ESCUELA POLITECNICA DEL EJÉRCITO

SEDE LATACUNGA

FACULTAD DE INGENIERIA DE SISTEMAS E INFORMATICA

ESTUDIO COMPARATIVO DE LAS TECNOLOGIAS PUNTO NET Y JAVA EN EL DESARROLLO DE APLICACIONES WEB

PROYECTO PREVIO A LA OBTENCIÓN DEL TITULO DE INGENIERO EN SISTEMAS E INFORMATICA

JAIME PATRICIO ZARATE PIRAY

Latacunga, Diciembre del 2007

AGRADECIMIENTO

Un agradecimiento muy especial a toda mi familia por El apoyo brindado a lo largo de mi vida estudiantil.

A la Escuela Politécnica del Ejército Sede Latacunga, por la formación académica que me ha brindado, a la Carrera de Ingeniería en Sistemas e Informática, al señor Ing. Edison Espinosa coordinador de la carrera. Y de manera muy especial a mis tutores Ing. Raúl Rosero y Santiago Jácome por el apoyo brindado en la culminación de mi proyecto de investigación.

Mil Gracias

DEDICATORIA

A Dios por haberme dado la vida y la oportunidad de hacer mi sueño realidad, a todas aquellas personas que de una u otra manera me apoyaron para culminar mi carrera.

En especial a mis padres, por su apoyo incondicional en cada momento.

CAPITULO I

1.- APLICACIONES DISTRIBUIDAS

En este capítulo se descripción de las Aplicaciones Multicapa y Distribuidas sus características, evolución y diseño.

1.1.- INTRODUCCIÓN

Los sistemas de información y las aplicaciones informáticas de las organizaciones, especialmente con el espectacular desarrollo de Internet y de las redes de comunicación han hecho que las técnicas tradicionales de diseño e implementación de aplicaciones comiencen a ser insuficientes, por lo que un nuevo enfoque de desarrollo se hace necesario.

En esta perspectiva, los avances en la investigación en el área de los sistemas distribuidos de computación en las últimas dos décadas han sido decisivos para la maduración de nuevas tecnologías que faciliten la distribución y la interoperabilidad de aplicaciones.

La creación de aplicaciones en 'n' niveles es el modelo para el desarrollo de software empresarial actual. Para la mayoría de usuarios, una aplicación en 'n' niveles es algo dividido en distintas partes lógicas. La opción más habitual está formada por una división en tres partes (presentación, lógica de negocio y datos), aunque existen otras posibilidades. Las aplicaciones en 'n' niveles surgieron por primera vez como una forma de resolver algunos de los problemas asociados a las aplicaciones cliente/servidor tradicionales, pero con la llegada de la Web, esta arquitectura ha llegado a dominar el nuevo desarrollo.

La revolución del conocimiento es un evento global a la que todas las compañías han de prestar mucha atención de cara a un futuro inmediato. Según se vaya avanzando en el nuevo milenio, las empresas, los países y las personas a lo largo de todo el mundo van a incrementar el desarrollo de su riqueza por medio de una comunicación global y una cooperación extensible a todos los niveles, las arquitecturas de n-capas proporcionan una gran cantidad de beneficios para las empresas que necesitan soluciones flexibles y fiables para resolver problemas complejos inmersos en cambios constantes.

Todas las aplicaciones basadas en n-capas permitirán trabajar con clientes ligeros, tal como navegadores de Internet, Teléfonos Inteligentes, PDAs (Personal Digital Assistants o Asistentes Personales Digitales) y muchos otros dispositivos preparados para conectarse a Internet. Las arquitecturas basadas en n-capas permiten a los componentes de negocio correr en una LAN, WAN o Internet. Esto significa que un usuario cualquiera, con un ordenador y conexión a la Red posee toda la funcionalidad, como si se encontrase delante de su sistema de escritorio.

1.2.- APLICACIONES DISTRIBUIDAS

Las primeras soluciones que incluían servicios de red vinieron con los denominados sistemas centralizados, en los que un solo computador con una o varias unidades centrales de proceso recibía las peticiones de los usuarios que se conectaban a él vía terminales tontas. Sin embargo, debido a razones tales como coste, confianza, falta de escabilidad, flexibilidad y la diversa naturaleza de las diferentes divisiones que conformaban una organización, hacían que este modelo no fuera especialmente atractivo.

El abaratamiento del hardware, la aparición en escena de los ordenadores personales (PC – Personal Computer), el aumento del rendimiento de los PCs y el desarrollo de redes locales, provocaron una evolución en los modelos arquitectónicos de las aplicaciones software, pareciendo razonable repartir el

proceso entre algunos sistemas más o menos distribuidos, en los que normalmente seguirían residiendo las bases de datos, y las máquinas desde las que se conectarían los usuarios, PCs o estaciones de trabajo donde se ejecutaría al menos la parte de interfaz de usuario. Había surgido así el modelo Cliente/Servidor (C/S), caracterizado por dividir la funcionalidad de la aplicación en dos papeles perfectamente definidos: cliente y servidor.

De modo abstracto, el servidor ofrece una serie de servicios que pueden ser utilizados por los clientes para completar la funcionalidad de la aplicación. Una interacción básica implica a un cliente que inicia una petición de algún servicio a un servidor. El servidor entonces realiza la función especificada por el cliente, devolviendo los posibles resultados que el servicio genera. En la práctica, los clientes y servidores se implementan como procesos que se están ejecutando en máquinas conectadas a una red. La infraestructura que se encarga de conectar a los procesos cliente y servidor se denomina middleware y se muestra en la Figura1.1.

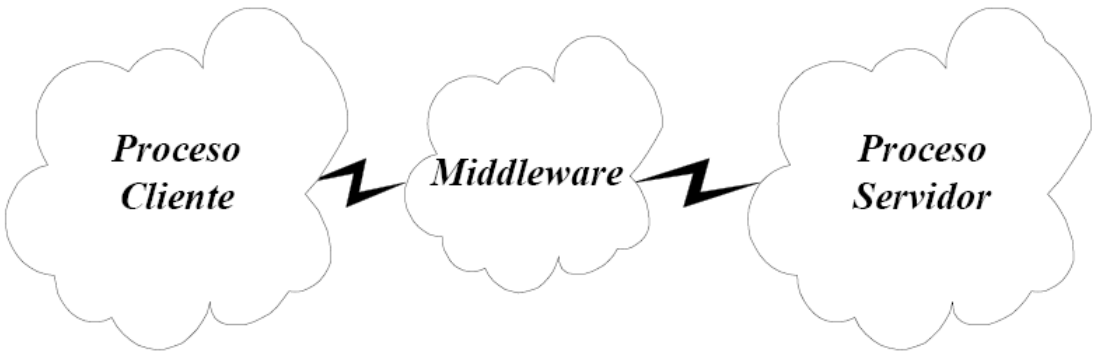

Figura 1.1.- Aplicación Cliente Servidor

La arquitectura C/S se organiza en niveles o capas, existiendo arquitecturas de dos, tres o en general *n* capas. La arquitectura C/S más popular es la de tres capas (Figura 1.2). El cliente se encarga de mantener la interfaz de usuario. En un nivel intermedio se encarga de implementar la lógica de la

aplicación, finalmente en el último nivel se encuentra la lógica de datos. Los clientes se construyen sobre la base de unos servicios encapsulados en los procesos que implementan la lógica de la aplicación, siendo más robustos frente a cambios en esta lógica o en la lógica de datos. La funcionalidad que implementan los clientes es muy sencilla, pudiendo los mismos reutilizar servicios estándares definidos en alguno de los niveles intermedios. La aplicación al estar dividida en partes más pequeñas, hace que el proceso de distribución de funcionalidad en los procesadores siendo más adecuados y mucho más flexible.

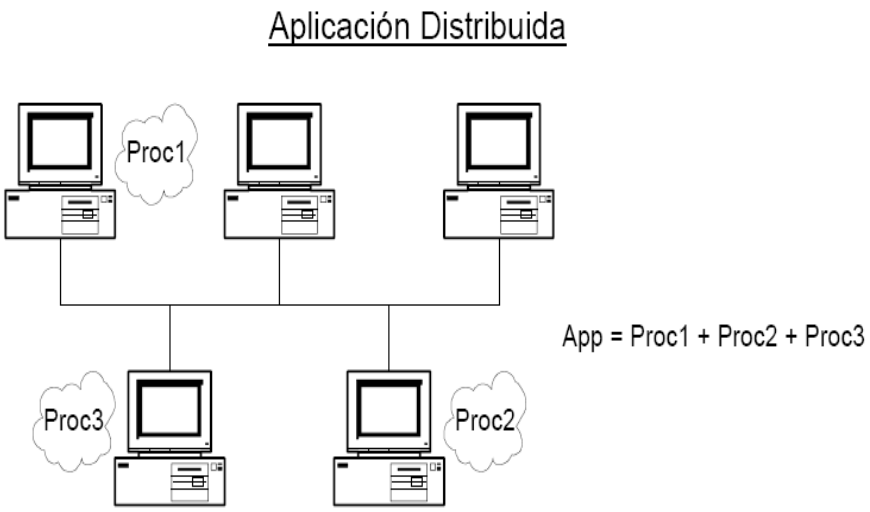

Figura 1.2.- Aplicación Distribuida

La computación distribuida implica el reparto y la ejecución de una serie de procesos a lo largo de varios nodos de una red.

Los programas tienden a superar el modelo monolítico según el cual dada una entrada obtenían una salida, caminando hacia la federación de un conjunto de componentes que interactúan entre sí, cada vez más en un entorno distribuido. Cada uno de estos componentes ofrece servicios a otros para constituir la aplicación software final.

Las características de los sistemas distribuidos son:

- Desarrollos paralelos (en cada capa)
- Aplicaciones más robustas debido al encapsulamiento.
- Mantenimiento y soporte más sencillo, (es más sencillo cambiar un componente que modificar una aplicación monolítica)
- Mayor flexibilidad (se pueden añadir nuevos módulos para dotar al sistema de nueva funcionalidad).

1.3.- DISEÑO DE APLICACIONES DISTRIBUIDAS

El diseño de una aplicación distribuida implica la toma de decisiones sobre su arquitectura lógica y física, se debe tener un conocimiento claro de los procesos empresariales que realizará la aplicación, como: requisitos (funcionales, no funcionales u operativos), los niveles de escalabilidad, seguridad y mantenimiento. A continuación se describe aplicaciones en 3 capas que son las mas utilizadas, el diseño de aplicaciones modernas involucra la división de una aplicación en múltiples capas; la interfase de usuario, la lógica de negocios, y la capa de acceso a datos como muestra la Figura 1.3.

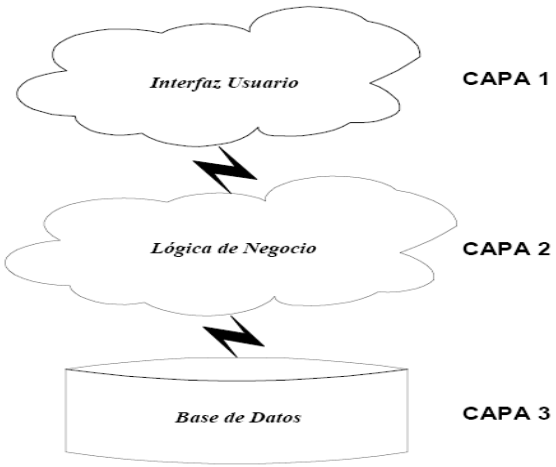

Figura 1.3.- Aplicación en 3 capas

La división de procesos y su distribución entre diferentes es conocido como **Procesamiento Distribuido**. Generalizando estos procesos dentro de estas tres categorías o capas es una distribución lógica y no refleja necesariamente alguna opción de diseño físico sobre computadoras, terminales u otros equipos. Se puede desarrollar una aplicación cliente/servidor distribuida basada sobre estas tres capas: Presentación, Lógica de Negocios y Acceso de Datos y tener la aplicación entera corriendo sobre una simple computadora. Alternativamente, se puede distribuir estas tres capas a través de un gran número de diferentes computadoras sobre una red.

1.3.1.- CAPA DE PRESENTACIÓN

La capa de Presentación provee a su aplicación la interfaz de usuario (IU). Aquí es donde su aplicación presenta información a los usuarios y acepta entradas o respuestas del usuario para ser usado por el programa que puede interactuar con otros usuarios, caplicaciones externas o servicios. Esto es importante, especialmente hoy en día, debido a que es muy común para una aplicación tener múltiples IU, o para los clientes o usuarios, que solicitan que elimine una IU y la remplace con otra. Por ejemplo, se puede desarrollar una aplicación Win32 (un programa en Visual Basic) y entonces solicitar remplazarla con una página HTLM., quizás usando tecnología ASP.

1.3.2.- CAPA DE NEGOCIOS

Toda aplicación tiene código para implementar reglas de negocios, procesos relacionados a los datos o cálculos como también otras actividades relativas a los negocios. Ellos se convierten en un modelo de las entidades de negocios conjuntamente con sus interrelaciones. Esto incluye tanto objetos físicos como conceptos abstractos. Estos son algunos ejemplos de objetos del mundo real: un empleado, un cliente, un producto, una orden de compra.

Todos estos son objetos en el mundo físico, y la idea en su totalidad detrás de usar objetos de negocios de software, es crear una representación de los mismos objetos dentro de la aplicación. Las aplicaciones pueden hacer que estos objetos interactúen unos con otros como ellos lo hacen en el mundo real. Por ejemplo, un empleado puede crear una orden de compra a un cliente que contiene una lista de productos. Siguiendo esta lógica se puede crear objetos de negocios de una orden, conteniendo el código necesario para administrarse a si mismo, así nunca necesitará replicar código para crear órdenes, solo se usará el objeto. Un buen diseño de un objeto cliente contiene todos los datos y rutinas necesitadas para representarlo a través del negocio completo, y puede ser usado a través de toda la aplicación de ese negocio.

1.3.3.- CAPA DE ACCESO A DATOS

Muchas aplicaciones interactúan con datos que ejecutan las reglas de datos relacionales y los almacenan en alguna forma de bases de datos. Hay algunas funciones básicas que son comunes a todos los procesos. Estas incluyen:

- Crear datos
- Leer datos
- Actualizar datos
- y Eliminar datos.

Adicionalmente, se cuenta con servicios más avanzados, tales como: búsquedas, ordenamientos, filtrados, etc.

Al crear una aplicación distribuida se debe cumplir con los siguientes objetivos:

- La aplicación debe solucionar el problema empresarial para la que se fue diseña.
- Tener en consideración la seguridad desde el principio, teniendo en cuenta los mecanismos adecuados de autenticación, la lógica de autorización y la comunicación segura.
- Proporcionar un alto rendimiento y estar optimizada para operaciones frecuentes entre patrones de implementación.
- Estar disponible y ser resistente, capaz de implementarse en centros de datos de alta disponibilidad y redundantes.
- Permitir la escalabilidad para cumplir las expectativas de la demanda y admitir un gran número de actividades y usuarios con el mínimo uso de recursos.
- Proveer una administración que permita a los administradores y operadores, implementar, supervisar y resolver los problemas de la aplicación en función del escenario.
- Funcionar en distintos escenarios de aplicaciones y patrones de implementación.

1.4.- TIPOS DE COMPONENTES EN APLICACIONES DISTRIBUIDAS

Para que una arquitectura de componentes pueda operar es necesario disponer de un entorno normalizado que proporcione soporte a los mecanismos con los que se comunican las diferentes interfaces. El desarrollo basado en componentes resuelve muchos de los problemas asociados con las aplicaciones monolíticas.

Todas las soluciones de software contienen tipos de componentes similares, independientemente de las necesidades empresariales que se desea cubrir, es así que en la mayoría de aplicaciones contienen componentes que tienen acceso a datos, encapsulan reglas empresariales y controla la ínter

actuación del usuario. La identificación y diseño de los tipos de componentes que se encuentran normalmente en las soluciones de software distribuido, facilitará el diseño de aplicaciones o servicios.

Existen una variedad de componentes utilizados en modelos realizados por componentes por capas, los componentes de software mas utilizados en la mayoría de soluciones distribuidas se describen a continuación.

1.4.1.- COMPONENTES DE INTERFAZ DE USUARIO (IU).

La mayoría de soluciones necesitan ofrecer al usuario un modo de interactuar con la aplicación. Las interfaces de usuario se implementan utilizando formularios de Windows Forms o Web Form, paginas Microsoft ASP.Net o a su vez paginas JSP, con el API Java Swing, controles u otro tipo de tecnología que permita procesar y dar formato a los datos de los usuarios, así como adquirir y validar datos entrantes procedentes de estos.

1.4.2.- COMPONENTES DE PROCESO DE USUARIO

En un gran número de interacciones el usuario realiza un proceso predecible Por ejemplo en una aplicación comercial se puede implementar un proceso que muestre los productos, permitiendo al usuario seleccionar un producto de una serie de categorías y entre esta seleccionar el producto deseado.

1.4.3. - FLUJO DE TRABAJO EMPRESARIAL

Una vez que el usuario ha recopilado todos los datos necesarios, estos se pueden utilizar para realizar un proceso empresarial siguiendo varios procesos los cuales deben ser organizados y llevar a cabo en un orden determinado.

Por ejemplo el uso de la tarjeta de crédito necesita calcular el valor a pagar, validar la información de la tarjeta de crédito, procesar el pago de la misma e imprimir el recibo.

1.4.4. COMPONENTES EMPRESARIALES

Independientemente de si el proceso empresarial consta de uno o varios procesos la aplicación requerirá el uso de componentes que implementen reglas empresariales y realicen tareas empresariales. Por ejemplo al emitir un recibo se deberá implementar una funcionalidad que calcule el precio total del pedido y agregue el costo adicional como el valor del IVA.

1.4.5.- COMPONENTES LÓGICOS DE ACCESO A DATOS

La mayoría de aplicaciones y servicios necesitan obtener acceso de una base de datos en un momento determinado. Por ejemplo si una aplicación empresarial necesita recuperar los datos de los productos de una base de datos para mostrar al usuario los detalles de los mismos para hacer un pedido. Por tanto, es razonable tener la lógica necesaria para obtener acceso a los datos en una capa independiente de componentes lógicos de acceso de base de datos, ya que de este modo se centraliza la funcionalidad de acceso a datos y se facilita la configuración y el mantenimiento de la misma.

1.5.- CARACTERÍSTICAS DE Java 2, Enterprise Edition (J2EE) PARA APLICACIONES DISTRIBUIDAS.

En JAVA, es habitual realizar una división de la arquitectura de las aplicaciones en tres capas funcionales para desarrollar aplicaciones distribuidas que trabajen en forma transaccional, conservando niveles de rapidez, seguridad y escalabilidad. La estructura tradicional de dos capas (cliente/servidor) ha ido evolucionando en estructuras más complejas, formadas por más capas, en las cuales se busca una división clara de los roles que forman parte de los componentes de las aplicaciones, contribuyendo a una mejor reutilización, crecimiento y mantenimiento de los sistemas existentes en este modelo en la primera capa se encuentra el usuario con un navegador Web, este invoca a la capa del negocio que se ejecuta sobre un hardware y que a su vez acceden a datos, estas tres capas mas usuales son:

- Capa de Presentación
- Capa de Lógica de Negocios
- Capa de Acceso a Datos J2EE

1.5.1.- CAPA DE PRESENTACIÓN

La Capa de Presentación está formada por la interfaz de usuario que puede constar de la seguridad y autentificación de usuarios y la validación de entrada que se maneja en esta capa.

La capa esta formada por interfaces la cual envía o recibe datos de la capa de la lógica de negocios, la capa de presentación normalmente consta de componentes de J2EE como, por ejemplo, componentes de Java Servlet o componentes de JSP que preparan datos para el envío en formato HTML o XML o que reciben solicitudes de procesamiento.

Esta capa también puede incluir un servicio de portal que proporcione acceso personalizado y seguro a un servicio de negocios de la capa de servicio de negocios.

1.5.2.- CAPA DE LÓGICA DE NEGOCIOS

La capa de negocios tiene como función generar respuestas a preguntas de la capa de presentación esto implica la generación de las funciones principales de la aplicación como: procesamiento de datos, implementación de funciones de negocios, coordinación de varios usuarios y administración de recursos externos como, por ejemplo, bases de datos o sistemas heredados. Normalmente, esta capa consta de componentes bien acoplados que se ajustan al modelo de componente distribuido J2EE, como los objetos Java, los componentes EJB o los Beans controlados por mensajes. Pueden montarse componentes J2EE individuales para ofrecer servicios de negocios complejos, como, por ejemplo, un servicio de inventario o uno de cálculo de impuestos, los componentes individuales y los montajes de servicios pueden encapsularse en forma de servicio Web acoplados libremente que en un modelo de arquitectura orientado al servicio y que se ajustan a los estándares de interfaz de SOAP (Simple Object Access Protocol).

1.5.3.- CAPA DE ACCESO A DATOS

La lógica de acceso a datos describe las transacciones con una base de datos o un servidor del directorio y este acceso se lo realiza mediante la capa de lógica de negocios.

Las descripciones hacen referencia a componentes de aplicaciones implementados con el modelo de componente J2EE esta lógica esta manejada por el conjunto de API estándares de JDBC.

1.6.- CARACTERÍSTICAS DE .NET PARA APLICACIONES DISTRIBUIDAS.

1.6.1 FILOSOFIA Y VENTAJAS

El principio fundamental de las aplicaciones distribuidas es la división lógica de una aplicación en tres capas fundamentales:

- Servicios de Presentación
- Lógica empresarial
- Acceso a datos y almacenamiento

Los programadores pueden generar aplicaciones altamente escalables y flexibles, organizando las aplicaciones según estos criterios. Para ello, deberán usar técnicas de programación basadas en componentes y emplear al máximo las características de la plataforma .NET y del sistema operativo Microsoft Windows.

Un modelo simple de aplicación distribuida consiste en un cliente que comunica con la capa intermedia, que a su vez es el servidor de aplicaciones y una aplicación que contiene la lógica empresarial. La aplicación, a su vez, comunica con una base de datos que suministra y almacena datos.

1.6.2.- SERVICIOS DE PRESENTACIÓN

La capa de presentación incluye tanto una interfaz de cliente enriquecida (o sencilla) como una aplicación. El cliente enriquecido proporciona al sistema operativo ya sea directamente mediante la API Win32 de Microsoft o indirectamente mediante formularios Windows Forms; una interfaz completa de programación, y usa ampliamente sus componentes.

El cliente sencillo (explorador Web) se está convirtiendo rápidamente en la interfaz preferida de muchos programadores. Un programador puede generar

lógica empresarial que pueda ejecutarse en cualquiera de los tres niveles de la aplicación. Con las aplicaciones para Web ASP.NET y los servicios Web XML, el cliente sencillo puede proporcionar a las aplicaciones una interfaz de usuario visualmente rica, flexible e interactiva. Los clientes sencillos también tienen la ventaja de aportar un mayor grado de portabilidad entre plataformas.

1.6.3.- LÓGICA EMPRESARIAL /SERVICIO DE LA APLICACIÓN

Esta capa se divide en servidores de aplicaciones y servicios, disponibles para ayudar a los clientes. Las aplicaciones Web pueden escribirse para sacar partido de los servicios de directorio y de los servicios de seguridad mediante .NET Framework. Los servicios de la aplicación, por el contrario, pueden interactuar con multitud de servicios en la capa de acceso a datos.

1.6.4.- ACCESO A DATOS Y SERVICIOS DE ALMACENAMIENTO

Los servicios de datos que admiten acceso y almacenamiento de datos son:

- ADO.NET, que proporciona un acceso simplificado a los datos mediante programación, ya sea por medio de lenguajes de secuencias de comandos o de programación.
- OLE DB, que es un proveedor de datos universal desarrollado por Microsoft.
- XML, que es un formato estándar para especificar estructuras de datos.
- XML es un estándar escogido por la comunidad ínter nauta. Mientras que HTML se centra en el modo en el que el explorador procesa la información y la presenta en pantalla, el objetivo de XML es manejar la estructura de datos y su representación.

1.6.5.- SERVICIOS DEL SISTEMA

.NET Framework y el sistema operativo Windows aceptan en su totalidad los elementos que conforman cada segmento de este modelo. Entre sus muchos servicios se encuentran los de directorio, seguridad, administración y comunicaciones, que operan en las tres capas. Las herramientas de programación, que incluyen el sistema de desarrollo Visual Studio .NET, permiten a los programadores generar componentes de aplicaciones entre las diferentes capas.

CAPITULO II

2.- TECNOLOGÍA JAVA 2 ENTERPRISE EDITION (J2EE)

2.1.- INTRODUCCIÓN A LA TECNOLOGÍA J2EE

La historia de J2EE comienza en el año 1995, cuando la compañía norteamericana Sun Microsystems introduce al mercado la primera plataforma de software universal diseñada desde y para el crecimiento de Internet y de las Intranets corporativas.

Sun realiza un cambio de paradigma, con Java las cosas cambian, es una plataforma de software, que independiente del hardware, se aloja en el sistema mientras exista el elemento conocido como Máquina Virtual de Java (MVJ, o JVM en inglés), es quién hace posible que una aplicación desarrollada en Java se ejecute en cualquier sistema operativo que soporte esta máquina virtual, como por ejemplo Unix, Linux, Macintosh, Windows, Solaris, entre otros.

Se le denomina plataforma porque proporciona técnicas específicas que describen el lenguaje pero, además, provee de herramientas para implementar productos software (aplicaciones) basados en dichas especificaciones.

El núcleo de la API (Application Programming Interface) viene con cada una de las implementaciones de la MVJ: tipos de datos, clases y objetos, applets, manejo de la red (networking), seguridad, y componentes (Java Beans). La principal tarea de MVJ es garantizar la portabilidad de las aplicaciones Java y especificar las instrucciones, llamadas bytecodes, que se pueden ejecutar. El editor crea un archivo cuya extensión es .java, luego el compilador lo convierte en los archivos cuya extensión es .class, y MVJ ejecuta esos archivos.

Java, como lenguaje de programación, reúne todas las características de un ambiente orientado a objetos: es sencillo, cuenta con capacidad de generación

de aplicaciones distribuidas, es robusto, seguro, de arquitectura neutral, portable, multihilo, dinámico y de alto rendimiento.

Otras características de J2EE que no se debe olvidar son:

- La plataforma J2EE esta compuesta por un conjunto de estándares abiertos como: EJB, JSP, Servlets, JDBC, RMI etc.

- La recolección de basura y el manejo automático de excepciones reduce el problema de que un componente bloquee la operación del servidor o de otro componente.

Sin embargo J2EE por ser una arquitectura distribuida dan lugar a una serie de problemas entre los cuales se puede citar:

- Problemas de perfomance, por las invocaciones remotas que se realizan son más lentas que las locales.

- Complejidad, al ser las aplicaciones distribuidas son duras de desarrollar, de poner en marcha, desplegar y de ofrecer su mantenimiento.

- Utilizando una arquitectura distribuida un diseño orientado a objetos no siempre es posible desarrollar su implementación.

2.2.- ARQUITECTURA DE JAVA 2 ENTERPRISE EDITION J2EE

Entre la arquitectura global de J2EE se puede dividir en dos componentes principales:

- Maquina Virtual de Java(JVM: Java Virtual Machine)

- La librería de clases de J2EE: entre las cuales tenemos:

- API: referente a las interfaces de usuario.
    - Java Swing: para aplicaciones de ventana.
- Tecnologías: para el desarrollo de aplicaciones Web: JSP, Servltes, EJB, JDBC

Java es un lenguaje de programación que unido a una librería de clases, compila a un código intermedio independiente de la máquina ("Bytecodes"). Este código se ejecutará en la máquina virtual, que proporciona un "entorno de ejecución" que transformará el lenguaje intermedio a código propio de la máquina en la que se ejecuta la aplicación ("Java Runtime Environment (JRE)"). Dispone de multitud de servicios que facilitan el trabajo de programación, entre ellos, el acceso a bases de datos, los servicios de directorio, las transacciones distribuidas.

A pesar de la gran portabilidad de J2EE, existe el problema de que J2EE es creado por varios fabricantes, también conlleva que las implementaciones de J2EE no son 100% compatibles entre si, debido a que cada vendedor ha realizado su propia interpretación y ha añadido nuevas características. Lo que si es cierto, es que todas las empresas que ofrecen sus productos basados en J2EE tienen versiones para los distintos sistemas operativos, por lo que una misma aplicación será portable entre los distintos sistemas operativos siempre y cuando se mantenga la solución del mismo vendedor.

2.3.- COMPONENTE JAVA CORE API (APPLICATION PROGRAM INTERFACE)

Java Core API (Application Program Interface) proporciona un conjunto de interfaces comunes a todas las plataformas que pueden trabajar con Java. Esta interfaz mejora bastante los sistemas de seguridad y permite que se formen las clases.

La API de Java viene con cada una de las implementaciones de la MVJ (Máquina Virtual de Java): tipos de datos, clases y objetos y componentes (Java Beans).

La Core API implementa funcionalidad básica relativa a seguridad, conectividad con fuentes de datos, conectividad de red, etc. Independientemente de la Core API se encuentran las APIs de escritorio, que permiten desarrollar aplicaciones para escritorio con interfaces gráficas potentes, basadas en la API Swing (Incluye todo desde botones hasta splitpanes o tablas.).

El API de Java presenta como características principales:

- En Java existen dos tipos de herencia sencilla y múltiple. Sencilla significa que sólo se hereda de una clase base, mientras que múltiple indica que se tiene varias clases base.

- Al utilizar la herencia aparecen dos conceptos**:** super y this, this representa al objeto completo, en cambio super, sólo representa la parte heredada de la clase base.

- Java Core API contiene una serie de reglas que posibilitan la detección y resolución de problemas como en la colisión de nombres de atributos. Si tiene el mismo nombre y diferentes parámetros: se produce sobrecarga de métodos permitiendo que existan varias maneras de llamar al mismo. Si solo cambia el valor devuelto: se da un error de complicación, indicando que no se pueden implementar los dos. Si coinciden en su declaración: se elimina uno de los dos, con lo que solo queda uno.

2.4.- INTERPRETE JAVA VIRTUAL MACHINE (JVM)

Cuando se escribe un programa en Java, con un entorno de desarrollo o un editor de texto, necesita ser compilado, generando un conjunto de instrucciones optimizadas denominadas programa bytecode. Este programa es independiente de la plataforma y no puede ser ejecutado por el procesador. En su lugar Java Virtual Machine ejecuta (interpreta) los bycodes.

Máquina Virtual Java (JVM) se encarga de la generación de código justo en el momento, es el intérprete de java que ejecuta los bytecodes.

La plataforma Java se encuentra por encima de otras plataformas, el código que generan sus compiladores no es específico de una maquina física en particular, sino de una máquina virtual. Aún cuando existen múltiples implantaciones de la Máquina Virtual Java, cada una específica de la plataforma sobre la cual subyace, existe una única especificación de la máquina virtual, que proporciona una vista independiente del hardware y del sistema operativo sobre el que se esté trabajando. De esta manera un programador en Java "escribe su programa una vez, y lo ejecuta donde sea".

La solución que propone Java es que el programa no corra sobre el hardware real, sino que corra sobre una maquina virtual. Es decir, dicha maquina no existe sino que es simulada por medio de software. Cada plataforma tendría instalada esta maquina virtual (que es un programa) y a la hora de correr los programas esta maquina virtual los correría. La maquina virtual actuaría como un interprete, el cual va leyendo el programa y ejecutándolo instrucción por instrucción sobre el hardware real.

Otra fortaleza de Java proviene de sus bibliotecas incorporados Los creadores del lenguaje Java Sun Microsystems distribuye gratuitamente el Java Development Kit (JDK) se compone de una biblioteca de clases estándar y una colección de utilidades para construir, probar y documentar los programas en

Java. Aunque JDK puede crear y mostrar aplicaciones graficas, su interfaz se basa en la línea de comandos. Por ejemplo, para ejecutar los programas hay escribir los comando en una ventana.

Los programas fuente de Java (.java) al compilarse producirían un código objeto (.class) constituido de las instrucciones de la maquina virtual (java byte codes). La maquina virtual ya incluye un set de librerías para las llamadas al sistema. Al hablar de la JVM es importante indicar el funcionamiento del compilador e interprete de Java:

- **El compilador Java (javac):** Compila programas escritos en el lenguaje de programación Java en *bytecodes*. Así, entonces, si tenemos un programa escrito en Java, que tiene un nombre con una extensión .java, para compilarlo solo hace falta hacer: *javac programa.java*. Al hacer esto y si la compilación es correcta, se generará un fichero 'programa.class'.

- **El intérprete de Java (java):** Ejecuta *bytecodes* de Java. Es decir, ejecuta programas escritos en el lenguaje de programación Java.

A continuación se describe ciertas ventajas y desventajas de JVM:

Ventajas Java Virtual Machine (JVM):

- Programas portables a cualquier plataforma (que cuente con una JVM).

- Diseño simple por no tener que ser backward compatible con nadie (que no era el caso de C++, ya que este tenía que ser backward compatible con C).

- Completamente orientado a objetos y con soporte de multithreading.

Ideal para Internet por producir programas pequeños

Desventajas Java Virtual Machine (JVM):

- Corre significativamente más despacio que si corriera en hardware real.

- Hay que aprender un lenguaje nuevo.

2.5.- TECNOLOGÍAS DE DESARROLLO DE APLICACIONES WEB: JAVASERVER PAGES (JSP) y SERVLETS

A lo largo de los años, la Web ha pasando de ser un mero contenedor de "texto" estático en forma de páginas HTML, a convertirse en un entorno dinámico, movido por un enorme escalafón de tecnologías que permiten y facilitan la creación de HTML en base a contenidos dinámicos provenientes de diversas fuentes, como pueden ser las bases de datos empresariales de una corporación.

Numerosas han sido, y son, las tecnologías encargadas de llevar a cabo esa transición entre la Web estática y una Web dinámica, donde cada día se sustituyen mas aplicaciones legacy por aplicaciones orientadas al Internet, gracias a la portabilidad que estas proveen (solo es necesario un navegador en el cliente) y la facilidad de mantenimiento, pues la aplicación se encuentra centralizada en el servidor Web.

2.5.1.- JAVA SERVER PAGES (JSP)

JSP es un acrónimo de Java Server Pages, que en castellano vendría a decir algo como Páginas de Servidor Java, los JSPs son archivos de texto que combina HTML y scripting de Java es decir básicamente es un archivo HTML con código Java intercalado, su extensión de fichero de una página JSP es ".jsp" en

vez de ".html" o ".htm", donde etiqueta <% identifica el inicio de un scriptlet, y la etiqueta %> identifica el final de un scriptlet.

Por lo tanto el contenido de una JSP se divide en dos categorías:

1) **Contenido Dinámico**, procesado en el contenedor Web (Código java).
2) **Contenido Estático,** HTML.

El objetivo de la tecnología JSP es proveer una separación entre la capa de presentación y la lógica de negocios: los diseñadores pueden diseñar y actualizar páginas sin aprender Java y los programadores java pueden escribir código sin ocuparse de los aspectos de diseño de las páginas**.**

Para ejecutar las páginas JSPs, se necesita un Servidor de Paginas Web con un contenedor Web que cumpla con las especificaciones JSP y Servlet, el contenedor Web se ejecuta en el servidor de paginas Web y maneja la ejecución de todas las páginas JSP y de los Servlets que se ejecutan en ese servidor Web indicando al servidor que esta pagina requiere un manejo especial que se conseguirá con una extensión del servidor o un plug-ins. Como se muestra en la Figura 2.1. Antes de que sean funcionales los archivos, el motor JSP lleva a cabo una fase de traducción de esa página en un servlet, implementado en un archivo .java y luego en un archivo class (Byte codes de Java). Esta fase de traducción se lleva a cabo habitualmente cuando se recibe la primera solicitud de la página .jsp.

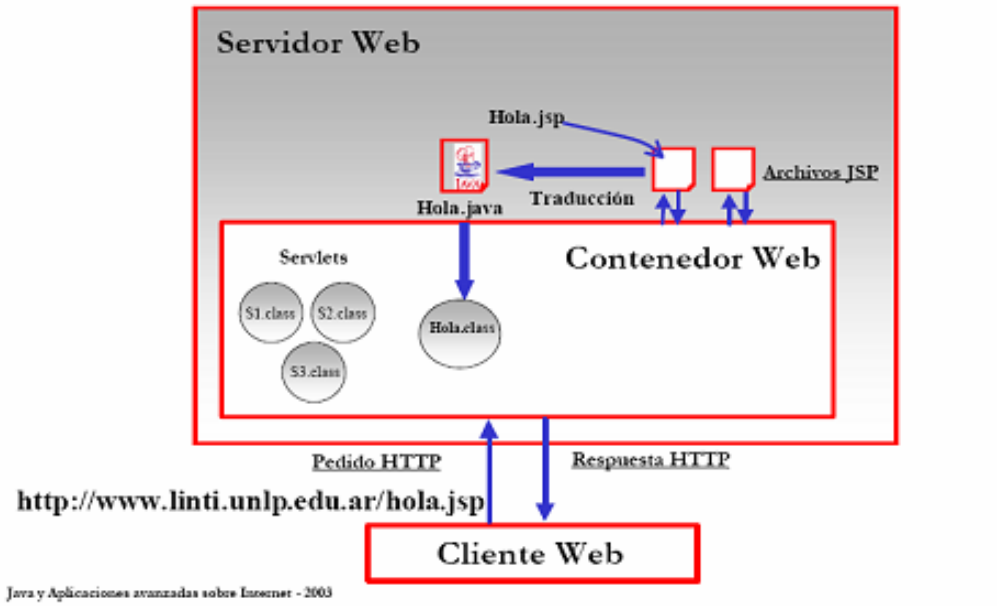

Figura 2.1.- Funcionamiento de JSP.

Continuando con el funcionamiento como se muestra en la figura Figura 2.1. Una de las partes más comunes en aplicaciones es un formulario HTML donde el usuario introduce alguna información usando JSP, los datos del formulario (la información que el usuario introduce en él) se almacenan en un objeto request (pedido) que es enviado desde el navegador hasta el contenedor JSP. La petición es procesada y el resultado se envía a través de un objeto response (respuesta) de vuelta al navegador. Estos dos objetos están disponibles implícitamente para nosotros.

Internamente cuando se llame a la página .jsp el motor de JSP la compilara en un Servlet Java. En este momento el Servlet es manejado por el motor Servlet como cualquier otro Servlet. El motor servlet carga la clase servlet usando un cargador de clases y la ejecuta para crear HTML dinámico para enviarlo al navegador. La siguiente vez que solicite la pagina, el motor JSP ejecuta el Servlet y lo carga a menos que la pagina JSP haya cambiado en cuyo caso es automáticamente recompilada en un Servlet y ejecutado.

2.5.1.2.- JAVA SERVER PAGE (JSP) y JAVABEANS

JavaBeans es una clase Java que sigue un conjunto de guías de estilo para permitir la creación de componentes de Java reutilizables, son utilizados en La representación del estado interno de la aplicación, uno de los aspectos mas poderosos del uso de JSP es la posibilidad de usar JavaBeans.

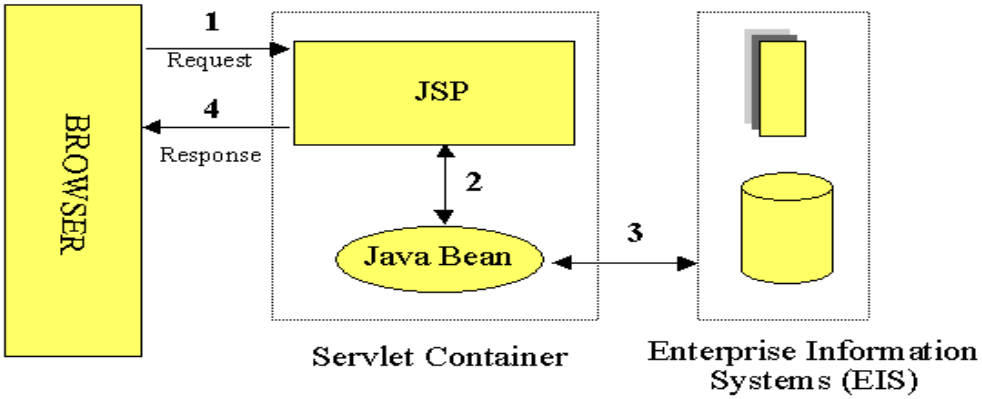

Figura 2.2.- Funcionamiento de un Java Beans y JSP

En la Figura 2.2 se muestra el funcionamiento JSP junto a un JavaBeans, se realiza una petición mediante el método request (petición) al contexto de JSP, los JavaBeans contienen la lógica de negocio que devuelve datos a un script en una página JSP, que a su vez formatea los datos devueltos por medio del método response (respuesta) y estos son visualizados en el navegador.

Hay muchos beneficios en la utilización de JavaBeans para mejorar las páginas JSP:

- **Componentes Reutilizables:** diferentes aplicaciones pueden reutilizar los mismos componentes.
- **Separación de la lógica de negocio de la lógica de presentación:** se puede modificar la forma de mostrar los datos sin que afecte la lógica del negocio.

VENTAJAS DE JSP SOBRE OTRAS TECNOLOGIAS

- **Contra Active Server Pages** *(ASP).* Las ventajas de JSP están duplicadas. **Primero,** la parte dinámica está escrita en Java, no en Visual Basic, otro lenguaje específico de MS, por eso es mucho más poderosa y fácil de usar. **Segundo**, es portable a otros sistemas operativos y servidores Web.

- **Contra los Servlets**. Es más conveniente escribir y modificar HTML normal que tener que hacer un billón de sentencias println que generen HTML. Además, separando el formato del contenido se puede poner diferentes personas en diferentes tareas: los expertos en diseño de páginas Web pueden construir el HTML, dejando espacio para que los programadores de servlets inserten el contenido dinámico.

- **Contra Server-Side Includes (SSI)**. SSI es una tecnología ampliamente soportada que incluye piezas definidas externamente dentro de una página Web estática. JSP es mejor porque permite usar servlets en vez de un programa separado para generar las partes dinámicas. Además, SSI, realmente está diseñado para inclusiones sencillas, no para programas "reales" que usen formularios de datos, hagan conexiones a bases de datos, etc.

- **Contra JavaScript**. Tiene una capacidad útil, pero sólo maneja situaciones donde la información dinámica está basada en el entorno del cliente. Con la excepción de las cookies, el HTTP y el envió de formularios no están disponibles con JavaScript. Y, como se ejecuta en el cliente, JavaScript no puede acceder a los recursos en el lado del servidor, como bases de datos, catálogos etc.

2.5.2.- SERVLET

Un servlet se puede considerar como una evolución de los CGIs desarrollada por SUN Microsystems como parte de la tecnología JAVA en si un servlet se puede definir como un componente Web escrito en Java gerenciada por un Contenedor Web (La palabra *servlet* deriva de otra anterior, *applet*, que se refería a pequeños programas escritos en Java que se ejecutan en el contexto de un navegador Web) Los clientes pueden invocarlo mediante el protocolo HTTP, el servlet es cargado y ejecutado por el servidor Web ya que estos aportan una manera fácil para que el servidor se comunique con el lado del cliente.

Así en su uso más habitual, un servlet acepta peticiones de cliente, procesa la información y devuelve los resultados, que podrán ser visualizados mediante applets, páginas HTML, etc. De forma general consiste en la ejecución de aplicaciones Java en el motor de servlets (Servlet engine) el cual hace parte del servidor Web, algo que lo hace ventajoso con respecto a los CGIs es que por cada petición de usuario no se crea un proceso sino un hilo, el cual es mucho mas económico para el sistema. Esta tecnología hace parte de la arquitectura propuesta por SUN en su plataforma J2EE (Java 2 Enterprise Edition).

Cada fabricante tiene su propia implementación de API de servlets. El Contenedor Web es responsable de:

- La conectividad con la red
- Capturar los requerimientos HTTP, traducirlos a objetos que el servlet lo entienda, entregarles dichos objetos al servlet quien lo utiliza para producir las respuestas y generar una respuesta http en un formato correcto.
- Gerenciar el ciclo de vida del servlet.
- Manejar errores
- Proveer seguridad

2.5.2.1.- Ciclo de Vida de un Servlet

El Contenedor Web gerencia el ciclo de vida de cada servlet, invocando a tres métodos definidos en la interfase java.servlet.Servlet:init,service y destroy.

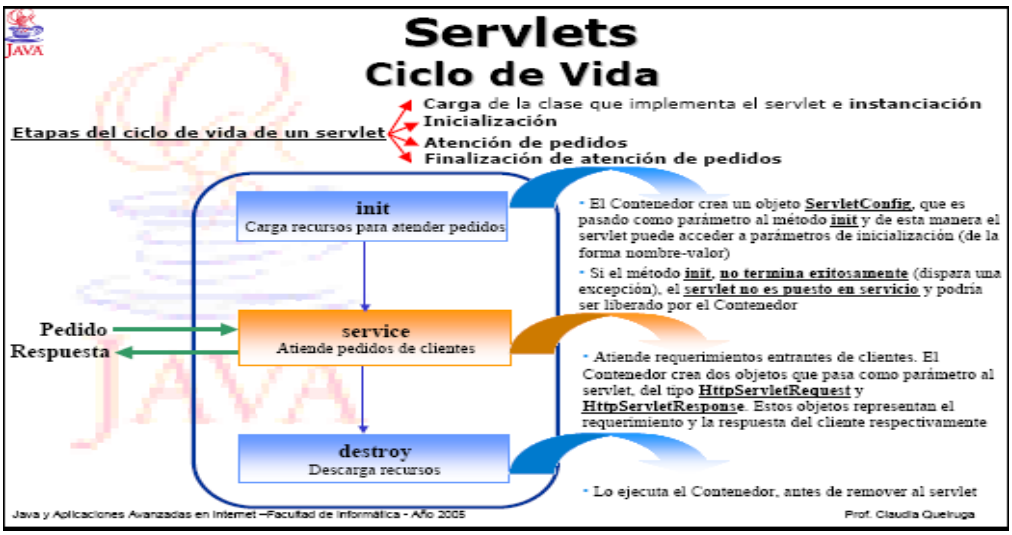

Figura 2.3.- Ciclo de vida de un Servlet

En la Figura 2.3 se muestra claramente el ciclo de un servlet .La interfaz que define esta estructura es javax.servlet.*. El ciclo de vida de servlet es muy sencillo define métodos: primero es construido inicializado con init () de esta manera el servlet puede acceder a parámetros de inicialización, luego se procesan las peticiones con el método service () se atiende los requerimientos de cliente mediante los métodos Http.ServletRequest y Http.ServletResponse que representan el requerimiento y respuesta respectivamente y por ultimo se ejecuta el servlet antes de ser destruido por el método destroy (). El javax.servlet.http**.***, que añade funcionalidad específica para HttpServlets, como son las sesiones y las cookies.

2.5.2.2.- Ventajas de un Servlet sobre CGI.

- Al estar escritos en Java son independientes de la plataforma, orientados a objetos, extensibles y más fiables. Tanto Java como la API de los servlet manipulan los datos con su tipo nativo, existe un recolector de basura y no existen punteros, lo cual protege al servlet de problemas de manejo de memoria.

- Consumen menos recursos del sistema servidor: los servidores Web cargan el servlet como un único proceso cuando se realiza la primera petición de servicio.

- Para sucesivas peticiones, crearán hilos de ejecución (Threads) de este proceso.

- Son más rápidos que los CGI esto es debido por un lado a que se precompilan, y por otro a que no se tiene que generar un nuevo proceso independiente cada vez que se invocan.

- Los servlets son persistentes, esto significa que un servlet puede mantener servicios, como por ejemplo una conexión a una base de datos, entre peticiones, lo cual agiliza su ejecución.

- Son seguros y portables, debido a que se ejecutan dentro de una máquina virtual Java (JVM) en el servidor. Como se ejecutan dentro del servidor Web, pueden interactuar muy estrechamente con el servidor y realizar tareas que no son posibles con scripts CGI.

2.6.- TECNOLOGÍA JAVA SWING

JAVA AWT es la librería visual mas antigua de java, usando esta librería, se podrán construir los tres tipos de programas más comunes como son FRAME,

WINDOW y APPLET. JAVA SWING es la librería de componentes visuales mas nueva que proporciona java, usando esta librería se podrán construir los tres tipos de programas o aplicaciones que son JFRAME, JWINDOW y JAPPLET.

El paquete Swinges parte de la JFC (Java Foundation Classes) en la plataforma Java, JFC provee facilidades para ayudar a la gente a construir GUIs. Swing abarca componentes como botones, tablas, marcos, etc., los componentes se identifican porque pertenecen al paquete *javax.swing*.

Antes de la existencia de Swing, las interfaces gráficas con el usuario se realizaban a través de AWT (Abstract Window Toolkit), de quien Swing hereda todo el manejo de eventos. Usualmente, para toda componente AWT existe una componente Swing que la reemplaza, por ejemplo, la clase Button de AWT es reemplazada por la clase JButton de Swing (el nombre de todas las componentes Swing comienza con "J").

Las componentes de Swing utilizan la infraestructura de AWT, incluyendo el modelo de eventos AWT, el cual rige cómo una componente reacciona a eventos tales como, eventos de teclado, mouse, etc. Es por esto, que la mayoría de los programas Swing necesitan importar dos paquetes AWT: *java.awt.** y *java.awt.event.**.

Como una regla, los programas no deben usar componentes pesados de AWT junto a componentes Swing, ya que los componentes de AWT son siempre pintados sobre los de Swing. (Por componentes pesadas de AWT se entiende Menú, ScrollPane y todas las componentes que heredan de las clases Canvas y Panel de AWT).

Las clases de swing se parecen mucho al las AWT todas las clases de AWT tienen una nueva versión de Swing con el prefijo J, así la clase Panel del AWT tiene una JPanel en Swing esto se cumple para todas las clases menos para Choice, Cavas, FileDialog y ScrollPane.

El API Swing es poderoso, flexible e inmenso. Pero la mayoría de los programas sencillos sólo usan un subconjunto de este API ofreciendo el código más común y guiándonos por los métodos y clases que podrías necesitar. La mayoría del código de esta sección usa sólo uno o dos los cuales son:

- **javax.swing**
- **javax.swing.event** (no siempre es necesario).

2.6.1.- DIFERENCIA DE LOS COMPONENTES SWING SOBRE LOS COMPONENTES ABSTRACT WINDOW TOOLKIT (AWT).

La mayor diferencia entre los componentes AWT y los componentes Swing es que éstos últimos están implementados sin nada de código nativo. Esto significa que los componentes Swing pueden tener más funcionalidad que los componentes AWT, porque no están restringidas al denominador común las características presentes en cada plataforma. El no tener código nativo también permite que los componentes Swing sean vendidos como añadidos al JDK en lugar de sólo formar parte del JDK.

Incluso el más sencillo de los componentes Swing tiene capacidades que van más allá de lo que ofrecen los componentes AWT. Por ejemplo.

- Los botones y las etiquetas Swing pueden mostrar imágenes además del texto.

- Se pueden añadir o modificar fácilmente los bordes dibujados alrededor de casi cualquier componente Swing. Por ejemplo, es fácil poner una caja alrededor de un contenedor o una etiqueta.

- Se puede modificar fácilmente el comportamiento o la apariencia de un componente Swing llamando a métodos o creando una subclase.

- Los componentes Swing no tienen porque ser rectangulares. Por ejemplo, los botones pueden ser redondos.

- Existen contenedores anidados es decir un componente puede estar anidado a otro. Por ejemplo un grafico se puede anidar a una lista.

El paso de AWT a Swing es muy sencillo y no hay que descartar nada de lo que se haya hecho con el AWT. Afortunadamente, los programadores de Swing han tenido compasión y, en la mayoría de los casos es suficiente con añadir una "J" al componente AWT para que se convierta en un componente Swing.

2.7.- TECNOLOGÍA DE ACCESO A DATOS: JAVA DATA BASE CONNECTIVITY (JDBC)

JDBC es para Java lo que **ODBC** es para Windows. JDBC es una especificación de un conjunto de clases y métodos de operación que permiten a cualquier programa Java acceder a sistemas de bases de datos de forma homogénea. Que permite lanzar queries a una base de datos relacional que puede ser Oracle, Infomix, SyBase, etc.

En la Figura 2.4 se indica la conectividad JDBC a una base de datos, Lógicamente, al igual que ODBC, la aplicación de Java debe tener acceso a un driver JDBC adecuado suele ser un fichero .jar. Este driver es el que implementa la funcionalidad de todas las clases de acceso a datos y proporciona la comunicación entre el API JDBC y la base de datos real.

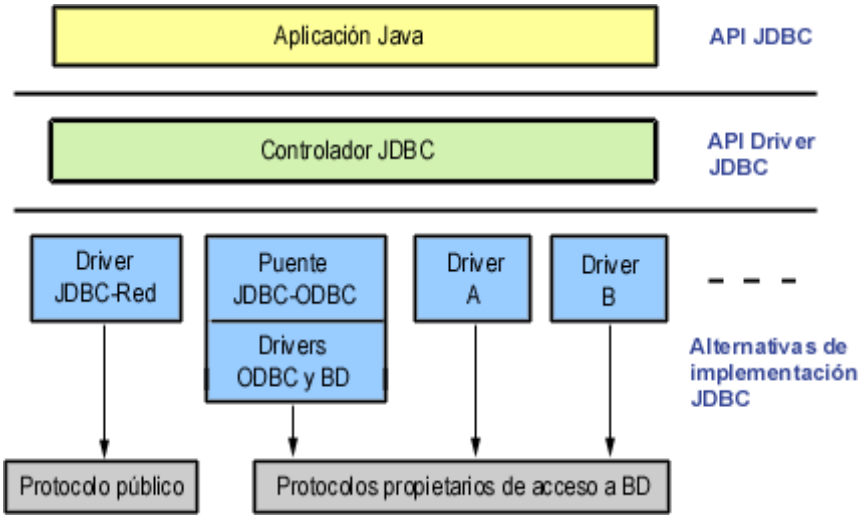

Figura 2.4.- Conectividad de JDBC a una base de datos.

La necesidad de JDBC, a pesar de la existencia de ODBC, viene dada porque ODBC es un interfaz escrito en lenguaje C, que al no ser un lenguaje portable, haría que las aplicaciones Java también perdiesen la portabilidad. Y además, ODBC tiene el inconveniente de que se ha de instalar manualmente en cada máquina; al contrario que los drivers JDBC, que al estar escritos en Java son automáticamente instalables, portables y seguros.

Básicamente un controlador JDBC realiza lo siguiente:

- Establece una conexión con una BD
- Envía sentencias SQL
- Procesa los resultados

Idealmente, si la aplicación cambia de BD, no necesitamos cambiar el código; simplemente, necesitamos otro drive. Sin embargo, desafortunadamente las BDs relacionales usan distintos dialectos de SQL (¡a pesar de que en teoría es un estándar!) como tipos de datos: varían mucho según la BD, Generación de identificadores: secuencias, auto numéricos, etc.

2.7.1.- ARQUITECTURA DE JAVA DATABASE CONNECTIVITY (JDBC)

La Figura 2.5 indica como JavaSoft proporciona tres componentes JDBC como la parte de la JDK: el JDBC driver manager, la JDBC driver test suite el puente JDBC-ODBC.

- El JDBC driver manager es la columna vertebral de la arquitectura de JDBC. Realmente es bastante pequeño y simple; su función es conectar las aplicaciones de Java al chofer de JDBC correcto.
- La JDBC driver test suite proporciona un poco de confianza en que drivers de JDBC ejecutarán su programa.
- El puente de JDBC-ODBC les permite a los drivers de ODBC ser usado como drivers de JDBC. Y a largo plazo proporcionará una manera de acceder alguno del DBMSs menos popular si no se crean los drivers de JDBC para ellos.

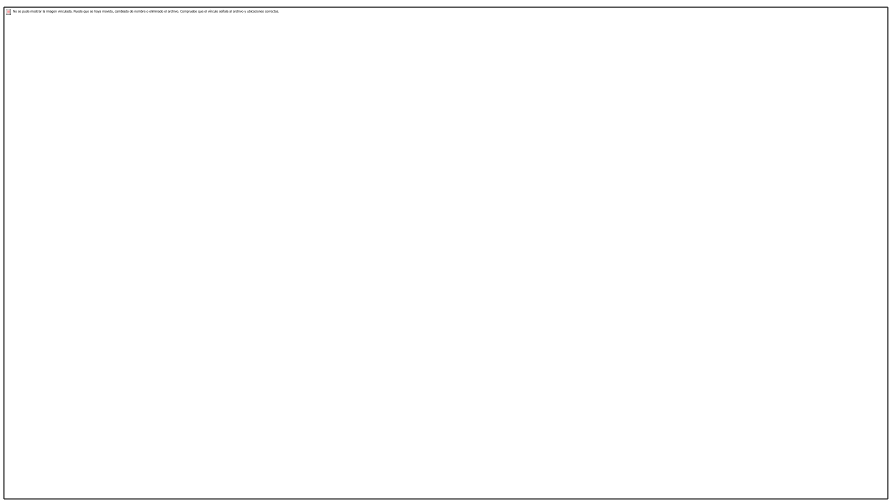

Figura 2.5.- Arquitectura de un JDBC

2.7.2.- TECNOLOGÍAS PARA EL DESARROLLO DE SERVICIOS WEB

Web Services, como indica su propio nombre, son servicios ofertados vía Web. Representan la solución con más futuro para la integración de aplicaciones en Internet y una buena fácil alternativa para integrar aplicaciones sencillas (Ej.: intercambio de información) en una intranet, los mismo que cuentan con el apoyo de todos los fabricantes, incluyendo a Microsoft.

Su funcionamiento esta basado en un escenario típico de Web Services, una aplicación de negocio envía una petición vía HTTP a un servicio situado en una URL. El servicio recibe la petición, la procesa y devuelve una respuesta también sobre HTTP.

La idea es sencilla pero requiere:

- Un protocolo de intercambio de mensajes petición/respuesta sobre HTTP.

- Una forma de que clientes y proveedores puedan interactuar a través de los mensajes, es decir, un lenguaje de especificación de interfaces.

Se ha optado por utilizar **SOAP** (Simple Objet Access Protocol) como protocolo de intercambio de mensajes. Es un protocolo sencillo basado en XML y estandarizado por el W3C (es un **consorcio** internacional que produce **estándares** para la **World Wide Web**).

El lenguaje de especificación de interfaces utilizado en servicios Web es **WSDL** (Web Services Description Language). WSDL permite especificar en XML las operaciones y tipos de datos de un servicio Web. Así, aunque el cliente y el servidor estén escritos en lenguajes distintos (por tanto, con sintaxis y tipos de

datos diferentes) pueden interactuar al utilizar un lenguaje neutral para comunicarse.

Una petición de un servicio Web constaría de los siguientes pasos:

1. En el cliente elabora una petición de una operación con parámetros.
2. La petición se transforma a formato XML utilizando WSDL.
3. La petición transformada se envía por medio HTTP utilizando SOAP.
4. El servidor de servicios Web recibe la petición.
5. El servidor determina que operación debe realizarse y transforma los parámetros de formato XML a su representación correspondiente en el lenguaje utilizado para implementar el servidor.
6. El servidor invoca la operación con los parámetros enviados, elabora una respuesta y se la envía al cliente de la misma forma que se ha explicado

Para implementar servicios Web existen varias APIs (comerciales y gratuitas) para los lenguajes más usuales. Las APIs no son estándares pero esto no afecta a la interoperabilidad ya que la misma es posible porque los protocolos (SOAP, WSDL, UDDI) están estandarizados.

Se puede distinguir dos tipos de APIs:

- *APIs de programación basadas en mensajes*: tanto el receptor como el emisor disponen de un proveedor de mensajería. Permiten mensajes síncronos y asíncronos, enviar a más de un receptor y calidad de servicio. Son APIs muy potentes pero complejas.

- *APIs de programación basadas en RPCs*: en el caso de un lenguaje orientado a objetos, este tipo de APIs permiten definir interfaces (en WSDL) cuyos métodos se pueden invocar remotamente (similar a CORBA). No es tan potente como las APIs basadas en mensajes pero es mucho más sencilla de usar.

2.8.- TECNOLOGÍA DE COMPONENTES: ENTERPRISE JAVA BEANS

Enterprise Java Beans es una arquitectura que define la forma de construir componentes distribuidos del lado del servidor. Esta tecnología garantiza que los componentes programados sean escalables, eficaces y seguro, se ejecuta dentro de un entorno de ejecución conocido como contenedor EJB el cual es un entorno de ejecución para controlar Beans Enterprise, el contenedor almacena y maneja los componentes, de la misma manera que un motor servlet hospeda un servlet Java, el contenedor EJB ejemplariza y controla los beans enterprise y les proporciona servicios de bajo nivel.

Un "Java Bean": Es un componente utilizado en Java que permite agrupar funcionalidades para formar parte de una aplicación, esto puede ser: un "Java Bean" agrupando información personal, datos sobre un pedido, requerimientos de ordenes. etc.

Un "Enterprise Java Bean": también agrupa funcionalidades para una aplicación, sin embargo, a diferencia de un "Java Bean" un "Enterprise Java Bean" es un "deployable component", este término implica que existe un ambiente de ejecución, éste ambiente es precisamente un "EJB (Enterprise Java Bean) Container" parte de un **java Aplication Server**.

Los EJB se ejecutan dentro de un servidor de aplicaciones y se encargan de gestionar recursos como red, los pool de conexiones, o la gestión de la seguridad. La tecnología EJB utiliza componentes basados en Java, del lado del servidor, este marco de trabajo facilita el manejo centralizado de la lógica de negocio.

2.9.- ESPECIFICACIÓN DE LA TECNOLOGÍA ENTERPRISE JAVA BEANS.

Los EJB son componentes software que se ejecutan en la parte servidora que incluye los mecanismos de retrollamadas EJBContext, JNDI ENC así como un estricto conjunto de reglas que describe como se comportaran los Beans Enterpriese y sus contenedores en tiempo de ejecución, como se comprobaran accesos de seguridad, como se manejaran las transacciones etc., de una aplicación y pueden ser ejecutados en un entorno multi-capa distribuido.

Los EJB son componentes que pueden estar físicamente alejados del cliente por ello se dice que su interfaz publica es remota, Remote Interface, la especificación EJB hace uso de RMI/IIOP (interface RMI de Java sobre el sistema CORBA) para ofrecer esta característica, RMI/IIOP todo objeto remoto dispone de un interfaz accesible a los clientes, el cliente puede hacer uso de esta interfaz a través del "stub" que es un proxy enviado al cliente.

Cuando el cliente invoca al "stub" este se comunica con el "skeleton", que es le proxy situado en el lado del servidor. Una vez recibido el mensaje por el "skeleton" este es el encargado de comunicárselo al objeto distribuido, el cual resolverá la operación y dará la respuesta al cliente a través del "skeleton" primero y el "stub" después.

Tanto el "stub" como el "skeleton" son transparentes para el cliente y este siempre tiene la sensación de estar invocando al objeto distribuido directamente.

En este tipo de operaciones es bastante pesada, requiere abrir sockets, transferir objetos a través de la red y transformar estos objetos en datos binarios según el protocolo (en este caso IIOP). Los EJB en vez de escribir llamadas a una API propietario directamente en los componentes de aplicación desarrollan especificaciones de requisitos de middleware mediante opciones de tiempo de ejecución definiendo el soporte a otros lenguajes y CORBA es el encargado de

manejar los aspectos relacionados con el protocolo de comunicaciones siendo el protocolo de comunicación entre componentes RMI sin embargo existen especificaiones que proporcionan interfaces con CORBAy soporte para DCOM...

EJB ofrece todas las características de Java (portabilidad, seguridad, gestión automática de memoria) a la capa intermedia permitiendo el desarrollo de objetos de negocio de servidor reusables y proporcionando programación basada en atributos para definir dinámicamente el comportamiento transaccional, seguridad.

2.10.- CONTENEDOR ENTERPRISE JAVA BEANS.

Los Beans Enterprise se ejecutan en un entorno especial llamado un contenedor EJB el mismo que contiene y maneja un Bean Enterprise, controla cada aspecto del Bean Enterprise en tiempo de ejecución incluyendo accesos remotos al bean, seguridad, persistencia, transacciones, concurrencia, y accesos a un almacén de recursos.

El contenedor aísla al bean enterprise de accesos directos por parte de aplicaciones cliente, cuando este invoca un método remoto de un bean enterprise, el contenedor primero intercepta la llamada para asegurar que la persistencia, las transacciones, y la seguridad son aplicadas apropiadamente a cada operación que el cliente realiza en el Bean manejando estos aspectos de forma automática, el desarrollador de beans enterprise puede enfocarse en encapsular las reglas del negocio, mientras el contenedor se ocupa de todo lo demás.

El Bean Enterprise interactúa con su contenedor a través de uno de estos tres mecanismos:

1. **Métodos de Retrollamada**
2. **EJBContext**

   Java Naming and Directory Interface (JNDI)

2.10.1.- COMPOSICIÓN DE JAVA ENERPRISE BEAN

Para crear un componente EJB del lado del servidor, hay que proporcionar dos interfaces que definen los métodos de negocio del bean, además de la implementación real de la clase bean. Entonces el cliente usa una interfase pública del bean para crear, manipular, y eliminar beans del servidor EJB, los clientes a través de la red usando sus interfaces remoto y home los que proporcionan todos los métodos necesarios para crear, actualizar, borrar e interactuar con el bean.

Interfaces Remoto y Home

Los interfaces remoto y home representan al bean, pero el contenedor aisla a los beans de accesos directos desde aplicaciones cliente. Cada vez que un bean es solicitado, creado, o borrado, el contenedor maneja todo el proceso.

- **La interfase home**: representa los métodos del ciclo de vida del componente (crear, destruir, encontrar). Los clientes usan la interfase home del bean para obtener referencias al interface remoto del bean.

- **Interfase remota:** define los métodos de negocio del bean; los métodos que son específicos para el concepto de negocio que representa. Los interfaces remotos son subclases del interfase **javax.ejb.EJBObject** que es una subclase de interfase **java.rmi.Remote**.

2.11.- TIPOS DE ENTERPRISE JAVA BEANS.

Se define tres tipos de beans enterprise distintos:

- Beans dirigidos por mensaje
- Beans de sesión
- Beans de entidad

A los beans de sesión y de entidad se les invoca síncronamente mediante un cliente bean enterprise. A los beans dirigidos por mensaje (MDBs) los invoca un contenedor de mensaje, como un tópico publicar/subscribir.

2.11.1.- BEANS DIRIGIDOS POR MENSAJE

Un bean dirigido por mensaje (MDB) es un bean enterprise que maneja de forma asíncrona los mensajes entrantes, actúa como un oyente de mensajes enviados desde un tópico publicar/suscribir. Un MDB puede recibir mensajes compatibles con JMS.

Un contenedor EJB maneja el ciclo de vida de un MDB, sin embargo, al contrario que los beans de sesión y entidad; los programas clientes no se dirigen al MDB con llamadas a métodos. En vez de eso, el MDB recibe mensajes de una fuente de mensajes a través de un método de retrollamada, onMessage.

2.11.2.- BEANS DE SESIÓN

Un bean de sesión representa un único cliente y no se comparte entre clientes. Un cliente llama a los métodos del bean de sesión, que son dirigidos a través del contenedor EJB al bean enterprise. El bean de sesión realiza la lógica de negocio para el cliente y el contenedor devuelve el control al cliente. Un bean de sesión no persiste entre varias sesiones. Hay dos tipos de beans de sesión:

2.11.1.1.- Beans de Sesión con Estado

Un bean de sesión con estado mantiene un estado conversacional con un cliente durante una sesión, implica que el bean de sesión con estado puede mantener ejemplares de variables a través de varias llamadas desde un cliente durante una única sesión. Cuando el cliente termina de interactuar con el bean enterprise y el contenedor EJB lo elimina, termina la sesión del bean y se borran todos los datos de estado del bean.

2.11.1.2.- Beans de Sesión sin Estado

Un bean de sesión sin estado no mantiene un estado conversacional para cada cliente individual. Cada llamada a un bean de sesión sin estado debería considerarse como una petición a un ejemplar totalmente nuevo, ya que cualquier estado de las variables de ejemplar se perderá entre dos llamadas.

2.11.1.3.- Beans De Entidad

Un bean de entidad representa la lógica de negocio de una entidad existente en un almacenamiento persistente, como una base de datos relacional, los mismos comparten algunas cualidades que podría encontrar en una base de datos relacional, por ejemplo:

- Los beans de entidad son persistentes: el estado de un bean de entidad existe más allá del ciclo de vida de la aplicación en la que se ha creado, o incluso más allá del ciclo de vida del contendor EJB. Esto implica que un contenedor EJB puede restaurar el bean de entidad a su estado original.

- Los beans de entidad permiten compartir accesos: los beans podrían compartirse entre varios clientes y la concurrencia la maneja el contenedor.

- Los beans de entidad tienen claves primarias: Existen clases Primary-key para identificar un ejemplar de un bean de entidad. La clave primaria contiene toda la información necesaria para encontrar un bean almacenado.

- Los beans de entidad podrían participar en relaciones: Se han presentado interfaces locales para manejar las relaciones entre beans.

CAPITULO III

3.- TECNOLOGÍA PUNTO NET

3.1.- INTRODUCCIÓN A LA TECNOLOGIA .NET

Visual Basic .NET no es el mismo lenguaje que el Visual Basic conocido en el pasado, cuya última versión fue la 6. Es falsa la idea de que Visual Basic .NET es la nueva versión de Visual Basic, pues se trata de un lenguaje completamente diferente. Net es una plataforma software, la cual proporciona un entorno de desarrollo independiente del lenguaje, que permite escribir programas de forma sencilla, e incluso permite combinar código escrito en diferentes lenguajes, esta plataforma es la estrategia de Microsoft para el desarrollo, despliegue y ejecución de aplicaciones orientadas a servicio, creando aplicaciones que se ejecutan de modo distribuido, a lo largo de todo Internet que día a día cada vez es más rápida y a un costo menor de la mano de la capacidad de los procesadores que continuamente son mas rápidos.

El poder acceder a las aplicaciones desde cualquier sitio y desde cualquier dispositivo, es una aspiración de Microsoft, que con la tecnología .NET puede llevarse a cabo de forma sencilla, mediante la utilización de servicios Web, cuyas características como independencia de lenguaje, soporte de bases de datos, XML, servicios Web y aplicaciones Web es la respuesta de Microsoft a Java, aunque tiene bastantes diferencias.

3.2.- ARQUITECTURA .NET

Visual Studio .NET perfila el proceso de desarrollo suministrando una plataforma de herramientas colaborativas que están en un número alrededor de 130 partners, que proporcionan actualmente recursos integrados incluyendo herramientas, lenguajes de programación y libros relacionados con Visual Studio .

Sin embargo, la programación en estos lenguajes está intrínsecamente ligada a la llamada plataforma .NET (o ".NET framework", en inglés) que se presenta brevemente a continuación.

La plataforma .NET es un conjunto de componentes software (programas y librerías) que se usa para compilar y ejecutar programas escritos en los lenguajes .NET. Su estructura se refleja en la Figura 3.1. En esta figura, los componentes de la plataforma están en color (los componentes en blanco no forman parte de la plataforma .NET, pero se incluyen para indicar cómo se relacionan con ella).

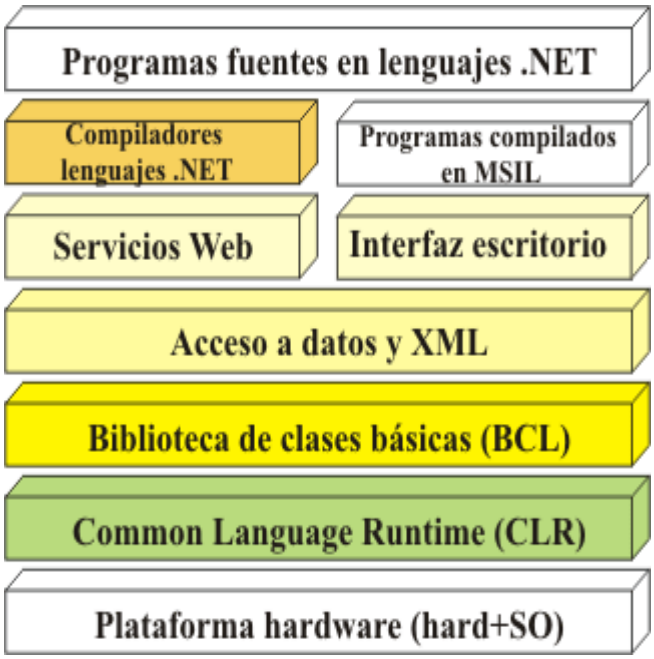

Figura 3.1. Arquitectura de la plataforma .NET (".NET Framework")

Como se muestra en la Figura 3.1, la plataforma .NET se ejecuta sobre la plataforma hardware, uno de sus componentes es el CLR. Todos los otros componentes se ejecutan sobre el CLR y de esta forma son independientes de la plataforma hardware (siempre que ésta sea Windows).

Sobre el CLR se ejecutan una serie de librerías (en amarillo en la figura) que son utilizadas por los programas, simplificando el desarrollo al ofrecer una

serie de servicios ya programados, listos para reutilizar. Hay una librería básica que contiene soporte para estructuras de datos, interoperabilidad con el código .NET, entre otros servicios. Hay librerías para acceso a bases de datos y XML así como para interfaz gráfica, tanto para el escritorio como para el Web.

Estas librerías son usadas por los programas compilados (escritos en lenguaje MSIL). Entre estos programas se destacan a los compiladores para los diferentes lenguajes .NET, que traducen los programas en cada uno de estos lenguajes a programas compilados MSIL.

3.3.- COMPONENTES: NET FRAMEWORK

El Framework de .Net es una infraestructura sobre la que se reúne todo un conjunto de lenguajes, servicios tecnologías necesarias para realizar aplicaciones que pueden comunicarse fácilmente en el Internet (o sobre cualquier red como intranet) usando estándares abiertos como el XML y SOAP. Mediante esta herramienta se ofrece un entorno de ejecución altamente distribuido, que permite crear aplicaciones robustas y escalables.

Actualmente, el Framework de .Net es una plataforma no incluida en los diferentes sistemas operativos distribuidos por Microsoft, por lo que es necesaria su instalación previa a la ejecución de programas creados mediante .Net. El Framework se puede descargar gratuitamente desde la Web oficial de Microsoft.

Los principales componentes como se muestra en la Figura 3.2 de este entorno son:

- Lenguajes de compilación
- Biblioteca de clases de .Net
- CLR (Common Language Runtime)

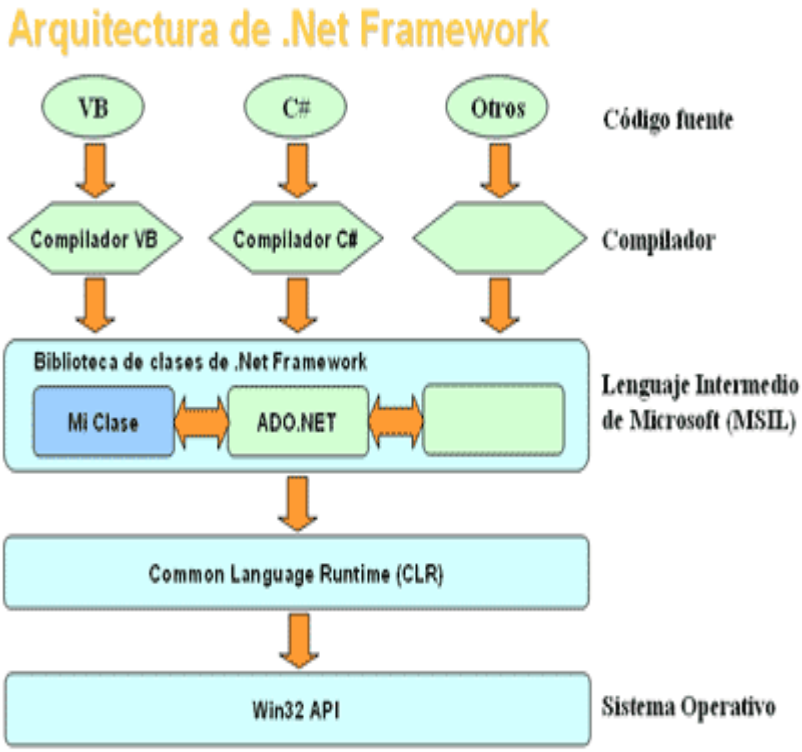

Figura3.2 Arquitectura de .Net Framework

3.3.1.- LENGUAJES DE COMPILACIÓN

.Net Framework soporta múltiples lenguajes de programación y aunque cada lenguaje tiene sus características propias, es posible desarrollar aplicaciones en cualquiera de estos lenguajes, los más conocidos como C# (C Sharp), Visual Basic o C++ y otros como Perl o Cobol.

3.3.2.- BIBLIOTECA DE CLASES .NET

Estas clases sirven para realizar acciones como manipulación de archivos, acceso a datos, conocer el estado del sistema, implementar seguridad, etc. Para ello, el Framework posee un sistema de tipos universal, denominado Common Type System (CTS). Este sistema permite que el programador pueda interactuar los tipos que se incluyen en el propio Framework (biblioteca de clases de .Net) con los creados por él mismo (clases). De esta forma se aprovechan las ventajas propias de la programación orientada a objetos, como la herencia de clases

predefinidas para crear nuevas clases, o el polimorfismo para modificar o ampliar funcionalidades de clases ya existentes.

La biblioteca de clases de .Net Framework incluye, entre otros, tres componentes clave:

- ASP.NET para construir aplicaciones y servicios Web.
- Windows Forms para desarrollar interfaces de usuario.
- ADO.NET para conectar las aplicaciones a bases de datos.

La forma de organizar la biblioteca de clases de .Net dentro del código es a través de los espacios de nombres (namespaces), donde cada clase está organizada en espacios de nombres según su funcionalidad. Por ejemplo, para manejar ficheros se utiliza el espacio de nombres System.IO y si es acceso a la base se utilizará System.Data,m de esta forma se tiene toda la biblioteca de clases de .Net centralizada bajo el mismo espacio de nombres (System). Además, desde cualquier lenguaje se usa la misma sintaxis de invocación, ya que a todos los lenguajes se aplica la misma biblioteca de clases.

3.4.- COMMON LANGUAJE RUTINE (CLR)

El CLR es el verdadero núcleo del Framework de .Net, ya que es el entorno de ejecución en el que se cargan las aplicaciones desarrolladas en los distintos lenguajes, ampliando el conjunto de servicios que ofrece el sistema operativo estándar Win32.

Entre las características más importantes del Framework de Net se tiene:

- Una plataforma diseñada desde un principio para escribir aplicaciones optimas orientadas al Internet y además que adopten estándares abiertos como XML, HTTP y SOAP.

- Tiene tecnologías muy poderosas para el desarrollo de aplicaciones, esta es el Windows Forms para construir aplicaciones graficas clásicas.

- Permite desarrollar aplicaciones en la que el programador se sienta confortable y productivo.

- Se ejecuta independiente y permitiendo la gestión del medio ambiente denominado Common Lenguaje Runtime, que garantiza que el código sea seguro, da una capa de abstracción en la cima del sistema operativo.

3.5.- **INTÉRPRETE:** COMMON LANGUAJE RUTINE **(CLR)**

CRL es el núcleo de la plataforma .NET, el motor encargado de gestionar la ejecución de las aplicaciones para el Framework, es como una maquina virtual que se encarga de gestionar la ejecución del código y proporcionar una serie de servicios al mismo, por esta razón se lo suele llamar código gestionado, y al código no escrito para ser ejecutado directamente en la plataforma .NET se le suele llamar código no gestionado.

CLR compila el código fuente de cualquiera de los lenguajes soportados por .Net en un mismo código, denominado código intermedio (MSIL, Microsoft Intermediate Lenguaje).

Para generar dicho código el compilador se basa en el Common Language Specification (CLS) que determina las reglas necesarias para crear código MSIL compatible con el CLR el mismo define un subconjunto del CTS (Common Type System) el cual soporta la integración entre lenguajes y el cual evita áreas conflictivas como sobrecarga de operadores, y está diseñado para ser suficientemente flexible y potente como para permitir la integración de un amplio número de lenguajes de programación

De esta forma, indistintamente de la herramienta de desarrollo utilizada y del lenguaje elegido, el código generado es siempre el mismo, debido a que el MSIL es el único lenguaje que entiende directamente el CLR. Este código es transparente al desarrollo de la aplicación ya que lo genera automáticamente el compilador.

Sin embargo, el código generado en MSIL no es código máquina y por tanto no puede ejecutarse directamente. Se necesita un segundo paso en el que una herramienta denominada compilador JIT (Just-In-Time) genera el código máquina real que se ejecuta en la plataforma que tenga la computadora, la compilación JIT la realiza el CLR a medida que se invocan los métodos en el programa y, el código ejecutable obtenido, se almacena en la memoria caché de la computadora, siendo recompilado sólo cuando se produce algún cambio en el código fuente.

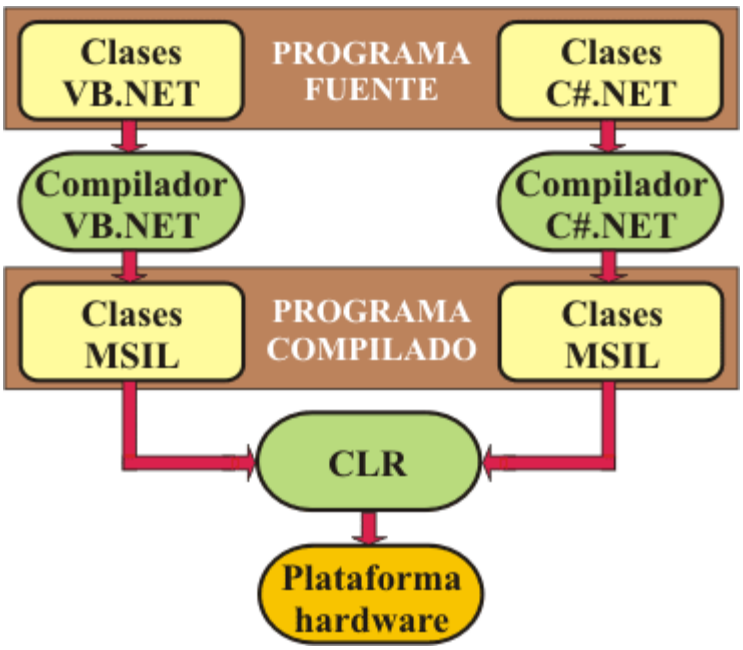

Figura 3.3. Compilación y ejecución de un programa en varios lenguajes .NET

Como se ve en la Figura 3.3, el programa contiene clases escritas en Visual Basic .NET y en C# .NET. Estos son dos lenguajes diferentes, pero, al ser compilados, los dos producen clases MSIL, las cuales pueden ser ejecutadas

conjuntamente por el CLR. De esta forma, se puede tener programas realizados en varios lenguajes simultáneamente (siempre que éstos sean .NET) y estos funcionan tan correctamente como un programa tradicional con un único lenguaje.

Los servicios que ofrece el CLR son varios entre los cuales se puede citar a los siguientes:

- Cargador de Clases: Permite cargar en memoria las clases.

- Compilador MSIL a nativo: Transforma código intermedio de alto nivel independiente del hardware que lo ejecuta a código de máquina propio del dispositivo que lo ejecuta.

- Administrador de Código: Coordina toda la operación de los distintos subsistemas del Common Language Runtime.

- Recolector de Basura: Elimina de memoria objetos no utilizados.

- Motor de Seguridad: Administra la seguridad del código que se ejecuta.

- Motor de Depuración: Permite hacer un seguimiento de la ejecución del código aún cuando se utilicen lenguajes distintos.

- Verificador de Tipos: Controla que las variables de la aplicación usen el área de memoria que tienen asignado.

- Administrador de Excepciones: Maneja los errores que se producen durante la ejecución del código.

- Soporte de multiproceso (threads): Permite ejecutar código en forma paralela.

- Empaquetador de COM: Coordina la comunicación con los componentes COM para que puedan ser usados por el .NET Framework.

- Soporte de la Biblioteca de Clases Base: Interfase con las clases base del .NET Framework.

3.6.- TECNOLOGÍAS DE DESARROLLO DE APLICACIONES WEB: ASP.NET

Active Server Pages .NET es la infraestructura construida sobre .Net Framework que proporciona un modelo de desarrollo Web que incluye los servicios necesarios para la creación de aplicaciones Web, ASP.Net es más que una nueva evolución de las páginas Active Server (ASP). Si bien ASP.NET es en gran medida compatible con la sintaxis de ASP a las cuales se les pueden ampliar agregándoles funcionalidades de ASP.NET, proporciona una estructura para crear aplicaciones más escalables y estables que ayuden a proporcionar mayor protección es un entorno compilado basado en .NET, se puede crear aplicaciones utilizando:

1) Formularios Web Forms
2) Servicios Web XML o combinarlas de la manera que más les convenga.

Estas dos características son compatibles con la misma infraestructura, que permite utilizar esquemas de autenticación, almacenar en caché datos que se utilizan con frecuencia y personalizar la configuración de la aplicación, entre otras muchas cosas.

- Los formularios Web Forms permiten crear páginas Web basadas en formularios muy eficaces, se pueden usar controles de servidor ASP.NET para crear elementos comunes de la interfaz de usuario y programarlos para que realicen tareas comunes, creando con rapidez un formulario Web

Forms a partir de componentes integrados reutilizables o personalizados, con un código de página simplificado.

- Un servicio Web XML proporciona los medios para obtener acceso a la funcionalidad del servidor de manera remota. Con los servicios Web XML, las empresas pueden exponer interfaces de programación a sus datos o lógica empresarial, que, a su vez, pueden obtener y manipular las aplicaciones de cliente y servidor, permiten el intercambio de datos en escenarios cliente-servidor o servidor-servidor, utilizando estándares como los servicios de mensajería HTTP y XML para que los datos pasen los servidores de seguridad.

Características principales de ASP.NET :

- **Modelo de Desarrollo de Aplicaciones mejorado**
  - Amplio Soporte de Class Library
  - Namespaces Ej. System.Data – System.Web.Services

- **Performance**
  - Las páginas se compilan antes de ser ejecutadas
  - Compilar es mejor que interpretar
  - Soporta Caching permite reusar versión compilada si no existen cambios.

- **Escalabilidad**: Soporta ejecución de aplicaciones Web en escenarios Web Garden y Web Farm

- **Seguridad:** ASP.NET tiene acceso al Built-in Security del Framework que incluye:
  - Acceso de seguridad de código y roles
  - Varios modos de autentificación
    - Forms based

- Passport authetification

- **Manejabilidad:** Los archivos de configuración se guardan en XML.
- **Extensibilidad**: Permite escribir propios componentes

3.7.- TECNOLOGÍAS DE DESARROLLO DE APLICACIONES WINDOWS FORM

Windows Forms es la nueva plataforma de desarrollo de aplicaciones para Microsoft Windows, basada en .NET Framework que permite desarrollar complejas aplicaciones para Windows. Además, los formularios Windows Forms pueden actuar como interfaz de usuario local en una solución distribuida de varios niveles, es el sistema "oficial" para el desarrollo de aplicaciones de usuario con interfaz Windows, dentro de .Net que utiliza los mismos recursos y servicios del sistema que el resto del Framework como es el Compìlador JIT, lenguajes intermedio MSIL, independencia del lenguaje de programación y acceso a todas las facilidades del sistema, tales como Web Services, ADO .Net, gestión automática de la memoria, gestión de Threads entre otros, permitiendo construir aplicaciones de usuario con formularios y controles visuales, eventos, cuadros de dialogo, alertas que funcionaran en el escritorio del usuario, en cualquier explorador o dispositivo cliente e implementa lógica de aplicación mediante el código de la parte servidor. La salida de las páginas de formularios Web Forms puede contener casi cualquier lenguaje compatible con HTTP, incluidos HTML, XML, WML y ECMAScript (JScript, JavaScript).

De forma similar Windows Forms nos ofrece una abstracción, comparable a la librería Swing de Java, en la que se manejan elementos de interfaz de usuarios.

Las páginas de formularios Web Forms reúnen las siguientes características:

- Se basan en la tecnología Microsoft ASP.NET, en la que el código que se ejecuta en el servidor genera de forma dinámica salida de páginas Web en un explorador o dispositivo cliente.
- Compatible con cualquier explorador o dispositivo móvil. Las páginas de formularios Web Forms presentan automáticamente el código HTML adecuado al explorador para funciones tales como estilos, diseño.

- Admiten cualquier lenguaje compatible con Common Language Runtime de .NET, incluidos Microsoft Visual Basic, Microsoft Visual C# y Microsoft JScript.NET.

- Se crean en el entorno Microsoft .NET Framework. Esto proporciona todos los beneficios del marco de trabajo, incluidos un entorno administrado, seguridad de tipos y herencia.

- Respaldadas en Visual Studio por eficaces herramientas de desarrollo rápido de aplicaciones (RAD, *Rapid Application Development*) destinadas al diseño y la programación de los formularios.

3.8.- TECNOLOGÍA DE ACCESO A BASE DE DATOS: ACTIVEX DATA OBJECTS.NET (ADO.NET)

En el momento del desarrollo de una aplicación siempre se trata con datos los mismos que pueden provenir de bases de datos, archivos o de hojas de calculo, ADO.NET es una manera nueva de acceder a los datos es una tecnología de acceso a datos que se basa en los objetos ADO (Objetos de Datos ActiveX) que es su antecesor.

ADO.NET utiliza XML como el formato para transmitir datos desde y hacia su base de datos y su aplicación Web, es un conjunto de clases que constituyen la interfaz fundamental de las aplicaciones para proporcionar servicios de acceso

a datos de los diferentes proveedores de datos Entre estos proveedores de datos cabe destacar:

- Proveedor de datos de .NET Framework para SQL Server.
- Proveedor de datos de .NET Framework para OLE DB.
- Proveedor de datos de .NET Framework para ODBC.
- Proveedor de datos de .NET Framework para Oracle.

ADO.NET proporciona distintos tipos de acceso a datos. Si la aplicación Web o el servicio Web de XML necesita tener acceso a datos de múltiples orígenes, o ínter operar con otras aplicaciones (tanto locales como remotas), o puede beneficiarse de la persistencia y transferencia de resultados almacenados en memoria caché, el conjunto de datos es una opción excelente. Como alternativa, ADO.NET proporciona comandos de datos y lectores de datos para comunicarse directamente con el origen de datos. Las operaciones directas con la base de datos mediante comandos de datos y lectores de datos incluyen la ejecución de consultas y de procedimientos almacenados, la creación de objetos de base de datos y la actualización y eliminación directa utilizando comandos DDL.

Asimismo, ADO.NET maximiza el uso compartido de datos ya que admite formato de transmisión y persistencia basado en XML para el objeto fundamental de las aplicaciones ADO.NET distribuidas de datos. Un conjunto de datos es una estructura de datos relacionales de la que se puede leer, en la que se puede escribir o que se puede realizar utilizando XML. Los conjuntos de datos de ADO.NET facilitan la creación de aplicaciones que requieran un intercambio de datos de correspondencia imprecisa entre niveles de aplicaciones y varios sitios Web.

Como los conjuntos de datos son remotos al igual que XML, los dos componentes pueden compartir datos y utilizar esquemas XML para definir la

estructura relacional del conjunto de datos. Y, debido a que el formato de serialización del conjunto de datos es XML, los objetos DataSet pueden traspasar fácilmente los servidores de seguridad sin restricciones. Además de cargar datos de XML, los conjuntos de datos se pueden rellenar con y mantener los cambios de datos de SQL Server así como de orígenes de datos expuestos a través de OLE DB.

A continuación se detalla algunos aspectos importantes a cerca de ADO.NET.

- ADO.NET es una tecnología de acceso a datos que se basa en los objetos ADO (Objetos de Datos ActiveX) anteriores.

- Es una manera nueva de acceder a los datos construida sobre ADO. ADO.NET puede coexistir con ADO.

- También se puede decir que ADO.NET es un conjunto de clases que exponen servicios de acceso a datos al programador de .NET.

- ADO.NET proporciona un conjunto variado de componentes para crear aplicaciones distribuidas de uso compartido de datos.

- ADO.NET es compatible con diversas necesidades de programación, incluida la creación de clientes de bases de datos clientes y objetos empresariales de nivel medio utilizados por aplicaciones, herramientas, lenguajes o exploradores de Internet.

- ADO.NET utiliza un modelo de acceso pensado para entornos desconectados. Esto quiere decir que la aplicación se conecta al origen de datos, hace lo que tiene que hacer, por ejemplo seleccionar registros, los carga en memoria y se desconecta del origen de datos.

- ADO.NET es un conjunto de clases que se utiliza para acceder y manipular orígenes de datos como por ejemplo, una base de datos en SQL Server o una planilla Excel.

- ADO.NET utiliza XML como el formato para transmitir datos desde y hacia su base de datos y su aplicación Web.

- Hay 3 espacios de nombres que se importará en un formulario Web o formulario Windows si esta usando ADO.NET:
    - System.Data.
    - System.Data.SqlClient.
    - System.Data.OleDb.

- El modelo de objetos ADO.NET provee una estructura de acceso a distintos orígenes de datos. Tiene 2 componentes principales: El Dataset y el proveedor de Datos .NET.

3.9.- TECNOLOGÍA PARA EL DESARROLLO DE SERVICIOS: WEB SERVICES

Los servicios Web son la revolución informática de la nueva generación de aplicaciones que trabajan colaborativamente en las cuales el software esta distribuido en diferentes servidores, permiten trabajar a las empresas con sus aplicaciones en una forma novedosa e innovadora. Microsoft para conseguir este propósito con su tecnología .NET emplea como protocolo de comunicación, una aplicación XML, llamada SOAP.

El funcionamiento de los Servicios Web depende de varios protocolos estándares de la industria como por ejemplo HTTP, XML SOAP, WSDL, UDDI y muchos otros protocolos que permiten que las aplicaciones se comuniquen entre si.

3.9.1.- Lenguaje de Marcado Extensible (XML)

XML (Lenguaje de Marcado Extensible) y derivado en ingles Extensible Markup Language es un método estándar para representar datos, XML son los bloques de construcción básicos en la transición al proceso distribuido en Internet.

- Probablemente existan tantas definiciones de los Servicios Web XML como compañías que los desarrollan, pero casi todas las definiciones tienen estos aspectos comunes:

- Los Servicios Web XML exponen funcionalidad útil a los usuarios Web mediante un protocolo Web estándar. En la mayoría de casos, el protocolo utilizado es Simple Object Access Protocol (SOAP).

- Los Servicios Web XML proporcionan un modo de describir sus interfaces con suficiente detalle para permitir a un usuario construir una aplicación cliente para hablar con ellos. Esta descripción se proporciona generalmente en un documento XML que responde al nombre de documento Web Services Description Language (WSDL).

- Los Servicios Web XML se registran de modo que los potenciales usuarios puedan encontrarlos. Esto se realiza mediante Universal Discovery Description and Integration (UDDI).

Se ha definido un servicio Web XML como un servicio software expuesto en la Web mediante SOAP, descrito con un archivo WSDL y registrado en UDDI.

Es un protocolo que define el formato XML para los mensajes de intercambio en el uso de un Web Service. Para aquellos programadores que solían utilizar llamadas del tipo RPC, SOAP también las soporta. Adicionalmente es posible mediante SOAP definir un mensaje HTTP y este punto es de especial interés puesto que el protocolo imprescindible para Internet es HTTP.

3.9.2.- Object Access Protocol (SOAP)

Son las siglas de Simple Object Access Protocol, este protocolo deriva de un protocolo creado por David Winer, XML-RPC en 1998, este es el primer protocolo de comunicación bajo HTTP mediante XML. Con este protocolo se podían realizar RPC o remote procedure calls, es decir, se podía bien en cliente o servidor realizar peticiones mediante HTTP a un servidor Web. Los mensajes debían tener un formato determinado empleando XML para encapsular los parámetros de la petición.

Es un protocolo que permite mover los datos entre aplicaciones y sistemas, es el mecanismo por medio del cual los servicios Web son invocados e interactúan, en el núcleo de los servicios Web se encuentra el protocolo simple de acceso a datos SOAP, que proporciona un mecanismo estándar de empaquetar mensajes. SOAP ha recibido gran atención debido a que facilita una comunicación del estilo RPC entre un cliente y un servidor remoto. Pero existen multitud de protocolos creados para facilitar la comunicación entre aplicaciones, incluyendo RPC de Sun, DCE de Microsoft, RMI de Java y ORPC de CORBA.

Una de las razones principales para que SOAP sea tan popular se debe al inmenso apoyo que prácticamente han dado las grandes compañías de software del mundo. Compañías que en raras ocasiones cooperan entre sí están ofreciendo su apoyo a este protocolo, entre las cuales están Microsoft, IBM, SUN, Microsystems, SAP etc.

Algunas de las Ventajas de SOAP son:

- **No esta asociado con ningún lenguaje**: los desarrolladores involucrados en nuevos proyectos pueden elegir desarrollar con el último y mejor lenguaje de programación que exista pero los desarrolladores responsables de mantener antiguas aflicciones heredadas podrían no

poder hacer esta elección sobre el lenguaje de programación que utilizan. SOAP no especifica una API, por lo que la implementación de la API se deja al lenguaje de programación, como en Java, y la plataforma como Microsoft .Net.

- **No se encuentra fuertemente asociado a ningún protocolo de transporte**: La especificación de SOAP no describe como se deberían asociar los mensajes de SOAP con HTTP. Un mensaje de SOAP no es más que un documento XML, por lo que puede transportarse utilizando cualquier protocolo capaz de transmitir texto.

- **No está atado a ninguna infraestructura de objeto distribuido** La mayoría de los sistemas de objetos distribuidos se pueden extender, y ya está alguno de ellos para que admitan SOAP.

- **Permite la interoperabilidad entre múltiples entornos**: SOAP se desarrollo sobre los estándares existentes de la industria, por lo que las aplicaciones que se ejecuten en plataformas con dicho estándares pueden comunicarse mediante mensaje SOAP con aplicaciones que se ejecuten en otras plataformas.

- La ubicuidad de HTTP y la simplicidad de SOAP los convierten en una base ideal para implementar servicios Web XML que pueden consumirse desde prácticamente cualquier entorno.

3.9.3.- Web Services Description Language (WSDL)

Las siglas WSDL significan Web Services Description Language, que permite publicar, encontrar y usar los Servicios Web basados en XML; es la 'Página Amarilla' de los servicios Web es decir un directorio para poder encontrarlos. Puede ser accedido con un explorador o programáticamente ya que UDDI es también un servicio Web.

Para el propósito, se puede decir que un archivo WSDL es un documento XML que describe un conjunto de mensajes SOAP y cómo se realiza el intercambio de mensajes. Dicho de otro modo, WSDL es a SOAP lo que IDL es a CORBA o COM. Como WSDL es XML, es legible y editable pero en la mayoría de casos, se genera y se consume por parte de software.

La notación que utiliza un archivo WSDL para describir formatos de mensajes está basada en el estándar XML Schema lo cual significa que es neutral respecto del lenguaje de programación y que está basado en estándares, lo que lo hace apropiado para describir interfaces de servicios Web XML accesibles desde una amplia variedad de plataformas y lenguajes de programación.

Muchos kits de SOAP actuales incluyen herramientas que generan archivos WSDL desde interfaces de programas existentes, pero existen pocas herramientas para escribir directamente WSDL, y el soporte de WSDL por parte de las herramientas no es tan completo como debería ser. Las herramientas que generan archivos WSDL y que también generan proxies y stubs de modo similar a las herramientas IDL de COM no deberían tardar en formar parte de la mayoría de implementaciones SOAP. Llegados a este punto, WSDL se convertirá en el modo preferido de generar interfaces SOAP para los servicios Web XML.

3.9.4.- Lenguaje de Descripción de Servicios Web (UDDI)

Por medio del cual un servicio Web describe entre otras cosas qué hace o qué funcionalidad implementa.

- Una entrada de directorio UDDI es un archivo XML que describe un negocio y los servicios que ofrece. Una entrada en el directorio UDDI está formada por tres partes.

- Las "páginas blancas" describen la compañía que ofrece el servicio: nombre, dirección, contactos, etc.

- Las "páginas amarillas" incluyen categorías industriales basadas en taxonomías estándares como el North American Industry Classification System y el Standard Industrial Classification.

- Las "páginas verdes" describen el interfaz al servicio con suficiente detalle para alguien que desarrolle una aplicación que consuma el servicio Web.

El modo en que se definen los servicios es mediante un documento UDDI llamado Type Model o tModel. En muchos casos, tModel contiene un archivo WSDL que describe un interfaz SOAP a un servicio Web XML, pero tModel es suficientemente flexible para describir prácticamente cualquier tipo de servicio.

El directorio UDDI también incluye varias formas de buscar los servicios que se necesitan para construir las aplicaciones. Por ejemplo, se puede buscar los proveedores de un servicio en una ubicación geográfica específica o un negocio de un tipo específico. El directorio UDDI proporcionará información, contactos, enlaces e información técnica para permitir evaluar qué servicios satisfacen los requerimientos.

3.10.- DISEÑO DE INTERFAZ DE SERVICIOS UTILIZANDO SERVICIOS WEB

Una interfaz de servicios es una entidad de software implementada normalmente como una fachada que controla los servicios de asignación y transformación para permitir la comunicación con un servicio y aplica un proceso y una política de comunicación

Una interfaz de servicios expone métodos, a los que se puede llamar de forma individual o en una secuencia especifica para formar una conversación que implemente una tarea empresarial.

Se debe tener en cuenta las siguientes recomendaciones al diseñar interfaces de servicios:

- Considerar una interfaz de servicio como un límite de confianza de su aplicación.

- Si las interfaces de servicios se exponen a organizaciones y consumidores externos o se hacen publicas, se recomienda diseñarlas de modo que los cambios a la implementación interna no requieren un camino en la interfaz de usuario.

- Puede que sea necesario que clientes diferentes consuman de formas distintas la lógica empresarial del servicio, por lo que tal vez sea preciso publicar varias interfaces de servicios para la misma funcionalidad.

Características de Interfaces de Servicios.

Para el diseño de las interfaces de servicios se debe tomar en cuenta lo siguiente:

- Las interfaces de servicios pueden implementar almacenamiento de datos en caché, asignación y transformación de esquemas y formatos simples; sin embargo, estas fachadas no deben implementar la lógica empresarial.

- Se recomienda que se diseñe la interfaz de servicios de modo que se obtenga el nivel máximo de interoperabilidad con otras plataformas y servicios, basándose en los estándares para las aplicaciones de comunicación, seguridad.

- La interfaz de servicios dispondrá de su propia identidad de seguridad y autenticar los mensajes entrantes pero no se podrá suplantarlos. Se debe tomar en cuenta este enfoque al llamar a componentes empresariales distribuidos en un servidor distinto a la interfaz de servicios.

CAPITULO IV

4.- METODOLOGÍA UML

4.1.- DESARROLLO DE LA METODOLOGÍA UML

UML (*Unified Modeling Language*) es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software. Un requisito es una condición o capacidad que debe cumplir o poseer un sistema o componente de un sistema para satisfacer un contrato, Standard, o especificación o algún otro documento impuesto. El conjunto de requisitos forma la base para el desarrollo de un sistema o un componente de sistema.

Documento de Requisitos

Los requisitos de software es un documento formal que es usado para comunicar los requisitos a los clientes, ingenieros de sistemas, de software y gerentes del proceso de ingeniería de sistemas. No hay un nombre estándar para este documento. En diferentes organizaciones, el documento puede tener diferentes nombres tales como documentos de requisitos, especificación funcional, especificación de requisitos del sistema, etc.

4.2.- ANÁLISIS DEL SISTEMA

4.2.1.- ESPECIFICACIÓN DE REQUISITOS

4.2.1.1.- Introducción

Este documento es una Especificación de Requisitos Software (ERS) para el Sistema de Información SICADOC (Sistema de Control de Asistencia de Docentes a Clases). Este documento ha sido elaborado teniendo en cuenta las necesidades observadas de los usuarios, en la experiencia de las autoridades de la Carrera de Sistemas e Informática.

Esta especificación se ha estructurado inspirándose en las directrices dadas por el estándar “IEEE Recommended Practice for Software Requirements Specification ANSI/IEEE 830 1998”.

4.2.1.2.- Propósito

El objetivo de la especificación es definir de manera clara y precisa todas las funcionalidades y restricciones del sistema que se desea construir.

El documento va dirigido al grupo de control y supervisión así como también servirá de guía a la persona encargada del desarrollo del sistema SICADOC, el mismo servirá como canal de comunicación entre las partes implicadas, tomando parte en su confección miembros de cada parte.

Esta especificación está sujeta a revisiones por el grupo de usuarios, que se recogerán por medio de sucesivas versiones del documento, hasta alcanzar su aprobación por parte del grupo de control y supervisión del sistema SICADOC.

Una vez aprobado servirá de base a la persona encargada del desarrollo para la construcción del nuevo sistema.

4.1.1.- ÁMBITO DEL SISTEMA

El sistema se llamara SICADOC (Sistema de Control de Asistencia de los Docentes a Clases).

El motor que impulsa la realización del presente sistema, es con la finalidad de tener una mejor percepción para el análisis de las dos tecnologías de estudio, como también proporcionar a la Carrera de Sistemas e Informática de la Escuela Politécnica del Ejército sede Latacunga de un sistema para el Control de Asistencia de Docentes.

4.1.2.- DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS

4.1.2.1.- Definiciones

Administrador	Persona que tiene altos privilegios para administrar el sistema.
Departamento	Cada una de las grandes divisiones de la ESPE-L correspondiente a una rama del saber, y en la que se dan las enseñanzas de una carrera determinada o de varias carreras afines.
Carrera	Conjunto de estudios que habilitan para el ejercicio de una profesión.
Docente	Personal que dicta una materia y debe registrar su entrada y salida de clases.
Cédula	Identificación del usuario o personal para poder ingresar al sistema.
Password ó Contraseña	Código secreto y personal de un usuario para seguridad.
Materia	Cada una de las materias que se enseñan que forman parte a un plan académico de estudios
Horario	Cuadro indicador de las horas en que deben ejecutarse determinadas actividades

4.1.2.2 Acrónimos

ERS	Especificación de Requisitos Software

4.1.2.3.- Abreviaturas

SICADOC	Sistema de Control de Asistencia de Docentes a Clases.

4.1.3.- REFERENCIAS

IEEE Recommended Practice for Software Requirements Specification. ANSI/IEEE std. 830, 1998.

4.1.4.- VISIÓN GENERAL DEL DOCUMENTO

Este documento consta de tres secciones. En la primera sección es la Introducción y proporciona una visión general de la Especificación de Requisitos Software.

En la Sección 2 se da una descripción general del sistema, con el fin de conocer las principales funciones que debe realizar, los datos asociados y los factores, restricciones, supuestos y dependencias que afectan al desarrollo, sin entrar en excesivos detalles.

En la sección 3 se definen detalladamente los requisitos que debe satisfacer el sistema.

4.2.- DESCRIPCIÓN GENERAL

En esta sección se presenta una descripción a alto nivel del sistema. Se presentarán las principales áreas de negocio a las cuales el sistema debe dar soporte, las funciones que el sistema debe realizar, la información utilizada, las restricciones y otros factores que afecten al desarrollo del mismo.

4.2.1.- PERSPECTIVA DEL PRODUCTO

El sistema en esta primera versión, permitirá a los usuarios acceder al sistema desde sus ordenadores conectados a la red

4.2.2.- FUNCIONES DEL SISTEMA

- En términos generales, el sistema deberá proporcionar soporte a las siguientes tareas:
- Gestión de Período.
- Gestión de Departamentos
- Gestión de Carrera.
- Gestión de Área.

- Gestión de Materia.
- Gestión de Personal.
- Gestión de Asignar Materia.
- Gestión de Asistencia.
- Gestión de Usuarios.
- Gestión de Reportes.

A continuación, se describirán con más detalle éstas tareas, y como serán soportadas por el sistema.

4.2.2.1.- Gestión de Período

El administrador podrá ingresar un período, también podrá realizar modificaciones en los datos del período al cual se podrá designar materias, personal, departamento y carrera.

4.2.2.2 .- Gestión de Departamento

El sistema permitirá al administrador crear un Departamento. Es necesario crear el Departamento para la designación de Carreras a esta.

4.2.2.3 .- Gestión de Carrera

El sistema permitirá al administrador crear una Carrera. La misma que se encuentra relacionada con departamento, materia y el personal.

4.2.2.4.- Gestión de Área

El administrador podrá ingresar una nueva área también podrá realizar modificaciones en los datos de las áreas, este será necesario para asignar una materia a un área.

4.2.2.5.- Gestión de Materia

El administrador, coordinador podrán ingresar una nueva materia y también podrá realizar modificaciones en los datos de las materias, este será

necesario para asignar a un docente una materia en una carrera y en un período determinado.

4.2.2.6.- Gestión de Personal

El administrador, coordinador es el encargado de ingresar un nuevo personal también podrá realizar modificaciones y asignar a este una materia en un período.

4.2.2.7.- Gestión de Asignación de Materias

El administrador, coordinador es el encargado de asignar una materia, paralelo, día, hora inicio, hora de fin a un personal también podrá realizar modificaciones.

4.2.2.8 .- Gestión de Asistencia Asignación de Materias

El administrador, coordinador podrá registrar la hora de entrada y salida de clases. Para el registro de la asistencia el mismo deberá ingresar su número de cédula, el sistema proporcionara las materias de dicho docente y capturara la hora del registro que estará acorde a la hora del servidor.

4.2.2.9.- Gestión de Usuarios

El administrador podrá ingresar un nuevo usuario dentro de una dependencia para la administración de la documentación, además podrá realizar consultas individuales y generales sobre los usuarios, y también podrá realizar modificaciones datos de los usuarios o eliminarlos definitivamente.

El administrador podrá ingrear un nuevo usuario para cada carrera para la administración de Asistencia de Docentes podrá realizar consultas sobre los usuarios y también podrá realizar modificaciones datos de los usuarios o eliminarlos definitivamente.

4.2.2.10.- Gestión de Reportes

El administrador podrá obtener los siguientes reportes:

- Reporte del día del registro de entradas y salidas de clase de los Docentes.
- Reporte del registro de entradas y salidas de clase de los Docentes en un rango de fecha.
- Reporte por Docente de sus asistencias a clases en un rango de fecha.
- Reporte por Materia de la asistencia del Docente a clase en un rango de fecha.

4.2.3.- CARACTERÍSTICAS DE LOS USUARIOS

El sistema de información deberá ofrecer una interfaz de usuario intuitivo, fácil de aprender, sencillo de manejar y sobretodo amigable. El sistema deberá presentar un alto grado de usabilidad. Lo deseable sería que un usuario nuevo se familiarizase con el sistema en una o dos horas.

4.2.4.- RESTRICCIONES

Dado que el sistema implementará la política y los procesos de negocio actualmente vigentes en las diferentes Carreras, es de esperar que futuros cambios en los modos de trabajo o en las políticas, ejerzan un fuerte impacto sobre el sistema.

Los procesos que el sistema no podrá realizar, se describen a continuación:

- El sistema no realizara un control de asistencia de alumnos lo hará de forma textual no porcentual.
- El sistema no permitirá un reporte de avance de materia.
- El sistema solo permitirá designar horarios mas no generarlos.

En cuanto a las restricciones Hardware/Software, la institución exige que el sistema funcione bajo el paradigma cliente / servidor.

4.2.5.- SUPOSICIONES Y DEPENDENCIAS

4.2.5.1.- Suposiciones

Se asume que los requisitos escritos en este documento son estables una vez que sean aprobados.

Cualquier petición de cambios, en la especificación debe ser bien analizada por las partes involucradas del sistema.

4.2.5.2.- Dependencias

El sistema SICADOC funciona autónomamente, sin necesidad de comunicarse con otros sistemas, por lo que no hay dependencias respecto a otros sistemas.

El sistema seguirá una arquitectura Cliente/Servidor, por lo que la disponibilidad del sistema dependerá de la conexión entre las maquinas en las que residirá el programa cliente y la máquina servidora de datos.

Tanto el cliente como el servidor estarán instalados bajo el sistema operativo Windows y como servidores Web se utilizara Internet Information Server para .Net y Jakarta Tomkat para Java.

4.3 REQUISITOS ESPECÍFICOS

En este apartado se presentan los requisitos funcionales que deberán ser satisfechos por el sistema. Todos los requisitos aquí expuestos son

ESENCIALES, es decir, no sería aceptable un sistema que no satisfaga alguno de los requisitos aquí presentados. Estos requisitos se han especificado teniendo en cuenta, entre otros, el criterio de “testabilidad”: dado un requisito, debería ser fácilmente demostrable si es satisfecho o no por el sistema.

4.1.1.- REQUISITOS FUNCIONALES

4.1.1.1.- Gestión de Departamento

Req 01: El sistema permite al administrador crear un nuevo Departamento para esto ingresará los siguientes datos: Código, Nombre, Director y Descripción, el sistema comprobara que este no se encuentre creado.

Req 02: El sistema permite al administrador eliminar un departamento si fuere necesario.

Req 03: El sistema permitire al administrador modificar los datos de un departamento registrado.

4.1.1.2.- Gestión de Carrera

Req 04: El sistema permite al administrador crear una nueva Carrera para esto ingresará los siguientes datos: Código, Nombre, Coordinador y Descripción. El sistema comprobará que este no se encuentre creado.

Req 05: El sistema permite al administrador si fuere necesario eliminar una carrera siempre y cuando no exista datos relacionados.

Req 06: El sistema permitirá al administrador modificar los datos de una Carrera registrada.

4.1.1.3.- Gestión de Usuarios

Req 07: El sistema permite al administrador crear un usuario para esto ingresará los siguientes datos: Número de cédula, Nombres, Apellidos, Password y Nivel (Administrador, Coordinador). El administrador tendrá aceso a todas las opciones del Sistema y el Coordinador solo tendra aceso a: Docente, Materia, Asignación de horarios y Reportes.

El sistema comprobará que este no se encuentre creado.

Req 08: El sistema permitirá al administrador dar de baja a un usuario siempre y cuando no exista datos relacionados.

Req 09: El sistema permitirá al administrador modificar los datos de un usuario registrado, exepto el Número de cédula.

4.3.1.4.- Gestión de Período

Req 10: El sistema permitirá al usuario ingresar un nuevo periodo para esto ingresará los siguientes datos: Fecha de Inicio, Fecha de Fin, y Descripción, el código será creado automáticamente por el sistema.

Req 11: El sistema permitirá eliminar un período siempre y cuando el mismo haya terminado y no existan datos relacionados.

Req 12: El sistema permitirá al administrador modificar los datos de un período registrado.

4.3.1.5.- Gestión de Materia

Req 13: El sistema permitirá al administrador como al coordinador ingresar una nueva materia para esto ingresará los siguientes datos: Código, Nombre, Número de Créditos. El Número de Horas será calculado por el sistema el mismo que resultará de la multipliación del número de créditos por 18 semanas laborables.

Req 14: El sistema permitirá eliminar una materia siempre y cuando no exista datos relacionados.

Req 15: El sistema permitirá al administrador modificar los datos de una materia registrada si se modifica el código también se deberá modificar en las tablas relacionadas.

4.3.1.6.- Gestión de Área

Req 16: El sistema permitirá al administrador ingresar una nueva área para esto ingresará los siguientes datos: Nombre, Descripción el código será gerado autimáticamente.

Req 17: El sistema permitirá eliminar un área siempre y cuando no exista datos relacionados.

Req 18: El sistema permitirá al administrador modificar los datos de una área registrada si se modifica el código también se deberá modificar en las tablas relacionadas.

4.1.1.4.- Gestión de Personal

Req 19: El sistema permitirá al administrador,coordinador ingresar un nuevo personal (docente) para esto ingresará los siguientes datos: Número de Cédula, Nombres, Apellidos, Dirección, Teléfono del Domicilio, Número de Celular, el sistema comprobará que la cédula sea válida y no se encuentre registrada.

Req 20: El sistema permitirá eliminar un docente siempre y cuando no exista datos relacionados.

Req 21: El sistema permitirá al administrador modificar los datos de un docente registrado, excepto el número de cédula.

4.1.1.5.- Gestión de Asignación de Materia

Req 22: El sistema permitirá al administrador, coordinador ingresar una asignación de materia para esto ingresará los siguientes datos: Paralelo, Día, Hora de Inicio Hora de Fin, el Código el sistema lo autogenera.

Req 23: El sistema permitirá eliminar una asignación de materia siempre y cuando no exista datos relacionados.

Req 24: El sistema permitirá al administrador,coordinador modificar los datos de una asignación registrada, excepto el código.

4.1.1.6.- Gestión de Asistencia

Req 25: El sistema permitirá al docente registrar la hora de entrada a clases para lo cual el sistema presentara un formulario, ingresará el número de cédula y el sistema capturará la hora de ingreso.

Req 26: El sistema permitirá al docente registrar su hora de salida de clases también deberá ingresar los siguientes datos: tema expuesto en clase, alumnos, faltos y observaciones si los hubiere.

4.1.1.7.- Gestión de Reportes

Req 27: El sistema permitirá al administrador generar un reporte del día del registro de entradas y salidas de clase de los Docentes

Req 28: El sistema permitirá al administrador generar un reporte del registro de entradas y salidas de clase de los Docentes en un rango de fecha.

Req 29: El sistema permitirá al administrador realizar un reporte por Docente de sus asistencias a clases en un rango de fecha.

Req 30: El sistema permitirá administrador realizar un reporte por Materia de la asistencia del Docente a clase en un rango de fecha.

4.3.2.- REQUISITOS DE INTERFACES EXTERNOS

4.3.2.1.- Interfaces de Usuario

La interfaz de usuario deberá ser orientada a ventanas, y el manejo del programa se realizará a través de teclado y ratón.

4.3.2.2.- Interfaces Hardware

Se trabajará en plataforma cliente / servidor.

4.3.2.3 .- Interfaces Software

De momento, no habrá ninguna interfaz software con sistemas externos.

4.3.2.4 .- Interfaces de Comunicación

La conexión a la red se establecerá por medio de una conexión directa a la red Fast Ethernet de la Escuela Politécnica del Ejército sede Latacunga. Esto será transparente para la aplicación, la cual, a todos los efectos, considerará que está en la misma red que el servidor.

La conexión a la red se establecerá por medio de una conexión directa a la red Intranet, y mediante conexiones a Internet con el protocolo TCP/IP.

4.3.2.5 .- Requisitos de Rendimiento

El número de puestos a los que se debe dar servicio simultáneamente es de 6. El tiempo de respuesta en las operaciones deberá ser inferior o igual a 10 segundos.

4.3.2.6 .- Requisitos de Desarrollo

El ciclo de vida elegido para desarrollar el producto será el Interactivo Incremental, de manera que se puedan incorporar fácilmente cambios y nuevas funciones.

El proyecto a desarrollarse fue escogido como caso práctico para el estudio del análisis comparativo de J2EE y .NET como tecnologías para el desarrollo Web.

4.3.2.7 .- Requisitos Tecnológicos

La aplicación cliente se ejecutará sobre un PC con una configuración mínima de:

Procesador	Pentium III, 1 GAZ
Memoria	256 MB de RAM
Disco Duro	2 GB de disco libre.
Unidades	CD-ROM o DVD-ROM
Video	Súper VGA (800 x 600) o superior con 256 colores
Mouse	Mouse Microsoft o dispositivo compatible
Tarjeta Ethernet	

La aplicación a generar residirá en un servidor central en la cual se instalará la base de datos la SqlServer para .Net y My Sql para J2EE.

Todos los PC's deberán estar conectados a la red.

El sistema operativo en la que se va interactuar los clientes es Windows y necesariamente deberá estar instalado un Navegador como Internet Explorer, Netscape o Mozilla.

La aplicación en el servidor se ejecutará en el Sistema operativo Windows con Internet Information Server para .NET y Tomcat para J2EE como servidores Web.

4.3.3.- ATRIBUTOS

4.3.3.1.- Seguridad

Cuando un administrador intente conectarse al sistema debe introducir su identificación (Cédula) y Password, y el sistema deberá comprobar que se trata de un usuario autorizado. Si el identificador introducido no corresponde a un usuario autorizado o la clave no coincide con la almacenada, se dará una indicación de error.

4.4.- MODELADO DE DATOS

Definición

Un modelo es un conjunto de herramientas conceptuales para describir datos, sus relaciones, su significado y sus restricciones de consistencia.

Características

Es el proceso de analizar los aspectos de interés para una organización y la relación que tienen unos con otros.

4.4.1.- MODELO CONCEPTUAL

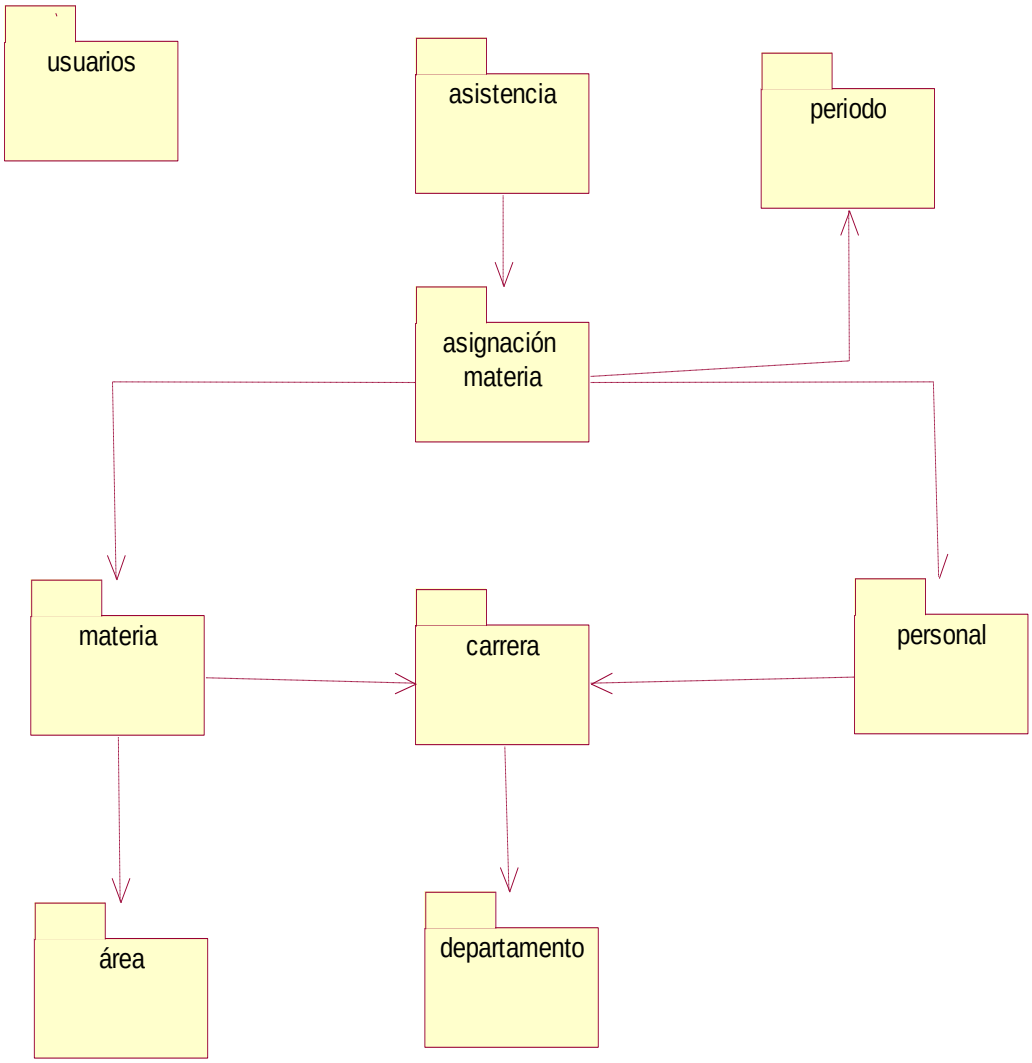

Figura 4.1. Modelo Conceptual

4.4.- DESCRIPCIÓN DE CASOS DE USO

Los diagramas de caso de uso presenta la funcionalidad de un sistema (o una clase) mostrando su interacción con los gentes externos. Esta representación se hace a través de las relaciones entre los actores (agentes externos) y los casos de uso (acciones) dentro del sistema. Los diagramas de casos de uso definen conjuntos de funcionalidades afines que el sistema debe cumplir para satisfacer todos los requerimientos que tiene a su cargo.

Ese conjunto de funcionalidades son representados por los casos de uso. Se pueden visualizar las funciones más importantes que la aplicación puede realizar.

4.5.1.- DIAGRAMA DE CASO DE USO GESTIÓN DE USUARIO

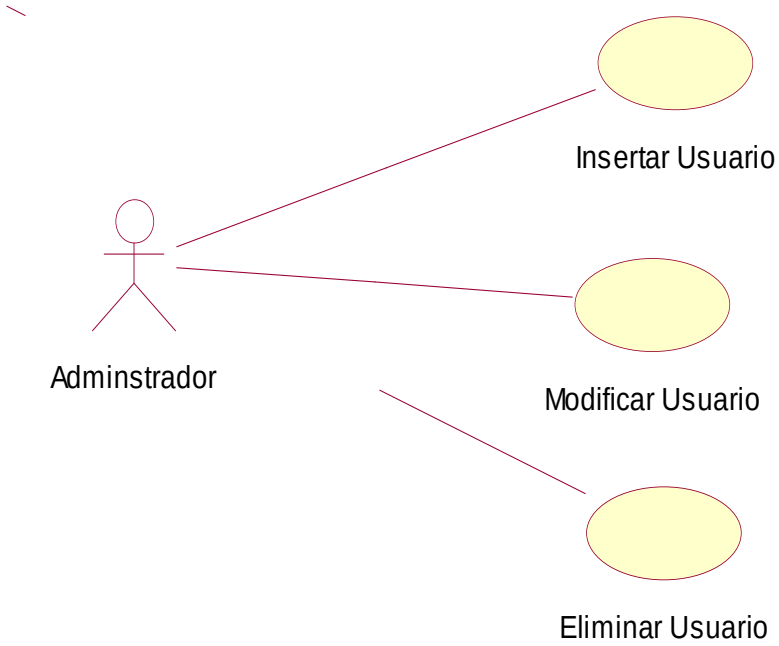

Figura 4.2. Gestión de usuario

4.5.1.1.- Definición de Casos de Uso Expandido

4.5.1.1.1.- Nombre del Caso de Uso: Insertar Usuario

Propósito: Insertar Usuario.
Actor: Administrador (iniciador)
Tipo: Primario real
Referencia: Req(07).

Visión General: El administrador selecciona la opción nueva del menú usuario, el sistema presenta un formulario electrónico del usuario y podrá ingresar los datos requeridos (Número de Cédula, Nombres, Apellidos, Password, Nivel) el sistema valida la información y crea una instancia usuario.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción nueva del menú usuario (op).	2 Presentar el formulario electrónico de nuevo usuario.
3 Ingresa datos (Número de cédula, Nombres, Apellidos, Password, Nivel de usuario.	4 Valida datos y crea instancia usuario

Casos Alternativos

2* No existe Formulario Electrónico termina caso de uso
4* Ya existe el usuario, no se crea la instancia. Termina el caso de uso.
4* Datos inválidos, no se creo la instancia. Termina el caso de uso.

Diagrama de Secuencia

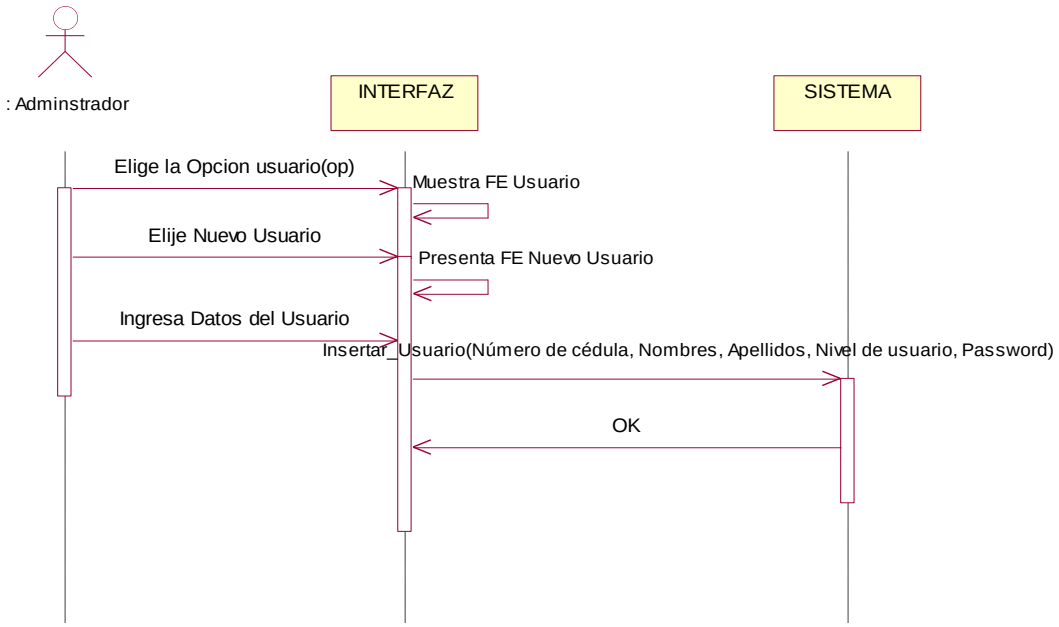

Figura 4.3. Diagrama de Secuencia Insertar Usuario

Diagrama de Colaboración

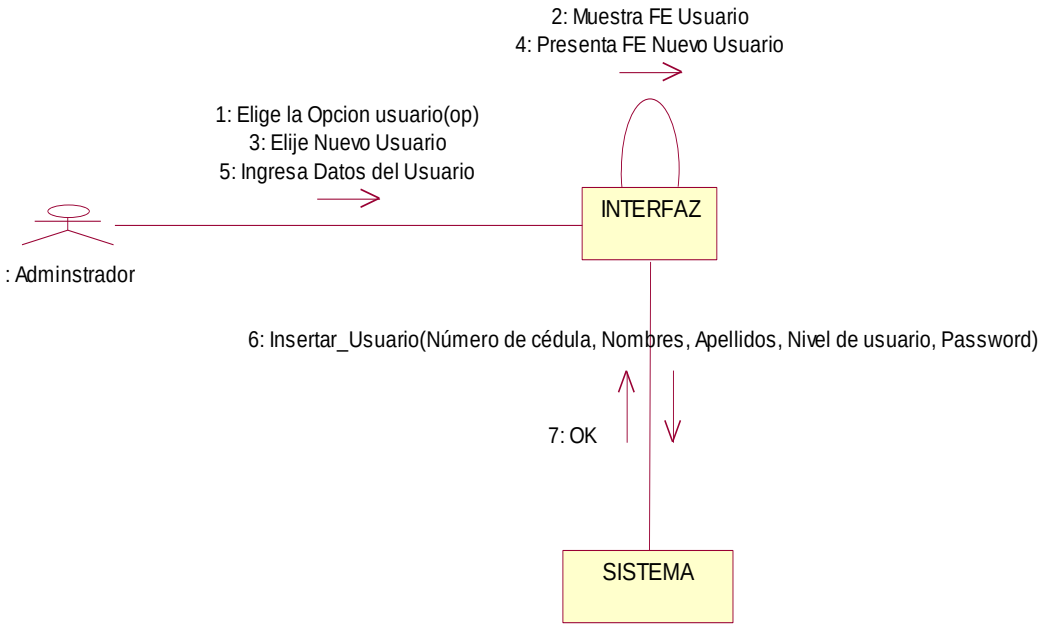

Figura 4.4. Diagrama de Colaboración Insertar Usuario

4.5.1.1.2.- Nombre del Caso de Uso: Modificar Usuario

Propósito: Modificar Usuario.

Actor: Administrador (iniciador)

Tipo: Primario Real

Referencia: Req(09)

Visión General: El usuario selecciona la opción modificar del menú usuario, el sistema presenta un listado con los usuarios registrados, se selecciona el usuario y se podrá modificar los datos del mismo.

Curso Típico de Eventos

ACTOR	**SISTEMA**
1 Seleccionar la opción modificar usuario del menú usuario (op).	2 Presenta un formulario electrónico con el listado de usuarios de búsqueda del usuario
3 Selecciona el usuario a modificar(usu_cédula)	4 Presenta FE de modificar con datos del usuario
5 Ingresa nuevos datos.	6 Valida datos y modifica instancia

Casos Alternativos

2* No existe FE de búsqueda, termina caso de uso

4* No existen datos de búsqueda, termina el caso de uso.

6* Datos incorrectos, no se modifica la instancia termina caso de uso.

Diagrama de Secuencia

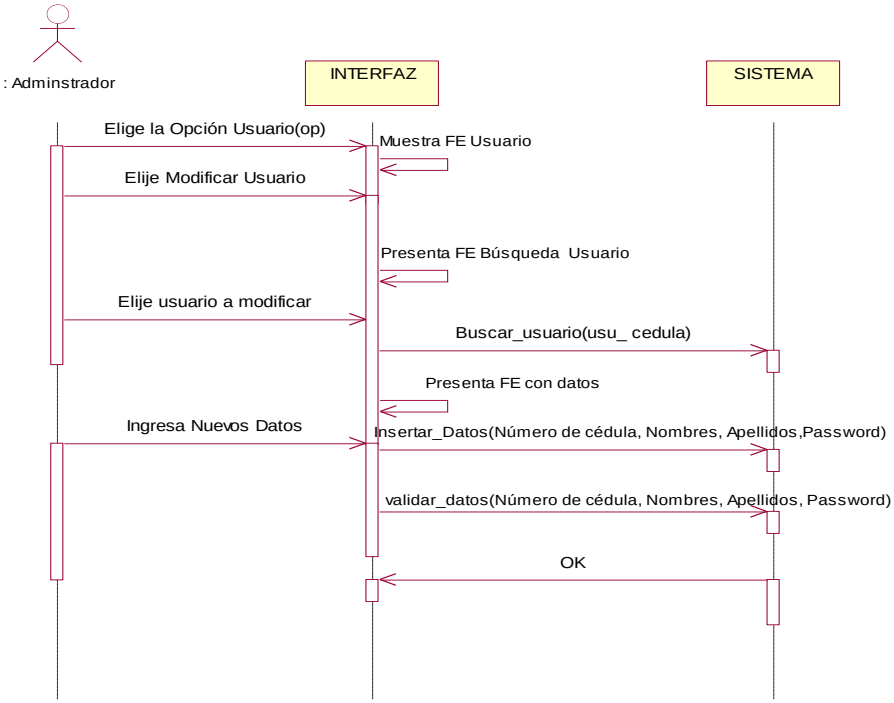

Figura 4.5. Diagrama de Secuencia Modificar Usuario

Diagrama de Colaboración

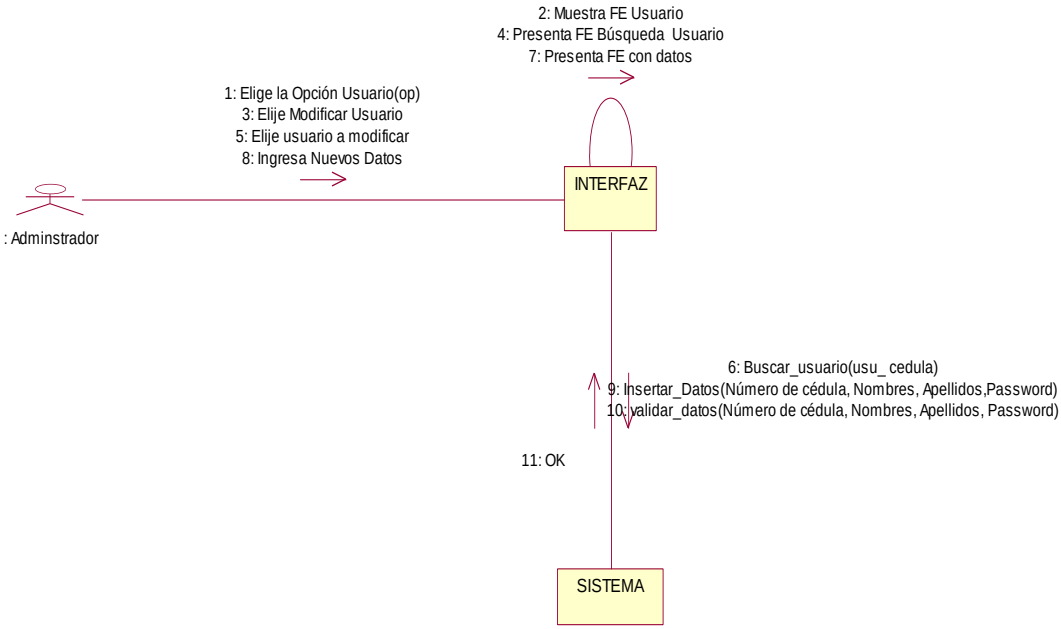

Figura 4.6. Diagrama de Colaboración Modificar Usuario

4.5.1.1.3.- Nombre del Caso de Uso: Eliminar Usuario

Propósito: Eliminar Usuario.

Actor: Administrador (iniciador).

Tipo: Primario Real.

Referencia: Req(8).

Visión General: El usuario selecciona la opción eliminar del menú usuario, el sistema presenta formulario electrónico del usuario y podrá eliminar los datos del usuario.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción eliminar usuario del menú usuario (op)	2 Presenta formulario electrónico búsqueda de usuario
3 Selecciona el usuario a eliminar(usu_cédula)	4 Presenta FE de eliminación de usuario con resultados de búsqueda.
5 Selecciona la opción eliminar(usu_cédula)	6 Presenta mensaje de confirmación de eliminación
7 Aceptar eliminación(OK)	Eliminar instancia

Casos Alternativos

2* No existe FE, termina caso de uso.

4* No existen resultados de búsqueda, termina el caso de uso.

6* No existe mensaje de confirmación, termina caso de uso.

8* No existe confirmación, no se elimina la instancia termina caso de uso.

Diagrama de Secuencia

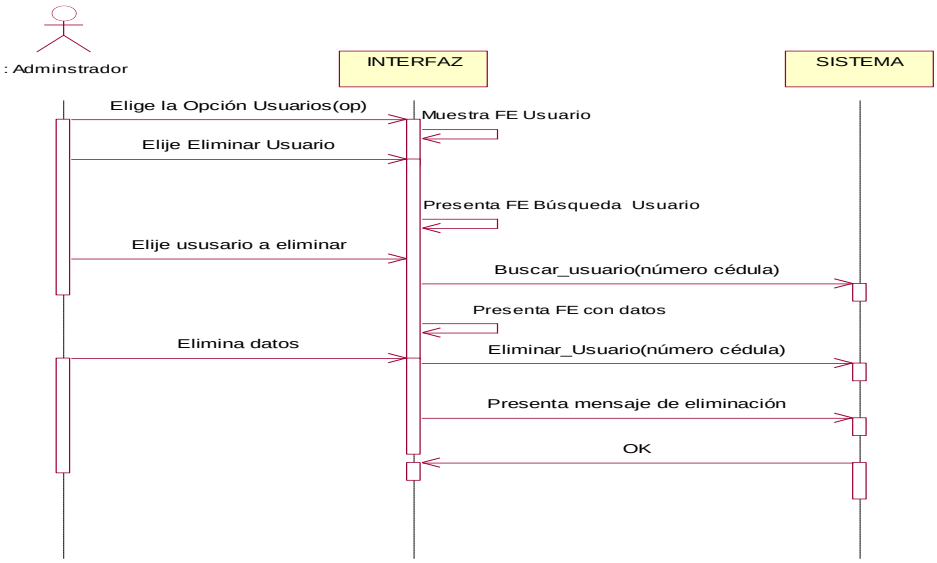

Figura 4.7. Diagrama de Secuencia Eliminar Usuario

Diagrama de Colaboración

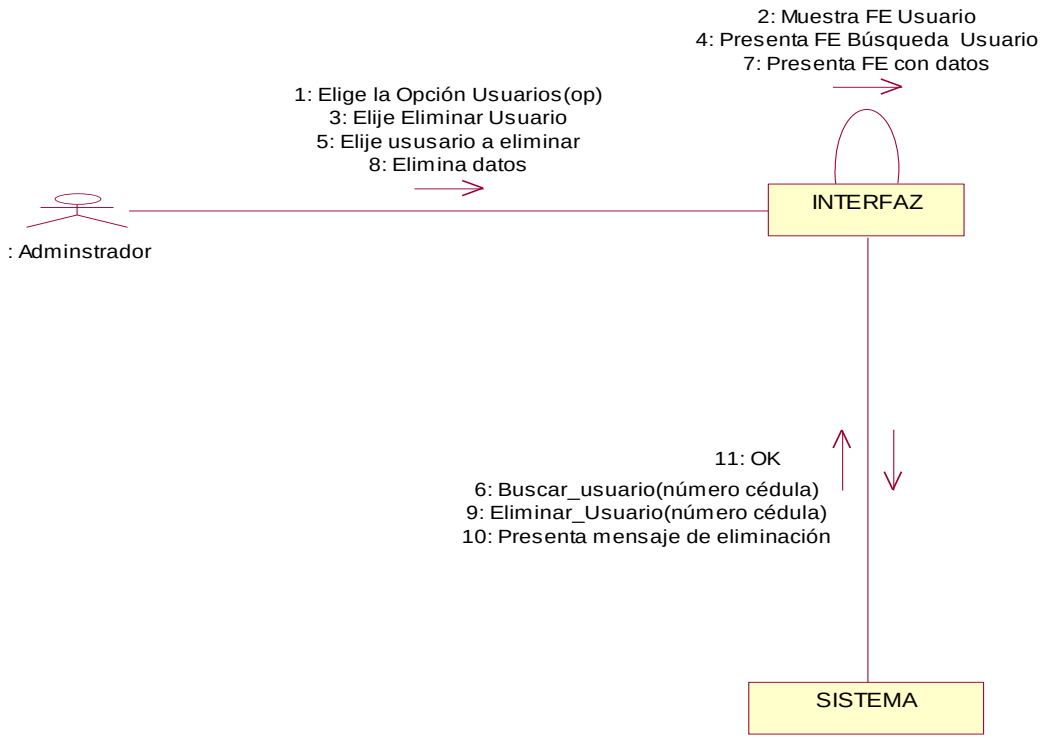

Figura 4.8. Diagrama de Colaboración Eliminar Usuario

4.5.2.- DIAGRAMA DE CASO DE USO GESTIÓN DE CARRERA

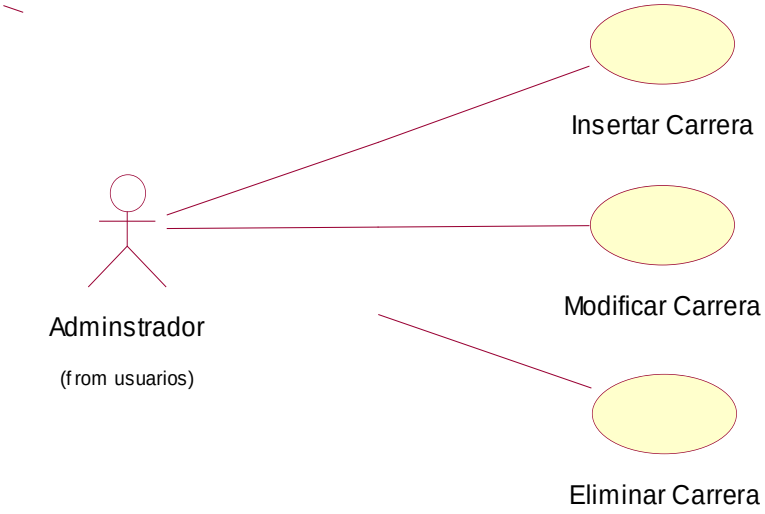

Figura 4.9. Gestión Carrera

4.5.2.1.- Definición de Casos de Uso Expandido

4.5.2.1.1.- Nombre del Caso de Uso: Insertar Carrera

Propósito: Insertar Carrera.

Actor: Administrador, Coordinador (iniciador)

Tipo: Primario Real.

Referencia: Req(04)

Visión General: El administrado, coordinador selecciona la opción nuevo del menú carrera, el sistema presenta formulario electrónico de la carrera y podrá ingresar los datos requeridos (Código, Nombre, Descripción, Departamento) el sistema valida la información y crea una instancia llamada carrera.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Seleccionar la opción nuevo del menú carrera(op)	2 Presenta formulario electrónico de nueva carrera.
3 Ingresa datos (Código, Nombre, Descripción)	4 Valida datos, crear instancia carrera.

Casos Alternativos

2* No existe FE termina caso de uso

4* Ya existe la carrera, no se creo la instancia termina el caso de uso.

4* Datos inválidos, no se crea la instancia termina el caso de uso.

Diagrama de Secuencia

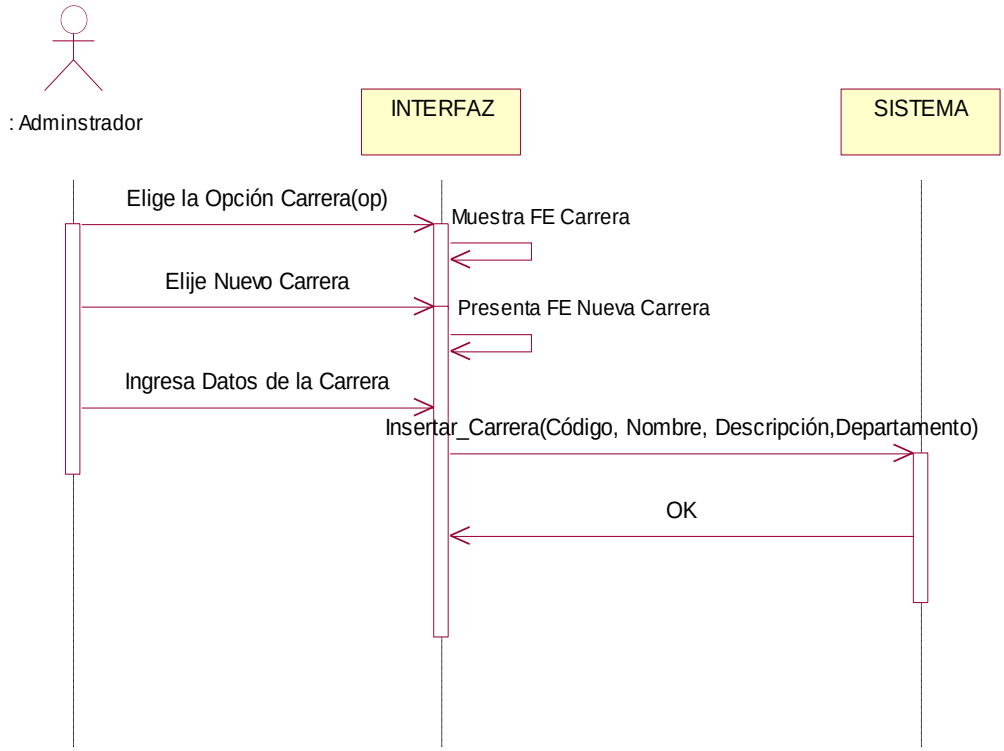

Figura 4.10. Diagrama de Secuencia Insertar Carrera

Diagrama de Colaboración

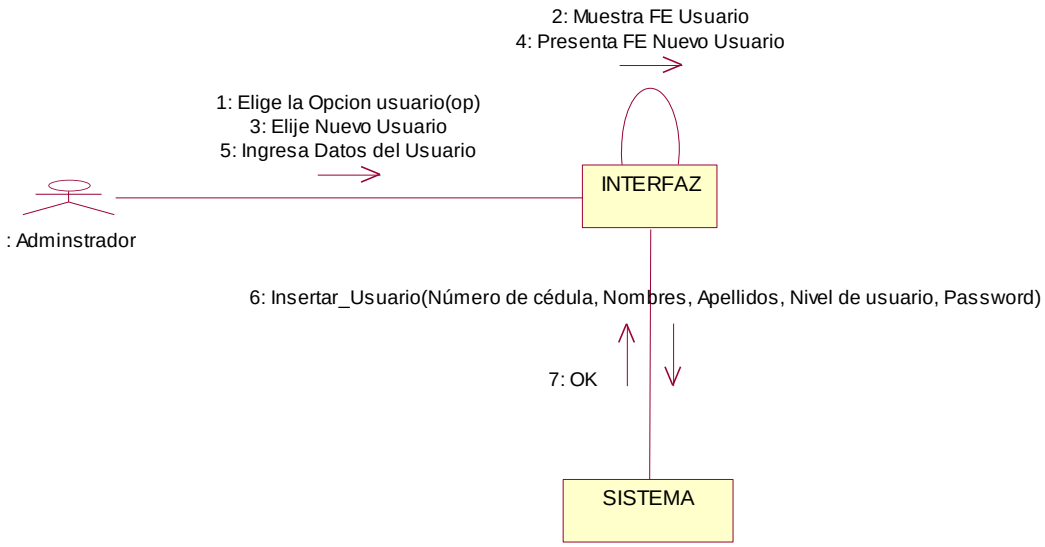

Figura 4.12. Diagrama de Colaboración Insertar Carrera

4.5.2.1.2.- Nombre del Caso de Uso: Modificar Carrera

Propósito: Modificar Carrera

Actor: Administrador,Coordinador (iniciador)

Tipo: Primario real

Referencia: Req(06)

Visión General: El administrador, coordinador selecciona la opción modificar del menú carrera, el sistema presenta formulario electrónico de la carrera y podrá modificar los datos de la misma.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción modificar carrera del menú carrera(op)	2 Presentar formulario electrónico de carrera
3 Selecciona la carrera a modificar(código carrera)	4 Presenta FE de modificar carrera con el resultado de la búsqueda
5 Ingresa nuevos datos.	6Valida datos y modifica instancia.

Casos Alternativos

2* No existe FE de búsqueda, termina caso de uso

4* No existen datos de búsqueda, termina el caso de uso.

6* Datos incorrectos, no se modifica la instancia y termina el caso de uso.

Diagrama de Secuencia

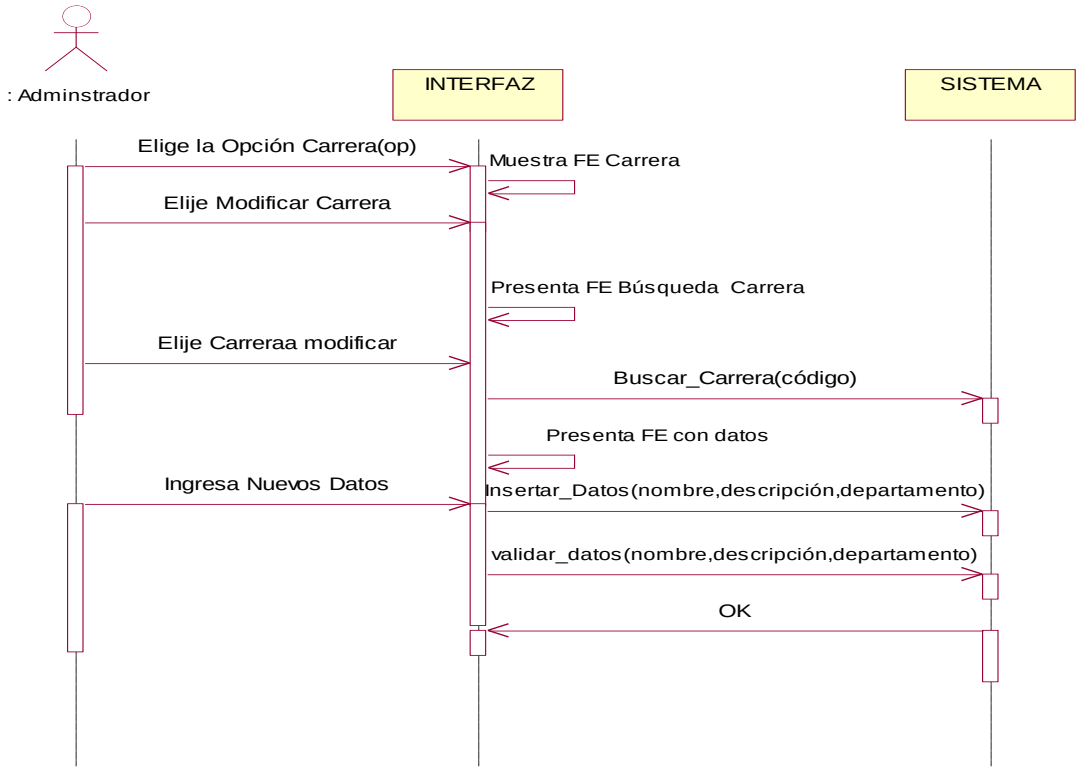

Figura 4.13. Diagrama de Secuencia Modificar Carrera

Diagrama de Colaboración

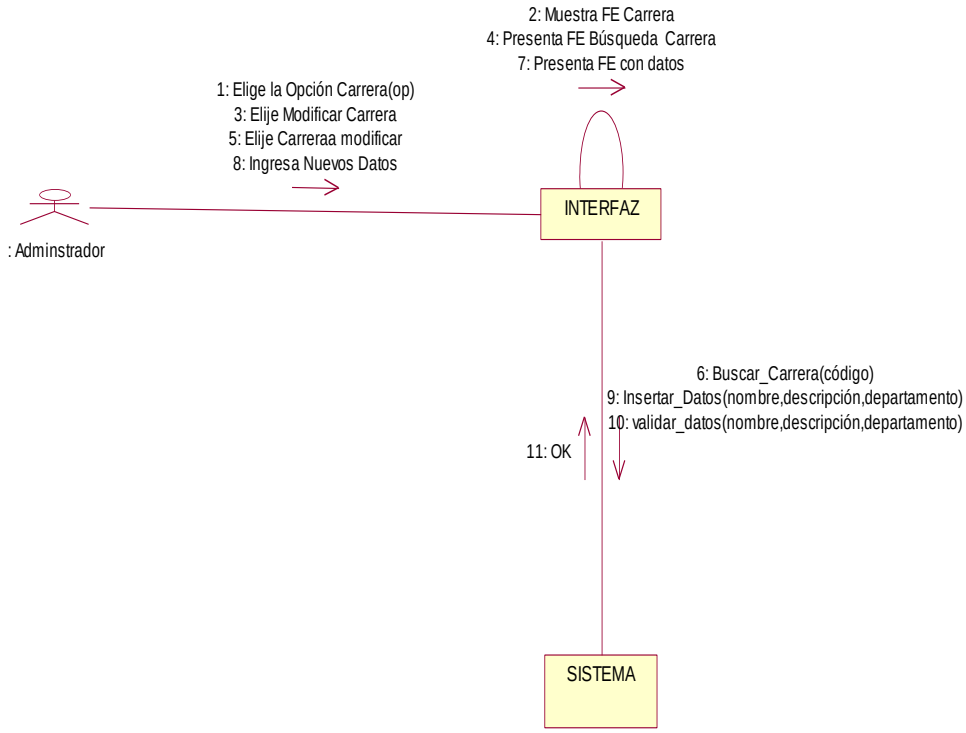

Figura 4.14. Diagrama de Colaboración Modificar Carrera

4.5.2.1.3.- Nombre del Caso de Uso: Eliminar Carrera

Propósito: Eliminar Carrera

Actor: Administrador (iniciador)

Tipo: Primario real

Referencia: Req(05).

Visión General: El administrador selecciona la opción eliminar carrera del menú carrera el sistema presenta formulario electrónico de la carrera y podrá eliminar los datos de la carrera.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción eliminar carrera del menú carrera (op)	2 Presenta formulario electrónico búsqueda de carrera
3 Selecciona la carrera a eliminar(código)	4 Presenta FE de eliminación de la carrera con resultados de la búsqueda.
5 Selecciona la opción eliminar (código)	6 Elimina instancia

Casos Alternativos

2* No existe FE, termina caso de uso.

4* No existen resultados de la búsqueda, termina el caso de uso.

6* No existe independencia de datos, termina caso de uso.

Diagrama de Secuencia

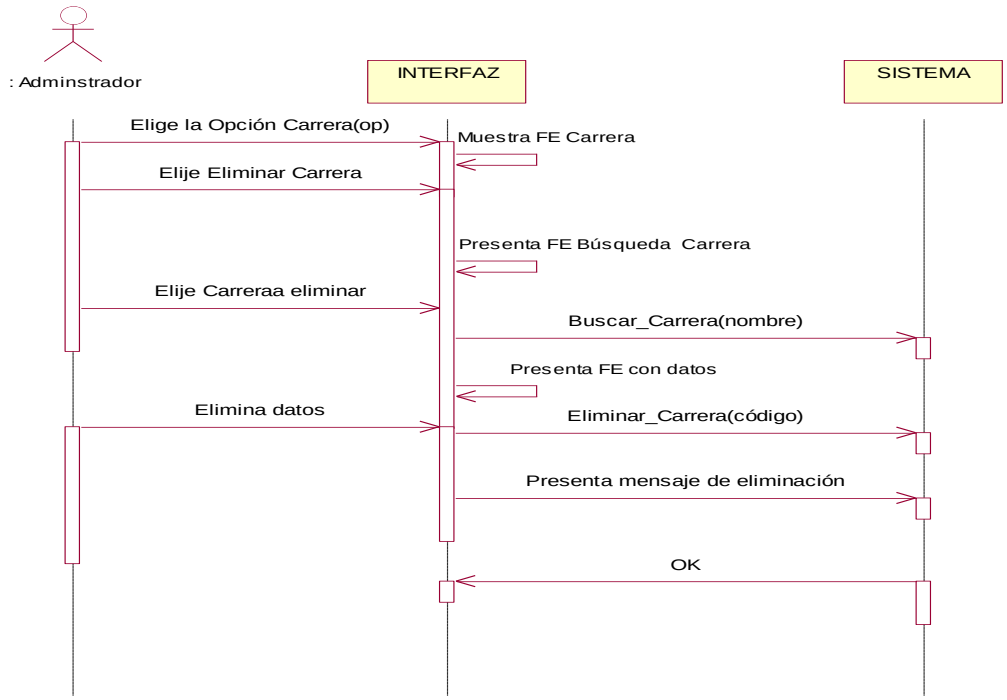

Figura 4.15. Diagrama de Secuencia Eliminar Carrera

Diagrama de Colaboración

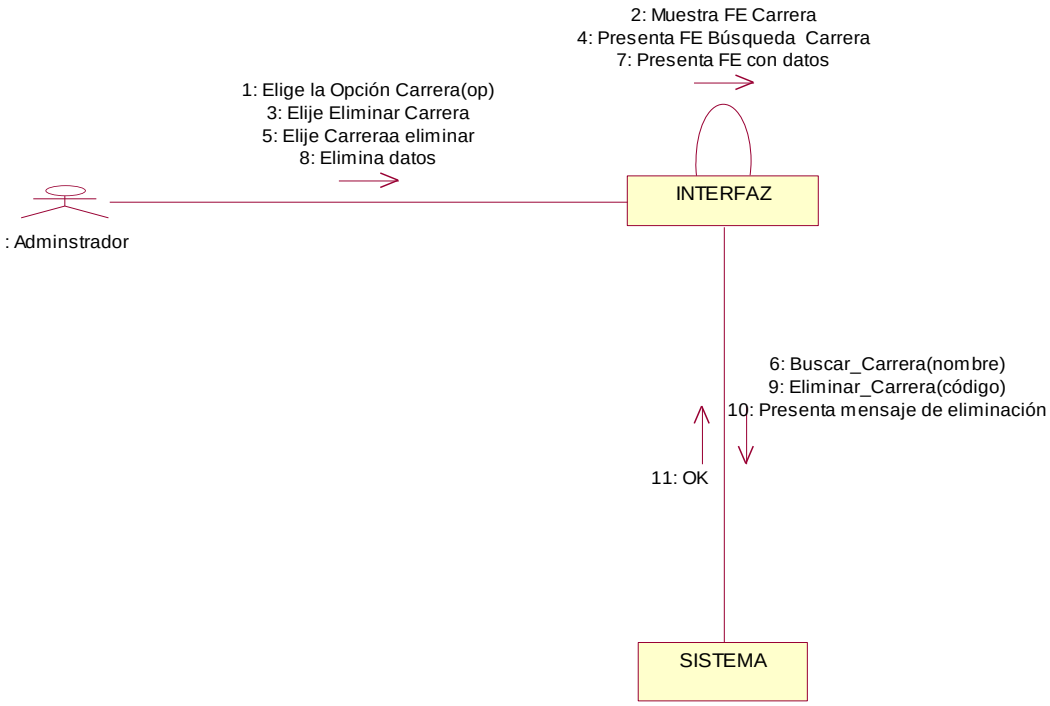

Figura 4.16. Diagrama de Colaboración Eliminar Carrera

4.5.3.- DIAGRAMA DE CASO DE USO GESTIÓN DE MATERIA

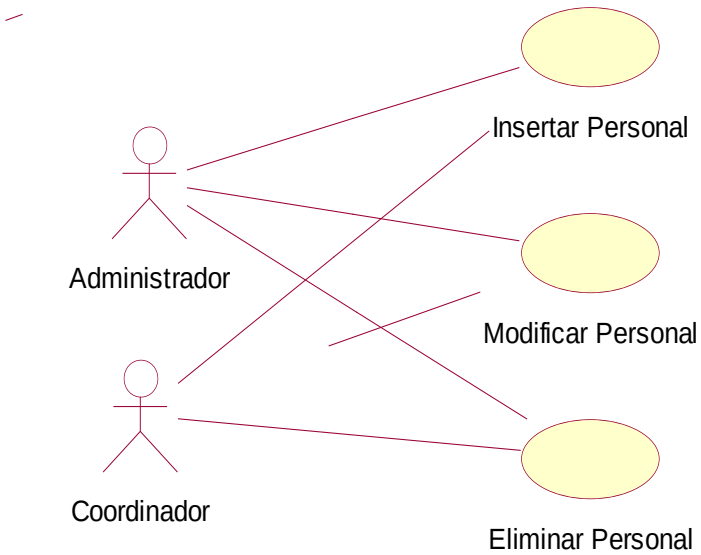

Figura 4.17. Gestión de Materia

4.5.3.1.- Definición de Casos de Uso Expandido

4.5.3.1.1.- Nombre del Caso de Uso: Insertar Materia

Propósito: Insertar Materia.

Actor: Administrador, Coordinador (iniciador)

Tipo: Primario real

Referencia: Req(13).

Visión General: El administrador, coordinador selecciona la opción nuevo del menú materia, el sistema presenta formulario electrónico de la materia y podrá ingresar los datos requeridos (Código, Carrera, Área, Nombre, Número de Créditos) el sistema valida la información y crea una instancia materia.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción nuevo del menú materia (op)	2 Presentar formulario electrónico de nueva materia.
3 Ingresa datos (Código, Carrera, Área, Nombre, Número de Créditos)	4 Valida datos, crea instancia materia

Casos Alternativos

2* No existe FE termina caso de uso

4* Ya existe la materia, no se crea la instancia. Termina el caso de uso.

4* Datos inválidos, no se crea la instancia. Termina el caso de uso.

Diagramas de Secuencia

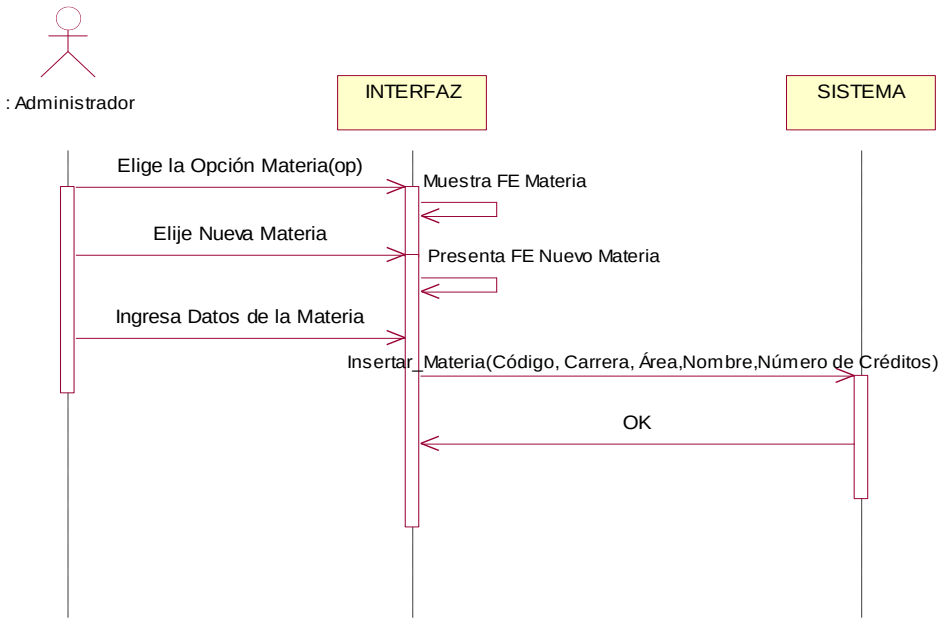

Figura 4.18. Diagrama de Secuencia Insertar Materia

Diagramas de Colaboración

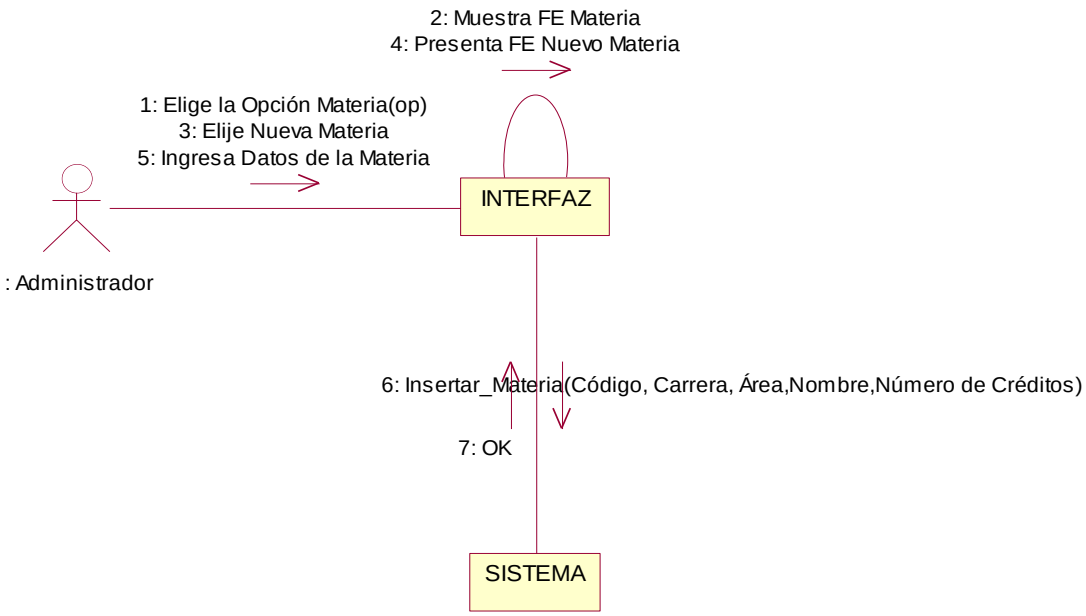

Figura 4.19. Diagrama de Colaboración Insertar Materia

4.5.3.1.2.- Nombre del Caso de Uso: Modificar Materia

Propósito: Modificar Materia.

Actor: Administrador, Coordinador (iniciador)

Tipo: Primario real

Referencia: Req(15)

Visión General: El administrador, coordinador selecciona la opción modificar del menú materia, el sistema presenta formulario electrónico de la materia y podrá modificar los datos de la materia.

Curso Típico de Eventos

ACTOR	**SISTEMA**
1 Seleccionar la opción modificar materia del menú materia (op)	2 Presenta formulario electrónico de búsqueda de materia
3 Selecciona la materia a modificar(nombre)	4 Presenta FE de modificar materia el resultado de la búsqueda
5 Ingresa nuevos datos.	6 Valida datos y modifica instancia.

Casos Alternativos

2* No existe FE de búsqueda, termina caso de uso

4* No existen datos de búsqueda, termina el caso de uso.

6* Datos incorrectos, no se modifica la instancia termina caso de uso.

Diagrama de Secuencia

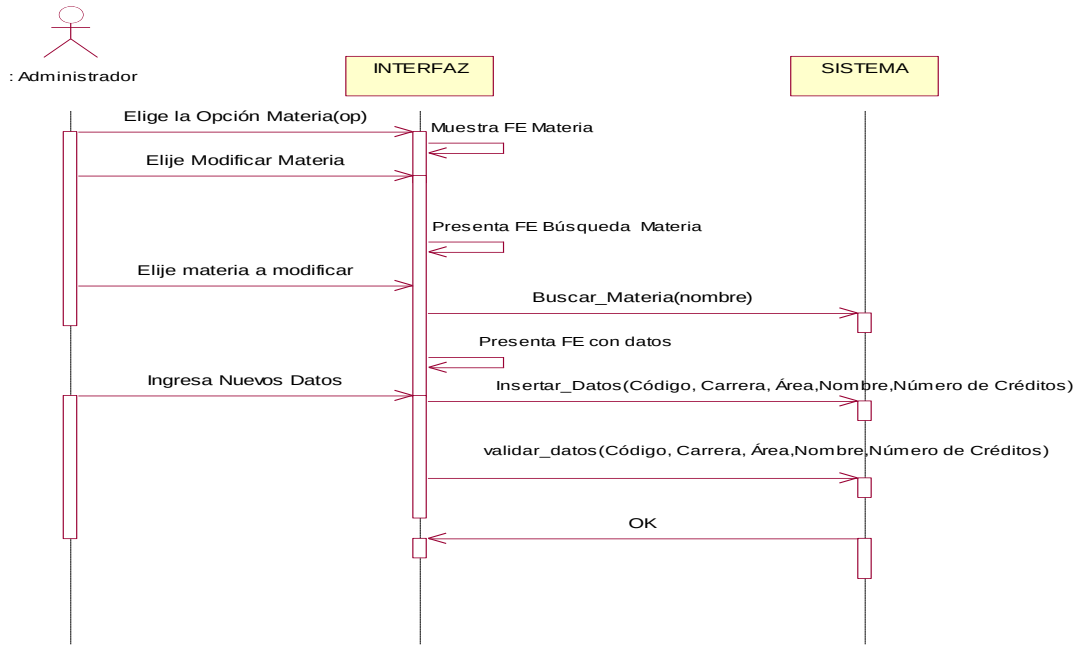

Figura 4.20. Diagrama de Secuencia Modificar Materia

Diagrama de Colaboración

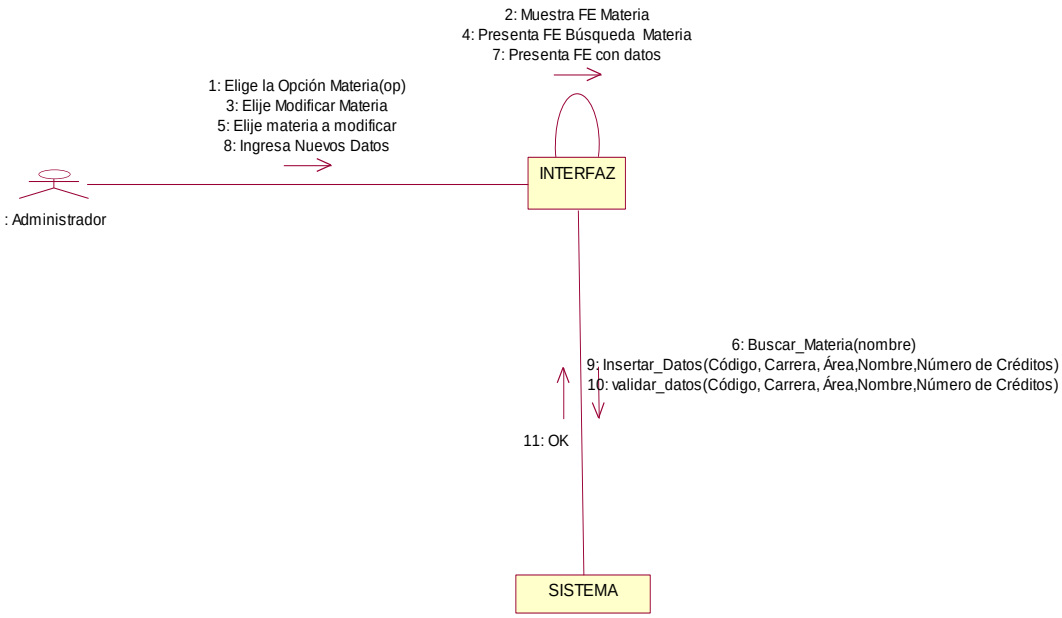

Figura 4.21. Diagrama de Colaboración Modificar Materia

4.5.3.1.3.- Nombre del Caso de Uso: Eliminar Materia

Propósito: Eliminar Materia.

Actor: Administrador (iniciador)

Tipo: Primario Real.

Referencia: Req(14)

Visión General: El administrador selecciona la opción eliminar del menú materia, el sistema presenta formulario electrónico de la materia y podrá eliminar los datos de la materia.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción eliminar materia del menú materia (op)	2 Presenta formulario electrónico búsqueda de materia
3 Selecciona la materia a eliminar(código)	4 Presenta FE de eliminación de materia con resultados de búsqueda.
5 Selecciona la opción eliminar(código)	6 Eliminar instancia.

Casos Alternativos

2* No existe FE, termina caso de uso.

4* No existen resultados de búsqueda, termina el caso de uso.

6* No existe independencia de datos, termina caso de uso.

Diagrama de Secuencia

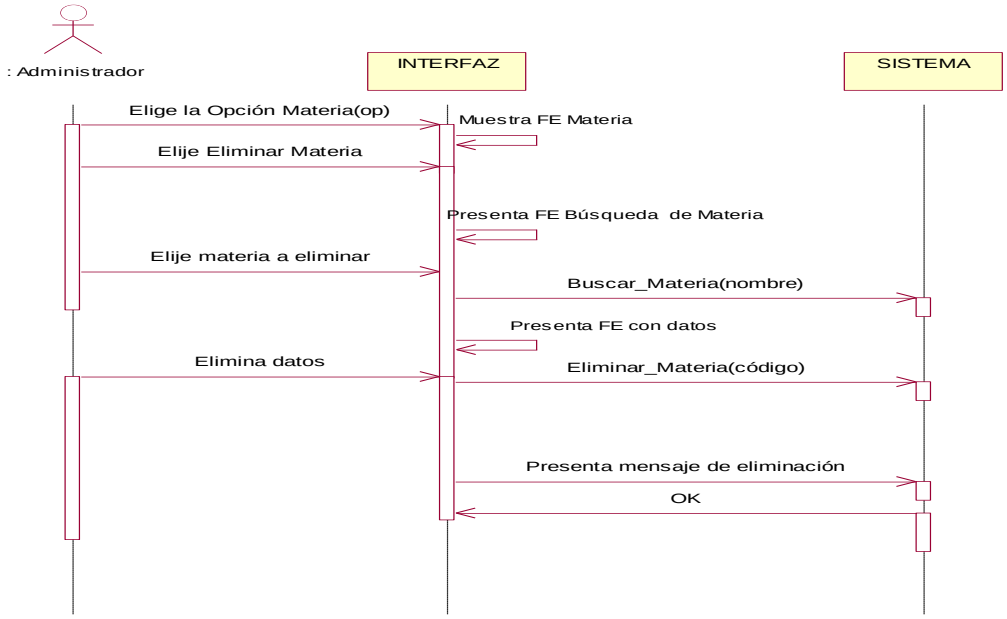

Figura 4.22. Diagrama de Secuencia Eliminar Materia

Diagrama de Colaboración

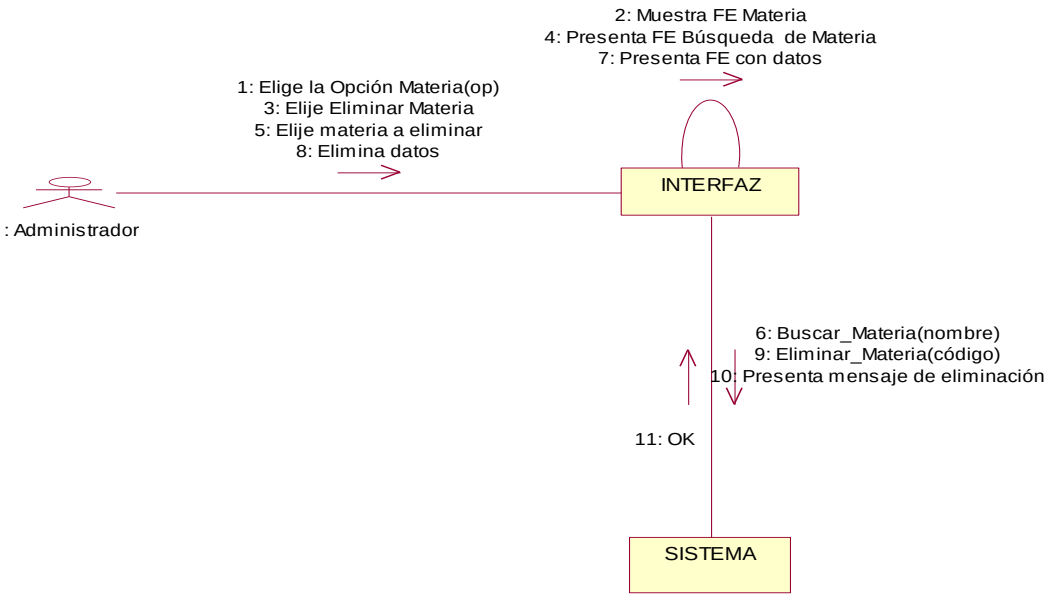

Figura 4.23. Diagrama de Colaboración Eliminar Materia

4.5.4.- DIAGRAMA DE CASO DE USO GESTIÓN DE PERSONAL

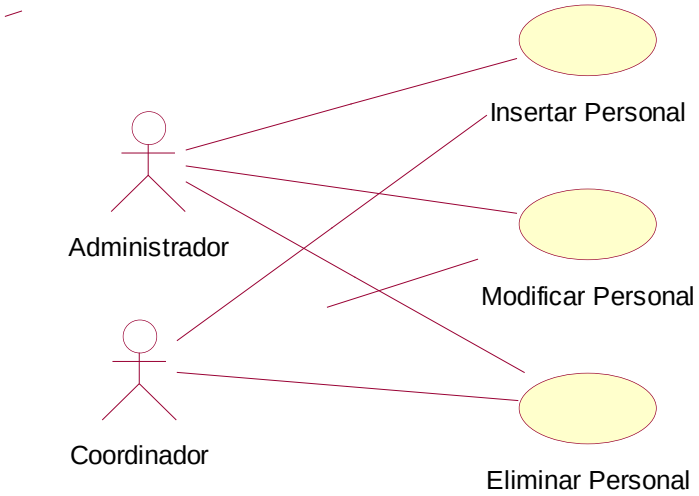

Figura 4.24. Gestión de Personal

4.5.4.1.- Definición de Casos de Uso Expandido

4.5.4.1.1.- Nombre del Caso de Uso: Insertar Personal

Propósito: Insertar Personal.

Actor: Administrador, Coordinador (iniciador)

Tipo: Primario Real

Referencia: Req(21).

Visión General: El administrador, coordinador selecciona la opción nueva del menú docente, el sistema presenta formulario electrónico del personal(docente) y podrá ingresar los datos requeridos (Número de Cédula, Carrera, Nombres, Apellidos, Dirección, Teléfono del Domicilio, Número de Celular) el sistema valida la información y crea una instancia personal.

Curso Típico de Eventos

ACTOR	SISTEMA

1 Selecciona la opción nuevo del menú docente (op)	2 Presentar formulario electrónico de nuevo docente.
3 Ingresa datos (Número de Cédula, Carrera, Nombres, Apellidos, Dirección, Teléfono del Domicilio, Número de Celular)	4 Valida datos, crea instancia personal

Casos Alternativos

2* No existe FE termina caso de uso

4* Ya existe el personal(docente), no se crea la instancia. Termina el caso de uso.

4* Datos inválidos, no se crea la instancia, termina el caso de uso.

Diagrama de Secuencia

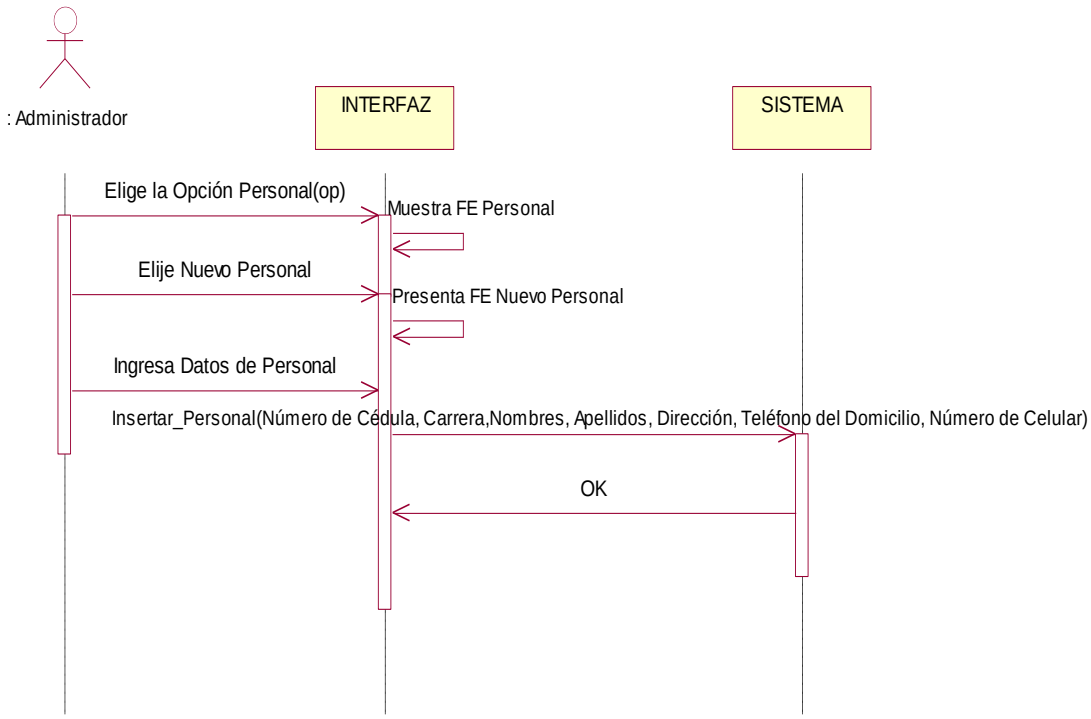

Figura 4.25. Diagrama de Secuencia Insertar Personal

Diagrama de Colaboración

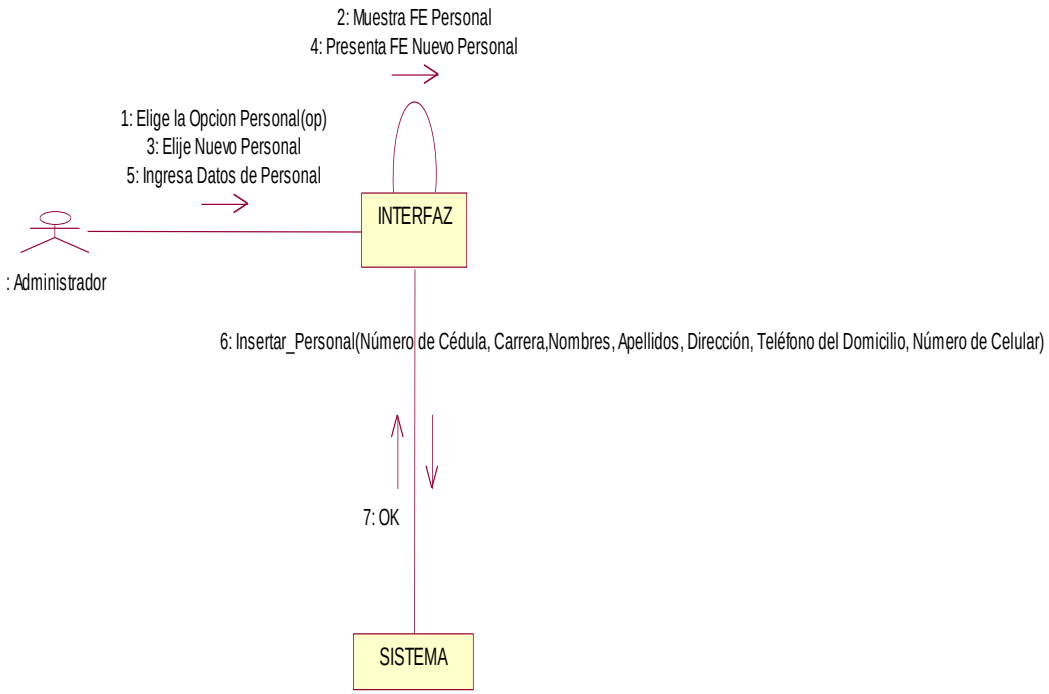

Figura 4.21. Diagrama de Colaboración Insertar Personal

4.5.4.1.2.- Nombre del Caso de Uso: Modificar Personal

Propósito: Modificar Personal.
Actor: Administrador, Coordinador (iniciador)
Tipo: Primario real

Visión General: El usuario selecciona la opción modificar del menú docente, el sistema presenta formulario electrónico de los docentes y podrá modificar los datos del personal.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Seleccionar la opción modificar docente del menú docente (op).	2 Presenta formulario electrónico de búsqueda de docente(personal)
3 Selecciona el personal a modificar(cédula)	4 Presenta FE de modificar personal con el resultado de la búsqueda
5 Ingresa nuevos datos.	6 Valida datos y modifica instancia.

Casos Alternativos

2* No existe FE de búsqueda, termina caso de uso.

4* No existen datos de búsqueda, termina el caso de uso.

6* Datos incorrectos, no se modifica la instancia, termina caso de uso.

Diagrama de Secuencia

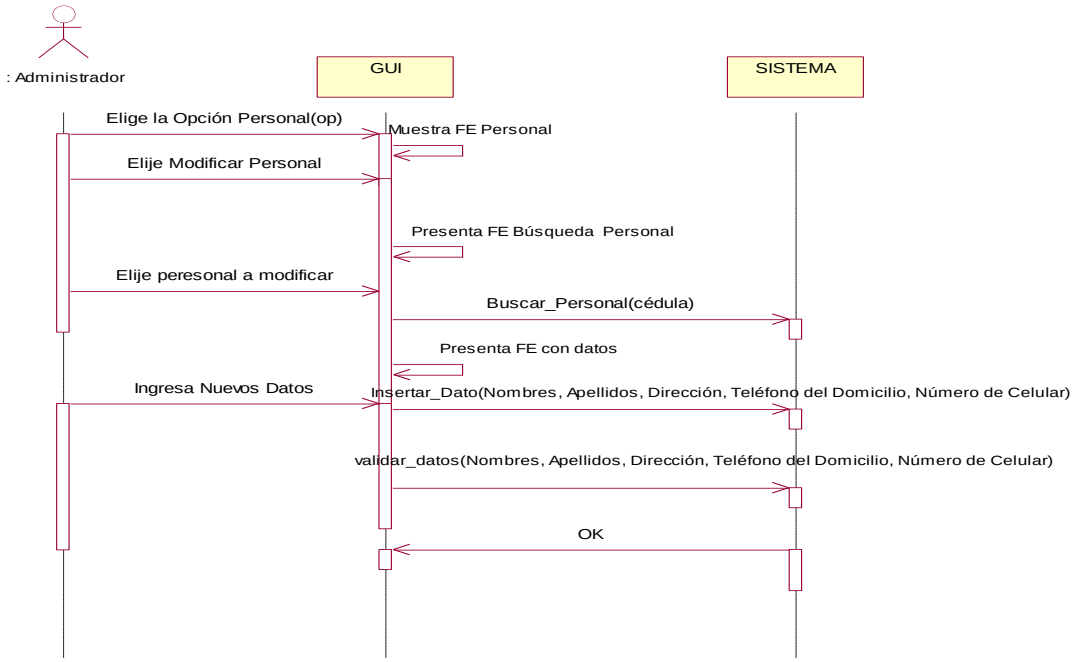

Figura 4.26. Diagrama de Secuencia Modificar Personal

Diagrama de Colaboración

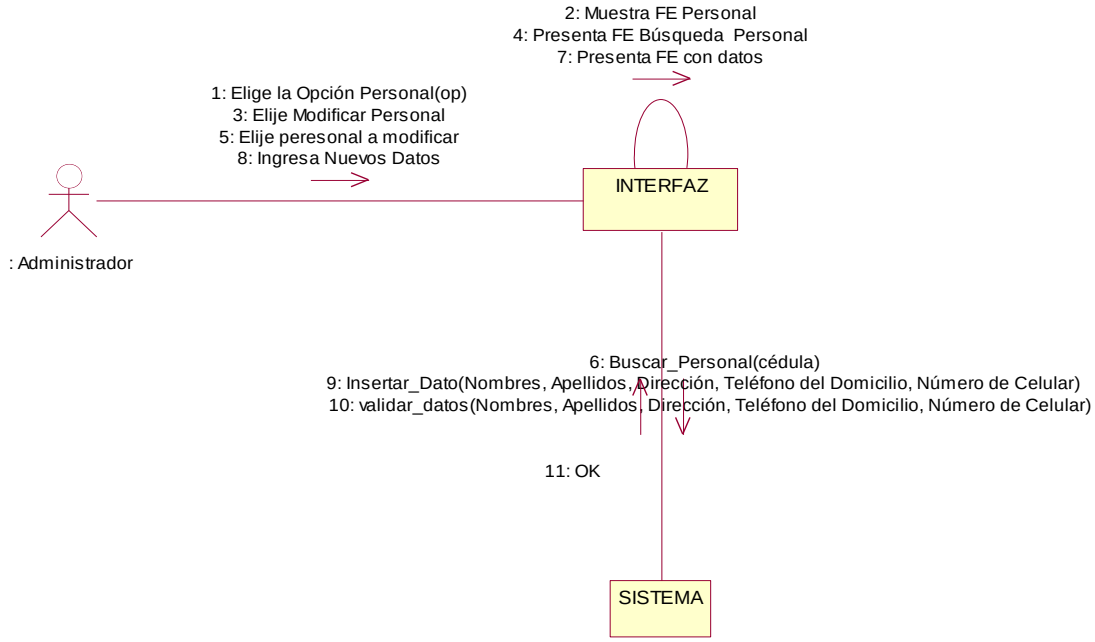

Figura 4.27. Diagrama de Secuencia Modificar Personal

4.1.1.7.1 Nombre del Caso de Uso: Eliminar Personal

Propósito: Eliminar Personal.

Actor: Administrador, Coordinador (iniciador)

Tipo: Primario real

Visión General: El usuario selecciona la opción eliminar del menú docente (personal), el sistema presenta formulario electrónico del personal y podrá eliminar los datos del personal.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción eliminar del menú docente (op)	2 Presenta formulario electrónico búsqueda de docente.
3 Selecciona el personal a eliminar(per_código)	4 Presenta FE de eliminación de personal con resultados de búsqueda
5 Selecciona la opción eliminar(per_código)	6 Eliminar instancia

Casos Alternativos

2* No existe FE, termina caso de uso.

4* No existen resultados de búsqueda, termina el caso de uso.

6* No existe independencia de datos, termina caso de uso.

Diagrama de Secuencia

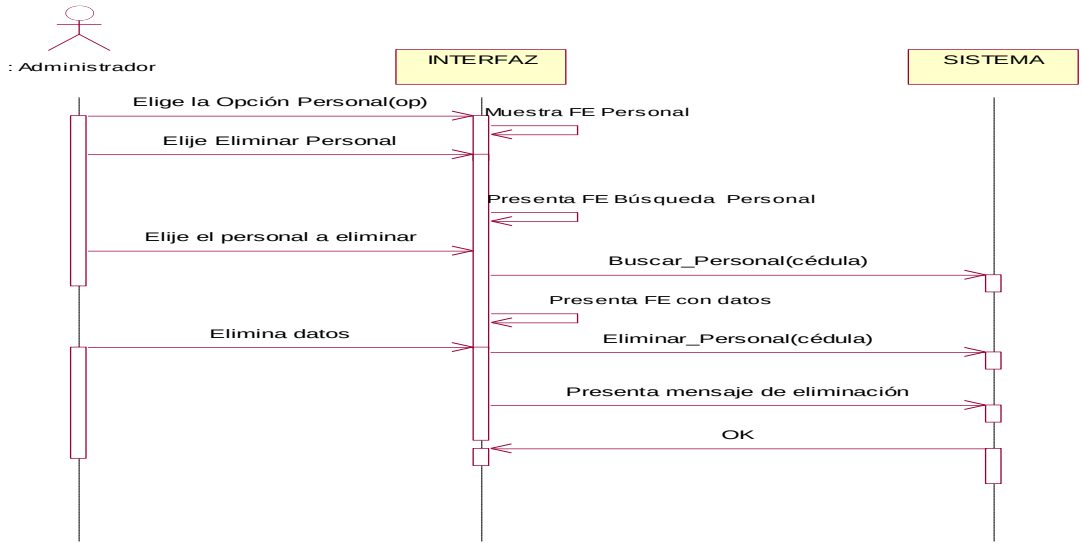

Figura 4.28. Diagrama de Secuencia Eliminar Personal

Diagrama de Colaboración

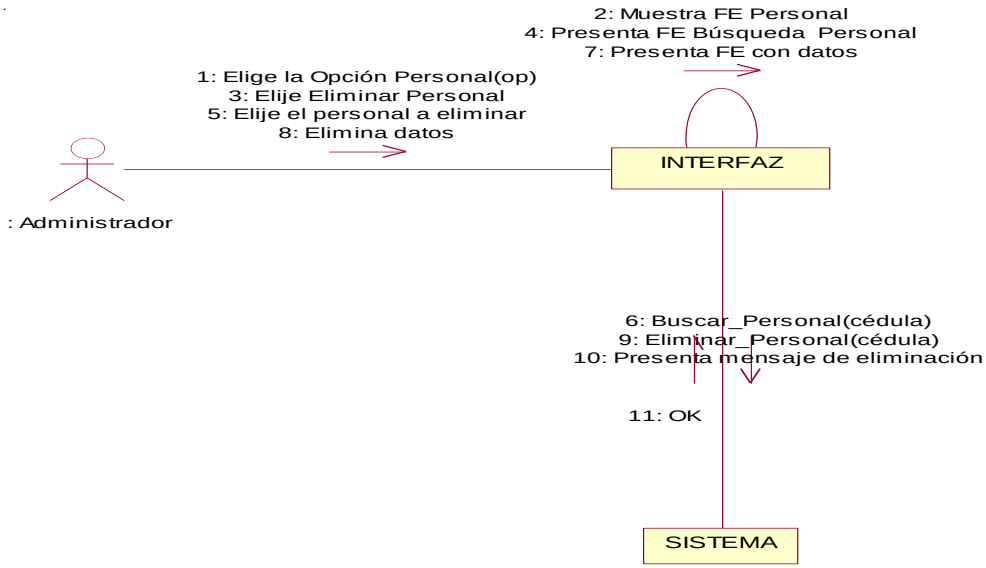

Figura 4.29. Diagrama de Colaboración Eliminar Persona

4.5.5.- DIAGRAMA DE CASO DE USO GESTIÓN DE PERÍODO

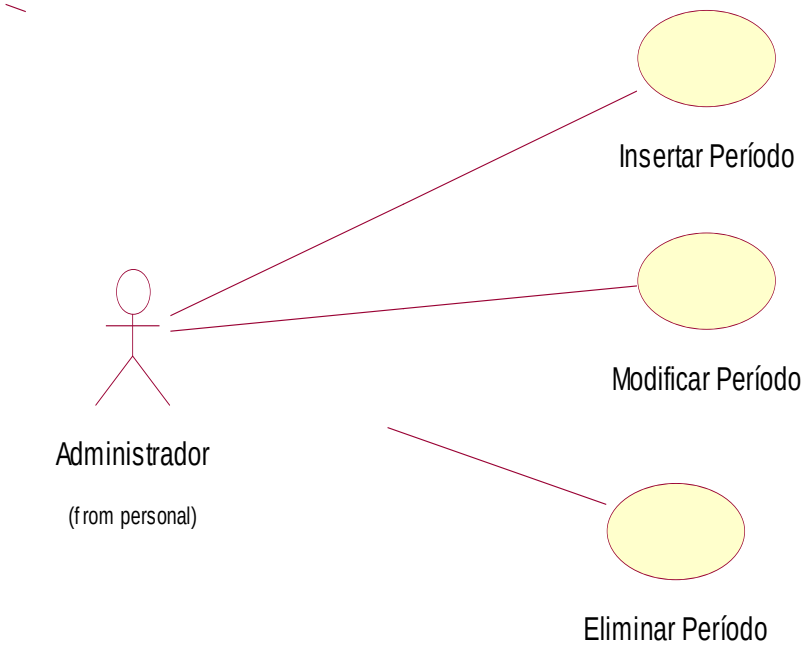

Figura 4.30. Gestión de Período

4.5.5.1.- Definición de Casos de Uso Expandido

4.5.5.1.1.- Nombre del Caso de Uso: Insertar Período

Propósito: Insertar Período.
Actor: Administrador (iniciador)
Tipo: Primario Real.
Referencia: Req(10)

Visión General: El administrador selecciona la opción nuevo del menú período, el sistema presenta formulario electrónico del período y podrá ingresar los datos requeridos (Fecha de Inicio, Fecha de Fin, Descripción) , el código se genera automáticamente, el sistema valida la información y crea una instancia período.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción nueva del menú período (op).	2 Presentar formulario electrónico de nuevo período.
3 Ingresa datos (Fecha de Inicio, Fecha de Fin, Descripción)	4 Valida datos, crea instancia período

Casos Alternativos

2* No existe FE termina caso de uso

4* Ya existe el período, no se creo la instancia. Termina el caso de uso.

4* Datos inválidos, no se creo la instancia. Termina el caso de uso.

Diagramas de Secuencia

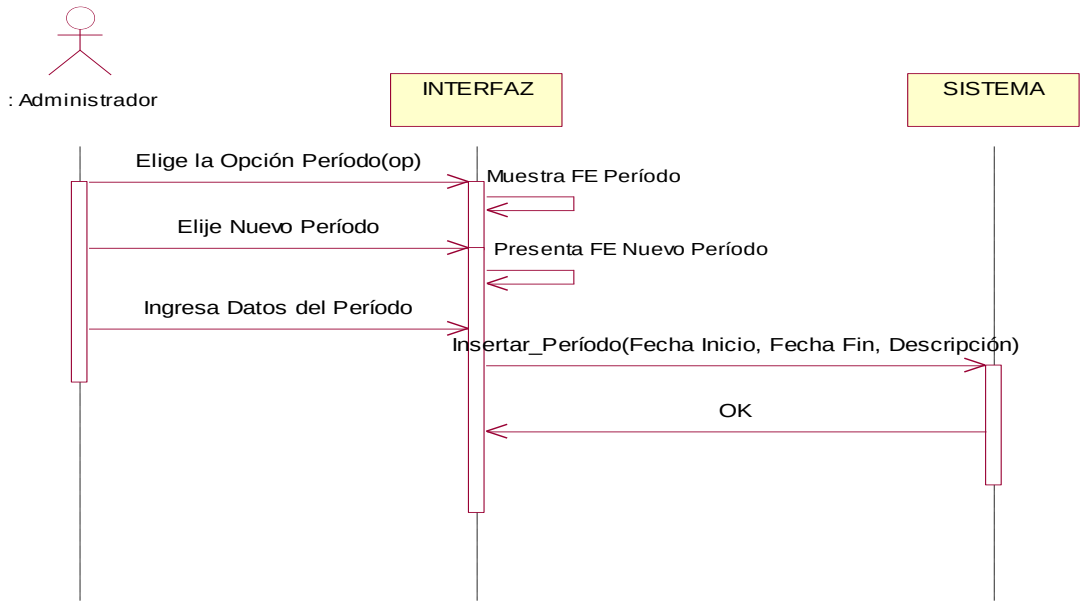

Figura 4.31. Diagrama de Secuencia Insertar Período

Diagramas de Colaboración

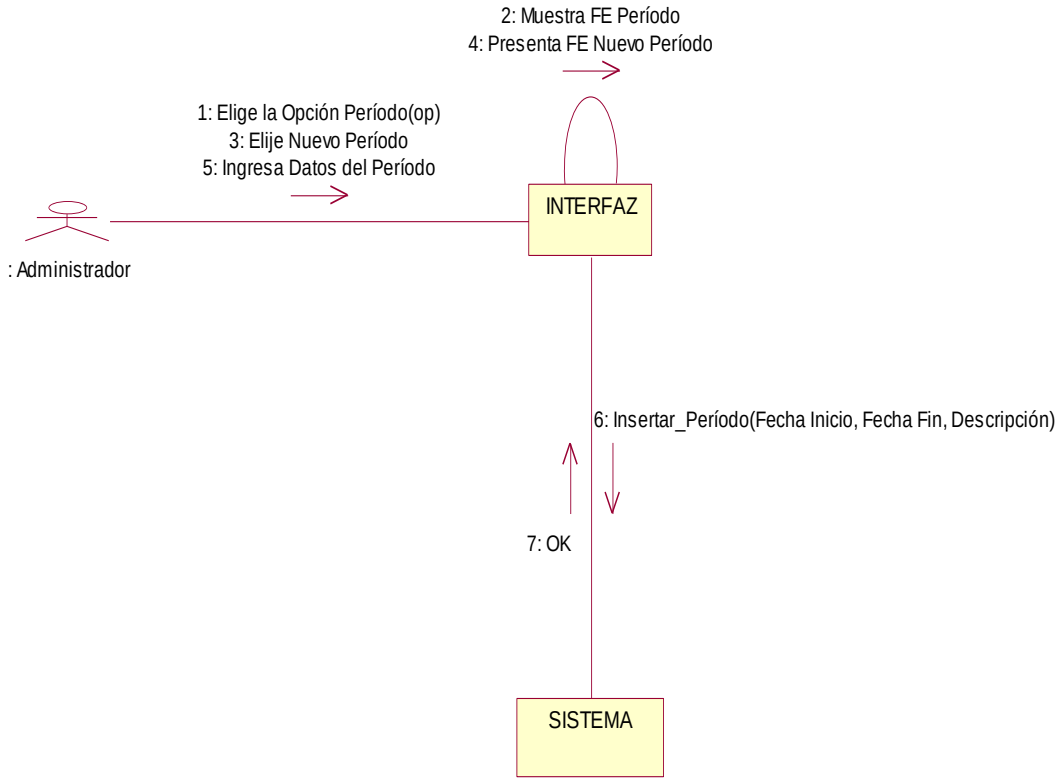

Figura 4.32. Diagrama de Colaboración Insertar Período

4.5.5.1.2.- Nombre del Caso de Uso: Modificar Período

Propósito: Modificar Período.
Actor: Administrador (iniciador)
Tipo: Primario Real.
Referencia: Req(12)

Visión General: El administrador selecciona la opción modificar del menú período, el sistema presenta formulario electrónico del período y podrá modificar los datos del período.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Seleccionar la opción modificar período del menú período(op)	2 Presenta formulario electrónico de búsqueda de período
3 Selecciona el período a modificar(per_código)	4 Presenta FE de modificar período con el resultado de la búsqueda
5 Ingresa nuevos datos.	6 Valida datos y modifica instancia.

Casos Alternativos

2* No existe FE de búsqueda, termina caso de uso

4* No existen datos de búsqueda, termina el caso de uso.

6* Datos incorrectos, no se modifica la instancia termina caso de uso.

Diagrama de Secuencia

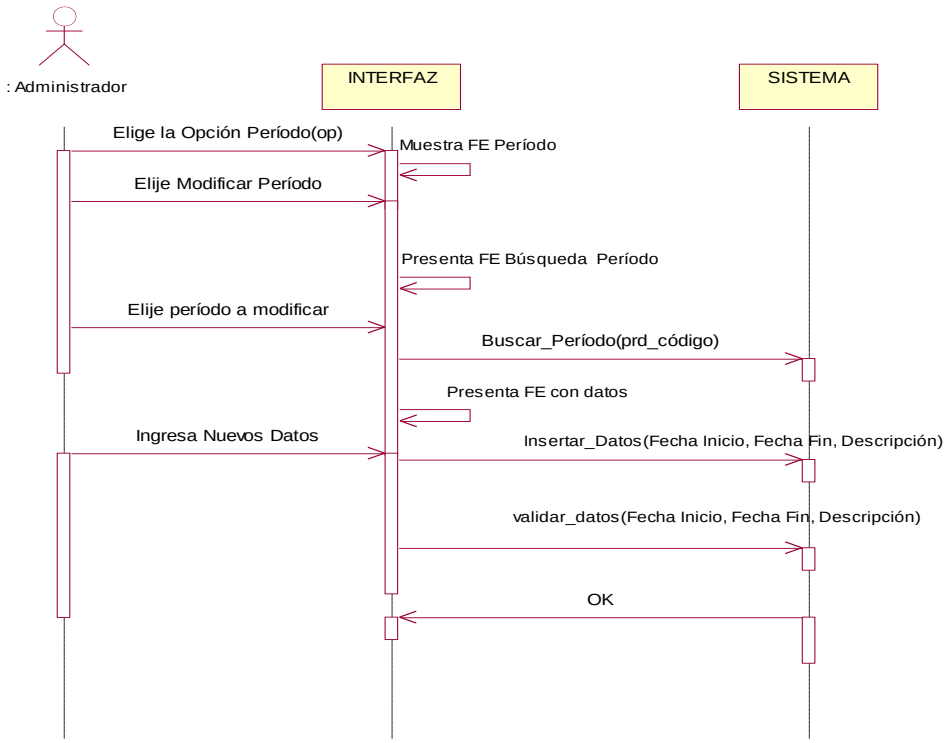

Figura 4.33. Diagrama de Colaboración Modificar Período

Diagramas de Colaboración

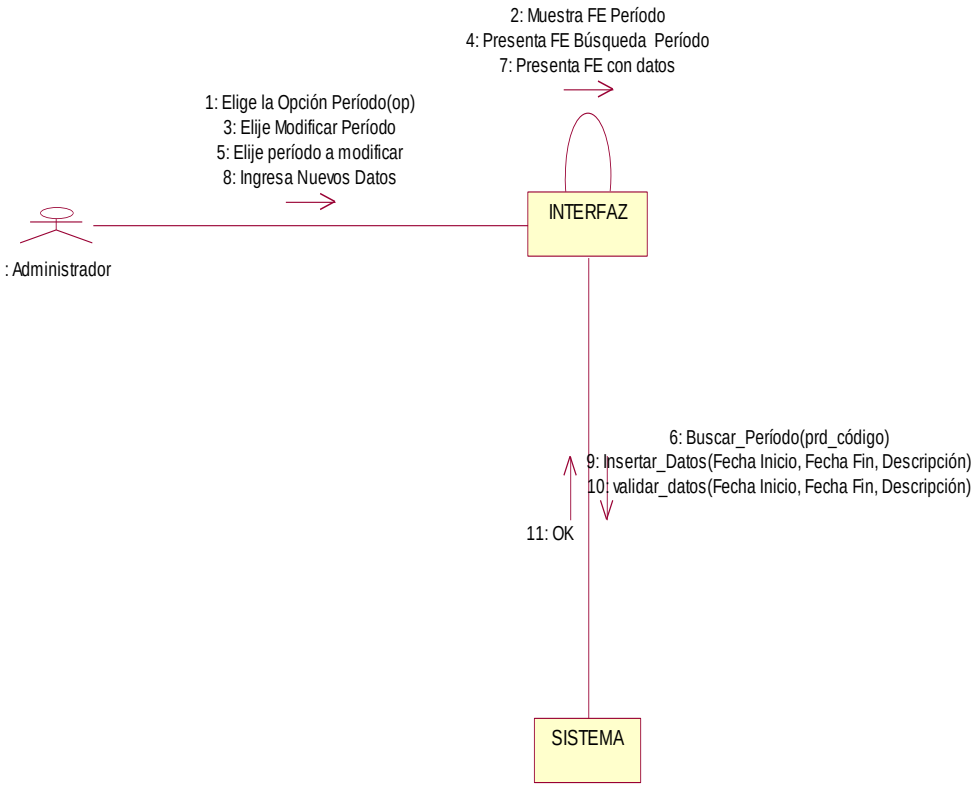

Figura 4.34. Diagrama de Colaboración Modificar Período

4.5.5.1.3.- Nombre del Caso de Uso: Eliminar Período

Propósito: Eliminar Período.
Actor: Administrador (iniciador)
Tipo: Primario Real
Referencia: Req(11)

Visión General: El administrador selecciona la opción eliminar del menú período, el sistema presenta formulario electrónico del período y podrá eliminar los datos del período si no tiene datos relacionales.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción eliminar periodo del menú período (op).	2 Presenta formulario electrónico búsqueda de período
3 Selecciona el período a eliminar(per_código)	4 Presenta FE de eliminación de período con resultados de búsqueda.
5 Selecciona la opción eliminar(per_código)	6 Eliminar instancia

Casos Alternativos

2* No existe FE, termina caso de uso.

4* No existen resultados de búsqueda, termina el caso de uso.

6* No existe independencia de datos, termina caso de uso.

Diagrama de Secuencia

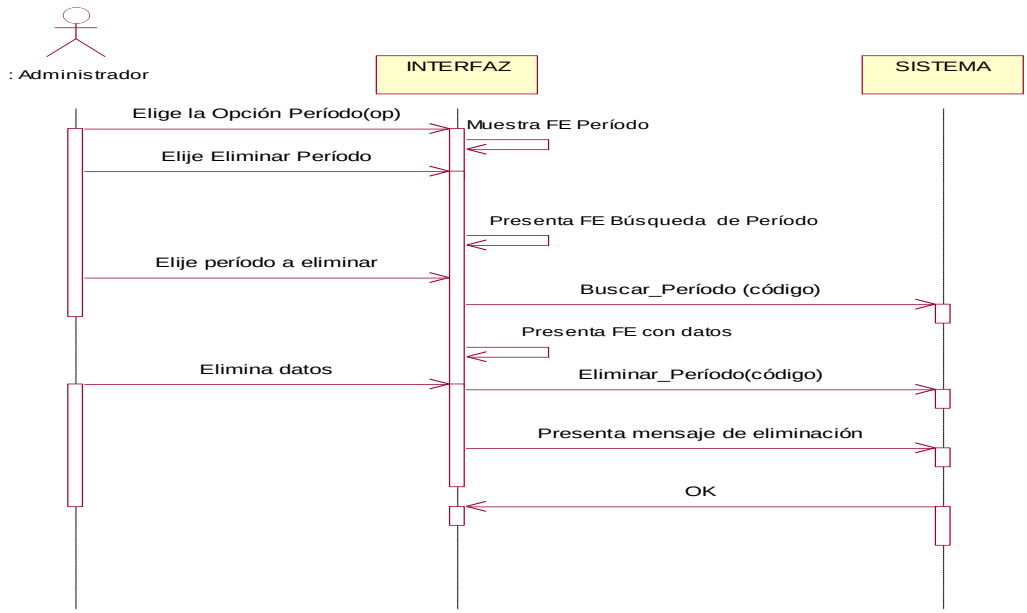

Figura 4.35. Diagrama de Secuencia Eliminar Período

Diagrama de Colaboración

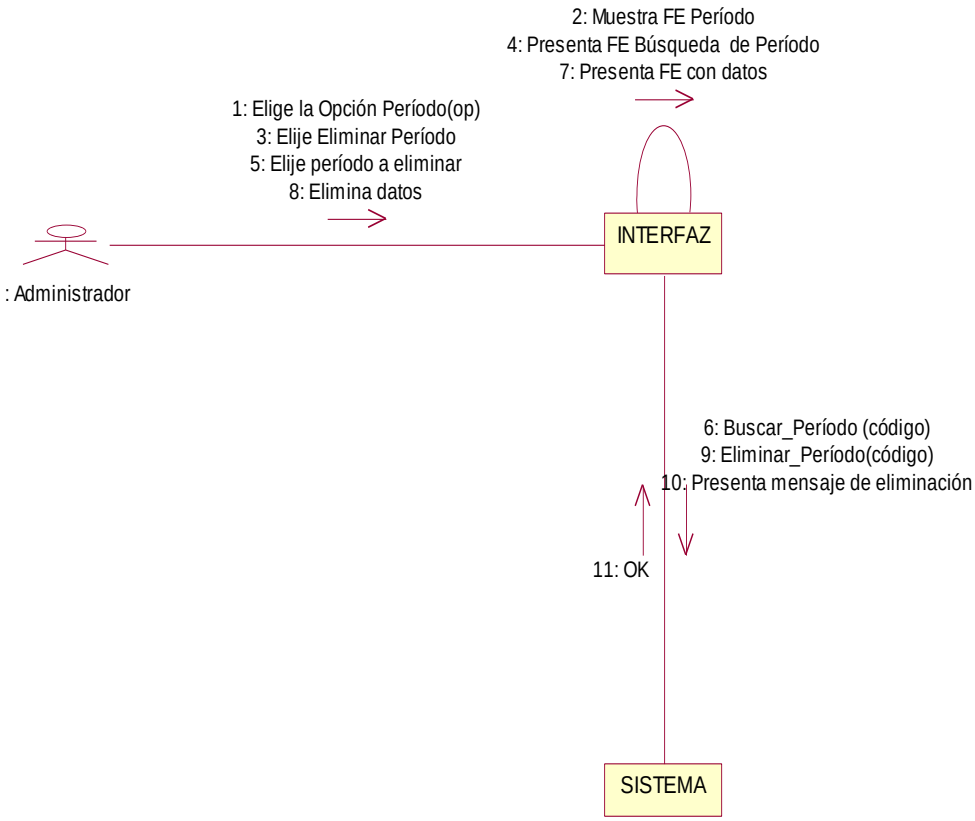

Figura 4.36. Diagrama de Colaboración Eliminar Período

4.5.6.- DIAGRAMA DE CASO DE USO GESTIÓN DE ÁREA

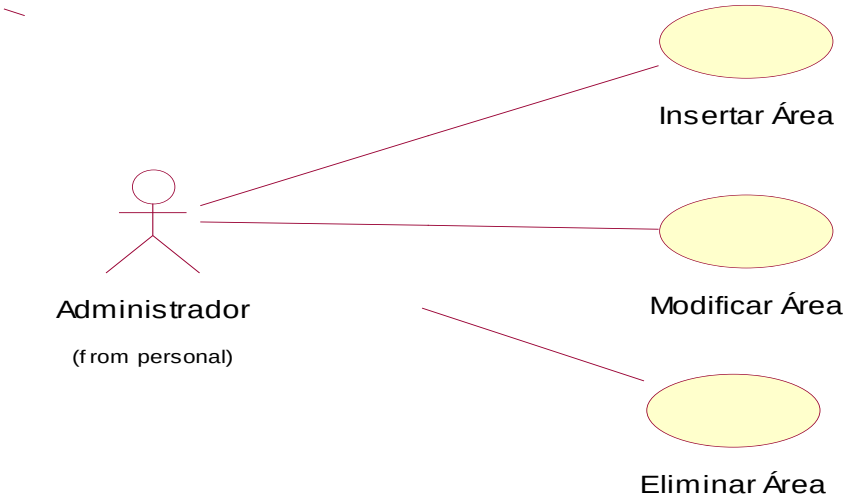

Figura 4.37. Diagrama Gestión de Área

4.5.6.1.- Definición de Casos de Uso Expandido

4.5.6.1.1.- Nombre del Caso de Uso: Insertar Área

Propósito: Insertar Área.

Actor: Administrador (iniciador)

Tipo: Primario real

Referencia: Req(16)

Visión General: El administrador selecciona la opción nuevo del menú área, el sistema presenta formulario electrónico del área y podrá ingresar los datos requeridos (Código, Nombre, Descripción) el sistema valida la información y crea una instancia área.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción nuevo del menú área(op)	2 Presentar formulario electrónico de la nueva área
3 Ingresa datos (Código, Nombre, Descripción)	4 Valida datos, crea instancia área

Casos Alternativos

2* No existe FE termina caso de uso
4* Ya existe el área, no se crea la instancia, termina el caso de uso.
4* Datos inválidos, no se crea la instancia, termina el caso de uso.

Diagrama de Secuencia

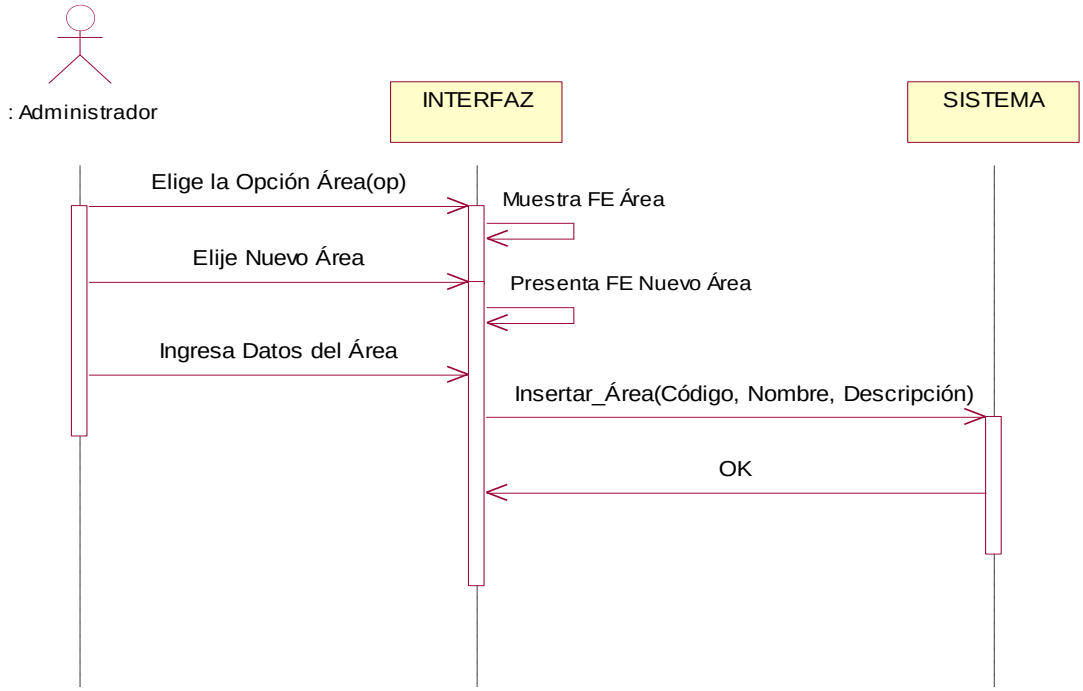

Figura 4.38. Diagrama de Secuencia Insertar Área

Diagrama de Colaboración

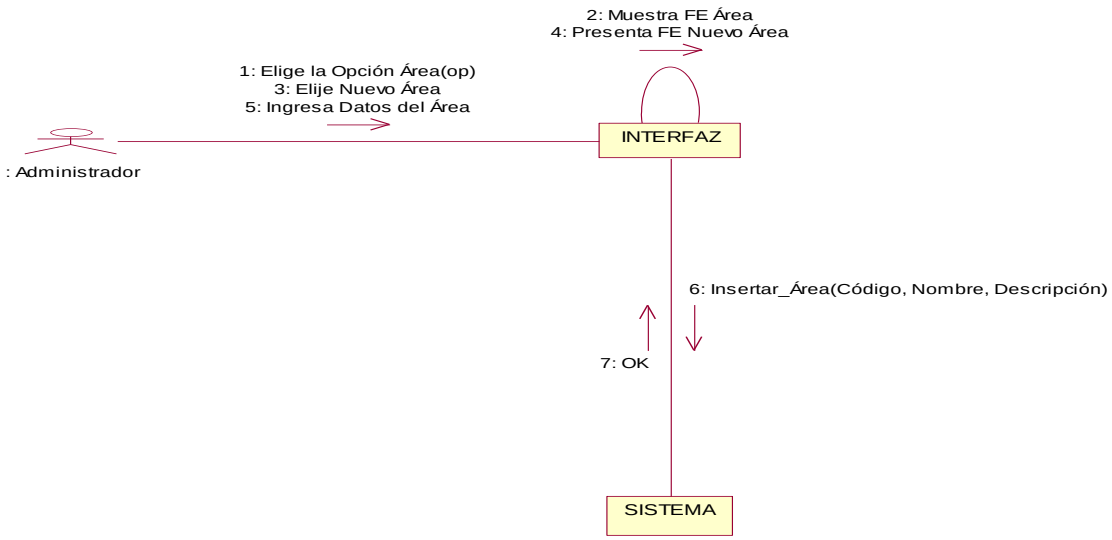

Figura 4.39. Diagrama de Colaboración Insertar Área

4.5.6.1.2.- Nombre del Caso de Uso: Modificar Área

Propósito: Modificar Área.

Actor: Administrador (iniciador)

Tipo: Primario Real

Referencia: Req(18)

Visión General: El administrador selecciona la opción modificar del menú área, el sistema presenta formulario electrónico del área y podrá modificar los datos del área.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Seleccionar la opción modificar área del menú área(op)	2 Presenta formulario electrónico de búsqueda de área
3 Selecciona el área a modificar(código)	4 Presenta FE de modificar área con el resultado de la búsqueda
5 Ingresa nuevos datos.	6 Valida datos y modifica instancia.

Casos Alternativos

2* No existe FE de búsqueda, termina caso de uso

4* No existen datos de búsqueda, termina el caso de uso.

6* Datos incorrectos, no se modifica la instancia termina caso de uso.

Diagramas se Secuencia

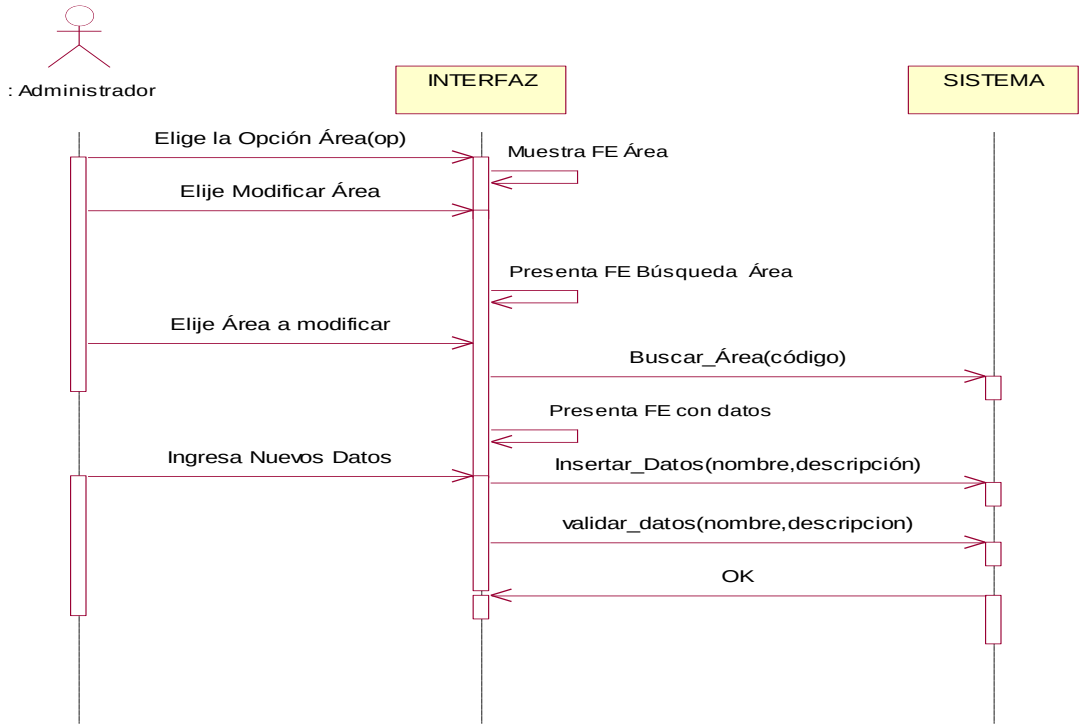

Figura 4.40. Diagrama de Secuencia Modificar Área

Diagrama de Colaboración

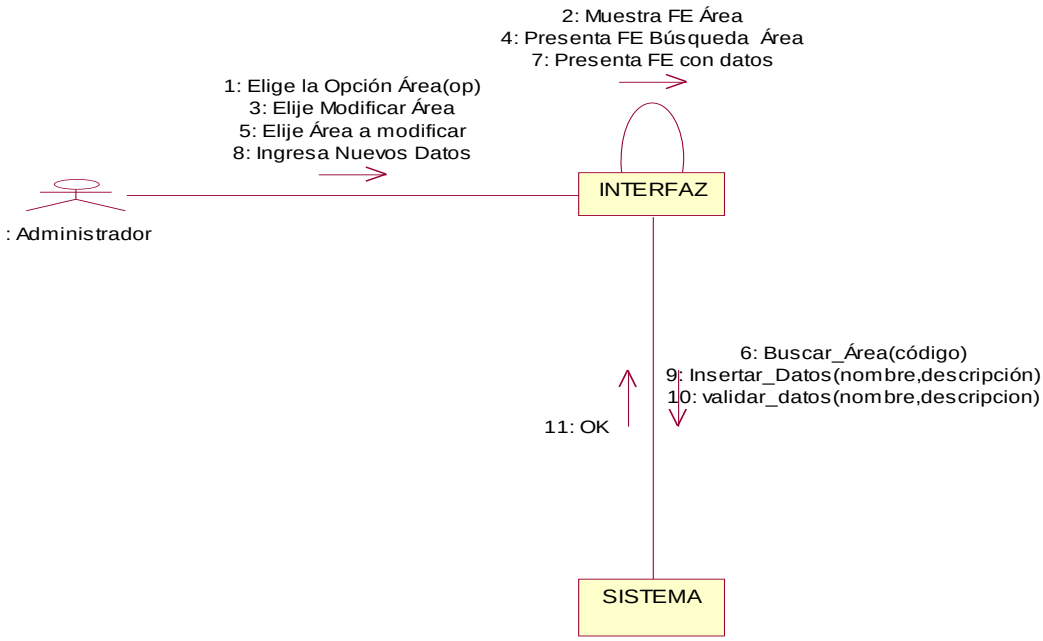

Figura 4.41. Diagrama de Colaboración Modificar Área

4.5.6.1.3.- Nombre del Caso de Uso: Eliminar

Propósito: Eliminar Área.

Actor: Administrador (iniciador)

Tipo: Primario real

Referencia: Req(17)

Visión General: El administrador selecciona la opción eliminar del menú área, el sistema presenta formulario electrónico del área y podrá eliminar los datos del área.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción eliminar área del menú área(op)	2 Presenta formulario electrónico búsqueda de área
3 Selecciona el área a eliminar(código)	4 Presenta FE de eliminación de área con resultados de la búsqueda.
5 Selecciona la opción eliminar (código)	6 Eliminar instancia

Casos Alternativos

2* No existe FE, termina caso de uso.

4* No existen resultados de búsqueda, termina el caso de uso.

6* No existe independencia de datos, termina caso de uso.

Diagrama de Secuencia

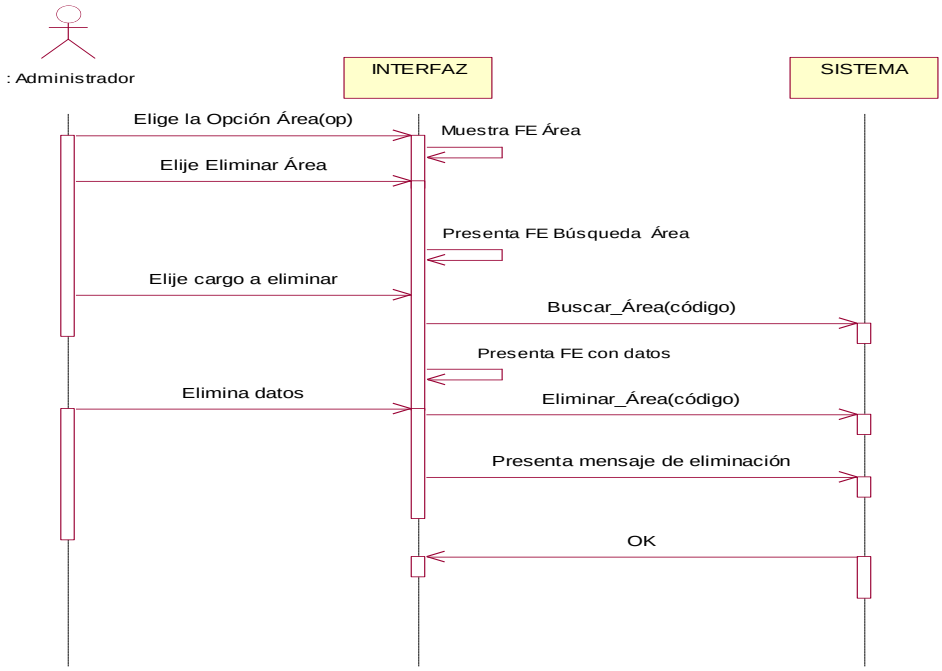

Figura 4.42. Diagrama de Secuencia Eliminar Área

Diagrama de Colaboración

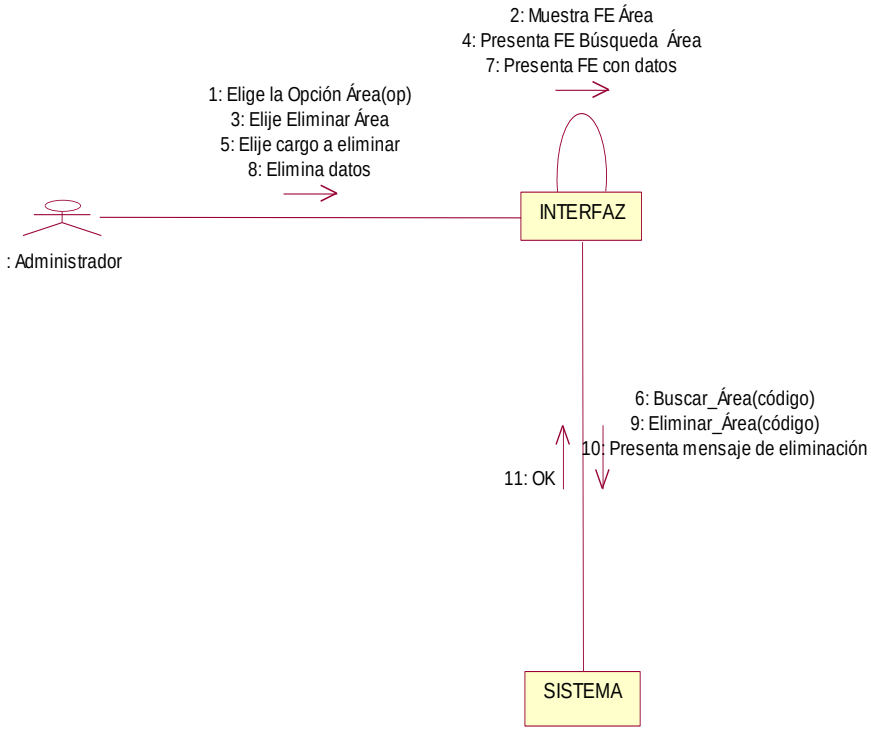

Figura 4.43. Diagrama de Colaboración Eliminar Área

4.5.7.- DIAGRAMA DE CASO DE USO GESTIÓN DE DEPARTAMENTO

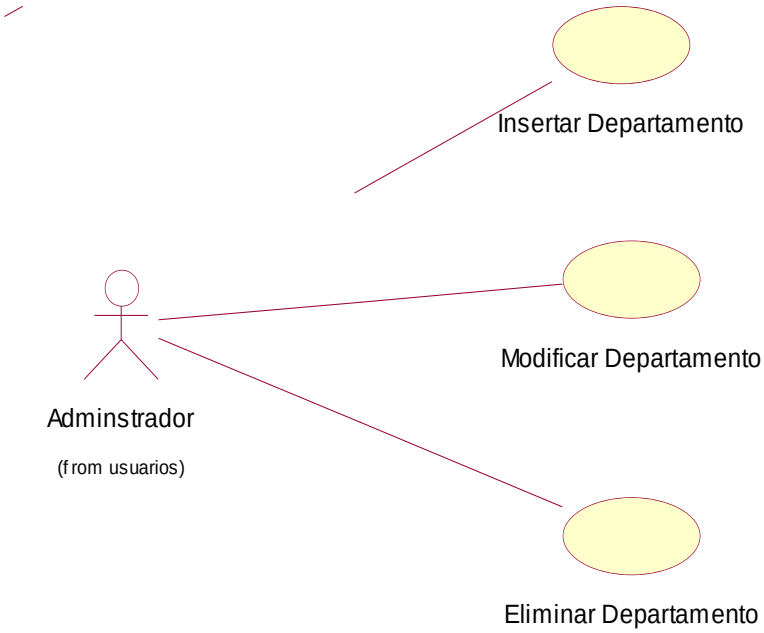

Figura 4.43. Diagrama de Gestión de Departamento

4.5.7.1.- Definición de Casos de Uso Expandido

4.5.7.1.1.- Nombre del Caso de Uso: Insertar Departamento

Propósito: Insertar Departamento.
Actor: Administrador (iniciador)
Tipo: Primario Real
Referencia: Req(01)

Visión General: El administrador selecciona la opción nuevo del departamento, el sistema presenta formulario electrónico del departamento y podrá ingresar los datos requeridos (Código, Nombre, Director, Descripción) el sistema valida la información y crea una instancia departamento.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción nuevo del menú departamento (op)	2 Presentar formulario electrónico de nuevo departamento
3 Ingresa datos (Código, Nombre, Director, Descrpción)	4 Valida datos crea instancia departamento

Casos Alternativos

2* No existe FE termina caso de uso

4* Ya existe el departamento, no se crea la instancia. Termina el caso de uso.

4* Datos inválidos, no se crea a instancia. Termina el caso de uso.

Diagrama de Secuencia

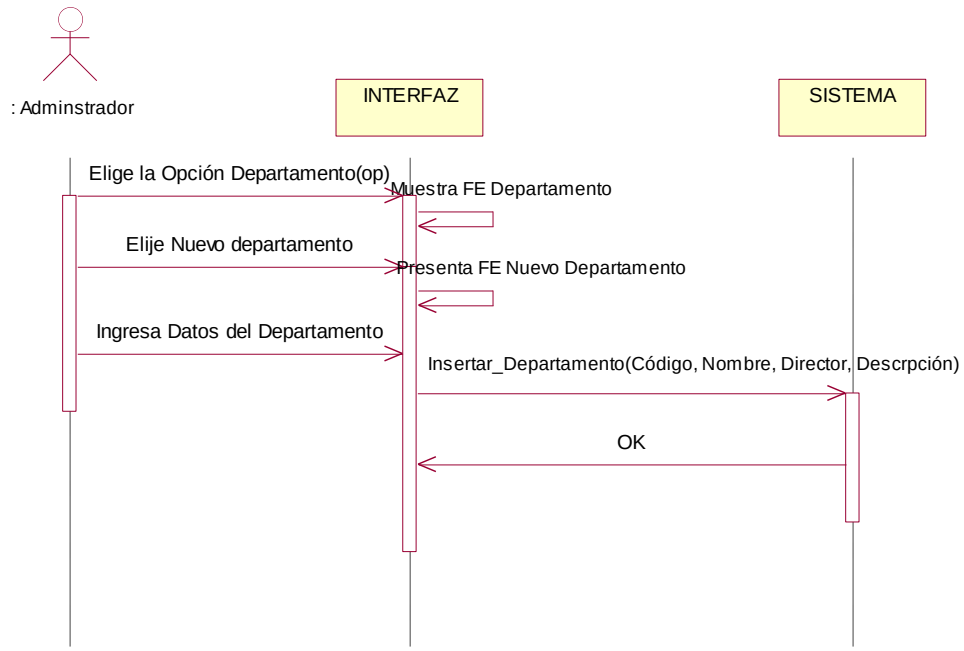

Figura 4.44. Diagrama de Secuencia Insertar Departamento

Diagrama de Colaboración

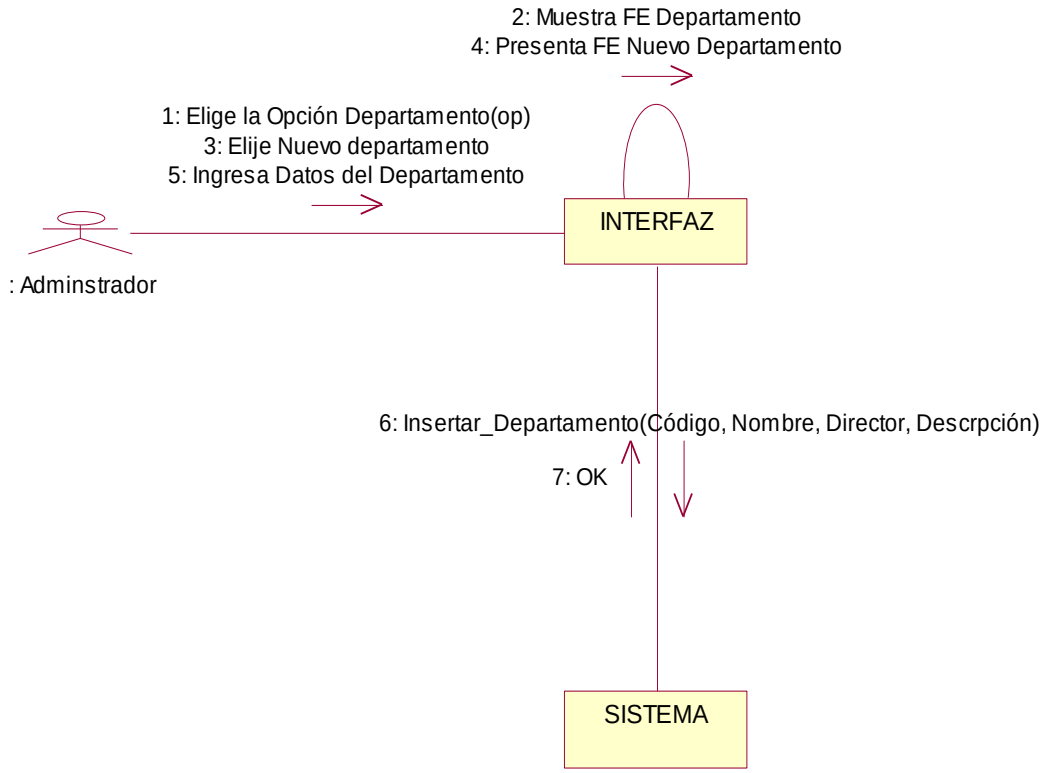

Figura 4.45. Diagrama de Colaboración Insertar Departamento

4.5.7.1.2.- Nombre del Caso de Uso: Modificar Departamento

Propósito: Modificar Departamento.

Actor: Administrador (iniciador)

Tipo: Primario Real.

Referencia: Req(03)

Visión General: El administrador selecciona la opción modificar del menú departamento, el sistema presenta formulario electrónico del departamento, y podrá modificar los datos del departamento.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Seleccionar la opción modificar departamento del menú departamento (op)	2 Presenta formulario electrónico de búsqueda del usuario
3 Selecciona el departamento a modificar(código)	4 Presenta FE de modificar con datos del departamento
5 Ingresa nuevos datos.	6 Valida datos y modifica instancia.

Casos Alternativos

2* No existe FE de búsqueda, termina caso de uso

4* No existen datos de búsqueda, termina el caso de uso.

6* Datos incorrectos, no se modifica la instancia termina caso de uso.

Diagrama de Secuencia

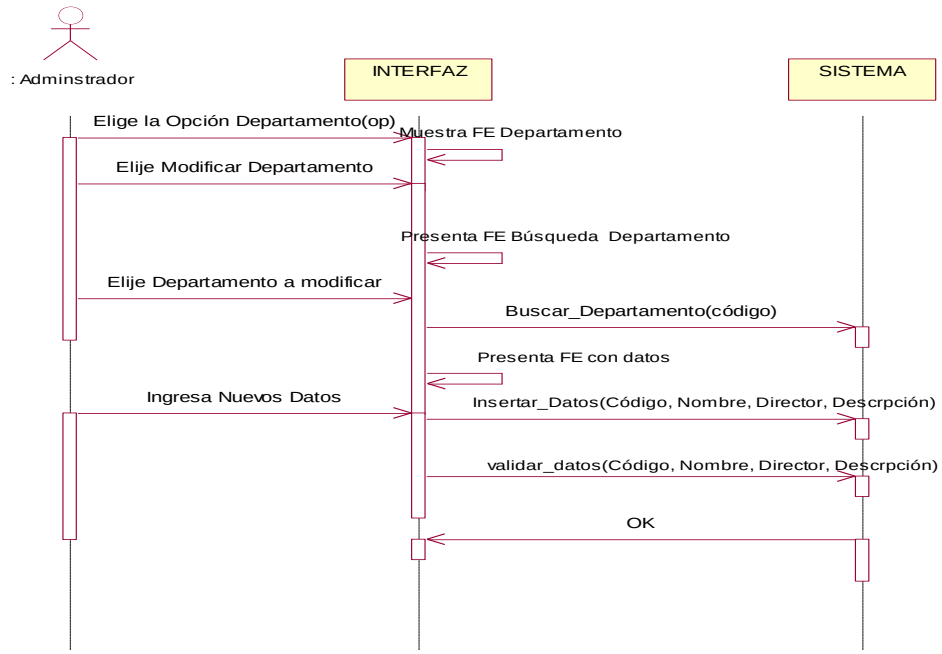

Figura 4.46. Diagrama de Secuencia Modificar Departamento

Diagrama de Colaboración

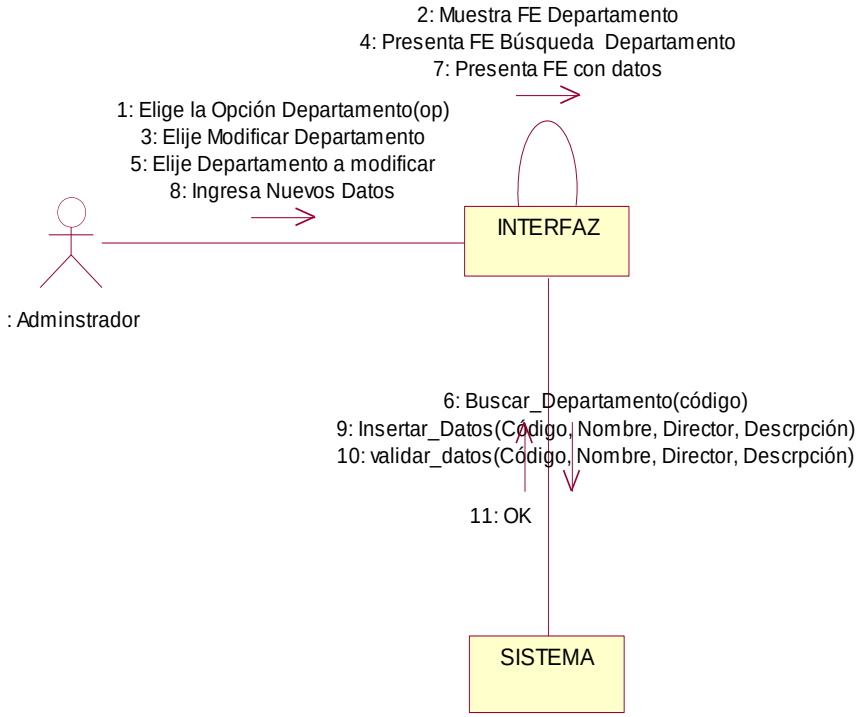

Figura 4.47. Diagrama de Colaboración Modificar Departamento

4.5.7.1.3.- Nombre del Caso de Uso: Eliminar Departamento.

Propósito: Eliminar Departamento.

Actor: Administrador (iniciador)

Tipo: Primario real

Referencia: Req(02)

Visión General: El administrador selecciona la opción eliminar del menú departamento, el sistema presenta formulario electrónico del departamento, y podrá eliminar los datos del departamento.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción eliminar departamento, del menú departamento(op)	2 Presenta formulario electrónico búsqueda de departamento
3 Selecciona el departamento a eliminar(código)	4 Presenta FE de eliminación de departamento, con resultados de búsqueda.
5 Selecciona la opción eliminar(usu_cédula)	6 Eliminar instancia

Casos Alternativos

2* No existe FE, termina caso de uso.

4* No existen resultados de búsqueda, termina el caso de uso.

6* No existe independencia de datos, termina caso de uso.

Diagrama de Secuencia

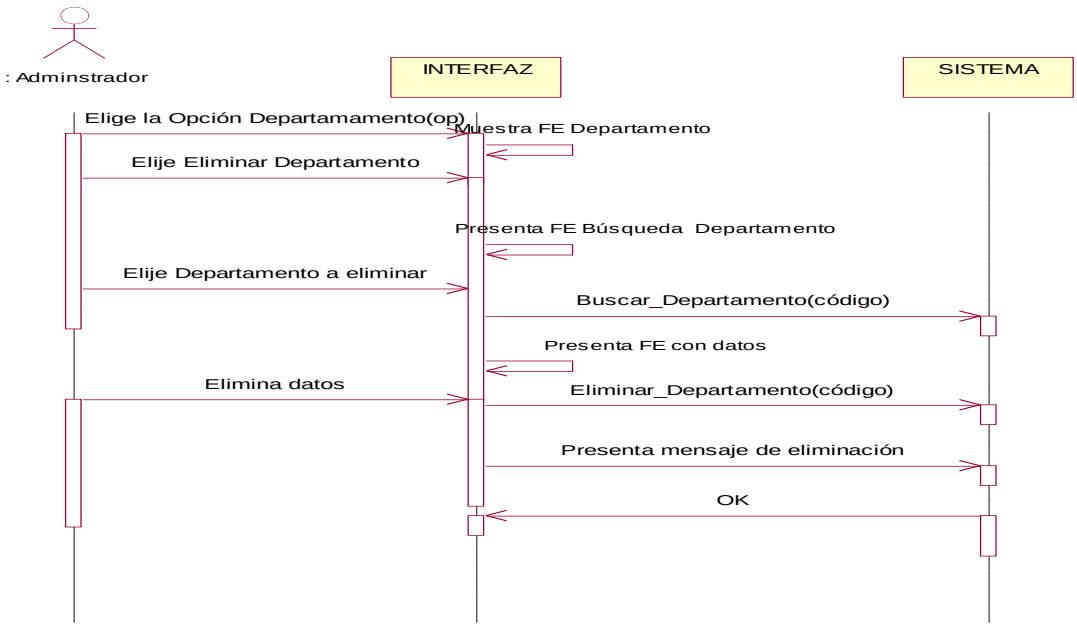

Figura 4.48. Diagrama de Secuencia Eliminar Departamento

Diagrama de Colaboración

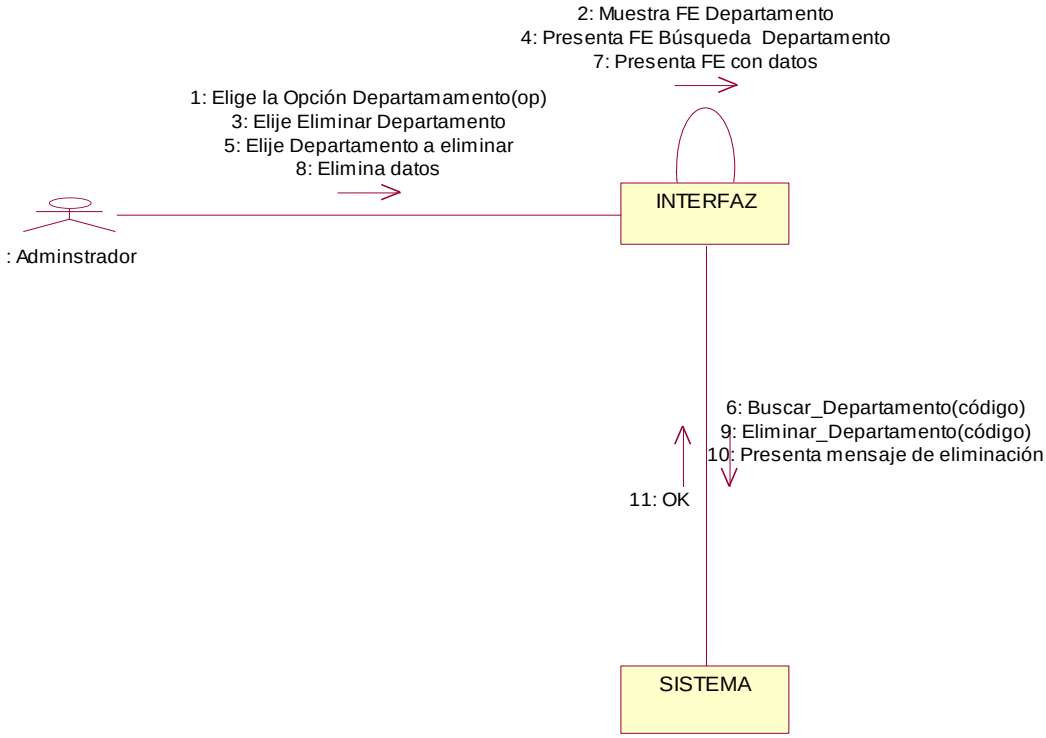

Figura 4.49. Diagrama de Colaboración Departamento

4.5.8.- DIAGRAMA DE CASO DE USO GESTIÓN DE ASIGNAR MATERIA

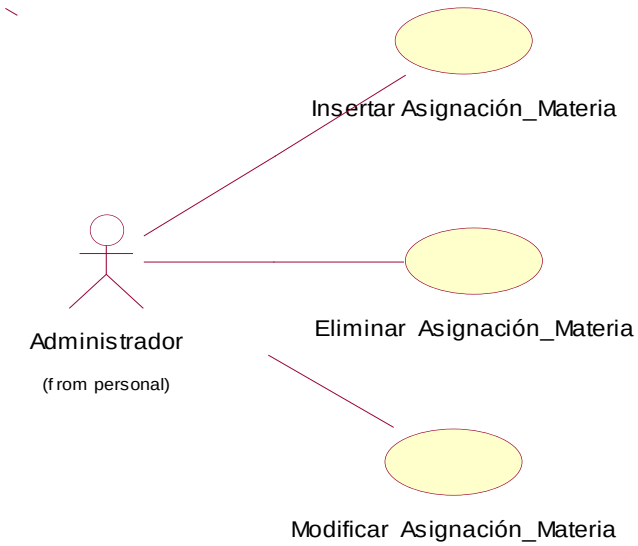

Figura 4.50. Diagrama de Asignar Materia

4.5.8.1.- Definición de Casos de Uso Expandido

4.5.8.1.1.- Nombre del Caso de Uso: Insertar una Asignación de Materia

Propósito: Insertar Asignación Materia

Actor: Administrador, Coordinador (iniciador)

Tipo: Primario real

Referencia: Req(22)

Visión General: El usuario selecciona la opción nuevo del menú asignación materia, el sistema presenta formulario electrónico se sebe escoger carrera, docente, período y paralelo como también podrá ingresar los datos requeridos (Paralelo, Día, Hora de Inicio Hora de Fin.) el sistema valida la información y crea una instancia asignación materia

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción nuevo del menú asignación materia(op)	2 Presentar formulario electrónico de nuevo asignación materia
3 Ingresa datos (Paralelo, Día, Hora de Inicio Hora de Fin)	4 Valida datos crea instancia asignación materia

Casos Alternativos

2* No existe FE termina caso de uso

4* Ya existe la asignación, no se crea la instancia. Termina el caso de uso.

4* Datos inválidos, no se crea la instancia. Termina el caso de uso.

Diagrama de Secuencia

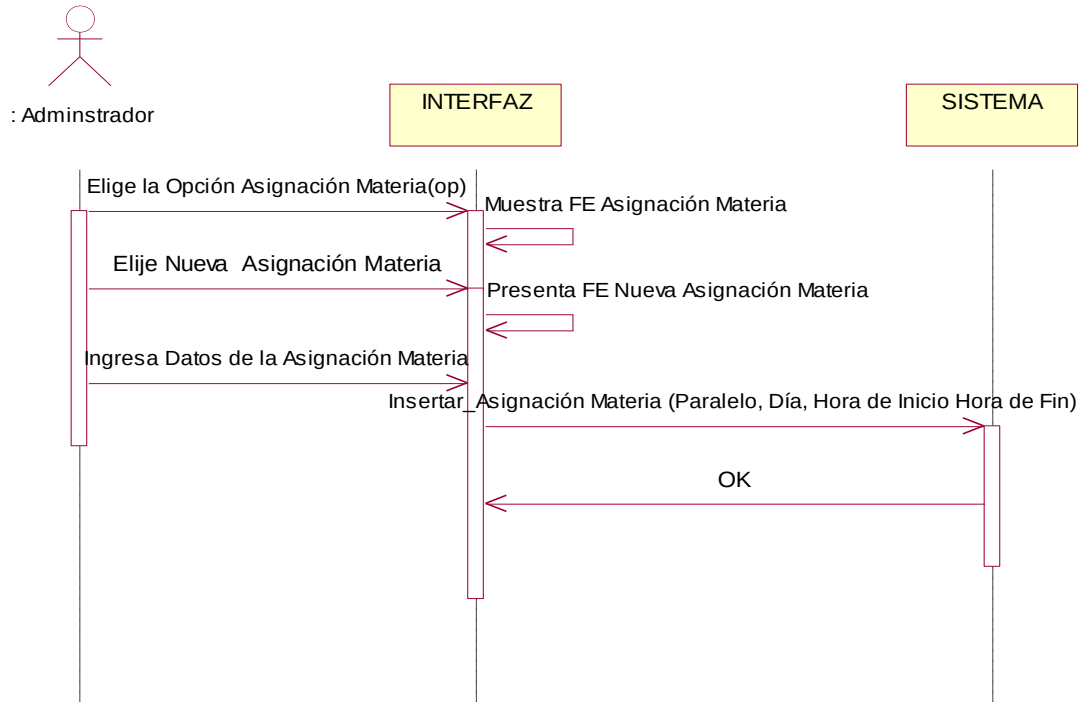

Figura 4.51. Diagrama de Secuencia Insertar Asignación de Materia

Diagrama de Colaboración

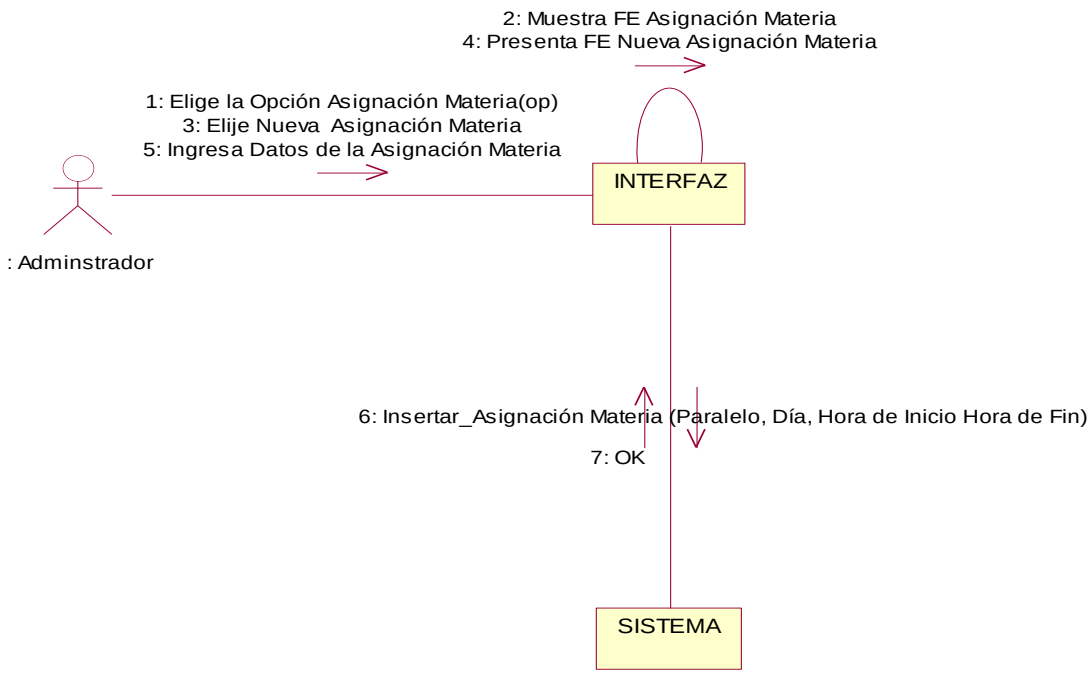

Figura 4.52. Diagrama de Colaboración Insertar Asignación de Materia

4.5.8.1.2.- Nombre del Caso de Uso: Modificar Asignación de Materia

Propósito: Modificar Asignación de Materia.

Actor: Administrador, Coordinador (iniciador)

Tipo: Primario real

Referencia: Req(24)

Visión General: El administrador selecciona la opción modificar del menú asignar materia, el sistema presenta formulario electrónico de la asignación de la materia y podrá modificar los datos de la asignación de materia.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Seleccionar la opción modificar del menú asignación materia(op)	2 Presenta formulario electrónico de búsqueda de la asignación materia
3 Selecciona la asignación materia a modificar(código)	4 Presenta FE de modificar con datos de la asignación materia
5 Ingresa nuevos datos.	6 Valida datos y modifica instancia.

Casos Alternativos

2* No existe FE de búsqueda, termina caso de uso

4* No existen datos de búsqueda, termina el caso de uso.

6* Datos incorrectos, no se modifica la instancia termina caso de uso.

Diagrama de Secuencia

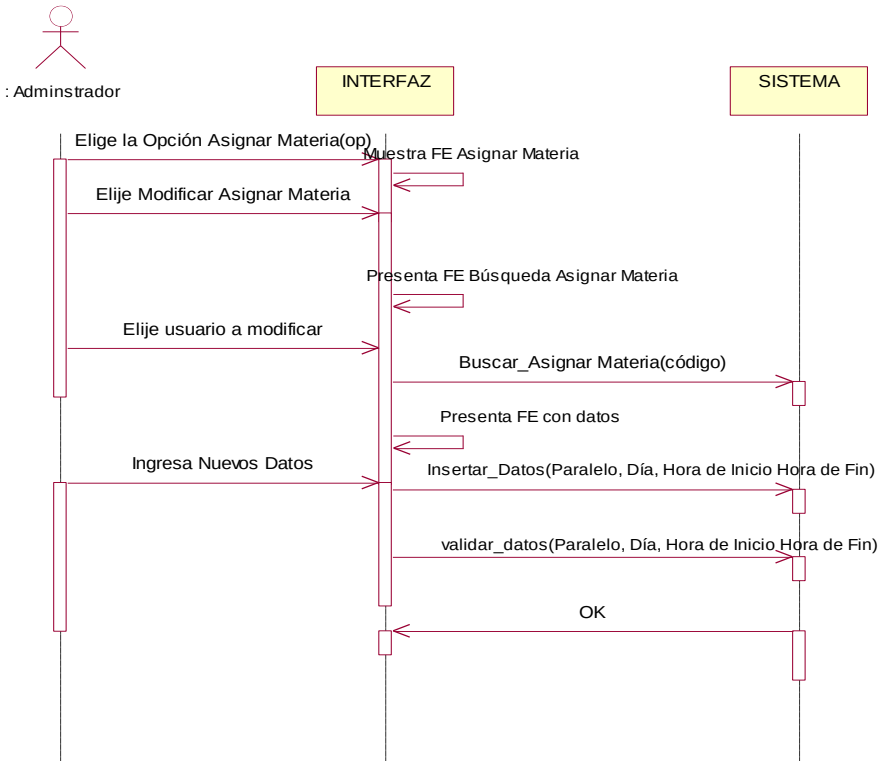

Figura 4.53. Diagrama de Colaboración Modificar Asignación de Materia

4.5.8.1.3.- Nombre del Caso de Uso: Eliminar Asignación de Materia

Propósito: Eliminar Asignación de Materia

Actor: Administrador (iniciador)

Tipo: Primario real

Referencia: Req(23)

Visión General: El administrador selecciona la opción eliminar del menú asignación materia el sistema presenta formulario electrónico de asignación y podrá eliminar los datos de la asignación de materia

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción eliminar del menú asignación de materia(op)	2 Presenta formulario electrónico búsqueda de asignación de materia
3 Selecciona la asignación de materia eliminar(código)	4 Presenta FE de eliminación de asignación de materia con resultados de búsqueda.
5 Selecciona la opción eliminar(código)	6 Eliminar instancia

Casos Alternativos

2* No existe FE, termina caso de uso.

4* No existen resultados de búsqueda, termina el caso de uso.

6* No existe independencia de datos, termina caso de uso.

Diagrama de Secuencia

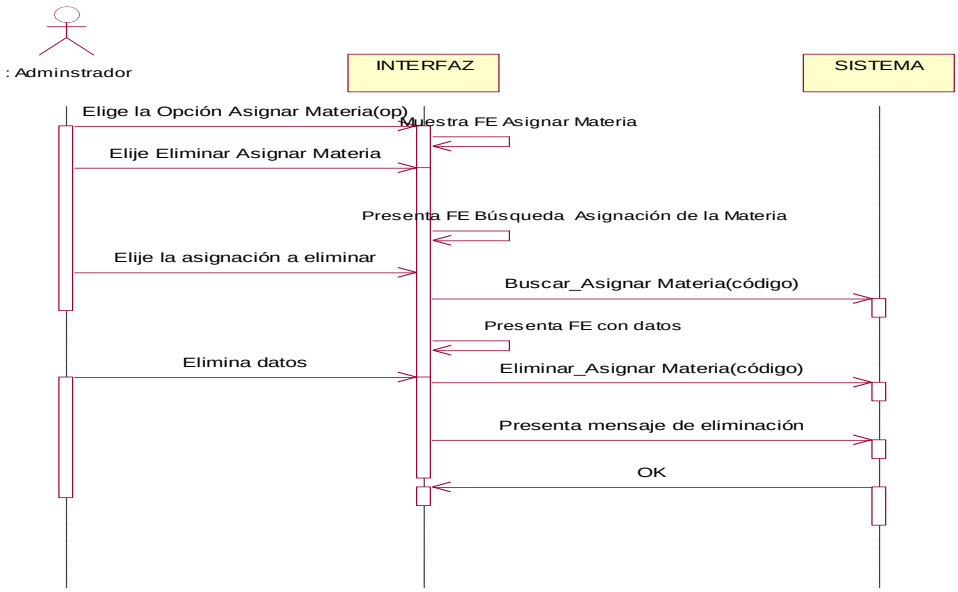

Figura 4.54. Diagrama de Colaboración Eliminar Asignación de Materia

Diagrama de Colaboración

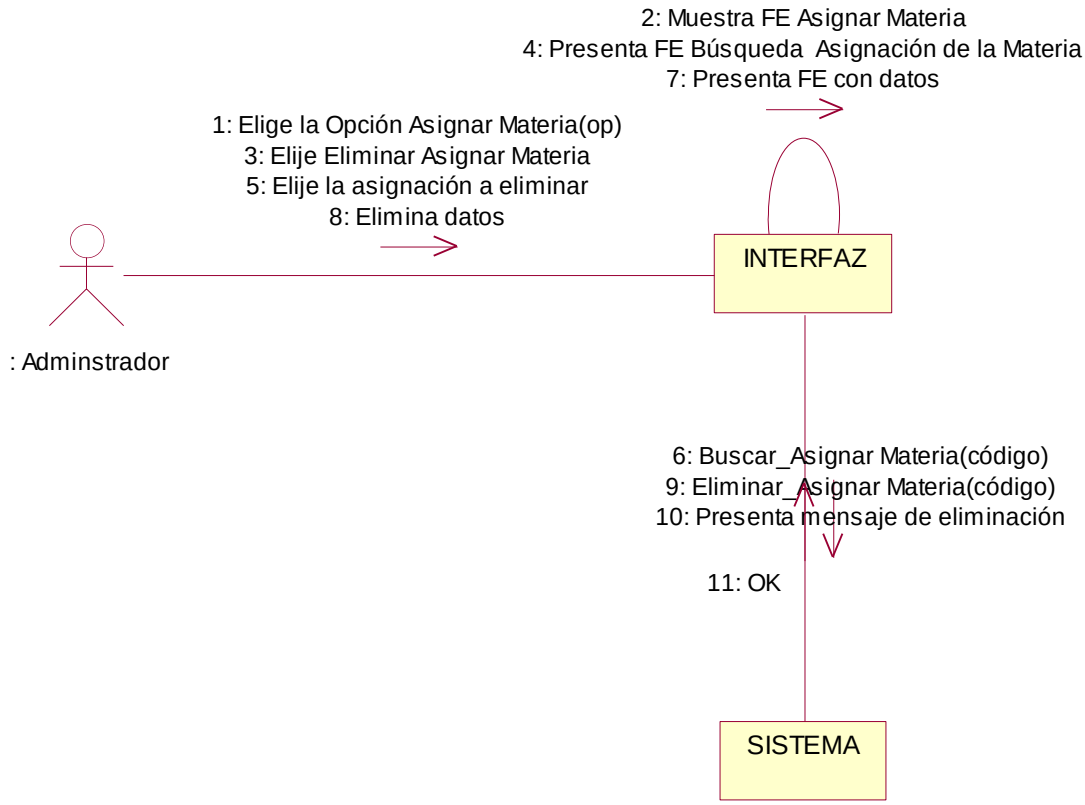

Figura 4.55. Diagrama de Colaboración Eliminar Asignación de Materia

4.5.9.- DIAGRAMA DE CASO DE USO GESTIÓN DE REGISTRO

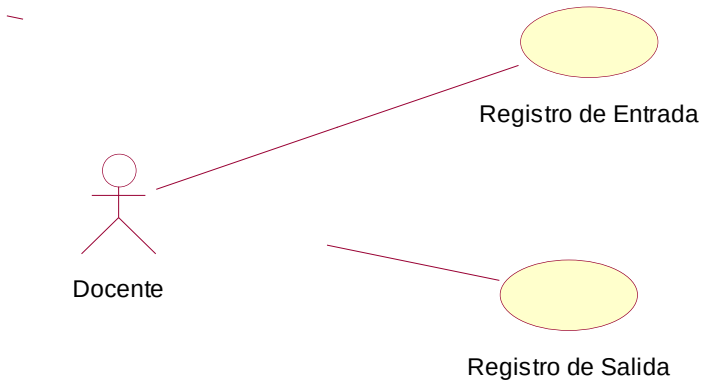

Figura 4.56. Diagrama de Gestión de Registro

4.5.9.1 Definición de Casos de Uso Expandido

4.5.9.1.1.- Nombre del Caso de Uso: Registro de Entrada

Propósito: Permitir al docente registre su hora de entrada a clases.

Actor: Docente (iniciador)

Tipo: Primario real

Referencia: Req(25)

Visión General: El docente selecciona la opción asistencia del menú principal, el sistema presenta formulario electrónico de la asistencia deberá ingresar el número de cédula y password, podrá registrar la hora de entrada y crea una instancia hora de entrada.

Curso Típico de Eventos

ACTOR	**SISTEMA**
1 Selecciona la opción asistencia del menú principal (op)	2 Presentar formulario electrónico de asistencia.
3 Ingresa datos (número de cedula, password)	4 Valida datos, presenta FE de asistencia
5 Selecciona asistencia	Captura hora del sistema.

Casos Alternativos

2* No existe FE termina caso de uso

4* Datos inválidos, no se crea la instancia, termina el caso de uso.

6* No captura la hora termina el caso de uso

Diagrama de Secuencia

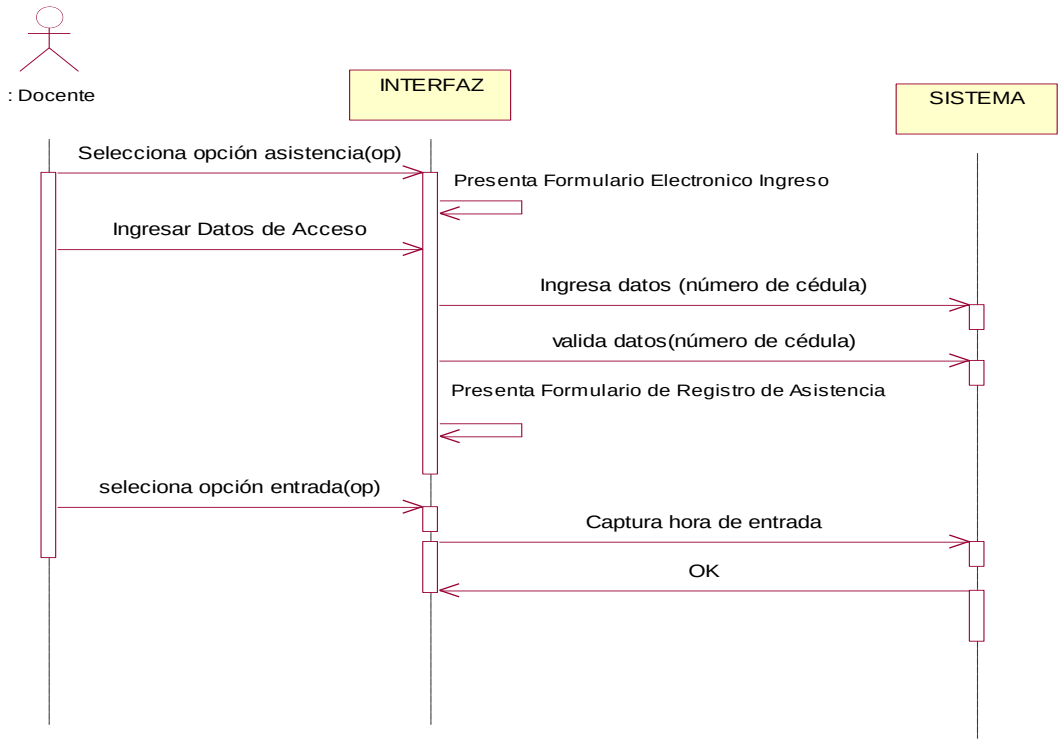

Figura 4.57. Diagrama de Secuencia Registro Entrada

Diagrama de Colaboración

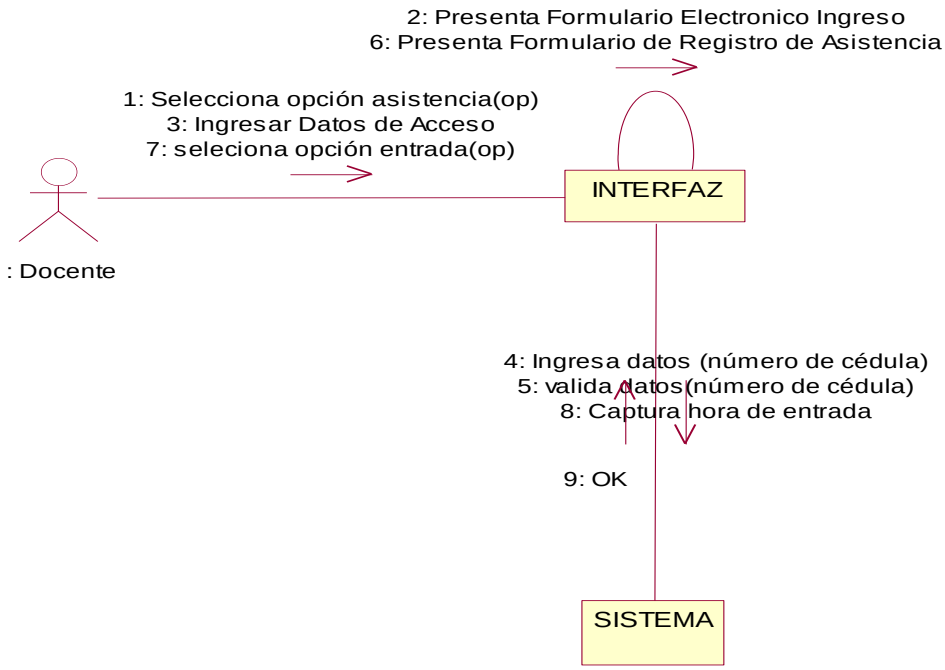

Figura 4.58. Diagrama de Colaboración Registro Entrada

4.5.9.1.2.- Nombre del Caso de Uso: Registro de Salida

Propósito: Permitir al docente registre su hora de salida de clases.

Actor: Docente (iniciador)

Tipo: Primario real

Referencia: Req(26)

Visión General: El docente selecciona la opción asistencia del menú principal, el sistema presenta formulario electrónico de la asistencia deberá ingresar el número de cédula y password, podrá registrar la hora de salida y crea una instancia hora de salida.

Curso Típico de Eventos

ACTOR	SISTEMA
1 Selecciona la opción asistencia del menú principal (op)	2 Presenta FE de asistencia.
3 Ingresa datos (número de cedula, password)	4 Valida datos, presenta FE de asistencia
5 Ingresa datos (tema tratado, alumnos faltos, observaciones)	6 Valida datos, presenta FE de asistencia
7 Selecciona asistencia	8 Captura hora del sistema.

Casos Alternativos

2* No existe FE termina caso de uso

4* Datos inválidos, no se crea la instancia, termina el caso de uso.

6* Datos inválidos, no se crea la instancia, termina el caso de uso.

8* No captura la hora termina el caso de uso

Diagrama de Secuencia

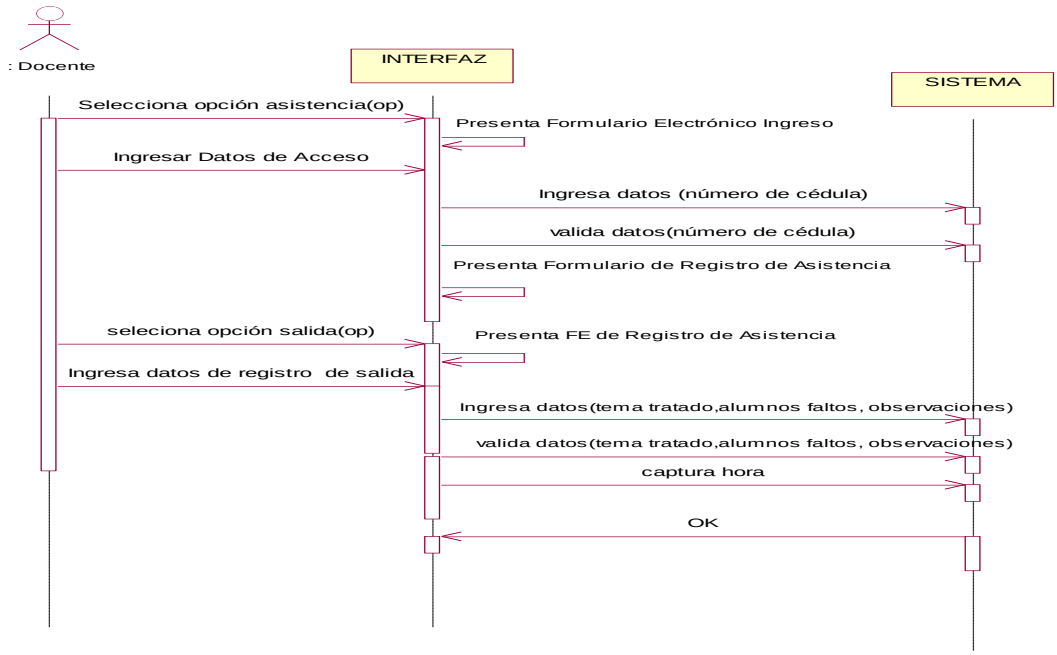

Figura 4.59. Diagrama de Secuencia Registro Salida

Diagrama de Colaboración

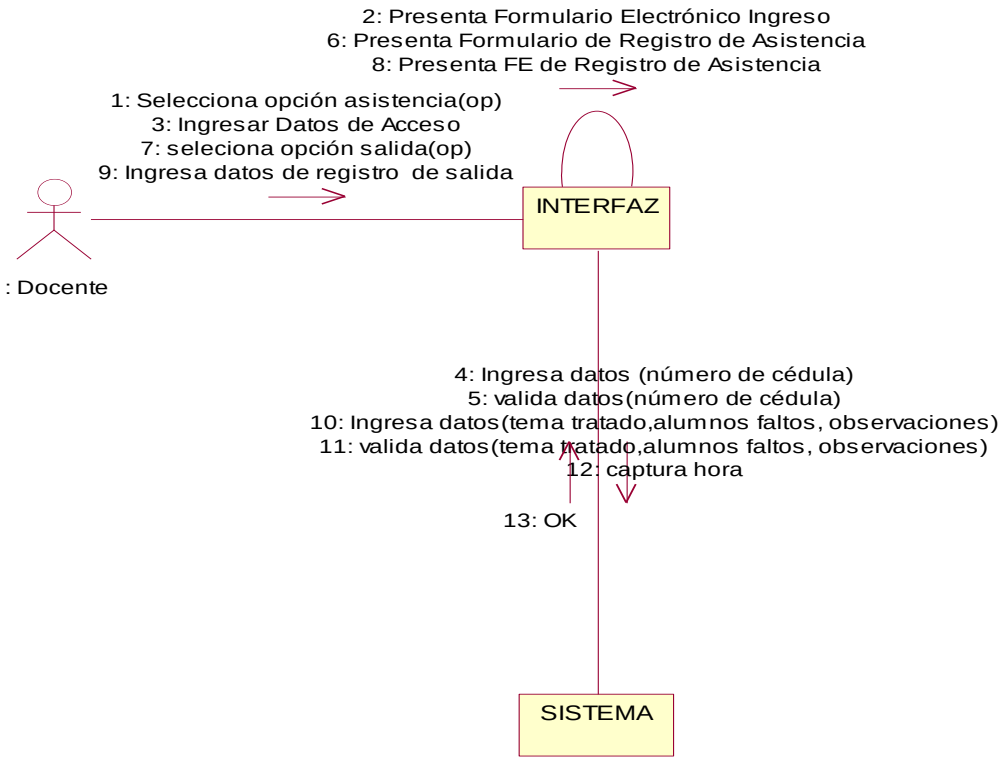

Figura 4.59. Diagrama de Secuencia Registro Salida

4.6.- DIAGRAMA DE CLASES

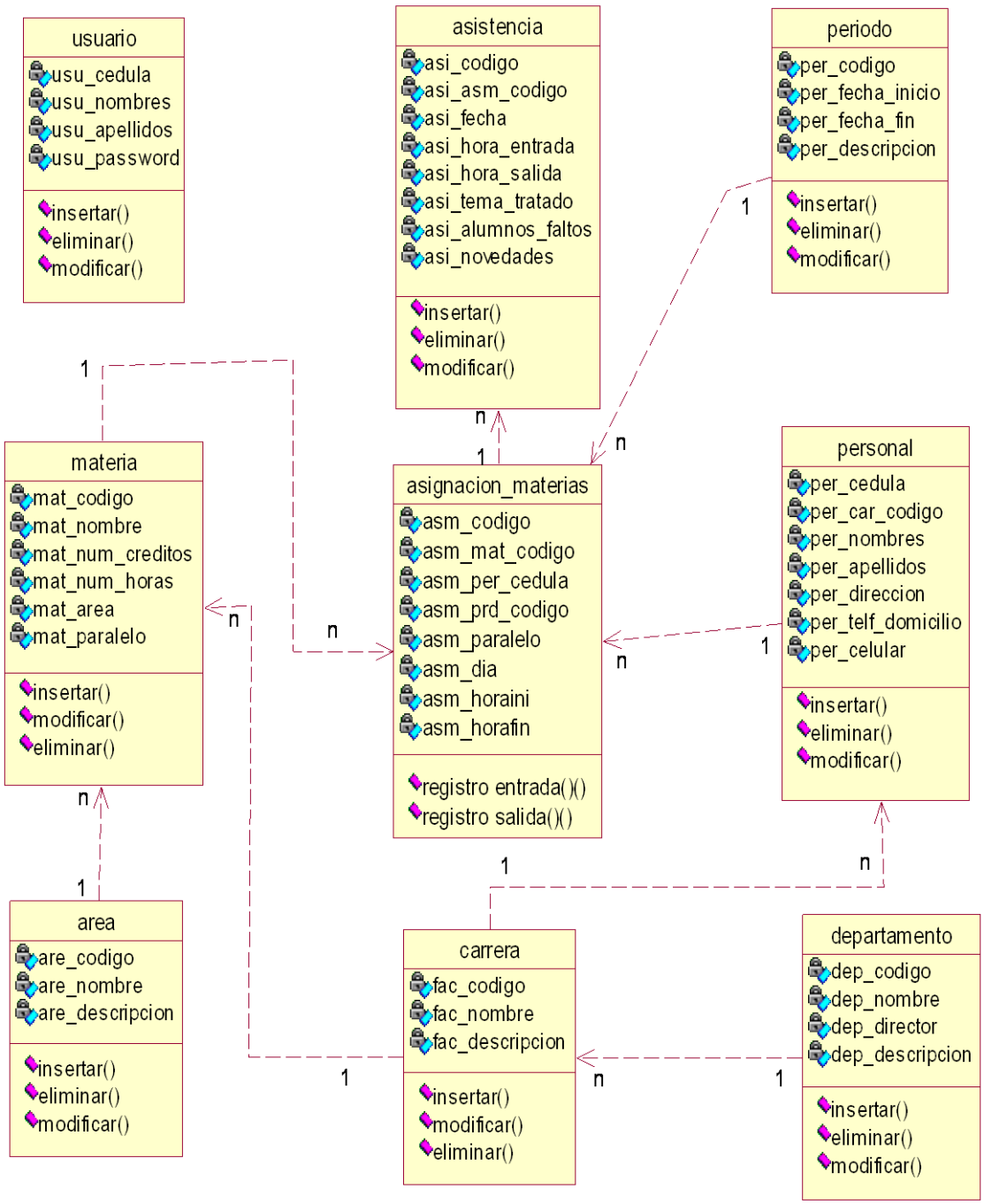

Figura 4.60. Diagrama de Clases

4.7 DIAGRAMA DE LA BASE DE DATOS

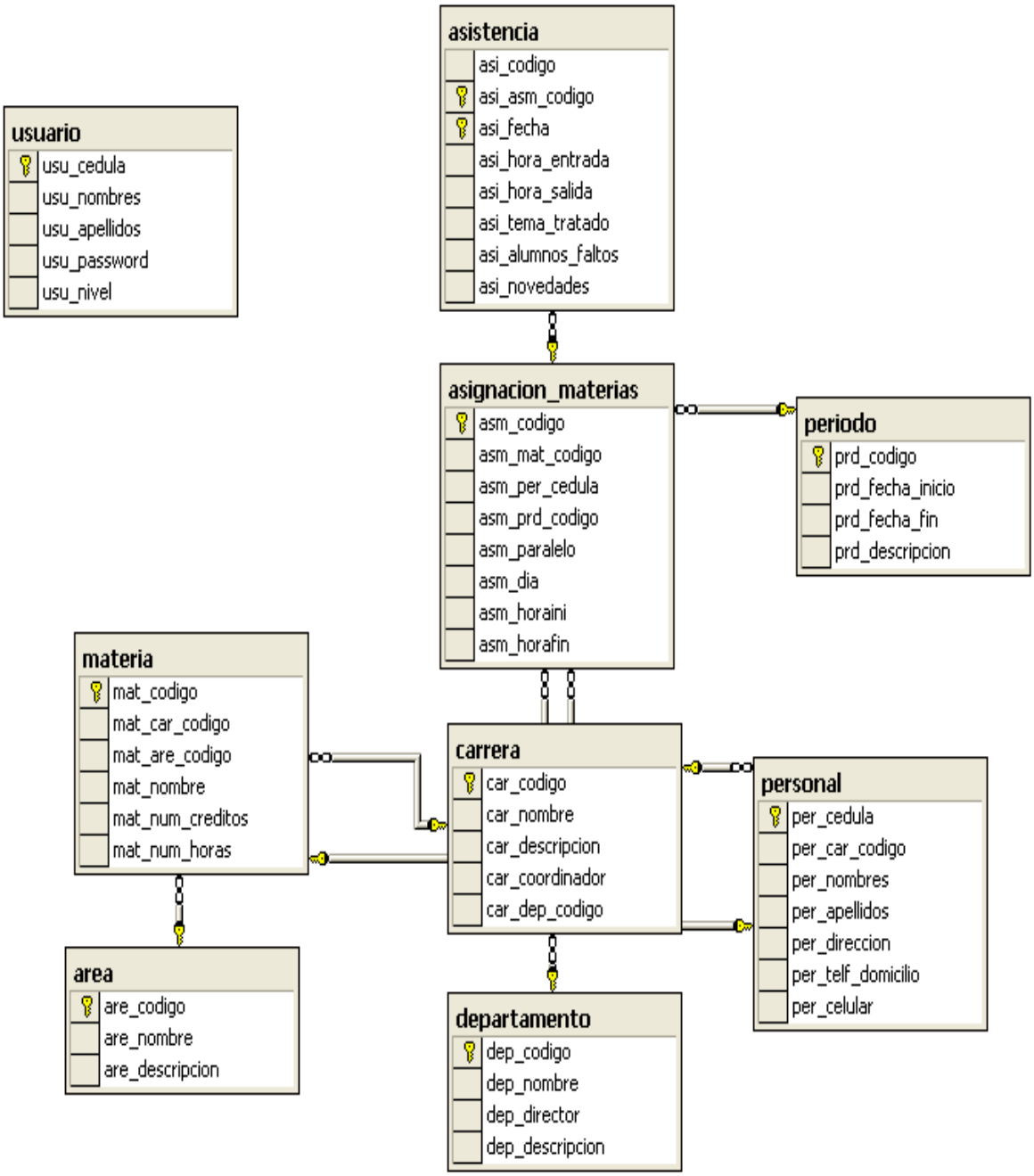

Figura 4.61. Diagrama de la Base de Datos

4.9 DISEÑO DE VISTAS

En esta fase se realiza el diseño de todas las interfaces que el usuario va a manejar, se generan las expectativas y necesidades que tiene el usuario en cada escenario para realizar las funciones del sistema y la secuencia en que suceden dichos escenarios. Se consideran escenarios las ventanas, las páginas Web, los diálogos, etc.

Una de las características que hace tan popular a la tecnología Web es su facilidad para mostrar contenidos de manera gráfica y para vincular de manera fácil documentos de diferentes orígenes.

Para ello, es de suma importancia que los sitios que se construyan cumplan efectivamente con ciertas características de publicación que permitan conseguir dos objetivos muy concretos:

- Que las páginas se desplieguen rápidamente y sin dificultades técnicas en los computadores de los usuarios;

- Que las páginas puedan ser visualizadas por los usuarios de la misma manera en que sus autores las han construido.

4.8.1.- Diagramación de las Páginas

4.8.1.1.- Normas para Incorporar Elementos Gráficos y Multimediales

Cuando en un Sitio Web se incorporan elementos gráficos y multimediales, se deben seguir normas muy concretas para evitar que su peso afecte el desempeño de la página cuando sea solicitada por los usuarios del Sitio Web.

Inicio de Sesión

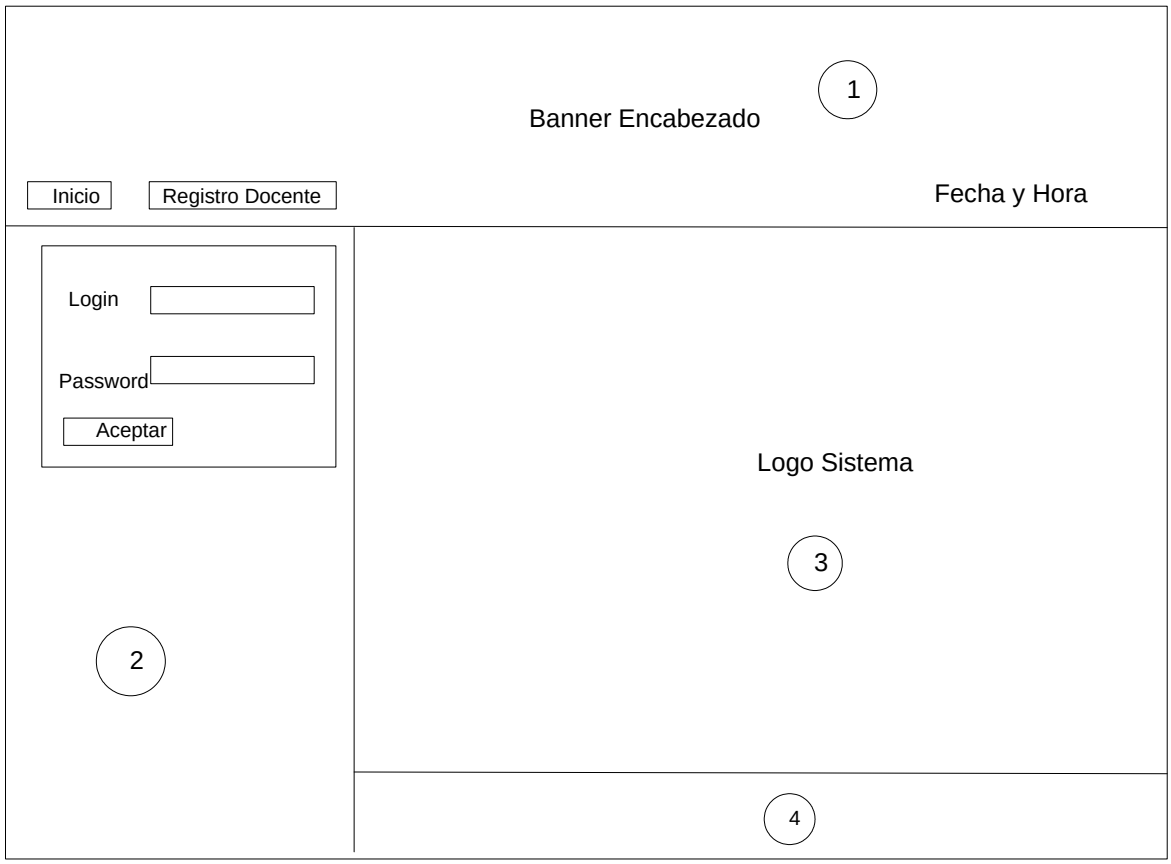

Figura 4.62. Diagrama de Inicio de Sesión

La diagramación de las páginas Web se encuentra fragmentada en varias secciones.

De esa manera, cuando el sitio se presente en el programa visualizador del usuario, siempre mostrará el inicio de sesión. (que normalmente llevará el logotipo y la identificación del sitio) de manera rápida, dando al usuario la sensación de haber llegado al destino elegido.

En la figura 4.62 se puede ver que el sitio está construido en secciones acorde al siguiente orden:

Sección 1: Muestra un banner de la institución.
Sección 2: Muestra la opción para el ingreso al sistema y registro.
Sección 3: Muestra un fondo.

Sección 4: Muestra el pie de la página con la identificación corporativa de la institución.

Paginas de Ingreso de Datos

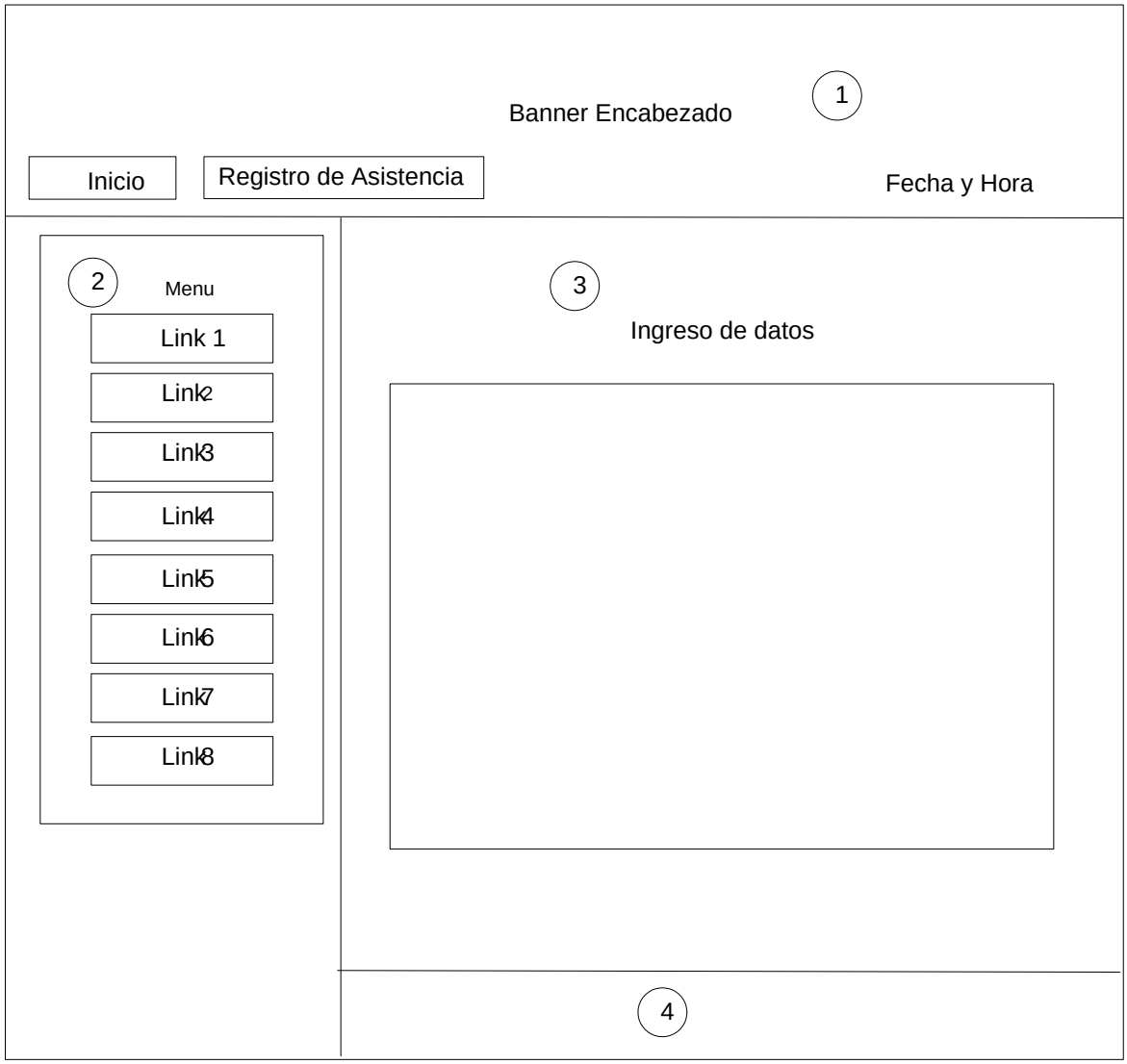

Figura 4.63. Diagrama de Páginas de Ingreso de datos.

En la figura 4.63 se puede ver que el sitio está construido en secciones acorde al siguiente orden:

Sección 1: Muestra un banner de la institución.

Sección 2: Muestra las diferentes opciones del Sistema.

Sección 3: Muestra los diferentes formularios de las opciones del sistema.

Sección 4: Muestra un fondo.

4.9.- DISEÑO DE LA NAVEGACIÓN

El diseño de navegación se refiere a como van interactuando las páginas, el seguimiento y enlaces que tiene una página con otra.

Es importante un diseño bonito, ágil y atractivo. La estructura debe ser coherente y fácil de navegar. La información que presenta debe de ser sencilla y clara. La navegación se mantendrá siempre a la izquierda.

La estructura hipertextual es una estructura bastante compleja que integra en sí misma varios tipos distintos de organización de la información: secuencial, jerárquica y en red, en nuestro sistema se ha aplicado la distribución secuencial.

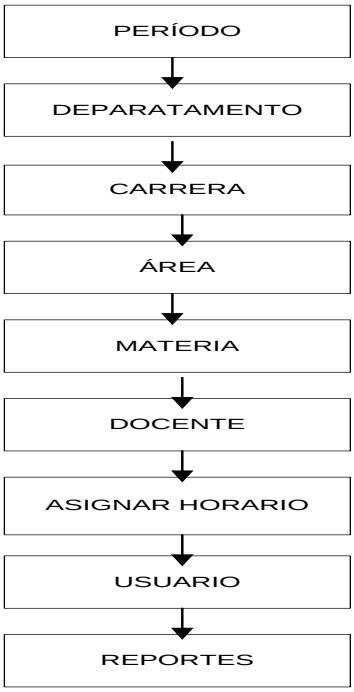

Figura 4.64. Diagrama de Navegación

CAPITULO V

5.- ESTUDIO COMPARATIVO DE LAS TECNOLOGIAS:

5.1.- INTRODUCCIÓN

El desarrollo de aplicaciones empresariales, orientadas al Web ha sufrido un auge muy importante durante los últimos años. Frente a esta nueva demanda surgen dos plataformas distintas para el desarrollo de este tipo de aplicaciones: J2EE de Sun y .NET de Microsoft.

Este documento podría ser útil a personas quienes desean desarrollar una aplicación utilizando unas de estas tecnologías, brindándoles una apreciación global de las características de las plataformas motivo de nuestro estudio.

A continuación en este documento se realizará tablas de resumen basados en lo descrito en los capítulos II y III tomando como referencia 4 componentes principales siendo los siguientes:

- Java CORE API vs .Net framework
- Java Virtual Machine vs Commom Language Runtime de .Net
- JSP y Servlets de J2EE vs ASP de . Net.
- Tecnologías de acceso a base de datos: JDBC vs ADO.NET
- Semejanzas entre las dos plataformas.
- Diferencias entre las dos plataformas.

También se realizara un análisis de la parte técnica como Fiabilidad, Portabilidad, Rendimiento, Seguridad, Costos acorde a lo investigado como también a la experiencia adquirida en el desarrollo de nuestra aplicación.

5.2.- JAVA CORE API VS .NET FRAMEWORK

	JAVA CORE API	**.NET FRAMEWORK**
PLATAFORMAS	Cualquier plataforma que ofrezca soporte para Java (Solaris, Linux, Windows Server, Mac OS	Microsoft Windows 9x, Server, Professional, XP o Server.
PLATAFORMA DE EJECUCIÓN	JVM	CLR
TRABAJO CON DISPOSITIVOS INALÁMBRICOS	SI	SI
TIPO DE LENGUAJE	Java	Lenguaje neutral que permite múltiples lenguajes
INTEGRACIÓN CON COMPONENTES DE VERSIONES ANTERIORES (ACCESORIOS)	SI	SI
SEGURIDAD DEL CÓDIGO EJECUTAD	SI	SI Garantizado por el CLR
HERRAMIENTAS DE DESARROLLO	Se incluye esta arquitectura en diferentes herramientas	Microsoft .NET Visual Studio. Es una herramienta completa

TABLA 5.1 Comparativa Componentes Comunes

5.3 .- JAVA VIRTUAL MACHINE VS COMMOM LANGUAGE RUNTIME DE .NET

	JAVA CORE API	**.NET FRAMEWORK**

PLATAFORMAS	Cualquier plataforma que ofrezca soporte para Java (Solaris, Linux, Windows Server, Mac OS	Microsoft Windows 9x, Server, Professional, XP o Server.
RENDIMIENTO	Bueno	Bueno.
LENGUAJE SCRIPTING	Soporta únicamente al lenguaje Java, .SCRIPT	Soporta cualquier lenguaje que pueda ser representado en el Common Intermedíate Lenguaje (CIL), a demás la CLR posibilita esta integración.
INTEROPERATIVIDAD ENTRE LENGUAJES	Esto no es necesario porque maneja un único lenguaje	Mediante CLS.
MOTOR DE EJECUCIÓN DE APLICACIONES	Java Core API	Net Framework
EJECUCIÓN	Ejecuta Bytecodes	Ejecuta el Lenguaje Intermedio CIL.
INTERPRETACIÓN	Los bytecooes son interpretados.	El CIL nunca se interpreta, es un lenguaje que se compila.
GESTION DE CÓDIGO	SI	SI MSIL se encarga de la carga y ejecución del código.
AISLAMIENTO DE MEMORIAS DE APLICACIÒN	NO	SI
CONVERSIÓN DE CODIGOS CON TÉCNICAS COMPILACIÓN	JIT	JIT
GESTION AUTOMÁTICA DE MEMORIA	SI	SI
SEGURIDAD DE LOS ACCESOS DEL CODIGO A LOS RECURSOS	En la mayoría de los casos	SI
MANEJO DE EXCEPCIONES	SI Una vez el programa correcto crea *.class	SI Incluyendo las excepciones entre código escrito en los diferentes lenguajes.
HERRAMIENTAS DE DESARROLLO	Se incluye esta arquitectura en diferentes herramientas.	Microsoft .NET Visual Studio
SERVICIOS PARA DESARROLLADORES	SI	SI

		Presenta un gran despliegue de información y servicios en relación a cualquier otra herramienta.

TABLA 5.2 Comparativa Intérpretes

5.4.- JSP Y SERVLETS DE J2EE VS ASP DE. NET

	Java		**.Net**
	Jsp	**Servlets**	
PLATAFORMA	Cualquier plataforma que ofrezca soporte para Java (Solaris, Linux, Windows Server, Mac OS, y variantes de UNIX).		Microsoft Windows Server, Proffesional, XP o Server.
SERVIDOR WEB	Cualquiera, los más populares son TonKat , Netscape, US.		Solo trabaja con servidores Microsoft como: Internet Information Server
LENGUAJE SCRIPTING	Si	Si	Si
Componentes	Java Beans, Enterprise		
ETIQUETAS ESPECIALES	**SI**		**NO**
INTEGRACION DE BASES DE DATOS	Base de datos. Cualquiera que soporte tecnología JDBC u ODBC.		Base de datos. Cualquiera que soporte tecnología ODBC.

TABLA 5.3 Comparativo de Herramientas de Desarrollo

5.5.- TECNOLOGÍAS DE ACCESO A BASE DE DATOS: JDBC VS ADO.NET

	JDBC	**ADO.NET**
PLATAFORMAS	Cualquier plataforma que ofrezca soporte para Java (Solaris, Linux, Windows Server, Mac OS	Microsoft Windows 9x, Server, Professional, XP o Server.
VELOCIDAD DE PROCESAMIENTO	Rápido.	Rápido.
COMPLEJIDAD DE TRABAJO CON CONJUNTOS DE DATOS DESCONECTADOS	DIFICIL	FACIL
FORMATO DE TRANSMISIÓN DE DATOS	SQL	XML
TIPOS DE ACCESO	DIFICIL	FACIL
FUNCIONAMIENTO IDÉNTICO PARA TODOS LOS DBMS	No Al ser J2EE una plataforma abierta.	Si SQL Server garantizado.
REFERENCIA	Con trotador. Dependiendo de la empresa propietaria del DBMS.	Tecnología única
HERRAMIENTAS DE DESARROLLO	Varias opciones	Microsoft .NET Visual Studio

TABLA 5.4 Comparativo de Tecnologías de acceso a datos.

5.6.- SEMEJANZAS ENTRE LAS DOS PLATAFORMAS

Acorde a lo descrito en los capítulos anteriores en forma resumida las similitudes tanto tecnológicas, como de servicios entre Microsoft .NET y J2EE podemos citar que todos los conceptos aparecen en ambas arquitecturas, un lenguaje de programación que unido a una librería de clases, compila a un código intermedio independiente de la máquina ("Intermediate Language" en el caso de Microsoft .NET y "Bytecodes" en el caso de Java). Este código se ejecutará en máquina virtual, que proporciona un "entorno de ejecución" que transformará el lenguaje intermedio a código propio de la máquina en la que se corre la aplicación ("Common Language Runtime (CLR)" en Microsoft .NET y "Java Runtime Environment (JRE)" en J2EE). Las dos plataformas disponen sin números de servicios que facilitan la labor del programador.

Las plataformas J2EE y .Net son muy similares entre sí, manteniendo bastantes puntos en común en J2EE el modelo de componentes viene dado por los JavaBeans y Enterprise JavaBeans, los cuales poseen los conceptos de propiedades, eventos que en el .NET aparecen incluidos en el sistema de tipos común, y por lo tanto, son soportados por los distintos lenguajes. Ambos modelos también poseen sistemas de acceso remotos (RMI en J2EE y el framework remoto en .NET), siendo el framework de .NET algo más potente que RMI, además de poseer una capacidad de extensibilidad mayor.

Con respecto al soporte para el desarrollo de clientes J2EE posee a los servlets y a las páginas JSP, mientras que .NET posee a ASP.NET. La comparación de ambas tecnologías inclina la balanza de forma clara sobre ASP.NET, pues aparte de que en ASP.NET es posible utilizar cualquier lenguaje y de que obtiene un rendimiento muy superior a los Servlets y JSPs, ASP.NET introduce un modelo de desarrollo basado en componentes similar al existente en el desarrollo de las interfaces de aplicaciones de escritorio que permite abstraerse del navegador cliente, lo que indica una potencia superior a los serviets y JSPs, desde el punto de vista de programador podemos indicar que net presenta mayores facilidades para el desarrollo de aplicaciones.

5.7.- DIFERENCIAS ENTRE LAS DOS PLATAFORMAS

Como se ha podido observar en este capítulo, por cada tecnología de da plataforma, existen grandes diferencias; es importante describirlas de acuerdo los parámetros ya establecidos, pero ahora tomando en cuenta a toda la plataforma en general.

Característica	**.NET**	**J2EE**
Tipo de tecnología	Producto	Estándar
Empresas que lo ofrecen	Microsoft	Más de 30
Librerías de desarrollo	.NET Framework	Java core API
Intérprete	CLR	JRE
Páginas dinámicas	ASP.NET	Servlets, JSP
Componentes	.NET Manager Components	EJB
Acceso a bases de datos	ADO.NET	JDBC, SQL/J
Servicios Web	SOAP, WDSL, UDDI	SOAP, WDSL, UDDI
Interfaces gráficas	Win Forms y Web Forms	Java Swing
Herramienta de programación	Visual Studio .NET	Dependiente del fabricante
Librería de encolado de mensajes	MSMQ	JMS 1.0
Lenguajes utilizados	C#, Visual Basic, C++, otros	Java
Lenguaje intermedio	IL	Bytecodes

TABLA 5.5 Diferencias entre las Tecnologías .Net y J2EE

5. 8.- FIABILIDAD

Es la probabilidad de que un sistema se comporte tal y como se espera de él, dentro de los parámetros que la integran están: confidencialidad, integridad y disponibilidad.

La escalabilidad es la capacidad de un sistema de incrementar sus prestaciones en función del número de usuarios simultáneos que lo utilizan. Tanto J2EE como .NET ofrecen métodos de escalabilidad como la carga balanceada que permite a un clúster de servidores colaborar y dar un servicio de forma simultánea.

La ventaja de usar J2EE respecto a .NET en la escalabilidad es debida a que existe hardware disponible más potente en el entorno Linux que en el entorno Windows, por lo que es necesario un menor número de máquinas para ofrecer el mismo rendimiento en las dos plataformas.

Por otra parte, Roger Sessions de objectwatch.com, remarca que la plataforma .NET puede escalar desde 16.000 transacciones por minuto a más de 500.000 transacciones por minuto, mientras que IBM WebSphere usando tecnología J2EE/Linux no puede conseguir nada mejor que pasar de 17.000 a 110.000 transacciones por minuto, con un coste monetario mucho mayor por transacción. Por lo tanto con .NET obtendríamos mayor posibilidad de escalado a un mejor precio.

a) ***Interoperabilidad.-*** Característica dada tanto por la plataforma como por el software que trabaja u opera en ella. Interoperabilidad es trabajo conjunto, para obtener sistemas de información integrados y ágiles.

El Framework de .Net es una infraestructura sobre la que se reúne todo un conjunto de lenguajes, servicios tecnologías necesarias para realizar aplicaciones que pueden comunicarse fácilmente en el Internet (o sobre cualquier red como intranet) usando estándares abiertos como el XML y SOAP. Mediante esta herramienta se ofrece un entorno de ejecución altamente distribuido, que permite crear aplicaciones robustas y escalables, el Framework de .Net es una plataforma no incluida en los diferentes sistemas operativos distribuidos por Microsoft, por lo que es necesaria su instalación previa a la ejecución de programas creados mediante .Net. El Framework se puede descargar gratuitamente desde la Web oficial de Microsoft, en el desarrollo de nuestra aplicación se utiliza Framework 1.1 que viene incorporado con el Visual Estudio 2003 que es la herramienta con la cual desarrollamos nuestra aplicación.

En Java se utiliza un (“kit”) llamado JSDK que es utilizado para ejecutar (“Runtime”) que sirve para mostrar la interfaz de nuestra aplicación. En la versión que se esta utilizando ya incluye un JRE que es la versión 5.0 es el compilador donde se generan todos los programas, interfaces y elementos programados en java, inclusive a partir de estas clases se puede definir otras clases específicas las mismas que deben ser compiladas para crear el denominado byte-code quién interpreta al JRE y ofrece la interoperabilidad de Java o el afamado “Write once run everywhere” = “Escríbelo una vez ejecútalo en todos lados”.

b) *Escalabilidad.-* Particularidad de un sistema de incrementar sus prestaciones en función del número de usuarios simultáneos que lo utilizan.

Descripción dada para un sistema, tecnología, plataforma, etc. que posee la propiedad de poder escalar horizontal y verticalmente según las necesidades (número de usuarios).

c) *Facilidad de desarrollo y manejo.-* Las plataformas de desarrollo de aplicaciones de empresa deben proporcionar facilidades para mejorar la productividad en dicho desarrollo, posibilitando el ahorro en tiempo y costes. En esta evaluación global principalmente se tomará el *Soporte Framework* como base se puede notar que por medio de este se puede realizar de mejor manera y rapidez ya tiene incorporados algunas opciones que permiten con tan solo dar un clic diseñar ventanas conjuntamente con mecanismo de métodos y eventos para controlar el código, lo cual demuestra que es un entorno sólido para el desarrollo, caso contrario sucede con J2EE que todo se debe desarrollar mediante código, para el desarrollo del interfaz se tuvo que utilizar otra herramienta como es Dreamweaver.

d) *Integración.-* Propiedad de una aplicación, tecnología o plataforma de poder integrar con facilidad software externo.

.Net nos brinda mejor integración ya que pertenece a una sola familia caso contrario sucede con J2EE que no todas las aplicaciones no se pueden integrar ya que depende del proveedor en el que fue desarrollado.

Las dos plataformas nos brindan una buena confidencialidad permitiendo acceder al sistema por elementos autorizados a ello, controlando que los objetos pueden ser modificados por elementos autorizados.

5.9.- PORTABILIDAD

Característica propia de componentes o aplicaciones que les permite ejecutarse en múltiples plataformas.

Microsoft continua con su voluntad de apoyar a Windows, por lo que, por supuesto Microsoft .NET funciona únicamente en plataformas basadas en Win32.

Por otro lado, es conocido el lema de Java "escríbelo una vez, ejecútalo en cualquier parte", que hace referencia a que este lenguaje es independiente de la plataforma hardware o sistema operativo utilizado. Esta portabilidad ocurre gracias a que el entorno de ejecución de java (JRE) existe para varios sistemas operativos y plataformas.

A pesar de la gran portabilidad que se le supone inicialmente a J2EE, existe el problema de que J2EE es un estándar y no un producto en sí. Este hecho, que facilita la adopción de esta tecnología por parte de varios fabricantes, también conlleva que las implementaciones de J2EE no son 100% compatibles entre sí, ya que cada vendedor ha realizado su propia interpretación del estándar y ha añadido nuevas características que no tienen por qué incluir el resto de competidores. Lo que sí es cierto, es que todas las empresas que ofrecen sus

productos basados en J2EE tienen versiones para los distintos sistemas operativos, por lo que una misma aplicación será portable entre los distintos sistemas operativos siempre y cuando mantengamos la solución del mismo vendedor. En definitiva, pasar de una implementación J2EE a otra requerirá de modificaciones en el código de la aplicación y la portabilidad se pierde en gran parte.

Chad Vawter and Ed Roman, en su documento "J2EE vs Microsoft .NET", realiza un análisis de cuando es importante la portabilidad al elegir entre una plataforma u otra. Consideran tres escenarios para la decisión de cual es la arquitectura correcta:

- "Si se desarrolla software para otros negocios, o si es una compañía consultora, y los usuarios tienen una gran variedad de plataformas, recomiendan especializarse en la arquitectura J2EE. Si no puedes garantizar que tus clientes aceptarán Windows como solución, estarás restringiendo a los vendedores de grandes cuentas que seguramente han desarrollado sus soluciones en UNIX o Mainframes.

- Si los clientes están en la plataforma Windows, en ese caso puede servir tanto J2EE como .NET ya que ambas soluciones se ejecutan en Windows.

- Si alojas tus propias soluciones, entonces controlas el entorno de implantación. Esto te permite escoger libremente entre J2EE y .NET.

Aunque el primer punto y el tercero coincido totalmente con su opinión, su valoración del segundo escenario no me parece correcta ya que en un entorno completamente basado en Windows, la solución de Microsoft .NET siempre funcionará mejor que la basada en J2EE, porque tanto Windows como .NET son productos de Microsoft, lo que asegura una mayor integración.

5.10.- RENDIMIENTO

Las dos ofrecen las herramientas para trabajar sistemas completos.

El rendimiento es uno de los temas más controvertidos a la hora de comparar estas dos plataformas porque cuando independientemente de los resultados que muestren unos u otros defendiendo una plataforma, la otra parte nunca parece dispuesta a aceptarlos alegando disparidad de criterios como el hardware empleado, distintos tipos de optimizaciones utilizados, etc.

La idea de tener una máquina virtual no es en absoluto nueva, estando presente en multitud de sistemas desde muchos años atrás, principalmente en Java (Java Virtual Maquine). Lo destacable de la máquina virtual de .NET es que esta ha sido diseñada con el rendimiento como uno de sus principales objetivos, y parece que lo ha conseguido, pues el rendimiento de dicha máquina es superior a la de otros sistemas como Java en lo que a gestión de memoria se refiere, pero la ventaja de Java es que es multiplataforma.

Windows Forms tiene mayor rendimiento que la tecnología Java Swing principalmente en lo que se refiere a tiempos de respuesta, Swing tiene cierto tiempo de retardo al dibujarse cada vez que se llama a una de sus funciones.

ASP.NET es una tecnología que obtiene un rendimiento muy superior a los serviets y JSPs. ASP.NET introduce un modelo de desarrollo basado en componentes similar al existente en el desarrollo de las interfaces de aplicaciones de escritorio que permite abstraerse del navegador cliente, lo que lo dota de una potencia muy superior a los servlets y JSPs.

Las referencias publicado por Microsoft sobre la implementación en .Net de una aplicación de referencia de Java para el desarrollo de aplicaciones de comercio electrónico: "Java Pet Store", se discuten aspectos de la prueba realizada donde

se ve que los resultados del rendimiento conseguido en la plataforma .Net es mayor que al que se puede obtener en J2EE.

En las siguientes gráficas se muestran los resultados de la prueba en los que se observa la clara superioridad de Microsoft.NET

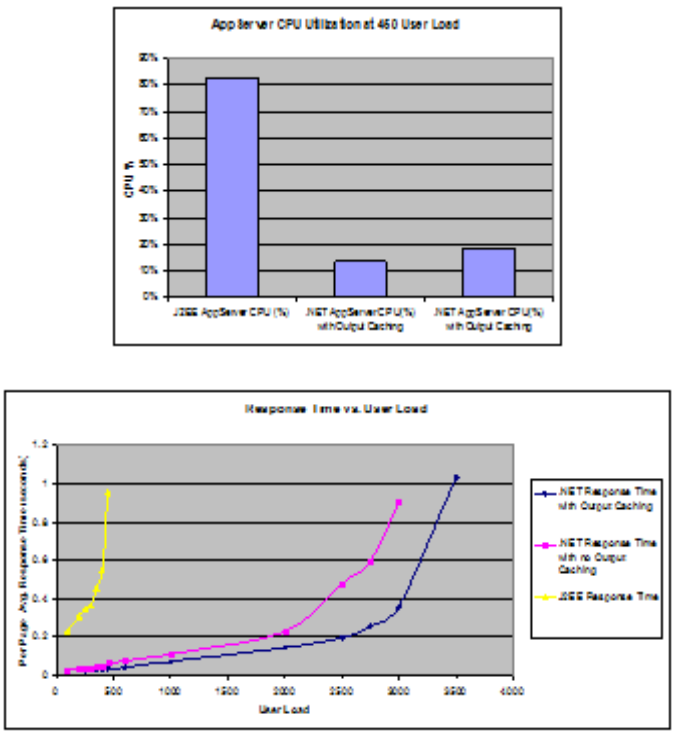

Figura 5.65. Análisis De Rendimiento

5.11.- SEGURIDAD

Son las características de cualquier sistema, tecnología o plataforma (informática o no) que indica que se encuentra libre de todo peligro, daño o riesgo, y que es, en cierta manera, infalible. En muy pocos casos se puede hablar de sistemas, tecnologías o plataformas seguras por lo que para suavizar el término y todo lo que él implica se lo tratará como fiabilidad.

Con respecto al sistema de seguridad de .NET, este asume la realidad de la incivilidad del código, proporcionando un sistema extensible y de gran granularidad que reduce el riesgo asociado; para J2EE es diferente, el estándar multiplataforma cuida los detalles en cuanto a su desempeño sobre las diferentes plataformas siendo poco extensible. La seguridad es un tema que dentro de las dos tecnologías para los administradores juegan un papel importantísimo, son los encargados de crear una política de seguridad robusta a todos los niveles, permitiendo a los desarrolladores abstraerse de estas cuestiones, aunque estos también pueden utilizar extender el modelo de seguridad, claro que en distintas proporciones.

5.12.- COSTOS

Para la arquitectura J2EE podemos encontrar un amplio rango de soluciones disponibles a precios que rondan desde lo prácticamente gratuito a lo realmente caro, en función de solución escogida. Hay que tener en cuenta que las soluciones gratuitas o baratas no incluyen algunos servicios realmente útiles.

Por lo general, la plataforma de Windows suele ser más barata que UNIX, tanto en precio del hardware como en el de las licencias de software necesarias ya que Microsoft siempre ofrece un precio muy competitivo.

Además de los costes debido al hardware y al de las licencias habría que tener en cuenta el de aplicaciones adicionales como bases de datos, cortafuegos, etc. No las tenemos en cuenta en este estudio debido a la posibilidad de las dos plataformas de trabajar con productos de distintas empresas.

Acorde lo analizado y descrito en los párrafos anteriores la plataforma .NET proporciona una serie de beneficios al desarrollo le las aplicaciones. Por una parte, acelera el desarrollo de las aplicaciones ágilmente, pues el sistema gestiona automáticamente una serie de aspectos como la gestión de la memoria, a la vez que proporciona un conjunto de librerías que permiten una mayor reusabilidad, mejorando de esta manera el rendimiento, Por otra parte, posibilita

el desarrollo multilenguaje y multiplataforma (en un futuro), con un nivel de integración entre lenguajes no existente en ningún otro sistema, pudiendo heredar entre sí clases escritas en diferentes lenguajes.

Una de las características más destacables sobre la integración de distintos lenguajes son las enormes posibilidades de reusabilidad, pues no solo están disponibles las librerías de clases base de .NET, sino que cada uno de los diferentes lenguajes que se adapten a .NET también adaptaran sus propias librerías.

CAPITULO VI

6.1.- CONCLUSIONES

- La plataforma J2EE es una plataforma independiente del hardware se aloja en el sistema mientras exista el elemento conocido como Máquina Virtual de Java (MVJ) quien hace posible que una aplicación desarrollada en Java se ejecute en cualquier sistema operativo que soporte esta máquina virtual, como por ejemplo Unix, Linux, Macintosh, Windows, Solaris, entre otros, que utiliza únicamente el lenguaje Java, multiplataforma y con múltiples proveedores. En cuanto a las herramientas de desarrollo existente para esta plataforma una amplia gama.

- La plataforma .NET es multilenguaje lo que facilita el desarrollo y la integración, permitiendo elegir al programador el lenguaje de programación que más domine, por defecto lleva integrado C#, VB.NET y J#, pero podría usar otro lenguaje, diseñada para una única plataforma y con un único proveedor, Microsoft, Así mismo se puede elegir desde el Notepad, hasta la sofisticada y potente Visual Studio .Net para el desarrollo de aplicaciones a todo estos aspectos importantes también se incorpora una rica biblioteca de clases.

- Mediante la utilización de estas tecnologías es posible desarrollar aplicaciones Web que permitan generar contenido dinámico y que además permitan al usuario administrar y controlar dicho contenido de una forma fácil e intuitiva, lo que hace que el desarrollo de aplicaciones de este tipo vaya ganando espacio día tras día en las diferentes instituciones y empresas.

- La utilización de la metodología de desarrollo orientada a objetos y de un lenguaje de modelado como UML (Lenguaje de Modelado Unificado) en el

Análisis y Diseño, fue de gran ayuda ya que permitió visualizar claramente todos los elementos y componentes que intervienen en el sistema.

- El proyecto ha sido concluido en todas las fases cumpliendo con los requerimientos de información propuestos, El sistema permite acceder a toda la información de la Asistencia de los Docentes, con una interfaz en la cual se utiliza el teclado y mouse permitiendo al usuario una navegación más rápida y simple.

- En el desarrollo de la aplicación se ha adquirido una experiencia positiva al trabajar con herramientas libres (Netbeans IDE 5.5, Tomcat, Mysql) para java y .net (Visual Studio. Net 2003, SqlServer) herramientas propietarias en los dos casos han facilitado el desarrollo y la implementación.

- Nos encontramos ante dos plataformas excelentes para el desarrollo de aplicaciones Web, empresariales y de servicios Web. Es importante citar que no se puede establecer cual es la mejor plataforma, esto estará dada acorde al tipo de aplicación a implementar por tal motivo el desarrollo de este proyecto servirá como fuente de consulta para el desarrollador que es quién decide las características del entorno de la Solución Software.

6.2.- RECOMENDACIONES

- Todas las instituciones deberían pensar en la sistematización de procesos manuales a automatizados los mismos que permitirían tener control y mantenerse a la vanguardia de los avances tecnológicos permitiendo ser más competitivos.

- Antes de desarrollar un sistema se debe tener un conocimiento tanto del software y hardware de la institución o empresa, como también de la plataforma para el desarrollo del proyecto y el adecuado manejador de base de datos que sea compatible con la misma para evitar pérdida de tiempo y recursos.

- Se recomienda obtener y elaborar bien los requerimientos para poder desarrollar de buena manera el producto.

- Se recomienda tomar esta aplicación como base y desarrollar nuevas características que le permitan tener mayor funcionalidad e interactuar con otras aplicaciones existentes dentro de la institución.

- Se recomienda que se revise los puntos de red ya que el sistema funcionará en la Intranet Institucional, como también el servidor donde se encuentre alojado el sistema.

- Recomiendo no dejarse llevar por los costos que cada una de estas tecnologías necesitan para su implementación si no también un factor muy importante como es la disminución del tiempo de desarrollo y la disponibilidad de recursos profesionales.

- A los estudiantes de la carrera recomendaría seguir con las investigaciones en este campo ya que es muy dinámico y por el continuo cambio de las tecnologías, como también el auge continuo de aplicaciones Web será una de las actividades de los Ingenieros de Sistemas Informáticos.

ÍNDICE

INDICE DE FIGURAS

INDICE DE TABLAS

ANEXOS

Los mismos se adjuntarán en CD- ROM siendo los siguientes:

Anexo I	Casos de uso realizado con la herramienta Rational Rouse Enterprise Edition
Anexo II	Código Fuente Programación en la Tecnología .Net
Anexo III	Código Fuente Programación en l Tecnología Java
Anexo IV	Manual de usuario del Sistema SICADOC en . Net
Anexo V	Manual de usuario del Sistema SICADOC en Java.