manuales, tutoriales y software auxiliar a la atención de incidentes.

Manuales/Tutoriales
Desplegado de documentos relacionados con la seguridad en cómputo.

Software
Desplegado de ligas a sitios con las herramientas más actuales para la atención de incidentes de cómputo.

Somos

Información general del DSC-FI y mecanismos de contacto.

3.7.3 Diccionario de Datos

El modelo de análisis conduce representaciones de objetos de datos, funciones y control. En cada representación los objetos de datos y/o elementos de control juegan un papel importante. Por tanto, es necesario proporcionar un enfoque organizado para representar las características de cada uno de ellos. Esto se realiza con el *diccionario de datos*

.

El diccionario de datos se ha propuesto como la gramática formal para describir el contenido de los objetos definidos durante el análisis estructurado. Es un listado organizado de todos los elementos de datos que son pertinentes para el sistema, con definiciones precisas y rigurosas que permiten que el usuario y el analista del sistema tengan la misma comprensión de las entradas, salidas, de los componentes de almacenes y de los cálculos intermedios.

De forma generalizada, debe contener la siguiente información:

- *Nombre*.- nombre principal del elemento de datos o de control, del almacén de datos, o de una entidad externa.
- *Alias*.- otros nombres usados para la primera entrada.
- *Dónde /cómo se usa*.- un listado de los procesos que usan el elemento de datos o de control y cómo lo usan (cómo entrada al proceso, salidas del proceso, almacén de datos)
- *Descripción del contenido*.- el contenido representado mediante una notación.
- *Información adicional*.- otra información sobre los tipos de datos, los valores implícitos, las restricciones o limitaciones.

La notación utilizada para desarrollar una descripción de contenido se indica en la tabla 3.2:

Construcción de datos	Notación	Significado
Agregación	=	está compuesto de
Secuencia	+	Y
Selección	[|]	uno u otro
Repetición	$\{\}^n$ ( ) *…*	n repeticiones de datos opcionales delimitadores de comentarios

Tabla 3.2 Notación de descripción de contenido del DD.

Para grandes sistemas de computadora el diccionario de datos crece rápidamente en tamaño y complejidad, por ello es necesario utilizar herramientas que faciliten su construcción evitando así su elaboración manualmente. La tabla 3.3 muestra un segmento del diccionario de datos para el sistema.

Diccionario de Datos	
Nombre	**Fecha**
Alias	Ninguno
dónde/cómo se usa	Registrar incidente (entrada) Crear Oficio (salida) Crear Reportes (salida) Crear Noticia (entrada) Búsquedas (salida)
Descripción	Es la fecha en que se da de alta un incidente en el sistema. Se registra en formato de dd/mm/yyyy, la cual es dada automáticamente por el servidor, es decir, dd= día del mes actual en número, mm= mes del año en curso en 2 dígitos, yyyy=año en curso en 4 dígitos.
Nombre	**Hora**
Alias	Ninguno
dónde/cómo se usa	Registrar incidente (entrada) Crear Noticia (entrada)
Descripción	Representa la hora del sistema y es proporcionada con el siguiente formato: hh+mm+ss, es decir, hh= hora actual en formato de veinticuatro horas, mm= minutos recorridos en la hora actual, y ss= segundos recorridos de la hora actual.
Nombre	**Descripción**

Alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada) Crear Noticia (salida) Búsquedas (salida)
Descripción	Es el resumen de una noticia|resumen de cómo sucede un incidente|frase al hacer una búsqueda específica.
Nombre	**Daños**
Alias	Ninguno
dónde/cómo se usa	Registro Incidente (entrada) Búsquedas (salida)
Descripción	Es una descripción de las situaciones que alteran los recursos lógicos o físicos de un equipo de cómputo debido a la ocurrencia de un incidente de seguridad en cómputo.
Nombre	**Usuario**
Alias	Ninguno
dónde/cómo se usa	Registro Incidente (entrada) Registro Usuarios DSC-FI (entrada)
Descripción	Nombre de la persona que registra el incidente de seguridad, puede o no corresponder con el nombre del dueño del equipo donde se detecto tal evento. Nombre de persona que pertenece al DSC-FI y requiere una cuenta de usuario para el sistema.
Nombre	**Dirección IP**
Alias	IP
dónde/cómo se usa	Registro Incidente (entrada) Creación Oficio (salida) Creación Reporte (salida) Búsquedas (salida)
Descripción	Es la dirección física de un equipo de cómputo perteneciente a la red de la FI, la cual se compone de 3 octetos. IP = [* número que sea distinto de 0] + *secuencia numérica de tamaño 2* + [símbolo punto .] + *cualquier secuencia numérica de tamaño 3* + [símbolo punto .] + *cualquier secuencia numérica de tamaño 3*
Nombre	**Dirección Física**
Alias	MAC
dónde/cómo se usa	Registro Incidente (entrada) Búsquedas (salida)
Descripción	Dirección física del equipo de cómputo que reporta un incidente de seguridad, se compone de 6 pares de combinaciones de letras y números. MAC = [*par de letras o números] + *símbolo -* + [*par de letras o números] + *símbolo -* + [*par de letras o números].
Nombre	**Estatus**
Alias	Ninguno
dónde/cómo se usa	Atención Incidente (entrada)
descripción	Es el estado en que se encuentra un incidente después de ser registrado. Estatus = [Atendido|Pendiente|Seguimiento|Detenido]
Nombre	**Herramienta**
Alias	Ninguno
dónde/cómo se usa	Atención Incidente (entrada) Búsquedas (salida)
Descripción	Es la herramienta lógica o física usada para la atención de un incidente por parte del asesor del DSC-FI.
Nombre	**Observaciones**
Alias	Ninguno

dónde/cómo se usa	Registro Incidente (entrada) Creación Noticia (salida) Creación Oficio (salida)
Descripción	Comentarios extra acerca de un incidente, noticia u oficio que deba mencionarse.

Tabla 3.3 Segmento del Diccionario de Datos del sistema.

El diccionario de datos completo del sistema se encuentra en el anexo 1.

3.8 Diseño detallado del sistema

El diseño del software es un proceso iterativo mediante el cual los requisitos se traducen en un plano para construir el software. Inicialmente, el plano representa una visión holística del software, es decir, el diseño se representa a u nivel alto de abstracción – un nivel que puede rastrearse directamente hasta conseguir el objetivo del sistema específico y según unos requisitos más detallados de comportamiento, funcionales y de datos – . A medida que ocurren las iteraciones del diseño, el refinamiento subsiguiente conduce a representaciones de diseño a niveles de abstracción mucho más bajos.

La calidad de la evolución del diseño se evalúa a lo largo de todo el proceso del diseño con una serie de revisiones técnicas formales. Existen tres características que sirven como guía para la evaluación de un buen diseño:

- deberá implementar todos los requisitos explícitos del modelo de análisis, y deberán ajustarse a todo los requerimientos implícitos que desea el cliente;
- deberá ser una guía legible y comprensible para aquellos que generan código y para aquellos que comprueban y consecuentemente, dan soporte al software;
- deberá proporcionar una imagen completa del software, enfrentándose a los dominios de comportamiento, funcionales y de datos desde una perspectiva de implementación.

El proceso del diseño del software fomenta el buen diseño a través de la aplicación de principios de diseño fundamentales de metodología sistemática y de una revisión cuidadosa.

3.8.1 Estructura del sistema

La arquitectura del software proporciona una visión global del sistema a construir. Describe la estructura y la organización de los componentes del software, sus propiedades y las conexiones entre ellos.

En la sección 3.5 se identifico a los actores y roles del sistema a construir, posteriormente se determino las relaciones entre los objetos identificados, también se dispone del diagrama entidad – relación, por lo que la siguiente herramienta a desarrollar son los casos de uso, a fin de determinar un diseño con mayor precisión de lo que se espera del sistema.

Principales casos de uso

Una vez determinada la identificación de los actores y roles en la etapa de "requerimientos", el siguiente paso para el diseño del sistema corresponde a la especificación de procesos, es decir, la identificación y construcción de los diagramas de caso de uso, agrupando y diferenciando responsabilidades de los actores involucrados.

En los siguientes diagramas se observa la descripción de los principales casos de uso, derivados de los procesos prioritarios y fundamentales en las actividades diarias que lleva a cabo el departamento.

- *Caso* 1: Atención de incidente (ver figura 3.17)

  Actores:
  - Sistema de detección de vulnerabilidades/amenazas
  - Asesor del departamento
  - Usuario del departamento
  - Computadora del usuario

  Procesos:
  - Clasificar incidente
  - Notificar área y usuario involucrados
  - Dar seguimiento a incidente
  - Resolver y documentar como histórico (incidente y solución)

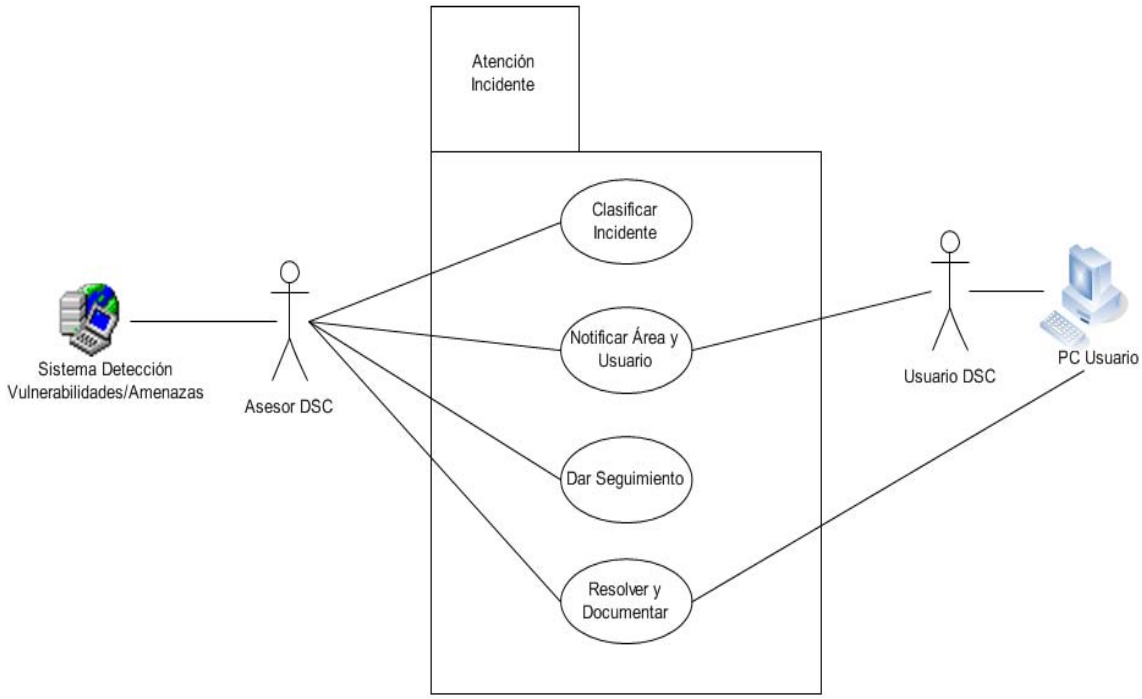


Figura 3.17 Diagrama caso de uso para atención de incidente.

- *Caso* 2: Difusión correctiva/preventiva (ver figura 3.18)

  Actores:
  - Sistema de detección de vulnerabilidades/amenazas
  - Asesor del departamento
  - Usuario del departamento

  Procesos:
  - Investigar amenazas
  - Determinar importancia de amenaza
  - Difundir y prevenir la amenaza

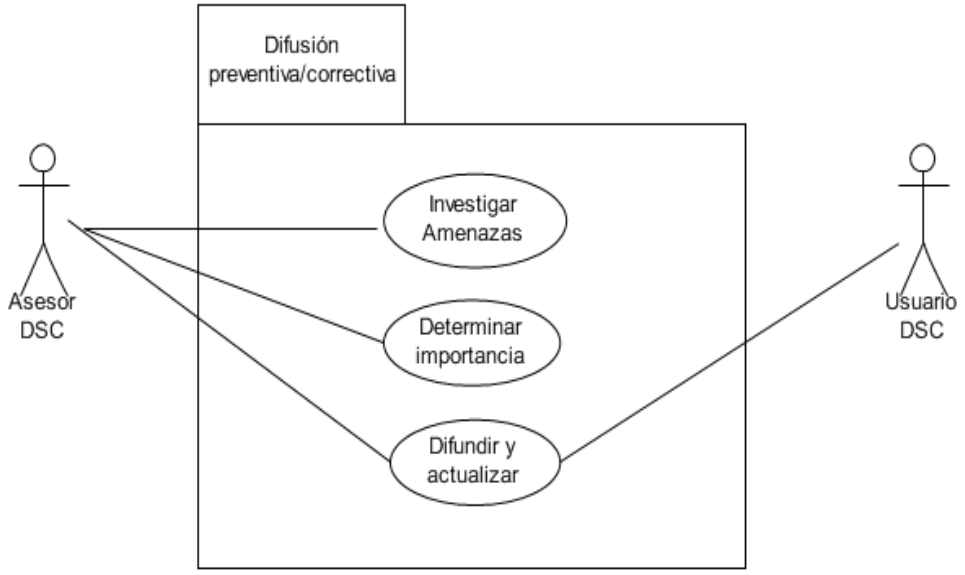


Figura 3.18 Diagrama caso de uso para difusión preventiva/correctiva.

- *Caso 3*: Análisis estadístico (ver figura 3.19)

  Actores:
  - Asesor del departamento

  Procesos:
  - Analizar información capturada
  - Generar reportes estadísticos

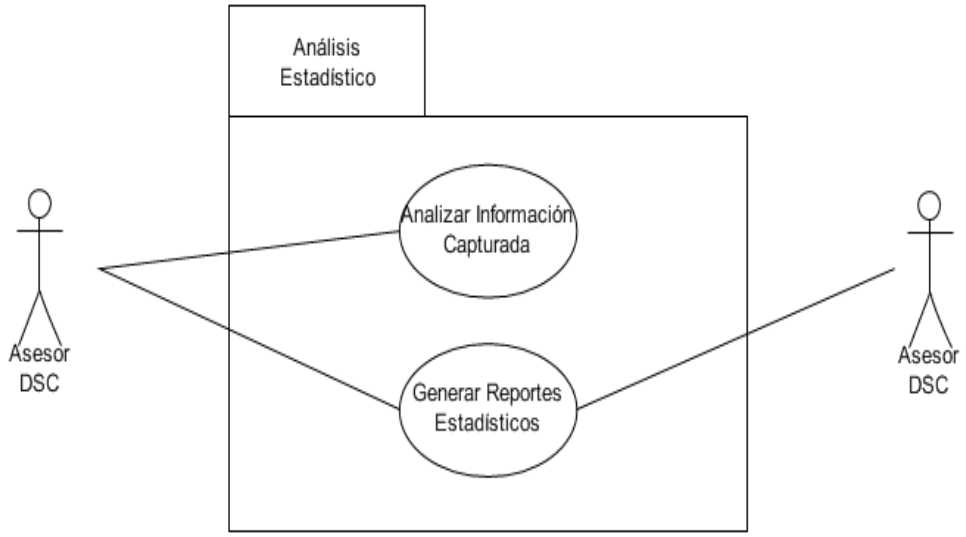


Figura 3.19 Diagrama caso de uso para análisis estadístico.

El diseño arquitectónico agrupa actividades de diseño que conducen a un modelo completo del diseño del software, la figura 3.20 muestra la estructura general del sistema a construir.

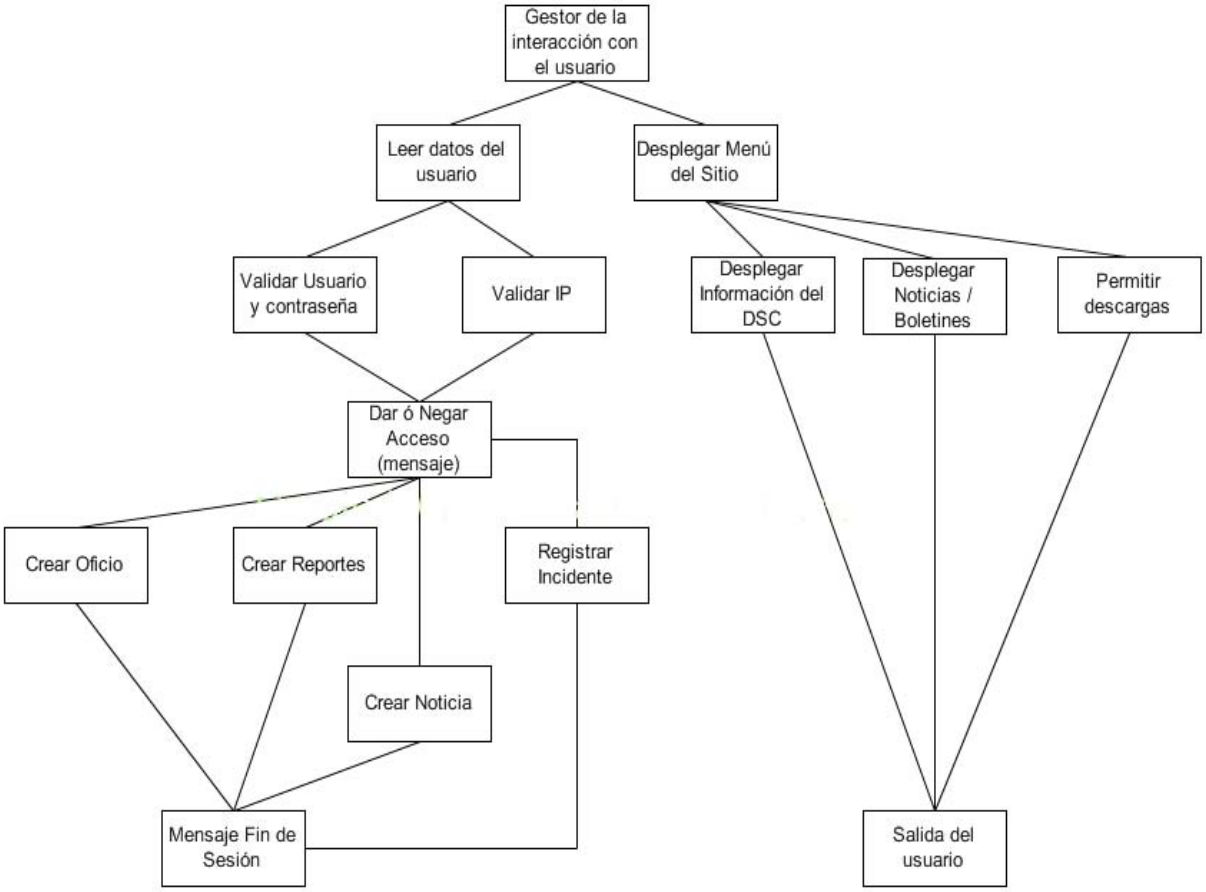


Figura 3.20 Estructura general del sistema a construir por niveles.

CAPÍTULO 4

DESARROLLO DEL SISTEMA

4.1 Objetivo

El presente capítulo describirá a nivel conceptual la arquitectura de cada interfaz involucrada en las necesidades del cliente a partir de los requerimientos recopilados en los capítulos 2 y 3 del presente trabajo.

4.2 Diseño de interfaz

Al disponer del diseño de datos se procederá a la derivación de una o más representaciones de la estructura arquitectónica del sistema. Los estilos arquitectónicos alternativos o patrones son analizados con el fin de obtener la estructura que mejor se ajuste a los requisitos del cliente y a las normas de calidad.

Para cada construcción de la interfaz correspondiente al sistema, el producto resultante estará revisado por el cliente y el constructor para clarificar, corregir, completar y dar consistencia acorde a los requisitos establecidos entre los mismos.

El diseño de la interfaz se centra en tres áreas de interés:

1. Diseño de la interfaz entre los componentes del software
2. El diseño de las interfaces entre el software y los otros productores y consumidores de información no humanos (entidades externas)
3. El diseño de la interfaz entre el hombre (usuario) y la computadora.

De acuerdo a Theo Mantel, creador de las tres reglas de oro para el diseño de la interfaz, las señala como sigue:

1. Dar el control al usuario
2. Reducir la carga de memoria del usuario
3. Consumir una interfaz consecuente

El proceso global para el diseño de la interfaz de usuario comienza con la creación de diferentes modelos de funcionamiento del sistema (la percepción desde fuera). Es entonces cuando se determinan las tareas orientadas al hombre y a la máquina que se requieren para lograr el funcionamiento del sistema; se tienen en consideración los temas de diseño que se aplican a todos

los diseños de interfaces; se utilizan herramientas para generar prototipos y por último para implementar el modelo de diseño, y evaluar la calidad del resultado.
El modelo de usuario representa el perfil de los usuarios finales del sistema. En general se pueden establecer las siguientes categorías de usuarios:

1. *Principiantes:* en general no tienen conocimientos sintácticos ni conocimientos semánticos de la utilización de la aplicación o del sistema.
2. *Usuarios esporádicos y con conocimiento:* poseen un conocimiento semántico razonable, pero una retención baja de la información necesaria para utilizar la interfaz.
3. *Usuarios frecuentes y con conocimiento:* poseen el conocimiento sintáctico y semántico suficiente para llegar al "síndrome del usuario avanzado", esto es, individuos que buscan breves interrupciones y modos abreviados de interacción.

4.2.1 Proceso de diseño de la interfaz de usuario

El proceso de diseño de las interfaces de usuario es iterativo y se puede representar mediante un modelo espiral, similar al presentado en el capítulo 3. Observe la figura 4.1, la cual describe las cuatro actividades de marco de trabajo que sigue el proceso de diseño, las cuales son:

1. Análisis y modelado de usuarios, tareas y entornos.
2. Diseño de la interfaz
3. Implementación de la interfaz
4. Validación de la interfaz

La espiral implica que cada una de las tareas anteriores aparecerán más de una vez, en donde a medida que se avanza por la espiral se representará la elaboración adicional de los requisitos y el diseño resultante. En la mayoría de los casos, la actividad de implementación implica la generación de prototipos – única forma práctica para validar lo que se ha diseñado-.

La actividad del análisis inicial se concentra en el perfil de los usuarios que van a interactuar con el sistema.

Se registran el nivel de conocimiento, la comprensión del negocio y la receptividad general del nuevo sistema, y se definen diferentes categorías de usuarios.

Definidos los requisitos generales, se lleva a cabo un análisis más detallado de las tareas. Se identifican, describen y elaboran las tareas que lleva a cabo el usuario para conseguir los objetivos.

El análisis del entorno de usuario se centra en el entorno del trabajo físico. Y se formula preguntas como: ¿dónde se ubicará físicamente la interfaz?, ¿dónde se situará el usuario?, ¿se adapta bien el hardware a las limitaciones de luz, espacio o ruido?

El objetivo del diseño de la interfaz es definir un conjunto de objetos y acciones de interfaz que posibiliten al usuario llevar a cabo todas las tareas definidas de forma que cumplan todos los objetivos de uso definidos por el sistema.

La actividad de implementación comienza normalmente con la creación de un prototipo que permita evaluar los escenarios de utilización.

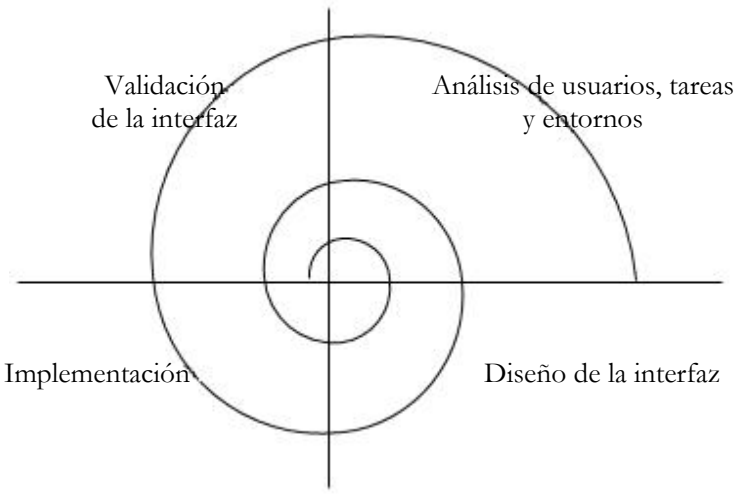


Figura 4.1 El proceso de diseño de la interfaz de usuario.

De acuerdo a la recopilación de información, análisis y diseño de los datos, y propuesta de prototipos de diseño de interfaz, en las siguientes secciones se definirá cómo estará constituida cada interfaz dentro del sistema a desarrollar para el departamento.

4.3 Interfaz administrativa

Está basada en las necesidades de facilitar al usuario la modificación de cada sección del mismo, la actualización de información y la explotación de la información registrada de forma exclusiva a los usuarios con mayores privilegios dentro del departamento.

Se describirá el escenario como el caso a estudiar en particular, es decir, de cómo se usará el sistema para así describir los elementos que comprenderá la interfaz administrativa.

El diseño de pantalla es un proceso interactivo en donde se lleva a cabo el diseño gráfico y la colocación de los iconos, la definición del texto descriptivo en pantalla, la especificación y títulos para las ventanas, y la definición de los elementos del menú principales y secundarios.

Escenario: El usuario (asesor del DSC-FI) desea ingresar al sistema portal del departamento para realizar una tarea de administrador. Al sistema puede acceder desde cualquier navegador a Internet desde cualquier computadora dentro de la red de la Facultad de Ingeniería. Después de realizar la tarea desea obtener los resultados correspondientes.

Las tareas de administrador que puede llevar a cabo el usuario son:

a) publicar una noticia,
b) generar oficios preventivos y correctivos,
c) generar reportes estadísticos sobre los incidentes de seguridad registrados,
d) generar gráficas sobre los incidentes de seguridad registrados,
e) dar seguimiento a los incidentes registrados, y
f) realizar búsquedas particulares de oficios, personal e incidentes.

Para acceder al sistema el usuario proporciona un nombre de usuario y una contraseña. Con esto se definen los niveles de acceso y se proporciona seguridad, dado que no todos los asesores del departamento disponen del mismo nivel de acceso, esto se observa en la figura 4.2 de forma conceptual y gráficamente en la figura 4.3. Si cumple el nivel requerido se dirige a la sección de su interés donde realiza y comprueba la tarea a ejecutar.

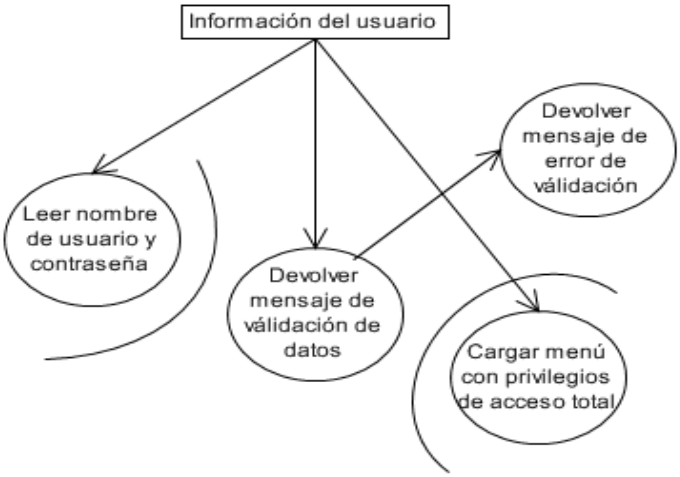


Figura 4.2 Análisis de datos para ingreso de usuarios.

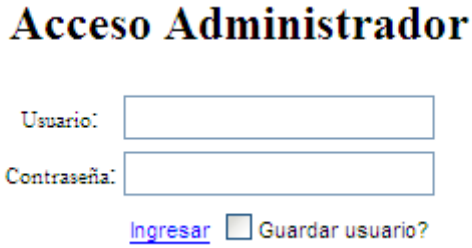


Figura 4.3 Pantalla de ingreso de usuarios con nivel administrador.

4.3.1 Relación usuario – interfaz administrativa

4.3.1.1 Publicar noticia

Después de ingresar al sistema, el asesor se dirige a la página a*dministración de noticias* donde ingresa el título, la descripción, el contenido y si debe estar activa o no. Posteriormente se dirige a noticias y si opto por activa, observará la noticia en la lista de noticias desde la opción del menú noticias, este proceso se describe conceptualmente con el diagrama de flujo de datos mostrado en la figura 4.4 y la transformación de los datos en el formulario mostrado en la figura 4.5.

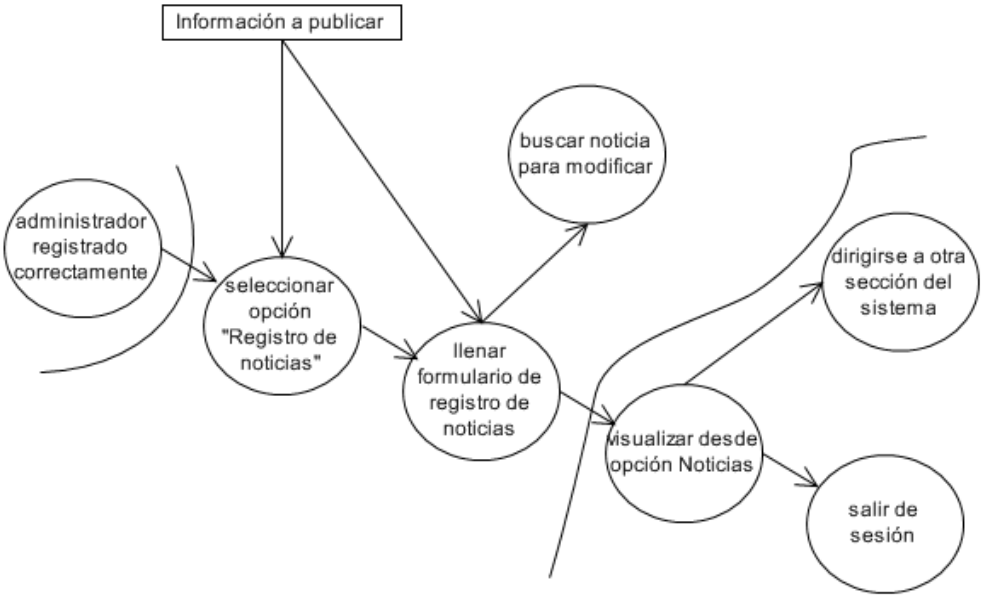


Figura 4.4 Análisis de datos para registro de noticias.

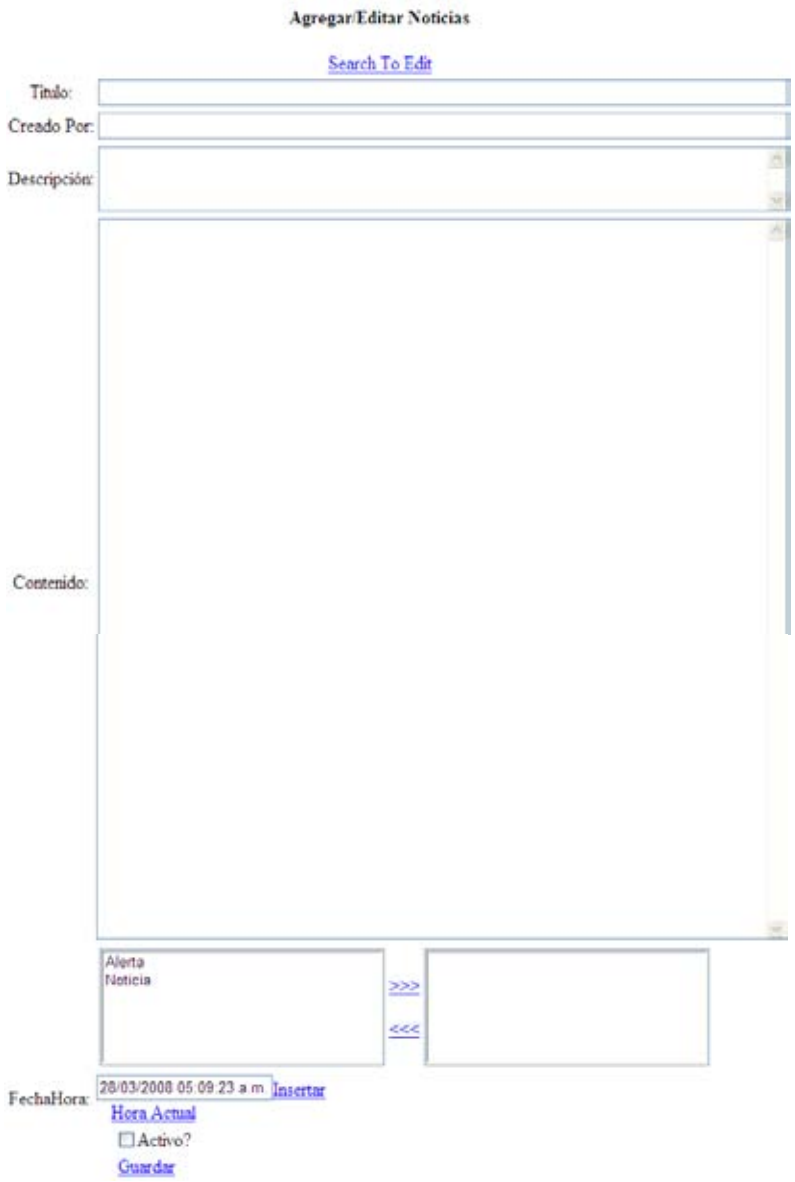


Figura 4.5 Pantalla de registro de noticias.

El código de esta sección se muestra en el anexo 2, donde se especifica cómo se registra una noticia, su despliegue y la creación de categorías para su agrupación correspondiente. Se añade además parte el código utilizado para el uso de reportes en C*rystal Reports*©.

4.3.1.2 Seguimiento a incidentes

Posterior al ingreso al sistema el usuario se dirige a la página s*eguimiento de incidente* donde observa una lista de los incidentes registrados, y de acuerdo a su horario y/o conocimientos da atención y seguimiento al incidente, toma los datos del equipo involucrado, área y persona a cargo y acude personalmente a prestar la atención solicitada, tal como lo muestra el diagrama de flujo de datos mostrado en la figura 4.6 obtenido de los requerimientos solicitados por el cliente.

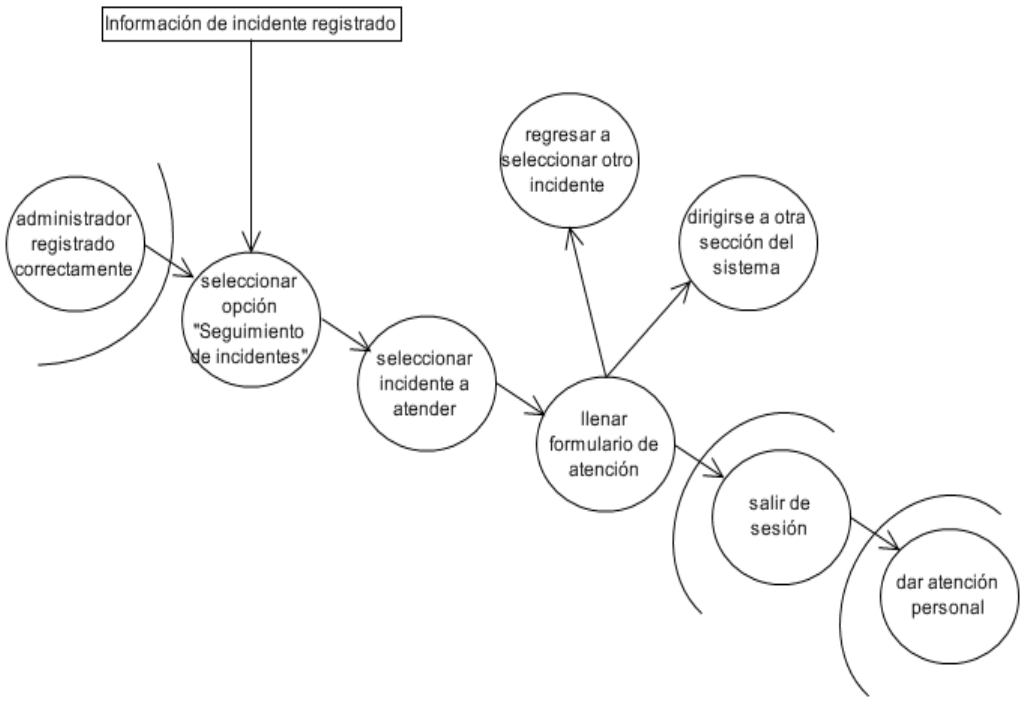


Figura 4.6 Pantalla de atención de incidentes.

La figura 4.7 muestra la interpretación a nivel diseño de los requerimientos recopilados para el seguimiento de incidentes de seguridad en cómputo.

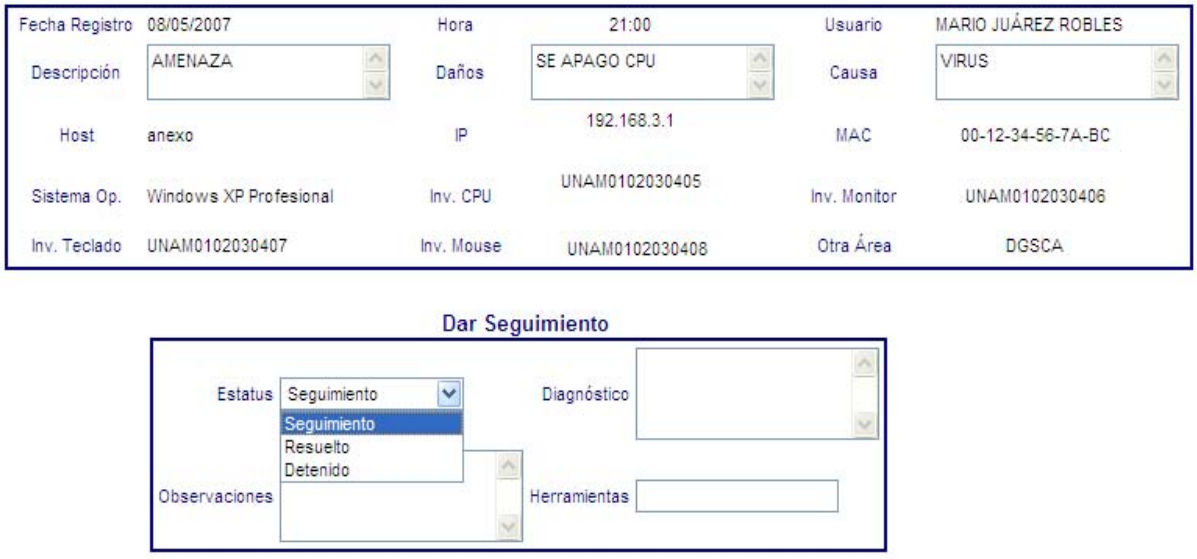


Figura 4.7 Pantalla de seguimiento de incidentes.

4.3.1.3 Realizar búsquedas

Después de ingresar al sistema el usuario se dirige a la página *histórico* y de acuerdo a sus necesidades busca por personal, oficio o incidente y en cada una puede hacer búsquedas particulares, la interpretación a nivel de datos se muestra en la figura 4.8.

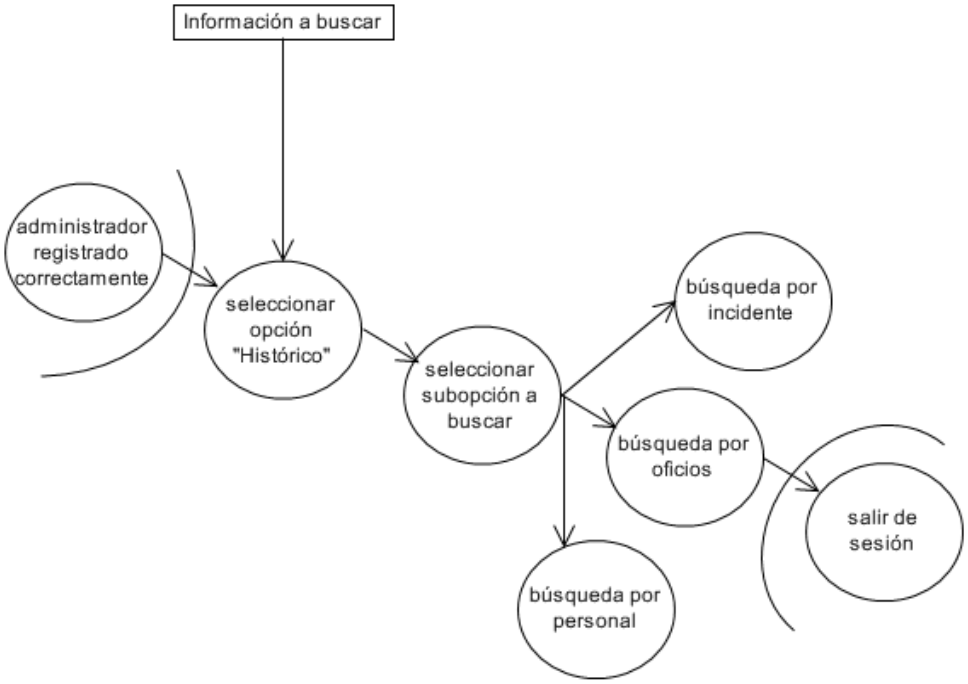


Figura 4.8 Análisis de datos de búsquedas en el sistema.

Las figuras 4.9, figura 4.10 y figura 4.11 ilustran a nivel diseño, las pantallas correspondientes a la clasificación de la información almacenada y cuya función es mantener un histórico sobre los oficios generados, el personal que registra un incidente de seguridad en cómputo y los incidentes de seguridad en cómputo reportados.

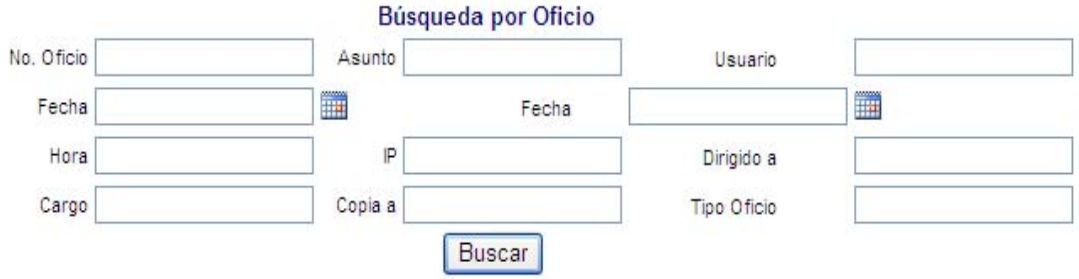


Figura 4.9 Pantalla de búsqueda por oficio.

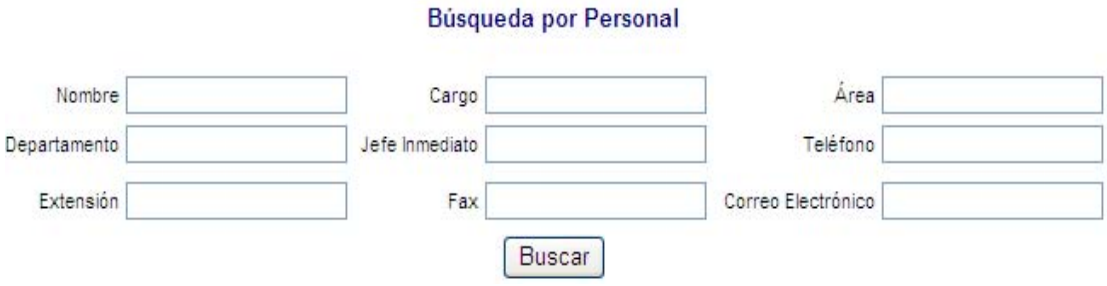


Figura 4.10 Pantalla de búsqueda por personal.

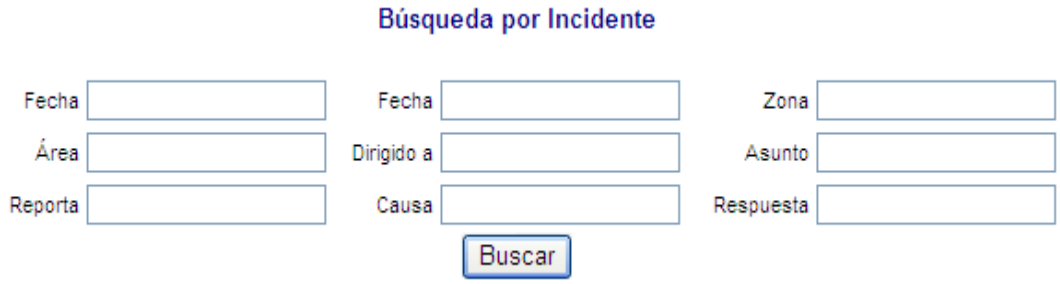


Figura 4.11 Pantalla de búsqueda por incidente.

Para la modificación y actualización del contenido de las páginas Web que conforman el sistema se lleva a cabo a través del siguiente formulario, donde se hace referencia a la página a actualizar y el texto a agregar o modificar, tal como se observa en la figura 4.12 a nivel de diseño de pantalla.

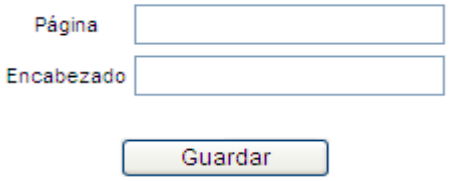


Figura 4.12 Pantalla de encabezados de páginas.

4.4 Interfaz Informativa

Se describen los componentes de diseño que forman la pantalla de interfaz informativa, la cual pretende desplegar la información correspondiente a las actividades que desarrolla el departamento, la información que ayudará a los usuarios finales del sistema a mantenerse protegidos ante los distintos y cada vez más frecuentes y sofisticados tipos de incidentes de seguridad en cómputo, e información acerca del personal y contacto directo del departamento.

Escenario: El DSC-FI requiere difundir a través de un sistema en Web, información sobre los últimos tipos de incidentes, medidas correctivas y preventivas, así como los proyectos que desarrolla y su vinculación con otras áreas, las acciones que lleva a cabo para la prevención y mejora en las distintas áreas de las que se compone y analiza.

Principalmente necesita una sección de Noticias donde se desplieguen artículos relacionados con la seguridad en cómputo y se puedan clasificar de acuerdo a su contenido en alertas, noticias o boletines. El diseño de datos es mostrado en la figura 4.13 y a nivel de pantalla en la figura 4.14.

La forma en cómo nace el departamento, quienes lo integran, qué proyectos realiza y cómo contactarse con él, son parte fundamental de la información a desplegar.

De acuerdo al mapa de sitio obtenido en el capítulo anterior el departamento también solicita se describan las actividades que realiza cada una de las distintas áreas de estudio que comprende: análisis de riesgos, auditorias, desarrollo de esquemas, análisis forense, asesorías, y su importancia en el contexto actual.

Asimismo el departamento y las áreas con las que está vinculado necesitan informar acerca de las herramientas que ofrecen para conocer e involucrarse más con el tema de la seguridad en cómputo, esto es: cursos, software, manuales, tutoriales.

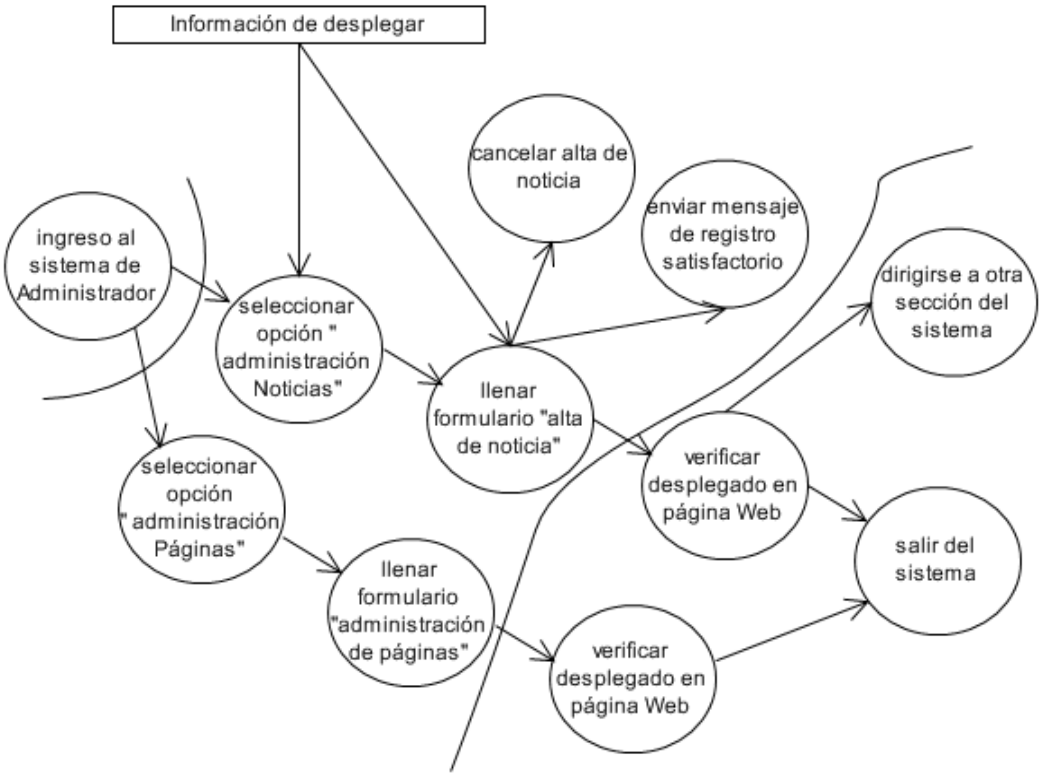


Figura 4.13 Análisis de datos sobre desplegado de información.

Figura 4.14 Pantalla de información de noticias sobre seguridad en cómputo.

4.5 Interfaz Estadística

A partir de los requerimientos del cliente, se detecto la necesidad de analizar la información que registra el sistema con la finalidad de disponer de indicadores cuantitativos que reflejen los resultados obtenidos en un periodo específico, y así detectar puntos de mejora dentro de la cobertura del departamento.

De lo anterior esta interfaz permite generar reportes que contabilizan los incidentes de seguridad registrados numérica o gráficamente.

Escenario: El usuario (asesor DSC-FI) desea obtener resultados cuantitativos de las tareas que administra el sistema, tales como el registro y atención de incidentes, generación de oficios. Para ello desea a través del sistema ingresar datos específicos (periodo, tipo de incidente, sistema operativo involucrado, estatus del incidente de seguridad) que generen un reporte en términos numéricos o gráficos de los mismos.

El sistema solicita datos particulares para generar un reporte numérico de los incidentes de seguridad registrados, tales son: periodo, tipo de incidente, sistema operativo involucrado y estatus del incidente, al pulsar un botón de *generar reporte* automáticamente se genera en *Crystal Reports* un desplegado numérico de los datos solicitados listo para imprimir o exportar.

Las gráficas desplegadas por el sistema son: *incidentes por área*, *incidentes por zona* y por *tipo de incidente*. Las cuales son calculadas automáticamente al ingresar a la sección Estadísticas del sistema y seleccionar la opción Gráficas.

4.5.1 Relación usuario – interfaz estadística

4.5.1.1 Generar oficios preventivos y correctivos

Dentro del departamento existen sistemas de detección de amenazas y/o vulnerabilidades que funcionan a toda hora, estos son los encargados de informar al administrador qué direcciones IP presentan irregularidades.

Con esta información el usuario ingresa al portal y se dirige a la página *Estadísticas*, donde selecciona la opción de *Oficios* y genera un oficio correctivo o preventivo dependiendo el tipo de incidente detectado ingresando:

- tipo de oficio
- número de oficio
- asunto
- dirigido a
- nombre del responsable del área
- dirección IP
- tipo de incidente
- copia a

Posteriormente selecciona la opción *generar oficio,* y con ello se despliega el oficio correspondiente en C*rystal Reports*©, quedando listo para importar o imprimir.

La representación de la generación de oficios a nivel de datos es mostrada en la figura 4.15.

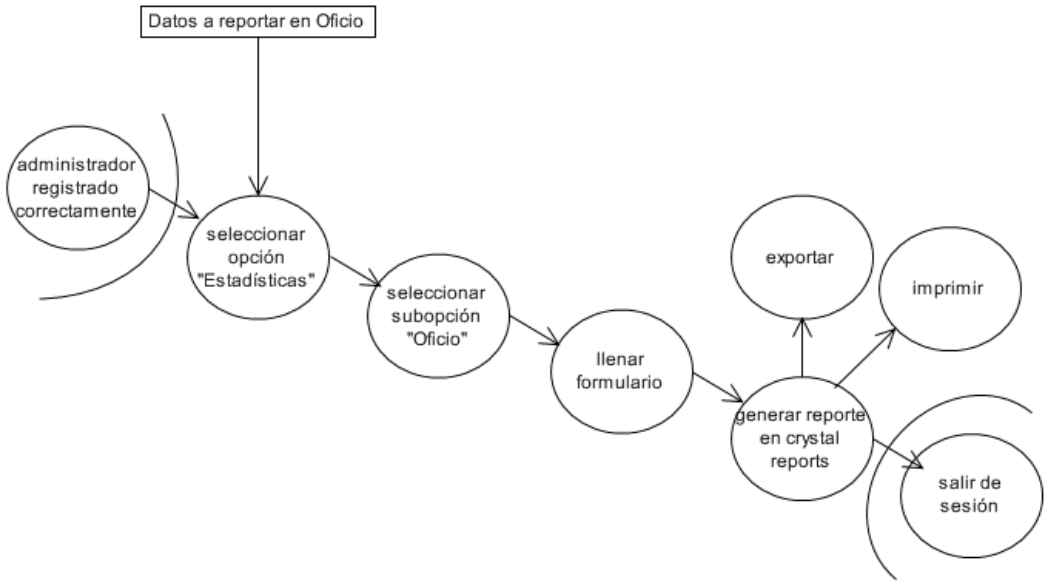


Figura 4.15 Análisis de datos para generación de oficios.

La figura 4.16 muestra la representación de la generación de oficios a nivel interfaz.

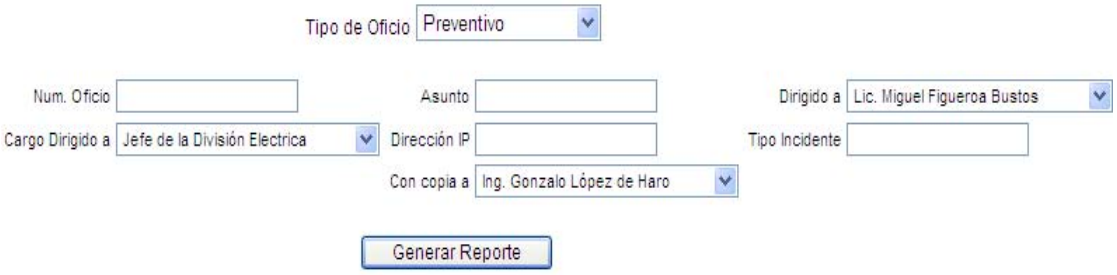


Figura 4.16 Pantalla para generación de oficios.

4.5.1.2 Generar reportes estadísticos

El usuario se dirige a la página *estadísticas*, y genera reporte por incidente ingresando periodo a cuantificar, sistema operativo y estatus del incidente. Al elegir *generar reporte*, se carga el reporte en C*rystal* y queda listo para su importación a formato pdf, excel o txt, o para su impresión.

De acuerdo al mismo análisis de datos para generación de oficios, se concluyo la pantalla para generar un reporte estadístico sobre los incidentes de seguridad en cómputo registrados en el sistema mostrado en la figura 4.17.

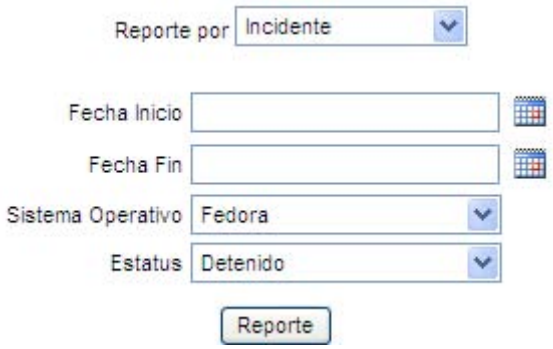


Figura 4.17 Pantalla de generación de reporte estadístico.

4.5.1.3 Generar gráficas

El usuario se dirige a la página e*stadísticas*, y selecciona la opción *gráficas* donde automáticamente se genera el reporte con las gráficas de incidentes por zona, por tipo de incidente y por sistema operativo involucrado en formato de C*rystal Reports*© y queda listo para su importación a formato pdf, excel o txt, o para su impresión.

La interpretación a nivel diseño está basada en el despliegue de las gráficas desde la selección del menú *gráficas,* las cuales son incidentes por zona, por área y por tipo de incidente, como se muestra en las figuras 4.18 y 4.19.

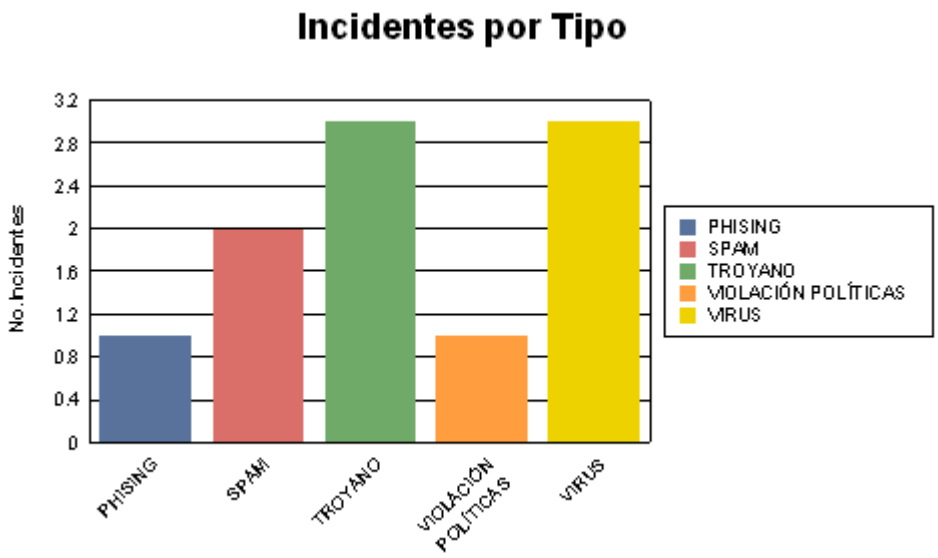


Figura 4.18 Pantalla de generación de gráfica incidentes por tipo.

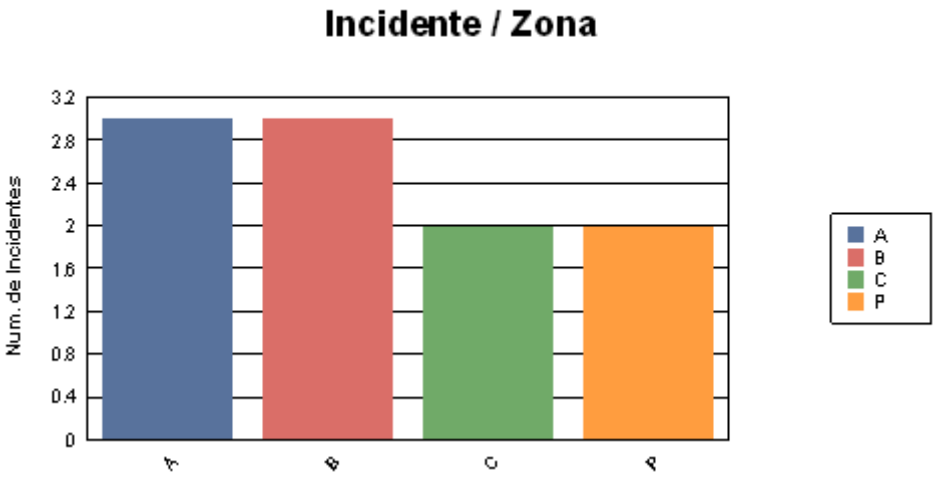


Figura 4.19 Pantalla de generación de gráfica incidentes por zona.

4.6 Interfaz atención a incidentes

Esta interfaz comprende la parte fundamental de la creación del sistema, al permitir a cualquier persona dentro de la cobertura del DSC-FI registrar un incidente de seguridad, evitando el largo proceso de envío y recepción de oficios donde por su formato no siempre se puede detallar el acontecimiento, y por ende se alarga la atención y solución de dicho incidente.

Escenario: El usuario final desea reportar al DSC-FI sobre un incidente de seguridad ocurrido en un equipo de cómputo bajo su resguardo por lo que recopila toda la información requerida y previamente establecida por el departamento, e inmediatamente aísla su equipo de la red compartida.

El usuario final ingresa al portal del departamento desde cualquier navegador Web, selecciona la opción del menú principal *Reporte de Incidentes* y empieza a llenar el formulario de atención a incidentes donde debe especificar datos personales en la primera sección y en la segunda datos referentes al incidente.

Puede cancelar o registrar dicho incidente, al registrar el incidente el sistema reflejará inmediatamente en su sección de *seguimiento de incidente* dicho incidente y aparecerá sin estatus, a partir de ello el personal del departamento debe contactar al usuario final que reporta el incidente y acudir personalmente para dar atención al mismo.

Es importante señalar que incluso el asesor del DSC-FI puede consultar el sistema desde cualquier equipo con Internet dentro de la cobertura del departamento y así proporcionar una atención oportuna.

Se puede observar el análisis de datos en la figura 4.20 y su representación de diseño en la figura 4.21.

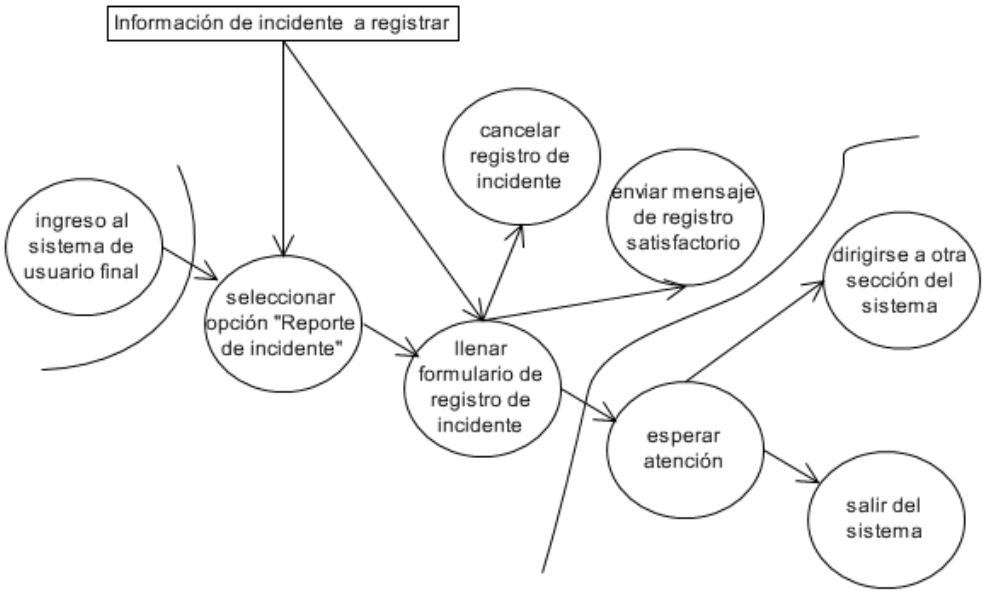


Figura 4.20 Análisis de datos para registro de incidentes.

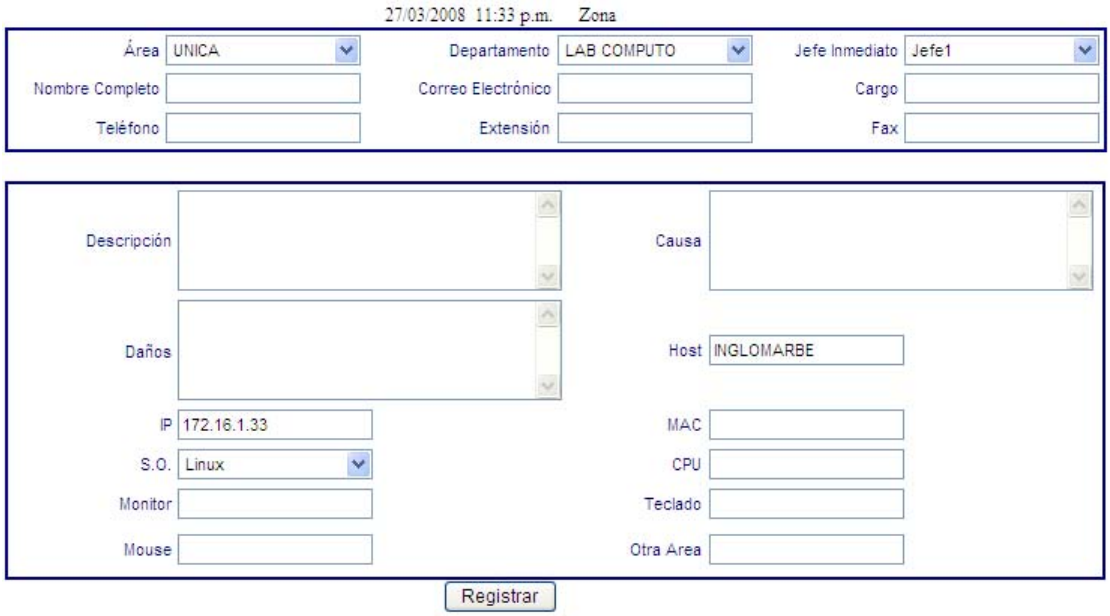


Figura 4.21 Pantalla de registro de incidentes.

4.7 Alcances de la herramienta desarrollada

Una de las responsabilidades del departamento es hacer consciencia a los responsables de las distintas áreas de cómputo y de cualquier usuario que interactué con equipos de cómputo, de la creciente y acelerada evolución que está teniendo la actividad maliciosa de aprovechar las mínimas vulnerabilidades de sistemas informáticos para destruir recursos físicos y/o lógicos, esperando fomentar una cultura de prevención ante dicha situación.

El portal del área se presenta como una herramienta auxiliar para cubrir algunas de las tareas que realiza dicho departamento. Su alcance esta dado a partir de la interacción directa que se desea generar entre el usuario final y el DSC-FI.

Esta herramienta permitirá informar a la comunidad de la Facultad de Ingeniería de los nuevos sucesos de seguridad informática, nuevas actividades y cursos, nuevas vulnerabilidades y problemáticas detectadas, auxiliarmente se proporcionarán documentos como manuales, buenas prácticas y recomendaciones para tener las redes y computadoras sanas.

Adicionalmente se le proporcionará al usuario diferentes mecanismos de comunicación para estar en contacto con el departamento y resolver sus dudas o problemas relativos a la seguridad informática.

Beneficios al DSC-FI

El portal proporcionará al mismo departamento una herramienta útil que le permitirá informar sobre seguridad informática, atender, dar seguimiento y resolver incidentes de seguridad en cómputo, administrar y controlar su información en general.

Representa además un mecanismo de resguardo para mantener íntegra, confidencial y oportuna la información generada.

La explotación adecuada de dicha herramienta permite llevar a cabo análisis cualitativo y cuantitativo sobre el desempeño del departamento, logrando con ello respaldar e informar a la comunidad de la Facultad de Ingeniería sobre las tareas importantes y fundamentales que lleva a cabo. De ello se desprenden puntos de mejora hacia las actividades internas y externas que realiza el departamento.

Igualmente representa un punto de partida para los becarios de UNICA, hacia el desarrollo, administración, mantenimiento (entre otras actividades) de aplicaciones bajo plataforma Windows, donde resultará de vital importancia sus conocimientos sobre seguridad informática para determinar su óptimo funcionamiento desde una perspectiva interna (nivel código) como externa (entorno).

Beneficios a usuario final

El usuario final comprende a la comunidad de la Facultad de Ingeniería que requiera información, asesoría, atención, contactar y/o investigar sobre incidentes e información en general de seguridad en cómputo. El usuario final a través de esta herramienta podrá realizar cada una de las acciones señaladas.

El principal beneficio de esta herramienta radica en recibir oportunamente la información sobre incidentes de seguridad informática, y con ello atender y resolver lo antes posible dicho incidente. Evitando así un largo proceso de envío y/o recepción de oficios, donde intervienen factores externos que de no estar disponibles agravan la situación del equipo o red donde se origino el incidente.

A la Facultad de Ingeniería en general se presenta como una alternativa para mantenerse informado, contactar a personal dentro de la misma que asesoré, atienda y resuelva problemas de seguridad informática.

En general el departamento representa la primera área de la Facultad de Ingeniería constituida con el fin de salvaguardar la información de la misma, crear áreas de investigación que involucren directamente a la comunidad estudiantil, fomentar la cultura de la prevención y disponer de personal capacitado en dicho rubro. Hacia el exterior se espera generar intercambio de conocimientos y su involucración directa con otras instituciones, departamentos, etc. (de seguridad informática) como resultado de su eficiente labor. Así, el portal del DSC-FI puede fungir como conector entre la comunidad de la Facultad de Ingeniería con dicho departamento.

CAPÍTULO 5

REQUERIMIENTOS DE IMPLEMENTACIÓN

5.1 Objetivo

El objetivo de este capítulo es obtener una visión específica de los requerimientos de hardware y software necesarios para una implementación óptima del sistema en cuestión. Igualmente se realiza una estimación del costo total del sistema hasta la implementación, a partir del análisis del mismo respecto al cumplimiento de las expectativas esperadas.

5.2 Equipo para la Implementación

Para llevar a cabo la correcta implementación del portal del departamento y considerando las herramientas descritas en el capítulo 3 del presente trabajo se harán la descripción de las características del equipo y software recomendado. Parte fundamental para la elección de los recursos fue el reducir los costos totales de la implementación.

5.2.1 Servidor

Las características del servidor a considerar para una implementación óptima a nivel hardware, se especifican a continuación.

Se requiere de un dispositivo de software que proporcione servicios de aplicación a las computadoras cliente., capaz de gestionar la mayor parte (o la totalidad) de las funciones de lógica de negocio y de acceso a los datos de la aplicación, permitiendo con ello la centralización y la disminución de la complejidad en el desarrollo de aplicaciones.

Debe incluir *middleware* (o software de conectividad) que permita intercomunicarse con variados servicios, para efectos de confiabilidad, seguridad, no-repudio; una Interfaz para Programación de Aplicaciones (API) para que lo desarrollos sean independientes del sistema operativo y de la gran cantidad de interfaces requeridas por un aplicación Web.

Debe tener integrado el sistema operativo *Windows Server 2003 Standard Edition*© y sus últimas actualizaciones, debido a que dicho sistema operativo proporciona los servicios de: IIS 6.0 (Active Server Pages, ASP), FTP, SMTP/POP, NNTP, HTTP/HTTPS, servidor de archivos, servidor impresiones, servidor aplicaciones, servidor de redes privadas virtuales (VPN), controlador de dominio (Active Directory), servidor DNS, servidor DHCP, servidor WINS.

Las recomendaciones físicas son como mínimo 1GB de memoria RAM, 80 GB de espacio en disco duro, procesador (duo) a más de 1.5 GHz de velocidad, tarjeta de red adicional, unidad de dvd +R/RW, soporte de puertos usb, unidad de respaldo.

En el Anexo 4 se establece la cotización y especificación detallada del servidor elegido.

5.2.2 Sistema Operativo

Se sugiere para la implementación *Windows Server 2003*© en su versión *Standard* considerando el menor costo y rendimientos esperados.

El modelo de la plataforma de servidor de la base *Windows Server System*© se establece en la figura 5.1 mostrada, donde se señalan los servicios ofrecidos e infraestructura, obtenida del sitio Web de dicha plataforma.

Figura 5.1 Plataforma de Windows Server System©.

Los productos y funcionalidades que ofrece son los siguientes:

- *Microsoft Internet Security and Acceleration (ISA) Server*©
  Es un firewall extensible y multicapa, así como un sistema caché para acceso a la Web, que proporciona conectividad con Internet de forma segura, rápida y manejable. *ISA Server*©, también es un proxy Web con un caché que mejora el rendimiento de los accesos y reduce los costos al administrar de forma eficiente el ancho de banda.

- *Microsoft Systems Management Server*©
  Es un entorno de administración y gestión de cambios de bajo costo, escalable para diversos roles de trabajo y servidores basados en *Microsoft Windows*, permite distribuir actualizaciones de seguridad a todos los entornos Microsoft de la organización.

- *Microsoft Operations Manager©*
  Es un entorno de administración de operaciones corporativo que proporciona una gestión integral de eventos, monitorización y gestión de alertas preactiva, informes y análisis de tendencias.

- *Microsoft Application Center©*
  Herramienta de instalación y administración para aplicaciones Web de alta disponibilidad basadas en Windows. *Application Center©* facilita el manejo de grupos de servidores y la implantación de aplicaciones, minimizando la necesidad de conocer detalles internos de las aplicaciones.

En cuanto a la infraestructura de aplicaciones, la plataforma de *Windows Server 2003©* ofrece los siguientes elementos:

Un modelo de programación integrado dentro de *.NET Framework* que permite un nivel de integración de software usando servicios Web basados en XML.

- Un servidor Web integrado mediante *Internet information server©* que ofrece seguridad, disponibilidad y rapidez para aplicaciones Web.
- Servicios de directorio integrados que proporcionan rendimiento, escalabilidad y flexibilidad.

Los productos ofrecidos son:

- *Microsoft SQL Server©*
  Paquete gestor de bases de datos y análisis completo, integrado con la Web, ofrece soporte nativo para XML y la capacidad de realizar consultas a través de Internet.

- *Microsoft BizTalk Server©*
  Conjunto de herramientas que permite construir e implantar rápidamente procesos de negocio así como la integración de aplicaciones dentro de la organización y con otros partners.

- *Microsoft Content Management Server©*
  Herramienta que permite construir y poner en servicio sitios Web basados en contenidos, con niveles de escalabilidad, fiabilidad y rendimiento.

- *Microsoft Host Integration Server©*
  Extiende sus aplicaciones basadas en Windows a otros sistemas mediante integración a nivel de red, aplicaciones y datos.

Inicialmente se ofrecía *Visual Studio 2003©, Windows Server 2003© y SQL Server©* juntos como la plataforma de aplicaciones unificada con un modelo de programación homogéneo para escalar desde un dispositivo móvil hasta el Centro de Datos. Actualmente las herramientas de desarrollo así como el entorno .NET están optimizados específicamente para el sistema operativo *Windows Server 2003 o 2005©, Visual Studio 2005© y SQL Server Express 2005©,* lo que permite un mayor rendimiento del sistema operativo

5.2.2.1 Costo de Sistema Operativo

Para hacer mención del costo del producto *Windows Server 2003 Standard Edition©*, es necesario señalar los tipos de licencia ofrecidos por *Microsoft.*

Licencia. Otorga el derecho a instalar y utilizar el software de servidor.

La licencia de dos componentes supone un modelo de bajo costo y una manera de pagar por la potencia utilizada: a mayor número de dispositivos o usuarios que acceden al software del servidor, mayor será el costo de licencia y el precio. Dicho modelo es por tanto factible a organizaciones de cualquier tamaño.
CAL. Es una licencia de acceso al cliente (CAL, por sus siglas en inglés *Client Access License*) de Windows que otorga el derecho a un dispositivo o usuario a utilizar los servicios-aplicaciones de un servidor.

Tipos de CAL. Existen dos tipos de CAL, por usuario (*user*) y por dispositivo (*device)* cuyo precio es el mismo.

Modos licenciar CALs. Existen dos modos de licencia aplicables a las CALs de Windows:

- *Modo por servidor (per server).* El número de CALs de Windows adquiridas debe ser proporcional al número de usuarios y/o usuarios que acceden al servidor en un momento dado. El número máximo de conexiones concurrentes es igual al número de CALs de Windows compradas.
- *Modo por dispositivo o por usuario (per sit).* Se debe adquirir una CAL por cada dispositivo o usuario, de forma que dicho usuario o dispositivo podrá acceder a cualquier servidor bajo esta modalidad. No hay límite en el número de dispositivos o usuarios que acceden a un servidor de forma concurrente.

El costo de la licencia de *Windows Server 2003*© al día 20 de junio del presente año, es:

Descripción de Producto	Precio Lista	Precio UNAM
Windows Server 20003 Standard Edition R2 x32 All Lng Lic/SA* Pack MVL©	**$ 14,628**	**$ 1,978.00**

Tabla 5.1 Costo de licencia de Windows Server 2003© para UNAM.

5.2.3 Manejador de Bases de Datos

El manejador de base de datos utilizado para el desarrollo del sistema y la administración de la correspondiente base de datos es *Microsoft SQL Server 2000 Enterprise Edition*©. Sin embargo se recomienda su última versión *Microsoft SQL Server 2005 Express Edition*©.

Considerando la migración de la base de datos del sistema a la versión de *SQL Server 2005*©, el costo de la licencia es como sigue, al día 20 de junio del presente año:

Descripción	Precio Lista	Precio UNAM
SQL Server 2005 Enterprise Edtn Win32 All Lng Lic/SA Pack MVL©	**$172,822.00**	**$38,307.00**

Tabla 5.2 Costo de licencia de SQL Server 2005©.

5.2.4 Lenguaje de Desarrollo

La herramienta bajo la cual está desarrollada la aplicación es *Microsoft Visual Studio 2005 Profesional Edition*©, a partir de ello se realizo la estimación del costo de la licencia y se especifican los requerimientos de hardware y software para la implementación.

El costo de la licencia al día 20 de junio del presente año para *Visual Studio 2005*© se especifica en la tabla 5.3.

Producto	Precio
Microsoft Visual Studio 2005 Profesional Edition©	**$ 8381.51**

Tabla 5.3 Costo de licenciamiento Visual Studio 2005©.

.

5.3 Verificación del Sistema

Las listas de verificación o *checklist* son listas de cosas a comprobar hasta asegurar que todo se ha verificado correctamente, en general ayudan a garantizar la coherencia e integridad en el desempeño de una tarea. Son utilizadas en el aseguramiento de la calidad de la ingeniería de software, para comprobar el cumplimiento de proceso, código de la normalización y la prevención de errores principalmente.

Las siguientes consideraciones permiten analizar y verificar que el sistema desarrollado cumple ciertas características de seguridad, diseño, validaciones, infraestructura, arquitectura, autenticación.

En la sección de infraestructura hacia el desarrollo del sistema se hace referencia a las consideraciones externas a la codificación tales como políticas de seguridad, seguridad externa tal como firewalls, cifrado, las estructuras del dominio, bases de datos y servidores, características de comunicación.

La lista 1 señala las consideraciones generales realizadas al diseño del sistema donde los íconos ☑, ☒ y ☐ señalan si cumple, no cumple, o está en proceso, respectivamente.

Consideraciones de Infraestructura

☑ El diseño identifica, entiende, y se acomoda a las políticas de seguridad de la organización.
☐ Las restricciones impuestas por la seguridad de infraestructura (incluyendo servicios disponibles, protocolos, y restricciones de cortafuegos) son identificadas.
☐ El diseño reconoce y acomoda restricciones impuestas por ambientes de alojamiento Web (incluyendo exigencias de aislamiento de aplicación).
☐ El diseño identifica las exigencias de infraestructura de despliegue y la configuración de despliegue de la aplicación.
☐ Las estructuras de dominio, los servidores remotos de aplicación y servidores de bases de datos son identificadas.
☑ El diseño identifica los puntos de mantenimiento de configuración de la aplicación (tales como qué se necesita para ser configurado y qué herramientas están disponibles para un administrador).
☐ Las características de comunicación segura proporcionados por la plataforma y la aplicación son conocidas.
☑ El diseño dirige la adaptabilidad requerida y los criterios de funcionamiento.

Lista 1. Consideraciones de infraestructura.

Las condiciones de infraestructura del sistema son cubiertas en su mayoría como resultado de la interacción directa del mismo con el entorno del departamento, donde existen políticas de seguridad bien definidas sobre aplicaciones Web, se dispone de un marco de seguridad físico y lógico planteado y administrado por el propio departamento, además de contar con personal capacitado en el tema.

Así, las especificaciones de la configuración y administración de las herramientas que engloba el sistema, son planteadas en el manual del administrador.

Arquitectura de Aplicación y Consideraciones de Diseño

Validación de Entrada

Respecto a la arquitectura y algunas consideraciones de diseño sobre la validación de entrada, consideradas en la lista 2, son en su mayoría cubiertas debido a la adecuada recopilación de requerimientos como etapa inicial, desde la cual se plasmaron las necesidades del área y se especificaron las necesidades de medidas de validación a las distintas entradas que el sistema ofrece.

☑ Todos los puntos de entrada y frontera son identificados por el diseño.
☐ La validación de entrada es aplicada siempre que la entrada es recibida desde fuera de la frontera actual.
☑ El diseño asume que la entrada de usuario es maliciosa.
☑ La validación de entrada centralizada es usada donde es apropiado.
☐La estrategia de validación de entrada que adopta la aplicación es modular y consistente.
☑ El acercamiento de validación es de obligar, rechazar, y luego sanear la entrada.
☑ Los datos son validados por tipo, longitud, formato y rango.
☐ El diseño dirige a la canalización de errores potenciales.
☑ Los nombres de archivos de entrada y rutas de archivos son evitados lo mayormente posible.

Lista 2. Verificación de arquitectura de aplicación y diseño.

La figura 5.2 muestra una de las validaciones necesarias a realizar para ingresar a las secciones p*ublicación de noticias*, g*eneración de reportes* y *atención de incidentes* del sistema, donde se solicitan los datos del administrador y se guardan en variables temporales, se comparan y comprueban a través de consultas SQL que devuelven valores de verdadero y falso, y al verificarse se permite el acceso, en caso contrario se notifica de error en alguno de los datos de la cuenta del administrador.

Figura 5.2 Ejemplo de validación de entrada.

Autenticación

La autenticación definida en el sistema establece cuentas con privilegios de acceso restringidos; en el uso de un algoritmo de cifrado para las claves de acceso; en la separación de carpetas públicas y restringidas, donde las públicas representan las páginas a las que cualquier usuario puede acceder y las restringidas son las páginas a las que sólo el administrador y usuarios con ciertos privilegios pueden utilizar; en la especificación de una tabla en la base de datos exclusiva para el control de cuentas de acceso; en políticas de creación de contraseñas seguras y fuertes (más de seis caracteres alfanuméricos); y monitoreo externo por parte del DSC-FI sobre la aplicación, la lista 3 despliega esta información.

☑ El diseño particiona el sitio Web en áreas públicas y restringidas usando carpetas separadas.
☐ El diseño identifica exigencias de cuentas de servicio.
☐ El diseño identifica almacenamiento seguro de credenciales, aceptadas de los usuarios.
☑ Las políticas de gestión de cuentas son tomadas en consideración por el diseño.
☑ El diseño asegura que la información de error mínima sea regresada en el evento de fracaso de autenticación.
☑ La identidad que es usada para la autenticación con la base de datos es identificada por el diseño.
☑ El diseño adopta una política de cuentas con menores privilegios.
☐ Los resúmenes de contraseñas son almacenados en el almacenamiento de usuario para verificación.
☑ Son usadas contraseñas fuertes.
☐ Las entradas de autenticación (cookies) no son transmitidas sobre conexiones no cifradas.

Lista 3. Verificación de autenticación.

La figura 5.3 corresponde a la p*ublicación de noticias* misma que debe estar disponible a cualquier usuario, y la figura 5.4 corresponde a la sección de r*egistro de incidentes* a la cual solo podrán ingresar los usuarios que conformen parte de la cobertura del área, asimismo se vuelve a observar la importancia y cumplimiento de validaciones de entrada de datos en los campos de c*orreo, teléfono*, ip y *mac*.

Figura 5.3 Ejemplo de páginas con acceso público.

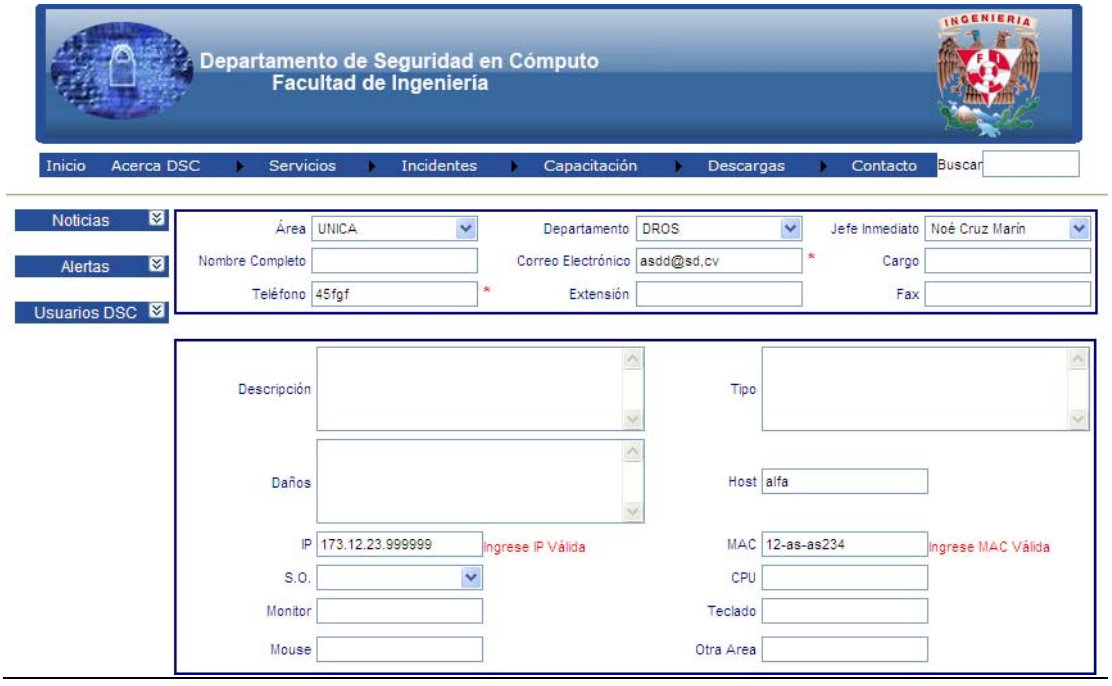


Figura 5.4 Ejemplo de páginas con acceso restringido.

Autorización

Ligado a la *autenticación,* la lista 4, sobre *autorización* establece lineamientos sobre roles, cuentas de ingreso, permisos, niveles de acceso sobre el sistema y su infraestructura. Cabe señalar que las mismas se cumplen como consecuencia de la *autenticación,* donde se establecen cuentas de acceso con distintos privilegios sobre los recursos del sistema (páginas Web), se resguardan bajo una base de datos independiente de las cuentas de acceso al sistema, y cada una de las identidades definidas y utilizadas dentro del sistema son vigiladas y registradas.

☑ El diseño de roles ofrece suficiente separación de privilegios (el diseño considera autorización granular).
☐ Las cuentas de ingreso a la aplicación son restringidas en la base de datos a procedimientos específicos de acceso almacenados.
☑ Las cuentas de ingreso a la aplicación no tienen permisos para acceder a las tablas directamente.
☐ El nivel de acceso a recursos del sistema es restringido.
☐ El diseño identifica exigencias de seguridad de acceso de código. Los privilegios de recursos y privilegios de operaciones son identificadas.
☐ Todas la identidades que son utilizadas por la aplicación son identificadas y el acceso a los recursos por cada identidad son conocidos.

Lista 4. Verificación de autorización.

Gestión de Configuración

La verificación de *gestión de configuración* cumple las especificaciones señaladas debido a la propia configuración del sistema y a la infraestructura de seguridad proporcionada por el ambiente de desarrollo. Las interfaces de administración se consideran seguras al ser creadas en un ambiente de capas donde el código html (diseño) es separado de la capa de datos (lógica del negocio) y el lenguaje manejado en xml con htmlencode; la configuración de almacenamiento radica en la configuración de la base de datos donde su seguridad radica en la administración directa por parte del departamento; los archivos de configuración del sistema son resguardados bajo archivos xml cifrados; y como se menciono en las listas de verificación anteriores, existen cuentas con separación de privilegios y roles para el sistema. Lo anterior lo ejemplifica la lista 5.

☑ Las interfaces de administración son seguras (utilizan autenticación y autorización fuerte).
☐Los canales de administración remota son seguros.
☐ La configuración de almacenamiento es segura.
☑ Los secretos de configuración no son sostenidos en texto plano en archivos de configuración.
☐ Los privilegios del administrador están separadas por roles.
☐ Cuentas con privilegios menores y cuentas de servicio son usadas.

Lista 5. Verificación de gestión de configuración.

5.4 Especificaciones Generales de Mantenimiento

Para el mantenimiento de la aplicación será necesario que se establezcan roles de administración y monitoreo del mismo, conocimientos generales de las herramientas y plataforma bajo las que trabaja, y un completo y detallado conocimiento del mismo.

Derivado de lo anterior podemos especificar que el administrador a cargo del sistema debe cubrir un perfil especificado a partir de los siguientes roles:

Sobre administración del sistema.- Requiere conocimientos en general de la plataforma Windows a nivel desarrollo: administración de IIS 5.0 o superior, en administración de bases de datos bajo *SQL Server 2000*© o superior, conocimientos de configuración y administración de *Visual Studio 2005*©, conocimientos mínimos de C*rystal Reports*©, e involucrarse directamente y conocer el sistema.

Su función corresponde a verificar el óptimo desempeño de la aplicación a nivel software, actualizar y/o modificar secciones de la aplicación. Dicho rol consiste en mantener la integridad y confidencialidad de la información del departamento.

Sobre administración de red: Requiere conocimientos especializados en la plataforma de red utilizada en el departamento, dominio de técnicas y herramientas de seguridad informática, administración y configuración de servidores bajo plataforma Windows.

Su función será mantener el entorno de red correspondiente a la aplicación en óptimas condiciones para un buen desempeño del mismo. Sobre este rol recae la responsabilidad de mantener disponible la información.

Sobre sistema: Requiere conocer detalladamente el funcionamiento del sistema, sin embargo su rol no corresponde al del administrador. Este rol se refiere a ser el encargado de capacitar y orientar al usuario, proponer mejoras, detectar fallas, y dar seguimiento a las mismas.

En este rol recae la responsabilidad de perfeccionar la aplicación en todos sus contextos a fin de que el área ofrezca eficientemente los servicios ofrecidos a través de ella.

5.5 Estimación General del Costo del Sistema

La siguiente especificación general del costo del sistema está basada en dos partes:

- La primera estimación corresponde al sistema, la cual se realizo a partir de las fases que comprendió el sistema, definiendo el tiempo total de producción del sistema en un periodo de seis meses (fines académicos) como se muestra en la tabla 5.4.

Fases	**Estimación [M.N.]**
Análisis de Requerimientos	$ 12,500.00
Diseño	$ 18,000.00
Desarrollo	$ 20,000.00
Construcción	$ 20,000.00
Requerimientos de Implementación	$ 12,000.00
Sub Total	$ 82,500.00
I.V.A.	$ 12,375.00
Total	**$ 94,875.00**

Tabla 5.4 Estimación de costo del sistema.

- La segunda corresponde a la estimación del costo de la implementación del sistema, con el software y hardware recomendado, mostrado en la tabla 5.5.

Recurso para Implementación	**Estimación [M.N.]**
Servidor HP ProLiant ML350G5	**$ 87,135.00**
Sistema Operativo Windows Server 20003 Standard Edition R2 x32 All Lng Lic/SA* Pack MVL©	**$ 1,978.00**
SQL Server 2005 Enterprise Edtn Win32 All Lng Lic/SA Pack MVL©	**$ 38,307.00**
Microsoft Visual Studio 2005 Profesional Edition©	**$ 8,381.50**
Sub Total	**135,801.50**
I.V.A.	**20,370.15**
Total	**156,171.65**

Tabla 5.5 Estimación de costo de implementación del sistema.

Cabe señalar que se investigo acerca de los convenios ofrecidos por Microsoft con la universidad y específicamente con la Facultad de Ingeniería, a través del laboratorio de Microsoft perteneciente al departamento de ingeniería en cómputo y localizado en el edificio de ingeniería eléctrica de la facultad. Con ello el costo de las licencias para el software mencionado en la tabla 5.5 se anula, con uso exclusivo para fines académicos al día 20 de junio de 2008.

Con lo anterior, la tabla 5.5 se reduce a la siguiente tabla 5.6:

Recurso para Implementación	**Estimación [M.N.]**
Servidor HP ProLiant ML350G5	**$ 87,135.01**
Sistema Operativo Windows Server 20003 Standard Edition R2 x32 All Lng Lic/SA* Pack MVL©	**$ 0.00**
SQL Server 2005 Enterprise Edtn Win32 All Lng Lic/SA Pack MVL©	**$ 0.00**
Microsoft Visual Studio 2005 Profesional Edition©	**$ 0.00**
Sub Total	**87,135.01**
Total	**87,135.01**

Tabla 5.6 Estimación de costo de implementación del sistema con convenio.

Dadas las consideraciones anteriores de hardware y software señalado, estos poseen características robustas que pueden ser altamente aprovechadas dentro del departamento para la generación de nuevos proyectos sobre plataforma Windows, que permitan crear un marco de investigación, desarrollo y mantenimiento de aplicaciones, ambientes e interacciones con otros contextos.

El potencial de crecimiento ofrecido se puede observar a través de los siguientes puntos:

- El servidor puede ser configurado para otras aplicaciones o servicios de la plataforma Windows para lograr un mayor aprovechamiento del mismo.

- A través de la configuración de dominio que rige en el departamento, y con este nuevo hardware y software ofrecidos, se pueden recrear ambientes experimentales para analizar y desarrollar proyectos vinculados directamente con la seguridad en cómputo bajo plataforma Windows y su interacción directa con otros ambientes.

- A partir de la implementación de *Visual Studio 2005 Profesional Edition*© que incorpora *SQL Server Express 2005*© y *Crystal Reports 10.0*©, se establece un ambiente de desarrollo para aplicaciones Web, de escritorio y móviles bajo plataforma .NET, y con la facilidad de usar el mismo entorno de desarrollo integrado (IDE) con los lenguajes ASP.NET, C#, J# y C++.

- El portal del departamento representa en sí mismo la vinculación directa de la ingeniería del software con cualquier rama de la computación y cualquier otra externa.

Con esto se consideran solo algunos de los campos de oportunidad que ofrece la implementación del sistema, sin embargo cada uno de los integrantes del DSC-FI puede aportar nuevas y mejores opciones para el aprovechamiento de dicha infraestructura.

CONCLUSIONES Y RECOMENDACIONES

El DSC-FI surge de la necesidad de mejorar la seguridad de redes y sistemas, asimismo afronta una era de gran crecimiento tecnológico en la que es necesario el uso de herramientas que faciliten cada una de las tareas que lleva a cabo. El portal desarrollado responde a las necesidades de administración y control de los incidentes registrados y detectados, así como a la difusión de información referente a la seguridad en cómputo, y propia del departamento.

El proyecto fue realizado con productos comerciales Microsoft Windows© por requerimiento del responsable del DSC-FI, mismo que pretende dirigir y analizar en su conjunto inicialmente el presente software desarrollado bajo plataforma Microsoft, posteriormente el desarrollo del mismo software bajo plataforma de software libre y finalmente hacer un comparativo entre ambos con pruebas de seguridad en general, derivando en la selección del software que mejor se adapte a las necesidades y requerimientos del departamento. Esto debido al gran avance tecnológico que va de la mano con nuevas formas de ataque, más sofisticadas amenazas lógicas, catástrofes naturales cada vez más frecuentes, sin olvidar el factor humano, factores que indistintamente de la plataforma se abren paso gracias a la gran herramienta que representa Internet. Por ello resulta imprescindible analizar los diferentes ambientes de mayor auge y mayor presencia que dentro de la Facultad de Ingeniería representan los sistemas operativos Windows y Linux.

Los objetivos planteados inicialmente se cumplieron satisfactoriamente al desarrollar el portal del DSC-FI con las herramientas: *ASP.NET*, *SQL Server 2000*© y bajo *Windows Server 2003*©.

El portal permite la administración y desplegado de resultados de los servicios ofrecidos por el departamento, a través de la información ofrecida en las distintas secciones que lo conforman. Existen restricciones de acceso por perfil de usuario y de esta forma se controla el acceso y control de la información.

La sección de atención a incidentes y seguimiento de incidentes permite la recopilación de información referente a los distintos incidentes de seguridad informática registrados, el departamento puede explotar dicha información a través de la generación de reportes gráficos y numéricos, con esto se logra hacer análisis estadístico referente a la seguridad informática basado en un estudio cuantitativo de la actividad diaria del departamento.

A través de la sección de *noticias* y otras secciones como *proyectos*, *manuales* y *descargas*, se proporciona información referente a amenazas, vulnerabilidades y ataques, y se promueve el uso de técnicas y/o herramientas de prevención y corrección.

El presente proyecto representa una alternativa fuerte, robusta, documentada y con perspectivas a futuro sustentadas por la calidad de desarrollo, el hardware sugerido, la metodología utilizada y las herramientas auxiliares presentadas.

El portal del DSC-FI aporta grandes beneficios al área y en general a los usuarios que interactúan con él.

La atención de incidentes en línea reduce el tiempo de solución de un incidente de seguridad en cómputo llevado a cabo en las áreas de cobertura del departamento, de igual forma el seguimiento del incidente por parte de personal adscrito al mismo obliga a los mismos a salvaguardar la información referente, manteniendo un histórico, actualizando datos y ofreciendo puntos de mejora hacia el propio sistema. De esta forma se garantiza que la información recopilada será confidencial, permanecerá íntegra y estará disponible cuando se requiera de una forma mucho más práctica y desde cualquier equipo conectado a la red de la Facultad de Ingeniería.

La explotación de la información a través de la generación de reportes y de búsquedas específicas permite al responsable del departamento y a los usuarios involucrados en la toma de decisiones del mismo departamento, ofrecer resultados basados en cifras y valores representativos de la actividad diaria desarrollada y no sólo de un análisis cualitativo o empírico. Dichos resultados derivan en la transparencia y justificación de la importante existencia del departamento como respuesta a la necesidad de áreas que analicen, investiguen, promuevan, informen y se mantengan a la vanguardia en temas relacionados a la seguridad informática, la cual crece a pasos agigantados.

Una tarea fundamental del departamento es mantener informado a los usuarios en general de la Facultad de Ingeniería con el fin de prevenir incidentes de seguridad en cómputo. Dicha tarea es cubierta por el portal y representa una útil herramienta al departamento por su facilidad de uso. Además ofrece la ventaja de incluir secciones donde bien puede ser información de prevención y/o corrección, o un simple comunicado.

En general cada una de las secciones del portal ofrece de forma breve y clara las distintas actividades llevadas a cabo por el DSC-FI, y se despliega la información de contacto con el departamento y cada uno de sus integrantes, enfatizando que como miembros de la Facultad de Ingeniería se pretende crear consciencia, educar y generar un acercamiento entre toda comunidad para reducir el número de incidentes y con ello proteger su información.

Para el DSC-FI resulto fundamental disponer de una herramienta en línea que facilite parte de las actividades que a diario realiza, ya que aumenta la productividad de la organización y se mantiene a la vanguardia en el uso de la tecnología. Asimismo el propio sistema representa una alternativa de análisis dentro del campo de la seguridad informática.

Aunque en general el sistema es robusto no se puede garantizar una seguridad al ciento por cierto dado que existen actualmente múltiples formas de vulnerar y atacar en plataformas indistintas. Sin embargo es posible proteger lo mayormente posible el sistema a través de la elección de hardware y software robusto.

RECOMENDACIONES

1. Se recomienda continuar con el mejoramiento del sistema a través de la realización de pruebas al software ya sea a través de herramientas auxiliares que analicen su comportamiento o con métodos de pruebas de caja blanca y caja negra directamente al código generado. En el anexo 3 se establecen las bases para el análisis del código a través de pruebas de caja negra y caja blanca cuyo objetivo es el encontrar el mayor número de defectos del software, donde debido a la falta del hardware de implementación no se completaron para el sistema.

   Las pruebas de caja blanca se centran en la estructura de control del programa. La prueba del camino básico ejemplificado en el Anexo 3, hace uso de grafos de programa para obtener el conjunto de pruebas linealmente independientes que aseguren la total cobertura.

   Las pruebas de caja negra son diseñadas para validar los requisitos funcionales sin fijarse en el funcionamiento interno del programa. La partición equivalente divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. El análisis de valores límite prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.

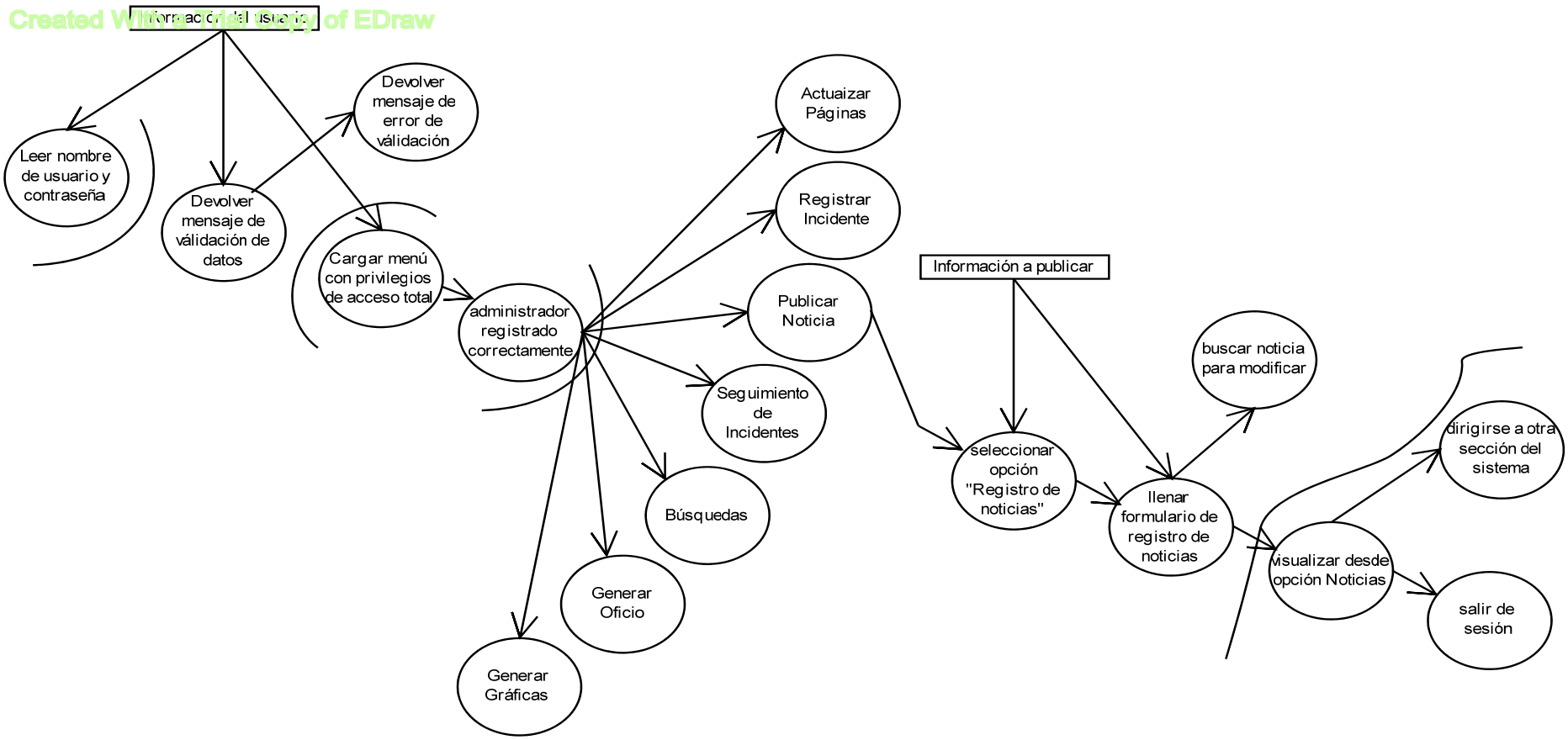
2. La siguiente recomendación consiste en hacer uso de los convenios establecidos acerca del licenciamiento de software propietario, entre la UNAM (Facultad de Ingeniería) y la empresa de software Microsoft a través de los diferentes laboratorios que impulsa y da seguimiento.

3. Por supuesto el hardware de adquisición detallado en el anexo 4 no obliga al departamento a adquirir exclusivamente ese modelo, solo se pretende proporcionar un marco de referencia para el óptimo funcionamiento del sistema.

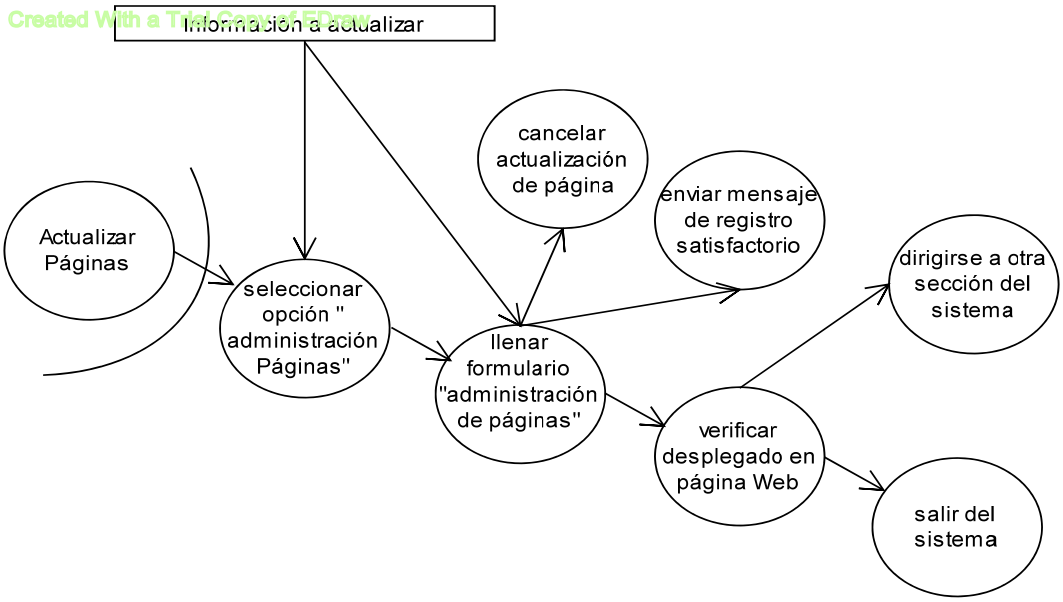
ANEXO 1

DIAGRAMA DE FLUJO DE DATOS Y DICCIONARIO DE DATOS

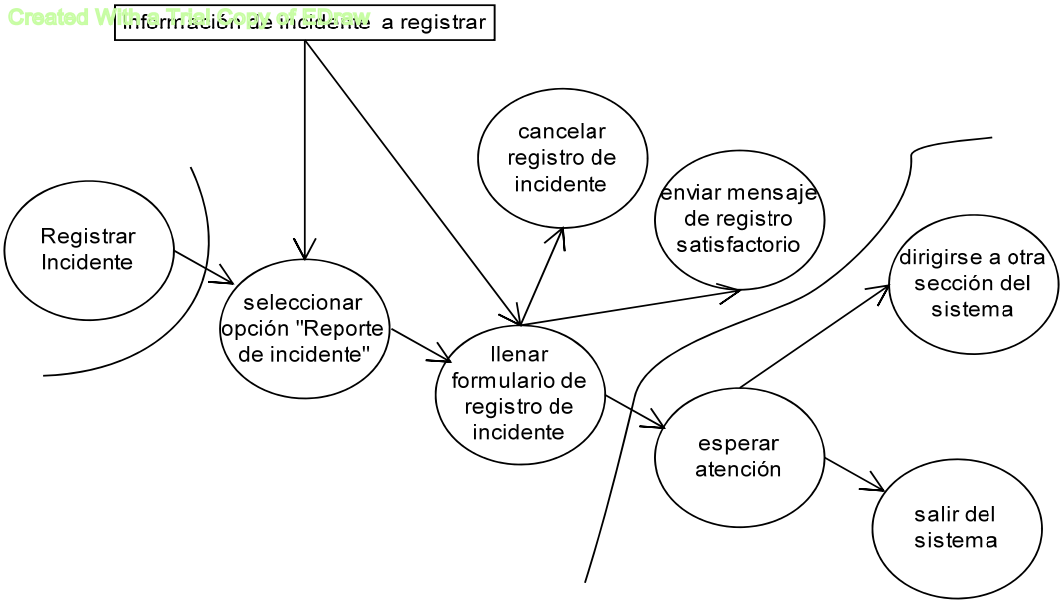
Diagramas de flujo de datos del sistema



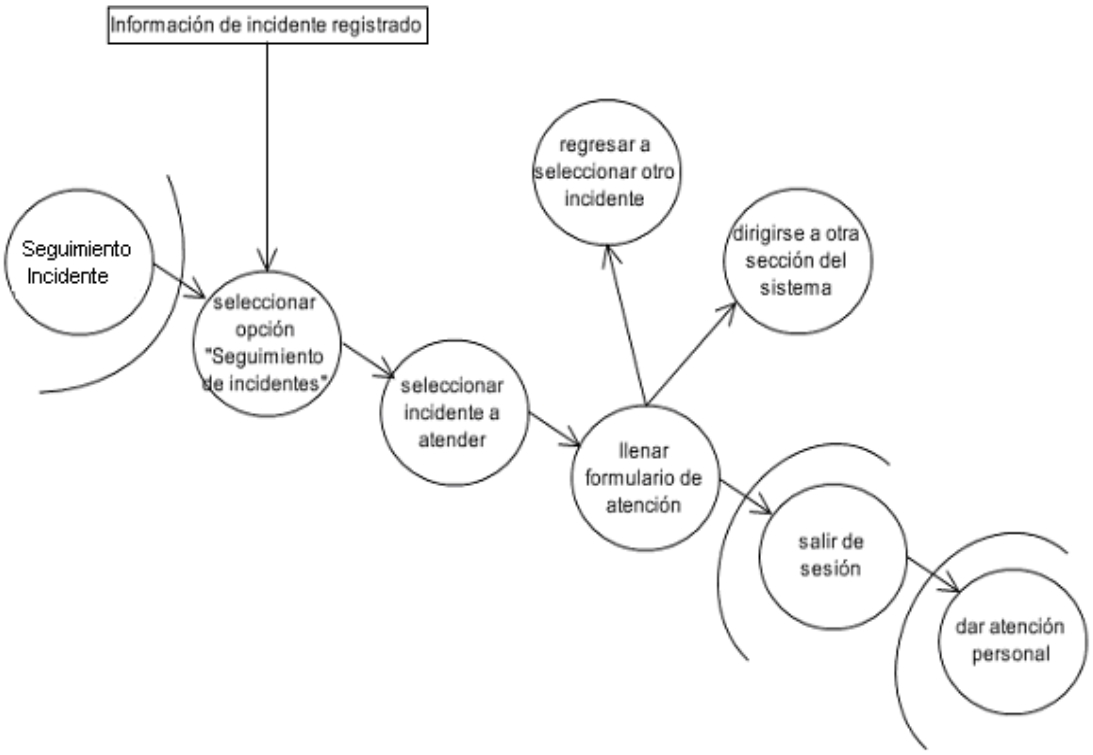
Actualizar Páginas



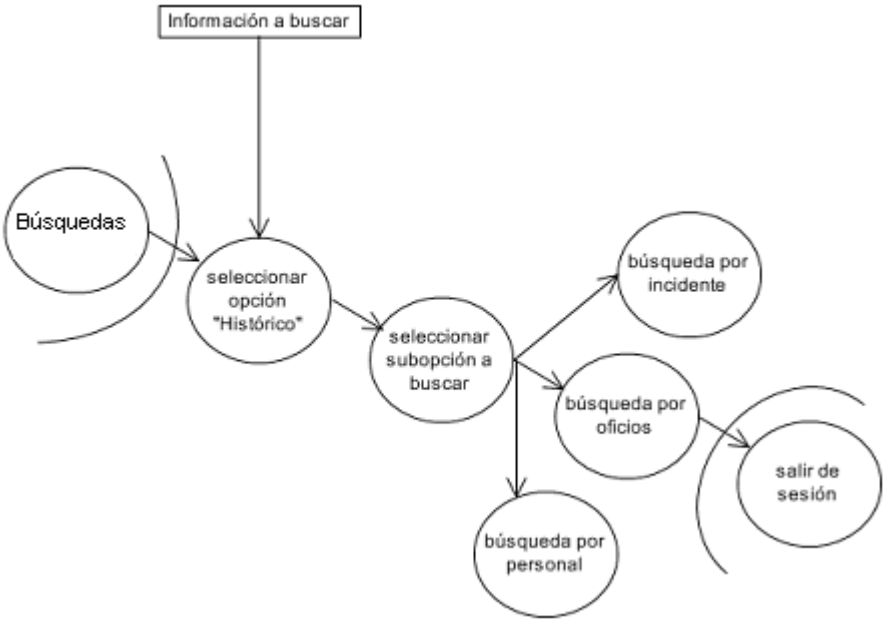
Registrar Incidente



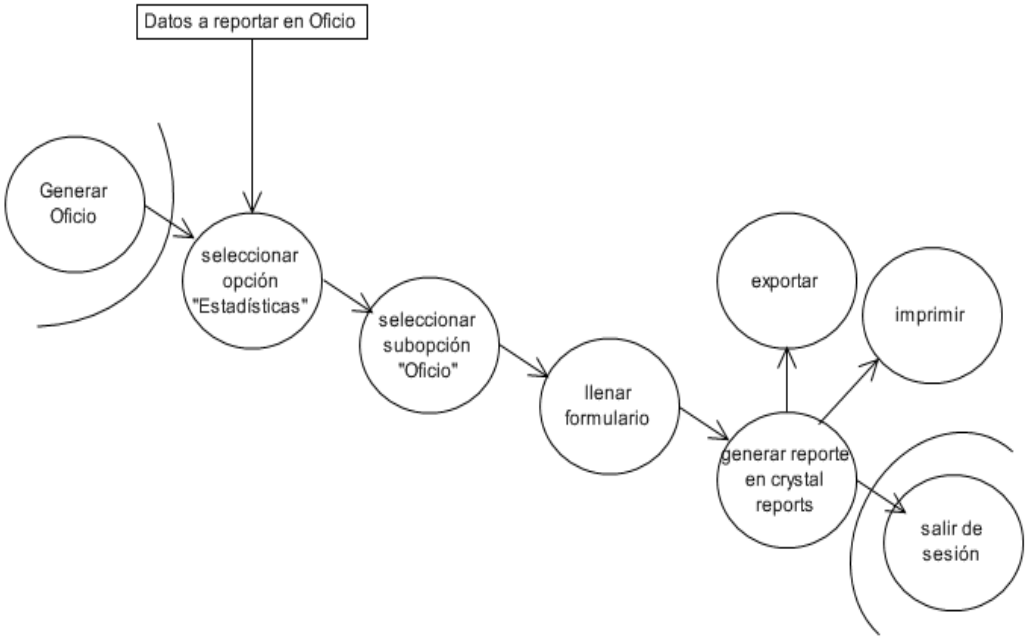
Seguimiento de Incidentes



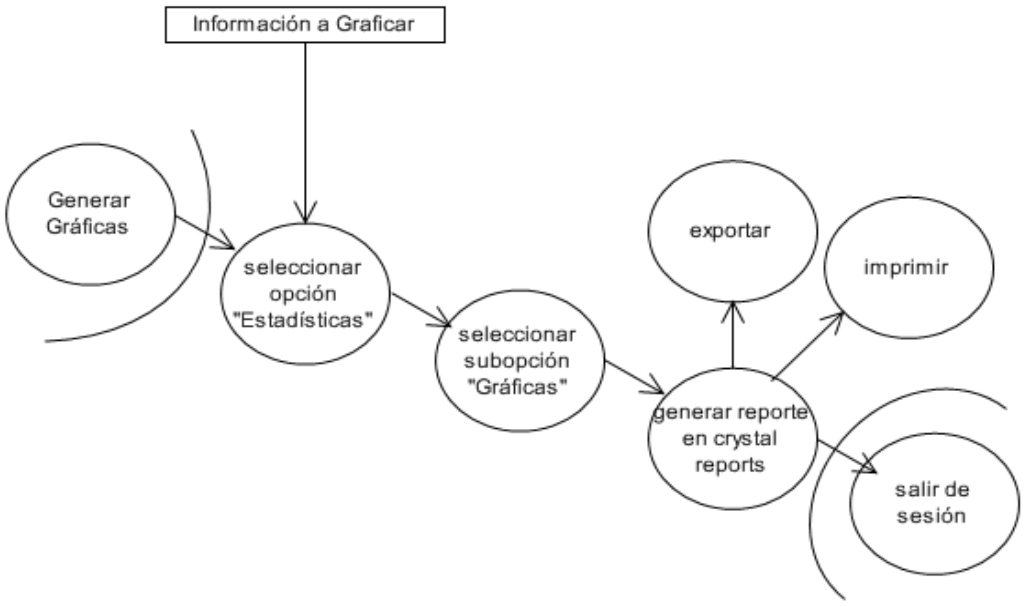
Búsquedas



Generar Oficio



Generar Gráficas



Diccionario de Datos	
nombre	**Activo**
alias	Ninguno
dónde/cómo se usa	Registrar Noticia (entrada)
descripción	Indicador de la fecha en que estará visible o no la noticia registrada.
nombre	**Área**
alias	Ninguno
dónde/cómo se usa	Registro Incidente (entrada) Búsquedas (salida)
descripción	Representa el nombre del área a la que pertenece el usuario dentro de las clasificaciones de atención del departamento.
nombre	**Asunto**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada) Generar Oficio (entrada) Búsquedas (salida)
descripción	Indica el tema a tratar en la generación de oficios, al registrar un incidente y el tema buscar en la sección de búsquedas.
nombre	**Borrado**
alias	Ninguno
dónde/cómo se usa	Registrar usuarios (entrada)
descripción	Representa si un dato: fue borrado por el usuario con un valor cero, si esta activo su valor corresponde a uno.
nombre	**Cargo**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada) Búsquedas (entrada, salida)
descripción	Cadena que representa el titulo de ocupación del usuario que registra un incidente de seguridad en cómputo.
nombre	**Cargo dirigido a**
alias	CargoDirigidoA
dónde/cómo se usa	Oficios (entrada) Búsquedas (entrada, salida)
descripción	Cadena que indica el cargo del usuario hacia quien se dirige un oficio de prevención o corrección sobre un incidente de seguridad.
nombre	**Clave**
alias	Ninguno
dónde/cómo se usa	Actualización de Páginas (salida)
descripción	Palabra(s) clave para la búsqueda de contenido en las páginas del sistema.
nombre	**Comentarios**
alias	Ninguno
dónde/cómo se usa	Registrar Noticia (entrada)
descripción	Texto que indica observaciones acerca de una noticia en particular.
nombre	**Contenido**
alias	Ninguno
dónde/cómo se usa	Registrar Noticia (entrada) Actualizar Página (entrada)

descripción	Información a desplegar como contenido de una sección en el sistema.
nombre	**Contraseña**
alias	Ninguno
dónde/cómo se usa	Inicio de Sesión (entrada)
descripción	Cadena de caracteres de longitud mínima seis y de preferencia alfanumérica para la identificación de usuarios. [A..Za…z0..9%$!?&*]* (seis o más veces)
nombre	**Copia**
alias	Ninguno
dónde/cómo se usa	Oficios (entrada)
descripción	Nombre de la persona a quien se le enviará copia del oficio generado.
nombre	**CPU**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Número de inventario correspondiente al equipo de cómputo en el que se registro el incidente de seguridad.
nombre	**CreadoPor**
alias	Ninguno
dónde/cómo se usa	Registrar Noticia (salida)
descripción	Es el nombre de usuario que da de alta una noticia en el sistema.
nombre	**Daños**
alias	Ninguno
dónde/cómo se usa	Registro Incidente (entrada) Búsquedas (salida)
descripción	Es una descripción de las situaciones que alteran los recursos lógicos o físicos de un equipo de cómputo debido a la ocurrencia de un incidente de seguridad en cómputo.
nombre	**Departamento**
alias	Depto
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Nombre del departamento correspondiente al área a la que corresponde el equipo de cómputo donde se registro el incidente de seguridad.
nombre	**Descripción**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada) Crear Noticia (salida) Búsquedas (salida)
descripción	Es el resumen de una noticia|resumen de cómo sucede un incidente|frase al hacer una búsqueda específica.
nombre	**Diagnostico**
alias	Ninguno
dónde/cómo se usa	Seguimiento de Incidente (entrada)
descripción	Análisis previo a la solución de un incidente de seguridad, por parte del personal capacitado en el DSC-FI.
nombre	**Dirección Física**
alias	MAC
dónde/cómo se usa	Registro Incidente (entrada) Búsquedas (salida)

descripción	Dirección física del equipo de cómputo que reporta un incidente de seguridad, se compone de 6 pares de combinaciones de letras y números. MAC = [*par de letras o números] + *símbolo -* + [*par de letras o números] + *símbolo -* + [*par de letras o números].
nombre	**Dirección IP**
alias	IP
dónde/cómo se usa	Registro Incidente (entrada) Creación Oficio (salida) Creación Reporte (salida) Búsquedas (salida)
descripción	Es la dirección física de un equipo de cómputo perteneciente a la red de la FI, la cual se compone de 3 octetos. IP = [* número que sea distinto de 0] + *secuencia numérica de tamaño 2* + [símbolo punto .] + *cualquier secuencia numérica de tamaño 3* + [símbolo punto .] + *cualquier secuencia numérica de tamaño 3*
nombre	**Email**
alias	Correo Electrónico
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Secuencia de caracteres como sigue: [a…zA…Z0…9]*+@+[nombre de servidor de correo]+.+[com|otra ext.]|+[mx|otra ext.]
nombre	**Estatus**
alias	Ninguno
dónde/cómo se usa	Atención Incidente (entrada)
descripción	Es el estado en que se encuentra un incidente después de ser registrado. Estatus = [Atendido|Pendiente|Seguimiento|Detenido]
nombre	**Extensión**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Conjunto de números complemento al número de teléfono del usuario que reporta el incidente de seguridad. Dentro de la Facultad de Ingeniería corresponden a: [0...9]+ [0...9]+[0...9]+[0...9]
nombre	**Fax**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Número de fax del área o departamento que reporta el incidente de seguridad. Fax = [5][0…9]+ [0…9] [0…9]+ [0…9] [0…9]+ [0…9] [0…9]
nombre	**Fecha**
alias	Ninguno
dónde/cómo se usa	Registrar incidente (entrada) Crear Oficio (salida) Crear Reportes (salida) Crear Noticia (entrada) Búsquedas (salida)
descripción	Es la fecha en que se da de alta un incidente en el sistema. Se registra en formato de dd/mm/yyyy, la cual es dada automáticamente por el servidor, es decir, dd= día del mes actual en número, mm= mes del año en curso en 2 dígitos, yyyy=año en curso en 4 digitos.
nombre	**Herramienta**

alias	Ninguno
dónde/cómo se usa	Atención Incidente (entrada) Búsquedas (salida)
descripción	Es la herramienta lógica o física usada para la atención de un incidente por parte del asesor del DSC-FI.
nombre	**Hora**
alias	Ninguno
dónde/cómo se usa	Registrar incidente (entrada) Crear Noticia (entrada)
descripción	Representa la hora del sistema y es proporcionada con el siguiente formato: hh+mm+ss, es decir, hh= hora actual en formato de veinticuatro horas, mm= minutos recorridos en la hora actual, y ss= segundos recorridos de la hora actual.
nombre	**Host**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Nombre genérico del equipo donde se registro el incidente de seguridad.
nombre	**IdArea**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente
descripción	Identificador que sirve de llave primaria en el Catalogo de Areas, dentro de la base de datos para el sistema.
nombre	**IdCategoria**
alias	Ninguno
dónde/cómo se usa	Registrar Noticia
descripción	Identificador que sirve de llave primaria en el Catalogo de Categorías de noticias, dentro de la base de datos para el sistema.
nombre	**IdDepto**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente
descripción	Identificador que sirve de llave primaria en el Catalogo de Departamentos, dentro de la base de datos para el sistema.
nombre	**IdEncabezado**
alias	Ninguno
dónde/cómo se usa	Actualizar Páginas
descripción	Identificador único para el encabezado de página en el proceso de Actualización de Páginas, dentro de la base de datos para el sistema.
nombre	**IdEstatus**
alias	Ninguno
dónde/cómo se usa	Seguimiento Incidente
descripción	Identificador que sirve de llave primaria en el Catalogo de Estatus, de la base de datos del sistema. Clasificación de estados de los incidentes registrados.
nombre	**IdIncidente**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente
descripción	Identificador único en el proceso de Registro de Incidentes.

nombre	**IdNoticia**
alias	Ninguno
dónde/cómo se usa	Registrar Noticia
descripción	Identificador único en el proceso de Registro de Noticias, dentro de la base de datos del sistema, que corresponde a la identificación de Noticias registradas.
nombre	**IdOficio**
alias	Ninguno
dónde/cómo se usa	Generar Oficio
descripción	Identificador único en el proceso de Generar Oficios, en la base de datos del sistema.
nombre	**IdPagina**
alias	Ninguno
dónde/cómo se usa	Actualizar Página
descripción	Identificador único en el proceso de Actualización de Páginas, correspondiente a la página que se desea actualizar.
nombre	**IdSO**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente
descripción	Identificador único en el Catalogo de Sistemas Operativos, clasificación de sistemas operativos que tienen los equipos de cómputo donde se registrar los incidentes.
nombre	**IdUsuario**
alias	Ninguno
dónde/cómo se usa	Inicio de Sesión
descripción	Identificador único en el proceso de registro de usuarios al sistema.
nombre	**IdZona**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente
descripción	Identificador único en el proceso de Registro de Incidentes respecto a las zonas de cobertura del DSC-FI.
nombre	**IdAtención**
alias	Ninguno
dónde/cómo se usa	Seguimiento Incidente
descripción	Identificador único en el proceso de Seguimiento de Incidente.
nombre	**Jefe_Inmed**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Nombre del jefe inmediato de quien registra el incidente de seguridad.
nombre	**Link**
alias	Liga (salida)
dónde/cómo se usa	Búsquedas
descripción	Liga de direccionamiento hacia la página o noticia correspondiente a la búsqueda realizada.
nombre	**Monitor**
alias	Ninguno

dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Número de inventario correspondiente al monitor del equipo de cómputo en el que se registro el incidente de seguridad.
nombre	**Mouse**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Número de inventario correspondiente al ratón (mouse) del equipo de cómputo en el que se registro el incidente de seguridad.
nombre	**Nombre**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Nombre de la persona que registra el incidente de seguridad.
nombre	**NombreCategoria**
alias	Ninguno
dónde/cómo se usa	Registrar Noticia (entrada)
descripción	Nombre de las categorías de las noticias a registrar.
nombre	**NombreDirigidoA**
alias	Ninguno
dónde/cómo se usa	Generar Oficios (entrada)
descripción	Nombre de la persona a quien se dirige el oficio de atención o prevención sobre el incidente de seguridad que se detecta.
nombre	**NumOficio**
alias	Ninguno
dónde/cómo se usa	Generar Oficios (entrada)
descripción	Número de oficio a generar para reportar incidente de seguridad.
nombre	**Observaciones**
alias	Ninguno
dónde/cómo se usa	Registro Incidente (entrada) Creación Noticia (salida) Creación Oficio (salida)
descripción	Comentarios extra acerca de un incidente, noticia u oficio que deba mencionarse.
nombre	**Otra_Area**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Nombre del área de dónde se reporta el incidente de seguridad y que no aparece en el catalogo de áreas del sistema.
nombre	**Pagina**
alias	Ninguno
dónde/cómo se usa	Actualizar Páginas (entrada)
descripción	Nombre de la página Web en el sistema que se desea actualizar.
nombre	**Sistema Operativo**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Nombre del sistema operativo con el que cuenta el equipo de cómputo donde se registro el incidente.
nombre	**Teclado**

alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Número de inventario correspondiente al teclado del equipo de cómputo en el que se registro el incidente de seguridad.
nombre	**Teléfono**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Número telefónico del área y/o departamento a la que pertenece el equipo de cómputo donde se registro el incidente. Telefono= [5][0…9]+[0...9] [0...9]+ [0...9] [0...9]+ [0...9] [0...9]
nombre	**Texto_Encabezado**
alias	Ninguno
dónde/cómo se usa	Actualizar Página (entrada)
descripción	Información que se colocará en el encabezado de la página Web del sistema a actualizar.
nombre	**TipoIncidente**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Nombre de las clasificaciones de incidentes establecidas por el DSC-FI.
nombre	**TipoOficio**
alias	Ninguno
dónde/cómo se usa	Generar Oficios (entrada)
descripción	Nombre de la clasificación de oficios enviados a las áreas vulnerables, correctivos o preventivos.
nombre	**Titulo**
alias	Ninguno
dónde/cómo se usa	Registrar Noticia (entrada)
descripción	Texto de reconocimiento o identificación de una noticia, es decir, el título de la noticia como se verá desplegada en el sistema.
nombre	**Titulo_Pagina**
alias	Ninguno
dónde/cómo se usa	Actualizar Página (entrada)
descripción	Nombre de la página o sección que se desea actualizar en el sistema.
nombre	**Usuario**
alias	Ninguno
dónde/cómo se usa	Registro Incidente (entrada) Registro Usuarios DSC-FI (entrada)
descripción	Nombre de la persona que registra el incidente de seguridad, puede o no corresponder con el nombre del dueño del equipo donde se detecto tal evento. Nombre de persona que pertenece al DSC-FI y requiere una cuenta de usuario para el sistema.
nombre	**Zona**
alias	Ninguno
dónde/cómo se usa	Registrar Incidente (entrada)
descripción	Nombre del lugar donde se presenta el incidente de seguridad, de acuerdo a la clasificación por segmentos que realiza el DSC-FI.

ANEXO 2

FRAGMENTO DE CÓDIGO

Código fuente sección "noticias"

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Web;

public class Noticia
{
        public Noticia()
        {
    _idNoticia = 0;
    _titulo = String.Empty;
    _creadoPor = String.Empty;
    _descripcion = String.Empty;
    _contenido = String.Empty;
    _fechaHora = new DateTime(2000, 1, 1);
    _activo = true;
    _comentarios = 0;
        }

  public Noticia(int idNoticia)
  {
    SqlConnection connection = new  SqlConnection(ConfigurationManager.ConnecctionStrings
                                    ["DSC_ConnectionString"].ConnectionString);
    connection.Open();
    SqlCommand command = new SqlCommand("SELECT Id_Noticia, Titulo, CreadoPor,
                                Descripcion, Contenido, FechaHora, Activo, Comentarios FROM
                                Noticia WHERE Id_Noticia = @Id_Noticia", connection);
    command.Parameters.AddWithValue("@Id_Noticia", idNoticia);
    SqlDataReader reader = command.ExecuteReader();
    if (reader.Read())
    {
      Noticia.LlenarNoticia(this, reader);      }
    reader.Close();
    connection.Close();
  }

  public void Create()
  {
    SqlConnection connection = new SqlConnection(ConfigurationManager.ConnectionStrings
                                                  ["DSC_ConnectionString"].ConnectionString);
    connection.Open();
    SqlCommand command = new SqlCommand("INSERT INTO Noticia(Titulo, CreadoPor,
                                        Descripcion, Contenido, FechaHora, Activo,
                                        Comentarios) VALUES (@Titulo, @CreadoPor,
                                        @Descripcion, @Contenido, @FechaHora, @Activo,
                                        @Comentarios)", connection);
    command.Parameters.AddWithValue("@Titulo", _titulo);
    command.Parameters.AddWithValue("@CreadoPor", _creadoPor);
    command.Parameters.AddWithValue("@Descripcion", _descripcion);
    command.Parameters.AddWithValue("@Contenido", _contenido);
```

```
        command.Parameters.AddWithValue("@FechaHora", _fechaHora);
        command.Parameters.AddWithValue("@Activo", _activo);
        command.Parameters.AddWithValue("@Comentarios", _comentarios);
        command.ExecuteNonQuery();
        command.Parameters.Clear();
        command.CommandText = "SELECT @@IDENTITY";
        _idNoticia = Convert.ToInt32(command.ExecuteScalar());
        connection.Close();
        HttpRuntime.Cache.Remove("25noticias");
    }


public void AgregarCategoria(Categoria c)
  {
        Cache cache = HttpRuntime.Cache;
        string key = "categorianoticias" + _idNoticia.ToString();
        if (cache[key] != null) cache.Remove(key);
        SqlConnection connection = new SqlConnection(ConfigurationManager.ConnectionStrings
                                            ["DSC_ConnectionString"].ConnectionString);
        connection.Open();
        SqlCommand command = new SqlCommand("INSERT INTO Categoria_Noticia (Id_Noticia,
                                                Id_Categoria) VALUES (@Id_Noticia,
                                                @Id_Categoria)", connection);
        command.Parameters.AddWithValue("@Id_Noticia", _idNoticia);
        command.Parameters.AddWithValue("@id_categoria", c.IdCategoria);
        command.ExecuteNonQuery();
        connection.Close();
    }
  } //end class
```

Código fuente sección "reportes"

```
Imports CrystalDecisions.CrystalReports.Engine
Imports CrystalDecisions.ReportSource
Imports CrystalDecisions.Shared

Partial Class Reportes_VistaGraficas
    Inherits System.Web.UI.Page

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load

        Dim vd As New ReportDocument
        Dim frmGrafica As New Reportes_VistaGraficas
        Dim reportPath As String = Server.MapPath("Graficas.rpt")

        vd.Load(reportPath)
        CrystalReportViewer1.ReportSource = vd
        frmGrafica.DataBind()

    End Sub
End Class
```

ANEXO 3

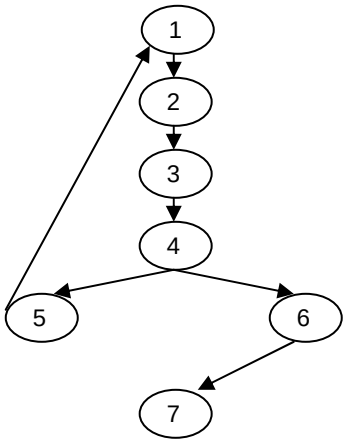
CASOS DE PRUEBA

Pruebas de caja blanca

Prueba del camino básico

```
'Código para leer una Noticia en el Portal
' Incialmente se declara el tipo de objeto, se establece la conexión con la Base de Datos y se ejecuta el
' comando SQL para verificar si dicha noticia existe, se devuelve a través del método '
' LlenarNoticia y finalmente se cierra la conexión con la Base de Datos por seguridad.

public Noticia(int idNoticia)
{
  SqlConnection connection = new   SqlConnection(ConfigurationManager.ConnecctionStrings
                                   ["DSC_ConnectionString"].ConnectionString);
   connection.Open();
1.  SqlCommand command = new SqlCommand("SELECT Id_Noticia, Titulo, CreadoPor,
Descripcion,
                                   Contenido, FechaHora, Activo, Comentarios FROM
                                   Noticia WHERE Id_Noticia = @Id_Noticia",
                                   connection);
2. command.Parameters.AddWithValue("@Id_Noticia", idNoticia);
3.  SqlDataReader reader = command.ExecuteReader();
4.  if (reader.Read())
    {
5.  Noticia.LlenarNoticia(this, reader);
    }
6. reader.Close();
7. connection.Close();
}
```



Grafo de flujo para secciòn de código Leer Noticia

Camino 1: 1-2-3-4-5-1
Camino 2: 1-2-3-4-6-7

Pruebas de caja negra

Partición equivalente

Registro de Incidentes

Una vez que el usuario ingresa al Portal del DSC-FI para registrar un incidente de seguridad en cómputo, debe llenar el siguiente formulario con los siguientes datos:

Área: cadena de caracteres de selección sin opción a modificar longitud 255

Departamento: cadena de caracteres de selección sin opción a modificar, dependiente del área elegida longitud 255

Jefe Inmediato: cadena de caracteres de selección sin opción a modificar, dependiente del área y departamento previamente seleccionados longitud 255.

Nombre Completo: cadena de caracteres de longitud 255.

Correo electrónico: valor alfanumérico bajo la siguiente estructura [0-9]*[a-z]*[A-Z]*[_.-]*@[nombre_servidor].[extensión][.]*[extensión2]*

Cargo: cadena de caracteres de longitud 255.

Teléfono: valor numérico de ocho dígitos con opción de uso de guión medio [-] entre dígitos.

Extensión: valor numérico de seis dígitos máximo.

Fax: valor numérico de ocho dígitos con opción de uso de guión medio [-] entre dígitos.

Descripción: cadena de caracteres de tipo texto de longitud 16.

Tipo de incidente: cadena de caracteres de tipo texto de longitud 16.

Daños: cadena de caracteres de tipo texto de longitud 16.

Host: cadena de caracteres de longitud 255.

IP: valor numérico agrupados en octetos seguidos de el signo punto a excepción del último octeto.

MAC: valor numérico de seis grupos de pares incluyendo el signo puntos entre grupos.

Sistema Operativo: cadena de caracteres de selección sin opción a modificar, longitud 255.

CPU: valor alfanumérico longitud 255.

Monitor: valor alfanumérico longitud 255.

Teclado: valor alfanumérico longitud 255.

Mouse: valor alfanumérico longitud 255.

Otra área: cadena de caracteres de longitud 255.

Las condiciones de entrada asociadas con cada elemento de la aplicación portal del DSC-FI se especifican como sigue:

Campo	*Clasificación*
Área	Condición de entrada, *conjunto* – contenida en el listado mostrado.
Departamento:	Condición de entrada, *conjunto* – contenida en el listado mostrado.
Jefe Inmediato	Condición de entrada, *conjunto* – contenida en el listado mostrado.
Nombre Completo:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – cadena de longitud variable hasta 255.
Correo electrónico:	Condición de entrada, *valor* – cadena de longitud variable hasta 255.
Cargo:	Condición de entrada, *valor* – cadena de longitud variable hasta 255.
Teléfono:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – longitud de 8 dígitos.
Extensión:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – longitud de 6 dígitos.
Fax:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – longitud de 8 dígitos.
Descripción:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – cadena de longitud texto 16.
Tipo de incidente:	Condición de entrada, *valor* – cadena de longitud texto 16.
Daños:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – cadena de longitud texto 16.
Host:	Condición de entrada, *valor* – cadena de longitud 255.
IP:	Condición de entrada, *rango* – valor especificado por segmento 132.168.xxx.xxx
MAC:	Condición de entrada, *valor* – seis grupos de dos dígitos.
Sistema Operativo:	Condición de entrada, *conjunto* – contenida en el listado desplegado.
CPU:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – cadena alfanumérica de longitud máxima 255.
Monitor:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – cadena alfanumérica de longitud máxima 255.
Teclado:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – cadena alfanumérica de longitud máxima 255.
Mouse:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *valor* – cadena alfanumérica de longitud máxima 255.
Otra área:	Condición de entrada, *lógica* – la cadena puede estar o no presente. Condición de entrada, *conjunto* – contenida en el listado desplegado.

ANEXO 4

ESTIMACIÓN COSTO DE SERVIDOR

Servidor

El servidor a considerar para una implementación óptima a nivel hardware, con fecha 20 de junio de 2008 en la empresa LA SFIDA S.A. DE C.V., se especifica a continuación.

Cant.	Descripción	Precio U.	Importe
1	SERVIDOR HP ProLiant ML350G5 SFF SAS/SATA, Chasis tipo Torre. SFF - 2.5" Drives. CTO Chassis Includes: Embedded NC373i Multifunction Gigabit Server Adapter with TCP/IP Offload Engine Integrated Smart Array E200i Controller (RAID 1, 1+0, 5) Eight Hot-Plug (SFF) SAS/SATA Drive Bays Six expansion slots: one 64-bit/133-MHz PCI-X, two 64-bit/100-MHz PCIX, and three x4 PCI Express (with x8 connectors) Six USB ports 800W Hot-Plug Power Supply Tower Form Factor, Keyboard, 2-button Mouse Este servidor tiene garantia 9x5 respuesta dia siguiente tres años	$ 11,968.56	$ 1,968.56
1	Procesador: 1 Quad-Core Intel Xeon Processor X5355 (2.66 GHz, 120 Watts, 1333 FSB), HP X5355 ML350G5 FIO Kit	$ 22,285.78	$ 2,285.78
1	Memoria: 2 GB FBD PC2-5300 2 x 1 GB Kit	$ 6,251.94	$ 6,251.94
3	Discos Duros SAS SFF Hot Plug. HP 146GB 3G SAS 10K SFF SP	$ 15,889.87	$ 5,889.87
1	128-MB Battery-Backed Write Cache (BBWC) PARA SMART ARRAY E200, 641 Y 642. PARA RAID 5.	$ 1,940.61	$ 1,940.61
1	Tarjeta de Red adicional: HP NC320T PCI Express Gigabit Server Adapter	$ 2,182.61	$ 2,182.61
1	Ventilador adicional: Redundant Fan ML350 G5 Kit	$ 716.01	$ 716.01
1	Fuente de poder adicional: Hot Plug Redundant Power Supply 350/370/380 G5 NEMA Kit	$ 3,021.81	$ 3,021.81
1	Unidad óptica: HP DVD+R/RW 16X Half Heigh	$ 1,456.50	$ 1,456.50
1	Floppy Drive ML350/370 G5 Kit	$ 473.33	$ 473.33
1	HP StorageWorks DAT 72 USB Internal Tape Drive (Carbonite)	$ 6,064.07	$ 6,064.07
1	HP DAT 72 Data Cartridge, 72 GB (Para DAT 36/72)	$ 151.13	$ 151.13
1	Monitor HP TFT L1749 PL766AA#ABA	$ 3,105.90	$ 3,105.90
1	Teclado USB Smartcard HP (Carbonite/Silver) Con lector de tarjetas intelijentes.	$ 261.45	$ 261.45
		Sub Total	**$ 75,769.57**
		I.V.A	**$ 11,365.44**
		Total	**$ 87,135.01**

APÉNDICE I
ÍNDICE DE TABLAS Y FIGURAS

Capítulo 1

Capítulo 2

Capítulo 3

Capítulo 4

Capítulo 5

APÉNDICE II
GLOSARIO

Abstracción.- es una descripción simplificada o especificación de un sistema que enfatiza algunos de los detalles o propiedades del sistema, mientras suprime otros.
Áreas clave de proceso (véase ACP)**.-** forman la base del control de gestión de proyectos del software y establecen el contexto en el que se aplican los métodos técnicos, se obtienen productos del trabajo (modelos, datos, informes), se asegura la calidad y el cambio se gestiona adecuadamente.
Arquitectura cliente – servidor.- agrupa conjuntos de elementos que efectúan procesos distribuidos y cómputo cooperativo.
ASP.NET.- lenguaje de desarrollo sobre plataforma .NET.
Bases de datos.- programa residente en memoria, que se encarga de gestionar todo el tratamiento de entrada, salida, protección y elaboración de la información de interés del usuario.
Campo.-representa cada columna de una tabla.
Casos de uso.- son una técnica de captura de requisitos, es decir, un fragmento de funcionalidad del sistema que proporciona al usuario un valor añadido y representan los requisitos funcionales del sistema.
Clase.- conjunto de objetos que comparten una estructura y comportamiento común.
Cliente.- conjunto de software y hardware que invoca los servicios de uno o varios servidores.
Concurrencia.- es la propiedad que distingue un objeto que está activo de uno que no lo está.
Dato.- unidad mínima de información; son hechos sin valuar, un valor sin significado.
Desarrollo basado en componentes.- incorpora muchas de las características del modelo en espiral. Es evolutivo por naturaleza, y exige un enfoque iterativo para la creación del software.
Diseño a nivel de componentes.- transforma los elementos estructurales de la arquitectura del software en una descripción procedimental de los componentes.
Diseño arquitectónico.- define la relación entre los elementos estructurales principales del software, los patrones de diseño que se pueden utilizar para lograr los requisitos definidos para el sistema, y las restricciones que afectan a la manera en que se pueden aplicar los patrones.
Diseño de datos.- transforma el modelo del dominio de información que se crea durante el análisis en las estructuras de datos que se necesitarán para implementar el software.
Diseño de la interfaz.- describe la manera de comunicarse con el software, con sistemas que interoperan dentro de él y con las personas que lo utilizan.
Encapsulación.- en el proceso de ocultar todos los detalles de un objeto que no contribuyen a sus características esenciales.
Esquema de grados.- determina la conformidad con un MMC que define las actividades clave que se requieren en los diferentes niveles de madurez del proceso.
Incidente de seguridad informática.- es un evento que concreta una amenaza al explotar una vulnerabilidad y que afecta a una o más propiedades de la información.
Ingeniería del software.- disciplina o área de la informática o ciencias de la computación, que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo.
Información.- conjunto de datos interrelacionados entre sí que tienen un significado del cual se pueden obtener conocimientos para una futura toma de decisiones.

Interfaz.- implica un flujo de información y tipo específico de comportamiento. Los diagramas de flujo de control y de datos son fuente para este modelo.
Jerarquía o herencia. Es el orden de las abstracciones organizado por niveles.
Lenguaje de programación.- consiste en todos los símbolos, caracteres y reglas de uso que permiten a las personas "comunicarse" con las computadoras.
Lenguaje de bajo nivel.- cadenas de números binarios (ceros y unos) y es el único que interpretan directamente los procesadores, cada instrucción se corresponde con un código máquina equivalente.
Lenguaje de alto nivel.- están dirigidos a solucionar problemas mediante el uso de estructuras dinámicas de datos y son independientes de la arquitectura del ordenador, entre sus principales características.
Lenguaje de nivel intermedio.- permiten un manejo abstracto (independiente de la máquina), pero sin perder mucho del poder y eficiencia que tienen los lenguajes de bajo nivel.
Lenguajes estructurados.- se basa en una metodología de desarrollo de programas llamada refinamiento sucesivo: se plantea una operación como un todo y se divide en segmentos más sencillos o de menor complejidad.
Lenguajes orientados a objetos.- enfatiza la creación de clases que encapsulan tanto los datos como los algoritmos que se utilizan para manejar los datos. Si se diseñan y se implementan adecuadamente, las clases orientadas a objetos son reutilizables por las diferentes aplicaciones y arquitecturas de sistemas basados en computadora.
Listas de verificación o *checklist*.- son listas de cosas a comprobar hasta asegurar que todo se ha verificado correctamente, en general ayudan a garantizar la coherencia e integridad en el desempeño de una tarea.
Mapa de sitio.- sirven como métodos para desplazarse por un sitio Web. Son una lista de vínculos a los archivos HTML del proyecto Web.
Modelado de amenazas.- análisis de una aplicación basado en la seguridad, que ayuda a un grupo de producto a conocer los puntos más vulnerables del producto.
Modelo de proceso.- estrategia de desarrollo que acompañe al proceso, métodos, capas de herramientas y fases descritas en el proceso del software, en la cual se selecciona un modelo de proceso según la naturaleza del proyecto y de la aplicación, los métodos y herramientas a utilizar, así como los controles y entregas requeridos.
Modelo lineal secuencial (ciclo de vida básico o modelo en cascada).- sugiere un enfoque sistemático, secuencial, para el desarrollo del software que comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento.
Modelo de construcción de prototipos.- ayuda en la identificación de los requisitos detallados de entrada, proceso o salida para el software, con el propósito de asegurar la eficacia de un algoritmo, la capacidad de adaptación de un sistema operativo, o la forma en que debería tomarse la interacción hombre-máquina.
Modelo de desarrollo rápido de aplicaciones.- modelo de proceso del desarrollo del software lineal secuencial que enfatiza un ciclo de desarrollo extremadamente corto. Es una adaptación a "alta velocidad" del modelo lineal secuencial en el que se logra el desarrollo rápido utilizando una construcción basada en componentes.
Modelo Entidad-Relación.- concepto de modelado para bases de datos, mediante el cual se pretende visualizar los objetos que pertenecen a la base de datos como entidades las cuales tienes unos atributos y se vinculan mediante relaciones.
Modelos evolutivos.- son iterativos, se caracterizan por la forma en que permiten a los ingenieros del software desarrollar versiones cada vez más completas del software.
Modelo incremental.- Combina elementos del modelo lineal secuencial (aplicados repetidamente) con la filosofía interactiva de construcción de prototipos.
Modelo espiral.- es un modelo evolutivo que conjuga la naturaleza iterativa de construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial. Proporciona el potencial para el desarrollo rápido de versiones incrementales.

Modelo espiral WINWIN.- define un conjunto de actividades de negociación al principio de cada paso alrededor de la espiral. Definiendo las siguientes actividades: identificación del sistema o subsistemas clave de los directivos, determinación de las "condiciones de victoria" de los directivos, negociación de las condiciones de "victoria" de los directivos para reunirlas en un conjunto de condiciones "victoria-victoria" para todos los afectados.
Modelo de desarrollo concurrente.- se puede representar en forma de esquema como una serie de actividades técnicas importantes, tareas y estados asociados a ellas.
Modelo de métodos formales.- son un conjunto de actividades que conducen a la especificación matemática del software de computadora, además permiten especificar, desarrollar y verificar un sistema basado en computadora aplicando una notación rigurosa y matemática.
Modularidad.- es la propiedad de un sistema que ha sido descompuesto en un conjunto de módulos coherentes e independientes.
Objeto.- es aquello que tiene estado (propiedades más valores), comportamiento (acciones y reacciones a mensajes) e identidad (propiedad que lo distingue de los demás objetos).
Persistencia.- es la propiedad de un objeto a través de la cual su existencia trasciende el tiempo (es decir, el objeto continua existiendo después de que su creador ha dejado de existir) y/o el espacio (es decir, la localización del objeto se mueve del espacio de dirección en que fue creado).
Plataforma .NET.- plataforma de software que conecta información, sistemas, personas y dispositivos.
Prácticas clave.- son normas, procedimientos y actividades que deben ocurrir antes de que se haya instituido completamente un área clave de proceso. El SEI define a los indicadores clave como "aquellas prácticas clave o componentes de prácticas clave que ofrecen una visión mejor para lograr los objetivos de un área calve de proceso".
Proceso del software.- marco de trabajo para la tecnología de ingeniería del software. Pág. 18.
Registro.-denominada fila o tupla, representa una colección de valores que describen una entidad del mundo real.
Relación.- representa una tabla.
Servidor.- conjunto de hardware y software que responde a los requerimientos del cliente.
Sistema distribuido.- es aquel en el que los componentes localizados en computadoras, conectados en red, comunican y coordinan sus acciones únicamente mediante el paso de mensajes.
Técnicas de cuarta generación.- herramientas de software que tienen algo en común: facilitan la especificación de algunas características del software de alto nivel, la herramienta genera automáticamente el código fuente basándose en la especificación del técnico.
Tipificación.- definición precisa de un objeto de tal forma que objetos de diferentes tipos no puedan ser intercambiados o puedan intercambiarse de manera muy restringida.

Siglas

ACP: Áreas Clave de Proceso.
ASP: *Active Server Pages*.
CERT: Equipos de Respuesta a Incidentes de Seguridad en Cómputo.
CGI: *Common Gateway Interface.*
CLR: *Common Language Runtime.*
DD: Diccionario de Datos.
DER: Diagrama Entidad- Relación.
DFD: Diagrama de Flujo de Datos.
DGSCA: Dirección General de Servicios de Cómputo Académico.
DRA: Modelo de Desarrollo Rápido de Aplicaciones.
DROS: Departamento de Redes y Servidores (UNICA).
DSC-FI: Departamento de Seguridad en Cómputo de la Facultad de Ingeniería.
DTE*:* Diagrama de Transición de Estados.
HTTP: *HyperText Transfer Protocol.*
IIS: Servicio de Información de Internet *(Internet Information Server*).
MMC: Modelo de Capacidad de Madurez.
NCSA: *Nacional Center for Supercomputing Applications.*
OTAN: Organización del Tratado del Atlántico Norte.
RISC (véase CERT).
RUP: Proceso Unificado Racional (*Rational Unified Process*)
SEI: *Software Engineering Institute.*
SSL: *Secure Socket Layer.*
T4G: Técnicas de Cuarta Generación.
UML: *Unified Modeling Language*. .
UNAM: Universidad Nacional Autónoma de México.
UNICA: Unidad de Servicios de Cómputo Académico.
USECAD: Unidad de Servicios de Cálculo Administrativo.

BIBLIOGRAFÍA

1. **BAUER** F.L. Software Engineering Information Processing 71. North-Holland Publishing Co. Amsterda, 1972.

2. **BERZAL** Fernando, Cortijo Francisco José, Cubero Juan Carlos. Desarrollo Profesional de Aplicaciones Web con ASP.NET.

3. **BOHEM** B.W., Software Engineering, IEEE Transactions on Computers C-25, n.12, dic. 1976, pp.1226-1241.

4. **CEBALLOS** Francisco Javier. Microsoft Visual Basic .Net Lenguaje y Aplicaciones, 2ª. Edición Alfaomega Ra-Ma, 2007.

5. **DATE** C.J. Sistemas de Bases de Datos. Séptima Edición, Pearson Education.

6. **JACOBSON**, I., Booch, G., Rumbaugh J., El Proceso Unificado de Desarrollo de Software, 2000 Addison Wesley.

7. **IEEE** Standards Collection: Software Engineering, IEEE Standard 610.12-1990, IEEE, 1993.

8. **LEVINE** Guillermo. Estructuras fundamentales de la computación: los principios, McGraw-Hill, México 1996.

9. **MORERA** Pascual Juan M, Pérez Campanero Atanasio Juan A. Teoría y Diseño de los Sistemas Operativos. Anaya Multimedia, 2002.

10. **PRESSMAN** Roger S., Darrel Ince adaptación. Ingeniería del Software, Quinta Edición, Mc Graw Hill, 2002.

11. **PUENTE**, Francisco. *Seguridad informática corporativa*. Contacto, 2006, num. 178 feb. p. 40-43.

12. **QUIÑONES**, Alicia. *(In) Seguridad Informática: Entre el delito y el orden público en la red.* Contacto*,* 2006, num.178 feb. p. 28-39.

13. **SANDOVAL**, Vázquez Rafael. *Documentación del Departamento de Seguridad en Cómputo.* Informe Inédito. Facultad de Ingeniería, UNAM, 2003.

14. **SHEPHERD** George. Microsoft ASP.NET 2.0 Step by Step. *M*icrosoft Press, 2005 Edition.

15. **STALLINGS** William, Comunicaciones y Redes de Computadoras*,* 4ª edición, Prentice-Hall, 1997.

16. **TANENBAUM** Andrew S. Redes de Computadoras, 3ª. Edición, Prentice-Hall, México 1997.

17. **ZELKOVITZ** M.V., Shaw A.C., Gannon J.D. *Principles of Software Engineering and Design*. Prentice Hall, 1979.

MESOGRAFÍA

1. **AMADOR** Posadas Juan Pablo.
   Teoría general de sistemas.
   http://www.elprisma.com/apuntes/administracion_de_empresas/teoriageneraldesistemas/
   Citado 20-05-2007, 13:00 hrs.

2. **CIBERAULA**.
   Tecnología orientada a objetos.
   http://java.ciberaula.com/articulo/tecnologia_orientada_objetos/
   Citado: 20-10-2006, 17:00 hrs.

3. Common Warehouse Metamodel (CWM) Specification
   http://www.omg.org/docs/formal/03-03-02.pdf
   Citado: 12-03-2007, 12:30 hrs.

4. **DE LA CÁMARA** Delgado M., Sanchis Marco F.
   Proceso de desarrollo con UML y el modelo CMM.
   http://www.ati.es/gt/calidad-software/SIMO00/SIMO2000-UML.ppt
   Citado: 12-03-2007, 13:00 hrs.

5. **DESARROLLO WEB**.
   Tipos de lenguajes de programación.
   http://www.desarrolloweb.com/articulos/2358.php
   Citado: 26-03-2007, 18:00 hrs.

6. **DGSCA**.
   Lenguajes de alto nivel.
   http://entren.dgsca.unam.mx/introduccion/leng_alton.html
   Citado: 08-04-2007, 13:00 hrs.

7. **FALCÓN** Docampo Amalia.
   Sistemas Operativos.
   http://idtv.det.uvigo.es/~avilas/SO/diapositivas/SISTEMAS%20OPERATIVOS%20-%20TEMA3-SPT.pdf
   Citado: 13-05-2007, 18:00 hrs.

8. **FEAMSTER** Nick, traducción Royer Patrick.
   Seguridad en la Ingeniería del Software.
   http://www.acm.org/crossroads/espanol/xrds7-4/onpatrol74.html
   Citado: 16-05-2007, 19:00 hrs..

9. **FERNANDEZ** Calvo Rafael.
   El software como proceso.
   http://www.ati.es/novatica/2004/171/171-2.pdf
   Citado: 20-05-2007, 11:00 hrs.

10. **FUMERO** Antonio y Roca Genís, colaboración de Fernando Sáez Vaca.
Web 2.0.
http://www.willydev.net/InsiteCreation/v1.0/WillyCrawler/2008.06.07.Articulo.Web%202.0%20y%20mapas%202.0.pdf
Citado: 09-06-2008, 16:00 hrs.
11. **MICROSOFT**
Visual Basic .NET y C# .NET.
http://geeks.ms/blogs/fdiaz/default.aspx
Citado: 06-05-2008, 21:00 hrs.

12. **MICROSOFT**
.NET Framework.
http://msdn.microsoft.com/es-mx/netframework/default.aspx
Citado: 06-05-2007, 20:00 hrs.

13. **MICROSOFT**
Aprender a dominar ASP.NET: presentación de las clases de entidad personalizada.
http://www.microsoft.com/spanish/msdn/articulos/archivo/030505/voices/CustEntCls.mspx#E4B#E4B
Citado: 03-06-2007, 18:00 hrs.

14. **MICROSOFT**
ASP.NET Web Applications.
http://msdn.microsoft.com/en-us/library/ms644563.aspx
Citado: 20-05-2007, 11:40 hrs.

15. **MICROSOFT**
Comprobar páginas Web en Visual Web Developer.
http://msdn.microsoft.com/es-es/library/df5x06h3(VS.80).aspx
Citado: 20-05-2007, 14:00 hrs.

16. **MICROSOFT**
Creating a Data Access Layer.
http://msdn.microsoft.com/en-us/library/aa581778.aspx
Citado: 20-05-2007, 16:00 hrs.

17. **MICROSOFT**
Diseño e implementación de bases de datos con Microsoft SQL Server 2000 Enterprise Edition.
http://www.microsoft.com/learning/es/es/exams/70-229.mspx
Citado: 22-07-2007, 20:00 hrs.

18. **MICROSOFT**
Guía de Programación en Visual Basic.
http://msdn.microsoft.com/es-es/library/y4wf33f0(VS.80).aspx
Citado: 20-05-2007, 10:50 hrs.

19. **MICROSOFT**
IIS 6.0.
http://www.microsoft.com/spain/technet/estudiantes/articulos/iis.mspx.
Citado: 29-07-2007, 13:00 hrs.

20. **MICROSOFT**
Instalación, configuración y administración de Microsoft SQL Server 2000 Enterprise Edition. http://www.microsoft.com/learning/es/es/exams/70-228.mspx
Citado: 22-07-2007, 21:00 hrs.

21. **MICROSOFT**
Novedades de los servicios de Internet Information Server.
http://www.microsoft.com/latam/windowsserver2003/evaluation/overview/technologies/iis.mspx
Citado: 29-07-2007, 13:30 hrs.

22. **MICROSOFT**
Security Guidance Training for Developers
https://www.microsoftelearning.com/eLearning/offerDetail.aspx?offerPriceId=127135.
Citado: 08-07-2007, 17:00 hrs.

23. **MICROSOFT**
Visual Studio 2005.
http://msdn.microsoft.com/es-mx/library/aa187919.aspx
Citado: 20-05-2007, 18:00 hrs.

24. **OLIVARES** José Luis, BUFFA Sistemas.
ASP.NET AJAX**.**
http://www.bs.com.ar/bsweb/RevistaBSknow/PDFs/n23/ajax.pdf
Citado: 05-08-2007, 12:00 hrs.

25. **Sin autor**
Programación Estructurada.
http://www.lenguajes-de-programacion.com/programacion-estructurada.shtml
Citado: 26-03-2007, 22:00 hrs.

26. **REVISTA ENTERATE** en línea.
Silva Alarcón Armando. Modelos de calidad.
La industria del software en México. Año 3, número 25, 01 de 2004.
http://www.enterate.unam.mx/Articulos/2004/01/modelos.htm
Citado: 22-10-2006, 16:30 hrs.

27. **REVISTA** IEEE **América Latina.**
Gutiérrez Carlos A., Fernández Medina Eduardo, Piattini Mario.
Proceso de Desarrollo para Seguridad de Servicios Web.
http://www.ewh.ieee.org/reg/9/etrans/vol4issue2April2006/Vol4issue2April2006TLA.htm Citado: 28-10-2006, 18:00 hrs.

28. **SOMMERVILLE**
Software Engineering, 6ª Edition, Capìtulo1, Addison-Wesley.
Procesos de Software.
http://www.e-market.cl/dir/u05r/ingsw/cap03.ppt
Citado: 03-11-2006.

29. **UNESCO**
Internet Movie Database 2003; International Telecommunication Union, ITU; CIA World Factbook, December 2003 [en línea].
http://www.itu.int, http://www.imdb.com
Citado: 03-11-2006, 16:30 hrs.

30. **UNIVERSIDAD** Politécnica de Valencia.
Introducción a RUP.
https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducción%20a%20RUP.doc
Citado: 20-05-2007, 17:00 hrs.

31. **VAN GIGCH** J.P.
Teoría General de sistemas, Cap. 2, Editorial Trillas 2ª edición, Mèxico.
http://www.itson.mx/diep/Especialidades/pagina%20porcino/cursos/formacionmet/sistemas.doc
Citado: 13-03-2008, 11:00 hrs.

32. **VIEIRA** Robert
Programming SQL Server 2005.
http://www.26ny.com/
Citado: 06-04-2008, 15:00 hrs.