# Universidad de las Ciencias Informáticas

# Facultad 5


Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

**Título:** Sistema para la Planificación de la Producción de Catering.

**Autores:**

Dayana Leticia López Chávez.

Javier Cruz Barrero.

**Tutores:**

Msc. Yuniesky Coca Bergolla.
Ing. Reinier Lemus Martínez.

La Habana, junio de 2015.

**Declaración de Autoría**

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

| | |
|---|---|
| _____ | _____ |
| Firma del Autor | Firma del Autor |
| Dayana Leticia López Chávez. | Javier Cruz Barrero. |
| _____ | _____ |
| Firma del Tutor | Firma del Tutor |
| Msc. Yuniesky Coca Bergolla. | Ing. Reinier Lemus Martínez. |

**Dedicatoria**

## *De Dayana:*

*A mi mamá Ileana.*

*A mi abuela Milagros.*

## *De Javier:*

*A mi mamá Loretta.*

*A mi papá Daniel.*

## Agradecimientos

*De Dayana:*

*A mi familia pues son las personas que hacen posible que mis sueños se hagan realidad.*

*A mi tía sin ti no hubiéramos podido trabajar en la tesis.*

*A mi abuelo por estar siempre presente.*

*A me querido intranquilo hermanito, pues me has ayudado a saber qué significa ser responsable.*

*A mi segunda familia. A mi hermana postiza Zahily, por siempre estar a mi lado aconsejándome y ayudándome. A mi segunda madre Dailys, por ser parte de mi familia y por estar siempre atenta de cómo estamos y qué necesitamos para ayudarnos.*

*A mi compañero de tesis, mi amigo, mi novio, mi confidente por ser como eres.*

*Al tutor que me guió en toda la carrera, Prevot, gracias por ser un ejemplo a seguir.*

*De Javier:*

*A mi familia por estar conmigo y apoyarme durante toda mi vida y sobre todo durante estos 5 años.*

*A mi hermano Marco que aunque no nos vemos muy a menudo sé que podemos contar siempre el uno con el otro, para cualquier cosa.*

*A Miriam y a Gustavo por ser como mis segundos padres para mí y a sus familias por hacerme sentir como un miembro más de ellos.*

*A la familia a de mi novia por tratarme como uno más de la familia.*

*A mi novia por ser mi orgullo, satisfacción y fiel compañera durante todo nuestro tiempo juntos.*

*De nosotros:*

*A nuestros tutores por rectificarnos y ayudarnos a lograr que el trabajo de diploma tuviera la mayor calidad posible, ellos son: Lemus, Coca, José Ángel y Yadira.*

*A la profe Susej por ser la mejor jefa de año que hemos tenido en toda la carrera.*

*A todos los amigos y amigas que hice en la carrera, por compartir momentos inolvidables.*

*A la UCI por ser tan dinámica y enseñarme que sí se puede estar en varias cosas a la vez.*

**Resumen**

La industria de la alimentación constituye uno de los principales pilares de la economía de nuestro país, en este sector destacan muchas empresas de gran prestigio y demanda como la empresa CUBA CATERING, S.A. que se dedica a brindar servicios de catering aéreo y de gastronomía.

Para lograr la meta de esta empresa, la cual no es más que la satisfacción total de cada cliente a quien brinda servicio, se hace necesario desarrollar un nuevo sistema confiable y capaz de ejercer todas las funciones necesarias para lograr que los servicios de catering aéreo y la gastronomía constituyan uno de los más eficientes y con más calidad de nuestro país.

Este trabajo describe el proceso de desarrollo de un software que permitirá informatizar todos los procesos que intervienen en la planificación de la producción de la empresa, permitiendo la interacción de información a través de reportes con otros sistemas logrando eliminar al máximo el trabajo manual, reducir pérdidas e incrementar ganancias.

# *Índice*

## Introducción

Los aeropuertos son las terminales en tierra donde se inician y concluyen los viajes de transporte en aeronaves hacia los diferentes países del mundo. En muchas ocasiones estos traslados se tornan largos, por lo que en cada estación de aviones existen servicios de catering aéreo para lograr un mayor confort de los pasajeros durante este largo período.

Inicialmente en los aeropuertos era habitual que los pasajeros comieran en un restaurante, antes de embarcarse, pero luego se impuso la modalidad de incluir comida caliente a bordo, elaborada en esos mismos restaurantes y conservada en grandes termos que mantenían una temperatura adecuada durante dos horas. Años más tarde con la invención de técnicas para la conservación en estado frío y caliente de la comida, además de la aparición de numerosas aerolíneas de bajo costo y el aumento del combustible, las empresas aéreas tradicionales comenzaron la creación de los catering y resulta que en la actualidad dicho negocio representa uno de los mayores ingresos de las compañías aeroportuarias del mundo.

En Cuba no es diferente, pues la prestigiosa empresa Cuba Catering S.A. es la organización de las once que integran la Corporación de la Aviación Civil, que se encarga de brindar los servicios de catering aéreo y gastronomía en todas las instalaciones aeroportuarias del país. Actualmente tiene 9 Unidades, 4 de ellas en Ciudad de La Habana, las restantes en los Aeropuertos de Varadero, Ciego de Ávila, Camagüey, Holguín y Santiago de Cuba. (1)

Actualmente para la planificación de los servicios de catering de esta empresa se utiliza un sistema que no cumple con todas las expectativas y exigencias de este negocio, por lo que surge la necesidad de adquirir un nuevo sistema.

Para la producción de los servicios de Catering se utiliza un sistema elaborado por la propia empresa desde el año 1997. Dicho sistema está soportado e implementado sobre

MS-DOS, sistema creado por Microsoft en el 1981, que fue sustituido gradualmente por sistemas que ofrecían una interfaz gráfica de usuario, por lo que para su soporte y mantenimiento se necesita un personal preparado en estas tecnologías, además el trabajo de las planificadoras en el mismo resulta complicado debido a la cantidad de código, información y cálculo que deben realizar manualmente, para después ingresar en el sistema. Además el sistema actual presenta algunas deficiencias como es el caso de la poca seguridad de la base de datos, ya que cualquier usuario que tenga conocimientos podría modificar los permisos de lectura y escritura, pues los archivos que genera la misma se pueden abrir con Excel. También en el momento que se desarrolló el actual sistema no se conocía mucho acerca de la Arquitectura de un Catering, por lo que no se pudo lograr una planificación por ciclos, ni tampoco generar todas las alertas requeridas. Otra de las grandes deficiencias es que actualmente la planificación de los servicios de Gastronomía se realiza de forma manual.

Para mitigar las deficiencias mencionadas anteriormente, la empresa decidió crear un nuevo sistema en la misma empresa con la colaboración de la Universidad de las Ciencias Informáticas, que cumpliera con todos los requisitos establecidos, debido a que la idea de comprarlo a una empresa comercializadora de software resultaría costosa y no se obtendría el producto deseado, debido a que estos software se basan en satisfacer las necesidades de una producción lineal, que no es el caso de Cuba Catering, pues en ésta, cada ingrediente pasa por diferentes procesos, sufriendo una serie de transformaciones que no necesariamente implica obtener el mismo producto.

Debido a la situación descrita anteriormente, sale a relucir el siguiente **problema a resolver**:
¿Cómo mejorar la planificación de la producción de la empresa Cuba Catering S.A atendiendo a las necesidades actuales de la organización?

Este problema se enmarca en el **objeto de estudio**: Los servicios de catering aéreo y gastronomía, dentro del que se ha identificado como **campo de acción:** Los procesos que intervienen en la planificación de la producción de la empresa Cuba Catering S.A. Como **idea a defender**: Se plantea que con el desarrollo de un sistema informático se

podrá mejorar los procesos que intervienen en la planificación de la producción de la empresa Cuba Catering S.A.

Se define como **objetivo** de este trabajo, desarrollar un sistema para la planificación de la producción de la empresa Cuba Catering S.A.

Dicho esto tenemos como **posibles resultados**: Obtener un sistema informático para la planificación de la producción de la empresa Cuba Catering S.A que intercambie información a través de reportes con otros sistemas que se encuentran en explotación e interactúan con este proceso, así como la documentación asociada al proceso de desarrollo del mismo.

Para dar cumplimiento al objetivo, se ha organizado el trabajo a partir de las siguientes **tareas de investigación**:

- Definición de los principales conceptos, las tendencias existentes y las posibles soluciones, a partir del estudio de los referentes teórico-prácticos.
- Selección de las tecnologías, herramientas y metodología de desarrollo a utilizar para la implementación del sistema para la planificación de la producción de Cuba Catering S.A.
- Definición de los requisitos funcionales y no funcionales que debe poseer el sistema para satisfacer las necesidades existentes.
- Diseño de las interfaces del sistema, teniendo en cuenta la conformidad del cliente y manteniendo las políticas principales de la empresa Cuba Catering S.A.
- Implementación del sistema, de acuerdo a los requisitos planteados y a las funcionalidades necesarias que el mismo debe poseer.
- Validación del funcionamiento correcto de la solución, mediante la realización de pruebas de software.

Para alcanzar los objetivos propuestos se utilizan los siguientes métodos científicos:

**Métodos Teóricos**

- *Análisis Histórico-Lógico*: Este método se empleó para la fundamentación de los aspectos teóricos contemplados en el desarrollo de la investigación acerca del catering aéreo y demás elementos relacionados con el contenido del trabajo.
- *Modelación*: Durante el proceso de desarrollo este método permitió esclarecer mediante diagramas todo el funcionamiento de los principales procesos del negocio.

**Métodos Empíricos**

- *Observación:* Este método se empleó para tener una mejor visión de cómo transcurre todo el proceso productivo en la empresa.
- *Entrevistas*: Este método permitió hacer una recopilación general de información acerca del negocio, así como las demandas y exigencias de parte de los trabajadores hacia el nuevo sistema.
  *Experimento*: Se utilizó para la elaboración de las interfaces con el objetivo de evaluar la conformidad.

El trabajo está estructurado por 3 capítulos, tal y como se describe a continuación:

**Capítulo 1: Fundamentación teórica**, se enuncian los conceptos fundamentales del contenido del trabajo. También se realiza un estudio del negocio sobre el catering en la empresa CUBA CATERING S.A y se mencionan las tecnologías, herramientas, metodología y lenguajes a utilizar en el desarrollo de la solución.

**Capítulo 2: Propuesta de solución**; En este capítulo se presenta la modelación del negocio actual, las reglas de negocio, diagrama de dominio, requisitos funcionales y no funcionales, así como también se muestra el diagrama de casos de uso del sistema, los patrones y modelos del diseño, el diagrama de la base de dato y diagrama de despliegue.

**Capítulo 3: Implementación**; En este capítulo se procede a realizar la implementación del sistema, el cual es necesario crear un diagrama de despliegue, donde se muestra la distribución física del sistema. También se confeccionó un diagrama de componentes, que representa cómo un sistema es dividido en componentes y las relaciones entre ellos.

Además se mencionan los aspectos relevantes de la implementación, como por ejemplo su estructura y estándares de codificación. Por último se realizan pruebas a la aplicación para garantizar la calidad de  la misma.

# Capítulo 1: Fundamentación Teórica

## 1.1 Introducción al Capítulo

En este capítulo se analizará de forma teórica los principales aspectos que fundamentan la solución propuesta, así como el objetivo que se persigue. Se hará un análisis crítico de las tecnologías actuales para el análisis, diseño y desarrollo de sistemas informáticos.

## 1.2 Elementos Fundamentales del Negocio

El **Proceso de Producción** de alimentos no es más que una serie de procedimientos para preparar los alimentos adquiridos e integrarlos en los menús que se servirán en los distintos servicios a los clientes o pasajeros de una determinada aerolínea. En la empresa Cuba Catering S.A el proceso básicamente está conformado por cuatro etapas principales: Almacenamiento, Áreas de Producción, Armado y Áreas de Despacho.

En las áreas de **Almacenamiento** se conserva toda la materia prima, clasificándose en tres tipos de almacenamiento: almacenamiento en congelación, almacenamiento en refrigeración y almacenamiento a temperatura ambiente. En el **Almacenamiento en Congelación** los cárnicos y derivados, embutidos y ahumados, los lácteos y derivados, frutas y hortalizas congeladas se conservan en congelación hasta que se requiera el retiro y utilización de alguno de estos alimentos. En el **Almacenamiento en Refrigeración** algunos embutidos y ahumados, los huevos, los lácteos y derivados, frutas y hortalizas y los productos industriales se almacenan en refrigeración para su adecuado cuidado y en el **Almacenamiento en Temperatura Ambiente** el objetivo principal es conservar en temperatura ambiente los lácteos y derivados, viandas, los cereales y derivados, leguminosas, grasas, aceites, azúcar, sirope, miel, confituras, conservas, condimentos, especias, sal, frutos secos, pasas, productos industriales, infusiones, bebidas y licores.

En las **Áreas de Producción** se realizan una serie de pasos y procedimientos que inciden en diferentes trasformaciones de los alimentos los cuales cada uno de ellos

necesita o no pasar por un área específica. Es necesario tener especificado en las áreas por las que pasan los productos, las horas de antelación con las que hay que solicitar del almacén la materia prima y con la que se debe especificar el plan de trabajo de cada área. A continuación se mencionan las diferentes áreas de producción:

- ✓ **Carnicería-Pescadería.**
- ✓ **Local de beneficio y cascado de huevos.**
- ✓ **Legumier Pre-elaboración de frutas y vegetales.**
- ✓ **Panadería-Dulcería.**
- ✓ **Área caliente.**
- ✓ **Pre-elaboración.**
- ✓ **Área del café.**
- ✓ **Área fría (Lunch y Área Especial).**
- ✓ **Frío 1ra clase y Frío económico.**
- ✓ **Nevera: fría, caliente, pan y dulce, tránsito y producción terminada de panadería.**
- ✓ **Comedor obrero.**
- ✓ **Atención al hombre.**

En el área de **Armado** se procede a montar todos los servicios en las bandejas o envasado adecuado para ser transportado y asegurando que no ocurra ningún lamentable accidente en su manipulación hacia la aeronave.

Las **Áreas de Despacho** son aquellas donde se encuentran todos los aseguramientos necesarios para darle salida a los alimentos una vez que salen del área de Armado y están listos para ser transportados en los camiones para las distintas entidades que contratan los servicios de la empresa CUBA CATERING S.A.

La **planificación de la producción** se refiere específicamente a la posibilidad de emitir, a partir de la información de las operaciones diarias planificadas, tres reportes fundamentales: *Solicitud de Materias Primas al Almacén* desde cada área que solicita, *Plan de Trabajo* para cada área de producción donde se diga a los elaboradores qué hacer en cada área, así como lo que deben recibir desde otras; además de las *Órdenes*

*de Facturación* según lo que se va a producir.

### 1.3 Objeto de estudio

Como empresa en perfeccionamiento, Cuba Catering S.A tiene realizado un ejercicio estratégico en el que se definió la misión y objetivos estratégicos para el período 2014 – 2015, los cuales se muestran a continuación:

La **Misión** de la empresa está dirigida a proveer servicios de catering aéreo y gastronómicos en las instalaciones aeronáuticas con la variedad y calidad concertada con los clientes.

Los **Objetivos Estratégicos** se basan en implementar nuevos programas que permitan automatizar e integrar los procesos de la empresa, modernizando y ampliando el equipamiento, aumentando la disponibilidad, confidencialidad e integridad en el uso de las redes de infocomunicaciones; fortaleciendo las medidas de control encaminadas al incremento de la Seguridad Informática. También se enfocan en potenciar la comercialización de los servicios de Catering Aéreo y Gastronomía en las instalaciones aeronáuticas del país, de forma que aseguren el crecimiento previsto de las ventas, a partir de poner en práctica los resultados de los Estudios de Mercado en cada zona donde la empresa ofrece sus servicios, en aras de identificar y conocer las características, necesidades y expectativas de los clientes, así como potenciar la comunicación con los clientes a partir de la Medición de la Satisfacción de los mismos, como mecanismo de retroalimentación para el mejoramiento continuo de los servicios ofrecidos por la organización.

#### 1.3.1 Flujo de los procesos del negocio

El flujo actual de los procesos comienza cuando un cliente solicita servicio a la Dirección Comercial por teléfono o correo y llena los datos necesarios. Una vez llenos, se archivan los datos de la compañía que solicita el servicio. Después prosigue **Gestionar Oferta de Servicios** en donde se elabora una Oferta de servicio siguiendo las indicaciones del cliente. Una vez el cliente conforme se procede a **Registrar Descripción de Servicios**

**Contratados** que contiene la información referente a los servicios solicitados por el cliente. Luego de registrada la descripción de los servicios contratados se realiza el proceso **Planificar Producción de Servicios Contratados**, en donde se planifican los servicios con determinada antelación, para esto se emiten tres reportes fundamentales: Solicitud de Materias Primas al Almacén, Planes de Trabajo por Áreas de Producción y Orden de Facturación.

### 1.4 Sistemas automatizados existentes vinculados al campo de acción

Para la planificación de la producción en la empresa Cuba Catering S.A se utiliza actualmente un sistema implementado en FoxPro, el mismo permite la gestión de los nomencladores[1], la descripción de los planes de trabajo por cada área, un sistema de reportes, entre otras funciones básicas para el desarrollo de un catering. Este sistema presenta algunas deficiencias, algunos ejemplos de esto son: la introducción de datos repetidos innecesariamente, la no incorporación de algunos procesos que intervienen directamente en la producción, como la conformación de la oferta de servicio, así como la no existencia de funcionalidades que gestionen los servicios de gastronomía. La generación de alertas es otra de las prestaciones que no brinda este sistema, así mismo, la poca seguridad en los datos y la ambigüedad de las plataformas sobre la que está soportado, constituyen otra de las agravantes del mismo.

En el mercado internacional, luego de continuas búsquedas en Internet, se pudo comprobar la variedad de sistemas de este tipo que existen. Algunos ejemplos de estos sistemas son:

**Fig 1**. Logo del sitio *www.Gategourmet.com.*

*Gategourmet* fundado en 1992 en Kloten, Suiza, ofrece soluciones de catering y aprovisionamiento para el servicio de trenes de pasajeros, salas de aeropuertos, tiendas de conveniencia y establecimientos afines. (2)

[1] ***Nomencladores:*** Es una forma de clasificar o agrupar diversas prácticas, actividades, tipo de especies, enfermedades, etc. (35)


**Fig 2**. Logo del sitio *www.servair.com.do.*

*Servair* creado en 1986 en la República Dominicana para proporcionar servicios de catering en todos los aeropuertos del país. (3)


**Fig 3.** Logo del sitio *www.lsgskychefs.com.*

*LSG Sky Chefs*: Es el grupo líder del catering aéreo en el mundo y de la administración y planificación de todos los procesos relacionados con los servicios de vuelo. El grupo consiste en 159 compañías con 210 servicios para el cliente en 51 países. (4)


**Fig 4.** Logo del sitio *www.chefexact.es.*

*Chefexact Catering:* Es un programa enfocado a la gestión ágil, segura y eficaz de cualquier negocio dedicado a la distribución de alimentos preparados. La interfaz de esta aplicación nos presenta dos áreas: una en la parte superior con las secciones y otra, en el lateral izquierdo, con las funciones, para poder entrar y trabajar en cada una de ellas. La aplicación se estructura en 5 secciones: sección de los parámetros y datos generales del negocio, sección de Gastos Generales, sección de recetas, sección de presupuestos y servicios y sección de Ventas donde se emite las facturas de los servicios realizados. (5)

Todos estos sistemas son comerciales y privativos por lo que solo se muestra información publicitaria e informativa y no se puede acceder a las funcionalidades y procesos internos. Dichos sistemas tienen como principal inconveniente el precio de adquisición, así como la dependencia tecnológica que en el caso específico de Cuba constituye una amenaza mayor. En el caso específico de la empresa este último inconveniente no constituye un problema, puesto que la misma mantiene sus productos a través de la compra de licencias a las compañías encargadas de proporcionar dichos productos.

## 1.5 Tecnologías utilizadas en el desarrollo del sistema

Actualmente en la empresa Cubacatering S.A se utiliza el sistema operativo Windows XP, pero en un futuro próximo se piensa migrar a Windows 7, por lo que las tecnologías y herramientas utilizadas se escogieron a partir de estas condiciones y siguiendo los estándares actuales de informatización de los demás sistemas de esta entidad.

Se utilizaron para la realización de este trabajo RUP (Proceso Racional Unificado) como metodología de desarrollo, utilizando UML (Lenguaje Unificado de Modelado). También se empleó C# (lenguaje de programación orientado a objetos), CSS (Hoja de estilos en cascada) mediante Bootstrap (Framework o conjunto de herramientas para diseño de sitios y aplicaciones web). Razor y HTML5 (lenguajes para la elaboración de las vistas de la página web) y Javascript para crear páginas web dinámicas utilizando jQuery como framework. La herramienta escogida para la modelación del negocio fue Visual Paradigm for UML versión 8.0 puesto que permite la creación de diagramas para analizar y tener registrado el funcionamiento del negocio, así como los diagramas de caso de uso del sistema. Como gestor de base de datos se utilizó SQL Server 2005 versión 9.0 y el entorno de desarrollo Visual Studio Ultimate 2013 utilizando el Framework ASP.NET que incluye Entity Framework y LINQ para las consultas a la base de datos. A continuación se esbozan las principales características de estas herramientas.

### 1.5.1 Metodologías de desarrollo

**RUP**

El Proceso Unificado de Desarrollo de Software, correspondientes a su nombre en inglés – Rational Unified Process, es un modelo de proceso de desarrollo de software que utiliza el lenguaje UML para preparar todos los esquemas de un sistema de software. Se distingue por promover un desarrollo basado en tres definiciones importantes:

- El Proceso Unificado de Desarrollo es dirigido por casos de uso.
- El Proceso Unificado de Desarrollo está centrado en la arquitectura.
- El Proceso Unificado de Desarrollo es iterativo e incremental.

Se ha decidido el uso de esta metodología y lenguaje de modelo debido a los elementos

anteriormente mencionados y que el presente trabajo debe quedar con todos los requerimientos de documentación posible para su posterior seguimiento, por parte de los desarrolladores de la empresa, que no han participado directamente en este trabajo. Esta metodología tiene como principal fortaleza la de crear un lenguaje común y transparente para un equipo de trabajo y su posterior seguimiento, precisamente es el motivo principal de la elección por parte de los desarrolladores del sistema.

**UML**

El Lenguaje de Modelado Unificado (UML - Unified Modeling Language) es un lenguaje que permite modelar, construir y documentar los elementos que forman un producto de software que responde a un enfoque orientado a objetos. Este lenguaje fue creado por un grupo de estudiosos de la Ingeniería de Software formado por: Ivar Jacobson, Grady Booch y James Rumbaugh en el año 1995. Desde entonces, se ha convertido en el estándar internacional para definir organizar y visualizar los elementos que configuran la arquitectura de una aplicación orientada a objetos. Con este lenguaje, se pretende unificar las experiencias acumuladas sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar.

UML no es un lenguaje de programación sino un lenguaje de propósito general para el modelado orientado a objetos y también puede considerarse como un lenguaje de modelado visual que permite una abstracción del sistema y sus componentes.

Entre sus objetivos fundamentales se encuentran:

- Ser tan simple como sea posible, pero manteniendo la capacidad de modelar toda la gama de sistemas que se necesita construir.
- Necesita ser lo suficientemente expresivo para manejar todos los conceptos que se originan en un sistema moderno, tales como la concurrencia y distribución, así como también los mecanismos de la ingeniería de software, como son el encapsulamiento y los componentes.
- Debe ser un lenguaje universal, como cualquier lenguaje de propósito general.
- Imponer un estándar mundial. (6)

### 1.5.2 Herramientas utilizadas

**Visual Paradigm for UML 8.0**

Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML, ideal para la construcción de sistemas a gran escala y que necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos.
Visual Paradigm también ofrece:

- Navegación intuitiva entre la escritura del código y su visualización.
- Potente generador de informes en formato PDF/HTML.
- Ambiente visualmente superior de modelado. (7)

Para el desarrollo del sistema propuesto es conveniente utilizar esta herramienta, pues la misma permite diseñar y modelar un conjunto de diferentes diagramas que se convertirán en el punto de partida para las posteriores tareas de investigación, que también aporta una documentación general del funcionamiento del negocio.

**SQL Server 2012**

SQL Server 2012 presenta mejoras importantes en el modelo de seguridad de la plataforma de base de datos, con la intención de ofrecer un control más preciso y flexible que permita una seguridad mayor de los datos. Sus desarrolladores han realizado una importante inversión en una serie de características a fin de proporcionar un alto nivel de seguridad para los datos que incluyen:

- Aplicación de directivas para las contraseñas de inicio de sesión de SQL Server en el espacio de la autenticación.
- Incorporación de mayor granularidad en términos de especificación de permisos en varios ámbitos en el espacio de la autorización.
- Capacidad de separación de propietarios y esquemas en el espacio de la administración de seguridad.

Para el almacenamiento de los datos que manejará el sistema se escoge SQL Server 2012 porque incluye un gran número de nuevas tecnologías que aportan un aumento considerable en la productividad del desarrollador. Desde compatibilidad con el Framework ASP.NET hasta la estrecha integración con Visual Studio, estas características ofrecen la capacidad de crear de forma más sencilla aplicaciones de bases de datos sólidos y seguros a un bajo costo. (8)

**Visual Studio Ultimate 2013**

Visual Studio 2013 es una de las mejores herramientas que permite construir y entregar aplicaciones modernas conectadas en todas las plataformas de Microsoft porque crea muchos tipos distintos de aplicaciones: aplicaciones de escritorio, aplicaciones web, servicios web, entre otras. Puede escribir código en Visual Basic, Visual C#, Visual C++, Visual F# y JavaScript, así como aplicaciones que pueden utilizar código en distintos lenguajes. (9)

Esta herramienta se utilizó porque es el IDE de desarrollo que contiene una colección completa de servicios y herramientas para crear aplicaciones web, esto se evidencia mediante el Framework ASP.NET que es el que se utiliza para el desarrollo de la propuesta de solución.

### 1.5.4 Lenguajes de Programación

Existen diversos y muy útiles lenguajes de programación para el desarrollo de aplicaciones Windows como la que se propone en este trabajo. Java, Delphi, Pascal, Visual Basic, C, C++, y C# entre otros, dentro de los cuales muchos de estos se encuentran en el ambiente de trabajo del Framework ASP.NET. Entre los lenguajes de programación con los que cuenta la plataforma de trabajo seleccionada se decidió utilizar C# por ser un lenguaje nuevo, fácil de usar y tener características innovadoras que posteriormente se expondrán. C# es un lenguaje de programación que integra según plantean sus creadores lo mejor de otros potentes lenguajes como Java, Delphi, Visual Basic y C++.

**C, C++ y C#**

El lenguaje de programación C# fue creado con el mismo espíritu que los lenguajes C y C++. Esto explica sus poderosas prestaciones y su fácil curva de aprendizaje. No se puede decir lo mismo de C y C++, pero como C# fue creado desde cero, Microsoft se tomó la libertad de eliminar algunas de las prestaciones más pesadas (cómo los punteros). C y C++ pueden dar como resultado aplicaciones muy potentes, pero debe asegurarse de que el código funciona bien. Tiene que escribir su propio código para manejar aspectos como la gestión de memoria y el control de errores. Un error en la escritura del programa puede hacer que toda la aplicación falle o se comporte de forma inesperada. C# elimina los aspectos que hacían que fuese difícil trabajar con C y C++. Por ejemplo, el código C es también código C++, C++, por tanto, tuvo que mantener todas las rarezas y deficiencias de C. C# parte de cero y sin ningún requisito de compatibilidad, así que puede mantener los puntos fuertes de sus predecesores y descartar las debilidades que complicaban las cosas a los programadores de C y C++. Estos últimos, son lenguajes con los que los creadores de este proyecto se encuentran familiarizados ya que durante el transcurso de sus carreras, dentro del plan de estudios estaba incluido el aprendizaje de la programación orientada a objetos con C++ como lenguaje base.

**C#**

Según sus constructores y la propia bibliografía consultada, es un lenguaje orientado a objetos, moderno y seguro. El Framework ASP.NET proporciona muchas clases que ayudan a los programadores a reutilizar el código. Las bibliotecas de clase ASP.NET contienen código para programar subprocesos, entrada y salida de archivos, compatibilidad para bases de datos, análisis XML y estructuras de datos, como pilas y colas. Toda esta biblioteca de clase está disponible para cualquier lenguaje de programación compatible con el Framework ASP.NET. (10)

**HTML**

Es un estándar que sirve de referencia para la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, entre

otros. El lenguaje HTML basa su filosofía de desarrollo utilizando la referencia. Para añadir un elemento externo a la página (imagen, vídeo, script, entre otros.), este no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene sólo texto mientras que recae en el navegador web (interpretador del código) la tarea de unir todos los elementos y visualizar la página final. (11)

**Razor**

Para crear las vistas en el Framework ASP.NET se utilizan diferentes tipos de archivos, como por ejemplo: HTML (.htm o .html), ASP Clásico (.asp), ASP.NET Clásico (.aspx) y ASP.NET Razor C# (.cshtml). Se decidió utilizar el lenguaje de Razor debido a que proporciona una sintaxis optimizada para la generación de HTML utilizando un enfoque de plantillas código-centrado, con la transición mínima entre HTML y el código. El diseño reduce el número de caracteres y pulsaciones de teclas y permite un flujo de trabajo de codificación más fluido porque no requieren bloques de servidores indicados explícitamente en el código HTML. Otras ventajas que se han observado es que soporta "layouts" (una alternativa al concepto de "página principal" en las páginas aspx clásicos).

Con frecuencia será necesario combinar Razor, con el texto y las etiquetas HTML dentro de los bloques de código. Cuando esto suceda, ASP.NET es capaz de decidir la diferencia entre ellos. En el motor ASPX necesita el tag que inicia el código de servidor (<%) y el que lo termina (%>), en cambio el motor Razor es lo suficientemente inteligente para saber cuándo termina el código de servidor, sin necesidad de que lo explicitemos. En Razor el símbolo de la arroba (@) marca el inicio de código de servidor. El uso de la @ funciona de dos maneras básicas:

1. @expresión: Muestra la expresión en el navegador. Así @item.Nombre muestra el valor de ítem.Nombre. Es decir @expresión equivale a <%: expresión %>
2. @ {código}: Permite ejecutar un código que no genera salida HTML. Es decir @ {código} equivale a <% Código %>. (12)

**CSS**

Hoja de estilo en cascada es un lenguaje usado para definir y crear la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El World Wide Web Consortium (W3C) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.

La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación.

La información de estilo puede ser definida en un documento separado o en el mismo documento HTML. En este último caso podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo «style». (13)

**Bootstrap**

Es un framework o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript.
En la propuesta de solución se utiliza este framework para lograr una interfaz amigable para el usuario y obtener como resultado un diseño adecuado para un sistema de trabajo. (14)

**Javascript**

Es un lenguaje de programación interpretado que se define como orientado a objetos, basado en prototipos, imperativo y dinámico. Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS). Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM). (15)

**jQuery**

Es una biblioteca gratuita de Javascript, cuyo objetivo principal es simplificar las tareas de creación de páginas web responsivas, acordes a lo estipulado en la Web 2.0, la cual funciona en todos los navegadores modernos. Por otro lado, se dice que jQuery ayuda a que nos concentremos de gran manera en el diseño del sitio, al abstraer por completo todas las características específicas de cada uno de los navegadores. Otra de las grandes ventajas de jQuery es que se enfoca en simplificar los scripts y en acceder/modificar el contenido de una página web. Finalmente, jQuery agrega una cantidad impresionante de efectos nuevos a Javascript. (16)

## Consideraciones Parciales

En este capítulo se han descrito de forma detallada los principales componentes del objeto de estudio de esta investigación. Entre ellos se detallaron los principales aspectos relacionados con los procesos de negocio que se desarrollan en la empresa Cuba Catering. Se describió el análisis realizado sobre los sistemas automatizados existentes, que están relacionados con el campo de acción. Este análisis confirmó la necesidad de realización de este proyecto. Finalmente se mostró las tecnologías utilizadas para el desarrollo de software, determinando las herramientas y plataformas a utilizar para el desarrollo de la aplicación. Aunque como resultado se ha obtenido un producto de software propietario, teniendo en cuenta que el mismo quedará documentado con el estándar más apropiado, es posible desarrollar con facilidad futuras versiones en software libre que podrán contar con el proceso de análisis y diseño de este trabajo, pues actualmente ya existen versiones de Visual Studio como Visual Studio 2013 Community que es una nueva versión gratuita y completa de Visual Studio, la cual puede descargarse, que da posibilidades de un ecosistema extensible y soporte para varias plataforma. (17)

# Capítulo 2: Propuesta de solución

## 2.1 Introducción

En este capítulo se presenta la modelación del negocio, donde se realiza el análisis previo para comenzar la confección del software, para transformar requisitos y necesidades de los usuarios de modo que satisfagan sus expectativas. Se presentará el modelo de dominio del negocio, donde se podrá observar las relaciones entre las entidades principales a tener en cuenta del negocio, así como también el diagrama de casos de uso del sistema, donde se puede apreciar cada caso de uso y el responsable de llevar a cabo el mismo. También se muestra la arquitectura que presenta el sistema, los patrones de diseño a utilizar y los modelos del diseño. También se muestra el diseño de la base de datos que no es más que la representación visual de cada tabla de la base de datos con las relaciones entre ellas.

## 2.2 Procesos Objeto a automatizar

Los procesos que serán objeto de automatización son:

Primeramente un cliente solicita servicio a la Dirección Comercial por teléfono o correo. Para ello debe llenar los datos asociados a ella que en el modelo diseñado por la Dirección Comercial aparecen y enviarlos por correo. Una vez llenos, se procede a archivar los datos de la compañía que solicita el servicio.

- ***Proceso de Gestionar Oferta de Servicio*:** Este proceso comienza cuando el especialista de comercial en coordinación con el de producción elabora una oferta de servicio siguiendo las indicaciones del usuario, teniendo en cuenta la solicitud de oferta diseñada por el cliente en caso de que exista. Una vez conformada se le envía al usuario para saber su conformidad. El usuario revisa la oferta y en caso de que sea necesario solicita una presentación de los servicios que los especialistas de la empresa deberán preparar. Allí se realizan los últimos ajustes de los servicios y una vez conforme el cliente se procede a realizar el contrato. El mismo tendrá anexo la Descripción de Servicios con los precios conformados.

- ***Proceso de Registrar Descripción de Servicios Contratados:*** El especialista comercial entregará al de producción la Descripción de Servicios Contratados que este a su vez procede a insertar en el Sistema los datos que faltan asociados a la producción de los mismos. El especialista de producción luego de autenticarse, guiado por el sistema introducirá en máquina una serie de datos que pueden o no estar (platos, recetas e ingredientes).
- ***Proceso de Planificación de Producción de Servicios Contratados:*** Este proceso comienza cuando el planificador de la producción revisa los datos de las operaciones especificadas por el cliente y/o el comisario encargado de esta tarea, si todos están en orden se planifican los servicios con determinada antelación con el objetivo de que los servicios estén listos a tiempo para el vuelo. Una vez planificado el vuelo el planificador debe realizar una propuesta de solicitud al almacén, con las necesidades de materias primas para la producción de los servicios, por cada área de trabajo desde la que se solicitan materias primas. Este proceso es uno de los más complejos porque incluye el cálculo de la materia prima necesaria, teniendo en cuenta los rendimientos, subproductos y desechos de cada materia prima en el área. Además deben tenerse en cuenta para ello los datos de la cantidad de pasajeros para el cálculo mencionado, ya que el mismo será a partir de los porcientos insertados en la descripción de servicio. Además pedirá al sistema un reporte con los planes de trabajo por áreas de producción, con las recetas y platos que se producirán en cada área. También emitirá un reporte correspondiente a las Órdenes de Facturación.

## 2.3 Reglas del negocio a considerar

Una regla de negocio es una declaración que define o limita algún aspecto del negocio. Las reglas del negocio se pueden clasificar en: Hechos, Restricciones, Disparadores de Eventos, Inferencias y Cálculos. (18)

A continuación se muestran las reglas de negocio clasificadas:

| Clasificación | Reglas de Negocio |
|---|---|

| | |
|---|---|
| Disparadores de Eventos. | Deben tenerse en cuenta las horas de antelación de cada área de producción para la planificación de los servicios para vuelos. |
| Hecho. | Para una Aerolínea, Ruta, Servicio, Ciclo o Clase existen varios platos que componen la descripción, pero los datos anteriores van a identificar la descripción. |
| Hecho. | Los servicios estarán compuestos por platos, especificando en cada tipo de servicio la cantidad de pasajeros correspondiente por el por ciento a servir, lo cual permitirá el cálculo de la cantidad de platos a producir en cada vuelo según la información del pasaje por clases. Los platos estarán compuestos por otros platos, recetas o ingredientes. Indicando en cada caso las cantidades que llevarán de cada uno. Las recetas están compuestas por otras recetas e ingredientes, disponiendo de igual manera las cantidades a utilizar de cada uno para la confección de la receta. En el caso de los ingredientes, debe existir un término medio entre los ingredientes de almacén y los que forman parte de una receta o plato, ya que las unidades de medida son diferentes. |
| Hecho. | Existen un grupo de servicios que pueden estar contratados por aerolíneas o no, estos son los llamados servicios especiales y adicionales. Los servicios especiales son aquellos que se ofertan a una clase determinada diferenciados del resto de los pasajeros de las clases regulares de la aerolínea por la naturaleza de su origen, dígase a los bebés o los vegetarianos, etc. En el caso de los servicios adicionales son servicios extras que se contratan para ofertar en la aeronave. |

Hecho. Tanto los ingredientes como las recetas tienen un rendimiento determinado en su paso por las áreas de producción lo que debe tenerse en cuenta en el cálculo de la materia prima a utilizar según los platos o recetas a elaborar en cada área.

Hecho. Los servicios pueden ser ofertados a líneas aéreas o clientes en tierra. Los servicios a líneas aéreas son diferentes para cada clase y por cada ciclo de rotación.

Hecho. Las líneas aéreas tienen nombres específicos para sus clases por lo que se necesita clasificarlas dentro de un estándar internacional de clases aéreas. En la aplicación actual también se especifican por tipos de clases a: Clases Aéreas, Servicios Especiales y Servicios a Tripulantes.

Hecho. Las áreas por las que pasan los productos deben tener especificado las horas de antelación con las que hay que solicitar del almacén la materia prima y con la que se debe especificar el plan de trabajo de cada área.

Hecho. Los servicios van montados en un equipamiento específico que debe ser detallado en cada caso y que puede ser suministrado por la empresa o por el propio cliente.

Inferencia. En el caso de sustitución de productos, las normas especificarán lo que debe llevar cada plato para poder confeccionar su ficha de costo plan, brindando la posibilidad de modificar los productos en existencia como parte de la producción real desde cada Unidad Empresarial de Base (UEB).

| | |
|---|---|
| Hecho. | La gestión de los nomencladores se hará de forma centralizada desde la casa matriz, brindando a cada UEB la posibilidad de actualización de la producción real. Sin que esto modifique las Normas Planificadas o establecidas por la empresa. |

## 2.4 Modelo de Dominio del Negocio

El **modelo de dominio** en la resolución de problemas e ingeniería de software, es un modelo conceptual de todos los temas relacionados con un problema específico y se crea con el fin de representar el vocabulario y los conceptos clave del dominio del problema. El modelo de dominio también identifica las relaciones entre todas las entidades comprendidas en el ámbito del dominio del problema y comúnmente identifica sus atributos. (19) Debido a que el negocio es muy amplio y no se encuentra bien definido, se decide realiza un modelo de dominio que refleje todo el funcionamiento del negocio de forma general.

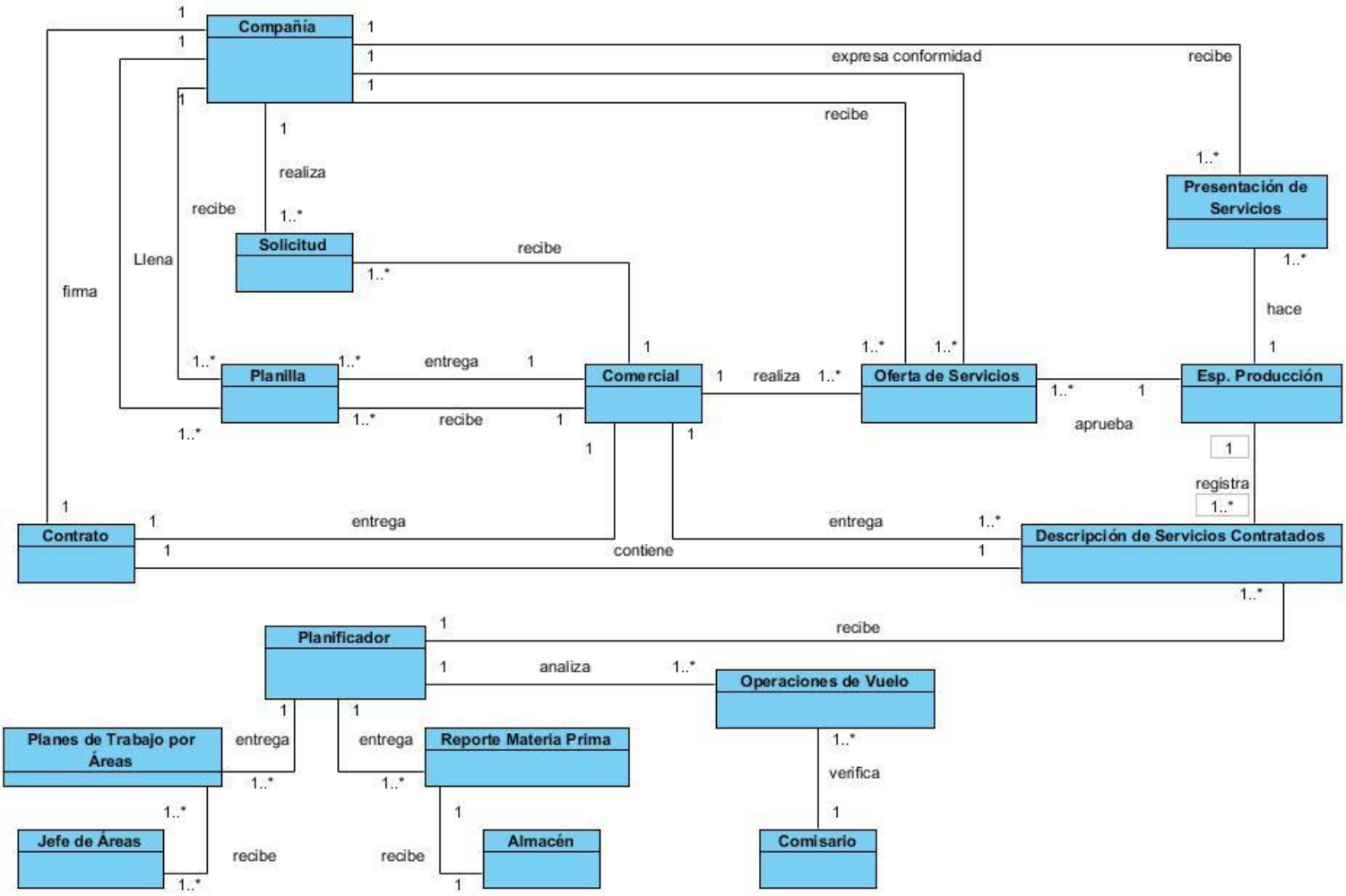


**Fig 5. Modelo de Dominio del Negocio.**

El negocio comienza cuando la compañía, ya sea una aerolínea u otra institución, solicita los servicios de catering de la empresa mediante teléfono o correo electrónico. Esta solicitud la recibe un especialista de comercial, que es el que se encarga de interactuar con el cliente en representación de la empresa. Posteriormente el mismo le entrega una planilla de solicitud al cliente, una vez que la compañía introduce los datos el comercial procede a preparar una oferta de servicio apoyándose en la solicitud del cliente, en caso de que exista. Si la compañía no está del todo convencida puede solicitar una presentación de servicios, que no es más que una muestra física de los platos a ofertar, el trabajador que realiza esta acción es el especialista de producción, que es el

encargado de elaborar los platos, las recetas y los ingredientes. Una vez que el cliente esté conforme el comercial prosigue a entregar el contrato que debe tener anexo el listado de los precios y la descripción de los servicios contratados, esta última contiene toda la información referente a los servicios contratados del cliente, dígase platos ofertados, cantidad de pasajeros, entre otros. Luego de que la compañía firme el contrato se procede a realizar el proceso de planificación, para ello el planificador debe consultar la descripción de servicios contratados. Para llevar a cabo el proceso de planificación se necesita analizar las operaciones de vuelo verificadas y confirmadas por el comisario, que es el encargado de esta tarea, en las operaciones de vuelo se plasma los detalles del vuelo, así como la cantidad de pasajeros por clase, el tipo de avión, el ciclo de rotación, entre otros. En base a lo anteriormente mencionado el planificador debe emitir dos reportes fundamentales: la solicitud de materias prima al almacén, donde debe quedar plasmado la cantidad de ingredientes necesarios en una determinada unidad de medida, para llevar a cabo la preparación de los platos y el reporte de los planes de trabajo por cada área, donde se estipula el modo de preparación de los diferentes platos por cada área de producción.

## 2.5 Requisitos Funcionales

*Ingeniería de Requisitos*, es el proceso de desarrollar una especificación de Software. Las especificaciones pretenden comunicar a los desarrolladores, las necesidades que el sistema debe cumplir a petición del cliente. Trata de los principios, métodos, técnicas y herramientas que permiten descubrir, documentar y mantener los requisitos para sistemas basados en computadora, de forma sistemática y repetible. (20)

Existen dos tipos de requisitos: *Requisitos Funcionales* y *Requisitos No Funcionales*.

Un **requisito funcional** define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas. Los requerimientos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone que un sistema debe cumplir. (21)

Existen diversas técnicas y herramientas que se utilizan para la captura de requisitos, en este caso se utilizó las entrevistas y cuestionarios, para reunir información proveniente

de personas o grupos, información que se obtiene conversando con el encuestado. Las preguntas suelen distinguirse en dos categorías: abiertas y cerradas. Las preguntas abiertas permiten que los encuestados respondan con su propia terminología, mientras que las preguntas cerradas predeterminan todas las posibles respuestas y el interrogado elige entre las opciones presentadas. (20)

A continuación se muestran los requisitos funcionales definidos:

**RF1-** El sistema debe permitir que el Especialista de Comercial conforme una Pre-oferta de Servicios.

**RF2-** El sistema debe permitir que el Especialista de Comercial y Producción realicen una búsqueda de los componentes disponible.

**RF3-** El sistema debe permitir que el Especialista de Comercial solicite los componentes elegidos por el cliente.

**RF4-** El sistema debe permitir que el Especialista de Comercial registre la Descripción de Servicios.

**RF5-** El sistema debe permitir que el Especialista de Comercial y Producción puedan diseñar ofertas de servicios, platos o recetas, independiente de si luego serán aprobados o no por el cliente. Debe permitir también incorporar en las mismas los ingredientes en el sistema, así como los componentes que tendrá una receta o un plato.

**RF6-** El sistema debe permitir que el Especialista de Comercial y Producción puedan observar el listado de los componentes existentes.

**RF7-** El sistema debe permitir que el Especialista de Comercial solicite la presentación de los servicios.

**RF8-** El sistema debe permitir que el Especialista de Comercial, Producción, Comisario y Planificador se autentiquen en el sistema, teniendo los privilegios y funciones correspondientes.

**RF9- -** El sistema debe permitir que el Especialista de Comercial, Producción, Comisario y Planificador puedan cambiar su contraseña.

**RF10-** El sistema debe permitir que el Planificador verifique las operaciones diarias, donde se especifica las confirmaciones de pasaje por clase para cada opción por compañía que se vaya a planificar.

**RF11-** El sistema debe permitir que el Planificador pueda planificar el vuelo.

**RF12-** El sistema debe permitir que el Planificador pueda rectificar el plan de vuelo.

**RF13-** El sistema debe permitir que el Planificador pueda observar las operaciones diarias.

**RF14-** El sistema debe permitir que el Planificador solicite materia prima al almacén y genere un reporte de dicha solicitud.

**RF15-** El sistema debe permitir que el Planificador solicite órdenes de facturación por vuelos y genere un reporte de dicha solicitud.

**RF16-** El sistema debe permitir que el Planificador solicite Plan de Trabajo por Áreas y genere un reporte de dicha solicitud.

**RF17-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Grupos de Áreas.

**RF18-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Áreas.

**RF19-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Banco.

**RF20-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Clases.

**RF21-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Cliente.

**RF22-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Equipamiento.

**RF23-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Monedas.

**RF24-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de País.

**RF25-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Servicio A.

**RF26-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Tipo de Avión.

**RF27-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Tipo de Servicio.
**RF28-** El sistema debe permitir que el Especialista de Comercial inserte, modifique, elimine y observe el nomenclador de Unidad de Medida.
**RF29-** El sistema debe permitir que el Comisario inserte, modifique, elimine y observe el nomenclador de Operaciones.
**RF30-** El sistema debe permitir que el Especialista de Producción inserte, modifique, elimine y observe el nomenclador de Plato.
**RF31-** El sistema debe permitir que el Especialista de Producción inserte, modifique, elimine y observe el nomenclador de Receta.
**RF32-** El sistema debe permitir que el Especialista de Producción inserte, modifique, elimine y observe el nomenclador de Ingrediente.
**RF33-** El sistema debe permitir al Administrador poder crear un nuevo usuario.
**RF34-** El sistema debe permitir al Administrador asigne un rol a un usuario.
**RF35-** El sistema debe permitir al Administrador elimine un rol de un usuario.
**RF36-** El sistema debe permitir al Administrador observe cual es el rol de un usuario.

### 2.6 Requisitos no funcionales

Un **requisito no funcional** o **atributo de calidad** es un requisito que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los requisitos funcionales. Por tanto, se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar. (22)

1- El sistema debe manifestarse en forma de aplicación Web.
2- Se necesita un servidor web SQL Server que funcione como plataforma para el servidor de Base de Datos.
3- Se necesita un servidor web Internet Information Service que funcione como plataforma para el montaje de la aplicación.
4- Se necesita un Navegador Web en cada estación de trabajo para acceder al sistema.

### 2.6.1 Usabilidad

Los sistemas desarrollados por la empresa Cubacatering deben prestar facilidades de usabilidad que satisfagan las necesidades de los usuarios, deben contar con un menú que les permita a los usuarios acceder a las principales funciones que son de su interés. Así como brindar a los usuarios la posibilidad de interactuar con los nomencladores y a toda la información necesaria para la producción, solo con los conocimientos necesarios del negocio, en este aspecto no se incluye la parte administrativa de las aplicaciones que sí requerirán de una preparación para su manipulación.

### 2.6.2 Confiabilidad

La confiabilidad debe ser alta ya que el sistema trabaja a través de la red. Debido a esto se utiliza un método de encriptación de contraseñas para garantizar la seguridad y además solo pueden acceder al sistema usuarios que se encuentren registrados en la base de datos y tengan un rol asignado.

### 2.6.3 Rendimiento

La utilización del sistema debe mantener un rendimiento adecuado, permitiendo hacer uso de las funcionalidades del sistema sin que se presente ningún problema. También debe reflejarse rapidez en la manipulación de los datos, para garantizar un funcionamiento eficiente del sistema.

### 2.6.4 Soporte

El soporte de la aplicación está basado en las pruebas sistemáticas y las mismas fallas que vayan saliendo a lo largo de su uso.

### 2.6.5 Interfaz

El sistema debe presentar una interfaz amigable, fácil de usar y con comodidades para el usuario ya que los que interactúen con él deben tener conocimientos elementales de computación. Los colores deben ser los que caracterizan la empresa, en este caso azul

y su apariencia debe ser profesional para mantener el prestigio de la empresa.

**Interfaces Hardware.**

La plataforma de montaje debe tener como mínimo:

-128 de RAM.

-1 GB de espacio disponible en el disco duro.

-100 Mbps en velocidad de la red (LAN).

-Microprocesador de más de 800 MHz.

**Interfaces Software**

Se recomienda a los usuarios tener instalado como navegador web el Mozilla Firefox 30.0, en otro caso utilizar cualquier otro navegador.

**Interfaces de Comunicación**

Conexión a una red de área local (LAN) no menor de 100 Mbps.

## 2.7 Diagrama de Caso de Uso del Sistema

Un diagrama de casos de uso es una forma de diagrama de comportamiento UML mejorado. UML no define estándares para que el formato escrito describa los casos de uso, sin embargo una notación gráfica puede solo dar una vista general simple de un caso de uso o un conjunto de casos de uso. (23)

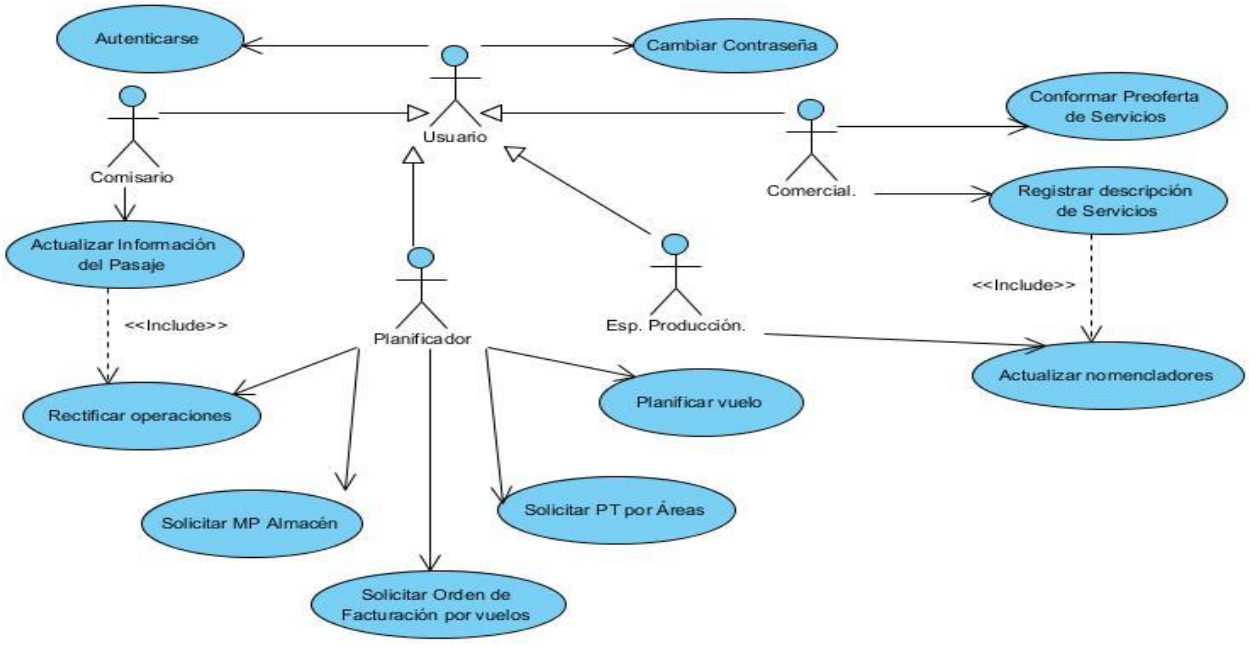


**Fig 6.** Diagrama de Casos de Uso del Sistema

Como se puede observar el sistema está compuesto por un total de 11 casos de usos, de los cuales 6 se dedican a la gestión de la información, 3 a generar reportes y 2 se refieren a las cuentas de usuario.

Los casos de uso que se encargan de la gestión de la información son: Conformar Pre-oferta de Servicios, Registrar descripción de Servicios, Actualizar Nomencladores, Actualizar Información del Pasaje, Rectificar Operaciones y Planificar Vuelo. Los casos de uso correspondientes a la generación de reportes son: Solicitar Materias Prima (MP) del Almacén, Solicitar Plan de Trabajo (PT) por Áreas y Solicitar Orden de Facturación por Vuelos. Los casos de uso referidos a las cuentas de usuarios son: Autenticarse y Cambiar Contraseña.

**Especificaciones de cada caso de uso del sistema (Ver Anexo 1)**

### 2.8 Arquitectura del Sistema

ASP.NET es un marco de desarrollo para la creación de páginas web y sitios web con HTML, CSS, JavaScript y secuencias de comandos del servidor. Es compatible con tres modelos de desarrollo diferentes: Web Page, MVC (Modelo Vista Controlador) y Web Forms. De los tres modelos mencionados anteriormente se determinó que para representar la arquitectura de este sistema el más adecuado era modelo-vista-controlador (MVC), debido a que es un patrón de arquitectura que ayuda a crear una separación lógica entre el modelo (la lógica de acceso a datos), la vista (la lógica de presentación) y el controlador (la lógica de negocio).

Además, ASP.NET MVC 5 incluye:

• ASP.NET Web API, un marco de trabajo para crear y consumir servicios HTTP asequibles para una gran variedad de clientes, incluidos los exploradores, teléfonos y tabletas.

• Páginas web de ASP.NET que ofrecen un modo rápido, asequible y sencillo de combinar código de servidor con HTML para crear contenido web dinámico.

• Optimización web, un marco de trabajo para crear paquetes y minimizar scripts y archivos CSS.

• NuGet, un sistema de administración de paquetes gratuito y de código abierto

centrado en el desarrollador, para intentar simplificar el proceso de incorporación de bibliotecas de terceros en aplicaciones .NET durante el desarrollo en la plataforma .NET. (24)

**MVC**

MVC es uno de los tres modelos de programación ASP.NET que proporciona un control total sobre HTML, CSS y JavaScript. MVC es un marco para la creación de aplicaciones web usando un diseño MVC (Model View Controller):

- El modelo representa el núcleo de la aplicación (por ejemplo una lista de registros de bases de datos).
- La vista muestra los datos (los registros de base de datos).
- El controlador se encarga de la entrada (a los registros de base de datos).

MVC define aplicaciones web con 3 capas lógicas:

- La capa de negocio (Controlador Lógico)
- La capa de presentación (Ver la lógica)
- La capa de acceso a datos (lógica Modelo)

**El modelo** es la parte de la aplicación que se encarga de la lógica para los datos de aplicación. A menudo, los objetos del modelo recuperan datos (y almacenan datos) de una base de datos.

**La vista** es la parte de la aplicación que se encarga de la visualización de los datos. Muy a menudo las vistas se crean a partir de los datos del modelo.

**El controlador** es la parte de la aplicación que se encarga de la interacción del usuario. Típicamente los controladores de leer datos de una entrada de usuario vista, control y enviar los datos de entrada al modelo. (25)

### 2.9 Patrones de diseño

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software, es decir, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Debemos tener presente los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios). (26)

| Nombre del patrón | Características |
|---|---|
| **GRASP** | |
| **Patrón Experto** | Asignan una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. |
| **Patrón Bajo Acoplamiento** | Asignan las responsabilidades de forma tal que las clases se comuniquen con el menor número de clases que sea posible. |
| **Patrón Alta Cohesión** | Asignan a las clases responsabilidades para que trabajen sobre una misma área de la aplicación y que no tengan mucha complejidad.<br>Muy baja cohesión: Una clase es la única responsable de muchas tareas en áreas funcionales muy heterogéneas.<br>Baja cohesión: Una clase tiene la responsabilidad exclusiva de una tarea compleja dentro de un área funcional.<br>Alta cohesión: Una clase tiene responsabilidades moderadas en un área funcional y colabora con las otras para llevar a cabo las tareas. |
| **Patrón Controlador** | Asignan la responsabilidad del manejo de mensajes de los eventos del sistema a una clase. |
| **Patrón Creador** | Asignan a la clase B la responsabilidad de crear una instancia de clase A en uno de los siguientes casos:<br>B agrega los objetos A.<br>B contiene los objetos A.<br>B registra las instancias de los objetos A.<br>B utiliza especialmente los objetos A.<br>B tiene los datos de inicialización que serán transmitidos a A cuando este objeto sea creado (así que B es Experto respecto a la creación de A). B es un creador de los objetos A. |

| GOF | |
|---|---|
| **Iterador** | Proporcionar una forma de acceder secuencialmente a los elementos de un objeto compuesto por agregación sin necesidad de desvelar su representación interna. Se utiliza cuando se quiere acceder a los elementos de un objeto agregado sin mostrar su representación interna y cuando se quieren permitir recorridos múltiples en objetos agregados. |

**Tabla 1.** Patrones de diseño GRASP.

El patrón Experto es uno de los más usados, debido a que se debe tener en cuenta que solo se asignarán responsabilidades a las clases que contengan la información para llevarlas a cabo. En la propuesta de solución el patrón experto se utiliza en la mayoría de los controladores debido a que cada uno de ellos realiza funcionalidades propias, como por ejemplo ClienteController. El patrón Bajo Acoplamiento se utiliza cuando se crea un Modelo Padre, el cual contiene la mayoría de los modelos, esto permite evitar la comunicación directa de los modelos. El patrón Alta Cohesión se emplea en la creación de los modelos, en donde se definen los datos asociados a las tablas de la base de datos, un ejemplo es el modelo Clase que solamente tendrá los datos referentes a la clase. El patrón Controlador se utiliza en cualquier controlador del sistema, los cuales permiten la manipulación de datos y el cumplimiento de todas las funcionalidades. El patrón Creador también se emplea en todos los controladores, puesto que permite crear instancias de cualquier modelo. El patrón Iterador se utiliza en el controlador PlanificadorController en el método ReportResult (), en donde es necesario realizar recorridos múltiples en objetos.

### 2.10 Modelos del Diseño.

El modelo de diseño es un modelo de objetos que:

- ✓ Describe la realización física de los casos de uso.
- ✓ Se centra en cómo los requisitos funcionales y no funcionales tienen impacto en el sistema a considerar.

- ✓ Descompone los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo. (27)

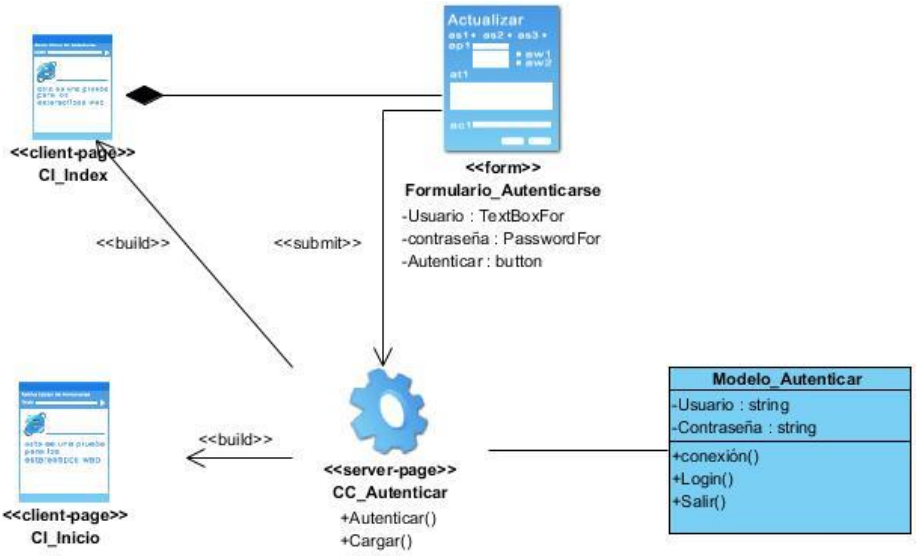


**Diagrama 1. Diagrama de Clase de Diseño. CU_Autenticar.**

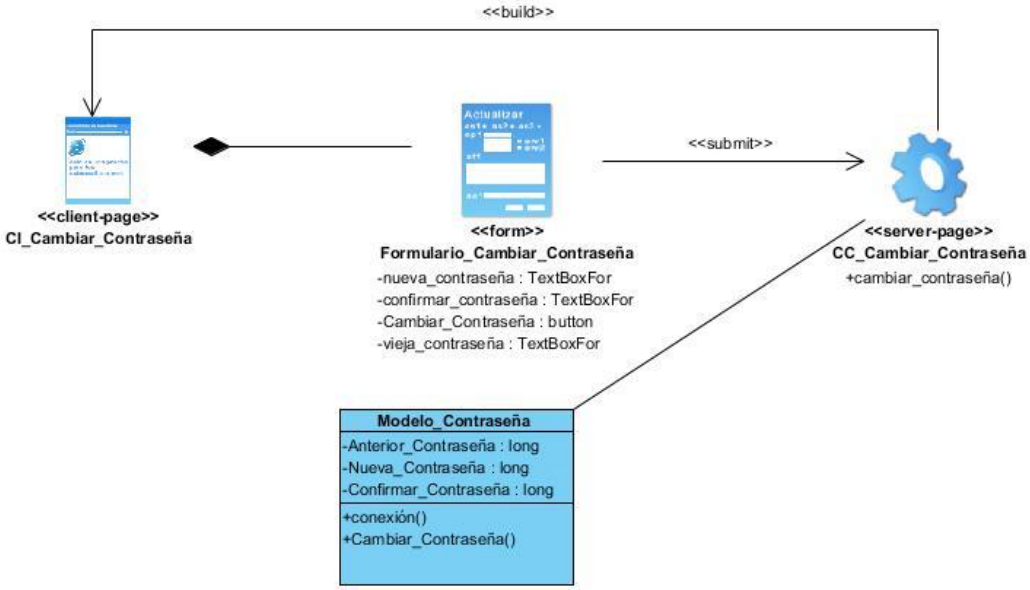


**Diagrama 2. Diagrama de Clase de Diseño. CU_Cambiar_Contraseña.**

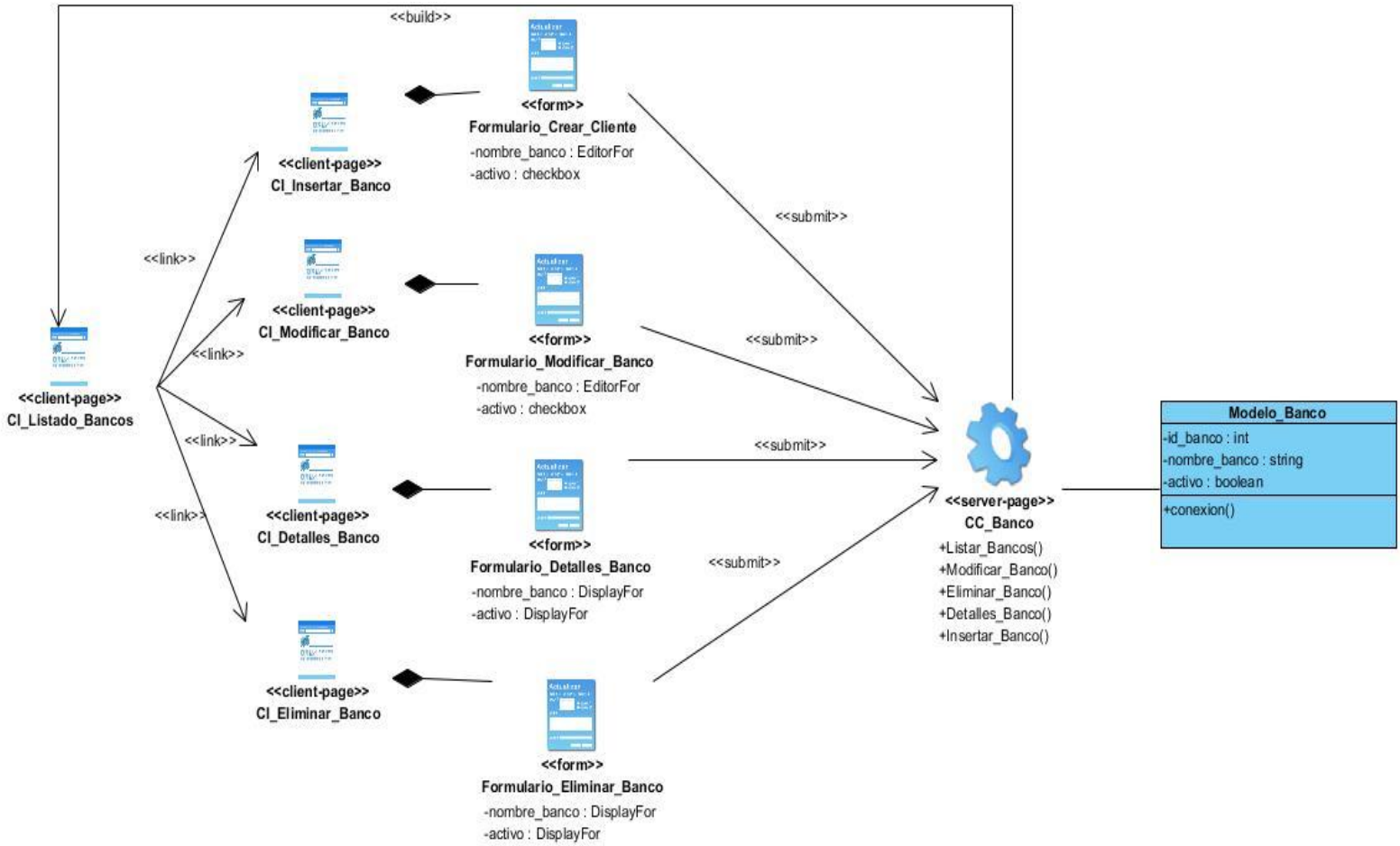


**Diagrama 3. Diagrama de Clase de Diseño. CU_Gestionar_Banco.**

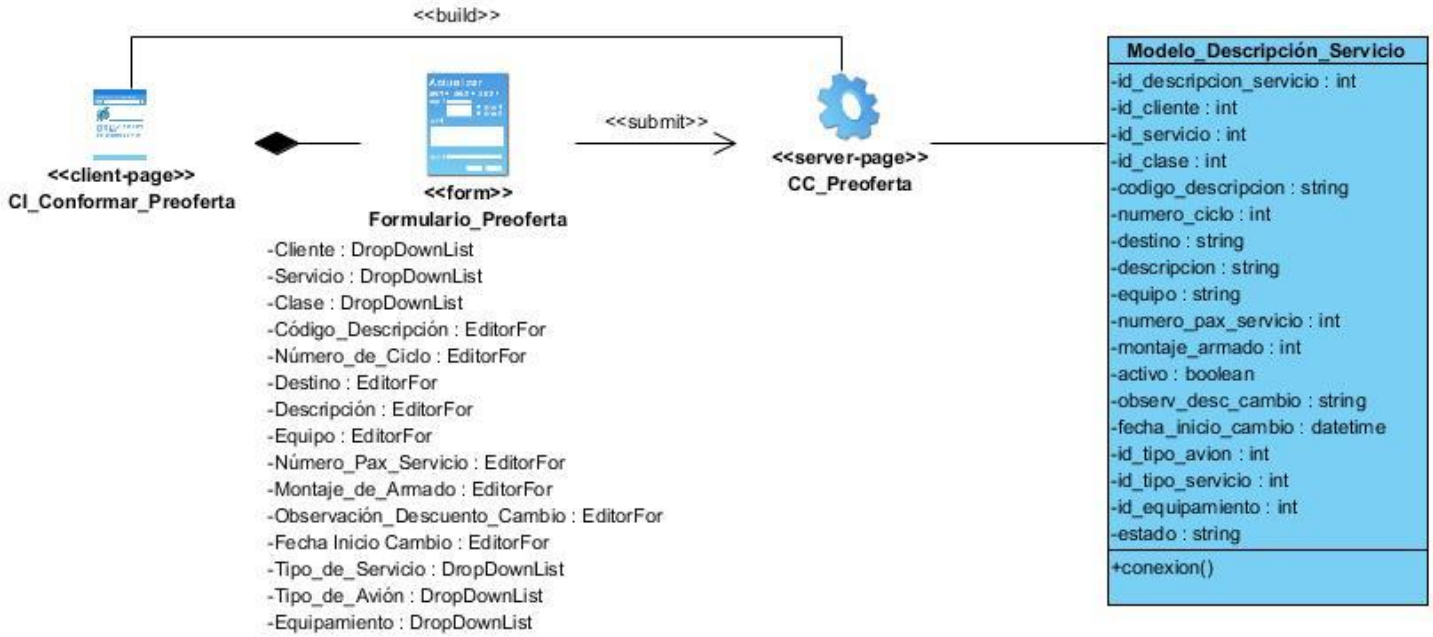


**Diagrama 4. Diagrama de Clase de Diseño. CU_Conformar_Preoferta_Servicio.**

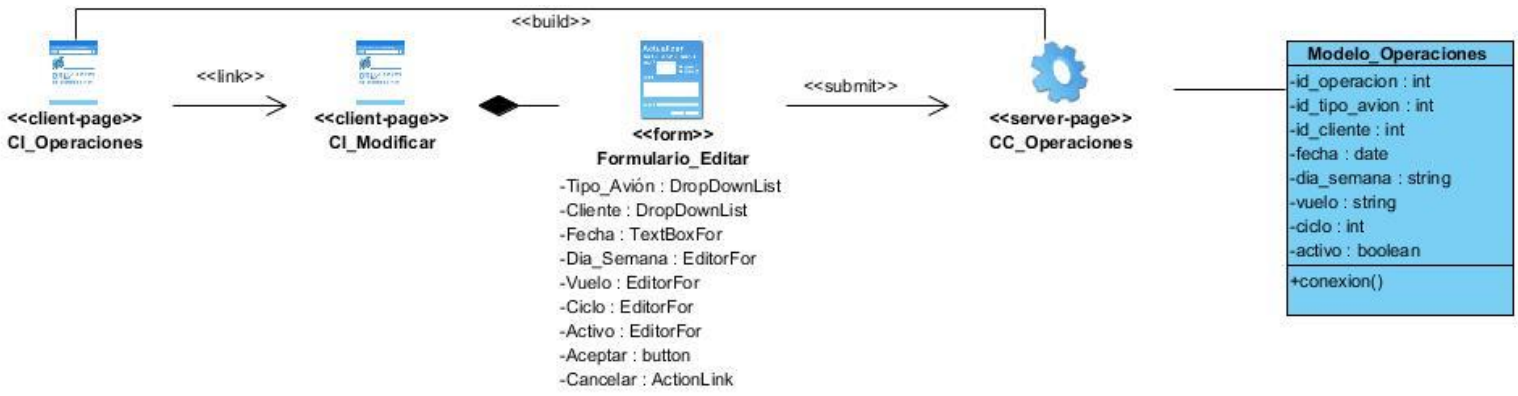


**Fig 5. Diagrama de Clase de Diseño. CU_Actualizar_Información_Pasaje.**

Como se puede observar solo se muestran los diagramas de clase de diseño de cinco casos de uso, los cuales se escogieron por ser los más significativos dentro de la aplicación.

### 2.11 Diseño de la Base de Datos

El modelo lógico de datos provee de una vista de las entidades lógicas de datos y sus relaciones con independencia de la plataforma de base de datos a utilizar. Una base de datos correctamente diseñada permite obtener acceso a información exacta y actualizada. El diseño de la base de datos incluye diferentes modelos y esquemas, como también software de gestión de la misma (SGBD). (28)

La base de datos nombrada Catering presenta un total de 34 tablas.

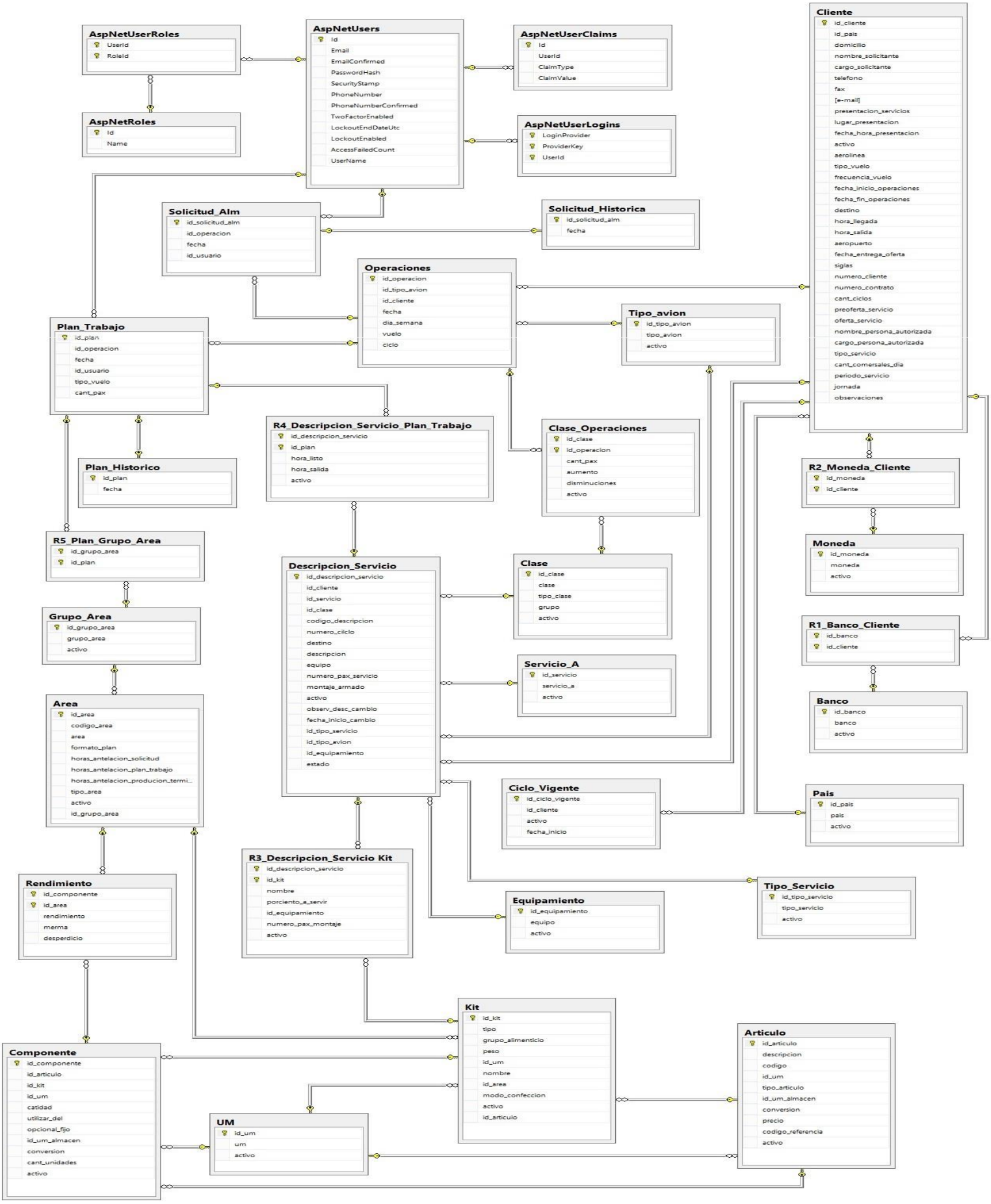
AspNetUserRoles
UserId
RoleId
AspNetUsers
Id
Email
EmailConfirmed
PasswordHash
SecurityStamp
PhoneNumber
PhoneNumberConfirmed
TwoFactorEnabled
LockoutEndDateUtc
LockoutEnabled
AccessFailedCount
UserName
AspNetUserClaims
Id
UserId
ClaimType
ClaimValue
AspNetRoles
Id
Name
AspNetUserLogins
LoginProvider
ProviderKey
UserId
Cliente
id_cliente
id_pais
domicilio
nombre_solicitante
cargo_solicitante
telefono
fax
[e-mail]
presentacion_servicios
lugar_presentacion
fecha_hora_presentacion
activo
aerolinea
tipo_vuelo
frecuencia_vuelo
fecha_inicio_operaciones
fecha_fin_operaciones
destino
hora_llegada
hora_salida
aeropuerto
fecha_entrega_oferta
siglas
numero_cliente
numero_contrato
cant_ciclos
preoferta_servicio
oferta_servicio
nombre_persona_autorizada
cargo_persona_autorizada
tipo_servicio
cant_comersales_dia
periodo_servicio
jornada
observaciones
Solicitud_Alm
id_solicitud_alm
id_operacion
fecha
id_usuario
Solicitud_Historica
id_solicitud_alm
fecha
Operaciones
id_operacion
id_tipo_avion
id_cliente
fecha
dia_semana
vuelo
ciclo
Tipo_avion
id_tipo_avion
tipo_avion
activo
Plan_Trabajo
id_plan
id_operacion
fecha
id_usuario
tipo_vuelo
cant_pax
R4_Descripcion_Servicio_Plan_Trabajo
id_descripcion_servicio
id_plan
hora_listo
hora_salida
activo
Clase_Operaciones
id_clase
id_operacion
cant_pax
aumento
disminuciones
activo
Plan_Historico
id_plan
fecha
R2_Moneda_Cliente
id_moneda
id_cliente
R5_Plan_Grupo_Area
id_grupo_area
id_plan
Moneda
id_moneda
moneda
activo
Descripcion_Servicio
id_descripcion_servicio
id_cliente
id_servicio
id_clase
codigo_descripcion
numero_ciclo
destino
descripcion
equipo
numero_pax_servicio
montaje_armado
activo
observ_desc_cambio
fecha_inicio_cambio
id_tipo_servicio
id_tipo_avion
id_equipamiento
estado
Clase
id_clase
clase
tipo_clase
grupo
activo
Grupo_Area
id_grupo_area
grupo_area
activo
R1_Banco_Cliente
id_banco
id_cliente
Servicio_A
id_servicio
servicio_a
activo
Banco
id_banco
banco
activo
Area
id_area
codigo_area
area
formato_plan
horas_antelacion_solicitud
horas_antelacion_plan_trabajo
horas_antelacion_producion_termi...
tipo_area
activo
id_grupo_area
Ciclo_Vigente
id_ciclo_vigente
id_cliente
activo
fecha_inicio
Pais
id_pais
pais
activo
R3_Descripcion_Servicio Kit
id_descripcion_servicio
id_kit
nombre
porciento_a_servir
id_equipamiento
numero_pax_montaje
activo
Rendimiento
id_componente
id_area
rendimiento
merma
desperdicio
Tipo_Servicio
id_tipo_servicio
tipo_servicio
activo
Equipamiento
id_equipamiento
equipo
activo
Kit
id_kit
tipo
grupo_alimenticio
peso
id_um
nombre
id_area
modo_confeccion
activo
id_articulo
Articulo
id_articulo
descripcion
codigo
id_um
tipo_articulo
id_um_almacen
conversion
precio
codigo_referencia
activo
Componente
id_componente
id_articulo
id_kit
id_um
catidad
utilizar_del
opcional_fijo
id_um_almacen
conversion
cant_unidades
activo
UM
id_um
um
activo

**Fig 6.** Diseño de la Base de Datos.

## Consideraciones Parciales

- En este capítulo se abordaron temas directamente relacionados con la propuesta de solución, como los procesos objeto a automatizar, los cuales reflejan en qué dirección está enfocado el proceso de desarrollo.
- El modelo de dominio del negocio muestra a través de un conjunto de entidades todo el funcionamiento general del negocio.
- Las reglas de negocio, los requisitos funcionales y no funcionales, permitieron obtener un diseño robusto de la aplicación teniendo en cuenta el patrón arquitectónico a utilizar.
- El diagrama de casos de uso del sistema muestra las funcionalidades generales del sistema, así como su relación con cada uno de los actores.
- El diseño de la base de datos aborda cómo se va a comportar el almacenamiento y captura de los datos.

# Capítulo 3: Implementación.

## 3.1 Introducción

En este capítulo se describen los elementos que se utilizaron para llevar a cabo la implementación del sistema. Uno de ellos es el diagrama de componentes el cual muestra cómo se distribuyen los componentes del sistema, también el diagrama de despliegue donde se evidencia la descripción física del sistema. Se mencionan los aspectos relevantes de la implementación, como la estructura y los estándares de codificación. Por último se aplican pruebas de caja negra que permiten garantizar la calidad de la aplicación, dentro de estas se utilizaron las pruebas de aceptación.

## 3.2 Diagrama de Componentes

Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas de Componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema. (29)

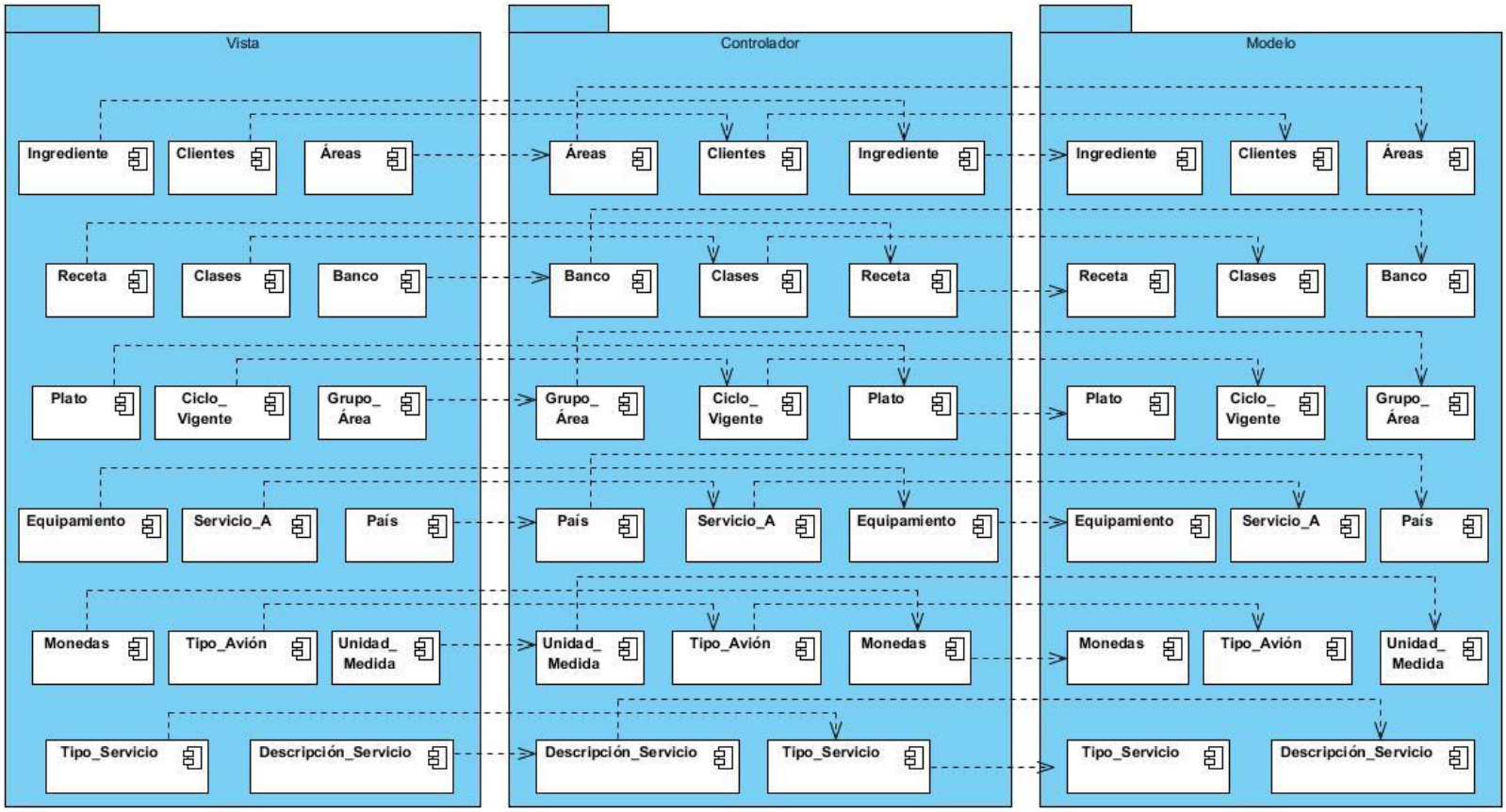


**Fig 8. Diagrama de Componentes.**

### 3.3 Diagrama de Despliegue

El diagrama de despliegue muestra la disposición física de los distintos nodos que componen el sistema y a su vez representan un recurso de cómputo dentro del mismo. En este caso la aplicación se encuentra hospedada en un servidor Web Internet Information Service (IIS) y la misma se comunica mediante ADO, con un servidor de base de datos SQL Server. Se emplea como protocolo de comunicación entre el Cliente y el Servidor Web el HTTPS.

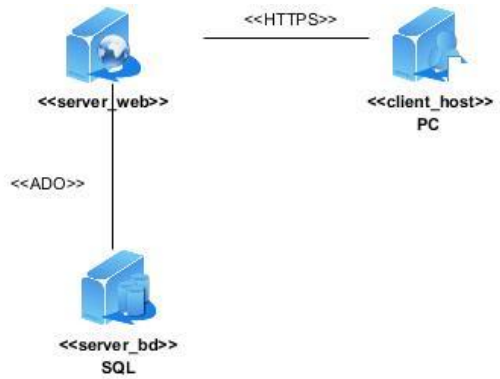


**Fig 9. Diagrama de Despliegue.**

### 3.4 Aspectos relevantes de la implementación

**Estándares de codificación:** Un estándar de codificación completo, comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. (Estándares de Codificación).

El estilo de código a utilizar es:

- ✓ @*Para poner un comentario en la vista*@ y // Para poner un comentario en el controlador o en el modelo.
- ✓ Nombre de las clases comienzan con mayúscula, en caso de que esté compuesto por más de una palabra, también comienza con mayúscula se separan con guión bajo.
- ✓ Todos los campos Id van a comenzar con el identificador (id) seguido del nombre del campo. Ejemplo: id_kit.
- ✓ Atributos de las clases comienzan con minúscula.
- ✓ Cada controlador está compuesto por el nombre que lo identifica que empieza con mayúscula y la palabra Controller que también comienza con mayúscula.

**Estructura de la aplicación:** La propuesta de solución está estructurada por el patrón modelo-vista-controlador el cual se encarga de separar los datos y la lógica de negocio de una aplicación de la interfaz de usuario, para ello existen tres carpetas fundamentales llamadas: *Controllers*, *Models* y *Views*.

La carpeta ***Controllers*** contiene todos los controladores, los cuales responden a eventos para proporcionarle a la 'vista' todos los datos del 'modelo' requerido, por tanto se podría decir que el 'controlador' hace de intermediario entre la 'vista' y el 'modelo'.

Algunos de los ficheros que se encuentran dentro de dicha carpeta son:

- *AccountController.cs:* Clase que se encarga de controlar las cuentas de usuario utilizando dos métodos fundamentales:

*1-Log*: Es el responsable de autenticar un usuario, para esto es necesario crear dos métodos en la clase, uno para cuando el usuario abra la aplicación se muestra la interfaz correspondiente y otro para cuando el usuario presiona el botón Aceptar, este método es el encargado de verificar si el usuario está previamente registrado en la base de datos, si el mismo tiene un rol asignado, así como otras validaciones.

```
// GET: /Account/Login
[AllowAnonymous]
0 references
public ActionResult Log(string returnUrl)
{
    ViewBag.ReturnUrl = returnUrl;
    return View();
}
```

```
// POST: /Account/Login
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
0 references
public async Task<ActionResult> Log(LoginViewModel model, string returnUrl)
{
    if (!ModelState.IsValid)
    {
        return View(model);
    }

    // This doesn't count login failures towards account lockout
    // To enable password failures to trigger account lockout, change to shouldLockout: true
    var result = await SignInManager.PasswordSignInAsync(model.Usuario, model.Password, model.RememberMe, shouldLockout: false);
    switch (result)
    {
        case SignInStatus.Success:
            return RedirectToAction("Index", "Home");
        case SignInStatus.LockedOut:
            return View("Lockout");
        case SignInStatus.RequiresVerification:
            return RedirectToAction("SendCode", new { ReturnUrl = returnUrl, RememberMe = model.RememberMe });
        case SignInStatus.Failure:
        default:
            ModelState.AddModelError("", "Usuario o contraseña incorrecta.");
            return View(model);
    }
}
```

*2-Register: Se encarga de registrar un nuevo usuario con una contraseña, solo puede acceder a esta funcionalidad un usuario que esté autenticado con el rol de Administrador.*

- *Para la gestión de los nomencladores se utiliza un controlador para cada modelo, es decir para cada tabla de la base de datos. Cada uno de estos controladores presentan las mismas funcionalidades debido a que se utiliza el patrón CRUD (Create, Read, Update, Delete) que permite crear, leer, actualizar y eliminar algún elemento de un nomenclador determinado. A continuación se muestra un ejemplo de un controlador que contiene estas cuatro funcionalidades y además realiza un listado de todos los elementos del nomenclador que existen en la base de datos.*

*Área será el nomenclador escogido y el fichero correspondiente a su controlador es AreasController.cs.*

*- El método Index es el responsable de listar todas las áreas existentes en la base de datos, para esto es necesario crear una instancia de la base de datos para poder acceder a sus tablas.*

```
private Catering db = new Catering();


// GET: Areas
0 references
public ActionResult Index()
{
    var areas = db.Areas.Include(a => a.Grupo_Area);
    return View(areas.ToList());
}
```

*-El método Details es el responsable de mostrar todos los datos de un área escogida pasándole por parámetro el id del área y buscando con la función Find (id) el área que se corresponde con el id que manda la vista hacia el controlador. Después de obtener dicha área el método retorna la vista con el área encontrada.*

```
// GET: Areas/Details/5
0 references
public ActionResult Details(int? id)
{
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Area area = db.Areas.Find(id);
    if (area == null)
    {
        return HttpNotFound();
    }
    return View(area);
}
```

*-El método Create es el encargado de insertar una nueva área en la base de datos. Para esto se necesitan dos métodos:*

** Uno para mostrar la vista Create.cshtml.*

```
// GET: Areas/Create
0 references
public ActionResult Create()
{
    ViewBag.id_grupo_area = new SelectList(db.Grupo_Area, "id_grupo_area", "grupo_area1");
    return View();
}
```

```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public ActionResult Create([Bind(Include = "id_area,codigo_area,area1,formato_plan,horas_antelacion_solicitud")] Area area)
{
    if (ModelState.IsValid)
    {
        db.Areas.Add(area);
        db.SaveChanges();
        return RedirectToAction("Index");
    }

    ViewBag.id_grupo_area = new SelectList(db.Grupo_Area, "id_grupo_area", "grupo_area1", area.id_grupo_area);
    return View(area);
}
```

** Otro para insertar una nueva área, de acuerdo a los valores introducidos por el usuario en la vista que van a corresponder con los datos de la tabla.*

*Nota: En la imagen no se pudieron mostrar todos los datos de la tabla Área debido a la longitud, a continuación se mencionan los atributos que van en continuación de* `horas_antelacion_solicitud` *dentro de Include:*

`horas_antelacion_plan_trabajo, horas_antelacion_producion_terminada, tipo_area, activo, id_grupo_area.`

-El método Edit se encarga de editar los valores de un área previamente insertada.

Para esto se crean dos *métodos*:

*Uno para mostrar la vista Edit.cshtml con los datos del área seleccionada. Para mostrar los datos es necesario pasarle por parámetro el id del área, para buscar el área seleccionada y enviarle a la vista sus datos.

```
// GET: Areas/Edit/5
0 references
public ActionResult Edit(int? id)
{
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Area area = db.Areas.Find(id);
    if (area == null)
    {
        return HttpNotFound();
    }
    ViewBag.id_grupo_area = new SelectList(db.Grupo_Area, "id_grupo_area", "grupo_area1", area.id_grupo_area);
    return View(area);
}
```

```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public ActionResult Edit([Bind(Include = "id_area,codigo_area,area1,formato_plan,horas_antelacion_solicitud")] Area
{
    if (ModelState.IsValid)
    {
        db.Entry(area).State = EntityState.Modified;
        db.SaveChanges();
        return RedirectToAction("Index");
    }
    ViewBag.id_grupo_area = new SelectList(db.Grupo_Area, "id_grupo_area", "grupo_area1", area.id_grupo_area);
    return View(area);
}
```

*Otro para modificar los valores de un área determinada, que el usuario haya cambiado en la vista.

*Nota: En la imagen no se pudieron mostrar todos los datos de la tabla Área debido a la longitud, a continuación se mencionan los atributos que van en continuación de* `horas_antelacion_solicitud` *dentro de Include:*

`horas_antelacion_plan_trabajo, horas_antelacion_producion_terminada, tipo_area, activo, id_grupo_area.`

-El método Delete es el responsable de eliminar de la base de dato un área determinada. Para esto es necesario utilizar dos métodos:

*Uno para mostrar la vista Delete.cshtml la cual muestra todos los datos del área, debido a esto es necesario pasarle por parámetro el id del área para poder capturar sus valores y retornarlos en la vista.

```
// GET: Areas/Delete/5
0 references
public ActionResult Delete(int? id)
{
    if (id == null)
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    Area area = db.Areas.Find(id);
    if (area == null)
    {
        return HttpNotFound();
    }
    return View(area);
}
```

```
// POST: Areas/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
0 references
public ActionResult DeleteConfirmed(int id)
{
    Area area = db.Areas.Find(id);
    db.Areas.Remove(area);
    db.SaveChanges();
    return RedirectToAction("Index");
}
```

*Otro para cuando el usuario presione el botón Eliminar el método elimine el área cuyo id sea igual al id pasado por parámetro, guarde los cambios y re-direccione al listado de áreas.

La carpeta ***Models*** contiene clases que representan cada una de las tablas de la base de datos, en las cuales se definen los atributos y algunas validaciones. Es donde se encuentra la información que se utiliza en la vista y se manipulan en el controlador.

A continuación se muestra un ejemplo de cómo se declara un modelo, en este caso es Banco.cs:

```
namespace WebCatering.Models
{
    using System;
    using System.Collections.Generic;
    using System.ComponentModel.DataAnnotations;
    using System.ComponentModel.DataAnnotations.Schema;
    using System.Data.Entity.Spatial;

    [Table("Banco")]
    12 references
    public partial class Banco
    {
        0 references
        public Banco()
        {Clientes = new HashSet<Cliente>();}
        [Key]
        0 references
        public int id_banco { get; set; }
        [Column("banco")]
        [Required]
        [StringLength(50)]
        1 reference
        public string banco1 { get; set; }
        0 references
        public bool activo { get; set; }
        1 reference
        public virtual ICollection<Cliente> Clientes { get; set; }  }  }
```

Cada modelo tiene un campo que representa la llave primaria en la tabla de la base de datos, este campo se define con [Key]. También se pueden especificar los campos obligatorios, mostrar mensajes, entre otras validaciones.

La carpeta ***Views*** contiene todas las interfaces de usuario, donde muestra la información

de uno o varios 'modelos'. Dentro de ella en la carpeta Home se encuentran las cinco interfaces principales, que definen las vistas de cada rol, a continuación se mencionan las mismas: IndexAdmin.cshtml, IndexComercial.cshtml, IndexComisario.cshtml, IndexPlanif.cshtml e IndexProd.cshtml. Cada una de estas vistas utiliza un _Layout, el cual contiene el diseño común de cada una de las funcionalidades que pueda tener un rol. En estas vistas se puede utilizar tanto código HTML como código Razor (cshtml). A continuación se muestra un ejemplo de una vista, en este caso es Index.cshtml que se encuentra dentro de la carpeta Banco.

```
@model IEnumerable<WebCatering.Models.Banco>
@{
    ViewBag.Title = "Banco";
    Layout = "~/Views/Shared/_LayoutComercial.cshtml";
}
<h2 style="margin-left:5%">Banco</h2>
<p style="margin-left:5%">
    @Html.ActionLink("Insertar", "Create")
</p>
<table class="table" style="display:block; margin-left:5%;">
    <tr>
        <th>
            <label>Banco</label>
        </th>
        <th>
            <label>Activo</label>
        </th>
        <th></th>
    </tr>
    @foreach (var item in Model)
    { <tr>
            <td>
                @Html.DisplayFor(modelItem => item.banco1)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.activo)
            </td>
            <td>
                @Html.ActionLink("Editar", "Edit", new { id = item.id_banco }) |
                @Html.ActionLink("Detalles", "Details", new { id = item.id_banco }) |
                @Html.ActionLink("Eliminar", "Delete", new { id = item.id_banco })
            </td>
        </tr> }</table>
```

### 3.5 Pruebas de Software

Las pruebas son elementos críticos que garantizan la calidad de un software, también permite profundizar en los costos que pueden estar asociados a los fallos que pueda poseer un sistema. Dichos costos han permitido la realización de pruebas minuciosas y bien planificadas con el fin de minimizarlos.

En las pruebas de caja negra se observa el resultado obtenido de la llamada a las funciones a probar de acuerdo a los parámetros de entrada, si una función retorna al menos un valor no esperado se procede a su depuración paso a paso para detectar los errores que pueda contener.

**3.5.1 Descripción General.**

El caso de uso se inicia cuando el usuario genérico ejecuta la aplicación y el mismo llena los campos y selecciona la opción Aceptar.

**3.5.2 Condiciones de Ejecución.**

El usuario genérico debe llenar los campos del formulario.

**3.5.3 Secciones a probar en el Caso de Uso.**

| Nombre de la sección | Escenarios de la sección | Descripción de la funcionalidad | Flujo Central |
|---|---|---|---|
| SC 1: Autenticar | EC 1.1: Autenticar | El usuario tendrá la posibilidad de introducir los datos pedidos por la aplicación. | El usuario puede acceder a esta funcionalidad de la siguiente forma:<br>1. Llenar los campos pedidos (Usuario y contraseña).<br>2. Dar clic en la opción aceptar. |
| | EC 1.2: Campos Vacíos. | El usuario genérico deja algunos campos pedidos (usuario y contraseña) sin llenar, por lo cual la aplicación | |
| | EC 1.3: Datos Incorrectos. | El usuario introduce datos incorrectos (usuario o contraseña), por lo que el sistema no permite acceder. | |

**Tabla 2:** Secciones a probar en el Caso de Uso Autenticar.

**3.5.4 Matriz de Datos**

**SC 1: Autenticar**

| Id del escenario | Escenario | V1 | V2 | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| EC 1.1 | Autenticar | *V* | *V* | Una vez introducidos los datos pedidos la aplicación muestra una ventana comenzando la labor del usuario autenticado. | *Satisfactorio* |
| EC 1.2 | *Campos Vacíos* | *I* | *V* | *La aplicación muestra el siguiente mensaje en cada campo: "El campo usuario es obligatorio" o "El campo contraseña es obligatorio".* | *Satisfactorio* |
| | | *V* | *I* | | |
| EC 1.3 | *Datos Incorrectos* | *I* | *V* | *La aplicación muestra el siguiente mensaje: "Usuario o Contraseña incorrecto"* | *Satisfactorio* |
| | | *V* | *I* | | |

**V1- Variable 1 *(Usuario)***
**V2- Variable 2 *(Contraseña)***

### 3.5.5 Descripción General

La aplicación le permitirá al usuario cambiar su contraseña.

### 3.5.6 Condiciones de Ejecución.

Debe haber iniciado la aplicación en cualquier rol y encontrarse en la página Cambiar Contraseña.

### 3.5.7 Secciones a probar en el Caso de Uso.

| **Nombre de la sección** | **Escenarios de la sección** | **Descripción de la funcionalidad** | **Flujo Central** |
|---|---|---|---|
| SC 1: Cambiar Contraseña | *EC 1.1 Cambiar Contraseña* | El usuario tendrá la posibilidad de cambiar su contraseña. | El usuario puede acceder a esta funcionalidad de la siguiente forma:<br>1. Presiona click sobre Hola + el nombre del usuario! que aparece en la parte superior derecha de la aplicación.<br>2. Llena los datos del formulario (contraseña anterior, nueva contraseña y su confirmación).<br>3. Presiona el botón Aceptar. |
| | *EC 1.2 Campos Vacíos.* | El usuario deja algunos campos sin llenar, por lo cual la aplicación no le permite acceder a cambiar su contraseña. | |
| | *EC 1.3 Datos Incorrectos.* | El usuario introduce datos incorrectos, por lo que el sistema no permite acceder a cambiar su contraseña. | |

**Tabla 3:** Secciones a probar en el Caso de Uso Cambiar Contraseña.

### 3.5.8 Matriz de Datos

**SC 1: Cambiar Contraseña**

| Id del escenario | Escenario | V1 | V2 | V3 | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|---|---|
| EC 1.1 | Cambiar Contraseña | *V* | *V* | V | *Una vez introducidos los datos pedidos la aplicación muestra un mensaje "Su contraseña ha cambiado* | *Satisfactorio* |
| EC 1.2 | *Campos Vacíos* | *I* | *I* | *I* | *La aplicación muestra el siguiente mensaje en cada campo: "El campo (V1 o V2 o V3 o V4) es obligatorio", es decir muestra un mensaje por cada* | *Satisfactorio* |
| EC 1.3 | *Datos Incorrectos* | *I* | *V* | *V* | *La aplicación muestra el siguiente mensaje: "Anterior Contraseña incorrecta"* | *Satisfactorio* |
| | | *V* | *I* | V | *La aplicación muestra el mensaje: "La Contraseña y Confirmar Nueva Contraseña no coinciden*." | *Satisfactorio* |
| | | *V* | *V* | I | *La aplicación muestra el mensaje*: *"La nueva contraseña debe tener al menos 2 caracteres especiales."* | *Satisfactorio* |
| | | *V* | *I* | I | *La aplicación muestra el mensaje*: *"La nueva contraseña debe tener al menos 2 caracteres especiales."* | *Satisfactorio* |

**V1- Variable 1 *(Contraseña Anterior)***
***V2*- Variable 2 *(Nueva Contraseña)***
***V3*- Variable 3 *(Confirmar Contraseña)***

### 3.5.9 Descripción General

La aplicación le permitirá al usuario insertar, editar, visualizar sus datos y eliminar un nomenclador determinado.

### 3.5.10 Condiciones de Ejecución.

Debe haber iniciado la aplicación en el rol de Esp. Comercial o Esp. Producción.

### 3.5.11 Secciones a probar en el Caso de Uso.

| Nombre de la sección | Escenarios de la sección | Descripción de la funcionalidad | Flujo Central |
|---|---|---|---|
| SC 1: Insertar | *EC 1.1 Insertar* | El usuario tendrá la posibilidad de insertar | El usuario puede acceder a esta funcionalidad de la siguiente forma: |

| | | | |
|---|---|---|---|
| | | un elemento del nomenclador escogido. | 1. Seleccionando en el menú la opción Nomenclador, la cual contiene un listado de nomencladores, que puede escoger para su gestión.<br>*2.* Introduce todos los datos necesarios del nomenclador.<br>3. Presiona el botón Insertar. |
| | *EC 1.2 Campos Vacíos.* | El usuario deja algunos campos sin llenar, por lo cual la aplicación no le permite insertar. | |
| | *EC 1.3 Datos Incorrectos.* | El usuario introduce datos incorrectos, por lo que el sistema no permite insertar. | |
| SC 2: Editar | *EC 1.4 Editar* | El usuario tendrá la posibilidad de editar un elemento del nomenclador escogido. | El usuario puede acceder a esta funcionalidad de la siguiente forma:<br>1. Seleccionando en el menú la opción Nomenclador, la cual contiene un listado de nomencladores, que puede escoger para su gestión.<br>2. Selecciona la opción Editar de un elemento del listado del nomenclador escogido.<br>3. Modifica los campos deseados.<br>4. Presiona el botón Guardar. |
| | *EC 1.2 Campos Vacíos.* | El usuario deja algunos campos sin llenar, por lo cual la aplicación no le | |
| | *EC 1.3 Datos Incorrectos.* | El usuario introduce datos incorrectos, por lo que el sistema no permite guardar los cambios. | |
| SC 3: Detalles | *EC 1.5 Detalles* | El usuario podrá visualizar todos los datos de un elemento del nomenclador escogido. | El usuario puede acceder a esta funcionalidad de la siguiente forma:<br>1. Seleccionando en el menú la opción Nomenclador, la cual contiene un listado de nomencladores, que puede escoger para su gestión.<br>2. Seleccionando la opción Detalles de un elemento del listado del nomenclador escogido.<br>3. Presiona el botón Cancelar. |

| | | | |
|---|---|---|---|
| SC 4: Eliminar | *EC 1.6* Eliminar | El usuario podrá eliminar un elemento del nomenclador escogido. | El usuario puede acceder a esta funcionalidad de la siguiente forma:<br>1. Seleccionando en el menú la opción Nomenclador, la cual contiene un listado de nomencladores, que puede escoger para su gestión.<br>*2. Seleccionando la opción Eliminar* de un elemento del listado del nomenclador escogido.<br>*3. Presiona el botón Eliminar.* |

**Tabla 4:** Secciones a probar en el Caso de Uso Actualizar Nomencladores.

### 3.5.12 Matriz de Datos

**SC 1: Cambiar Contraseña**

| Id del escenario | Escenario | V1 | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|
| EC 1.1 | *Insertar* | *V* | *Una vez introducidos los datos del nomenclador y se encuentren correctamente, la aplicación re-direcciona al listado de los elementos del nomenclador escogido con el nuevo elemento previamente insertado.* | *Satisfactorio* |
| EC 1.2 | *Campos Vacíos.* | *I* | *La aplicación muestra el siguiente mensaje en cada campo: "El campo (V1) es obligatorio", es decir muestra un mensaje por cada campo que esté vacío.* | *Satisfactorio* |
| EC 1.3 | *Datos Incorrectos* | *I* | *La aplicación muestra un mensaje en cada campo especificando cómo debería introducirlo.* | *Satisfactorio* |
| EC 1.4 | Editar | *V* | *Una vez modificado los datos del nomenclador y se encuentren correctamente, la aplicación redirecciona al listado de los elementos del nomenclador escogido con las modificaciones del elemento.* | *Satisfactorio* |
| EC 1.5 | Detalles | *V* | *Muestra todos los datos de un elemento del listado del nomenclador escogido.* | *Satisfactorio* |
| EC 1.6 | Eliminar | *V* | *Muestra todos los datos de un elemento del listado del nomenclador escogido, además el siguiente mensaje: "Está seguro que desea eliminar este elemento?".* | *Satisfactorio* |

**V1- Variable 1 *(Atributos del Nomenclador)***

***Resultado:***

Se realizaron aproximadamente 17 casos de prueba de los cuales en una primera

iteración se identificaron 11 deficiencias relacionadas con errores de validaciones en los formularios, faltas de ortografías en la base de datos, en una segunda iteración se arreglaron algunas de estas deficiencias quedando solo 3 con respecto a las validaciones de los formularios y en una última iteración se eliminaron todos los errores y todas las funcionalidades se lograron satisfactoriamente.

## Consideraciones Parciales

En este capítulo se diseñaron todos los diagramas de implementación que dieron paso a la programación de los componentes que forman parte de la aplicación y al diseño del diagrama de despliegue. También se mencionaron los aspectos relevantes de la implementación de la aplicación y se realizaron pruebas de software que permitieron comprobar el correcto funcionamiento del sistema, teniendo en cuenta las funcionalidades definidas y las especificaciones de los casos de uso del sistema.

## Conclusiones Generales

- ✓ Se logró como principal resultado el Sistema para la Planificación de la Producción de Catering.
- ✓ Se obtuvo toda la documentación asociada al proceso de desarrollo donde se generaron todos los artefactos y descripciones correspondientes al flujo de trabajo de análisis y diseño.
- ✓ Se hizo el levantamiento de los requerimientos del sistema y se generaron todos los artefactos y descripciones correspondientes al flujo de trabajo de análisis y diseño.
- ✓ Por último se validó el sistema utilizando prueba de software, arrojando resultados satisfactorios y además se realizaron pruebas de aceptación del cliente.
- ✓ El desarrollo del Sistema para la Planificación de la Producción de Catering aportó a la empresa CUBA CATERING S.A una herramienta de trabajo que gestiona de forma segura todo el proceso de planificación de la producción.

**Recomendaciones**

Después de cumplidos los objetivos generales se recomienda:

- ✓ Mejorar el diseño de las páginas de presentación al usuario a personas especializadas en el tema.
- ✓ Seguir trabajando en el perfeccionamiento del sistema, adicionándoles nuevas funcionalidades, de acuerdo a las necesidades de la empresa.
- ✓ Automatizar el proceso de Registrar Solicitud de Servicio en futuras versiones.

# Referencias Bibliográficas

1. **Hernández, María Elena Albert Díaz y Alexis Ametller.** *Diseño de Cuadro de Mando Integral de la Empresa Cubacatering.* La Habana : s.n., 2008.
2. **Gategroup.** Gategourmet. [En línea] [Citado el: 11 de marzo de 2015.] http://www.gategourmet.com/home.
3. **Gustavo Mejia Ricart.** Servair. [En línea] [Citado el: 11 de marzo de 2015.] http://www.servair.com.do/es.
4. **LSG Sky Chefs. [En línea] [Citado el: 11 de marzo de 2015.] http://www.lsgskychefs.com/.**
**5. Chefexact. [En línea] [Citado el: 11 de marzo de 2015.] http://www.chefexact.es/.**
**6. Ramos, Anay Carrillo.** ***Herramienta Multimedia de apoyo a la Enseñanza de la Metodología*** **. La Habana : s.n., 2009.**
**7. software.com.ar. [En línea] 2015. [Citado el: 21 de enero de 2015.] http://www.software.com.ar/visual-paradigm-para-.**
**8. Microsoft Corporation (US). Microsoft.** ***Microsoft SQL Server 2005.*** **[En línea] 2015. [Citado el: Lunes 2 de febrero de 2015.] https://www.microsoft.com/latam/technet/productos/servers/sql/2005/overview.mspx.**
**9. [En línea] 17 de octubre de 2013. [Citado el: 24 de febrero de 2015.] http://blogs.msdn.com/b/somasegar/archive/2013/10/17/visual-studio-2013-available-for-download.aspx .**
**10. Rodríguez, Marco Besteiro y Miguel.** ***Introducción al lenguaje C#.***
**11. Mora, Sergio Luján.** ***Programación en Internet: Clientes Web (libro completo gratuito en pdf.*** **s.l. : Club Universitario., 2002.**
**12. Freeman, Adam.** ***Working with Razor.*** **24 Dec 2013 . pp 95-118.**
**13. Bos, Bert.** ***Descripción de todas las especificaciones del CSS, por la W3C (en inglés).*** **18 de febrero de 2011.**
**14. Mark Otto, Jacob Thornton.** ***Bootstrap 3, el manual oficial.*** **s.l. : Librosweb.**
**15. Gauchat, Juan Diego.** ***El gran libro de HTML5, CSS3 y Javascript.*** **2012.**
**16. Sergio Manzur / Ing. en Tecnologías de Información y Comunicaciones (ITESM Campus Tampico). [En línea] 2015. [Citado el: 19 de junio de 2015.] http://www.mexired.com/blog/que-es-jquery.**
**17. Luis Espino. Taringa! [En línea] [Citado el: 9 de 6 de 2015.] http://www.taringa.net/post/info/18278558/Visual-Studio-es-ahora-de-codigo-libre-y-gratuito.html.**
**18. Ing. Audrey Amador, Mag.** ***06 Reglas de Negocio.pdf(SECURED).***
**19. Kruchten, Philippe.** ***The Rational Unified Process. An Introduction Second Edition*** **. s.l. : Addison-Wesley Object Technology Series, 2000.**
**20. Giraldo, O.P.** ***Ingeniería de Requisitos. Volumen, 13.*** **2007.**
**21. Greene, Andrew Stellman and Jennifer.** ***Applied Software Project Management. Cambridge.*** **s.l. : O'Reilly Media, 2005. ISBN 0-596-00948-8..**
**22. RUMBAUGH, I.J.G.B.J.** ***El Proceso Unificado de Desarrollo de Software. Addison Wesley. Capítulos 7, 8 páginas 125-163, 187-202.*** **2000.**
**23. Jacobson, I., P. Jonsson, M. Christerson and G. Overgaard.** ***Ingeniería de Software Orientada a Objetos - Un acercamiento a través de los casos de uso.*** **s.l. : Addison Wesley Longman, Upper Saddle River, N.J.**
**24. Microsoft.** ***ASP.NET MVC 4 para Visual Studio 2010 SP1 y Visual Web Developer 2010 SP1.*** **[En línea] [Citado el: 24 de febrero de 2015.] http://www.microsoft.com/es-es/download/details.aspx?id=30683.**
**25. W3Schools.com.** ***The MVC Programming Model.*** **[En línea] [Citado el: 24 de febrero de 2015.] http://www.w3schools.com/aspnet/mvc_intro.asp .**
**26. Server., Nicolás Tedeschi es Analista de Sistemas de la Facultad de Ingeniería del Uruguay y se encuentra desarrollando tecnología .net y SQL. ¿Qué es un Patrón de Diseño? [En línea] 2015. [Citado el: 19 de junio de 2015.] https://msdn.microsoft.com/es-es/library/bb972240.aspx.**
**27.** ***Conferencia Modelo del diseño, Ingeniería de Software 2.*** **Curso 20013-20014.**
**28. Universidad FASTA. Diseño de Base de Datos. [En línea] [Citado el: 4 de marzo de 2015.] http://www.ufasta.edu.ar/wp-content/uploads/04-Dise%C3%B1o-de-bases-de-datos.pdf.**
**29. Ambler, Scott W.** ***UML 2 Component Diagram Guidelines.***
**30. Heller, Martin.** ***REST and CRUD: the Impedance Mismatch.*** **s.l. : Developer World. InfoWorld., 29 de enero de 2007.**
**31. Más de 30 aerolíneas en la mira de Cuba Catering.** ***DTCUBA.*** **[En línea] Ministerio de Turismo, Lunes 3 de septiembre de 2007. [Citado el: 20 de enero de 2015.] http://www.dtcuba.com/shownews.aspx?c=23256.**

**32. Jacobson, Ivar y Booch, Grady y Rumbaugh, James.** ***El Proceso Unificado de Desarrollo de Software, Capítulo 8.*** **La Habana : Félix Varela, 2004.**
**33. —.** ***El Proceso Unificado de Desarrollo de Software, Capítulo 9.*** **La Habana : Félix Varela, 2004.**
**34. Sarmiento, Johana. UML: Diagrama de Despliegue. [En línea] 12 de abril de 2013. [Citado el: 15 de abril de 2015.] http://umldiagramadespliegue.blogspot.com/.**
**35. Alvaro López. Significado. [En línea] 12 de abril de 2012 . http://www.significadode.org/nomenclador.htm.**

# Anexos

## Anexo 1. Especificaciones de Casos de Uso del Sistema

**CU 1: Autenticar**

| | |
|---|---|
| **Objetivo** | Permitir que el usuario acceda al sistema con los permisos correspondiente al rol que ocupa dentro del sistema. |
| **Actores** | *Usuario* |
| **Resumen** | El caso de uso se inicia cuando el usuario necesita acceder al sistema. |
| **Complejidad** | *Alta* |
| **Prioridad** | *Crítico* |
| **Precondiciones** | El usuario debe estar registrado en el sistema antes de acceder al mismo. |
| **Poscondiciones** | El usuario pueda acceder a los permisos dados por el responsable. |

| **Flujo de eventos** | | |
|---|---|---|
| **Flujo básico** | | |
| | **Actor** | **Sistema** |
| 1. | Ingresa Usuario y Contraseña.<br><br>1.1 Presiona el botón Aceptar. | 1.2 Verificar las siguientes acciones:<br>1.3- Que no existan campos en blancos.<br>1.4- Que los datos insertados sean correctos. |
| 2. | | 2.1 Accede a la interfaz correspondiente según el rol del usuario autenticado. |
| 3. | | 3.1 Termina Caso de Uso. |
| **Flujos alternos** | | |
| **Nº Evento 1.3< "Campos vacíos" >** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje por cada campo que se encuentre vacío especificando que es obligatorio. |
| **Nº Evento 1.4<"Datos Incorrecto">** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje por cada campo que esté incorrecto, en el caso de la contraseña muestra "Contraseña Errónea" y en el caso del usuario muestra "Usuario Incorrecto". |

**CU 2: Cambiar Contraseña.**

| | |
|---|---|
| **Objetivo** | Permitir que el usuario acceda al sistema con los permisos correspondiente al rol que ocupa dentro del sistema. |
| **Actores** | *Usuario* |
| **Resumen** | El caso de uso se inicia cuando el usuario necesita acceder al sistema. |
| **Complejidad** | *Alta* |
| **Prioridad** | *Crítico* |
| **Precondiciones** | El usuario debe estar autenticado en el sistema. |
| **Poscondiciones** | El usuario cambió su contraseña. |

| **Flujo de eventos** | | |
|---|---|---|
| **Flujo básico** | | |
| | **Actor** | **Sistema** |
| 1. | Da click sobre el hipervínculo que dice Hola + el nombre del usuario! que se encuentra en la parte superior derecha de la aplicación. | 1.2 Muestra la interfaz Cambiar Contraseña. |
| 2. | Introduce la contraseña anterior, la nueva contraseña y su confirmación. | 2.1 Verificar las siguientes acciones:<br>2.2- Que no existan campos en blancos.<br>2.3- Que los datos insertados sean correctos.<br>2.4- Que la nueva contraseña sea igual a su confirmación. |
| 3. | | 3.1 Guarda los Datos. |
| 4. | | 4.1 Termina Caso de Uso. |
| **Flujos alternos** | | |
| **Nº Evento 2.2< “Campos vacíos” >** | | |
| | **Actor** | **Sistema** |
| 2. | | 1.1 Muestra un mensaje por cada campo que se encuentre vacío especificando que es obligatorio. |
| **Nº Evento 2.3<“Datos Incorrecto”>** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje por cada campo que esté incorrecto y especifica cómo debe introducirlo. |
| **Nº Evento 2.4<“Desigual Nueva Contraseña y Confirmar”>** | | |
| | **Actor** | **Sistema** |

| | | |
|---|---|---|
| 1. | | 1.1 Muestra el mensaje “Las contraseñas no coinciden” |

**CU 3: Conformar Pre-oferta de Servicio.**

| | |
|---|---|
| **Objetivo** | Permitir que el usuario pueda conformar una pre-oferta de servicio atendiendo a la solicitud del cliente. |
| **Actores** | *Esp. Comercial* |
| **Resumen** | El caso de uso se inicia cuando el usuario necesita acceder al sistema. |
| **Complejidad** | *Alta* |
| **Prioridad** | *Crítico* |
| **Precondiciones** | El usuario debe estar registrado y autenticado en el sistema con el rol de Esp. Comercial. |
| **Poscondiciones** | El usuario puede visualizar el listado de todas las pre-ofertas ya conformadas. |

| **Flujo de eventos** | | |
|---|---|---|
| **Flujo básico** | | |
| | **Actor** | **Sistema** |
| 1. | Escoge la opción del menú Oferta que incluye Pre-oferta de Servicio. | 1.1 Accede a la interfaz de Pre-oferta de Servicio en donde se encuentra un listado de las pre-ofertas y la opción Conformar Pre-oferta de Servicios. |
| 2. | Selecciona la opción Conformar Pre-oferta de Servicios | 2.1 Muestra la interfaz correspondiente a Conformar Pre-oferta de Servicios con un formulario referente a la información de la pre-oferta. |
| 3. | Ingresa los datos asociados a la pre-oferta (Cliente, Servicio, Clase, Código descripción, Número de ciclo, Destino, Descripción, Equipo, Número de pasajeros, Montaje en armado, Observaciones, Fecha inicio cambio, Tipo de Servicio, Tipo de Avión, Equipamiento).<br><br>3.1 Agrega componentes, ver Sección 1: | 3.3 Muestra la opción escogida:<br>* Si presiona el botón Aceptar:<br>3.4 Verificar las siguientes acciones:<br>- Que no existan campos en blancos.<br>- Que los datos insertados sean correctos.<br>*Si presiona el botón Cancelar, ver Sección 2: Cancelar. |

| | | |
|---|---|---|
| | Agregar Componentes.<br><br>3.2 Presiona el botón Aceptar o Cancelar. | |
| 4. | | 3.1 Guarda los datos. |
| 5. | | 5.1 Termina el caso de uso. |
| **Sección 1: "Agregar Componentes"** | | |
| | **Actor** | **Sistema** |
| 1 | Selecciona el plato que desea insertar a la pre-oferta.<br>1.1 Presiona el botón Agregar. | 1.2 Muestra un listado de componentes agregados. |
| **Sección 2: "Cancelar"** | | |
| | **Actor** | **Sistema** |
| 1 | | 1.1 Re-direcciona a los listados de pre-oferta y descripciones de servicios. |
| **Flujos alternos** | | |
| **Nº Evento 2.4<"Datos Incorrecto">** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo incorrecto especificando como debería introducirlo. |
| **Nº Evento 4<"Buscar Componentes">** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo incorrecto especificando como debería introducirlo. |

**CU 4: Registrar Descripción de Servicios**

| | |
|---|---|
| **Objetivo** | Permitir que el usuario pueda registrar una descripción de servicios a partir de una pre-oferta de servicio. |
| **Actores** | *Esp. Comercial* |
| **Resumen** | El caso de uso consiste en hacer los ajustes finales de una pre-oferta de servicios atendiendo a las especificaciones del cliente. |
| **Complejidad** | *Alta* |
| **Prioridad** | *Crítico* |
| **Precondiciones** | El usuario debe estar registrado y autenticado en el sistema con el rol de Esp. Comercial. |

| | |
|---|---|
| **Poscondiciones** | El usuario puede visualizar el listado de todas las pre-ofertas ya conformadas así como también el listado de las descripciones de servicios. |

| **Flujo de eventos** | | |
|---|---|---|
| **Flujo básico** | | |
| | **Actor** | **Sistema** |
| 1. | Escoge la opción del menú Oferta que incluye Registrar Descripción de Servicio. | 1.1 Accede a la interfaz de Registrar Descripción de Servicio donde muestra un listado de las pre-ofertas conformadas y también un listado de las descripciones de servicios registradas. |
| 2. | Selecciona la opción Registrar de una pre-oferta determinada. | 2.1 Muestra un formulario con todos los datos de la pre-oferta seleccionada (Cliente, Servicio, Clase, Código descripción, Número de ciclo, Destino, Descripción, Equipo, Número de pasajeros, Montaje en armado, Observaciones, Fecha inicio cambio, Tipo de Servicio, Tipo de Avión, Equipamiento). |
| 3. | Puede realizar las siguientes opciones:<br><br>- Realiza las últimas modificaciones y presiona el botón Registrar.<br><br>- Presiona el botón Cancelar. | 3.1 Muestra la opción escogida:<br>- Presiona el botón Registrar:<br>- Verificar que no existan campos en blancos.<br>- Verificar que los datos insertados sean correctos.<br>- Actualiza Nomencladores, ver CU 5: Actualiza Nomencladores.<br>-Re-direcciona a los listados de pre-oferta y descripciones de servicios.<br>-Presiona el botón Cancelar, ver Sección 1: Cancelar. |
| 4. | | 4.1 Guarda los datos. |
| 5. | | 5.1 Termina el caso de uso. |
| **Sección 1: “Cancelar”** | | |
| | **Actor** | **Sistema** |
| 1 | | 1.1 Re-direcciona a los listados de pre-oferta y descripciones de servicios. |
| **Flujos alternos** | | |

| Nº Evento 3.1<“Campos Vacío”> | | |
|---|---|---|
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo vacío especificando que es obligatorio |
| **Nº Evento 3.1<“ Datos Incorrecto”>** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo incorrecto especificando como debería introducirlo. |

**CU 5: Actualizar Nomencladores.**

El caso de uso Actualizar Nomencladores se refiere a la gestión de todos los nomencladores, en total 16. Todos tienen el mismo comportamiento por lo que se va a ejemplificar uno para brindar una muestra de su funcionamiento. En cada uno de ellos se utilizó el patrón **CRUD,** sus siglas significa Crear, Obtener, Actualizar y Borrar (del original en inglés: **C**reate, **R**ead, **U**pdate and **D**elete). Se usa para referirse a las funciones básicas en bases de datos o la capa de persistencia en un software. (30)

**Gestionar Banco**

| | |
|---|---|
| **Objetivo** | Permitir que el usuario pueda insertar, modificar, eliminar y visualizar los datos de un banco. |
| **Actores** | *Esp. Comercial* |
| **Resumen** | El caso de uso consiste en actualizar o visualizar la información referente a un banco, ya sea insertar, modificar o eliminar. |
| **Complejidad** | *Alta* |
| **Prioridad** | *Crítico* |
| **Precondiciones** | El usuario debe estar registrado y autenticado en el sistema con el rol de Esp. Comercial. |
| **Poscondiciones** | El usuario actualizó o visualizó la información referente a un banco. |
| **Flujo de eventos** | |
| **Flujo básico** | |

| | **Actor** | **Sistema** |
|---|---|---|
| 1. | Escoge la opción del menú Nomencladores que incluye Banco. | 1.1 Accede a la interfaz con un listado de Bancos. |

| | | |
|---|---|---|
| 2. | Escoge la opción:<br><br>-Insertar.<br><br>-Editar.<br><br>-Detalles.<br><br>-Eliminar. | 2.1 Muestra la opción:<br><br>-Insertar, ver Sección 1: Insertar Banco.<br><br>-Editar, ver Sección 2: Editar Banco.<br><br>-Detalles, ver Sección 3: Detalle de un Banco.<br><br>-Eliminar, ver Sección 4: Eliminar Banco. |
| 3. | | 3.1 Guarda los datos. |
| 4. | | 4.1 Termina el caso de uso. |
| **Sección 1: "Insertar Banco"** | | |
| | **Actor** | **Sistema** |
| 1 | | 1.2 Muestra la interfaz Insertar que tiene un formulario con el dato necesario de un banco (nombre del banco). |
| 2 | Llena el formulario con el dato del banco (nombre del banco, activo).<br>2.1Puede escoger las siguientes opciones:<br>- Presiona el botón Insertar.<br>- Presiona el botón Cancelar. | 2.2 Muestra la opción escogida:<br>- Presiona el botón Insertar: Valida que los campos no estén vacíos, que estén correctos y re-direcciona para el listado de los banco mostrando el nuevo banco que agregó.<br>-Presiona el botón Cancelar, ver Sección 5: Cancelar. |
| **Sección 2: "Editar Banco"** | | |
| | **Actor** | **Sistema** |
| 1 | | 1.2 Muestra la interfaz Editar con un formulario que tiene el dato de un banco (nombre del banco, activo) a modificar. |
| 2 | Modifica el nombre del banco.<br>2.1Puede escoger las siguientes opciones:<br>-Presiona el botón Guardar.<br>-Presiona el botón Cancelar | 2.2 Guarda los Cambios.<br>2.3 Muestra la opción escogida:<br>- Presiona el botón Guardar: Re-direcciona para el listado de los banco mostrando el nuevo banco que agregó.<br>- Presiona el botón Cancelar, ver Sección 5: Cancelar. |
| **Sección 3: "Detalles de un Banco"** | | |
| | **Actor** | **Sistema** |
| 1 | | 1.2 Muestra la interfaz Detalles la cual muestra los datos del banco (nombre del banco, activo). |
| 2 | Visualiza la información del banco seleccionado (nombre del banco, activo).<br>2.1Puede escoger las siguientes opciones: | 2.2 Muestra la opción escogida:<br>- Presiona el Botón Editar, ver Sección 2: Editar Banco. |

| | | |
|---|---|---|
| | - Presiona el Botón Editar.<br>- Presiona el Botón Cancelar. | - Presiona el Botón Cancelar, ver Sección 5: Cancelar. |
| **Sección 4: "Eliminar Banco"** | | |
| | **Actor** | **Sistema** |
| 1 | | 1.2 Muestra la interfaz Eliminar con un mensaje "Está seguro que desea eliminar este banco" junto con los datos del banco seleccionado (nombre del banco, activo). |
| 2 | Visualiza la información del banco seleccionado (nombre del banco, activo).<br>2.1Puede escoger las siguientes opciones:<br>- Presiona el Botón Eliminar.<br>- Presiona el Botón Cancelar. | 2.2 Muestra la opción escogida:<br>- Presiona el Botón Eliminar: Elimina el banco escogido y re-direcciona para el listado de los bancos.<br>- Presiona el Botón Cancelar, ver Sección 5: Cancelar. |
| **Sección 5: "Cancelar"** | | |
| | **Actor** | **Sistema** |
| 1 | | 1.2 Re- direcciona para el listado de los bancos. |
| **Flujos alternos** | | |
| **Sección 1: "Insertar Banco"** | | |
| **Nº Evento 2.2<"Campo vacío">** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo vacío especificando que es obligatorio. |
| **Nº Evento 2.2<" Dato Incorrecto">** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo incorrecto especificando como debería introducirlo. |

**CU 6: Actualizar Información del Pasaje.**

| | |
|---|---|
| **Objetivo** | Permitir que el usuario pueda actualizar y confirmar las operaciones diarias de cada vuelo. |
| **Actores** | *Comisario.* |
| **Resumen** | El caso de uso consiste en actualizar y confirmar la información referente al pasaje. |
| **Complejidad** | *Alta* |
| **Prioridad** | *Crítico* |
| **Precondiciones** | El usuario debe estar registrado y autenticado en el sistema con el rol de Comisario. |

| | |
|---|---|
| **Poscondiciones** | El usuario actualizó y confirmar la información referente al pasaje. |

| **Flujo de eventos** | | |
|---|---|---|
| **Flujo básico** | | |
| | **Actor** | **Sistema** |
| 1. | Escoge la opción del menú Actualizar Pasaje. | 1.1 Muestra la interfaz correspondiente a la opción Actualizar Pasaje donde se encuentran todas las operaciones del día listadas. |
| 2. | Selecciona una de las opciones siguientes:<br>- Presiona Editar una operación determinada.<br>- Presiona Eliminar una operación determinada. | 2.1 Muestra la interfaz correspondiente a la opción escogida:<br>- Presiona Editar, ver Sección 1: Editar Operación.<br>- Presiona Eliminar, ver Sección 2: Eliminar Operación. |
| 3. | | 3.1 Guarda los datos. |
| 4. | | 4.1 Termina el caso de uso. |
| **Sección 1: "Editar Operación"** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1Muestra la interfaz Editar con los datos de la operación seleccionada (Tipo de Avión, Cliente, Fecha, Día de la Semana, Vuelo, Ciclo, Activo). |
| 2 | Modifica los datos deseados (Tipo de Avión, Cliente, Fecha, Día de la Semana, Vuelo, Ciclo, Activo) y presiona el botón Guardar. | 2.1 Guarda los datos de la operación y re-direcciona hacia la interfaz con el listado de las operaciones. |
| **Sección 2: "Eliminar Operación"** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1Muestra la interfaz Eliminar con los datos de la operación seleccionada (Tipo de Avión, Cliente, Fecha, Día de la Semana, Vuelo, Ciclo, Activo). |
| 2 | Modifica los datos de la operación y presiona el botón Eliminar. | 2.1 Elimina la operación seleccionada y re-direcciona hacia la interfaz con el listado de las operaciones. |
| **Flujos alternos** | | |
| **Nº Evento 2.2<"Campo vacío">** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo vacío especificando que es obligatorio. |

| | | |
|---|---|---|
| | | |
| **Nº Evento 2.2<“ Dato Incorrecto”>** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo incorrecto especificando como debería introducirlo. |

**CU 7: Planificar Vuelo.**

| | |
|---|---|
| **Objetivo** | Permitir que el usuario pueda insertar las operaciones de cada vuelo. |
| **Actores** | *Planificador.* |
| **Resumen** | El caso de uso consiste en insertar las operaciones de cada vuelo. |
| **Complejidad** | *Alta* |
| **Prioridad** | *Crítico* |
| **Precondiciones** | El usuario debe estar registrado y autenticado en el sistema con el rol de Planificador. |
| **Poscondiciones** | El usuario insertó las operaciones de cada vuelo. |

| **Flujo de eventos** | | |
|---|---|---|
| **Flujo básico** | | |
| | **Actor** | **Sistema** |
| 1. | Escoge la opción del menú Planificar Vuelo. | 1.1 Muestra la interfaz correspondiente a la opción Planificar Vuelo que contiene el listado de los vuelos planificados. |
| 2. | Selecciona la opción Insertar. | 2.1. Muestra la interfaz Insertar con los datos necesarios de una operación (Tipo de Avión, Cliente, Fecha, Día de la Semana, Vuelo, Ciclo, Activo).. |
| 3. | Introduce los datos correspondientes a una operación.<br><br>3.1 Puede escoger las siguientes opciones:<br><br>- Presiona el botón Aceptar.<br><br>- Presionar el botón Cancelar. | 3.2 Muestra la interfaz de acuerdo a la opción escogida:<br>- Presiona el botón Aceptar:<br>- Verificar que no existan campos en blancos.<br>- Verificar que los datos insertados sean correctos.<br>- Re-direcciona a la interfaz del listado de las operaciones.<br>- Presiona el botón Cancelar, ver Sección 1: Cancelar. |

| | | |
|---|---|---|
| 4. | | 4.1 Guarda los datos. |
| 5. | | 5.1 Termina el caso de uso. |
| **Sección 1: ”Cancelar”** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Re-direcciona a la interfaz del listado de las operaciones. |
| **Flujos alternos** | | |
| **Nº Evento 3.2<“Campo vacío”>** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo vacío especificando que es obligatorio. |
| **Nº Evento 3.2<“ Dato Incorrecto”>** | | |
| | **Actor** | **Sistema** |
| 1. | | 1.1 Muestra un mensaje en el campo incorrecto especificando como debería introducirlo. |