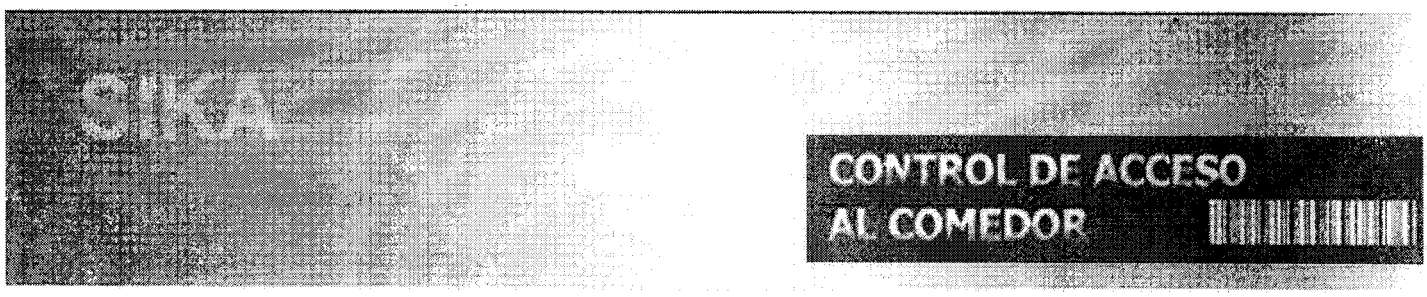
Universidad de las Ciencias Informáticas

DIRECCIÓN DE INFORMATIZACIÓN

Sistema Informático área de Alimentación.



TRABAJO DE DIPLOMA PARA OPTAR POR EL

Título de Ingenierio Informático

Autor: Sergio Barrios Díaz

Tutor: Ing. Yosvany Umpierre

Ciudad de la habana, CUBA
diciembre de 2005

Resumen.

En los momentos en que vivimos, las herramientas y la conexión de la era digital ponen a nuestra disposición medios para obtener información, compartirla, y actuar en función de ella de muchas maneras diferentes y notables. Para poder tener aceptación en este mundo hay que desarrollar una nueva infraestructura digital, tanto mental como técnica, de tal forma que todos los elementos se relacionen de manera armónica y cohesionada, con capacidad para funcionar de manera fluida y eficiente.

Este presente trabajo está enmarcado dentro del área de alimentación, Sistema Informático Área de Alimentación "UCI Ciudad Digital" y se encarga de llevar a cabo la etapa de análisis y diseño de una aplicación Web que automatizará el acceso al comedor así como el estudio de nuevas tecnologías de control de acceso al mismo (Sistema Informático Área de Alimentación).

En los comedores se cuenta con una metodología que lleva a cabo el acceso al mismo, pero no es lo que se quiere lograr en realidad, lo que de desea alcanzar es un sistema automatizado que abarque toda las necesidades de dicha área, optimizando todos los procesos para un mejor funcionamiento del mismo, el acceso al comedor se realiza prácticamente de forma manual obstruyendo y atrasando los proceso que ocurren en el mismo. Todo esto origina una gran masa de datos almacenada de forma manual, difícil de procesar y aun con manejo eficiente de la información a la hora de hacer búsquedas u obtener reportes que permitan hacer un análisis del proceso habrá perdida de tiempo y esfuerzo, obteniendo en casos duplicidad de información, entre otros problemas, por tanto esto no es suficiente para resolver los problemas que suceden, se hace necesario la puesta en marcha de un sistema , acorde a las tendencias del desarrollo de la informática actual y que permita llevar toda la información como el control de acceso al comedor de una forma unida e integrada entre si.

Se desea que esta aplicación logre una interrelación con otros sistemas para el intercambio de información, a través de servicios Web que permitan evitar

duplicidad innecesaria y contar con información de primera mano brindando así a los usuarios acceso a la información de manera oportuna sin interrupciones, obteniendo informes que faciliten la toma de decisiones..

Para el cumplimiento de esta etapa se utilizará el Proceso Unificado de Rationall (RUP) como metodología de construcción de software, la cual resulta muy poderosa por sus características de conducción por casos de uso y orientación a objetos.

Índice.

Indice de figuras.

Capitulo1.....Fundamentación Teórica.

1.1 - Introducción.

El proyecto a realizar se encuentra dentro del área del control de acceso al comedor, y en la Universidad en el proyecto "UCI Ciudad Digital". Este proyecto "CIUDAD DIGITAL" es uno de los primeros surgidos en nuestra universidad que a logrado grandes aportes para la misma, constituye punto de partida para lograr la informatización general de la sociedad cubana. A través de todos estos años en el proyecto "UCI CIUDAD DIGITAL" se han hecho uso de nuevas tecnologías logrando estar actualizados con el ámbito mundial en ese caso.

En este capítulo también se da una breve explicación sobre las características y potencialidades de la plataforma .NET, las potencialidades de los gestores de Bases de Datos utilizados. Haciendo énfasis en los aspectos más importantes de los servicios Web y las bases de datos desarrolladas en SQL Server que son obtenidos a través de un estudio detallado de la bibliografía existente sobre el tema.

A razón de esto se determino realizar una aplicación que permita el control de acceso al comedor en la Universidad pero para esto se deben tener en cuenta varios aspectos que seran referenciados en este capitulo donde se describen de una forma detallada todos los fundamentos teóricos de los procesos que intervienen en la aplicación teniendo en cuenta los momentos actuales, se hace alusión a algunos de los software's existentes tanto en el ambito nacional como internacional donde se analizan pequeñas deficiencias a dichos software's, se da una breve descripción de la metodología a utilizar para el modelo del análisis y diseño del software y de su herramienta de uso.

1.2 - Tecnologías Utilizadas en el Sistema.

1.2.1 - La Tecnología.NET

La revolución de Internet que tuvo lugar a finales de los años 90 supuso un drástico cambio en la forma en la que los individuos y las empresas se comunican entre sí. Las aplicaciones tradicionales como procesadores de texto y programas de contabilidad se conciben como productos independientes, ya que permiten a los usuarios realizar operaciones utilizando para ello datos almacenados en el sistema en el que se encuentran y desde el que se ejecuta la aplicación.

Por el contrario, los nuevos programas se basan en un modelo computacional distributivo en el que las aplicaciones colaboran entre sí para ofrecer sus servicios y compartir sus funciones. Como resultado, el papel principal de los nuevos programas se centra en la compatibilidad de intercambio de información (por medio de servidores y navegadores Web), colaboración (por medio de correo electrónico y mensajes instantáneos) y expresiones individuales (a través de registros Web, conocidos como Blogs y e-zines, revistas basadas en la Web). Básicamente, el nuevo software ha pasado de ofrecer una discreta funcionalidad a ofrecer servicios mucho más completos.

La tecnología .NET representa un conjunto de servicios y unificado y orientado a objetos que engloba el nuevo papel de los programas basados y dirigidos a la red. De hecho, esta estructura es la primera plataforma diseñada esencialmente con Internet como base. En este capítulo presentaremos la tecnología .NET en función de las ventajas que ofrece.

1.2.2 - Ventajas de la tecnología .NET

La estructura .NET ofrece distintas ventajas que pueden ser utilizadas por los programadores:

Un modelo de programación consistente.

Compatibilidad directa con cuestiones de seguridad.

Operaciones simplificadas de desarrollo.

Sencilla implementación y mantenimiento de las aplicaciones.

1.2.3 - Microsoft Framework .NET

NET Framework es un componente integral de Windows que admite la creación y la ejecución de la siguiente generación de aplicaciones y servicios Web XML. El diseño de .NET Framework está enfocado a cumplir los objetivos siguientes:

- Proporcionar un entorno coherente de programación orientada a objetos, en el que el código de los objetos se pueda almacenar y ejecutar de forma local, ejecutar de forma local pero distribuida en Internet o ejecutar de forma remota.
- Proporcionar un entorno de ejecución de código que reduzca lo máximo posible la implementación de software y los conflictos de versiones.
- Ofrecer un entorno de ejecución de código que fomente la ejecución segura del mismo, incluso del creado por terceras personas desconocidas o que no son de plena confianza.
- Proporcionar un entorno de ejecución de código que elimine los problemas de rendimiento de los entornos en los que se utilizan secuencias de comandos o intérpretes de comandos.
- Ofrecer al programador una experiencia coherente entre tipos de aplicaciones muy diferentes, como las basadas en Windows o en el Web.
- Basar toda la comunicación en estándares del sector para asegurar que el código de .NET Framework se puede integrar con otros tipos de código.

.NET Framework contiene dos componentes principales: Common Language Runtime y la biblioteca de clases de .NET Framework. Common Language Runtime es el fundamento de la tecnología. El motor de tiempo de ejecución se puede considerar como un agente que administra el código en tiempo de ejecución y proporciona servicios centrales, como la administración de memoria, la administración de subprocesos y la interacción remota, al tiempo que aplica una seguridad estricta a los tipos y otras formas de especificación del código que fomentan su seguridad y solidez. De hecho, el concepto de administración de código es un principio básico del motor de tiempo de ejecución. El código destinado al motor de tiempo de ejecución se denomina *código administrado*, a diferencia del resto de código, que se conoce como *código no administrado*. La biblioteca de clases, el otro componente principal de .NET Framework, es una completa colección orientada a objetos de tipos reutilizables que se pueden emplear para desarrollar aplicaciones que abarcan desde las tradicionales herramientas de interfaz gráfica de usuario (GUI) o de línea de comandos hasta las aplicaciones basadas en las innovaciones más recientes proporcionadas por ASP.NET, como los formularios Web Forms y los servicios Web XML.

.NET Framework puede alojarse en componentes no administrados que cargan Common Language Runtime en sus procesos e inician la ejecución de código administrado, con lo que se crea un entorno de software en el que se pueden utilizar características administradas y no administradas. En .NET Framework no sólo se ofrecen varios hosts de motor de tiempo de ejecución, sino que también se admite el desarrollo de estos hosts por parte de terceros.

Por ejemplo, ASP.NET aloja el motor de tiempo de ejecución para proporcionar un entorno de servidor escalable para el código administrado. ASP.NET trabaja directamente con el motor de tiempo de ejecución para habilitar aplicaciones de ASP.NET y servicios Web XML, que se tratan más adelante en este tema.

Internet Explorer es un ejemplo de aplicación no administrada que aloja el motor de tiempo de ejecución (en forma de una extensión de tipo MIME). Al usar Internet Explorer para alojar el motor de tiempo de ejecución, puede incrustar componentes administrados o controles de Windows Forms en documentos HTML. Al alojar el motor de tiempo de ejecución de esta manera se hace posible el uso de código móvil administrado (similar a los controles de Microsoft® ActiveX®), pero con mejoras significativas que sólo el código administrado puede ofrecer, como la ejecución con confianza parcial y el almacenamiento aislado de archivos.

En la ilustración siguiente se muestra la relación de Common Language Runtime y la biblioteca de clases con las aplicaciones y el sistema en su conjunto. En la ilustración se representa igualmente cómo funciona el código administrado dentro de una arquitectura mayor.

Explicacion del .NET Framework.

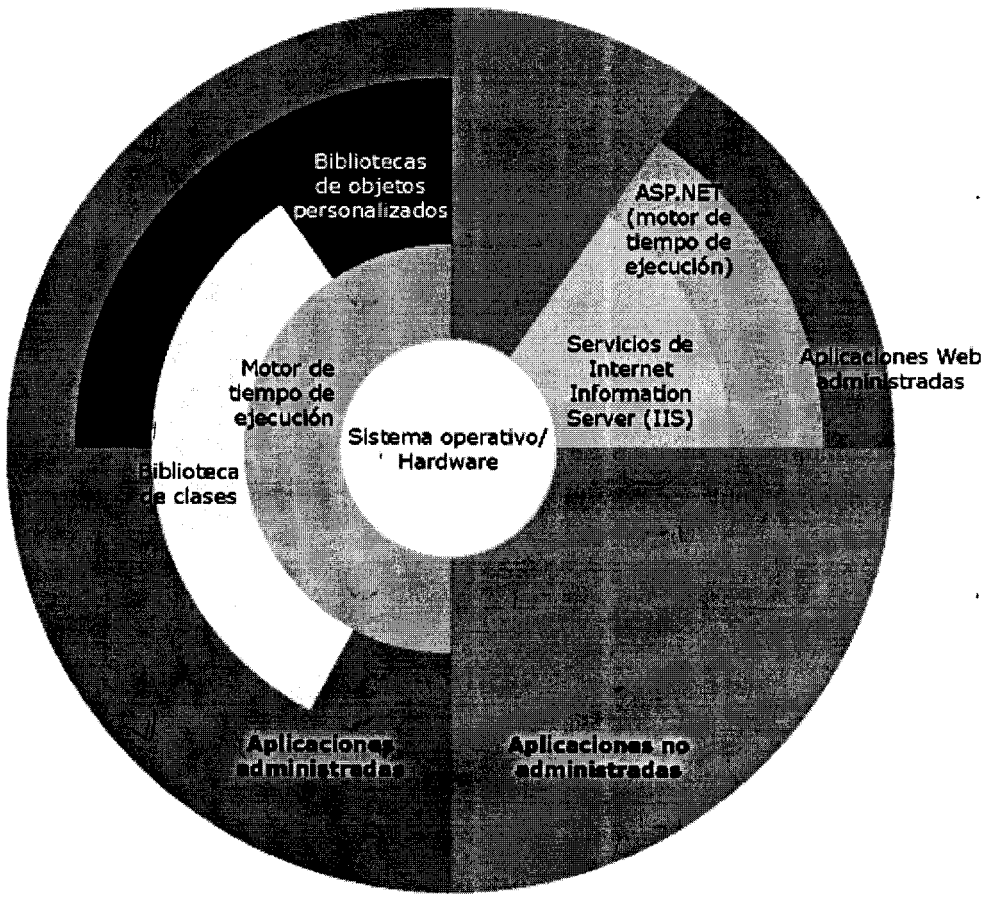


Figura1:Funcionamiento del Framework

1.2.4 - Visual Studio .NET 2003.

Presentamos Visual Studio .NET 2003, la herramienta de segunda generación de Microsoft para crear e implementar software seguro y eficaz para la plataforma Microsoft .NET.

Creado para satisfacer las necesidades de desarrollo de software más exigentes de hoy día, Visual Studio .NET 2003 mejora y optimiza a su predecesor, con el que es altamente compatible. Visual Studio .NET 2003 incluye una completa gama de funciones, desde modeladores que ayudan a componer visualmente las aplicaciones empresariales más complejas hasta la implementación de una aplicación en el más pequeño de los dispositivos. Utilizados por compañías de todos los tamaños en el mundo entero, Visual Studio .NET y la plataforma .NET Framework de Microsoft Windows proporcionan una completa herramienta, eficaz y sofisticada, para diseñar, desarrollar, depurar e implementar aplicaciones seguras para Microsoft Windows® y Web, a la vez sólidas y fáciles de utilizar.

Visual Studio .NET 2003 contiene una versión mejorada de Windows .NET Framework. Windows .NET Framework 1.1 incluye nuevas funciones, mejoras y actualizaciones de la documentación. Gracias a la compatibilidad integrada con .NET Compact Framework, Visual Studio .NET 2003 incorpora a .NET los dispositivos móviles e incrustados, como Pocket PC, y otros dispositivos que utilizan el sistema operativo Microsoft Windows CE .NET. Ahora, los programadores pueden utilizar el mismo modelo de programación, las mismas herramientas para programadores y los mismos conocimientos para crear aplicaciones orientadas tanto a dispositivos pequeños como a los centros de datos de mayor tamaño.

1.2.4.1 - Ventajas Visual Studio .NET 2003.

Los programadores pueden utilizar Visual Studio .NET para:

- Crear aplicaciones basadas en Windows rápidas y eficaces.
- Crear aplicaciones para Pocket PC rápidas y eficaces.
- Crear aplicaciones Web sofisticadas y seguras.

- Crear aplicaciones Web inteligentes, sofisticadas y seguras para dispositivos móviles.
- Utilizar servicios Web XML en cualquiera de las aplicaciones mencionadas.
- Evitar conflictos entre archivos .DLL.
- Eliminar los costosos problemas de implementación y mantenimiento de las aplicaciones.
- Visual Studio .NET es el único entorno de desarrollo creado exclusivamente para permitir la integración con servicios Web XML. Al hacer posible que las aplicaciones compartan datos a través de Internet, los servicios Web XML permiten a los programadores ensamblar aplicaciones a partir de código nuevo y existente, independientemente de la plataforma, el lenguaje de programación o el modelo de objetos.

1.2.5 - Asp . NET.

La creación de aplicaciones Web que respondan rápidamente a las solicitudes de los usuarios, incluso cuando se estén procesando muchas solicitudes en el servidor, ha supuesto un gran reto para los programadores y el personal de tecnología de la información desde los comienzos de Internet. Supervisar el rendimiento de un sitio Web es algo que todos los programadores de Internet e intranets deben ser capaces de hacer. ASP.NET se diseñó teniendo en cuenta este principio.

El modelo de ASP.NET proporciona diversas mejoras de rendimiento integradas que no se suministraban con las versiones anteriores de ASP. En particular, existen dos mejoras con relación al procesamiento de las solicitudes HTTP. En primer lugar, cuando se solicita una página ASP.NET por primera vez, se compila dinámicamente una instancia de la clase Page. (En las versiones anteriores de ASP, el código de página de las solicitudes se interpretaba en el orden en que aparecían en la página.) El compilador JIT (Just-In-Time) de Common Language Runtime compila el código administrado de las páginas ASP.NET para generar código nativo del servidor de procesamiento en tiempo de ejecución. En segundo lugar, una vez compilada la instancia de **Page** para la primera solicitud, se almacena en caché del servidor. Para las siguientes solicitudes de la página se ejecuta la instancia de la clase almacenada en caché. Después de la solicitud inicial, la clase **Page** se vuelve a compilar únicamente cuando se cambia el origen de la página o una de sus dependencias.

Además, ASP.NET almacena en caché los objetos internos, como las variables de servidor, para acelerar el acceso a los mismos por parte del código de usuario. Como parte de .NET Framework, ASP.NET se beneficia de las mejoras de rendimiento que ofrece Common Language Runtime, como la compilación JIT ya mencionada, un Common Language Runtime bien ajustado para equipos de un solo procesador o multiprocesador, etc.

Desafortunadamente, estas mejoras no pueden impedir que se escriba código que provoque problemas de rendimiento cuando la aplicación debe procesar simultáneamente un gran número de solicitudes HTTP. Es necesario probar las aplicaciones para asegurarse de que satisfarán la demanda de los usuarios. Existen cuatro medidas de rendimiento comunes que pueden probarse para garantizar el correcto funcionamiento de la aplicación.

Tiempo de ejecución

Tiempo que tarda en procesarse una solicitud; normalmente se mide entre el primer y el último byte devuelto al cliente desde el servidor. El tiempo de ejecución afecta directamente al cálculo del rendimiento.

Tiempo de respuesta

Tiempo que transcurre entre la emisión de una solicitud y la llegada al cliente del primer byte devuelto por el servidor. A menudo, éste es el aspecto que mejor percibe el usuario cliente. Si una aplicación tarda mucho en responder, el usuario puede impacientarse y dirigirse a otro sitio. El tiempo de respuesta de una aplicación puede variar independientemente de la tasa de rendimiento (o incluso en proporción inversa).

Escalabilidad

Medida de la capacidad de una aplicación para mejorar su rendimiento cuando se le asignan más recursos (memoria, procesadores o equipos). Suele tratarse de una medida de la tasa de cambio de rendimiento con respecto al número de procesadores.

Rendimiento

Número de solicitudes que una aplicación Web puede atender por unidad de tiempo; suele medirse en solicitudes por segundo. El rendimiento puede variar según la carga (número de subprocesos de cliente) que se aplique al servidor. A menudo se considera el factor de rendimiento que es más importante optimizar.

1.3 - Lenguajes de Programación Utilizados

1.3.1 - C#.

C# ("C sharp") es un lenguaje de programación sencillo y moderno, orientado a objetos y con seguridad de tipos. Los programadores de C y C++ se familiarizarán inmediatamente con él. C# combina la gran productividad de los lenguajes de desarrollo rápido de aplicaciones (RAD, *Rapid Application Development*) con la eficacia de C++.

Visual C# .NET es la herramienta de desarrollo en C# de Microsoft. Incluye un entorno de desarrollo interactivo, diseñadores visuales para generar aplicaciones para Windows y Web, un compilador y un depurador. Visual C# .NET forma parte de un conjunto de productos denominado Visual Studio .NET que también incluye Visual Basic .NET, Visual C++ .NET y el lenguaje de secuencias de comandos JScript. Todos estos lenguajes proporcionan acceso a Microsoft .NET Framework, que incluye un motor de ejecución común y una nutrida biblioteca de clases. La plataforma .NET Framework define una CLS (*Common Language Specification*, especificación común de lenguaje), es decir, una especie de "lingua franca" que asegura una interoperabilidad perfecta entre los lenguajes y las bibliotecas de clases compatibles con ella. Para los desarrolladores de C# esto significa que, a pesar de que C# es un lenguaje nuevo, tiene un acceso completo a las mismas ricas bibliotecas de clases que las que utilizan herramientas tan contrastadas como Visual Basic .NET y Visual C++ .NET. C# no incluye una biblioteca de clases.

En el resto de este capítulo se explican las características esenciales del lenguaje. Aunque en capítulos posteriores se describen las reglas y las excepciones de una forma muy detallada e incluso a veces matemática, este capítulo se ha redactado dando prioridad a la claridad y la brevedad, a veces incluso a expensas de la integridad. El propósito es proporcionar al lector una introducción al lenguaje que pueda facilitarle la programación de sus primeros programas y la lectura de posteriores capítulos.

1.3.1.1 - Por qué utilizar C#.

- C# es un lenguaje orientado a objetos simple, elegante y con seguridad en el tratamiento de tipos, que permite a los programadores de aplicaciones empresariales crear una gran variedad de aplicaciones.
- C# también proporciona la capacidad de generar componentes de sistema duraderos en virtud de las siguientes características:
- Total compatibilidad entre COM y plataforma para integración de código existente.
- Gran robustez, gracias a la recolección de elementos no utilizados (liberación de memoria) y a la seguridad en el tratamiento de tipos.
- Seguridad implementada por medio de mecanismos de confianza intrínsecos del código.
- Plena compatibilidad con conceptos de metadatos extensibles.
- Además, es posible interaccionar con otros lenguajes, entre plataformas distintas, y con datos heredados, en virtud de las siguientes características:
- Plena interoperabilidad por medio de los servicios de COM+ 1.0 y .NET Framework con un acceso limitado basado en bibliotecas.
- Compatibilidad con XML para interacción con componentes basados en tecnología Web.
- Capacidad de control de versiones para facilitar la administración y la implementación.

1.4 - Web Servises.

Los servicios Web tienen una función fundamental en la arquitectura de aplicaciones .NET. La idea de .NET es crear aplicaciones que utilicen varios servicios Web en interacción para obtener datos y servicios. Sin embargo, aunque se suela hablar de servicios Web en el contexto de .NET, no tiene que esperar al lanzamiento de Microsoft .NET Framework o Microsoft® Visual Studio .NET para generar, implementar o consumir servicios Web. Los servicios Web son un modelo muy general para crear aplicaciones y se pueden implementar en cualquier sistema operativo que permita las comunicaciones a través de Internet.

Un servicio Web es lógica de aplicación programable y accesible mediante protocolos de Internet. Los servicios Web combinan los mejores aspectos del desarrollo basado en componentes con los mejores aspectos del Web. Al igual que los componentes, los servicios Web representan funcionalidad de caja negra que se puede reutilizar sin conocer los detalles de la implementación. A diferencia de las tecnologías de componentes actuales, los servicios Web no son accesibles a través de protocolos específicos del modelo de objetos, como el Modelo de objetos componentes distribuidos (DCOM), Remote Method Invocation (RMI) o Internet Inter-ORB Protocol (IIOP). El acceso a los servicios Web se realiza a través de protocolos para Web y formatos de datos estándar, como el Protocolo de transferencia de hipertexto (HTTP) y el Lenguaje de marcado extensible (XML). Además, la interfaz de un servicio Web se define estrictamente en términos de los mensajes que el servicio Web acepta y genera. Los consumidores del servicio Web pueden implementarse en cualquier plataforma mediante cualquier lenguaje de programación, con tal de que pueda crear y consumir los mensajes definidos para la interfaz del servicio Web.

Este diagrama, al igual que la definición de servicios Web que acabamos de proporcionar, se ocupa de la apariencia externa de los servicios Web. Después de todo, hemos dicho que, con tal de que una aplicación de cliente pueda crear y consumir los mensajes apropiados, no tiene que conocer los detalles de la estructura interna de los servicios Web que utiliza. En cambio, a los programadores de servicios Web les interesará conocer también esta estructura interna.

En la Figura 2 se muestra una arquitectura genérica de servicio Web. La arquitectura se divide en cinco capas lógicas. La más lejana al cliente es la *capa de datos*, que

almacena información requerida por el servicio Web. Por encima de la capa de datos está la *capa de acceso a datos*, que presenta una vista lógica de los datos físicos a la capa de negocios. La capa de acceso a datos aísla la lógica de negocios de los cambios realizados a los almacenes de datos subyacentes y garantiza la integridad de los datos. La capa de negocios implementa la lógica de negocios del servicio Web. Como en la Figura 2, se suele subdividir en dos partes: la *fachada de negocios* y la *lógica de negocios*. La fachada de negocios proporciona una interfaz sencilla que se asigna directamente a las operaciones expuestas por el servicio Web. Utiliza servicios proporcionados por la capa de lógica de negocios. En un servicio Web sencillo, toda la lógica de negocios podría implementarse mediante la fachada de negocios, que interactuaría directamente con la capa de acceso a datos. Las aplicaciones de cliente interactúan con el *agente de escucha* de servicios Web. El agente de escucha se encarga de recibir los mensajes entrantes que contienen solicitudes de servicios, analizar los mensajes y enviar la solicitud al método apropiado en la fachada de negocios. Si el servicio devuelve una respuesta, el agente de escucha también es responsable del empaquetado de la respuesta de la fachada de negocios en un mensaje y su envío al cliente. Además, el agente de escucha controla las solicitudes de contratos y otros documentos acerca del servicio Web. Si lo piensa, la única parte del servicio Web que sabe que forma parte de un servicio Web es el agente de escucha.

1.4.1 - Arquitectura de los Web Services.

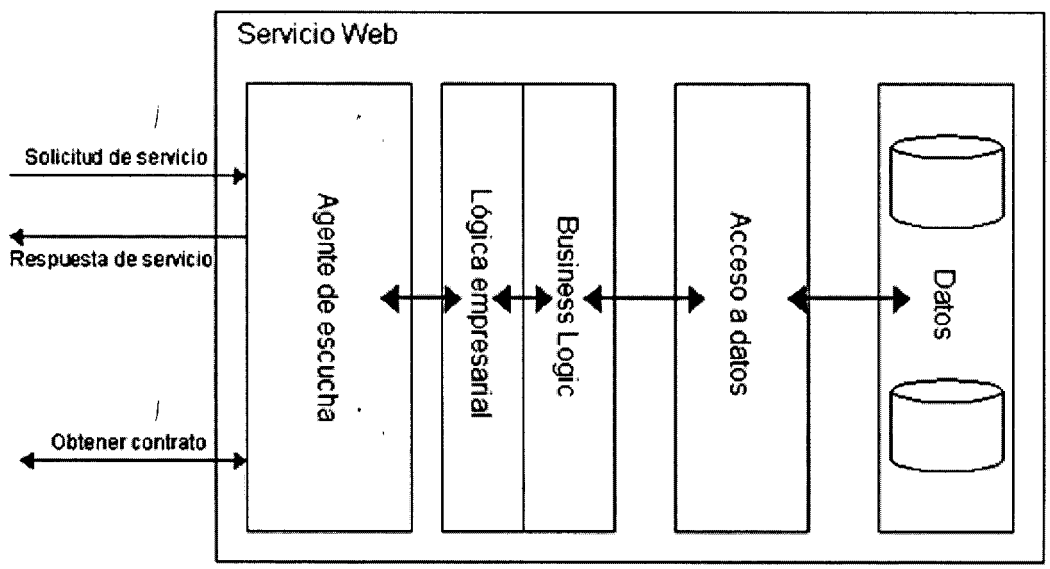


Figura 2: Arquitectura De los Web Services.

1.5 – SQL.

Las aplicaciones en red son cada día más numerosas y versátiles. En muchos casos, el esquema básico de operación es una serie de scripts que rigen el comportamiento de una base de datos.

Debido a la diversidad de lenguajes y de bases de datos existentes, la manera de comunicar entre unos y otras sería realmente complicada a gestionar de no ser por la existencia de estándares que nos permiten el realizar las operaciones básicas de una forma universal .

Es de eso de lo que trata el Structured Query Language que no es mas que un lenguaje estándar de comunicación con bases de datos. Hablamos por tanto de un lenguaje normalizado que nos permite trabajar con cualquier tipo de lenguaje (ASP o PHP) en combinación con cualquier tipo de base de datos (MS Access, SQL Server, MySQL...).

El lenguaje de consulta estructurado (SQL) es un lenguaje de base de datos normalizado, utilizado por el motor de base de datos de Microsoft Jet.

El hecho de que sea estándar no quiere decir que sea idéntico para cada base de datos. En efecto, determinadas bases de datos implementan funciones específicas que no tienen necesariamente que funcionar en otras.

Aparte de esta universalidad, el SQL posee otras dos características muy apreciadas. Por una parte, presenta una potencia y versatilidad notables que contrasta, por otra, con su accesibilidad de aprendizaje.

1.6 – XML.

XML es un meta-lenguaje de marcado que proporciona un formato para describir datos estructurados y permitir a los programadores describir y ofrecer datos estructurados y complejos desde cualquier aplicación, de forma estándar y coherente. XML facilita la realización de declaraciones de contenido más precisas y proporciona resultados de búsqueda más significativos en varias plataformas. Además, se utiliza XML en la nueva generación de aplicaciones para Web con el fin de presentar y manipular datos.

La eficacia y la belleza de XML se debe a que mantiene la separación entre la interfaz de usuario y los datos estructurados. A diferencia del Lenguaje de marcado de hipertexto (HTML), cuyas etiquetas permiten mostrar una palabra en negrita o cursiva, XML proporciona un marco para etiquetar datos estructurados. Una etiqueta XML puede indicar que sus datos asociados son un precio de distribuidor, un impuesto de venta, el título de un libro, una cantidad de precipitación o cualquier otra información deseada. A medida que se extiendan las etiquetas XML en las organizaciones para sus intranets y entre los usuarios de Internet, aumentará la capacidad de buscar y manipular datos, independientemente de las aplicaciones en que éstos residan. Cuando una aplicación encuentra datos XML, puede transmitirlos a través de la red y mostrarlos en un explorador Web (como Microsoft® Internet Explorer) de varias maneras diferentes, o pasarlos a otras aplicaciones que los procesarán y presentarán.

Se puede utilizar XML, que proporciona un estándar de datos para codificar contenido, semántica y esquemas para varios escenarios distintos, tanto sencillos como complejos, con el fin de marcar:

Un documento normal

Un registro estructurado, como un registro de citas o un pedido

Un objeto con datos y métodos, como el formato permanente de un objeto de Java o un control ActiveX®

Un registro de datos, como el conjunto de resultados de una consulta

Metacontenido de un sitio Web, como el Formato de definición de canales (CDF, *Channel Definition Format*)

Una representación gráfica, como la interfaz de usuario de una aplicación

Entidades y tipos estándar de esquemas

Todos los vínculos entre la información y las personas en el Web

Cuando los datos están en el equipo cliente, se pueden manipular, editar y presentar de varias maneras diferentes sin viajes de retorno al servidor. Ahora los servidores pueden ser más escalables, a causa de la menor carga computacional y de ancho de banda. Además, como los datos se intercambian en formato XML, se pueden combinar fácilmente desde distintos orígenes.

Para obtener más información sobre XML y sus aplicaciones, vea Por qué XML.

1.6.1 - Servicios Web y XML.

A diferencia de las tecnologías de componentes actuales, los servicios Web no utilizan protocolos específicos del modelo de objetos como el Modelo de objetos componentes distribuidos (DCOM), Remote Method Invocation (RMI) o Internet Inter-ORB Protocol (IIOP), que requieren infraestructuras homogéneas y específicas en los equipos cliente y en los equipos que ofrecen servicios. Aunque las implementaciones muy acopladas a tecnologías de componentes específicas son perfectamente aceptables en un entorno controlado, no resultan prácticas en el Web. A medida que cambia el grupo de participantes en un proceso de negocios y cambia la tecnología con el tiempo, se hace más difícil garantizar una sola infraestructura unificada para todos los participantes. Los servicios Web se basan en un enfoque distinto: se comunican mediante protocolos ubicuos para Web y formatos de datos como el Protocolo de transferencia de hipertexto (HTTP) y XML. Cualquier sistema compatible con estos estándares para Web será compatible con los servicios Web.

Además, un contrato de servicio Web describe los servicios proporcionados en términos de mensajes que el servicio acepta y genera en lugar de en términos de la implementación del servicio. Al centrarse únicamente en los mensajes, el modelo de servicios Web es totalmente independiente del lenguaje, la plataforma y el modelo de objetos que se utilice. Se puede implementar un servicio Web mediante las características de cualquier lenguaje de programación, modelo de objetos y plataforma.

XML es la opción obvia para definir un lenguaje estándar a la vez que ampliable para representar comandos y datos con tipo. Aunque se podrían definir reglas para representar comandos y datos con tipo mediante otras técnicas, (como la codificación en cadenas de consulta), XML está diseñado específicamente como un metalenguaje estándar para describir datos.

XML es también la tecnología que habilita los contratos de servicios Web. El Lenguaje de contratos de servicio (SCL, *Service Contract Language*) es una gramática XML para documentar contratos de servicios Web. Como SCL está basado en XML, los programadores y las herramientas de programación pueden crear e interpretar los contratos fácilmente.

1.7 - Lenguaje de Modelación

1.7.1 – UML.

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

El UML esta compuesto por diversos elementos graficos que se combinan para conformar diagramas. Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos. En lugar de indicarle a usted cuales son 10s elementos y las reglas, veamos directamente 10s diagramas ya que 10s utilizara para hacer el análisis del sistema.

UML es una especificación de notación orientada a objetos. Se basa en las anteriores especificaciones BOOCH, RUMBAUGH y COAD-YOURDON. Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representa la arquitectura del proyecto.

Con UML nos debemos olvidar del protagonismo excesivo que se le da al diagrama de clases, este representa una parte importante del sistema, pero solo representa una vista estática, es decir muestra al sistema parado. Sabemos su estructura pero no sabemos que le sucede a sus diferentes partes cuando el sistema empieza a funcionar. UML introduce nuevos diagramas que representa una visión dinámica del sistema. Es decir, gracias al diseño de la parte dinámica del sistema podemos darnos cuenta en la fase de diseño de problemas de la estructura al propagar errores o de las

partes que necesitan ser sincronizadas, así como del estado de cada una de las instancias en cada momento. El diagrama de clases continua siendo muy importante, pero se debe tener en cuenta que su representación es limitada, y que ayuda a diseñar un sistema robusto con partes reutilizables, pero no a solucionar problemas de propagación de mensajes ni de sincronización o recuperación ante estados de error. En resumen, un sistema debe estar bien diseñado, pero también debe funcionar bien.

UML también intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implementar un lenguaje de modelado común para todos los desarrollos se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

UML es ahora un Standard, no existe otra especificación de diseño orientado a objetos, ya que es el resultado de las tres opciones existentes en el mercado. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que UML ha sido diseñado para modelar cualquier tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otro ramo.

UML permite la modificación de todos sus miembros mediante estereotipos y restricciones. Un estereotipo nos permite indicar especificaciones del lenguaje al que se refiere el diagrama de UML. Una restricción identifica un comportamiento forzado de una clase o relación, es decir mediante la restricción estamos forzando el comportamiento que debe tener el objeto al que se le aplica.

1.8 - Metodología de Desarrollo Utilizada

1.8.1 – RUP.

El Rational Unified Process (RUP) es un proceso de ingeniería de software que mejora la productividad del equipo de trabajo y entrega las mejores prácticas del software a todos los miembros del mismo. Los contenidos específicos para e-business del RUP proporcionan una guía específica en áreas tales como la de Modelamiento de Negocios, Arquitecturas Web, Pruebas y Calidad. También proporcionan lineamientos para desarrollar en plataformas IBM Websphere y Microsoft Solution con el fin de acelerar nuestros proyectos de desarrollo Web. RUP esta fuertemente integrado con las diferentes herramientas Rational, permitiéndole a los equipos de desarrollo de ne Digital alcanzar todos los beneficios de las características de los productos Rational, el Unified Modeling Language (UML), y otras mejores prácticas de la industria.

1.9.2 - Funcionamiento de RUP.

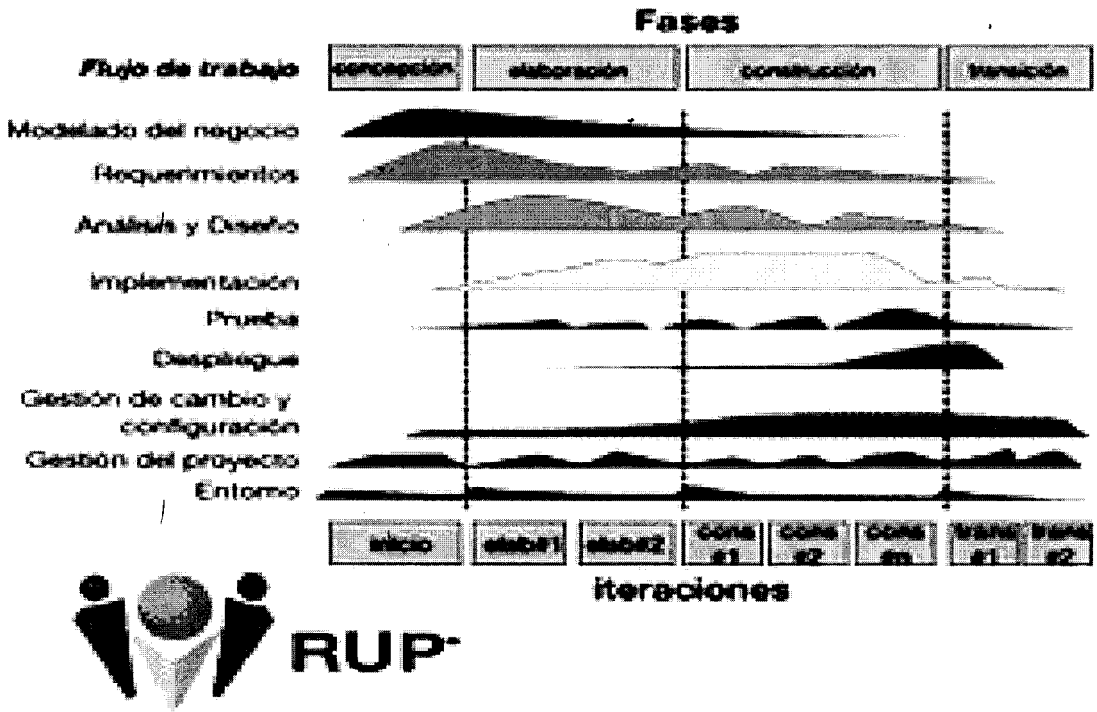


Figura 3 :Funcionamiento de RUP

La creación sólida de software de calidad requiere el conocimiento específico de las tareas que deben llevarse a cabo en cada entorno. Ahí radica la importancia de aplicar un proceso de desarrollo flexible y adaptado a cada objetivo de desarrollo. El proceso RUP (Rational Unified Process) combina un conjunto básico de mejores prácticas aprobadas por el sector con una serie de complementos opcionales del proceso a fin de dar cabida y soporte a proyectos de cualquier envergadura o alcance. Cualquier tipo de proyecto (incluidos los pequeños, los basados en Web, aquéllos fundamentales para un proyecto y los proyectos integrados) permiten obtener unos resultados más acordes con las previsiones gracias a la aplicación del proceso RUP .

1.9 - Gestores Bases de Datos.

1.9.1 - SQL Server 2000.

En la actualidad, las compañías demandan una clase diferente de solución de base de datos. El rendimiento, la escalabilidad y la confiabilidad son esenciales y la anticipación al mercado es crítica. Aparte de estas cualidades empresariales fundamentales, SQL Server 2000 proporciona agilidad a sus operaciones de análisis y administración de datos al permitir a su organización adaptarse rápida y fácilmente para obtener ventaja competitiva en un entorno de cambios constantes. Desde una perspectiva de administración de datos y análisis, resulta crítico transformar los datos sin procesar en inteligencia empresarial y aprovechar las oportunidades que presenta el Web. SQL Server 2000 es un paquete completo de base de datos y análisis de datos que abre las puertas al rápido desarrollo de una nueva generación de aplicaciones comerciales de nivel empresarial, que pueden proporcionar a su compañía una ventaja competitiva crítica. SQL Server 2000 ha obtenido importantes galardones en pruebas de referencia por su escalabilidad y velocidad. Es un producto de base de datos totalmente habilitado para Web que proporciona una compatibilidad fundamental con el Lenguaje de marcado extensible (XML, *Extensible Markup Language*) y la capacidad para realizar consultas en Internet y por encima del servidor de seguridad. Para obtener una descripción detallada de SQL Server 2000, descargue la Guía del producto *(en inglés)* o visite la página Características para obtener más información.

Totalmente habilitado para Web

SQL Server 2000 proporciona completas capacidades de programación de bases de datos basadas en estándares Web. La perfecta compatibilidad con el lenguaje XML y los estándares de Internet le proporcionan la capacidad para almacenar y recuperar fácilmente datos en formato XML con procedimientos almacenados integrados. También puede utilizar datagramas de actualización de XML para insertar, actualizar y eliminar datos con facilidad.

Acceso fácil a los datos a través de Web Con SQL Server 2000, puede utilizar HTTP para enviar consultas a la base de datos, realizar búsquedas de texto en documentos almacenados en la base de datos y ejecutar consultas a través del Web con el lenguaje natural.

Análisis basado en Web eficaz y flexible. Las capacidades de Analysis Services de SQL Server 2000 se extienden a Internet. Puede tener acceso a los datos del cubo y manipularlos por medio de un explorador Web.

Alta escalabilidad y confiabilidad

Con SQL Server 2000 obtendrá una escalabilidad y confiabilidad incomparables. Las capacidades de ampliación de SQL Server satisfacen las necesidades de las exigentes aplicaciones empresariales y de comercio electrónico.

Escalado de ampliación. SQL Server 2000 aprovecha los sistemas de multiproceso simétrico (SMP, *Symmetrical Multiprocessor*). SQL Server Enterprise Edition puede utilizar hasta 32 procesadores y 64 GB de RAM.

Escalado de distribución. El escalado distribuye la base de datos y la carga de datos entre servidores.

Disponibilidad. SQL Server 2000 consigue la máxima disponibilidad gracias a los clústeres de conmutación por error mejorados, el trasvase de registros y las nuevas estrategias de copia de seguridad.

Mayor anticipación al mercado

SQL Server 2000 es la estructura de administración y análisis de datos de Microsoft® Windows Server System™. SQL Server 2000 incluye herramientas que aceleran el desarrollo desde el concepto inicial a la entrega final.

Servicios de análisis integrados y extensibles. Con SQL Server 2000, puede generar soluciones de análisis de extremo a extremo con herramientas integradas para crear valor con los datos. Además, puede llevar a cabo automáticamente

procesos empresariales basados en los resultados del análisis y recuperar de manera flexible conjuntos de resultados personalizados de los cálculos más complejos.

Rápido desarrollo, depuración y transformación de los datos. SQL Server 2000 presenta la capacidad para optimizar y depurar consultas de manera interactiva, mover y transformar rápidamente datos provenientes de cualquier origen y definir y utilizar funciones como si estuvieran integradas en Transact-SQL. Puede diseñar y codificar visualmente aplicaciones de base de datos con cualquier herramienta de Visual Studio.

Administración y optimización simplificadas. Con SQL Server 2000, resulta sencillo administrar bases de datos de forma centralizada junto a todos los recursos empresariales. Permanezca en línea mientras mueve y copia fácilmente bases de datos entre equipos o instancias.

1.10 - Herramientas

1.10.1 - Rational Rose.

Rational Rose es la herramienta CASE desarrollada por los creadores de UML (Booch, Rumbaugh y Jacobson), que cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables.

Es la herramienta CASE que comercializan los desarrolladores de UML y que soporta de forma completa la especificación del UML.

Esta herramienta propone la utilización de cuatro tipos de modelos para realizar el diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software.

"Rational Rose utiliza un proceso de desarrollo iterativo controlado (controlled iterative process development), donde el desarrollo se lleva a cabo en una secuencia de iteraciones. Cada iteración comienza con una primera aproximación del análisis, diseño e implementación para identificar los riesgos del diseño, los cuales se utilizan

para conducir la iteración, primero se identifican los riesgos y después se prueba la aplicación para que estos se hagan mínimos" [**]

Cuando la implementación pasa todas pruebas que se determinan en el proceso, esta se revisa y se añaden los elementos modificados al modelo de análisis y diseño. Una vez que la actualización del modelo se ha modificado, se realiza la siguiente iteración.

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

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

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

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

1.11 – Conclusiones.

Con el estudio de los fundamentos teóricos de las herramientas, tecnologías, lenguajes que se han abordado en este capítulo se ha llegado a la conclusión que el sistema se desarrollará utilizando como gestor de base de datos, el SQL Server 2000, la programación se hará con Visual Studio.NET y en especial con el lenguaje de programación C# debido a su uso en páginas Web, para el análisis y desarrollo se utilizará el Proceso Unificado de Desarrollo (RUP) que a su vez hará uso del Lenguaje Unificado de Modelado(UML) utilizando como herramienta el Racional Rose.

Capítulo2..... Características del Sistema

2.1 – Introducción.

La creciente demanda de servicios Web como estándar para realizar aplicaciones vinculadas a los negocios y aplicaciones de todo tipo, en el mundo, y en particular en la UCI hace necesario buscar una forma de organización y estructuración de esta nueva tecnología para un mejor rendimiento y eficiencia en sus negocios.

Los objetivos de este capítulo son abordar diferentes temas que explicarán la utilidad de poner las nuevas tecnologías de la informática y las comunicaciones para este proyecto; exponer las características de las herramientas y tecnologías escogidas para la solución del problema, así como argumentar la elección de las mismas y comentar los aportes del nuevo sistema a través de una comparación con productos similares.

La (UCI) Universidad de las Ciencias Informáticas es punto de partida para el desarrollo informático de nuestro país en ella se suceden una serie de proyectos en aras de digitalizar la sociedad cubana.**"UCI CIUDAD DIGITAL"** es uno de los proyectos fundadores de estas ideas futuristas.

Como parte de este proyecto se desarrollará un sistema para la informatización del área de alimentación en la UCI, el cual se enmarca dentro del área de Alimentos específicamente donde constará de una aplicación que coleccione y de respuesta a todos los problemas que suceden en el área en aras de mejorar el funcionamiento y el nivel de trabajo de dicha área y de la universidad en general. Este proyecto podrá ser utilizado en todas las instituciones que conste con las herramientas tecnológicas necesarias, tenga las condiciones de trabajo similares a la de nuestro centro.

2.2 - Objeto de Estudio.

En los momentos actuales el control de acceso al comedor, en la Universidad, es tomado de forma manual prácticamente lo cual es muy engorroso a la hora de trabajar con ellos, por lo que es de suma importancia lograr de forma automática el control efectivo de todas estas tareas para mejorar el funcionamiento del área donde suceden estos hechos.

2.3 – Objetivo.

Confeccionar una aplicación bien concebida, que conste con todos los requisitos necesarios y que permita el control de acceso al comedor en la Universidad, que se realice de la forma más organizada y fructífera posible.

2.4 - Situación problémica.

El proceso actual del control de acceso al comedor, en la Universidad, es llevado a cabo prácticamente de forma manual con la excepción de algunos de los procesos que necesitan perfeccionarse aún, todo esto origina que exista una gran masa de datos en papel y a su vez difícil de procesar, esto da a lugar al manejo incorrecto de la información como por ejemplo gastos innecesarios de recursos, duplicidad de información, entre otros inconvenientes hace casi imposible el trabajo en general ya que se debe hacer uso de grandes volúmenes de datos y procesar con ellos múltiples métodos que están relacionados con todos los datos del proceso en sí, estos datos son recogidos en la distintas áreas de la universidad, generalmente en las áreas de alimentación, para proceder a la automatización del mismo y evitar que se vuelva muy engorroso este trabajo, además existe la dificultad del procesamiento de estos datos los cuales son procesados por un sistema existente en la dirección de alimentación que es un sistema prácticamente obsoleto y no brinda las prestaciones necesarias con que debe contar la aplicación en general, el cual no brida todos los servicios que se desean como por ejemplo lograr que los reportes puedan ser vistos e interpretados por todos los directivos del área, también tiene un inconveniente que es que no brinda todas las facilidades ni todos los reportes deseados , como por ejemplo la inserción de los menús que se utilizan diariamente, la entrada del recetario técnico donde se encuentran toda la información acerca de todos los posibles alimentos a consumir.

2.5 - Definición del problema.

¿Automatizar el control de control de acceso al comedor en la Universidad?

2.6 - Campo de acción.

Este software podrá ser utilizado en cualquier empresa del país donde se tengan las condiciones mínimas necesarias como por ejemplo, que exista una PC bien equipada, que tenga instalado el SQL Server e Internet Explorer o cualquier otro navegador, no obstante hay que realizar una serie de configuraciones para adaptarlas a las necesidades que ocurren.

2.6.1 - Objeto de automatización.

El Dpto. de alimentación en la Universidad de las Ciencias Informáticas, tiene varias funciones, dentro de ellas se encuentra realizar las tareas pertinentes al área del comedor y en específico obtener reportes diarios de todo lo que acontece dentro de el.

En lo referente al acceso al comedor primeramente llega un comensal que le ofrece el solaping a la auxiliar del comedor que es la encargada de introducir los datos en un papel que da constancia de que el comensal accedió al comedor, una vez introducido los datos comprueba que el comensal no este marcado con anterioridad verificando la hoja que contiene estos datos y comprobando que no exista ninguna anomalía, estas pueden ser que el comensal ya halla pasado al comedor con anterioridad y desee entrar al mismo violando así las normas establecidas, en este caso la auxiliar del comedor dará un reporte especificando que el comensal.

En lo referente al acceso al comedor por parte de lo que se desea automatizar quedaría de la siguiente manera, primeramente llega un comensal que le ofrece el solaping a la auxiliar del comedor que es la encargada de introducir los datos en la aplicación y comprobar que no exista ninguna anomalía, estas pueden ser que el comensal ya halla pasado al comedor con anterioridad y desee entrar al mismo

violando así las normas establecidas, en este caso la aplicación dará un reporte especificando que el estudiante ya paso con anterioridad así como a que hora lo hizo y porque puerta pasó.

Posteriormente si la auxiliar del comedor ingresa los datos del comensal y si no existen anomalías entonces la aplicación le brinda información acerca del usuario procesado obteniendo así la foto del mismo, el nombre completo y a que facultad pertenece y a su vez obtiene los reportes de mostrar tipo de Persona donde el sistema debe mostrar los diferentes tipos de personas que accederán al comedor tales como: Estudiantes, Trabajadores, Profesores y Personal de Seguridad.

Debe mostrar también total a acceder donde el sistema debe mostrar el total de comensales a acceder según el tipo de persona.

También debe calcular total accedido donde el sistema debe calcular y mostrar el total de comensales que acceden al comedor según el tipo de persona (Esto sucede cuando la auxiliar del comedor ha insertado el código de barra).

Debe calcular total denegados donde el sistema debe calcular y mostrar el total de comensales que son denegados según el tipo de persona (Esto sucede cuando la auxiliar del comedor ha insertado el código de barra).

Debe mostrar la diferencia donde el sistema debe calcular y mostrar la diferencia de comensales que existen entre el total a acceder y el total de los accedidos (Esto sucede cuando la auxiliar del comedor ha insertado el código de barra).

El sistema debe ser capaz de comprobar credencial donde debe poder comprobar si una credencial esta activa o no, o sea el estado de la misma.

Debe ser capaz de activar las credenciales donde el sistema debe poder activar una credencial en caso que no este activada.

El sistema de be ser capaz de buscar una persona donde debe buscar una persona en especial según la credencial utilizada y mostrar los resultados de los datos de la persona buscada como por ejemplo: Foto, Nombre, Apellido, Credencial.

Debe validar la entrada al módulo de administración donde el administrador que debe ser una persona con privilegios avanzados, el mismo se autentica para acceder al módulo de administración, de aquí procede si realmente esta registrado como administrador.

Ya dentro del módulo de administración el administrador debe poder crear un nuevo complejo.

Debe poder crear un nuevo comedor así como insertar una puerta donde el administrador debe poder crear una nueva puerta, asignársela a un comedor ya creado e insertarle el IP de la máquina por donde se va a encontrar dicha puerta.

Debe poder insertar una sesión donde el administrador debe poder crear una nueva sesión, debe poder iniciar dicha sesión donde el administrador debe poder insertar una sesión creada, asignarle una distribución específica y ponerla en funcionamiento.

Debe poder registrar un destino, debe brindar reportes de administración donde el administrador obtiene un reporte con todos los datos necesarios para balancear y obtener todo lo relacionado a lo utilizado en el área.

Debe poder restaurar los datos donde el administrador debe poder restaurar los datos para el comienzo de una nueva jornada.

2.7 - Actividades a desarrollar para lograr el funcionamiento de este Sistema.

1. Conectividad en todos los puntos de entrada y salida: Esto posibilita controlar el acceso al comedor, que envíe información a los servidores SQL Server, que realice una serie de evaluaciones y de respuesta rápido.
2. Acoplamiento de la información con la del Área de Alimentación, ya que los datos obtenidos y los reportes deberán ser conocidos e interpretados por el organismo que se dedica al trabajo con esta información
3. Mantener un registro de las personas que entran a la UCI en las últimas semanas, diarias, en una hora en especifico, por áreas y deforma general.
4.

2.8 - Aportes prácticos esperados del trabajo.

El principal aporte que presenta este proyecto es la plena automatización de todos los procesos del control de acceso al comedor de una forma unida e integrada entre si, logrando así una gran facilidad para el trabajo con estos datos por el personal enmarcado en el Área de alimentación de la Universidad (UCI).

2.9 - Aporte teórico.

Se propone una metodología para interactuar con los comensales, como debe ser y organizar el acceso al comedor que sirvan de uso para otros centros del mismo tipo.

2.10 – Hipótesis.

Si se desarrolla una aplicación para el acceso al comedor se ahorrara gran cantidad de recursos, el trabajo será mas confiable, habrá mayor aprovechamiento del tiempo, evitará duplicidad de la información y facilitara el trabajo para los trabajadores del área.

2.11 - Modelo de Negocio.

El Dpto. de alimentación en la Universidad de las Ciencias Informáticas, tiene varias funciones, dentro de ellas se encuentra realizar las tareas pertinentes al área del comedor y en específico obtener reportes diarios de todo lo que acontece dentro de el.

En lo referente al acceso al comedor primeramente llega un comensal que le ofrece el solaping a la auxiliar del comedor que es la encargada de decepcionarlos datos y comprobar que no exista ninguna anomalía, estas pueden ser que el comensal ya halla pasado al comedor con anterioridad y desee entrar al mismo violando así las normas establecidas, en este caso la auxiliar del comedor le dirá al comensal que ya paso con anterioridad.

Posteriormente si la auxiliar del comedor ingresa los datos del comensal y si no existen anomalías entonces accederá sin problemas.

La propuesta del negocio es automatizar todos estos procesos de manera que sea menos engorroso el trabajo manual que se hace. Debido a la cantidad de trámites necesarios. De manera que el acceso al comedor de forma masiva se automatice.

Actores del negocio.

Trabajadores del Negocio.

Casos de uso del negocio.

Diagrama de actividades.

Diagrama de clases modelo objetos.

Explicación de dichos casos de usos.

Ver anexo 1.

2.12 - Definición de los Casos de Uso del sistema.

Un caso de uso constituye una técnica utilizada para describir el comportamiento del sistema, a través de un documento narrativo que define la secuencia de acciones que obtienen resultados de valor para un actor que utiliza un sistema para completar un proceso, sin importar los detalles de la implementación.

Para la definición de los casos de uso se necesita:

Identificar los actores.

Identificar los casos de uso.

Describir los casos de uso.

Identificación de los Actores

Los actores se definen como los roles que puede tener un usuario, pueden ser humanos, otros sistemas, máquinas, hardware, etc. que interactúan con un sistema para de esta forma intercambiar datos, aunque en algunos casos pueden constituir un recipiente pasivo de información.

Actores del negocio	Justificación
Estudiante Trabajadores Profesores Personal de Seguridad.	Son los que inician la acción estando interesados en acceder al comedor.

Trabajadores del negocio	Justificación
Auxiliar del comedor	Es la encargada de introducir los datos del solapin que le entrega el estudiante y comprobar que todo lo referente al mismo. En caso necesario elimina acceso.
Administrador del sistema.	Es el encargado de realizar todas las tareas administrativas en el sistema y el trabajador principal de la aplicación teniendo acceso a todos los módulos de nuestro sistema.Es la persona que "reinicia" la BD una vez finalizado el turno de almuerzo y comida.
Director del complejo. Sub_Director del complejo.	Es el encargado de iniciar la sesión creada y tiene acceso a todos los reportes que revela la aplicación.
Vicerrector. Dpto. Económico. Especialistas.	Tienen acceso a ver los reportes pertinentes a sus áreas.

2.12.1 - Casos de Usos relacionados por Módulos.

2.12.1.1 - Módulo Usuario.

1. Autenticarse
2. Insertar barcode.
3. Cancelar Acceso.

2.12.1.2 - Módulo Administrativo.

1. Validar entrada modulo de administración.
2. Crear un Nuevo Complejo.
3. Crear un Nuevo Comedor.
4. Insertar Puerta.
5. Insertar Sesión.
6. Iniciar Sesión.
7. Registrar Destino.
8. Brindar Reporte.
9. Restaurar Datos.

2.13 - Requerimientos funcionales.

1. Autenticarse.
2. Insertar barcode.
3. Mostrar Foto.
4. Cancelar Acceso.
5. Mostrar tipo de Persona.
6. Mostrar Total a Acceder.
7. Calcular Total Accedido.
8. Calcular Total Denegados.

9. Mostrar Diferencia.
10. Validar entrada modulo de administración.
11. Crear un Nuevo Complejo.
12. Crear un Nuevo Comedor.
13. Insertar Puerta.
14. Insertar Sesión.
15. Iniciar Sesión.
16. Registrar Destino.
17. Brindar Reporte.
18. Restaurar Datos.

2.13.1 - Especificación de los Requisitos Funcionales.

1 Autenticarse

1.1 La auxiliar del comedor debe tener acceso a la aplicación mediante algún usuario activo.

2 Insertar barcode.

2.1 La auxiliar del comedor inserta mediante un escáner el código de barra del solapín del comensal, una vez insertado procede a comprobar la validez del mismo y según el resultado obtenido le da acceso al comedor o se lo rechaza.

3 Mostrar Foto.

3.1 El sistema una vez insertado el código de barra debe mostrar la foto correspondiente al comensal que esta siendo procesado, así se evita los accesos de comensales con solapines ajenos (Esto sucede cuando la auxiliar del comedor ha insertado el código de barra).

4 Cancelar Acceso.

4.1 La auxiliar del comedor debe cancelar el acceso a un comensal cuando ocurre algún problema con el mismo (Esto sucede cuando la auxiliar del comedor ha insertado el código de barra).

5 Mostrar tipo de Persona.

5.1 El sistema debe mostrar los diferentes tipos de personas que accederán al comedor tales como:
Estudiantes
Trabajadores
Profesores
Personal de Seguridad.

6 Mostrar Total a Acceder.

6.1 El sistema debe mostrar el total de comensales a acceder según el tipo de persona.

7 Calcular Total Accedido.

7.1 El sistema debe calcular y mostrar el total de comensales que acceden al comedor según el tipo de persona (Esto sucede cuando la auxiliar del comedor ha insertado el código de barra).

8 Calcular Total Denegados

8.1 El sistema debe calcular y mostrar el total de comensales que son denegados según el tipo de persona (Esto sucede cuando la auxiliar del comedor ha insertado el código de barra).

9 Mostrar Diferencia.

9.1 El sistema debe calcular y mostrar la diferencia de comensales que existen entre el total a acceder y el total de los accedidos (Esto sucede cuando la auxiliar del comedor ha insertado el código de barra).

10 Validar entrada modulo de administración.

10.1 El administrador que debe ser una persona con privilegios avanzados, el mismo se autentica para acceder al módulo de administración, de aquí procede si realmente esta registrado como administrador.

11 Crear un Nuevo Complejo.

11.1 El administrador debe poder crear un nuevo complejo.

12 Crear un Nuevo Comedor.

12.1 El administrador debe poder crear un nuevo comedor.

13 Insertar Puerta.

13.1 El administrador debe poder crear una nueva puerta, asignársela a un comedor ya creado e insertarle el IP de la máquina por donde se va a encontrar dicha puerta.

14 Insertar Sesión.

14.1 El administrador debe poder crear una nueva sesión.

15 Iniciar Sesión.

15.1 El administrador debe poder insertar una sesión creada, asignarle una distribución específica y ponerla en funcionamiento.

16 Registrar Destino.

16.1 El administrador debe poder registrar un nuevo destino.

17 Brindar Reporte.

17.1 El administrador obtiene un reporte con todos los datos necesarios para balancear obtener todo lo relacionado a lo utilizado en el área.

18 Restaurar Datos.

18.1 El administrador debe poder restaurar los datos para el comienzo de una nueva jornada.

2.14 - Requisitos no funcionales.

2.14.1 - Requerimientos de Interfaz o apariencia.

Interfaz amigable con un diseño sencillo y cómodo para los usuarios.

Fácil de usar.

Debe ser confiable y seguro.

El sistema debe ser integrado, controlar todas y cada una de las actividades que se realicen durante los procesos de acceso.

Estará implementado sobre una tecnología Web, facilitando su uso a través de la red.

La base de datos que utilizará el sistema como medio de almacenamiento de la información estará soportada sobre un gestor de bases de datos SQL Server 2000, permitiéndosele interactuar con otros sistemas estableciendo vías de compatibilidad.

2.15 - Requisitos de funcionalidad.

2.15.1 - Tiempo de entrenamiento.

El sistema debe ser sometido a una fase de pruebas en las cuales los usuarios se familiaricen con este y a la vez se puedan detectar posibles errores de este.

2.15.2 - Requisitos de confiabilidad.

1) Disponible todo el tiempo.
2) Tiempo de respuesta mínimo.
3) Preciso en la información.
4) Extensibilidad.
5) Mantenimiento.
6) Salvas de la información.

2.15.3 - Tiempo de respuesta mínimo.

El sistema debe ser capaz de tener un tiempo mínimo de recuperación ante cualquier fallo en una de las operaciones.

2.15.4 - Mantenimiento.

El sistema debe estar bien documentado de forma tal que el tiempo de mantenimiento sea mínimo.

2.15.5 - Salvas de la información.

El sistema debe permitir hacer copias persistentes en otros dispositivos de toda la información.

2.15.6 - Requisitos de performance.

Las características de ejecución del sistema pueden ser esquematizadas en esta sección.

1.- El tiempo de respuesta debe ser corto con respuestas rápidas y eficientes (menos que 10 segundos).

2.- El sistema podrá hospedar alrededor de 20 000 o más comensales.

2.15.7 - Requisitos de portabilidad.

El sistema deberá funcionar sobre plataforma Windows, que es la que más usada hasta el momento en la universidad y para la que se prevé implantar el software.

2.15.8 – Interfaces.

El sistema debe tener una interfaz sencilla, intuitiva, amigable y mantener el formato en páginas similares. En general, fácil de usar.

2.16 - Descripción de los Casos de Uso del sistema.

Un caso de uso constituye una técnica utilizada para describir el comportamiento del sistema, a través de un documento narrativo que define la secuencia de acciones que obtienen resultados de valor para un actor que utiliza un sistema para completar un proceso, sin importar los detalles de la implementación.

Ver anexo 2.

2.17 - Conclusiones.

En este capitulo se aborda todo lo referente a los requisitos funcionales y no funcionales del sistema a implementar así como los casos de uso del mismo que se implementaron en la aplicación.

Capítulo III: Análisis y Diseño del Sistema.

3.1 - Introducción

En este capítulo se realizará el análisis y diseño del sistema, teniendo en cuenta los requerimientos del que se hicieron en el capitulo anterior, aquí se definirán las clases preliminares, el diagrama de clases del análisis, además se muestran los diagramas de secuencia por cada caso de uso expandido al igual que los diagramas de clases del diseño. El diseño pone de relieve la solución lógica: cómo el sistema cumple con los requerimientos.

3.2 - Modelo Conceptual

El paso esencial de un análisis orientado a objetos es descomponer el problema en conceptos u objetos individuales. El modelo conceptual es la representación gráfica de los mismos.

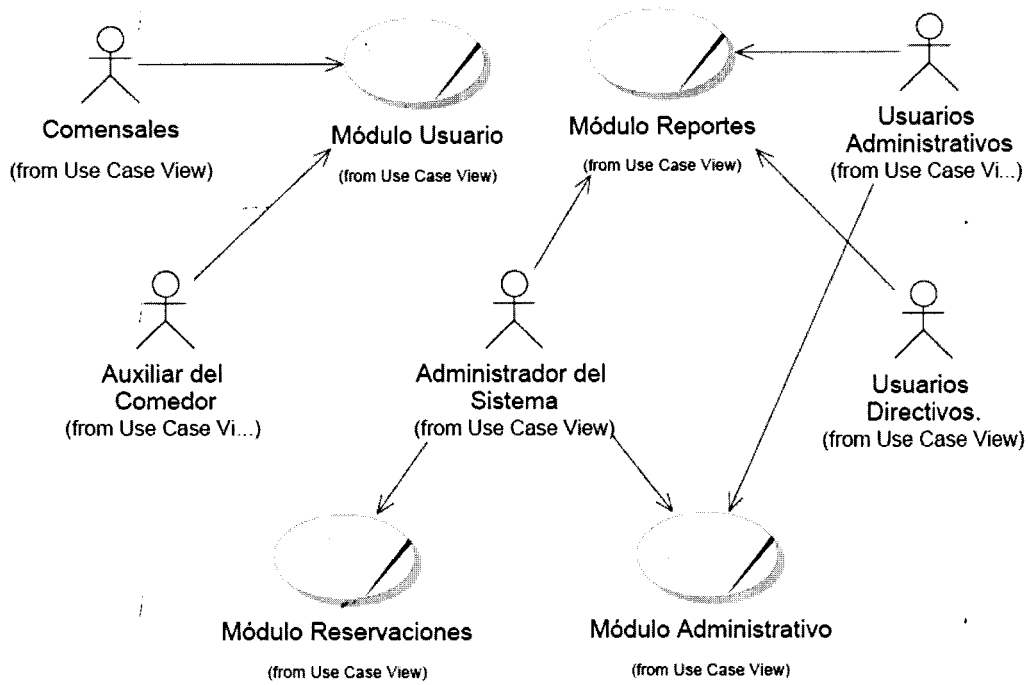


Figura 7: Especificación de los Actores del Negocio.

3.3 - Diseño

3.3.1 - Diagrama De Clases de Análisis.

Modulo Usuario

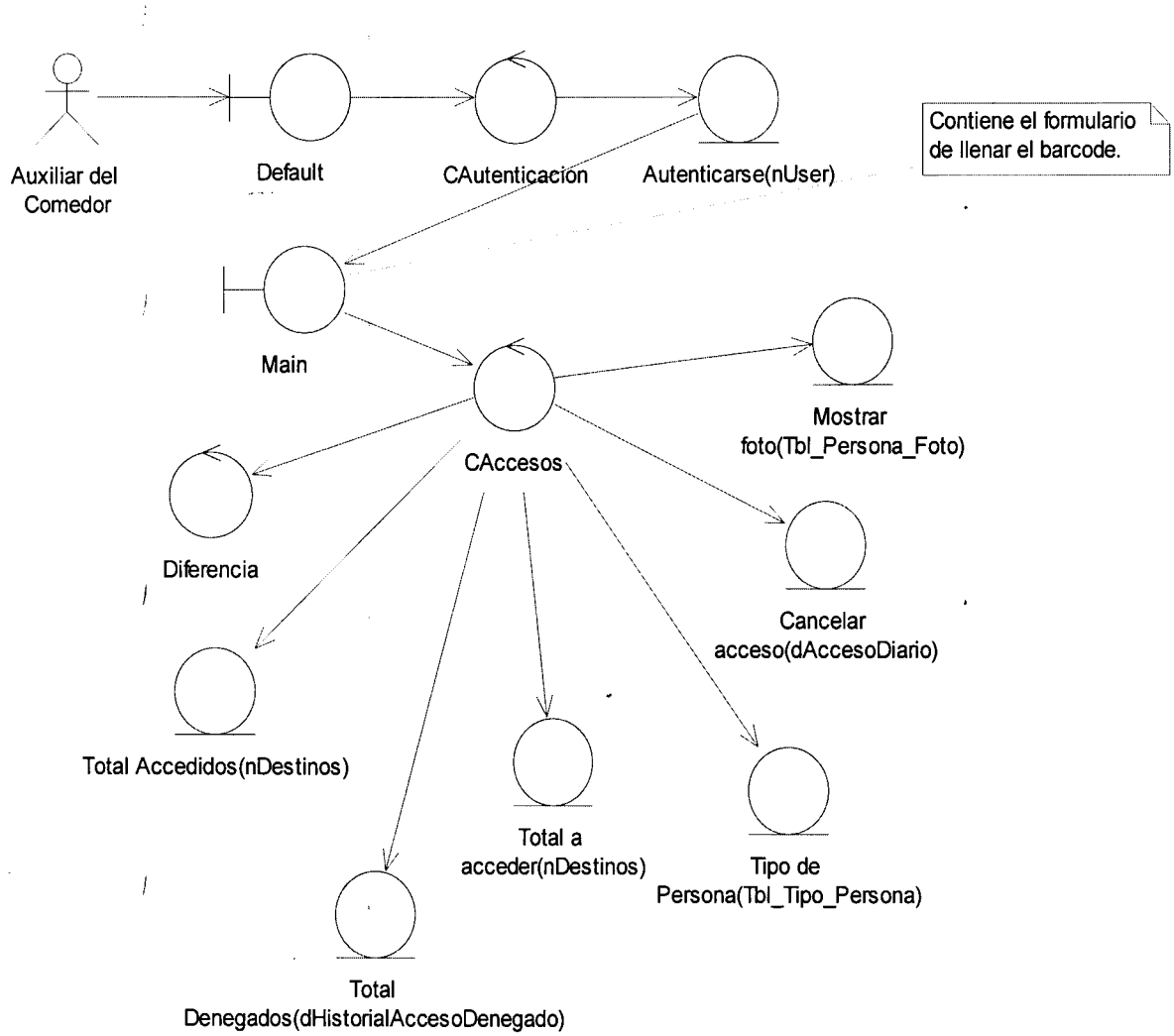


Figura26: Especificación del Diagrama de Clases Modulo Usuario.

Modulo Administración.

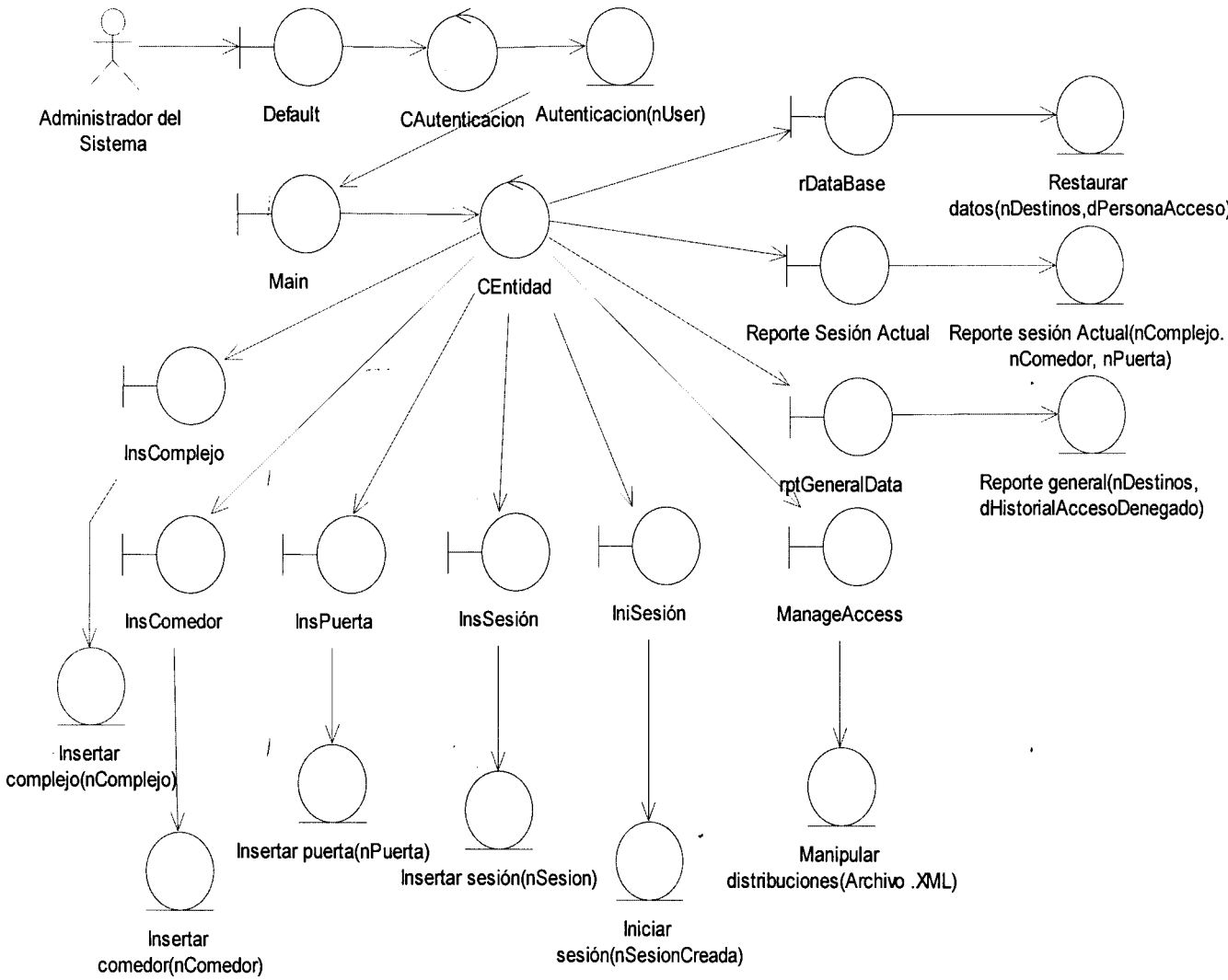


Figura27: Especificación del Diagrama de Clases Modulo Administración.

3.4 - Descripción de las clases.

Las clases que a continuación se describen son las que aparecen en el diagrama de clases, y que tienen atributos y/o métodos.

Clases controladoras

Nombre: CEntidad	
Tipo de clase : Controladora	
Nombre:	`Insertar(params object[] AArgs)`
Descripción:	Este método tiene como objetivo insertar los parámetros pasados.
Nombre:	`Modificar(params object[] AArgs)`
Descripción:	Este método tiene como objetivo modificar los parámetros seleccionados.
Nombre:	`Actualizar(params object[] AArgs)`
Descripción:	Este método tiene como objetivo actualizar los parámetros insertados.
Nombre:	`Eliminar()`
Descripción:	Este método tiene como objetivo eliminar parámetros seleccionados.

Nombre: CAcceso	
Tipo de clase : Controladora	
Nombre:	`ValidaCodigo(string Code)`
Descripción:	Este método tiene como objetivo controlar que el barcode insertado sea el correcto y de no serlo brindar una serie de instrucciones como forma de denegado.
Nombre:	`BuscarEstadoAcceso(string idPersona)`
Descripción:	Este método tiene como objetivo hacer la búsqueda de la persona que se esta procesando según el barcode insertado.
Nombre:	`BuscarInterno(string idPersona)`
Descripción:	Este método tiene como objetivo devolver si la persona buscada es interno o no.
Nombre:	`BuscarTipoPersona(string idPersona)`
Descripción:	Este método tiene como objetivo devolver el tipode persona que esta siendo procesada ya sea estudiante, dirigente, eventual etc.
Nombre:	`ChequeReservacion(string idPersona)`
Descripción:	Este método tiene como objetivo controlar que se halla hecho la reservación pertinente para dicho comensal o que se encuentre activa la distribución asignada.
Nombre:	`ChequeaAccesoDoble(string idPersona)`
Descripción:	Este método tiene como objetivo controlar el comensal no accede al comedor mas de una vez.

Nombre: CAcreditador	
Tipo de clase : Controladora	
Nombre:	`Acreditar(System.Web.Services.Protocols.SoapHttpClientProtocol ws)`
Descripción:	Este método tiene como objetivo acreditar los WEB SERVICES para ponerlos en funcionamientos.

Nombre: CAccesoDiario	
Tipo de clase : Entidad	
Nombre:	`BuscarAccesoDoble(string aIdPersona, int aIdSesion)`
Descripción:	Este método tiene como objetivo controlar el comensal no accede al comedor mas de una vez.Devuelve 0 si ya paso y 1 si no ha pasado.
Nombre:	`EliminarAccesoDiario(string aIdAcceso)`
Descripción:	Este método tiene como objetivo eliminar el acceso.

Nombre: CAccesoDiarioDenegado	
Tipo de clase : Entidad	
Nombre:	`CreaAccesoDiarioDenegado(string aIdPersona, int aIdSesion, int aIdMensaje, int aIdConjunto, string aDenegadoPor)`
Descripción:	Este método tiene como objetivo crear acceso diario denegado donde se inserta los datos a entrar.
Nombre:	`EliminarAccesoDiarioDenegado(string aIdAcceso)`

Descripción:	Este método tiene como objetivo eliminar el acceso diario denegado.

Nombre: CAccesoTemp	
Tipo de clase : Entidad	
Nombre:	`CreaAccesoTemp(int aIdSesion, string aCodigo, int aIdConjunto)`
Descripción:	Este método tiene como objetivo crear acceso temporal donde se inserta los datos a entrar.
Nombre:	`EliminarAccesoTemp(string aIdAcceso)`
Descripción:	Este método tiene como objetivo eliminar el acceso diario denegado.
Nombre:	`BuscarAccesoDobleTemp(string aCode, int aIdSesion)`
Descripción:	Este método tiene como objetivo buscar si ha accedido doble, devuelve 0 si no ha pasado y 1 si ya paso.

Nombre: CCausa	
Tipo de clase : Entidad	
Nombre:	`CreaCausa(string aCausa)`
Descripción:	Este método tiene como objetivo crear la causa ya sea : 1 Acceso Permitido 2 Acceso Denegado 3 Advertencia.

Nombre:	`EliminarCausa(int aIdCausa)`
Descripción:	Este método tiene como objetivo eliminar una causa que ya halla sido creada.
Nombre:	`ModificarCausa(string aCausa ,int aIdCausa)`
Descripción:	Este método tiene como objetivo modificar una causa que ya halla sido creada.
Nombre:	`ListarCausas()`
Descripción	Este método tiene como objetivo listar las causas que ya hallan sido creadas.

Nombre: CComedor	
Tipo de clase : Entidad	
Nombre:	`CreaComedor(string aNombre, int aComplejo)`
Descripción:	Este método tiene como objetivo insertar un nuevo comedor y asignárselo a un complejo.
Nombre:	`EliminarComedor(int aIdComedor)`
Descripción:	Este método tiene como objetivo eliminar comedor que ya halla sido creado.
Nombre:	`ModificarComedor(string aComedor ,int aIdComedor)`
Descripción:	Este método tiene como objetivo modificar un comedor que ya halla

	sido creado.
Nombre:	`ListarComedor()`
Descripción	Este método tiene como objetivo listar los comedores que ya hallan sido creados.

Nombre: CComplejo	
Tipo de clase : Entidad	
Nombre:	`CreaComplejo(string aNombre)`
Descripción:	Este método tiene como objetivo insertar un nuevo complejo.
Nombre:	`EliminarComplejo(int aIdComplejo)`
Descripción:	Este método tiene como objetivo eliminar un complejo que ya halla sido creado.

Nombre: CDestino	
Tipo de clase : Entidad	
Nombre:	`CreaDestino(string aNombre)`
Descripción:	Este método tiene como objetivo insertar un nuevo destino.
Nombre:	`EliminarDestino(int aIdDestino)`

Descripción:	Este método tiene como objetivo eliminar un destino que ya halla sido creado.

Nombre: CPersonaAcceso	
Tipo de clase : Entidad	
Nombre:	`BuscarEstadoPersona(string idPersona)`
Descripción:	Este método tiene como objetivo buscar el estado en que se encuentra la persona que esta siendo procesada ya sea: 1 Acceso Permitido 2 Acceso Denegado `//si retorna 0 es estado no fue reportado`

Nombre: CPuerta	
Tipo de clase : Entidad	
Nombre:	`CreaPuerta(int aIdComedor, string aPuerta, string aIPpc)`
Descripción:	Este método tiene como objetivo insertar una puerta, asignárselo a un complejo ya creado y asignarle un IP real.
Nombre:	`EliminarPuerta(int aIdConjunto)`
Descripción:	Este método tiene como objetivo eliminar una puerta que ya ha sido creada.
Nombre:	`ModificarPuerta(int aIdPuerta, int aIdComedor, string`

	aPuerta, string aIPpc,int aIdConjunto)
Descripción:	Este método tiene como objetivo modificar la información insertada después que se crea una puerta y se le da sus propiedades pertinentes.
Nombre:	ListarPuertas()
Descripción:	Este método tiene como objetivo mostrar las puertas insertadas así como sus propiedades pertinentes.

Nombre: CSesion	
Tipo de clase : Entidad	
Nombre:	CreaSesion(string aSesion)
Descripción:	Este método tiene como objetivo insertar una nueva sesión.
Nombre:	EliminarSesion(int aIdSesion)
Descripción:	Este método tiene como objetivo eliminar sesión ya creada.
Nombre:	ModificarComedor(string aSesion ,int aIdSesion)
Descripción:	Este método tiene como objetivo modificar la información insertada después que se crea una sesión y se le da sus propiedades pertinentes.
Nombre:	ListarSesiones()

Descripción:	Este método tiene como objetivo mostrar las sesiones creadas así como sus propiedades pertinentes.

Tipo de clase :Interfaces (WEB SERVICES) **Nombre: WSComedor**	
Nombre:	`ReportAccess(string idPersona, int idEstadoAcceso, string idDestino, string Causa)`
Descripción:	Este método actualiza el reporte correspondiente al día.

Nombre:	`wsAcceso()`
Descripción:	Este método brinda la URL donde se encuentra el web service del reporte.

Tipo de clase :Interfaces (WEB SERVICES) **Nombre: WSAreas**	
Nombre:	`Actualizar_Area_Dado_Codigo_DS(string DCodigo, string strDNombre, string DSubordinadaA, string strDResponsable, bool blnDEliminada, ref int returnValue)`
Descripción:	Este método actualiza área a la que pertenece dado el código de la persona y retorna una serie de datos actualizados

Nombre:	`Actualizar_ParoArea_DS(System.DateTime dtmDFecha, string DCodigoArea, string strDDescripcion, ref int returnValue)`

Descripción:	Este método actualiza el paro del área dado la fecha, el código del área y retorna datos actualizados.
Nombre:	`Actualizar_Plantilla_DS(string DIDArea, long DIDCargo, int DCantidad, ref int returnValue)`
Descripción:	Este método actualiza la plantilla dado el ID del área, el ID de cargo y la cantidad retorna datos actualizados.
Nombre:	`Borrar_Area_Dado_Codigo_DS(string DCodigo, ref int returnValue)`
Descripción:	Este método borra el área dado el código.
Nombre:	`Desactivar_Activar_Area_Dado_Codigo_DS(string DCodigo, bool blnDEliminada, ref int returnValue)`
Descripción:	Este método desactiva un área dado el código.

Tipo de clase : Interfaces **Nombre: AutenticacionUsuario**	
Nombre:	`Crear_Pagina_autenticar_User()`
Descripción:	Esta Clase muestra la página de autenticación del usuario.

Tipo de clase : Interfaces **Nombre: ErrorUser**	
Nombre:	`Crear_Pagina_Error()`
Descripción:	Esta Clase muestra la página de error al autenticarse

	erróneamente el usuario.

Tipo de clase :Interfaces **Nombre: Usuario**	
Nombre:	`Crear_Pagina_Usuario()`
Descripción:	Esta Clase muestra la página de Usuario.

Tipo de clase :Interfaces **Nombre: AutenticacionAdmin**	
Nombre:	`Crear_Pagina_autenticar_Admin()`
Descripción:	Esta Clase muestra la página de autenticación del administrador.

Tipo de clase :Interfaces **Nombre: ErrorAdmin**	
Nombre:	`Crear_Pagina_Error()`
Descripción:	Esta Clase muestra la página de error al autenticarse erróneamente el administrador.

Tipo de clase :Interfaces **Nombre: Administración**	
Nombre:	`Crear_Pagina_Administracion()`
Descripción:	Esta Clase muestra la página de administración.

Tipo de clase :Interfaces **Nombre: InsComplejo**	
Nombre:	`Crear_Pagina_insComplejo ()`
Descripción:	Esta Clase muestra la página de insertar un complejo.

Tipo de clase :Interfaces **Nombre: InsComedor**	
Nombre:	`Crear_Pagina_insComedor ()`
Descripción:	Esta Clase muestra la página de insertar un comedor y asignárselo a un complejo ya creado.

Tipo de clase :Interfaces **Nombre: InsPuerta**	
Nombre:	`Crear_Pagina_InsPuerta ()`
Descripción:	Esta Clase muestra la página de insertar una puerta, asignárselo a un complejo ya creado y asignarle un IP.

Tipo de clase :Interfaces **Nombre: ActualizarIP**	
Nombre:	`Crear_Pagina_ActualizarIP()`
Descripción:	Esta Clase muestra la página de actualizar IP, Puerta, Complejo.

Tipo de clase :Interfaces **Nombre: InsSesion**	
Nombre:	`Crear_Pagina_InsSesion ()`
Descripción:	Esta Clase muestra la página de insertar una sesión.

Tipo de clase :Interfaces **Nombre: InicSesion**	
Nombre:	`Crear_Pagina_Reporte_InicSesion()`
Descripción:	Esta Clase muestra la página de Iniciar sesión y asignársela a una distribución ya creada.

Tipo de clase :Interfaces **Nombre: ManageAccess**	
Nombre:	`Crear_Pagina_Reporte_ManageAccess ()`
Descripción:	Esta Clase muestra la página de distribuciones.

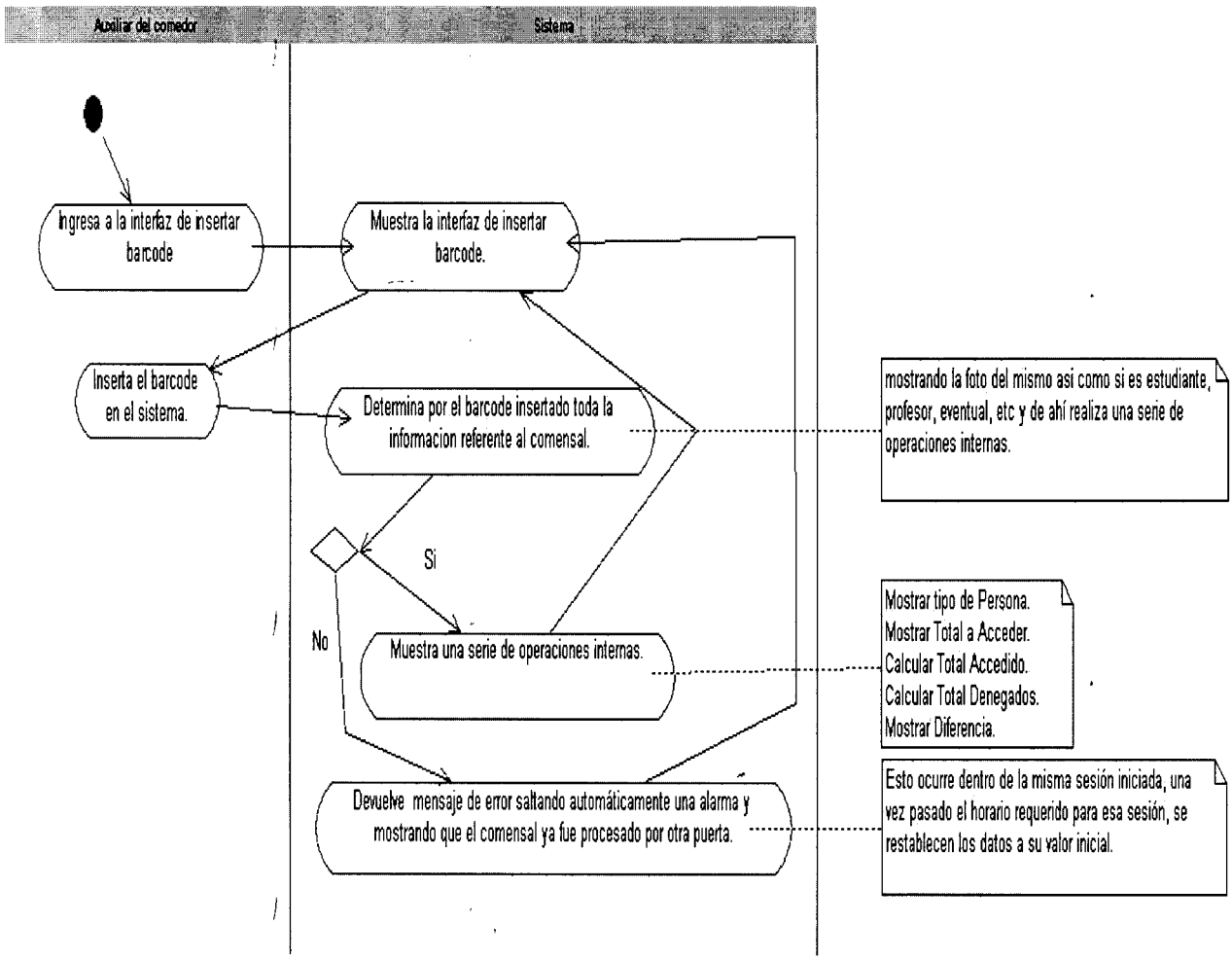
Tipo de clase :Interfaces **Nombre: DocReport**	
Nombre:	`Crear_Pagina_Reporte_General()`
Descripción:	Esta Clase muestra la página de reportes generales.

Tipo de clase :Interfaces **Nombre: DocReport 1**	
Nombre:	`Crear_Pagina_Reporte_Sesion()`
Descripción:	Esta Clase muestra la página de reportes por sesión.

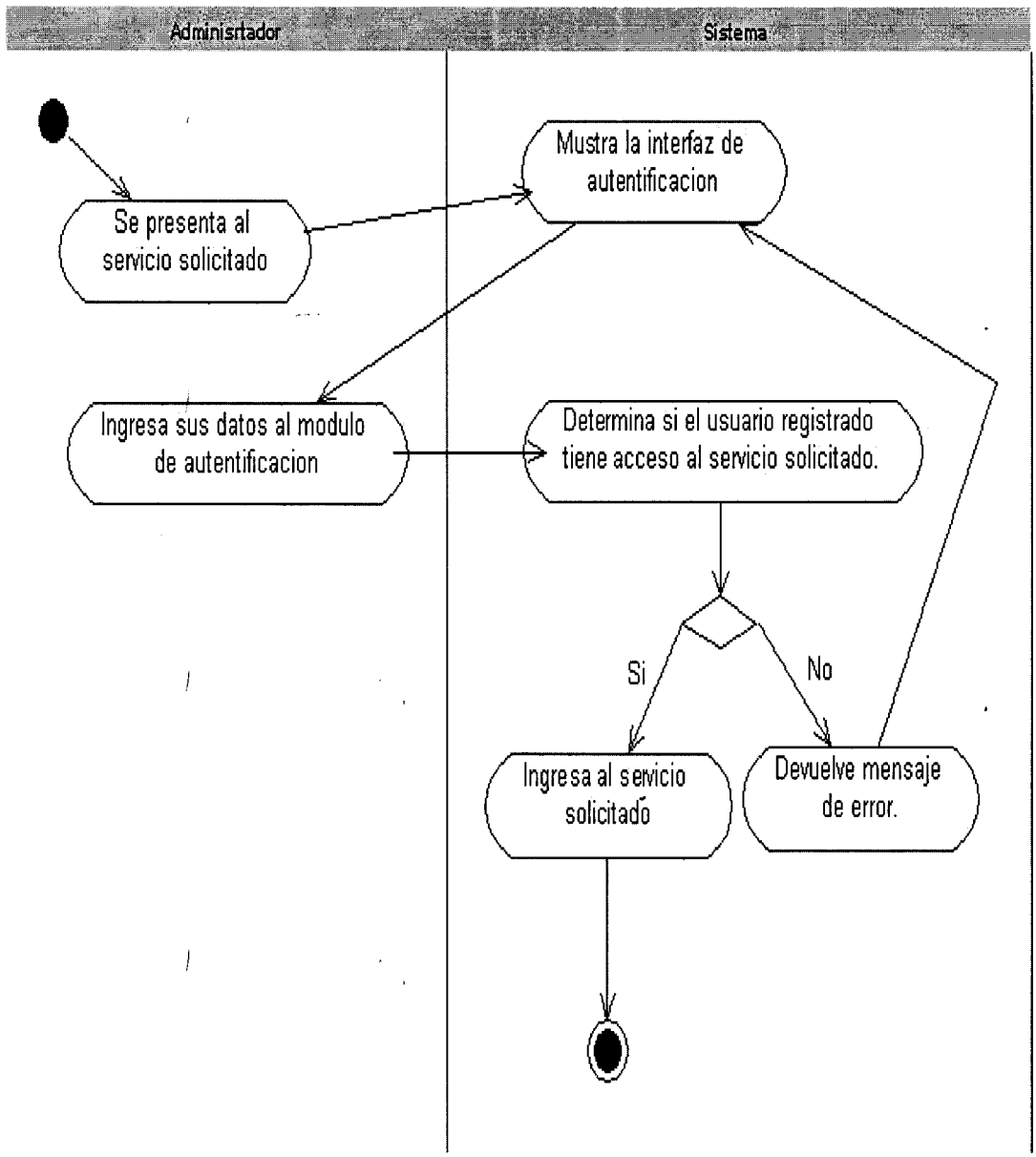
Tipo de clase :Interfaces **Nombre: rDataBase**	
Nombre:	`Crear_Pagina_Reporte_rDataBase ()`
Descripción:	Esta Clase restaura la Base de datos y retorna a la página de administración.

3.5 - Diagrama De Actividades.

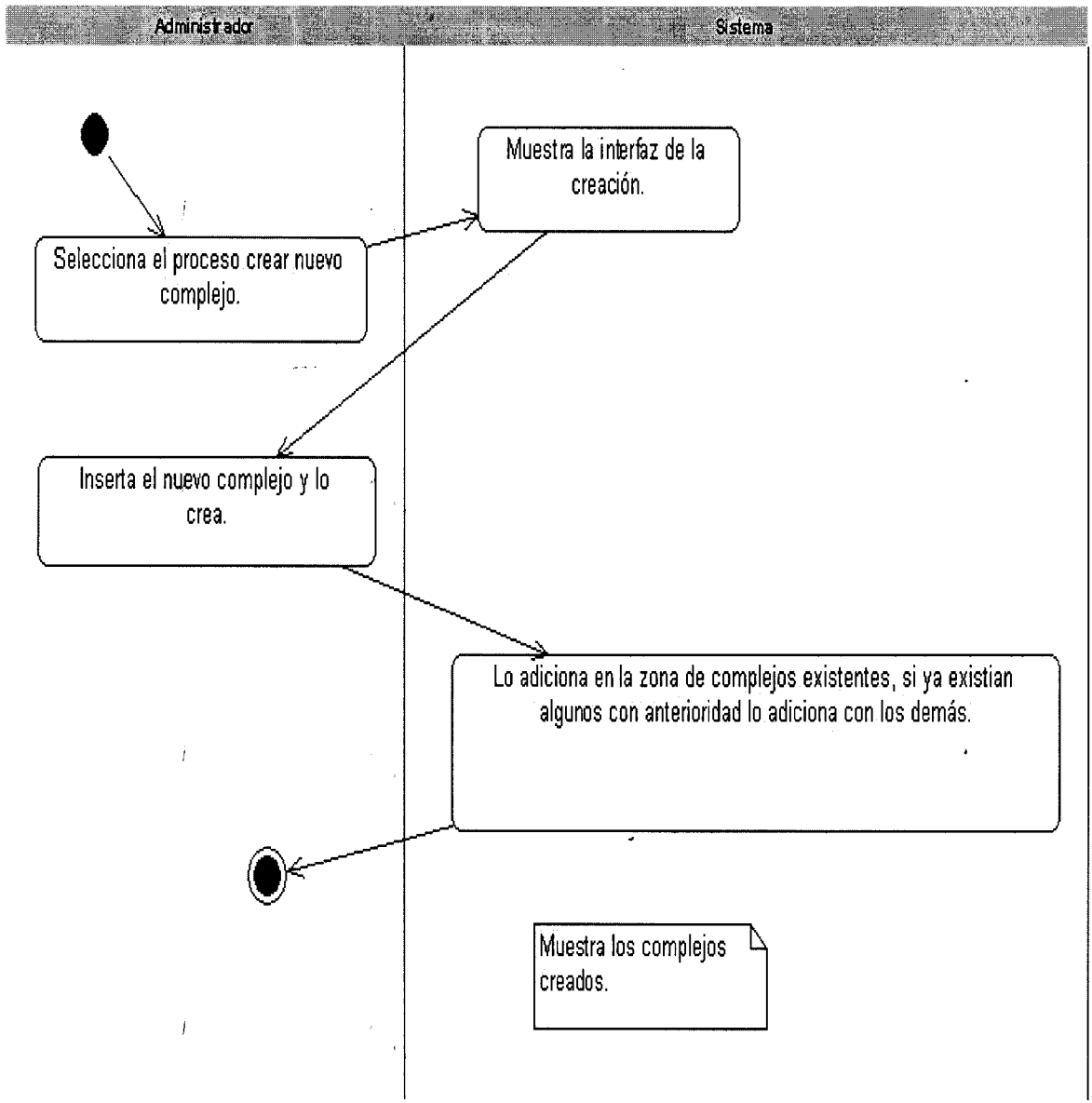
3.5.1 - Diagrama De Actividades caso de uso Insertar Barcode.



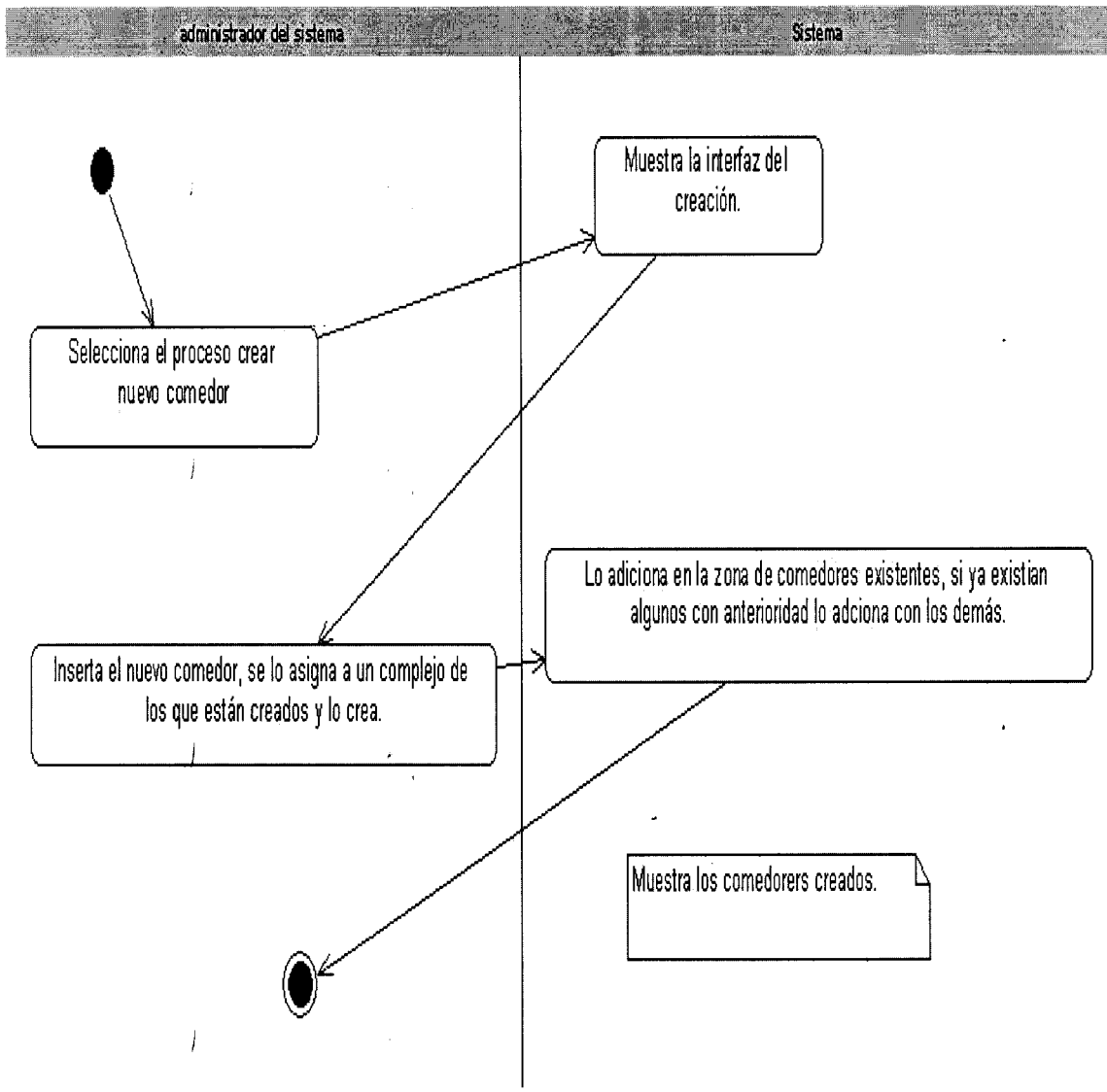
3.5.2 - Diagrama De Actividades caso de uso Autenticar Administrador.



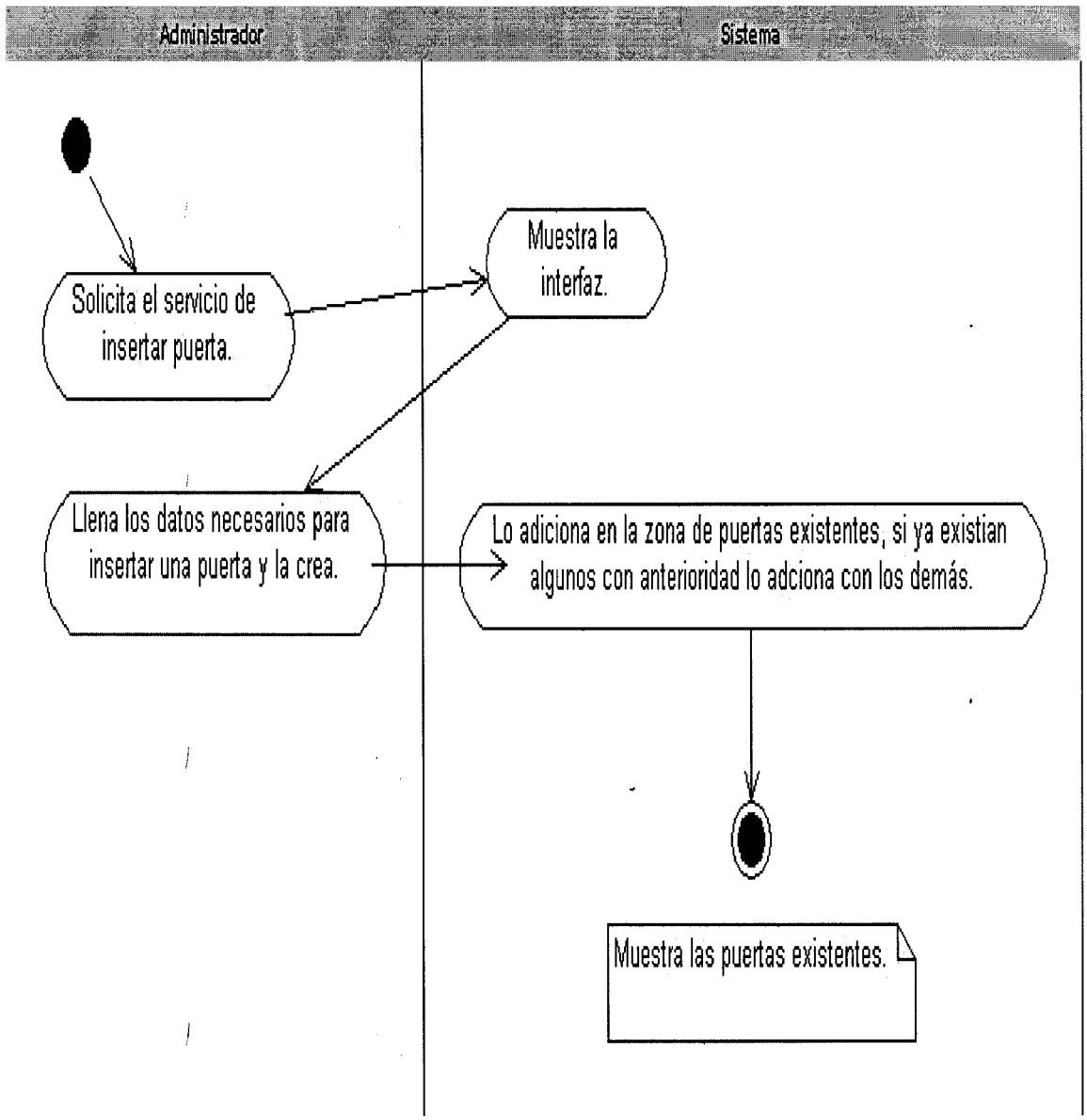
3.5.3 - Diagrama De Actividades caso de uso Crear nuevo Complejo.



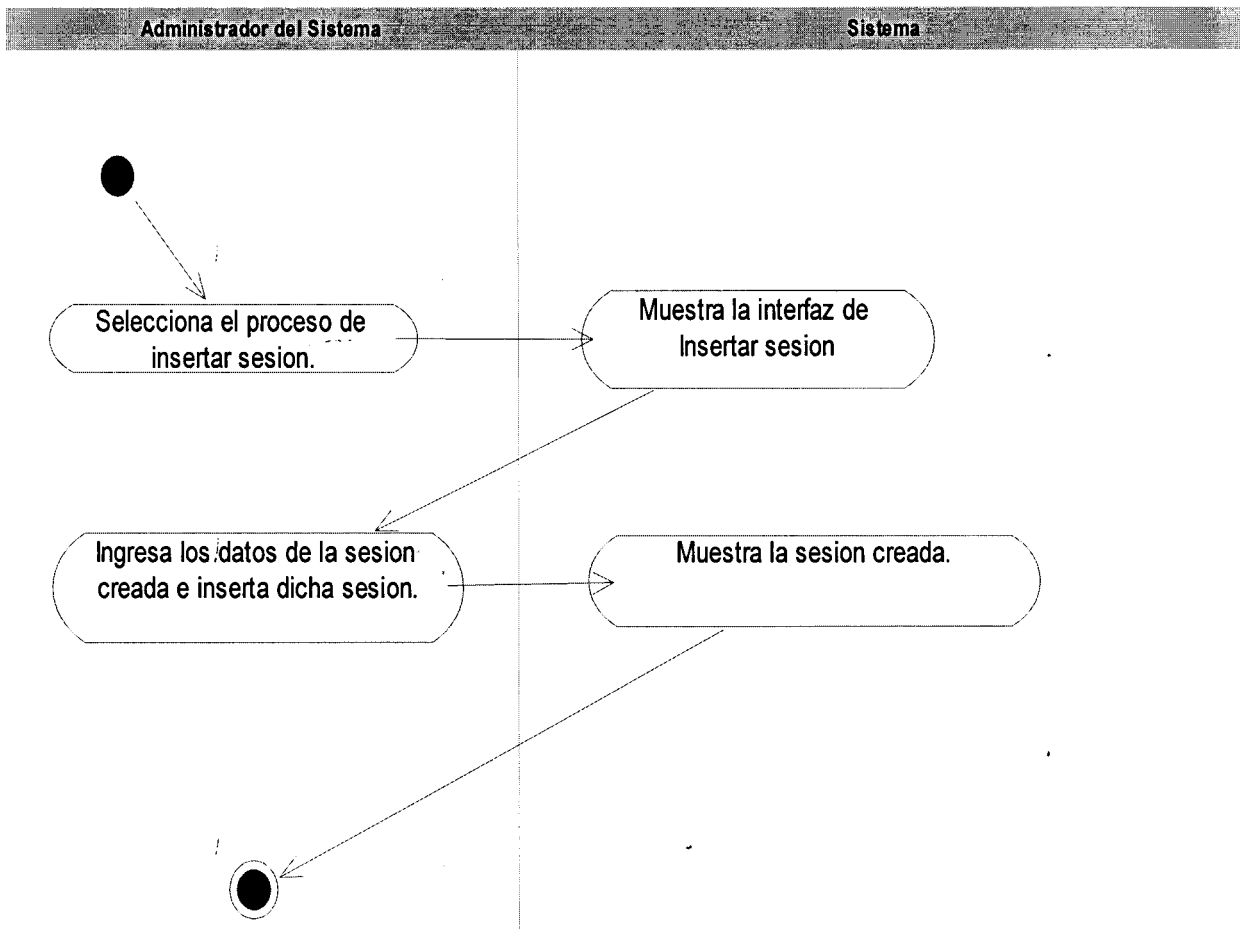
3.5.4 - Diagrama De Actividades caso de uso Crear nuevo Comedor.



3.5.5 - Diagrama De Actividades caso de uso Crear nueva Puerta.



3.5.6 - Diagrama De Actividades caso de uso Insertar Sesión.



3.6 - Diagrama De Secuencia.

3.6.1 - Diagrama De Secuencia Insertar Barcode.

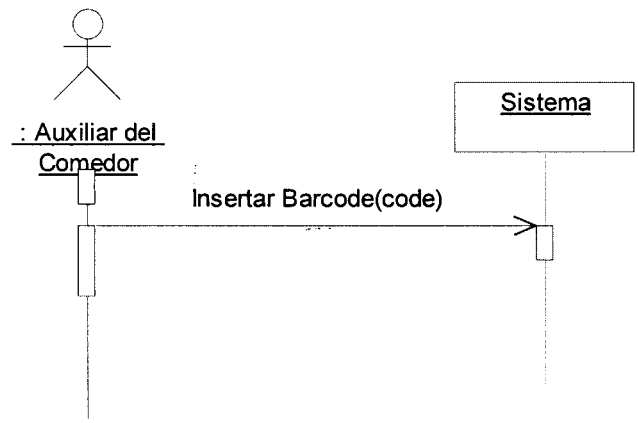
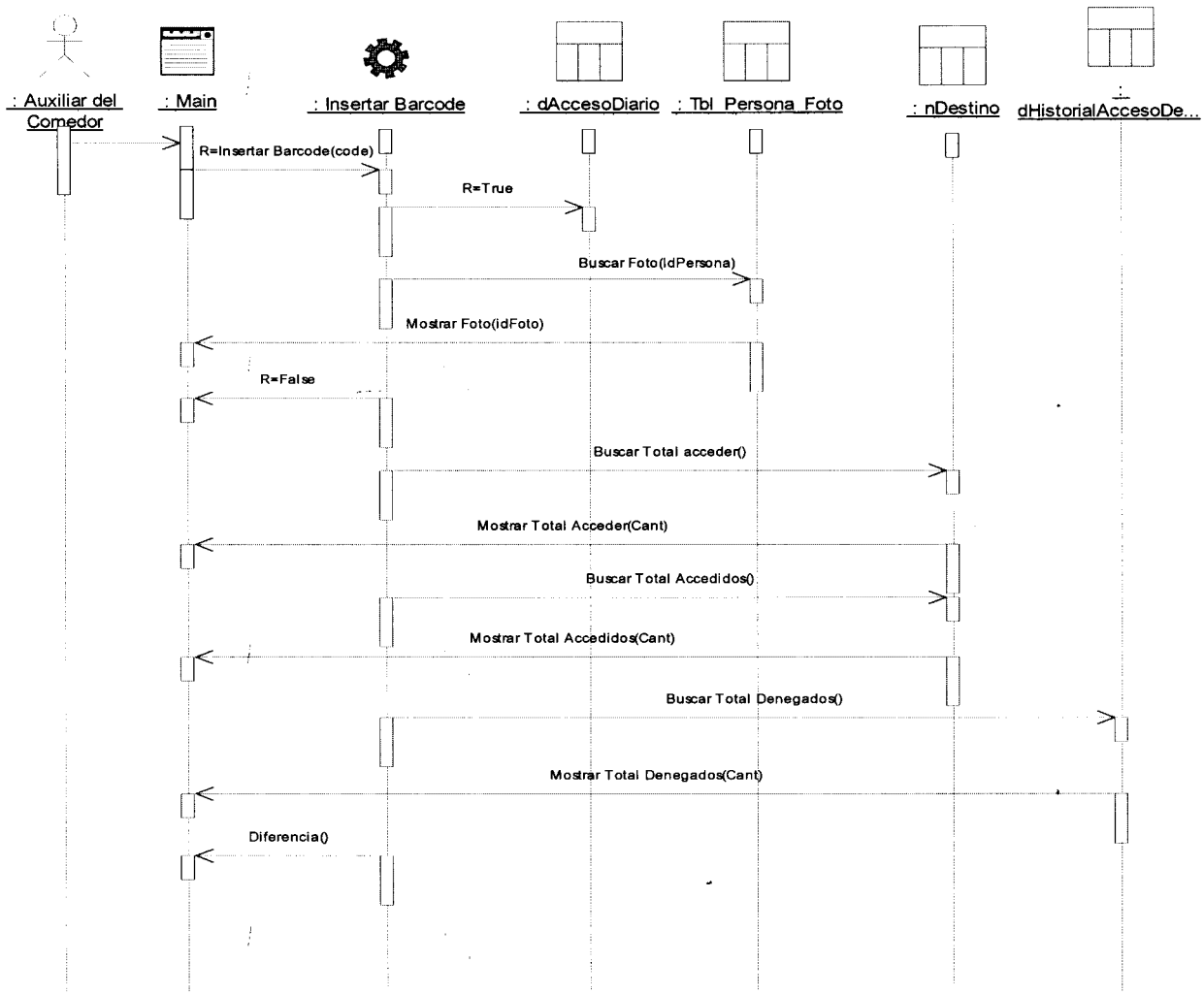


Diagrama De Secuencia Insertar Barcode Generalizado.



3.6.2 - Diagrama De Secuencia Autenticar Administrador.

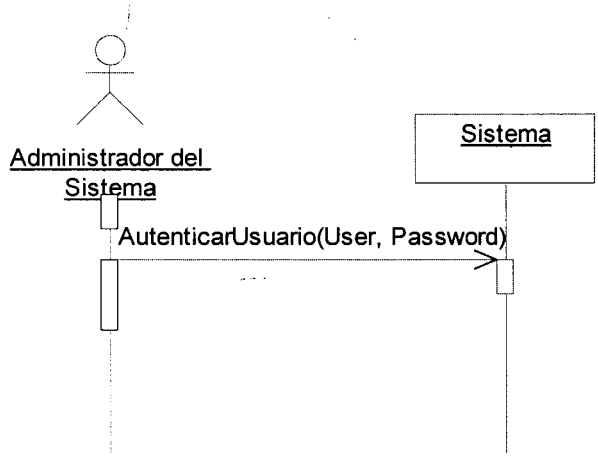
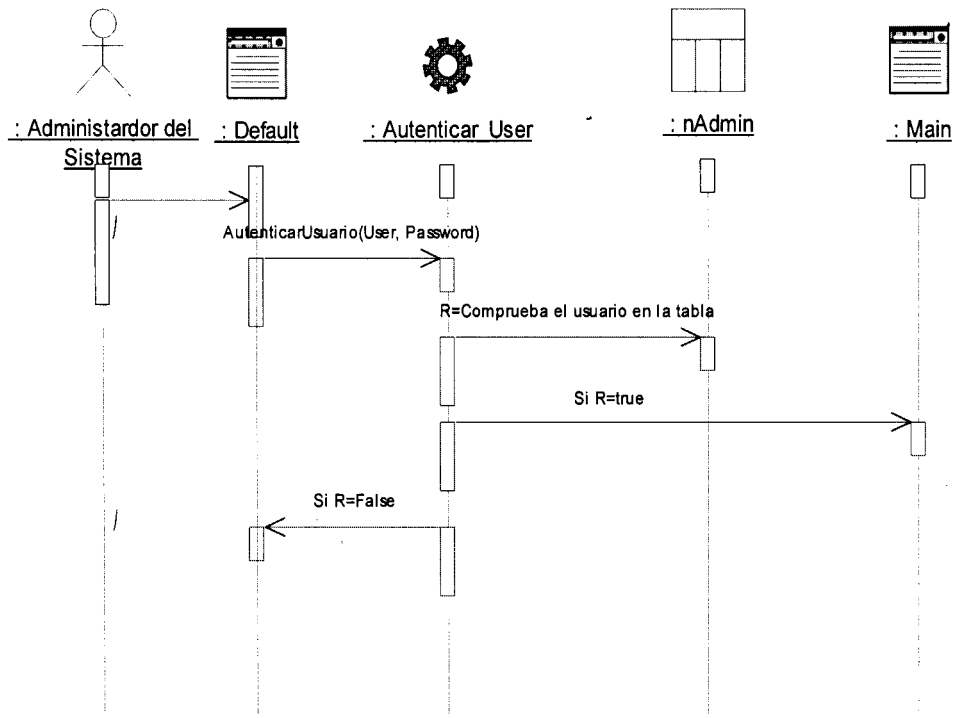


Diagrama De Secuencia Insertar Autenticar Administrador Generalizado.



3.6.3 - Diagrama De Secuencia Crear Nuevo Complejo.

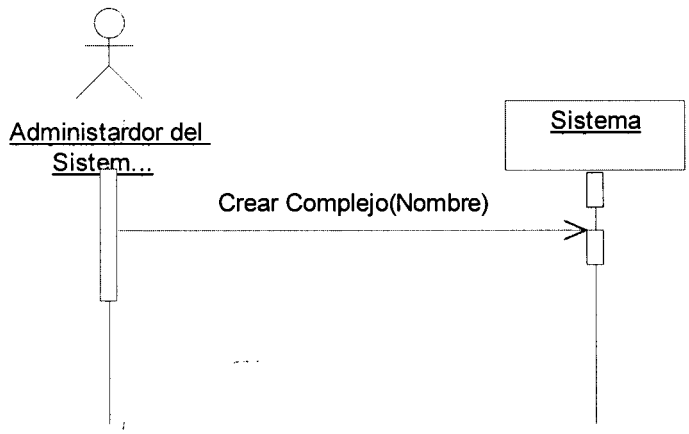
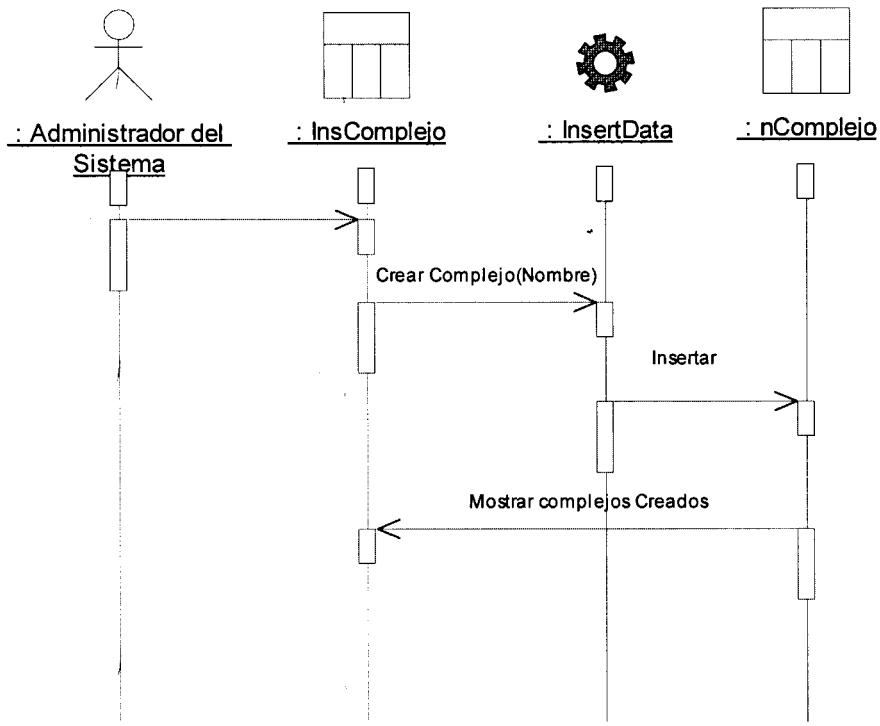


Diagrama De Secuencia Insertar Crear Nuevo complejo Generalizado.



3.6.4 - Diagrama De Secuencia Crear Nuevo Comedor.

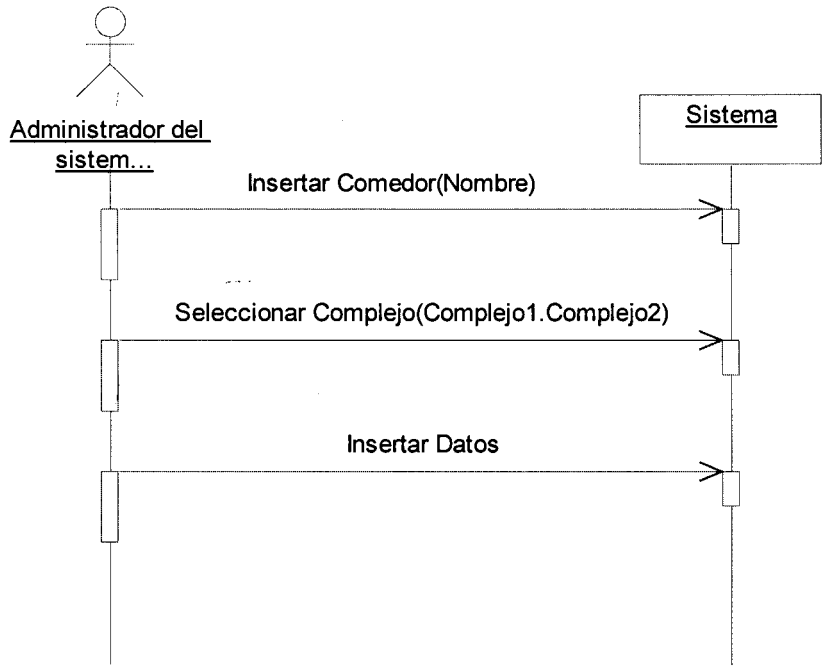
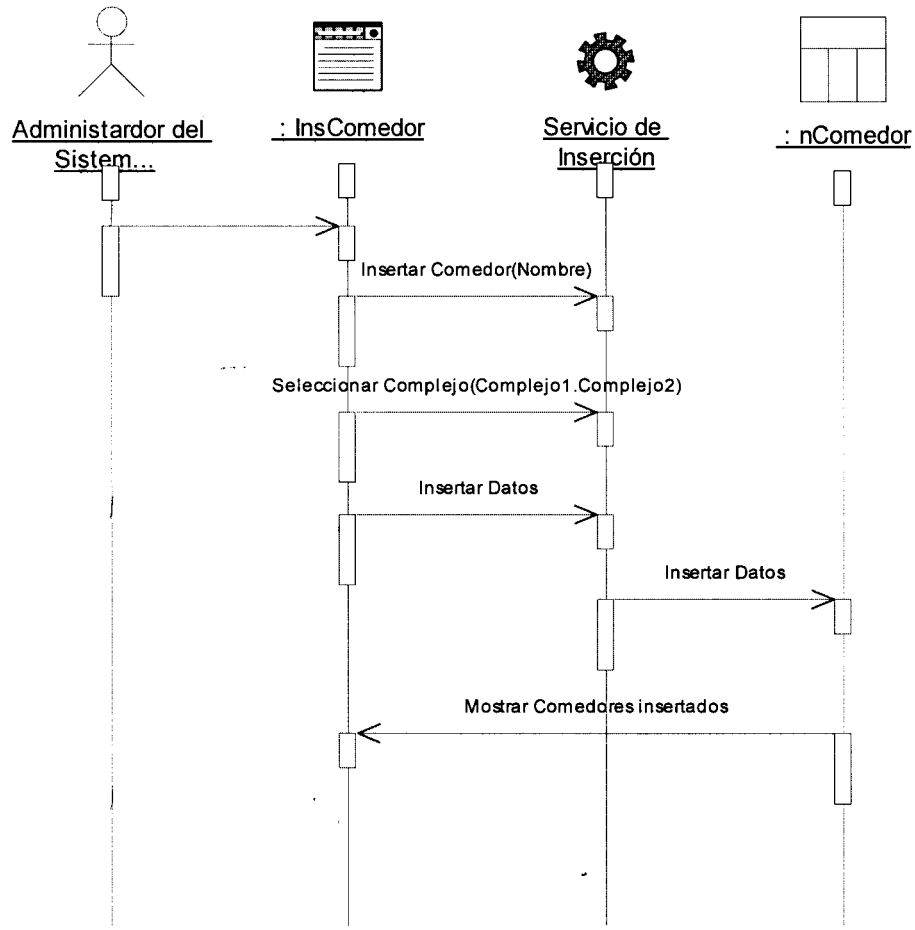


Diagrama De Secuencia Insertar Crear Nuevo Comedor Generalizado.



3.6.5 - Diagrama De Secuencia Crear Nueva Puerta.

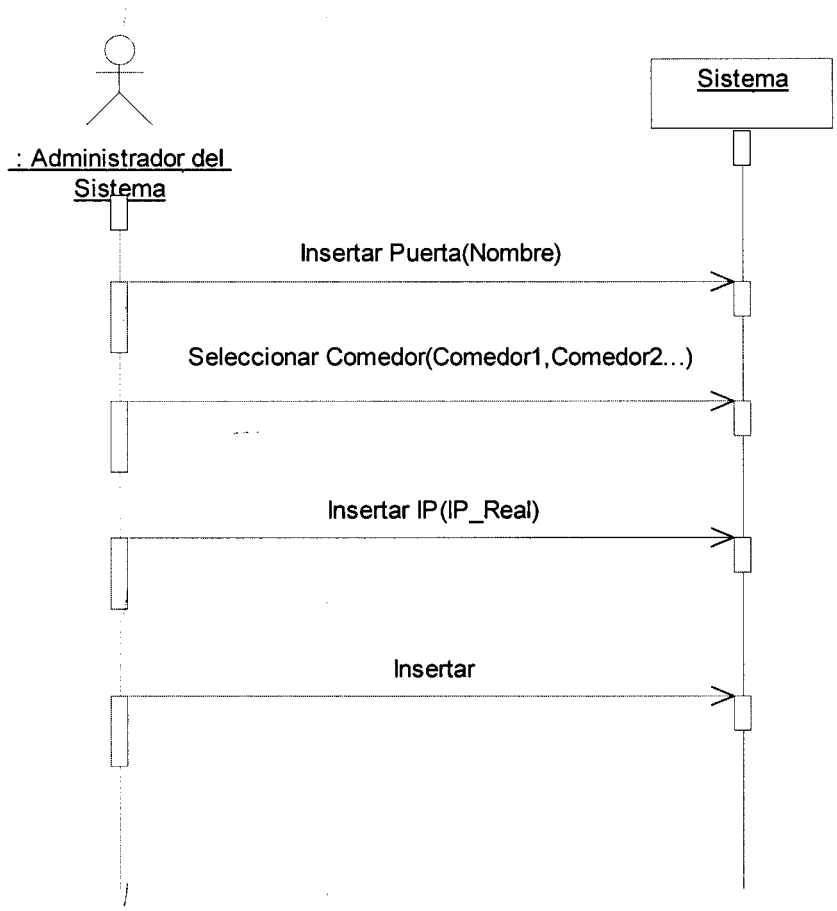
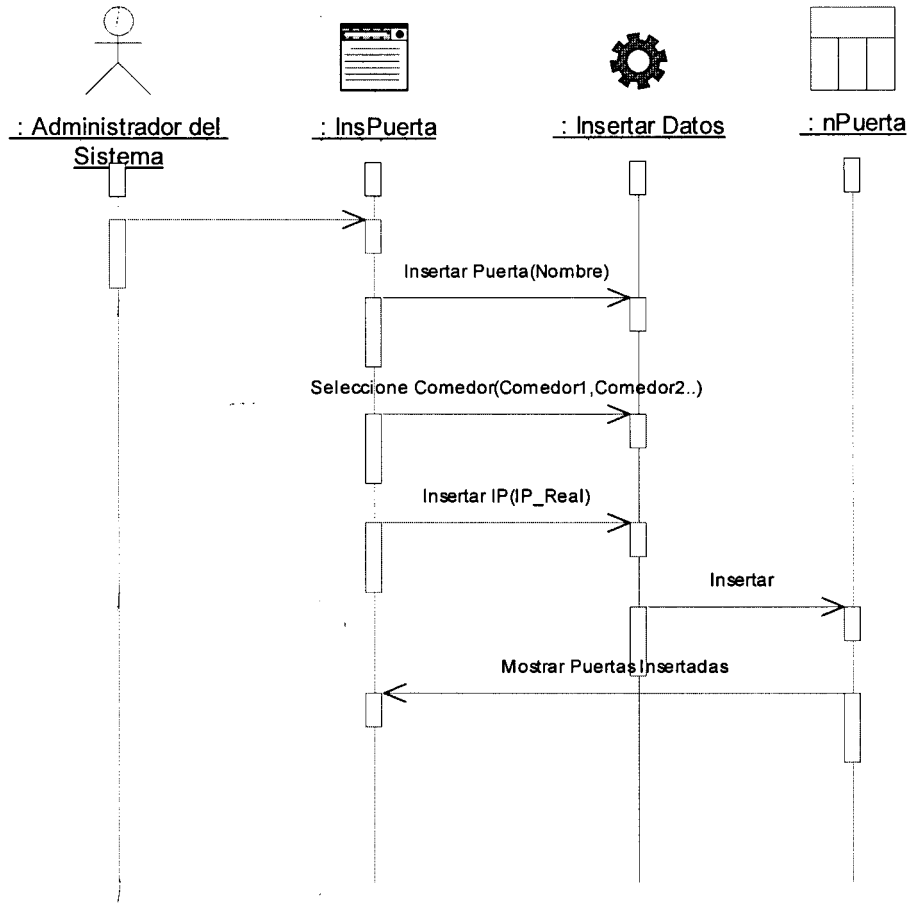


Diagrama De Secuencia Insertar Crear Nueva Puerta Generalizado.



3.6.6 - Diagrama De Secuencia insertar Sesión.

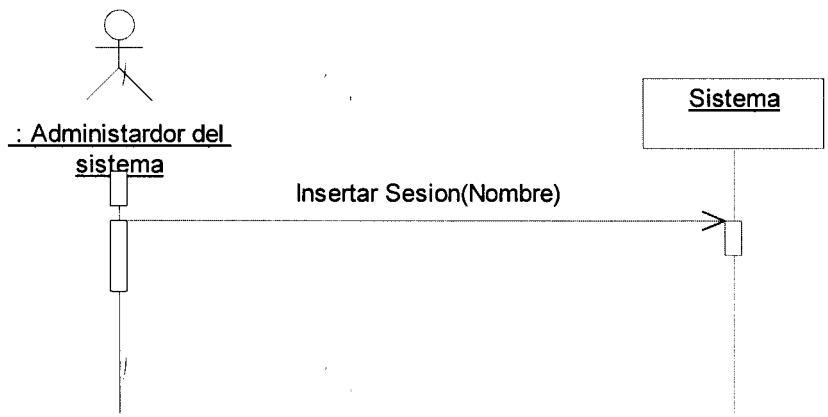
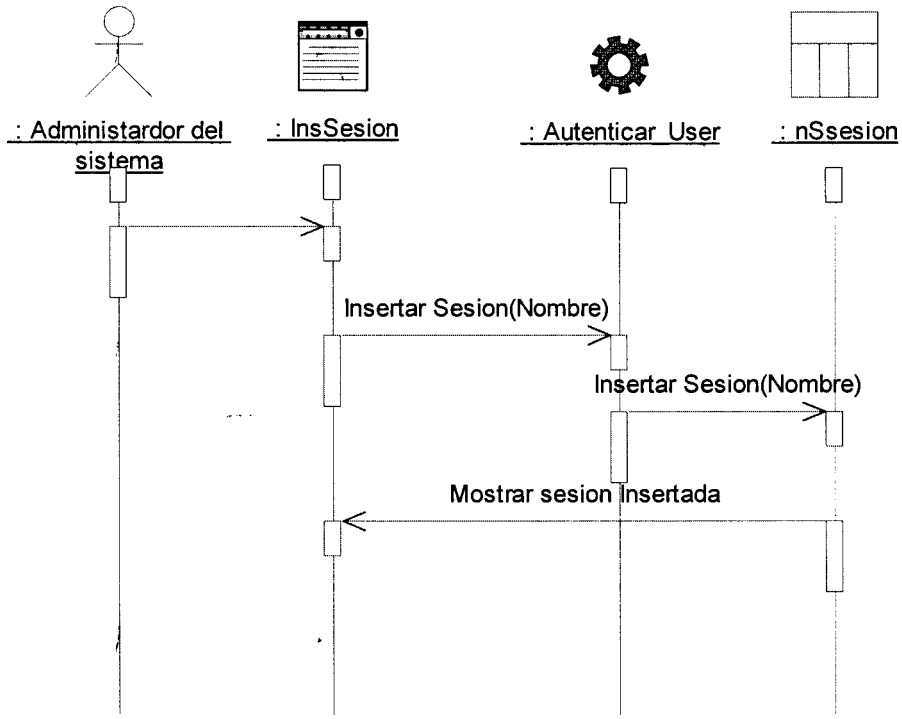
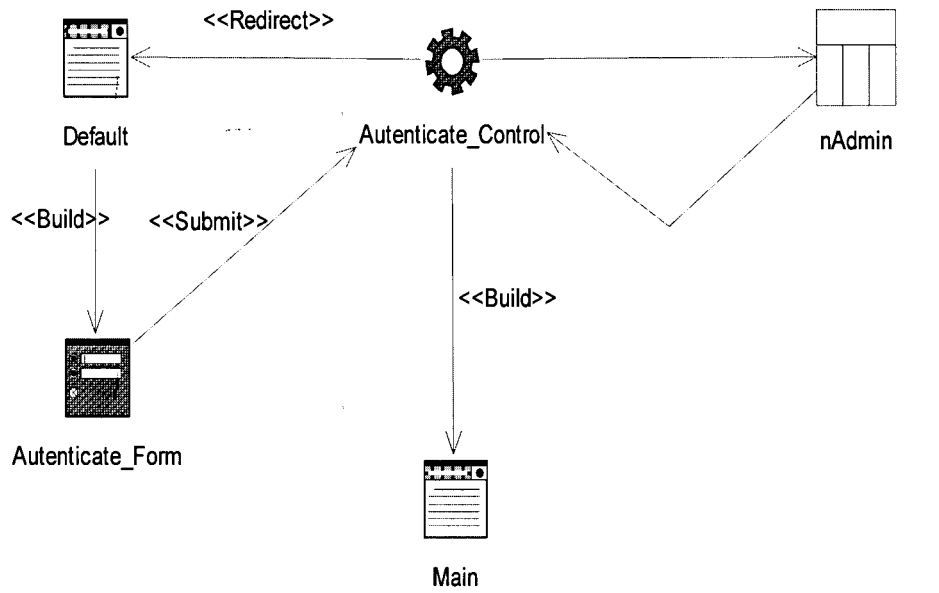


Diagrama De Secuencia Insertar Insertar Sesión Generalizado.

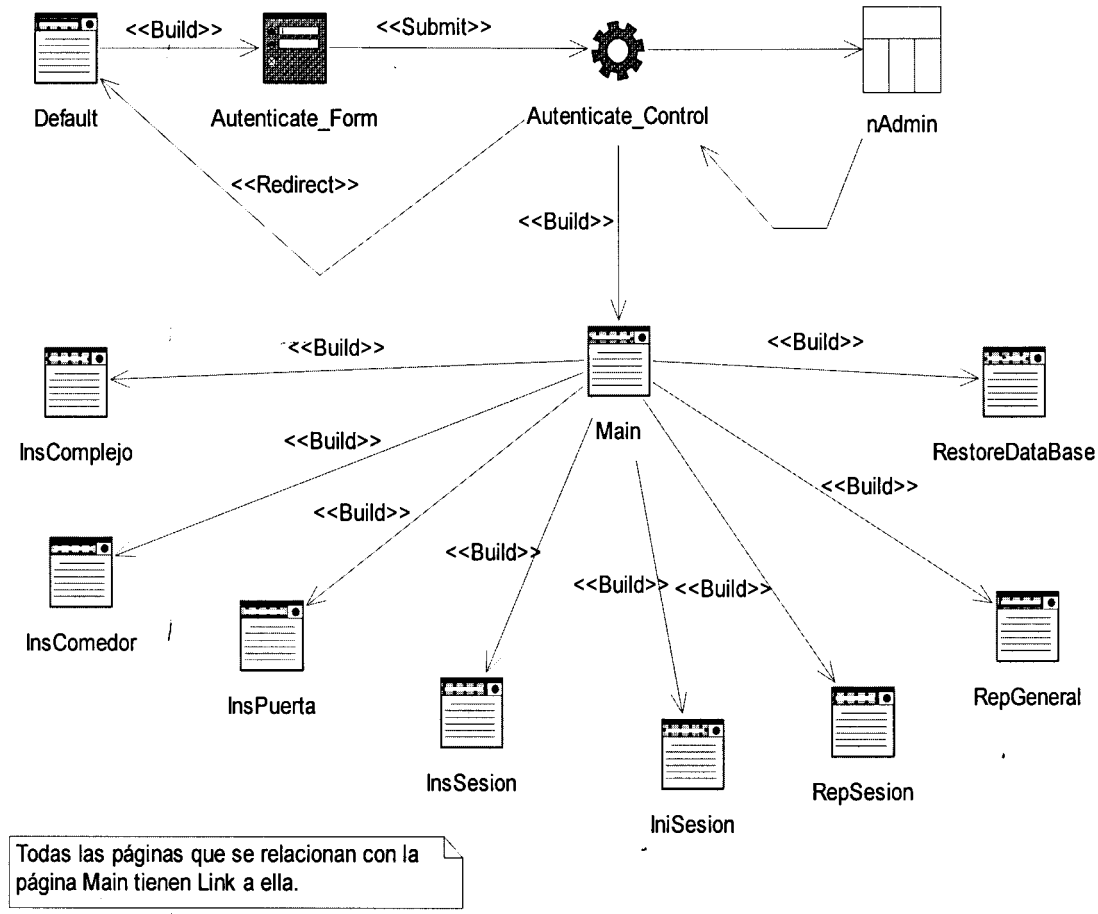


3.7 - Diagrama de Clases Diseño Web.

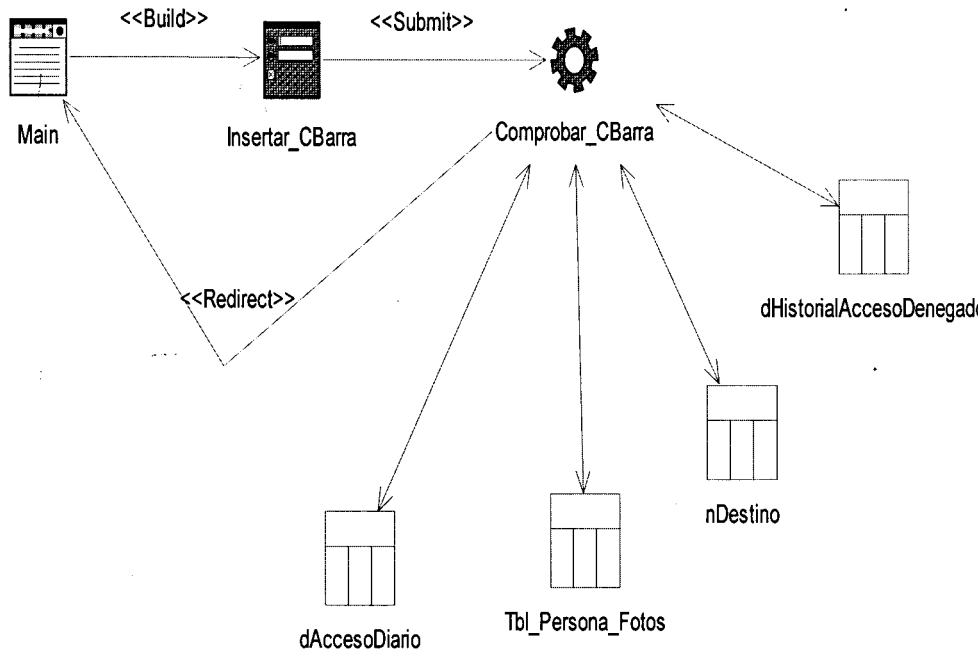
3.7.1 - Diagrama de Clases Diseño Web: Modulo Usuario Generalizado.



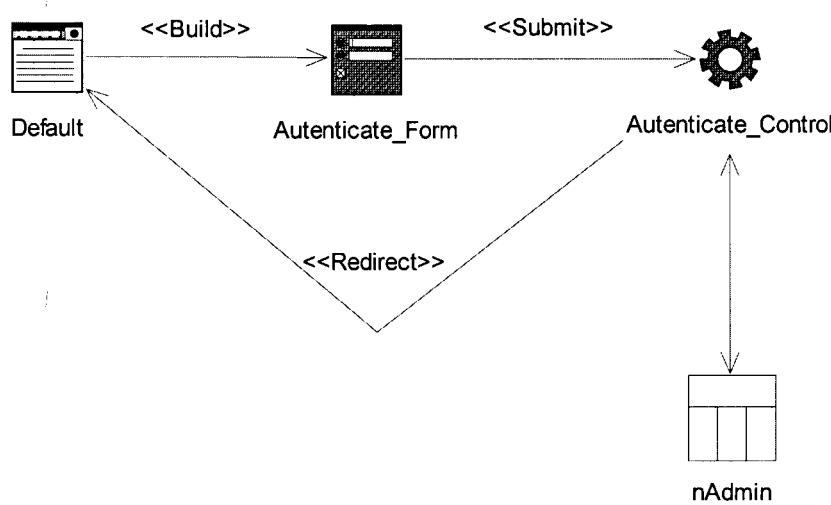
3.7.2 - Diagrama de Clases Diseño Web: Modulo Administración Generalizado.



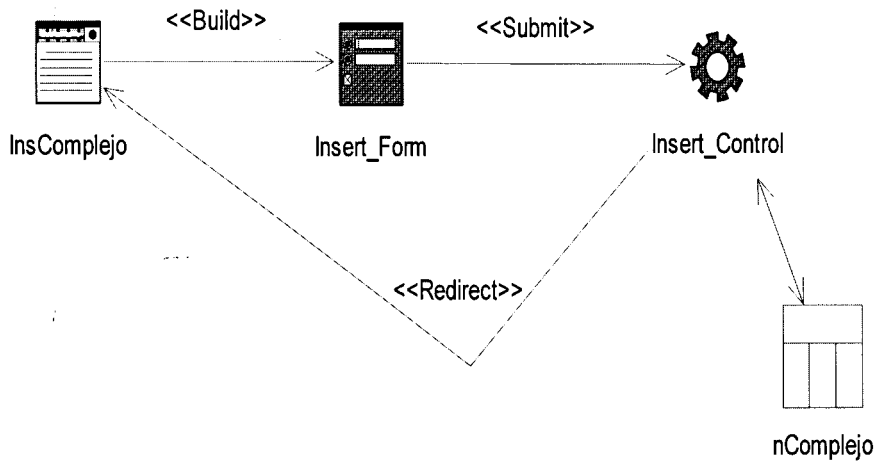
3.7.3 – Diagrama de Clases Diseño Web: Autenticar Insertar Barcode.



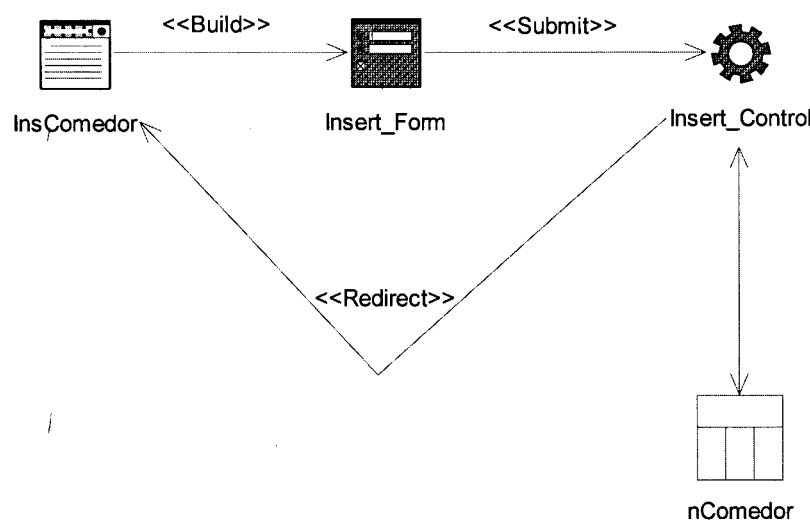
3.7.4 – Diagrama de Clases Diseño Web: Autenticar Administrador.



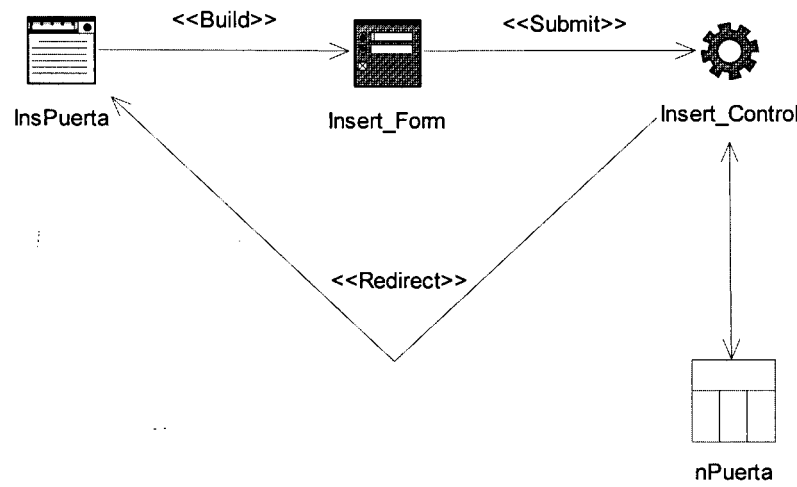
3.7.5 - Diagrama de Clases Diseño Web: Insertar Complejo.



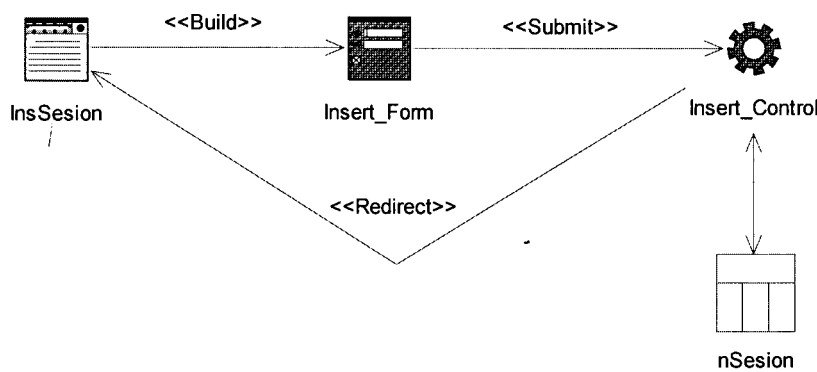
3.7.6 - Diagrama de Clases Diseño Web: Insertar Comedor.



3.7.7 - Diagrama de Clases Diseño Web: Insertar Puerta.

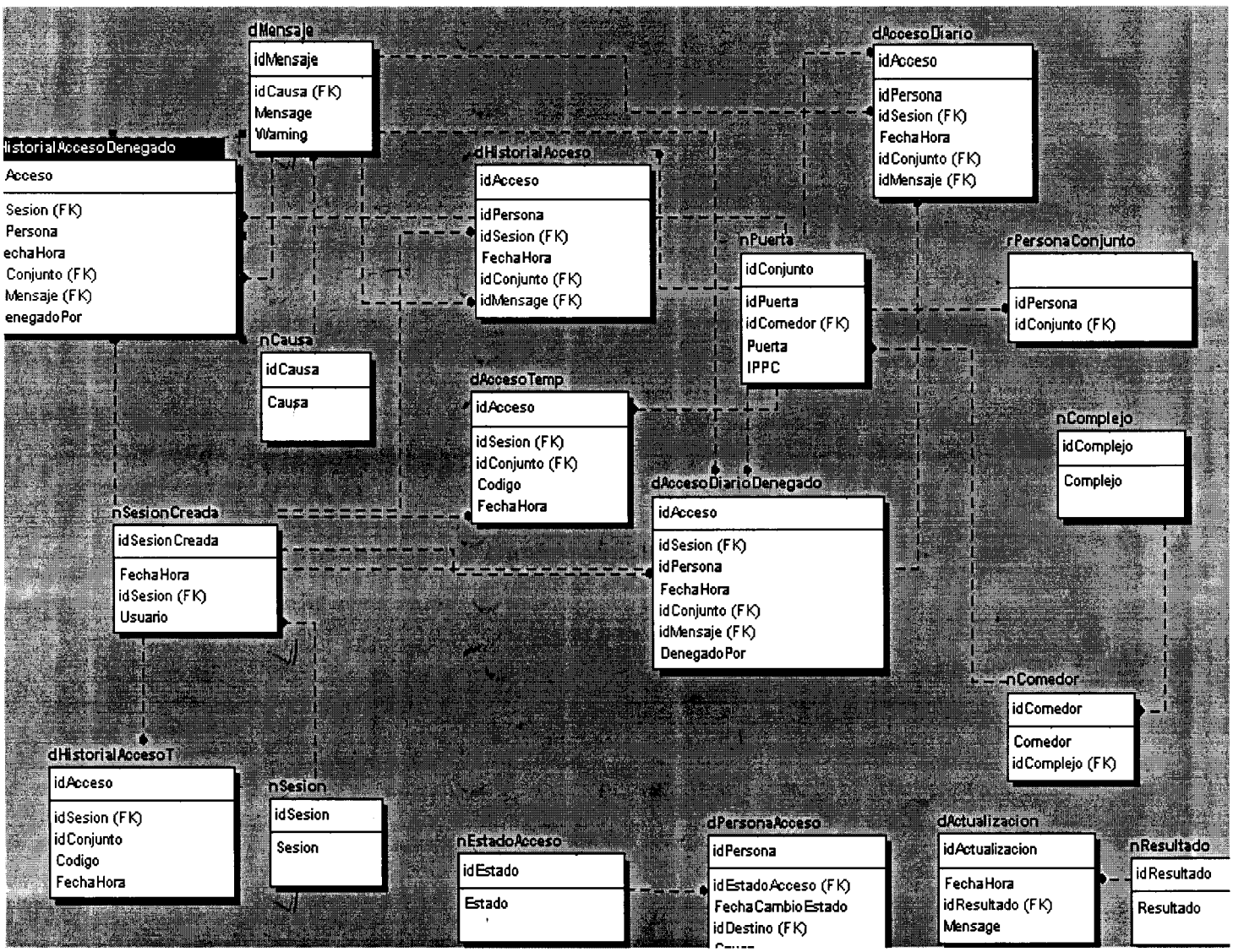


3.7.8 - Diagrama de Clases Diseño Web: Insertar Sesión.



3.8 - Diseño Base Datos.

3.8.1 - Diagrama De La Base de Datos.



3.8.2 – Descripción de las tablas de la Base Datos.

Nombre: dPersonaAcceso		
Descripción: Contiene los datos referentes a la persona que esta siendo procesada.		
Atributo	**Tipo**	**Descripción**

idPersona	Nvarchar 36	Contiene el id de la persona.
idEstadoAcceso	Int 4	Contienen id del estado de acceso.
FechaCambioEstado	datetime 8	Contiene la fecha delcambio de estado.
idDestino	nvarchar 36	Contiene el id del destino.
Causa	nvarchar 50	Contiene la causa.

Nombre: nCausa		
Descripción: Contiene la causa por la que fue accedido la persona.		
Atributo	**Tipo**	**Descripción**
idCausa	Int 4	Contiene el id de la causa.
Causa	nvarchar 50	Contiene la causa.

Nombre:ncomedor		
Descripción: Contiene los datos referentes al comedor.		
Atributo	**Tipo**	**Descripción**
idComedor	Int 4	Contiene el id del comedor.
Comedor	nvarchar 36	Contiene el comedor.

Nombre:ncomplejo		
Descripción: Contiene los datos referentes al complejo.		
Atributo	**Tipo**	**Descripción**
idComplejo	Int 4	Contiene el id del comedor.
Complejo	nvarchar 36	Contiene el complejo.

Nombre:nPuerta

Descripción: Contiene los datos referentes a la puerta.		
Atributo	**Tipo**	**Descripción**
idConjunto	Int 4	Contiene el id del conjunto.
idPuerta	Iqueidentifier 16	Contiene el id de la puerta.
idComedor	Int 4	Contiene el id del comedor.
Puerta	Nvarchar 36	Contiene la puerta.
IPPC	Nvarchar 12	Contiene el ip de la PC.

Nombre:nSesion		
Descripción: Contiene los datos referentes a la sesión.		
Atributo	**Tipo**	**Descripción**
idSesion	Int 4	Contiene el id del la sesión.
Sesión	Nvarchar 50	Contiene la sesión.

Nombre:nSesioncreada		
Descripción: Contiene los datos referentes a la sesión creada.		
Atributo	**Tipo**	**Descripción**
idSesionCreada	Int 4	Contiene el id del la sesión creada.
FechaHora	Datetime 8	Contiene la fecha de creación.
idSesion	int 4	Contiene el id de la sesión.
Usuario	Nvarchar 50	Contiene el usuario.

Nombre: dAccesoDiarioDenegado		
Descripción: Contiene los datos referentes a la acceso diario denegado.		
Atributo	**Tipo**	**Descripción**
idAcceso	uniqueidentifier 16	Contiene el id del acceso.
idSesion	int 4	Contiene el id de la sesión.
idPersona	nvarchar 36	Contiene el id de la persona.

FechaHora	datetime 8	Contiene la fecha denegada.
idConjunto	int 4	Contiene el id del conjunto.
idMensaje	int 4	Contiene el id del mensaje.
DenegadoPor	nvarchar 36	Contiene por quien fue denegado.

Nombre: dAccesoTemp		
Descripción: Contiene los datos referentes a la acceso temporales.		
Atributo	**Tipo**	**Descripción**
Id Acceso	uniqueidentifier 16	Contiene el id del acceso.
idSesion	int 4	Contiene el id de la sesión.
idConjunto	int 4	Contiene el id del conjunto.
Código	nvarchar 50	Contiene el código.
FechaHora	datetime 8	Contiene la fecha del acceso.

Nombre: dActualizacion		
Descripción: Contiene los datos referentes a la actualización.		
Atributo	**Tipo**	**Descripción**
idActualizacion	int 4	Contiene el id de la actualización.
FechaHora	datetime 8	Contiene la fecha de la actualización.
idResultado	Int 4	Contiene el id del resultado.
Mensage	Nvarchar 255	Contiene el mensaje.

Nombre: dHostorialAcceso		
Descripción: Contiene los datos referentes al historial de acceso.		
Atributo	**Tipo**	**Descripción**
idAcceso	Uniqueidentifier 16	Contiene el id de acceso.
idPersona	Nvarchar 36	Contiene el id de la persona.
idSesion	Int 4	Contiene el id de la sesión.

FechaHora	Datetime 8	Contiene la fecha del acceso.
idConjunto	Int 4	Contiene el id del conjunto.
idMensage	Int 4	Contiene el id del mensaje.

Nombre: dHostorialAccesodenegadoPor		
Descripción: Contiene los datos referentes al historial de acceso que ha sido denegado.		
Atributo	**Tipo**	**Descripción**
idAcceso	Uniqueidentifier 16	Contiene el id de acceso.
idPersona	Nvarchar 36	Contiene el id de la persona.
idSesion	Int 4	Contiene el id de la cesión.
FechaHora	Datetime 8	Contiene la fecha del acceso.
idConjunto	Int 4	Contiene el id del conjunto.
idMensage	Int 4	Contiene el id del mensaje.
DenegadoPor	nvarchar 36	Contiene el usuario que deniega el acceso al historial.

Nombre: dHistorialAccesoT		
Descripción: Contiene los datos referentes al historial de acceso temporal.		
Atributo	**Tipo**	**Descripción**
idAcceso	Nvarchar 36	Contiene el id de acceso.
idSesion	Int 4	Contiene el id de la sesión.
FechaHora	Datetime 8	Contiene la fecha del acceso.

idConjunto	Int 4	Contiene el id del conjunto.
Código	nvarchar 50	Contiene el código.

Nombre: dmensaje		
Descripción: Contiene los datos referentes al historial de acceso temporal.		
Atributo	**Tipo**	**Descripción**
idMensaje	Int 4	Contiene el id del mensaje.
idCausa	Int 4	Contiene el id de la causa.
Mensage	Nvarchar 100	Contiene el mensaje.
Warning	Nvarchar 50	Contiene el warning.

3.9 - Principios Generales de diseño.

La interfaz ha de ser lo más uniforme posible utilizando un mismo sistema de colores , con modelación consistente y razonable , tratando siempre de utilizar colores similares en todas las paginas , con textos claros y sin mezclar muchos tipos de letras y tamaños en cada una , la navegación a través del sitio debe ser amigable para lograr así que el usuario final se sienta cómodo cuando esté en el sitio.

3.10 - Forma general del tratamiento de errores.

El sistema trata de minimizar al máximo los posibles errores que puedan existir. Para ello se decidió, a la hora de realizar operaciones de modificación o eliminación de elementos, mostrar los mismos en una lista desde donde el usuario pueda seleccionar los mismos. De esta manera siempre serán válidos los datos de entrada.

3.11 - Forma general y principios de la protección y seguridad.

El sistema analizado debe tener determinado nivel de seguridad ya que cualquier usuario que intente ingresar a cualquier página de administración antes de pasar por la de autenticado lo redireccione a la misma.

3.12 – Conclusiones.

El presente capítulo muestra los resultados del análisis y diseño del sistema, en este se desarrollan los diagramas de actividades, secuencias, clases WEB, se realizó la definición del tratamiento de errores y el sistema de seguridad.

Conclusiones

Una vez realizado el estudio de los diferentes procesos que se han llevado a cabo para el acceso al comedor se ha podido arribar a las siguientes conclusiones:

Se planteó un sitio Web que permite de forma fácil y rápida el control del acceso al comedor en la UCI.

Se realizó el estudio de cómo se llevan a cabo los procesos por parte de la dirección de alimentación para poder definir y especificar las funcionalidades que el sistema contempla.

Para la implementación, la herramienta que se utilizó fue el ASP.net C#, como gestor de base de datos se utilizo SQL Server y para la modelación se utilizó RUP.

Se realizó el análisis y diseño para el primer ciclo de desarrollo.

Recomendaciones

Se recomienda:

Continuar con el cumplimiento de los casos de uso de los ciclos 2 de desarrollo del sistema.

Que se desarrollen, ensamblen los demás módulos del sistema de área de alimentación.

Referencias Bibliográficas

[1] **eTime** Manual www.austincc.edu/hr/eTime/**etime**.htm (*25/9/2002*)

[2] The World Wide Web Consortium. http://www.w3.org/ (03/03/2004)

[3] Web services. http://web-services.bankhacker.com (04/03/2004)

[4] The Official Microsoft ASP.NET Site. http://www.asp.net (19/05/2004)

[5] Universidad de las Ciencias Informáticas. Proyección estratégica de la UCI. 2002

[6] Larman, Craig. UML y Patrones. Introducción al análisis y diseño orientado a objetos. Prentice Hall Hispanoamérica, México, 1999.

[7] Jacobson, Ivar y Booch, Grady y Rumbaugh, James. El proceso unificado de software. Primera edición. Pearson Educación, S.A. 2000.

[8] SQL server2000 libros de pantalla actualizado 2003

Bibliografía

Sistema Control de Asistencia de trabajadores Informe Técnico para optar por el título de INGENIERO INFORMÁTICO Autor: Yosvani Umpierre Martínez. Tutora: Lic. Lida Gonzalez Alvarez.

Universidad de Oriente, Facultad de Matemática y computación, Departamento de computación, Sistema Gestión del alojamiento, En la residencia. UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS: INFORME TÉCNICO PARA OPTAR POR EL TÍTULO DE LICENCIADO EN CIENCIA DE LA COMPUTACIÓN. Autor: Manuel Enrique Puebla Martínez Tutor: Ariadna Falcón López.

Jacobson, Ivar y Booch, Grady y Rumbaugh, James. El proceso unificado de software. Primera edición. Pearson Educación, S.A. 2000.

Anexo 1

A.1.0 - Actores del negocio.

Actores del negocio	Justificación
Estudiante Trabajadores Profesores Personal de Seguridad.	Son los que inician la acción estando interesados en acceder al comedor.

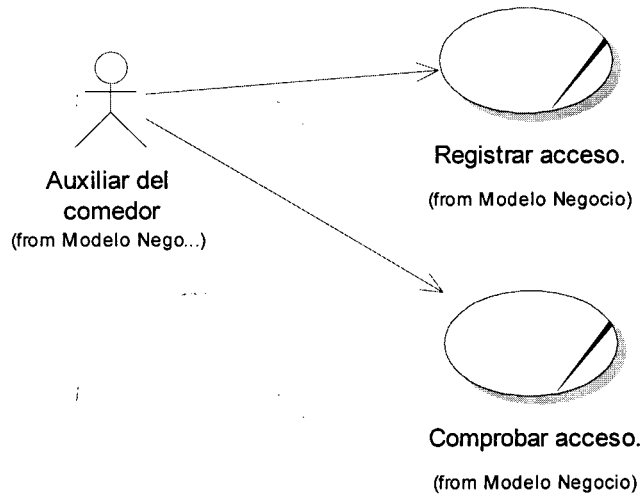
Trabajadores del negocio	Justificación
Auxiliar del comedor	Es la encargada de introducir los datos del solapin que le entrega el estudiante y comprobar que todo lo referente al mismo. En caso necesario elimina acceso.

Figura 5: Especificación de los Trabajadores del Negocio.

A.1.1 – Casos de Uso del Negocio.

1) Registrar acceso.
2) Comprobar acceso.

A.1.2 - Diagrama de casos de uso de negocio.



A.1.4 - Descripción de los procesos del negocio mediante los Diagramas de Actividades.

Diagrama de actividades Registrar Acceso.
Caso de uso: Registrar Acceso.

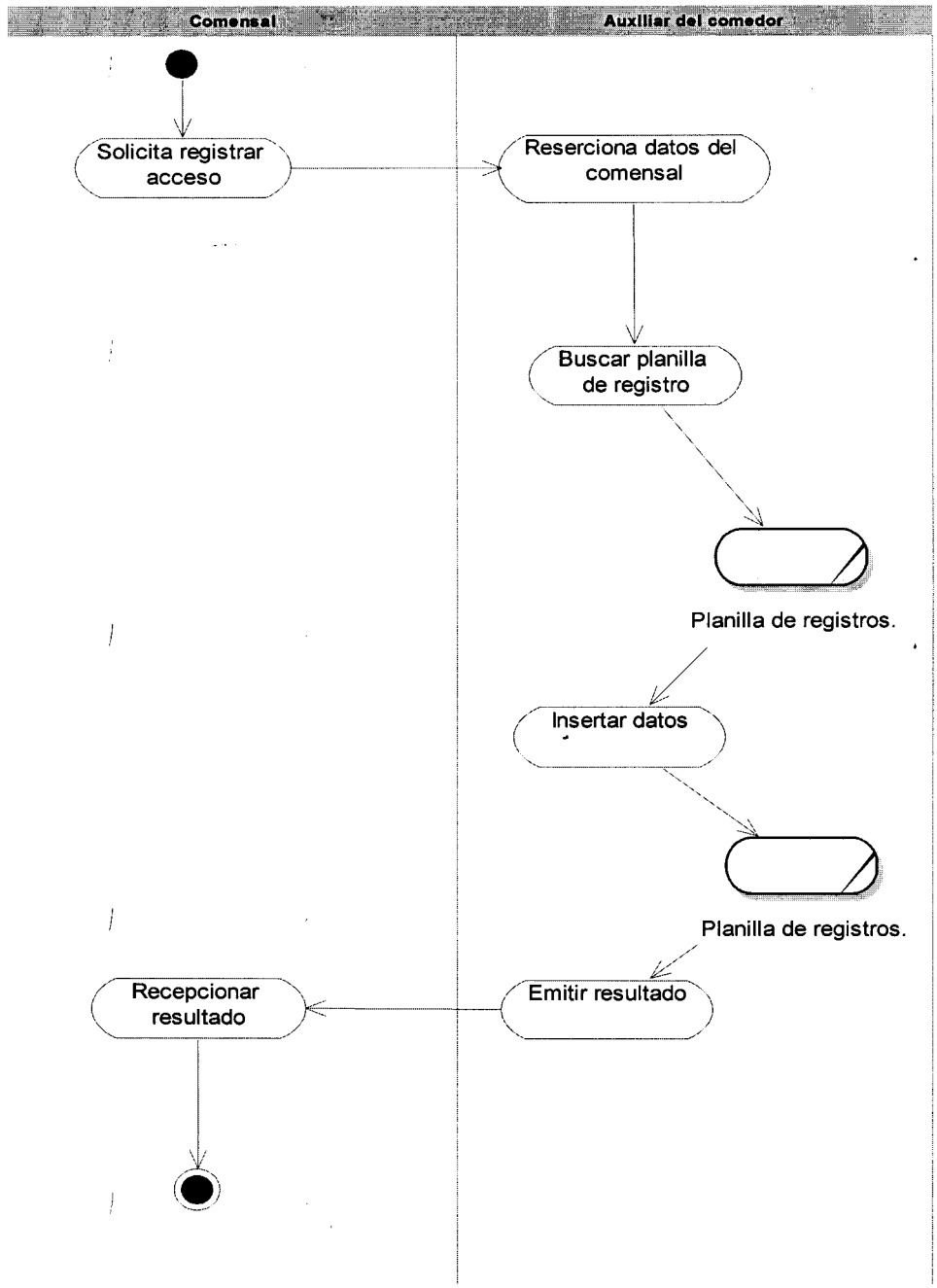
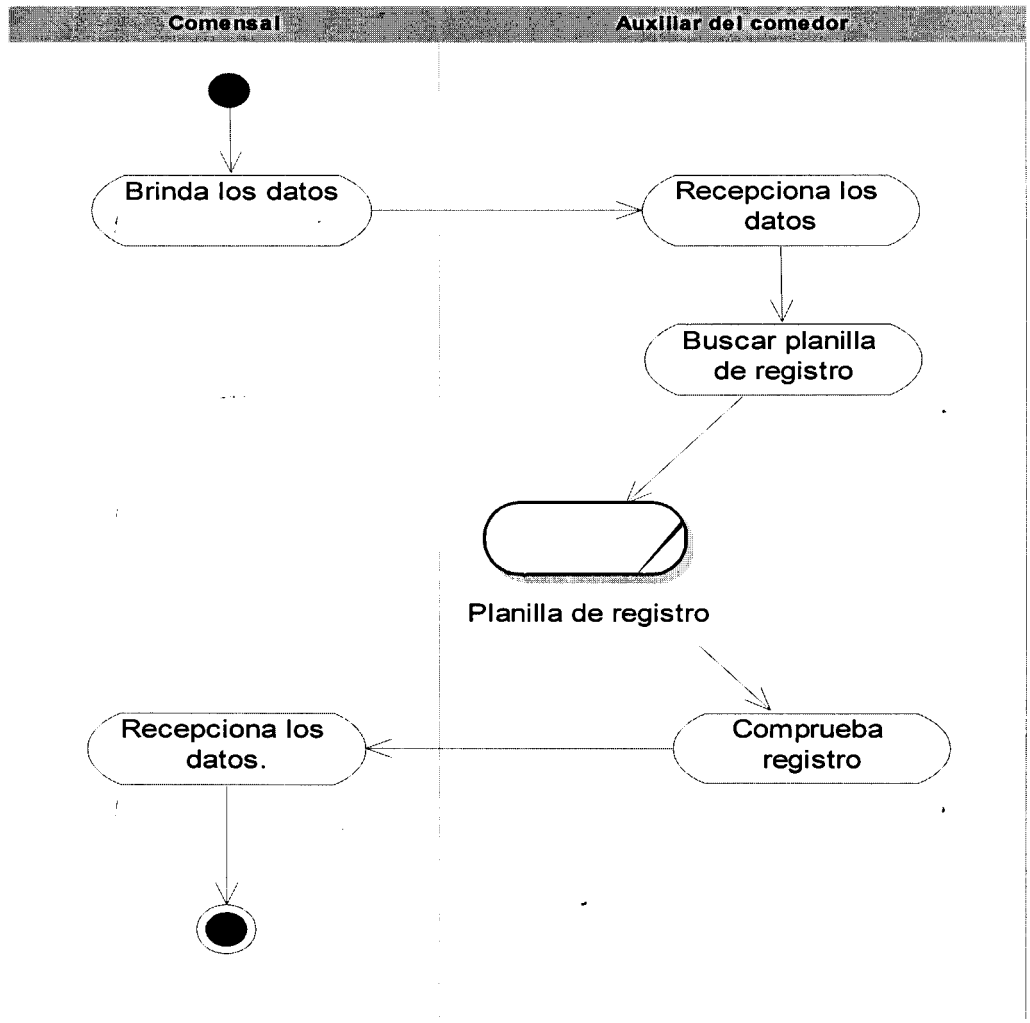
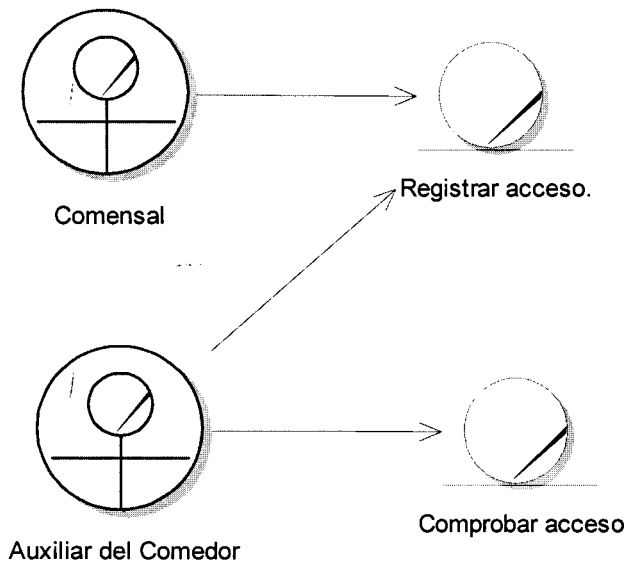


Diagrama de actividades Comprobar Acceso.
Caso de uso: Comprobar Acceso.



A.1.5 - Diagrama de clases Modelo Objeto.



A.1.6 - Especificación textual de los casos de usos del negocio:

Registrar Acceso.

Nombre del caso de uso del negocio:	Registrar Acceso
Actores del negocio:	Comensal, auxiliar del comedor
Propósito:	Que todos los comensales sean registrados en el registro de acceso
Resumen: El caso de uso se inicia cuando el comensal, en este caso los trabajadores, profesores, estudiantes, etc. Se dirigen al área del comedor para solicitar su acceso, el comensal brinda sus datos y la auxiliar del comedor los recepciona, a su vez este lo inserta en la registro de acceso y da autorización de acceder al servicio solicitado.	
Flujo de trabajo **Acción del actor**	**Respuesta del negocio**

1) El comensal se presenta ante la auxiliar del comedor y le solicita que su acceso al comedor. 4) El comensal recepciona la información de que su acceso ha sido insertado en el registro de acceso.	2) Se recepciona la solicitud, por parte de la auxiliar del comedor. 3) Se inserta los datos en el registro de acceso.
Prioridad:	
Mejoras:	
Cursos alternos: No se puede insertar los datos en el registro hasta que no se compruebe que el comensal no ha accedido al comedor.	

Nombre del caso de uso del negocio:	Comprobar Acceso
Actores del negocio:	Comensal, auxiliar del comedor
Propósito:	Que todos los comensales no accedan más de una vez al servicio solicitado.
Resumen: El caso de uso se inicia cuando el comensal, en este caso los trabajadores, profesores, estudiantes, etc. Se dirigen al área del comedor para solicitar su acceso, el comensal brinda sus datos y la auxiliar del comedor los recepciona, a su vez este lo inserta en la registro de acceso y revisa que el comensal no acceda mas de una vez al comedor.	
Flujo de trabajo **Acción del actor**	**Respuesta del negocio**

1) El comensal se presenta ante la auxiliar del comedor y le solicita que su acceso al comedor. 5) El comensal recepciona la información de que su acceso ha sido insertado en el registro de acceso.	2) Se recepciona la solicitud, por parte de la auxiliar del comedor. 3) Se inserta los datos en el registro de acceso. 4) Se revisa en el registro de acceso que el comensal no halla echo uso del servicio mas de una vez.
Prioridad:	
Mejoras:	
Cursos alternos: Si el comensal ya hizo uso del servicio devolverle sus datos y restringirle el acceso al comedor.	

Anexo 2

Descripción Casos de uso del módulo Usuario:

Caso de Uso	**Autenticar Usuario**
Actores	Auxiliar del comedor
Propósito	Ingresar al módulo de usuario.
Resumen: El caso de uso se inicia cuando la auxiliar del comedor se presenta en la puerta de los comedores de la UCI y se autentica. Una vez autenticada la auxiliar del comedor se procede a la toma de solapines de los estudiantes que les permitirá ingresar al comedor.	
Acción del actor	**Respuesta del Sistema**
1- La auxiliar del comedor se presenta en la puerta de los comedores de la UCI y se autentica.	2- El sistema determina si el usuario registrado tiene acceso al servicio solicitado.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
Si el usuario no esta autorizado	**El sistema devuelve mensaje de error.**

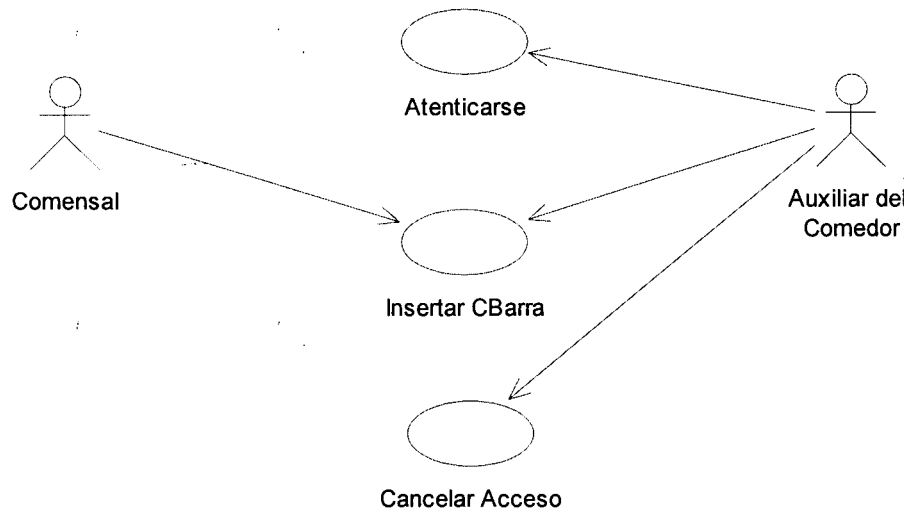
Caso de Uso	**Insertar Barcode**
Actores	Auxiliar del comedor, comensal
Propósito	Ingresar el barcode situado en el solapin del comensal que desea acceder.
Resumen: El caso de uso se inicia cuando el comensal le entrega a la auxiliar del	

comedor el solapín a su vez lo inserta en el barcode del sistema, este reconoce y da información acerca del comensal que esta siendo procesado.	
Acción del actor	**Respuesta del Sistema**
1- La auxiliar del comedor inserta el barcode en el sistema.	2- El sistema determina por el barcode insertado toda la información referente al comensal procesado mostrando la foto del mismo Ali como si es estudiante, profesor, eventual, etc., muestra el total a acceder, muestra el total accedidos, muestra el total denegados y la diferencia existentes entre el total a acceder y los accedidos.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
La auxiliar del comedor inserta el barcode en el sistema.	**El sistema devuelve un mensaje de error saltando automáticamente una alarma y mostrando que el comensal ya fue procesado por otra puerta. Esto ocurre dentro de la misma sesión iniciada, una vez pasado el horario requerido para esa sesión, se restablecen los datos a su valor inicial.**

Caso de Uso	**Cancelar acceso.**
Actores	Auxiliar del comedor
Propósito	Cancelar el acceso al comedor del

	comensal.
Resumen: El caso de uso se inicia cuando la auxiliar del comedor inserta el barcode en el sistema, este reconoce y da información acerca del comensal que esta siendo procesado, si este ya ingreso en esa sesión por otra puerta, la alarma se iniciara y dará la opción de cancelar el acceso al mismo.	
Acción del actor	**Respuesta del Sistema**
1- La auxiliar del comedor inserta el barcode en el sistema.	2- El sistema determina por el barcode insertado toda la información referente al comensal procesado, si este ya ingreso en esa sesión por otra puerta, la alarma se iniciara y dará la opción de cancelar el acceso al mismo.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
La auxiliar del comedor inserta el barcode en el sistema.	**El sistema dara la opción de cancelar el acceso al mismo anulando el acceso del comensal.**

Diagrama de caso de Uso para el módulo Usuario.



Descripción Casos de uso del módulo de Administración:

Caso de Uso	**Autenticación del Administrador**
Actores	Administrador
Propósito	Ingresar al módulo de administración.
Resumen: El caso de uso se inicia cuando administrador del sistema inserta su usuario y password dándole acceso al módulo de administración.	
Acción del actor	**Respuesta del Sistema**

1 El administrador ingresa al módulo de administración introduciendo su usuario y su password.	2- El sistema determina si el usuario registrado tiene acceso al servicio solicitado.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
Si el usuario no está autorizado.	**El sistema muestra un mensaje de error.**

Caso de Uso	**Crear nuevo complejo**
Actores	Administrador
Propósito	Crear un nuevo complejo.
Resumen: El caso de uso se inicia cuando administrador del sistema desea crear un nuevo complejo.	
Acción del actor	**Respuesta del Sistema**
1 El administrador selecciona el proceso crear nuevo complejo.	2- El sistema muestra la interfaz de la creación.
3 El administrador inserta el nuevo complejo y lo crea.	4- El sistema lo adiciona en la zona de complejos existentes, si ya existían algunos con anterioridad lo adiciona con los demás.

Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
----------	**Muestra los complejos creados.**

Caso de Uso	Crear nuevo comedor
Actores	Administrador
Propósito	Crear un nuevo comedor.
Resumen: El caso de uso se inicia cuando administrador del sistema desea crear un nuevo comedor, esta crea el nuevo comedor se lo asigna a un complejo y lo adiciona.	
Acción del actor	**Respuesta del Sistema**
1 El administrador selecciona el proceso crear nuevo comedor. 3 El administrador inserta el nuevo comedor, se lo asigna a un complejo de los que están creados y lo crea.	2- El sistema muestra la interfaz de la creación. 4- El sistema lo adiciona en la zona de comedores existentes, si ya existían algunos con anterioridad lo adiciona con los demás.

Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
----------	**Muestra los comedores creados.**

Caso de Uso	Insertar Puerta

Actores	Administrador
Propósito	Insertar una nueva puerta.
Resumen: El caso de uso se inicia cuando administrador del sistema desea insertar una nueva puerta, este debe poder crear una nueva puerta, asignársela a un comedor ya creado e insertarle el IP de la máquina por donde se va a encontrar dicha puerta.	
Acción del actor	**Respuesta del Sistema**
1 El administrador selecciona el proceso insertar nueva puerta. 3 El administrador inserta la nueva puerta, se le asigna a un comedor de los que están creados e inserta el IP de la máquina por donde se va a encontrar dicha máquina.	2- El sistema muestra la interfaz de la creación. 4- El sistema lo adiciona en la zona de puertas existentes, si ya existían algunos con anterioridad lo adiciona con los demás.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
----------	**Muestra las puertas creadas.**

Caso de Uso	**Insertar Sesión.**
Actores	Administrador
Propósito	Insertar una nueva sesión.
Resumen: El caso de uso se inicia cuando administrador del sistema desea insertar una nueva sesión, este debe poder crear una nueva sesión.	

Acción del actor	Respuesta del Sistema
1 El administrador selecciona el proceso insertar nueva sesión. 3 El administrador inserta la nueva sesión.	2- El sistema muestra la interfaz de la creación. 4- El sistema lo adiciona en la zona de sesiones existentes, si ya existían algunos con anterioridad lo adiciona con los demás.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
----------	**Muestra las sesiones creadas.**

Caso de Uso	Iniciar Sesión.
Actores	Administrador, Director del complejo, Sub_Director del complejo.
Propósito	Iniciar una nueva sesión.
Resumen: El caso de uso se inicia cuando administrador o el Director del complejo o el Sub_Director del complejo desea iniciar una nueva sesión, este debe poderla iniciarla.	
Acción del actor	**Respuesta del Sistema**
1- El administrador o el Director del complejo o el Sub_Director del complejo selecciona el proceso iniciar nueva sesión.	2- El sistema muestra la interfaz de la creación.

3 El administrador inicia la nueva sesión.	4- El sistema lo adiciona en la zona de sesiones existentes, si ya existían algunos con anterioridad lo adiciona con los demás.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
----------	**Muestra las sesiones creadas.**

Caso de Uso	**Registrar destino.**
Actores	Administrador
Propósito	Registrar destino.
Resumen: El caso de uso se inicia cuando administrador del sistema desea registrar un destino.	
Acción del actor	**Respuesta del Sistema**
1 El administrador selecciona el proceso registrar destino. 3 El administrador registra el destino.	2- El sistema muestra la interfaz de la creación. 4- El sistema lo registra.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
----------	----------

Caso de Uso	Brindar reportes.
Actores	Director del complejo. Sub_Director del complejo. Vicerrector. Dpto. Económico. Especialistas.
Propósito	Obtener reportes.
Resumen: El caso de uso se inicia cuando Director del complejo, el Sub_Director del complejo, el Vicerrector, el Jefe del Dpto. Económico o los Especialistas del sistema desea obtener algunos de los reportes que deben generarse.	
Acción del actor	**Respuesta del Sistema**
1- Los actores del caso de uso seleccionan el proceso brindar reportes.	
	2- El sistema muestra la interfaz de los reportes.
3- Los actores del caso de uso seleccionan el reporte conveniente.	
	4- El sistema lo muestra.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
----------	**El sistema muestra los reportes.**

Caso de Uso	Restaurar Datos.
Actores	Administrador
Propósito	Restaurar datos.
Resumen: El caso de uso se inicia cuando el administrador desea restaurar los datos	

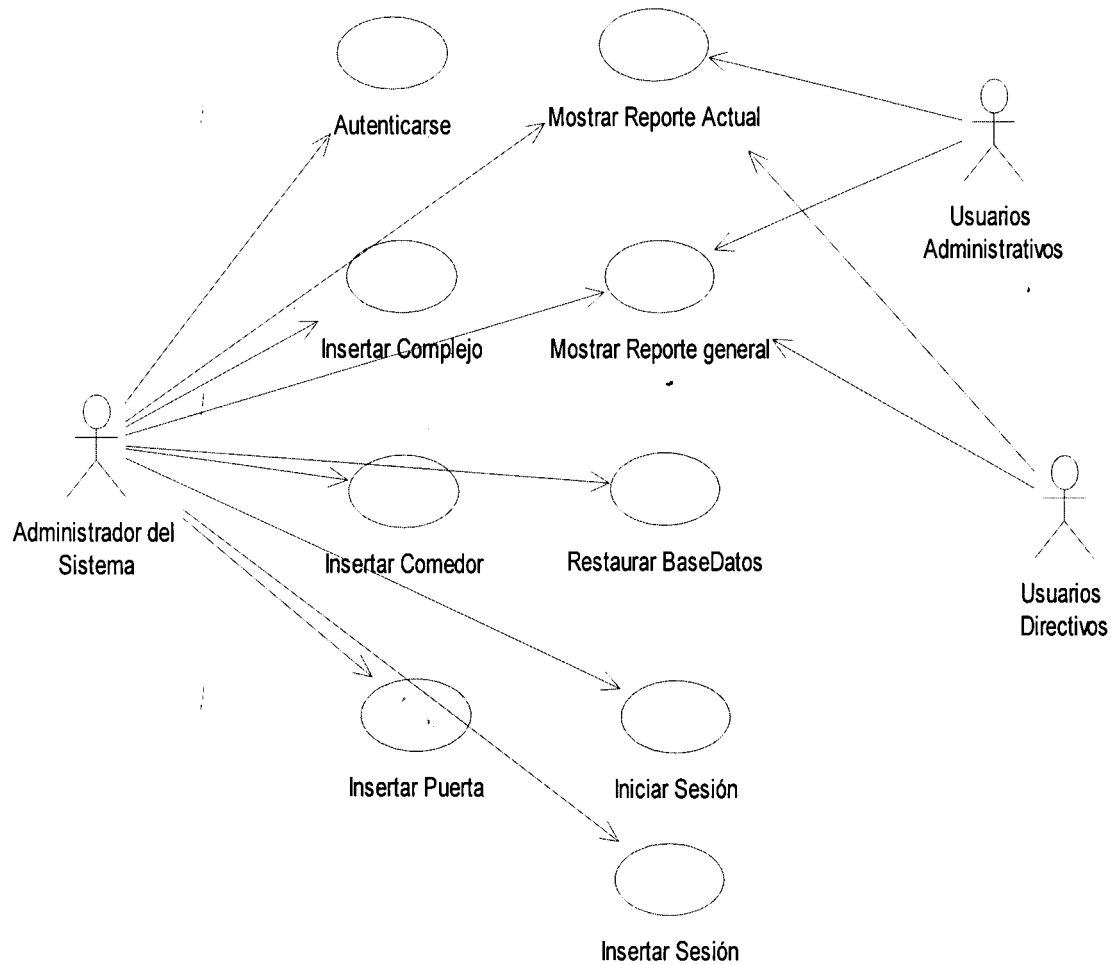
para poder utilizarlos nuevamente en la siguiente sesión.	
Acción del actor	**Respuesta del Sistema**
1-El administrador selecciona restaurar los datos.	2- El sistema muestra la interfaz de restauración brindando un mensaje de si fueron restaurados los datos satisfactoriamente o no.
Flujo alternativo	
Acción del Actor	**Respuesta del sistema**
----------	**El sistema muestra el mensaje de los datos restaurados.**

Selección de los Casos de Uso para cada ciclo

Ciclo	Caso de Uso	Justificación de la selección
1	• **Insertar barcode.** • **Validar entrada modulo de administración.** • **Crear un Nuevo Complejo.** • **Crear un Nuevo Comedor.** • **Insertar Puerta.** • **Insertar Sesión.**	Estos casos de uso influyen en la arquitectura básica del sistema, en ellos están presentes los procesos más complejos y que representan la funcionalidad principal del sistema. En estos casos de uso es necesario una autentificación por lo cual solamente se les puede dar acceso a la aplicación a aquellos usuarios autorizados, para de este modo proteger la información que se maneja. Los otros casos de usos seleccionados son

		necesarios para la futura acciones que se realizan.
2	• **Cancelar Acceso.** • **Registrar Destino** • **Restaurar Datos.** • **Brindar Reporte** • **Iniciar Sesión.**	Estos casos de uso son necesario para lograr un sistema mas integro que realice múltiples acciones y evitar una vez mas duplicidad de la información así como ahorrar tiempo y trabajo a los implicados.

Diagrama Caso De Uso Módulo Administración.



Anexo 3

Glosario de términos

SCA

Sistema de control de asistencia de trabajadores: Software propuesto a realizar en esta tesis.

UCI

Universidad de las Ciencias Informáticas: Universidad de la Revolución que tiene el objetivo formar profesionales de la esfera Informática con el objetivo de aumentar el conocimiento de la informática en cuba y poder ayudar a países vecinos

Assert: Sistema de recursos humanos que esta implantado en la UCI.

Estructura cliente / servidor: Consiste en que los clientes piden que una tarea sea realizada y el servidor realiza dicha tarea y regresa la información al cliente a través de la red.

Herramienta Case: Ingeniería de sistemas asistida por ordenador (Computer-Aided Systems Engineering - CASE) es la aplicación de tecnología informática a las actividades, las técnicas y las metodologías propias de desarrollo de sistemas. Su objetivo es automatizar o apoyar una o más fases del ciclo de vida del desarrollo de sistemas.

HTTP: Protocolo usado para la transferencia de documentos WWW. Estas transferencias requieren un programa cliente http en un extremo de la comunicación y un servidor http en el otro.

I

Interfaz web:

Intranet: Red privada, desarrollada dentro de una compañía que utiliza el mismo software y provee de información similar que Internet, solo que es únicamente para el uso interno.

Power Builder: Es un entorno de programación que está compuesto por diferentes herramientas, para el desarrollo rápido de una aplicación en el ambiente cliente - servidor.

Protocolo FTP: Protocolo de transmisión de ficheros.

Reporte: Informe detallado sobre alguna información, o sobre el estado de la información.

TCP/IP: Sistema de protocolos, definidos en RFC 793, en los que se basa buena parte de la comunicación de Internet. TCP/IP es el estándar de protocolo de comunicaciones requerido por las computadoras que acceden a Internet.

VBScript: Lenguaje de programación de la parte del cliente.