CAPÍTULO I

EL PROBLEMA

1.1. Tema de Investigación

"DESARROLLO DE UN SISTEMA DE INFORMACIÓN DE GESTIÓN DE LAS CARPETAS DE ASPIRANTES A LA QUITO MOTORS"

1.2. Planteamiento del Problema

1.2.1. Contextualización

El sistema planteado de automatizar la gestión de carpetas a los aspirantes a la Quito Motors es para facilitar el manejo de la información con que se trabaja en la empresa, debido a que se lleva manualmente.

Algunos aspectos y características de los sistemas de gestión automatizada que funcionan bajo el modelo cliente-servidor:

- Los sistemas de automatización de bibliotecas y archivos, funcionan bajo la arquitectura cliente-servidor.
- Los sistemas requieren de la creación de una base de datos en blanco que soporta la creación de tablas.

En general cualquier sistema que se pretende implementar en la actualidad es de mucha utilidad en cualquier campo o departamento de las empresas,

esto es debido a los cambios tecnológicos que día a día van evolucionando y se pretende automatizar todo lo que se pueda.

1.2.2. Análisis Crítico

El origen del problema se da, cuando existe algún puesto vacante en la empresa, en donde cualquier persona que cumpla con los requisitos del puesto puede aplicar, pero con la condición de que en el lugar donde aplica solo ahí puede averiguar los resultados del puesto y en esa sucursal será donde trabajará, debido a que se lleva la información manualmente es decir en libros, sin embargo esto no hubiera mucho problema si no fuera una empresa muy grande, pero no es el caso se trata de una las empresas más grandes del país que tiene sucursales en cada provincia y es muy difícil manejar toda la información de todas las sucursales por ende se maneja la información en cada sucursal, tanto de los empleados como de los aspirantes, también se les puede considerar a los mismos empleados como aspirantes todo depende de las políticas de la empresa.

Se pretende manejar toda la información de los empleados y aspirantes desde la matriz (Quito). Las decisiones para algún puesto son tomadas por la Lcda. Patricia Redin Jefe de Recursos Humanos, quien es la encargada de enviar la información a cada Jefe de Departamento para que ellos evalúen, y aprueben ha dicho candidato al puesto.

La persona más perjudicada es el Jefe de Recursos Humanos debido que tiene que estar revisando carpeta por carpeta para saber de qué área ha sido solicitada la información, es muy complicado revisando las carpetas, debe tener en cuenta que un empelado también está participando para el puesto, además tener la paciencia necesaria y una buena habilidad de lectura, brindar una buena atención, toda institución pública o privada debe brindar un buen servicio a los requerimientos de los diferentes departamentos y a

su vez a los usuarios que hacen de su uso, esto es el caso del Departamento de Recursos Humanos de cualquier institución.

1.2.3. Prognosis

En caso que el sistema no sea desarrollado la empresa seguirá llevando la información individual mente en cada sucursal y después enviar a la matriz, provocando el mismo problema de no poder llevar un control adecuado de la información de los empleados y aspirantes, es decir no se podrá brindar un servicio más, esto es un problema para la cada sucursal que tienen que esperar los resultados desde la matriz, cuando soliciten un nuevo empleado además este problema podría afectar a la empresa.

1.3. Formulación del Problema

¿Cómo identificar las formas de llevar Información de los aspirantes?

1.3.1. Preguntas Directrices

a. ¿Analizar el manejo de la información que tiene la empresa?
b. ¿Investigar el origen del problema que está afectando a la empresa?
c. ¿Plantear la solución del problema?
d. ¿Estudiar las herramientas necesarias que se utilizarán en su desarrollo?
e. ¿Implantar el sistema y verificar que los resultados obtenidos sean los requeridos?

1.3.2. Delimitación del Tema

Tema: Desarrollo de un Sistema de Información De Gestión de las carpetas de Aspirantes a la Quito Motor S.A.C.I, ubicado en las Av.10 de Agosto, Provincia de Pichincha, en el periodo comprendido entre Enero 2010 – Mayo 2011.

1.4. Justificación

Se realizó el estudio y las investigaciones necesarias en la empresa de todo su manejo, por lo que se ha llegado a la conclusión de que en el Departamento de Recursos Humanos necesita el desarrollo de un Sistema de Automatización de Gestión de carpetas a los Aspirantes a la Quito Motor seria de gran ayuda, especialmente para el personal encargado de recibir y analizar las carpetas de los aspirantes y empleados.

El motivo para el desarrollo del sistema es mantener la información de los aspirantes y empleados ordenados de tal forma de que al momento de hacer una consulta se pueda obtener los datos requeridos y necesarios, de tal manera que al momento de hacer la consulta sobre el puesto de trabajo por el cual están aplicando. La empresa es una de las más grandes del país por lo que no puede quedarse atrás de la tecnología y poco a poco debe ir evolucionando no solo por la empresa sino por brindar un mejor servicio a la ciudadanía.

El principal interés es controlar el ingreso de información de las carpetas para los diferentes puestos de trabajo de los diferentes departamentos con los que cuenta la empresa, además de almacenar la información de forma más segura y automatizada, y no se corra el riesgo de que estos se pierdan. Se capacito al personal propio de la empresa que utilizara el sistema, puesto que ellos ya están familiarizados con el entorno de trabajo.

La importancia de toda empresa es seguir cada día avanzando en sus negocios, sin dejar de lado a la tecnología, para seguir siendo una de las empresas más grandes del país por el reconocimiento de la ciudadanía y de su país. No se contará con ningún tipo de publicidad, radio, televisión. La novedad que ofrece es poder tener a todos los empleados y aspirantes en sistema almacenados de todas las sucursales y no manejar por independiente brindando así una opción más rápida de manejar vía web.

Las utilidades o beneficios que se puede tener en la empresa con el desarrollo del sistema se podrá observar cuando la persona encargada de administrar el sistema obtenga los datos requeridos acerca de las personas que aplican para el puesto, la forma de atención sea más rápida, sin pérdida de tiempo debido a que sus datos se encuentran almacenados en la base de datos y no se tendrá que revisar manualmente.

Se puede contar con el ahorro de tiempo que es un factor muy importante para cualquier empresa debido a que se podrá brindar una mejor atención a más clientes. El impacto que se podría tener es la rapidez tanto de los empleados en la forma de atender como de los aspirantes en averiguar sobre su carpeta, esto es una gran ayuda que ofrece el departamento de recursos humanos.

Existe la posibilidad de que en un futuro la empresa siga siendo reconocida por la ciudadanía de todo el país como es en la actualidad, se debe ir automatizando a cada departamento con el pasar del tiempo, pero sin dejar de lado los avances tecnológicos que día a día siguen mejorando y hacer uso de estas herramientas..

1.5. Objetivos de la Investigación

1.5.1. Objetivos Generales

Desarrollar e Implantar un Sistema de Automatización para la gestión de manejo del Departamentos de Recursos Humanos que permita solucionar el control del manejo de información llegando a satisfacer las necesidades de los empleados del departamento de la empresa Quito Motors. S.A.C.I.

1.5.2. Objetivos Específicos

a. Identificar las causas del problema, por la que está cruzando la empresa.

b. Realizar un estudio previo de la información del departamento de recursos humanos.
c. Manejar información del departamento de modo fiable, seguro y oportuno.
d. Implantar y verificar el Sistema.

CAPÍTULO II

MARCO TEÓRICO

2.1 Antecedentes Investigativos

Se han realizó las investigaciones en la Empresa Quito Motors como es la administración de cada departamento llegando así a determinar, cuáles fueron los problemas por la que están afectando a la empresa, los inconvenientes existen en cada departamento se necesita de un sistema de automatización pero nos enfocaremos en el de recursos humanos que es el control de gestión de las carpetas de aspirantes y empleados a las diversas sucursales y a la matriz, además se han realizado investigaciones con problemas similares, llegando así a determinar que hasta para una empresa pequeña es necesario que exista un sistema de automatización de los empleados, con mucha más razón para un empresa grande .

2.2 Fundamentación

2.2.1 Fundamentación Legal

La Empresa Quito Motors cuenta con licencia de software por lo que, la institución no podría tener problemas al momento de implantar el sistema.

2.2.2 Fundamentación Teórica

Figura Nº 2.1: Gráfico GCA (Sistema de Información de Automatización)

¿Qué es un GCA?

GCA es el nombre con el cual se identificara al software. Un Sistema de Información de Automatización y gestión de las carpetas de los aspirantes a la Quito Motors y todas sus sucursales atreves de la web, es una integración organizada de toda sus compañías y sucursales, diseñado para administrar, almacenar, manipular, analizar y desplegar en todas sus formas la información de las personas que vengan a dejar sus carpetas de Información, referenciada con el fin de resolver problemas complejos de planificación y gestión.

Un sistema de información es un conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio.

- **El Equipo Computacional**: El hardware necesario para que el sistema de información pueda operar.

- **El Recurso Humano**: Quién interactúa con el Sistema de Información, el cual está formado por las personas que utilizan el sistema.

Funciones del GCA

Un sistema de información realiza cuatro actividades básicas:

- Entrada
- Almacenamiento
- Procesamiento y
- Salida de información.

Entrada de Información: Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos. Esto último se denomina interfaces automáticas.

Almacenamiento de Información: El almacenamiento es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos.

Procesamiento de Información: Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida. Estos cálculos pueden efectuarse con datos

introducidos recientemente en el sistema o bien con datos que están almacenados.

Salida de Información: La salida es la capacidad de un sistema de información para sacar la información procesada o bien datos de entrada al exterior. Debe desplegar la información de las personas que están registradas como aspirantes para los diferentes departamentos con que cuenta la empresa.

Actividades que realiza un Sistema de Información:

Entradas:

- Datos generales del cliente: nombre, dirección, tipo de cliente, etc.
- Políticas de créditos: límite de crédito, plazo de pago, etc.
- Facturas (interface automático).
- Pagos, depuraciones, etc.

Proceso:

- Cálculo de antigüedad de saldos.
- Cálculo de intereses moratorios.
- Cálculo del saldo de un cliente.

Almacenamiento:

- Movimientos del mes (pagos, depuraciones).
- Catálogo de clientes.
- Facturas.

Salidas:

- Reporte de pagos.
- Estados de cuenta.
- Pólizas contables (interface automática)
- Consultas de saldos en pantalla de una terminal.

Bases de Datos

Concepto de una Base de Datos

Una base de datos es un almacén que nos permite guardar grandes cantidades de información de forma organizada para que luego podamos encontrar y utilizar fácilmente. El término de bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, USA.

Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada ó estructurada.

Desde el punto de vista informático, la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos. Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas.

Las columnas guardan una parte de la información sobre cada elemento que queramos guardar en la tabla, cada fila de la tabla conforma un registro.

Definición de una Base de Datos

Se define una base de datos como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular.

Característica de una Base de Datos

Entre las principales características de los sistemas de base de datos podemos mencionar:

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Consultas complejas optimizadas.
- Seguridad de acceso y auditoria.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

Ventajas de una Base de Datos

- **Control sobre la redundancia de datos:** Los sistemas de ficheros almacenan varias copias de los mismos datos en ficheros distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar la falta de consistencia de datos.- En los sistemas de bases de datos todos estos ficheros están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una base de datos no se puede eliminar la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos.

- **Consistencia de datos:** Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema puede encargarse de garantizar que todas las copias se mantienen consistentes.
- **Compartición de datos:** En los sistemas de ficheros, los ficheros pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de bases de datos, la base de datos pertenece a la empresa y puede ser compartida por todos los usuarios que estén autorizados.
- **Mantenimiento de estándares:** Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales. Estos estándares pueden establecerse sobre el formato de los datos para facilitar su intercambio, pueden ser estándares de documentación, procedimientos de actualización y también reglas de acceso.
- **Mejora en la integridad de datos:** La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el SGBD (Sistema de Gestión de Bases de Datos) quien se debe encargar de mantenerlas.
- **Mejora en la seguridad:** La seguridad de la base de datos es la protección de la base de datos frente a usuarios no autorizados. Sin unas buenas medidas de seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de ficheros.
- **Mejora en la accesibilidad a los datos:** Muchos SGBD proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos,

sin que sea necesario que un programador escriba una aplicación que realice tal tarea.

- **Mejora en la productividad:** El SGBD proporciona muchas de las funciones estándar que el programador necesita escribir en un sistema de ficheros. A nivel básico, el SGBD proporciona todas las rutinas de manejo de ficheros típicas de los programas de aplicación.- El hecho de disponer de estas funciones permite al programador centrarse mejor en la función específica requerida por los usuarios, sin tener que preocuparse de los detalles de implementación de bajo nivel.
- **Mejora en el mantenimiento:** En los sistemas de ficheros, las descripciones de los datos se encuentran inmersas en los programas de aplicación que los manejan. Esto hace que los programas sean dependientes de los datos, de modo que un cambio en su estructura, o un cambio en el modo en que se almacena en disco, requiere cambios importantes en los programas cuyos datos se ven afectados.- Sin embargo, los SGBD separan las descripciones de los datos de las aplicaciones. Esto es lo que se conoce como independencia de datos, gracias a la cual se simplifica el mantenimiento de las aplicaciones que acceden a la base de datos.
- **Aumento de la concurrencia:** En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información o se pierda la integridad. La mayoría de los SGBD gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.
- **Mejora en los servicios de copias de seguridad:** Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos.

Desventaja de una Base de Datos

- **Complejidad:** Los SGBD son conjuntos de programas que pueden llegar a ser complejos con una gran funcionalidad. Es preciso comprender muy bien esta funcionalidad para poder realizar un buen uso de ellos.
- **Coste del equipamiento adicional:** Tanto el SGBD, como la propia base de datos, pueden hacer que sea necesario adquirir más espacio de almacenamiento. Además, para alcanzar las prestaciones deseadas, es posible que sea necesario adquirir una máquina más grande o una máquina que se dedique solamente al SGBD. Todo esto hará que la implantación de un sistema de bases de datos sea más cara.
- **Vulnerable a los fallos:** El hecho de que todo esté centralizado en el SGBD hace que el sistema sea más vulnerable ante los fallos que puedan producirse. Es por ello que deben tenerse copias de seguridad (Backup respaldos).

Modelo Entidad-Relación

Los diagramas o modelos entidad-relación son una herramienta para el modelado de datos de un sistema de información.

Estos modelos expresan entidades relevantes para un sistema de información, sus inter-relaciones y propiedades.

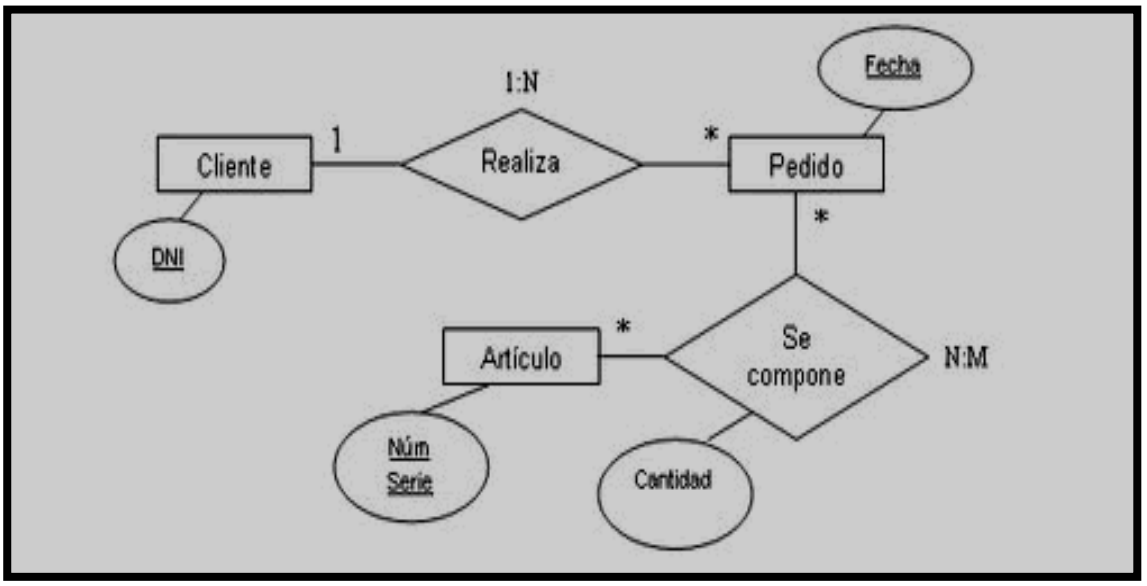


Figura Nº 2.2: Modelo Entidad Relación

- **Entidad:** Es el objeto sobre el cual se requiere mantener o almacenar información. Las entidades se las representa mediante cajas que se colocan el nombre de la entidad con letras mayúsculas. Ejemplo:

Figura Nº 2.3: Diagrama Entidad

- **Relación***:* Es la asociación significativa y estable entre dos entidades Las relaciones se representan con líneas que conectan las cajas de las entidades. Ejemplo:

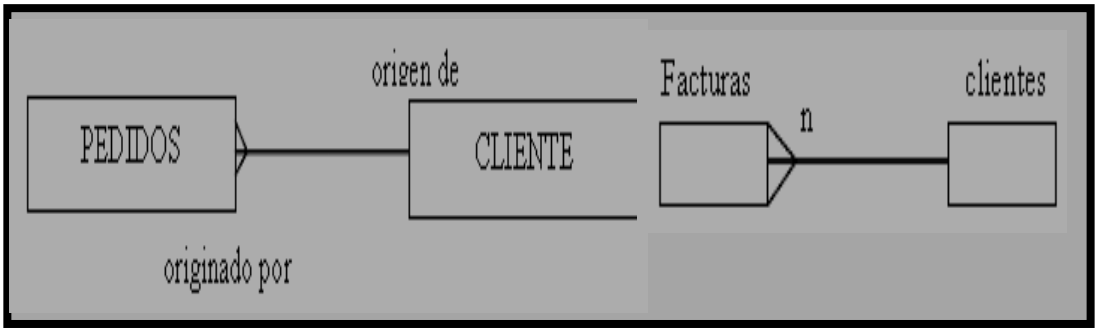


Figura Nº 2.4: Diagrama Relación

- **Atributo***:* Son las propiedades que describen y califican una entidad. Los atributos se incluyen dentro de las cajas de las entidades y se escriben con minúsculas. Ejemplo:

Figura Nº 2.5: Diagrama Atributo

Entidades: Se puede considerar entidades a los sujetos, objetos, a los eventos, a los lugares y a las abstracciones.

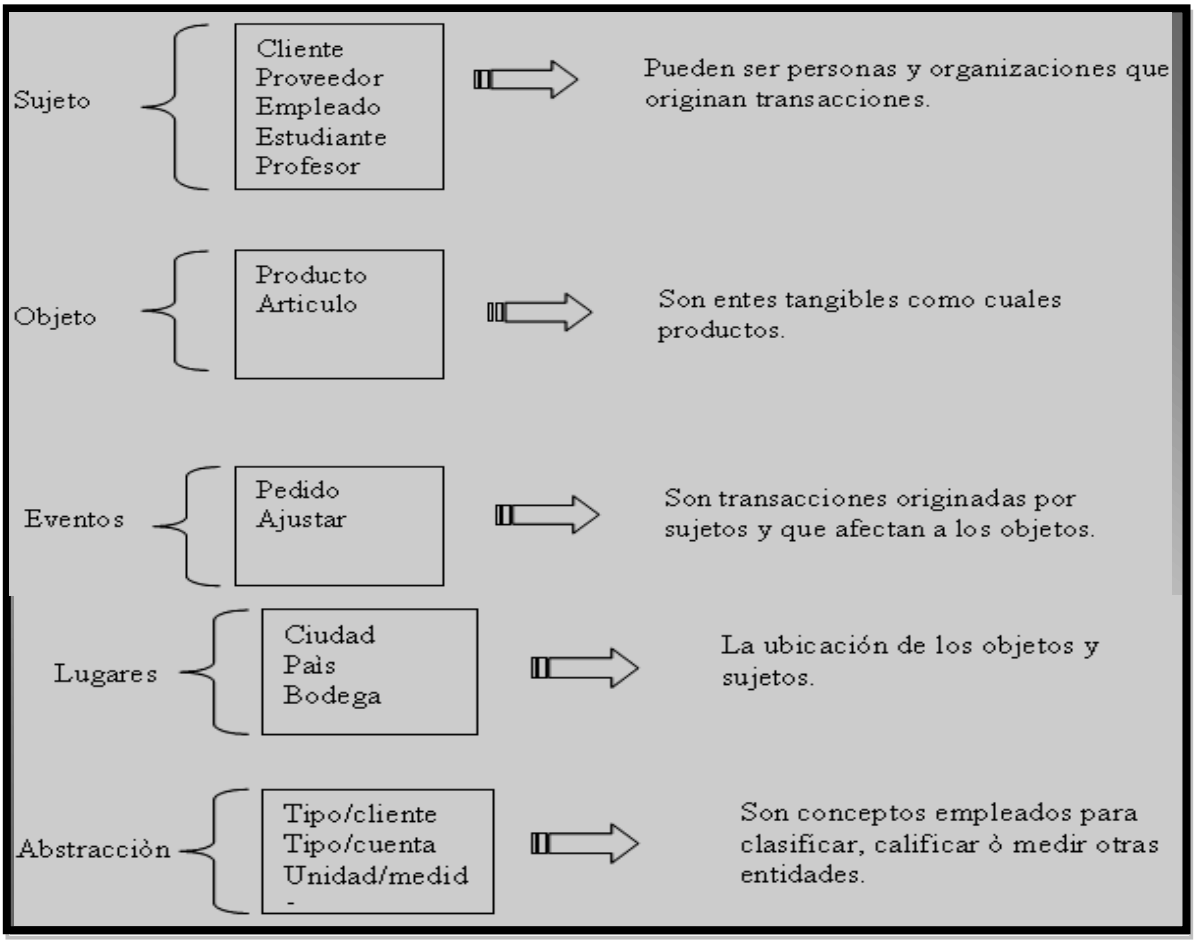


Figura Nº 2.6: Diagrama Entidades

Cardinalidad de las Relaciones

El diseño de relaciones entre las tablas de una base de datos puede ser la siguiente:

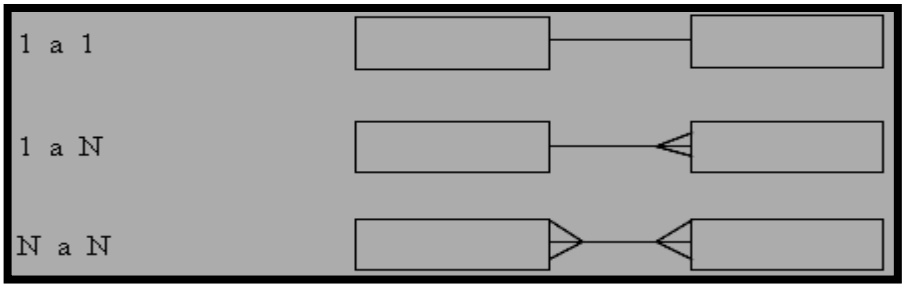


Figura Nº 2.7: Clasificación de Cardinalidad de las Relaciones

- **Relaciones de uno a uno:** Una instancia de la entidad A se relaciona con una y solamente una de la entidad B.

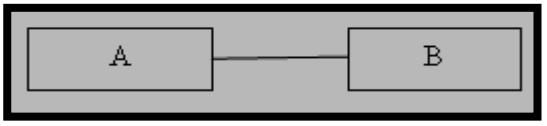


Figura Nº 2.8: Relación de uno a uno

- **Relaciones de uno a muchos:** Cada instancia de la entidad A se relaciona con varias instancias de la entidad B.

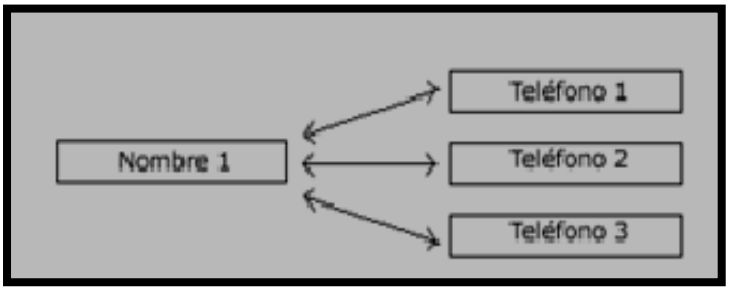


Figura Nº 2.9: Relación de uno a muchos

- **Relaciones de muchos a muchos:** Cualquier instancia de la entidad A se relaciona con cualquier instancia de la entidad B.

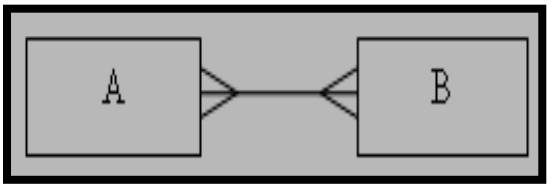


Figura Nº 2.10: Relación de uno a muchos

Por consiguiente una base de datos posee el siguiente orden jerárquico:

- Tablas
- Campos
- Registros
- Lenguaje SQL

Tipos de Campos

Cada Sistema de Base de Datos posee tipos de campos que pueden ser similares o diferentes. Entre los más comunes podemos nombrar:

- **Numérico:** entre los diferentes tipos de campos numéricos podemos encontrar enteros "sin decimales" y reales "decimales".
- **Booleanos:** poseen dos estados: Verdadero "Si" y Falso "No".
- **Memos:** son campos alfanuméricos de longitud ilimitada. Presentan el inconveniente de no poder ser indexados.
- **Fechas:** almacenan fechas facilitando posteriormente su explotación. Almacenar fechas de esta forma posibilita ordenar los registros por fechas o calcular los días entre una fecha y otra.
- **Alfanuméricos:** contienen cifras y letras. Presentan una longitud limitada (255 caracteres).
- **Auto incrementables:** son campos numéricos enteros que incrementan en una unidad su valor para cada registro incorporado. Su utilidad resulta: Servir de identificador ya que resultan exclusivos de un registro.

Sistema de Gestión de Base de Datos (SGBD)

Los Sistemas de Gestión de Base de Datos son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición

de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta.

Tipos de Gestión de Base de Datos

Entre los diferentes tipos de base de datos, podemos encontrar los siguientes:

- **MySql:** Es una base de datos con licencia GPL (Licencia Pública General) basada en un servidor. Se caracteriza por su rapidez. No es recomendable usar para grandes volúmenes de datos.
- **PostgreSql y Oracle:** Son sistemas de base de datos poderosos. Administra muy bien grandes cantidades de datos, y suelen ser utilizadas en intranets y sistemas de gran calibre.
- **Access:** Es una base de datos desarrollada por Microsoft. Esta base de datos, debe ser creada bajo el programa Access, el cual crea un archivo .mdb.
- **Microsoft SQL Server:** Es una base de datos más potente que Access desarrollada por Microsoft. Se utiliza para manejar grandes volúmenes de informaciones. Es un sistema de gestión de bases de datos relacionales (SGBD) basado en el lenguaje Transact-SQL, y específicamente en Sybase IQ, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Este lenguaje nos permite realizar consultas a nuestras bases de datos para mostrar, insertar, actualizar y borrar datos.

    - **Mostrar:** Para mostrar los registros se utiliza la instrucción Select. Select * From comentarios.
    - **Insertar:** Los registros pueden ser introducidos a partir de sentencias que emplean la instrucción Insert. Insert Into comentarios (titulo, texto, fecha) Values ('saludos', 'como esta', '22-10-2007')

- **Borrar:** Para borrar un registro se utiliza la instrucción Delete. En este caso debemos especificar cuál o cuáles son los registros que queremos borrar. Es por ello necesario establecer una selección que se lleva a cabo mediante la cláusula Where. Delete From comentarios Where id='1'.
- **Actualizar:** para actualizar los registros se utiliza la instrucción Update. Como para el caso de Delete, necesitamos especificar por medio de Where cuáles son los registros en los que queremos hacer efectivas nuestras modificaciones. Además, tendremos que especificar cuáles son los nuevos valores de los campos que deseamos actualizar. Update comentarios Set titulo='Mi Primer Comentario' Where id='1'.

- **Que es Sybase IQ:** Sybase IQ es un motor de bases de datos altamente optimizado para inteligencia empresarial, desarrollado por la empresa Sybase.- diseñado específicamente para entregar resultados más rápidos en soluciones de inteligencia empresarial analítica de misión crítica, almacenes de datos y generación de reportes, combina velocidad y agilidad, con un bajo costo total de propiedad, lo que permite a las empresas llevar a cabo análisis de datos y generación de reportes antes impensables, imprácticos o costosos.
- **Sybase**: Es una compañía líder en el desarrollo y expansión de tecnología innovadora para la movilización de información. Se ha ganado la confianza de muchas de las compañías más importantes del mundo por su habilidad en la gestión de información**.** Con una base global y leal de clientes y una fuerte presencia en mercados verticales clave, como servicios financieros, telecomunicaciones, salud y gobierno, ha permitido materializar, a clientes de todos los tamaños, la información fluye libremente y de manera segura dentro de una organización, así los empleados lleven a cabo sus negocios dentro de la oficina, o fuera de ella.

Servidores Web (IIS)

Internet Information Services o IIS es un servidor web y un conjunto de servicios para el sistema operativo Microsoft Windows. Originalmente era parte del Option Pack para Windows NT. Luego fue integrado en otros sistemas operativos de Microsoft destinados a ofrecer servicios, como Windows 2000 o Windows Server 2003. Windows XP Profesional incluye una versión limitada de IIS.

Los servicios que ofrece son:

- FTP
- SMTP
- NNTP
- HTTP/HTTPS.

Este servicio convierte a una PC en un servidor web para Internet o una intranet, es decir que en las computadoras que tienen este servicio instalado se pueden publicar páginas web tanto local como remotamente. Los servicios de Internet Information Services proporcionan las herramientas y funciones necesarias para administrar de forma sencilla un servidor web seguro.

Ventajas de los Servidores Web (IIS)

- **Confiable y escalable**: Proporciona un entorno de servidor web más inteligente y confiable para lograr la confiabilidad óptima
- **Seguro y administrable:** Proporciona una seguridad y capacidad de administración significativamente mejoradas. Las mejoras de

seguridad incluyen cambios tecnológicos y de procesamiento de solicitudes.

- **Desarrollo y compatibilidad internacional mejorados**: Los desarrolladores de aplicaciones se benefician con un único entorno de alojamiento de aplicaciones integrado, con una compatibilidad total con las características avanzadas y con el caché en modo de núcleo.

Microsoft .Net

Microsoft.Net es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando durante los últimos años con el objetivo de obtener una plataforma sencilla y potente para distribuir el software en forma de servicios que puedan ser suministrados remotamente y que puedan comunicarse y combinarse unos con otros de manera totalmente independiente de la plataforma, lenguaje de programación y modelo de componentes con los que hayan sido desarrollados.

Para crear aplicaciones para la plataforma .Net, tanto servicios Web como aplicaciones tradicionales (aplicaciones de consola, aplicaciones de ventanas, servicios de Windows NT, etc.), Microsoft ha publicado el denominado kit de desarrollo de software conocido como .Net Framework SDK, que incluye las herramientas necesarias tanto para su desarrollo como para su distribución y ejecución y Visual Studio.Net.

Microsoft.Net también incluye un conjunto de nuevas aplicaciones que Microsoft y terceros han (o están) desarrollando para ser utilizadas en la plataforma .Net. Entre ellas podemos destacar aplicaciones desarrolladas por Microsoft tales como:

- Windows.NET

- Hailstorm
- Visual Studio.NET
- MSN.NET
- Office.NET
- Los nuevos servidores para empresas de Microsoft (SQL Server.NET, Exchange.NET, etc.)

Lenguaje de Programación C Sharp

Lenguaje de programación diseñado por Microsoft en 2001 como parte de su plataforma .NET combina el lenguaje de bajo nivel de C y la velocidad. C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (es una organización internacional basada en membrecías de estándares para la comunicación y la información) e ISO (Organización Internacional de Normalización).

Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma.NET, similar al de Java aunque incluye mejoras derivadas de otros lenguajes (entre ellos Delphi).

C#, como parte de la plataforma.NET, está normalizado por ECMA desde diciembre de 2001 (C# Language Specification "Especificación del lenguaje C# "). El 7 de noviembre de 2005 salió la versión 2.0 del lenguaje, que incluía mejoras tales como tipos genéricos, métodos anónimos, tipos parciales y tipos anulables. El 19 de noviembre de 2007 salió la versión 3.0 de C#, destacando entre las mejoras los tipos implícitos, tipos anónimos y LINQ (Language Integrated Query -consulta integrada en el lenguaje).

Asp.Net

Es un framework para aplicaciones web desarrollado y comercializado por Microsoft. Es usado por programadores para construir sitios web dinámicos, aplicaciones web y servicios web XML. Apareció en enero de 2002 con la versión 1.0 del .NET Framework, y es la tecnología sucesora de la tecnología Active Server Pages (ASP).

Cualquier persona que está familiarizada con el desarrollo de aplicaciones web sabrá que el desarrollo web no es una tarea simple. Ya que mientras que un modelo de programación para aplicaciones de uso común está muy bien establecido y soportado por un gran número de lenguajes, herramientas de desarrollo, la programación web es una mezcla de varios lenguajes de etiquetas, un gran uso de lenguajes de script y plataformas de servidor.

Que es Asp.Net

Herramienta de desarrollo Web comercializado por Microsoft. Es usado por programadores para construir sitios Web dinámicos, aplicaciones Web.

Cualquier persona que está familiarizada con el desarrollo de aplicaciones Web sabrá que el desarrollo Web no es una tarea simple. Ya que mientras que un modelo de programación para aplicaciones de uso común está muy bien establecido y soportado por un gran número de lenguajes, herramientas de desarrollo, la programación Web es una mezcla de varios lenguajes de etiquetas, un gran uso de lenguajes de script y plataformas de servidor. Desafortunadamente para el programador de nivel intermedio, el conocimiento y habilidades que se necesitan para desarrollar aplicaciones

Web tienen muy poco en común con las que son necesarias en el desarrollo tradicional de aplicaciones.

Punto Net

Net es una iniciativa llevada a cabo por la empresa Microsoft que abarca toda su gama de productos y servicios, dada la presencia de esta empresa en el mercado tecnológico hace que .Net sea considerada una Tecnología de Vanguardia.

Características de Punto Net

Tiene las siguientes características que le hacen más flexible a la hora de manejarle:

- **Cargador de clases:** Permite cargar en memoria las clases.
- **Compilador msil a nativo:** Transforma código intermedio de alto nivel independiente del hardware que lo ejecuta a código de máquina propio del dispositivo que lo ejecuta.
- **Administrador de código:** Coordina toda la operación de los distintos subsistemas del Common Language Runtime (entorno en tiempo de ejecución de lenguaje común).
- **Recolector de basura:** Elimina de memoria objetos no utilizados.
- **Motor de seguridad:** Administra la seguridad del código que se ejecuta.
- **Motor de depuración:** Permite hacer un seguimiento de la ejecución del código aún cuando se utilicen lenguajes distintos.
- **Verificador de tipos:** Controla que las variables de la aplicación usen el área de memoria que tienen asignado.
- **Administrador de excepciones**: Maneja los errores que se producen durante la ejecución del código.

- **Soporte de multiproceso (threads):** Permite ejecutar código en forma paralela.
- **Empaquetador de COM:** Coordina la comunicación con los componentes COM para que puedan ser usados por el .NET Framework.
- **Soporte de la biblioteca de clases base**: Interfaz con las clases base del .NET Framework. Esto quiere decir que existen tipos de estructuras como es la de java y la .NET

Tecnologías para crear Páginas Dinámicas

Es un lenguaje de lado cliente servidor y otra serie de nociones básicas como es la programación en ASP o PHP. Hay cientos de tecnologías a disposición del web máster, que hacen un sitio sea dinámico, amigable y exitoso. Las tecnologías más utilizadas son:

- **PHP**: Este lenguaje es, como ASP, usado en el lado del servidor. Es similar a ASP y puede ser usado en circunstancias similares. Es muy eficiente, permitiendo el acceso a bases de datos puede ser usado para crear páginas dinámicas complejas. Es un lenguaje de programación interpretado. Es fácil de aprender e implementar prácticamente no hay mayores inconvenientes al momento de llevar cuestiones teóricas a la práctica profesional. Fué creado por un capo (**Rasmus Lerdorf**), y licenciado bajo la "PHP license" compatible con GPL, por lo tanto es software libre. Soporta nativamente la conexión a un gran número de base de datos, tiene implementaciones muy maduras para MySQL, PosgreSQL, SQLServer, etc. No es muy usado para muy grandes proyectos.
- **ASP (Active server pages):** Es una tecnología para hacer aplicaciones web. Las Páginas Activas se utilizan para ejecutar acciones del lado del servidor. De forma opuesta al Javascript, que realiza procedimientos en la máquina de cada usuario, el ASP forma en el servidor los resultados que luego se mostrarán en las pantallas

de cada navegante. Un ejemplo de esto son los buscadores, donde uno realiza una petición de información y el servidor del buscador nos entrega un resultado a medida de nuestro pedido. Todo este procedimiento se realiza en el servidor y no en nuestra máquina. Y aunque hay versiones de ASP para Unix y Linux, fue desarrollado principalmente para ser usado en servidores web basados en sistemas Microsoft. Las páginas activas, o dinámicas, son especialmente útiles para mantener bases de datos, crear buscadores dinámicos, hacer carritos de compras, y todo aquello que necesite una interacción del navegante y el servidor para elaborar un resultado.

- **JSP (Java Server Pages)** Es una tecnología Java que permite a los programadores generar contenido dinámico para web, en forma de documentos HTML *(*Lenguaje de Marcado de Hipertexto), XML (lenguaje de marcas extensible), o de otro tipo. Permite al código Java y a algunas acciones predefinidas ser incrustadas en el contenido estático del documento web. Utiliza una variante del lenguaje Java (bastante más limitado). Permite una perfecta integración con clases Java. Muy utilizado en grandes proyectos de IT (Tecnologías de la Información), por su escalabilidad y código robusto. Facilita bastante la conexión a las bases de datos. Es software libre. Posee generosa cantidad de códigos de ejemplo en la red. JSP no es totalmente un leguaje de script ya que antes de ejecutarse el servidor compila y genera un servet, por lo tanto, se puede decir que no deja de ser una aplicación compilada. La ventaja de esto es algo más de rapidez y disponer del API de Java en su totalidad.
- **ColdFusion:** La fusión fría de la ex Macromedia (ahora propiedad de Adobe), es un lenguaje distinto, ya que su sintaxis está basada en tags, al mejor estilo HTML, si bien es cierto que se aprende bastante más rápido, en algunas cosas no es tan potente o robusto como ellos.
- **CGI (Common Gateway Interface).-** La Interface Común de Entrada es uno de los más antiguos estándares en internet para trasladar la información desde una página web a un servidor web. Es

el método más utilizado para manipular objetos como libros de visitas, formularios de emails, foros de discusión y elementos así. Actualmente lo más común para subir y bajar información. No es en absoluto un lenguaje de scripting, de hecho las rutinas de CGI (Interfaz de Entrada Común) son habitualmente escritas en lenguajes interpretados como Perl o por lenguajes compilados como C.

- **CSS (Cascading Style Sheets).-** Las Hojas de Estilo las usamos para formatear las páginas web en todo aquello que deseamos. CCS (Calculo de Sistema Comunicantes) es complicado, pero no es nada si comprendemos que podemos definir, en una sola línea de código, un estilo de texto o títulos para todo el sitio. Una vez que modificamos esta línea en segundos, todo el sitio cambiará automáticamente.
- **Htaccess.-** Nos permite configurar parámetros para nuestro sitio web y para las carpetas (directorios). El uso más común es el de proteger los directorios con claves de acceso. Htaccess puede ser usado para muchas otras cosas como: alejar a las arañas (spiders) que rastrean información en la WWW, acceso limitado a ciertas partes de nuestro sitio, etc. El lado oscuro del htaccess sería que su lenguaje suele ser muy oscuro, difícil de entender y demasiado preciso. Un mínimo error en el archivo htaccess puede hacer que todo el sitio web esté offline, hasta que sea reparado este error.
- **Java.-** Este es un lenguaje que trabaja en el cliente, se ejecuta en el browser del navegante y no en el servidor. Es eficiente y muy poderoso. La principal ventaja de Java sobre ActiveX es que Java tiene un modelo de seguridad sano, llamado la Caja de Arena, mientras que el modelo de ActiveX es demasiado lograr desafiar nuestra imaginación. Java es también mucho menos amigo de los cuelgues del sistema. Es considerablemente más lento que ActiveX, y hay muchas tareas que Java directamente no puede realizar porque no tiene acceso al sistema operativo o al disco mismo.
- **JavaScript.-** Este es un lenguaje que se interpreta y se ejecuta en el cliente. Es muy útil para realizar tareas en el lado del cliente

(navegante), como mover imágenes por la pantalla, crear menús de navegación interactivos, utilizar algunos juegos, etc.

- **Perl.-** Un gran lenguaje de encriptado que nos permite que el CGI tradicional se ejecute en el servidor. Perl es muy fácil de aprender y llano en su funcionamiento. Es principalmente usado para libros de visita, formularios de consulta y otras tareas sencillas. La principal contra que posee es que, cada vez que un proceso se desarrolla y el lenguaje es interpretado, el código es recompilado nuevamente cada vez que se corre. Para tareas complejas, un lenguaje del lado del servidor como PHP o ASP es mucho más conveniente.
- **SSI.-** Si tu sitio está alojado en un típico servidor Apache, seguramente podrás usar algo que se denomina "Server Side Includes". Esto es una forma de lograr que el servidor web ejecute tareas antes de que se muestre una página web en nuestro navegador. Uno de los usos más comunes es incluir, adecuadamente, texto común. SSI es muy común pero fue ampliamente superado por lenguajes como PHP. La sobrecarga de SSI en el servidor es alta puesto que cada página es escaneada para saber si posee indicaciones SSI antes de mostrarla en el browser.
- **VBScript (Visual Basic Scripting)**.- Es una buena herramienta para cualquier sitio destinado a ser mostrado exclusivamente el navegador Microsoft Internet Explorer. Por este motivo discriminante y limitante, consideramos que no debe usarse VBScript en ningún sitio, al menos en ninguno que aspire a tener difusión universal. Es preferible usar JavaScript con su amplia aceptación en los diversos navegadores de internet.

Sistemas Web

Los sistemas desarrollados en plataformas Web, tienen marcadas diferencias con otros tipos de sistemas, lo que lo hacen muy beneficio tanto para las empresas que lo utilizan, como para los usuarios que operan en el sistema. Este tipo de diferencias se ven reflejada en los costos de las

empresas, en la rapidez de obtención de la información, en la optimización de las tareas por parte de los usuarios y en alcanzar una gestión íntegramente informatizada dentro y fuera de la empresa.

Hoy día las empresas se han reconvertido desde el punto de vista informático, para hacer más fácil y eficiente tareas que antes llevaban mucho tiempo. Los sistemas web son un escalón más, en la administración de la información y en la facilidad de acceso informático para todos los empleados de cada empresa.

La instalación del sistema se realizó en un servidor, no siendo necesario instalarlo en cada terminal que lo va a utilizar. Dentro y fuera de la empresa el acceso al sistema se realiza desde cualquier PC que tenga conexión a Internet, e inclusive sin contar con conexión dentro de la empresa, igual se puede acceder al sistema si las terminales están conectadas a través de la red interna.

Ventajas del Sistemas Web

- **Ahorra tiempo**: Se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.
- **No hay problemas de compatibilidad**: Basta tener un navegador actualizado para poder utilizarlas.
- **No ocupan espacio** en nuestro disco duro.
- **Actualizaciones inmediatas**: Como el software lo gestiona el propio desarrollador, cuando nos conéctanos estamos usando siempre la última versión que haya lanzado.
- **Consumo de recursos bajo**: Dado que toda (o gran parte) de la aplicación no se encuentra en nuestro ordenador, muchas de las tareas que realiza el software no consumen recursos nuestros porque se realizan desde otro ordenador.

- **Multiplataforma**: Se pueden usar desde cualquier sistema operativo porque sólo es necesario tener un navegador.
- **Portables**: Es independiente del ordenador donde se utilice porque se accede a través de una página web (sólo es necesario disponer de acceso a Internet). La reciente tendencia al acceso a las aplicaciones web a través de teléfonos móviles requiere sin embargo un diseño específico de los ficheros CSS (Hojas de estilo en cascada) para no dificultar el acceso de estos usuarios.
- **La disponibilidad suele ser alta** porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.
- **Los virus no dañan** los datos porque éstos están guardados en el servidor de la aplicación.
- **Colaboración**: Gracias a que el acceso al servicio se realiza desde una única ubicación es sencillo el acceso y compartición de datos por parte de varios usuarios. Tiene mucho sentido, por ejemplo, en aplicaciones online de calendarios u oficina.

Inconvenientes de los Sistemas Web

- Habitualmente ofrecen menos funcionalidades que las aplicaciones de escritorio. Se debe a que las funcionalidades que se pueden realizar desde un navegador son más limitadas que las que se pueden realizar desde el sistema operativo. Pero cada vez los navegadores están más preparados para mejorar en este aspecto.
- La disponibilidad depende de un tercero, el proveedor de la conexión a internet o el que provee el enlace entre el servidor de la aplicación y el cliente.

Para realizar el sistema se contara con las siguientes herramientas de desarrollo:

- **Punto net** aunque no es un lenguaje de programación en sí mismo, sino una arquitectura de desarrollo web.
- **La interfaz del usuario** se lo realizó en Asp.Net con esta plataforma nos permite realizar páginas web, debido a que es orientado a realizar páginas dinámicas y amigables, además es la plataforma que más se está usando y aprovechando las licencias que posee la empresa.
- **Motor de base de datos** como es SQL Server 2008 debido a que tiene su respectiva licencia.

La empresa Quito Motors siempre ha llevado el control de la información de los empleados y aspirantes manualmente es decir no muy actualizada y no se puede obtener los reportes diarios, lo que ocasiona un desperdicio de tiempo en procesos de la institución que deberían ser más automatizado. Se incorporo un sistema automatizado con el cual se podrá ahorrar tiempo.

Ellos cuentan con suficientes equipos de computo por lo que no hay ningún inconveniente para el desarrollo e implantación del sistema y pueda mostrar como resultado los datos deseados. A la hora de utilizar el sistema el empleado podrá encontrar la información requerida.

De acuerdo a la investigación realizada en la institución se ha logrado determinar que es necesario el desarrollo del Sistema de Automatización de Gestión de Carpetas de Aspirantes más eficiente para poder desempeñar sus actividades de manera más rápida y confiable. Este nuevo sistema incluye reportes de la información que se ingrese a la base de datos.

Además los empleados de Recursos Humanos brindarán una buena atención a los aspirantes y sus compañeros de trabajo, esto implica la manera en que son atendidos, al entregar la información solicitada. Por medio de este nuevo sistema, la calidad se va a detectar desde el momento en que el empleado hace contacto con el aspirante hasta que el aspirante termina de realizar los trámites de su información.

La institución ha decidido que el sistema será implantado, entonces también se aplicara nuevos programas para estar a la par de este mundo cambiante. En la actualidad toda empresa o institución pública o privada está obligada a que cada vez se actualice con los avances tecnológicos que ocurren en el mundo cambiante, por ende debe estar a la par con la tecnología.

2.3 Variables

2.3.1 Variable Independiente

Desarrollo de un sistema de información de gestión de las carpetas de aspirantes a la Quito Motors

2.3.2 Variable Dependiente

Quito Motors Matriz.

2.4 Hipótesis

Con el desarrollo del sistema se automatizará la información de los empleados y aspirantes a la empresa que es administrada por el departamento de recursos humanos, obteniendo los datos requeridos por los usuarios de manera oportuna y confiable.

CAPÍTULO III

METODOLOGÍA

3.1 Enfoque

En el trabajo se aplicara un enfoque eminentemente cualitativo porque se buscará llevar un control de las carpetas de la empresa determinada, para esto se realizará las investigaciones que permitirá establecer el origen o las causas por lo que la empresa ha comenzado a tener problemas. Se realizara un estudio profundo del tema, para llevar a cabo la investigación se ayuda con entrevistas en la empresa.

En el campo tecnológico de desarrollo se utilizó el ciclo de vida clásico de software que se describirá en el capítulo V.

3.2 Modalidad Básica de la Investigación

3.2.1 Investigación de Campo

Una investigación de campo es importante porque permitirá averiguar las causas que provocaron el problema de la gestión de las carpetas de los aspirantes a la Quito Motors, estos datos servirán para conocer la realidad de la empresa mencionada. Se realizó la investigación en la institución la cual es el objeto de estudio.

3.2.2 Investigación Documental

Con esta investigación se podrá realizar una investigación más profunda sobre el problema del departamento de Recursos Humanos, realizando una investigación con problemas similares que ocurren en otras empresas como es el caso de la "Importadora Sánchez Montoya" que a pesar de ser una empresa pequeña si es necesario tener un control de los empleados, no se entrará más en detalle porque no es el caso de este empresa.

Para de esta manera poder recopilar información valiosa y que sirvió de mucho en la realización de proyecto.

3.2.3 Proyecto Factible

Es un proyecto factible ya que mediante el desarrollo del sistema que se aplicara el ciclo de vida clásico, se solucionará el problema del control de las carpetas de los aspirantes y empleados que quieren trabajar y trabajan en la empresa matriz y sus sucursales, el sistema de automatización es una alternativa viable.

3.3 Nivel o Tipo de Investigación

Será exploratorio debido a que es necesario realizar las visitas cuantas veces sea necesaria a la matriz que esta ubica en la ciudad de Quito en la Av. 10 de agosto, para poder investigar y establecer el origen o las causas del problema que se dio en la empresa.

Es descriptivo ya que se analizará y se hará un estudio previo del origen del problema, las posiciones y las dificultades por la que está atravesando.

Es correlacional porque establecerá una relación entre las causas, las dificultades para proponer algunas alternativas de solución.

3.4 Población y Muestra

3.4.1 Población

Mencionaremos las características de la población de estudio que son seres humanos:

1. Nombre de la Empresa: Quito Motors.
2. La población está caracterizada por persona mayores de edad, que tendrán la oportunidad de un trabajo estable.
3. El periodo de la preselección puede variar de una semana a un mes.

3.4.2 Muestra

Es una parte de la población que se selecciona para realizar el estudio. Para nuestro estudio del proyecto tomaremos a los empleados que trabajan solo en la matriz que es de 50.

3.5 Operación de las Variables

Es un paso más para el desarrollo del proyecto. Cuando se identifican las variables, el próximo paso es su operación, es decir hacerla tangible, operativa, medible o por lo menos registrable en la realidad.de tres tipos de definiciones:

- Nominal: Nombre de la variable Quito Motors.
- Real: Consiste en determinar las dimensiones que contienen las variables nominales.
- Operacional o indicadores: Esta da la base para su medición y la definición de los indicadores que constituyen los elementos más concretos de una variable y de donde el investigador derivará los

items o preguntas para el instrumento con que recolectará la información desarrollo de la investigación. La información será recolectada mediante entrevistas realizadas a los empleados de la empresa.

3.6 Recolección de la Información

3.6.1 Plan para Procesamiento de la Información.

Para la entrevista que fue realizada en la matriz de la Empresa Quito Motors entre sus trabajadores, antes fueron preparados con un bloque de preguntas que fueron analizadas previamente, y fueron enfocadas a investigar el origen del problema por la que la empresa está atravesando.

El bloque de preguntas fue:

1. ¿Cuál opina usted que sería el un problema fundamental por lo que la empresa está pasando?
2. ¿Cuál cree usted que es la principal causa del origen del problema?
3. ¿Qué opinión podría ofrecer para encontrar una solución optima al problema?
4. ¿Estaría usted de acuerdo en la utilización de recursos informáticos para la solución como puede ser un sistema de automatización?

3.7 Análisis e Interpretación de la Información

3.7.1 Comprobación de la Hipótesis

La hipótesis cumple con los requisitos para su respectiva evaluación:

- Es comprobada, es decir, establece claramente su referente empírico.

- Está en correlación y armonía con el conjunto de las hipótesis del proyecto de la investigación
- Responde en términos claros y precisos al problema planteado, es decir, señala la relación que se espera de las variables.
- Son susceptibles de ser cuantificadas.

La comprobación de la hipótesis significa someterla a contrastación de una realidad. Se ha sometido a la prueba de la encuesta para comprobar el enunciado de su hipótesis, y para ello ha de establecer, mediante alguna técnica de contrastación si su hipótesis concuerda o no.

Para la comprobación de la hipótesis se pone en práctica tres procedimientos básicos: la observación, la experimentación, junto con la encuesta.

La demostración de la hipótesis es la actividad que consiste en constar, mediante la observación y/o experimentación, si una hipótesis empírica es verdadera o falsa. Para el proyecto la hipótesis ha sido comprobada y considerada científica.

CAPÍTULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- Con la investigación de campo que se realizo en la institución, se recopiló información con la aplicación de una encuesta a los empleados de la empresa, en la cual nos arrojo datos positivos que fueron de gran ayuda para el desarrollo del proyecto.
- En la realización del proyecto se utilizó el ciclo de vida del software aplicando la ingeniería, se elaboró una Propuesta de un Sistema Automatizado para el Control de las Carpetas de los aspirantes a la Quito Motors el cual fue aceptado y desarrollado, como una manera para darle solución a las fallas que se presentan en el departamento de recursos humanos que lo están realizando en forma manual y ahora lo hacen en forma automatizada.
- Debido a la necesidad de mejorar el sistema de control de información que se llevo anterior mente y a la pérdida de tiempo que significaba llevar este control manual mente, con la implantación del sistema automatizado para controlar la información, el proceso de consultas es más rápido.
- El sistema está estructurado para ingresar, almacenar, actualizar, ordenar, procesar y emitir resultados deseados mediantes reportes.
- De acuerdo con la información recolectada, cuenta con un personal convencido de las ventajas que ofrece el sistema de automatización y la necesidad de manejar información precisa y veraz, de manera

que la idea de una propuesta de un Sistema de Automatización tiene una alta posibilidad de éxito.

- Con el desarrollo del sistema se pretende automatizar la información del departamento de recursos humanos, para poder llevar un control también de los empleados que trabajan y no solo de los aspirantes.
- Es una página web animada y dinámica agradable para el usuario que va a utilizar, y no tenga problemas en familiarizarse, debido a que fue capacitado anteriormente para su utilización.

4.2 Recomendaciones

- Con la implantación del sistema automatizado en el departamento requerido, se puede agilizar los procesos manuales, se puede instalar en las sucursales, siempre que cumpla con los requisitos establecidos para un correcto funcionamiento del software.
- Es recomendable diseñar un plan de seguridad y respaldo, que permita rescatar los datos en caso de cualquier eventualidad, debido a que los equipos tecnológicos tienden a dañarse o pueden sufrir algún desperfecto, tanto en hardware como en software existen programas maliciosos por ende es necesario la configuración del servidor tenga respaldos de información .
- Adiestramiento a los usuarios que van a utilizar el sistema, para un mejor funcionamiento del mismo, tanto de las sucursales como de la matriz, esta capacitación incluye como está diseñado el software, ingreso, procesamiento, actualización, consulta de los datos, etc.
- Mantenimiento constante al sistema, como a la base de datos una revisión cada semana para verificar el respaldo de la información y el manteniendo del software para un correcto funcionamiento.
- El sistema necesita de un mínimo de requerimientos para funcionar correctamente, antes de proceder a la instalación y poner a funcionar la aplicación, se deben tomar en cuenta todas estas variables que

serán fundamentales para el correcto funcionamiento del sistema.

- Reemplazar los equipos de oficinas obsoletos (computadoras, fax, otros) por nuevos equipos con tecnología de punta, cuando exista el presupuesto de cada departamento.

CAPÍTULO V

PROPUESTA FINAL

5.1 Cicló de Desarrollo del Software

5.1.1 Introducción al Cicló de Vida del Software

Los productos que se relacionan a avances tecnológicos en lo que se refiere a software han sido de gran ayuda a las actividades humanas, debió a que estos responden a las necesidades de personas, empresas, organizaciones, etc. Los productos de software tienen un ciclo de vida que consta de dos etapas

Etapa de Concepción y Desarrollo

- Se define las necesidades que debe cumplir el software que en este caso administrar la información de recursos humanos
- Se analiza la aplicación y diseña para su correcto funcionamiento
- Se desarrolla y se realizan prueba del producto

Etapa de Operación, Mantenimiento y Retiro

- Se instala para realizar las pruebas
- Se mejora las fallas presentadas
- Eventualmente se desecha para su mejora

La ingeniería de software es la rama de las Ciencias de la Computación que se encarga de estudiar las teorías, métodos y herramientas que se necesitan para desarrollar el software

5.1.2 Definición de Ingeniería de Software

Enfoque en el Cicló de Vida de Software

Es la aplicación sistemática, disciplinada y cuantificable del desarrollo, operación y mantenimiento de software.

Enfoque en las personas involucradas

La construcción del software que es el sistema de automatización que será destinado para ser usado por el departamento de recursos humanos. Estarán involucrados personal contratado para la digitación, la persona de redes, y el desarrollador del proyecto.

5.1.3 Software su naturaleza y características

Se entiende por software al código generado por programas, escritos en algún lenguaje de programación que resolverá el problema. Al software también se le debe implementar los documentos y la configuración de datos que se requieren para un correcto funcionamiento

El software desarrollado es nuevo y no se parece a ningún otro artefacto que es tangible, también tiene una naturaleza por lo que se debe entenderla.

Algunas características de la naturaleza del software son:

1. El software se desarrolla y no se fabrica
2. No es tangible como los productos de otras ingenierías.
3. Es fácil modificarle
4. No se desgasta con el pasar del tiempo pero si se deteriora si al mantenerlo se le incorporan errores al tratar de corregir errores de otros.

Características de calidad que debe tener un producto de software:

- **Funcionalidad**: Si se comporta de acuerdo a las especificaciones de las funciones a ejecutar, el sistema de automatización despliega los resultados esperados.
- **Confiabilidad**: Si el usuario puede depender de sistema o si se comporta razonablemente, el sistema ha sido desarrollado de forma amigable para la persona a utilizar.
- **Usable:** Si el usuario le encuentra fácil de entender, aprender y usar, la persona encargada de usar el software ha sido debidamente capacitada para no tener contratiempos.
- **Eficiente:** Si usa sus recursos adecuadamente y proporciona un desempeño adecuado, el sistema utiliza los recursos como es un gran ahorro de tiempo.
- **Mantenible:** Si es fácil de hacer modificaciones tales como correcciones, mejoras o adaptaciones, es software si es adaptable hacer modificaciones el requisito es manejar un poco de ASP Net.
- **Portables:** Es posible correrlo en diversos ambientes o plataformas, el sistema puede ser ejecutado en cualquier PC que tenga internet.
- **Reusables:** Si se pueden usar partes o todo para el desarrollo de un nuevo producto, el sistema si puede ser usado por partes debido a que esta desarrollado por módulos.
- **Interoperable:** Si puede coexistir y cooperar con otros sistemas, el sistema si puede ser interoperable.

5.1.4 Principios de la Ingeniería del Software

La ingeniería de software es relativamente reciente, por lo que no están madura, existen algunos principios:

- **Generalidad**: Descubre los aspectos más generales que existen en un problema. Es utilizado para desarrollar herramientas y paquetes genéricos. Los aspectos más generales son el manejo de la información que se lleva manualmente siendo una de las empresas

más grande del país, debe ser actualizadas.

- **Abstracción**: Identificar inicialmente los aspectos más importantes e ir incorporando los detalles gradualmente. Estudiar el manejo de toda la empresa y sus sucursales en cada departamento.
- **Modularidad**: Dividir el problema en subproblemas, para el estudio del proyecto se trabajara solo en las matriz en el departamento de recursos humanos y no en las sucursales.
- **Incrementabilidad**: Se obtendrá un producto de software incrementado la funcionalidad de varios ciclos, con el avance del proyecto se irá analizando cada ciclo.
- **Separación de conceptos**: Manejar diferentes aspectos de un problema concentrándose en cada uno por separado. Entender los conceptos tecnológicos que serán de las herramientas las cuales se van a utilizar como son las bases de datos, la herramienta utilizada para el desarrollo.
- **Anticipación al cambio**: Diseñar el software para que pueda evolucionar a nuevas versiones, que se administran de manera controlada. El sistema estará sujeto a cambios sin complicaciones cuando la persona encargada de administrar así lo necesite.

5.1.5 Procesos del Software

El proceso del software es el conjunto de actividades que se necesita para transformar las necesidades de un cliente en un producto de software.

El proceso del software se ve diseñado por diagramas de clases UML:

- Los rectángulos representan clases
- La relación entres las clases se representan con un pequeño rombo que significa agregación o la relación
- Las líneas significan asociaciones entre clases.

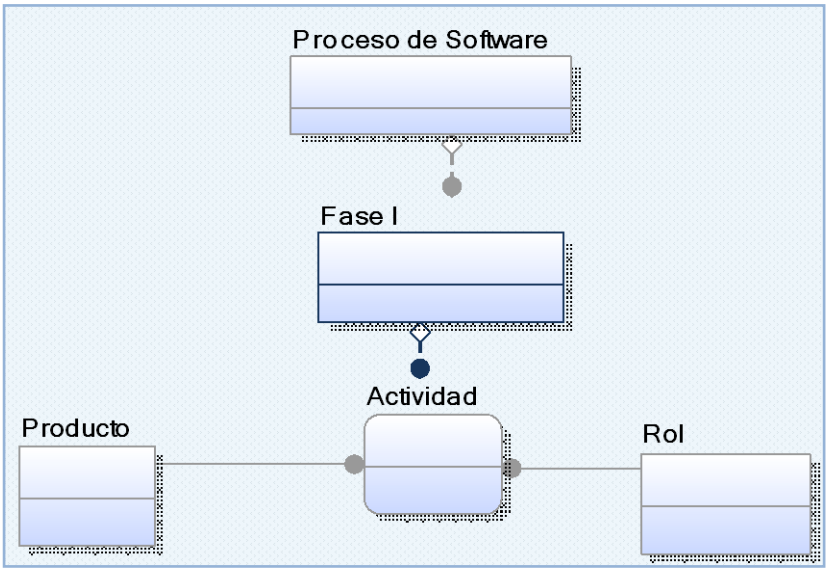


Figura N° 5.11: Proceso de Software

Un proceso de software está compuesto por fases y debe incluir al menos una fase a su vez las fases están compuestas por actividades, que tiene asociado uno o varios productos y uno o varios roles. Un rol es responsable de al menos una actividad.

- **Las fases:** constituyen pasos significativos del proceso del software, tienen un objetivo dentro del sistema, que se identifican como los roles, actividades y productos que son necesarios para cumplir el objetivo de la base. Para nuestro sistema las fases son: la información que será necesario de su ingreso para procesar la información.
- **Actividades:** Define las acciones que se llevan a cabo en el desarrollo del software y utilizan y generan los productos.
- **Productos:** Las entradas y salidas de las actividades, puede ser documentos, diagramas de diseño, código, planes de prueba, reportes, manuales de usuarios.
- **Roles:** Los responsables por llevar a cabo las actividades del proceso.

5.1.6 Cicló de Desarrollo del Software

Se enfoca a la etapa de desarrollo del software que representa el inicio del ciclo de vida. En la etapa de desarrollo se usará tecnología orientada a objetos para hacer la construcción del software.

Guiado por los Casos de Uso: Un caso de uso es la funcionalidad del sistema que proporciona al usuario un valor o servicio, el usuario es la persona que interactúa con el software. Los casos de usos los casos de usos se guían para el desarrollo del software para cumplir las necesidades del sistema. El usuario encargado de manejar el sistema, ha aceptado con satisfacción al sistema por ser agradable y amigable.

Iterativo e Incremental: Iterativo es la ejecución de todos los pasos del ciclo de desarrollo, incremento es la evolución que va teniendo el producto a lo largo del tiempo. Las iteraciones deben ser planeadas y controlada.

La iteración seguirá el ciclo de desarrollo del software, el incremento es la evolución del sistema que se dará en el avance de cada modulo de cada sistema

La iteración del ciclo de desarrollo consta de las siguientes fases:

- **Especificación de requerimientos:** Se utilizaran los casos de usos para especificar los requerimientos de esta manera tener claro los requerimientos del producto que serán detallados con el avance del proyecto.
- **Análisis:** Es analizado los requisitos para un mejor entendimiento y se construye el modelo de análisis y se identifican los elementos que servirán de base para estructurar el sistema. Se construye dos modelos la vista estática (diagrama de clase) y dinámica (diagrama de secuencia y de estados).
- **Diseño:** Se identifican los paquetes principales del software y la forma en que se distribuirán en las computadoras que interviene en

una aplicación. La implantación del sistema será realizado en el departamento de recursos humanos y la base de datos en el departamento de sistemas.

- **Construcción**: El objetivo es hacer la construcción de software y entregar el código probado de las unidades. La construcción del software se lo realizara en la herramienta que ha sido planteada.
- **Integración y pruebas:** El objetivo es hacer la integración del sistema y probar que cumpla con los requisitos, el sistema según los avances que ha tenido, hasta llegar hasta su culminación y ha presentado los resultados deseados.

5.1.7 Lenguaje de Modelado Unificado UML

El Lenguaje de Modelado Unificado (UML) es utilizado para el modelado orientado a objetos:

- Provee de un lenguaje de modelado expresivo y visual
- Es independiente del lenguaje de programación y los procesos de desarrollo
- Está enfocado a proporcionar un lenguaje de modelado estándar y no un proceso estándar.

Hay dos tipos de diagramas UML:

Diagramas Estructurales: Muestran la estructura estática y los elementos del sistema entre ellos tenemos:

- Clases
- Objetos
- Componentes
- Paquetes
- Distribución

Diagramas de Comportamiento: Muestran la dinámica de la funcionalidad del sistema, entre ellos tenemos:

- Casos de usos

- Interacción
  - Secuencia
  - Comunicación
  - Tiempo
- Actividades
- Estados

5.2 Especificación de Requerimientos

5.2.1 Entender el Problema

Se realizo un estudio a la empresa como es su administración, y utilizando técnicas como son las entrevistas y cuestionarios para ayudar a determinar las causas del problema.

Se desarrollara un sistema de automatización que permitirá ayudar al departamento de recursos humanos a la selección del personal que trabajara en la empresa.

Los puestos vacantes que exista en la matriz y en sus sucursales se podrán observar el una página web. La persona encargada de manejar el sistema podrá ingresar, consultar la información de los candidatos para el puesto que está vacante mediante su usuario y contraseña que se le será designada.

El sistema estará creado en ambiente web y podrá ser accesible mediante cualquier explorador web.

Glosario de Términos

Caso de Uso: Es la descripción de un conjunto de sentencias de acciones de un sistema

Usuario: Persona capacitada para manejar el sistema

Nombre de usuario: Identificación del usuario para que pueda acceder al sistema

Contraseña: Clave de seguridad que se le asigna a un usuario.

Empresa: Organización que tiene la necesidad de contratar personal para resolver por el problema que está atravesando

Vacante: Puesto disponible en alguna empresa

5.2.2 Especificación de Requerimientos

Los requerimientos deben formularse de forma clara y precisa. Los requerimientos de software son fundamentales para la construcción del software de calidad es decir que cumpla con las necesidades de los usuarios.

Los requerimientos son:

- El equipo de desarrollo (el desarrollador de la aplicación, la persona de redes, la persona de servidores, digitadores)
- El cliente (la empresa Quito Motors el presidente de la empresa)
- Los usuarios (La persona del departamento de recursos humanos)

Una característica de los requerimientos es que pueden ser cambiadas por algunas razones:

- Se modifican a las necesidades de los clientes
- Cambia el ambiente
- La tecnología

Se puede utilizar técnicas para describir los requerimientos como por ejemplo:

- Utilizar un lenguaje natural
- Debe ser claro para el cliente y no confuso
- El modelado gráfico debe ser claro para el desarrollador
- El prototipo de interfaz de usuario que es un modelo de cómo funcionara el sistema.

El problema ya ha sido planteado de forma clara y precisa llegando a un acuerdo entre el presidente de la empresa y el desarrollador del proyecto.

5.2.3 Requerimientos Funcionales

Los requerimientos funcionales son usadas por los casos de usos estos serán utilizados para todo el proceso de desarrollo. A partir de los casos de usos se diseña, implementa y se realiza pruebas del software.

5.2.3.1 Diagrama de Casos de Usos

Definen como utilizan el software sus usuarios, el conjunto de casos de usos conforman los modelos de casos de usos que describen el comportamiento del sistema.

Elementos de los Diagramas de Casos de Usos

- **Caso de uso:** Describe un conjunto de secuencias de acciones que realiza el sistema para entregar un resultado. El nombre del caso debe iniciar con un verbo en infinitivo.

  En el sistema tenemos la opción de consultas y reportes para obtener resultados solicitados.

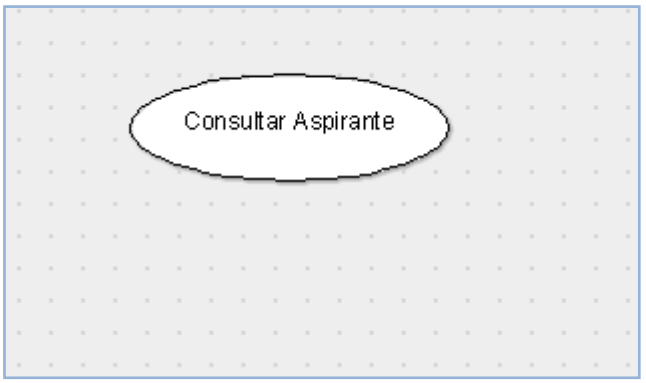


Figura N° 5.12: Elementos de los Casos de Usos

- **Actor:** Es externo que intercambia información con el sistema que es un usuario u otro usuario. El objetivo es completar la funcionalidad del sistema. Se representa con una figura humana con el nombre del actor. El sistema externo con lo que interacciona el software que se desarrolla se llama actores.

En el sistema se le conocerá como actor a la persona del Departamento de recursos humanos quien es la que se relacionara con el sistema, además de las digitadores para el ingreso del personal propio de la empresa.

Figura N° 5.13: Elemento de Casos de Uso (actor)

- **Alcance del sistema:** Representa la frontera del sistema y contiene los casos de usos que se realizan en cada ciclo de desarrollo. Se representa mediante un rectángulo que incluye los casos de uso dentro del alcance del sistema.

- **Diagrama de casos de usos:** Incluye actores identificados, el objetivo general es mostrar de forma grafica y clara las funcionalidades del sistema por lo que deberá ser simple.

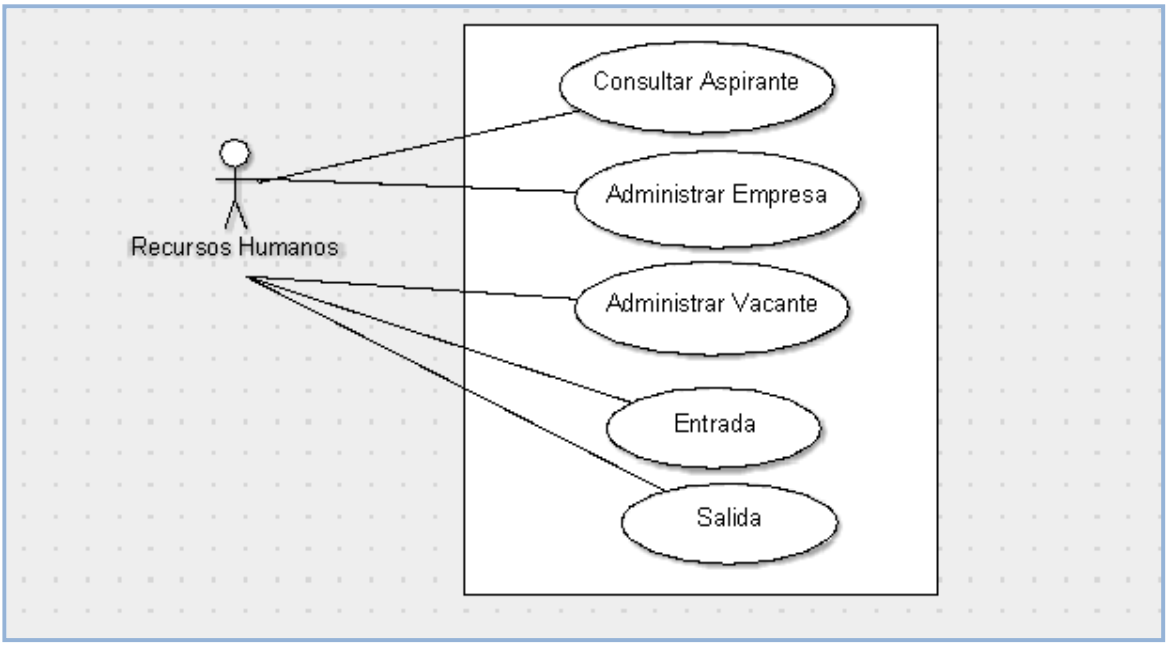


Figura N° 5.14: Ejemplo de Diagrama de Caso de Uso

- **Proceso para la creación de los diagramas de casos de usos**
  - Identificar los actores del sistema. La persona del Departamento de Recursos Humanos
  - Identificar las funcionalidades o casos de uso generales de cada actor. A todas las personas y empleados de la empresa les conocerá en forma general como aspirantes.
  - Definir el alcance o los casos de uso que se desarrollan. Lo indispensable es obtener los resultados de cada aspirante a más breve posible.
  - Detallar cada caso de uso. Se detallará todas las funcionales generales que requiere.

- **Detalle de los casos de usos**
  - **Nombre:** Debe ser un verbo en infinitivo del caso de uso
  - **Actor:** Encargado de iniciar la acción
  - **Diagrama detallado:** El caso de uso indicando el actor y sus detalles
  - **Descripción:** Texto breve de la descripción de la acción.

Ejemplo de Caso de Uso: Consultar Aspirante

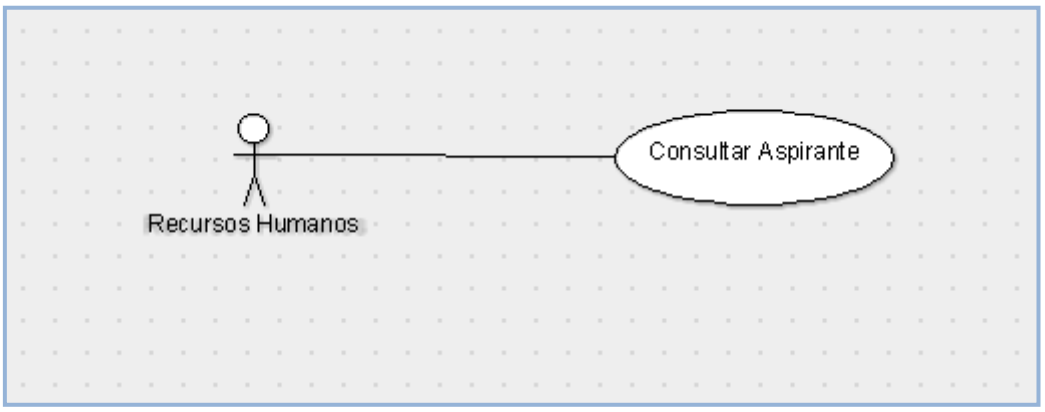


Figura N° 5.15: Consultar Aspirante (Casos de Usos)

- **Descripción:** La persona encargada de manejar el sistema ingresa al mismo para consultar los aspirantes seleccionados al puesto
  - La persona conoce la dirección URL de sistema CGA
  - Debe conocer los puestos de trabajo disponibles

Flujo de Eventos Normales:

Usuario		**Sistema**		
Paso	**Acción**	**Paso**	**Acción**	**Excepción**
1	Cuando el usuario está en la página selecciona el vinculo consultas	2	Despliega la información de los resultados.	E1
3	Selecciona por el puesto o por la cedula para consulta sus datos			
4	Selecciona consultar	5	Se despliega la información solicitada de la cual se puede hacer reportes	E1, E2

Tabla N° 5.1: Flujo de Eventos Normales (Casos de usos)

Flujo de Eventos Alternativos

Id	Nombre	Acción
E1 **Tabla**	La conexión con la base de datos no está establecida o se interrumpió	Se manda un mensaje de error, el cual indica que los datos no se pueden mostrar debido a que no hay conexión con la base de datos.
E2 **N°**	No se ha seleccionado ninguna vacante	No hace nada

Tabla N°5.2: Flujo de Eventos Alternativos (Casos de usos)

5.2.4 Prototipo de la Interfaz

Un prototipo de interfaz es una representación parcial de la interfaz de usuario que tendrá el software que interactúa con el usuario. El prototipo es un elemento muy importante para la comunicación con el usuario.

Se diseñan las pantallas de ingreso, consulta actualización, etc.

En un prototipo debe constar lo siguiente:

- Las opciones de un menú en el sistema
- Los nombres de las pantallas
- Los nombres de los botones.

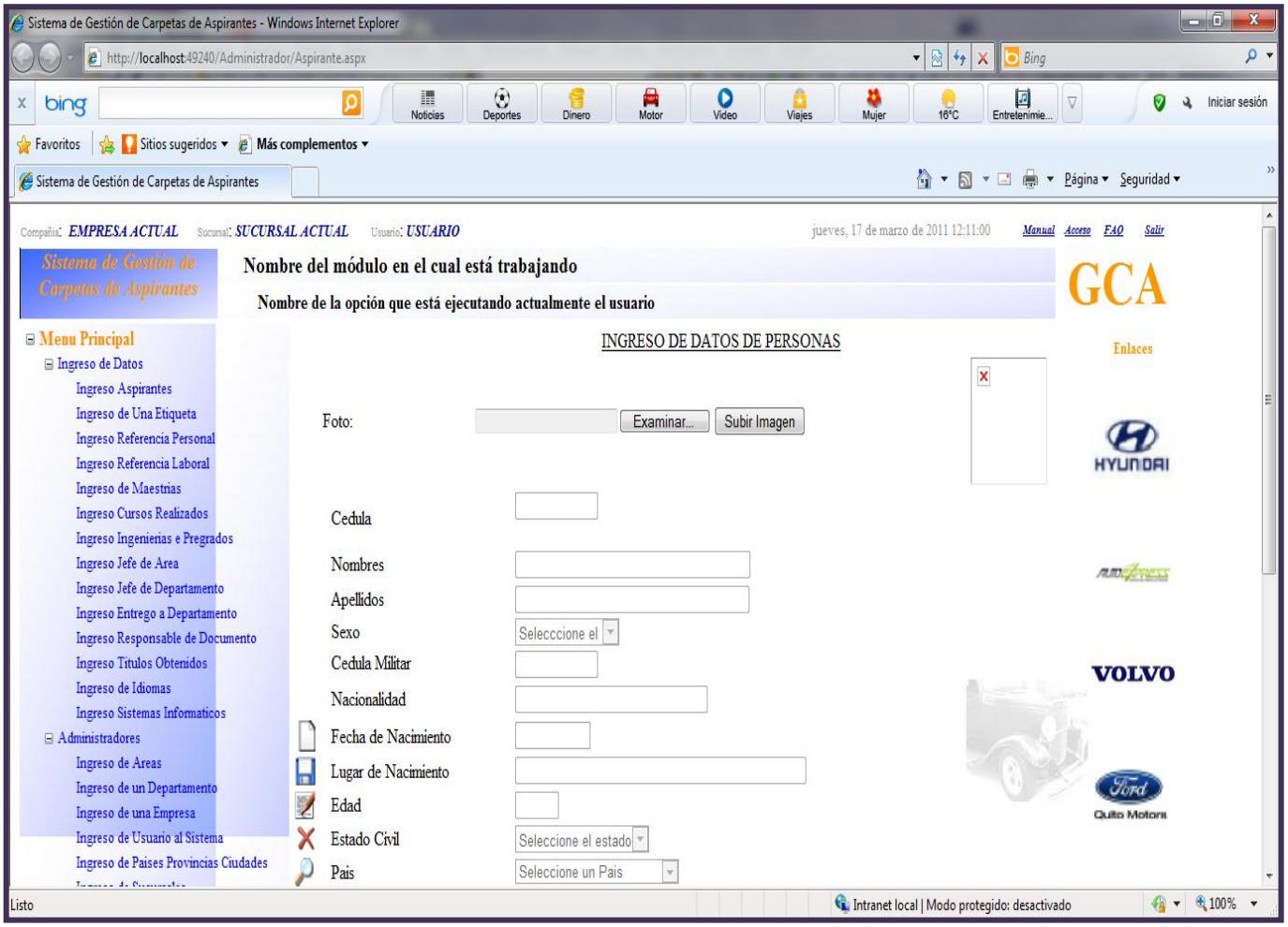


Figura N° 5.16: Prototipo de la Interfaz (Ingreso de un Aspirante)

5.2.5 Requerimiento no Funcionales

Los requerimientos no funcionales tienen que ver con los atributos o características que la empresa solicita para el funcionamiento del software:

Los requerimientos pueden ser:

- **Interfaz con el usuario:** Describe las características que permite al software apoyar al usuario en sus tareas. El software es vía web, animado y fácil de usar para que no sea complicado que el usuario se adapté al mismo, la interfaz debe ser agradable con colores suaves que no afecte la vista. Debe estar activo todo el día y a toda hora.
- **Interfaz externa:** Relación con otros sistemas. El sistema es parecido a una página web común y corriente.
- **Confiabilidad:** Solicitud del desempeño respecto a seguridad, tolerancia a fallas y su recuperación. El sistema si puede superar con

facilidad alguna falla presentada en la misma debido que se dispondrá de dos servidores en donde estarán las bases es decir tendrá un respaldo. Además los datos ingresados son confidenciales para el aspirante.

- **Eficiencia:** Limites de tiempo de respuesta. La respuesta de los resultados es muy rápido en su procesamiento, esto implica que el sistema no debe contener imágenes muy pesadas.
- **Mantenimiento:** Facilidad de compresión y realización de modificaciones futuras. Ya existen nuevas modificaciones al sistema pero eso es en un proyecto más adelante, como puede ser unir todas las sucursales en una sola base y que todas las manejen vía web, los reportes que se desplieguen vía Excel o Word, como ha sido la manejar de trabajar.
- **Portabilidad:** Facilidad de transferencia de un ambiente de ejecución a otro. El sistema es portable es decir se ha creado un archivo ejecutable para que pueda ser instalado en otra PC.
- **Restricciones de diseño y construcción:** Las que imponga el cliente.
    - Sera vía web
    - Estará programado en C #
    - La base de datos será SQL Server
    - Para el desarrollo de utilizará las siguientes herramientas
        - Modelar el sistema con los diagramas UML
        - Para su codificación Visual Studio 2008
        - Microsoft Word para la documentación
        - Para el desarrollo del sitio Web ASP.NET
        - SQL Server para la base de datos
        - Internet Information Server para la configuración del sitio Web
- **Legales y reglamentos:** Necesidades impuestas por leyes o reglamentos de otros.

5.3 Análisis

5.3.1 Introducción al Análisis

El objetivo del análisis es revisar los requerimientos para crear una descripción abstracta del sistema a construir, es decir construir el modelo del análisis que consiste en identificar los elementos con los que se construirá el sistema que va a hacer las clases.

5.3.2 Vista Estática

Interactúan las siguientes clases:

- **Clase de interfaz:** Es la interfaz de usuario. Son los elementos que interactúan directamente el usuario:
  - Las Ventanas de ingreso, consulta, eliminación
  - Páginas web de ingreso usuario
  - Informes de las consultas
- **Clase de control:** Son las reglas de negocio o aplicación, son los elementos que implementan las reglas de negocio y la lógica de la aplicación.

Reglas del Negocio

Las reglas del negocio es introducirse en cada uno de los casos de uso del negocio identificados, para describirlo en detalle. Esta descripción puede ser validada fácilmente por los usuarios. Determinar los agentes internos que juegan un rol en cada caso de uso del negocio. Los roles del caso del uso del negocio: Registrar aspirante son Persona, Recursos Humanos, Jefe del Departamento.

- La persona entrega su carpeta, que debe incluir su hoja de vida y sus certificados de trabajos anteriores, esto es recibido por la secretaria o recepcionista de la empresa.
- La recepcionista hace llegar la carpeta al Departamento de recursos humanos quien ingresa la información en el sistema.

- La persona encargada de administrar la información debe considerar:
    - Si el aspirante cumple con los requisitos solicitados para el puesto es aceptado, su carpeta para otra preselección.
    - En caso contrario, no es considerado como candidato para el puesto.
- Después de obtener los candidatos preseleccionados es llevado al jefe departamento para el cual se está solicitando el puesto.
- El jefe de departamento que solicita el puesto envía un informe de la persona aceptada.

- **Clase de entidad:** Las bases de datos, archivos, son los elementos que guardan la información para asegurar su persistencia. Los datos serán almacenados en una base de datos y será confidencial.

5.3.2.1 Diagrama de Clases

Los diagramas de clases se usan para modelar gráficamente la vista estática del software. Contiene los siguientes elementos:

- **Clase:** Describe un conjunto de objetos que comparten los mismos atributos, operaciones, métodos, relaciones y comportamiento.

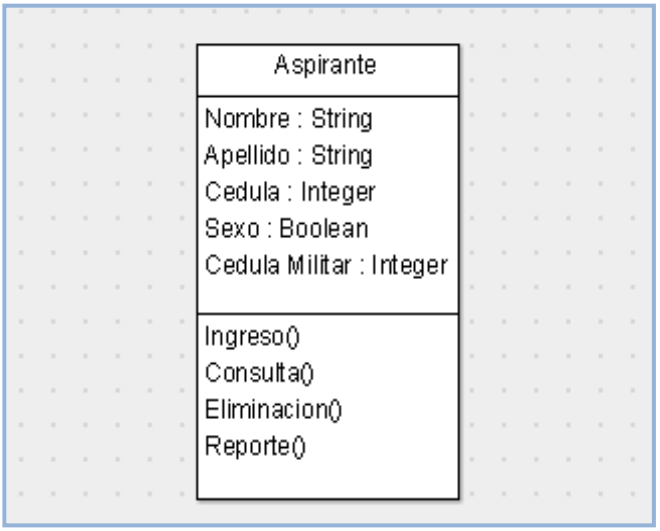


Figura N° 5.17: Ejemplo de Diagrama de Clase

- **Relación:** Muestra la dependencia entre dos o más clases, los tipos principales de relaciones entre las clases son: asociativas, agregación y generalización.

Asociación: Se representa por líneas que conecta las clases, estas líneas describen ligas entre objetos de las clases.

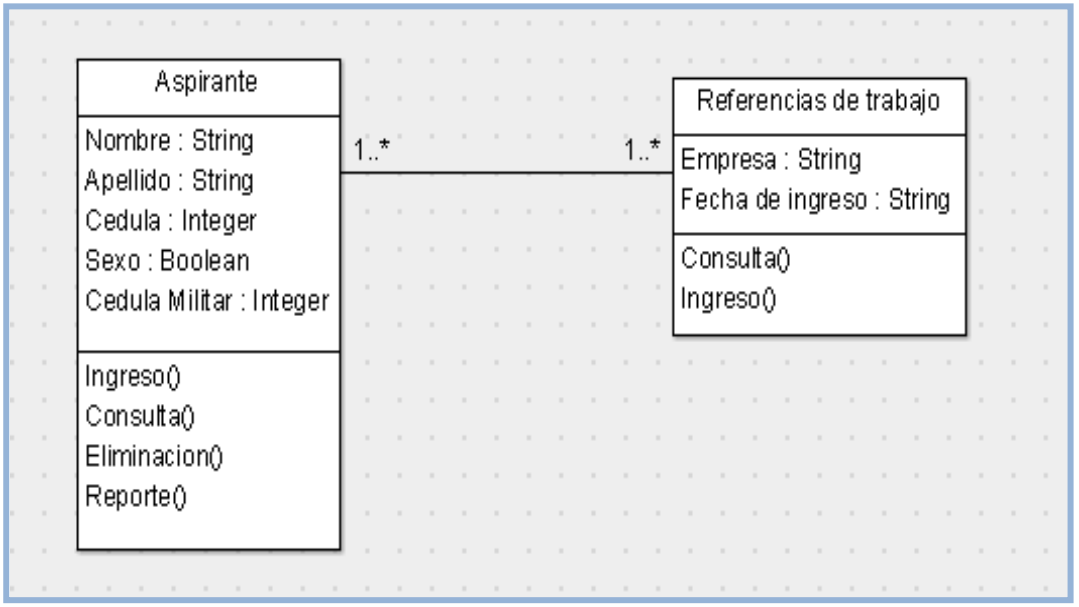


Figura N° 5.18: Ejemplo de la Relación Asociación (Diagrama de Clase)

Esta relación entre las clases aspirante y referencias de trabajo indica que uno o más aspirantes pueden tener uno o más referencias de trabajo

- **Agregación:** Este tipo de asociación se representa por una línea que es la composición y un rombo, la clase aspirante está compuesto por los objetos de la clase referencias de trabajo, es decir un libro está compuesto por varias hojas, en nuestro caso el aspirante está compuesto por las referencias de trabajo.

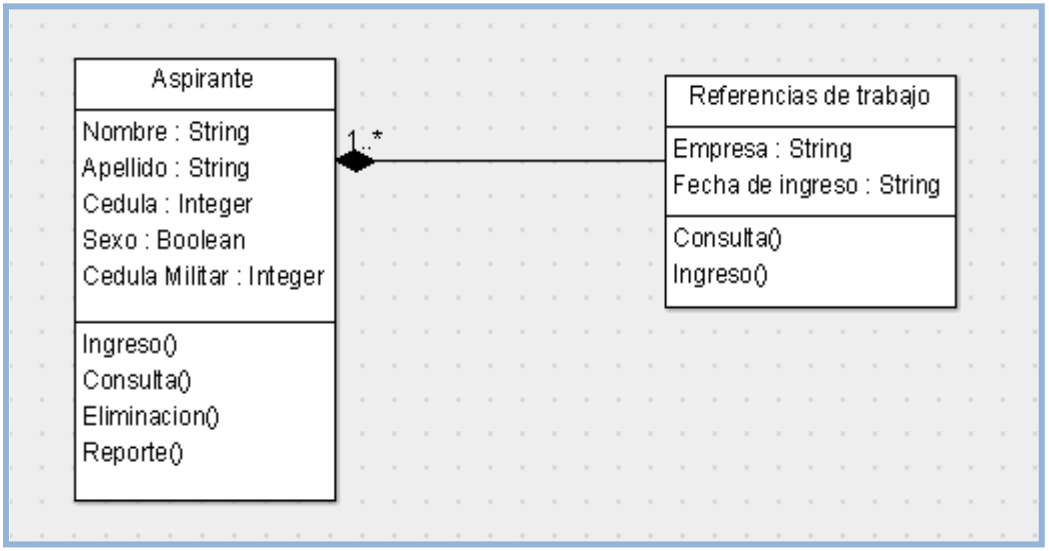


Figura N° 5.19: Ejemplo de la Relación Agregación

(Diagrama de Clase)

- **Generalización:** Relaciona una clase general o abstracta que comparte sus atributos con clases que los especializan las subclases heredan sus atributos. La relación entre la clase general y sus especializaciones se identifican por un triangulo del lado de la clase general y un línea.

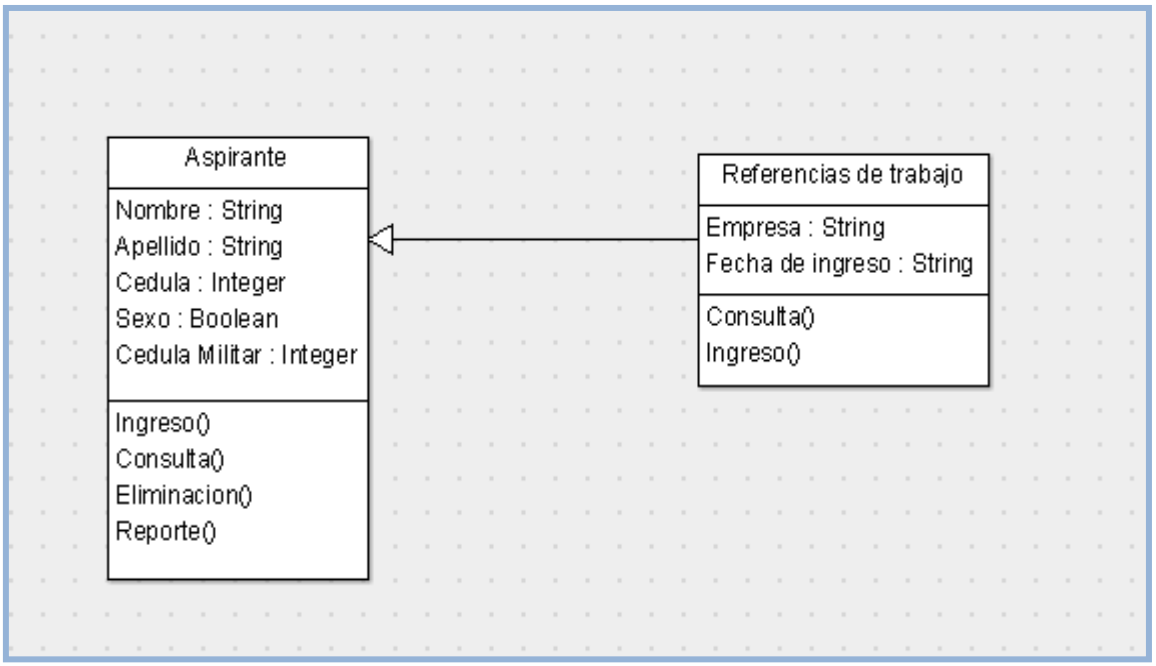


Figura N° 5.20: Ejemplo de la Relación Generalización

(Diagrama de clase)

Identificación de Clases

Para identificar las clases, se analizan cada caso de uso. Se recomienda iniciar por la identificación de las clases de control. Para el proyecto tenemos un clase administrar aspirante, que representa a los aspirantes y por lo tanto tenemos una clase aspirante como una clase de control. Los atributos de esta clase son los datos personales y sus métodos pueden ser:

- Ingreso,
- Modificación
- Eliminación,
- Consulta,
- Reporte.

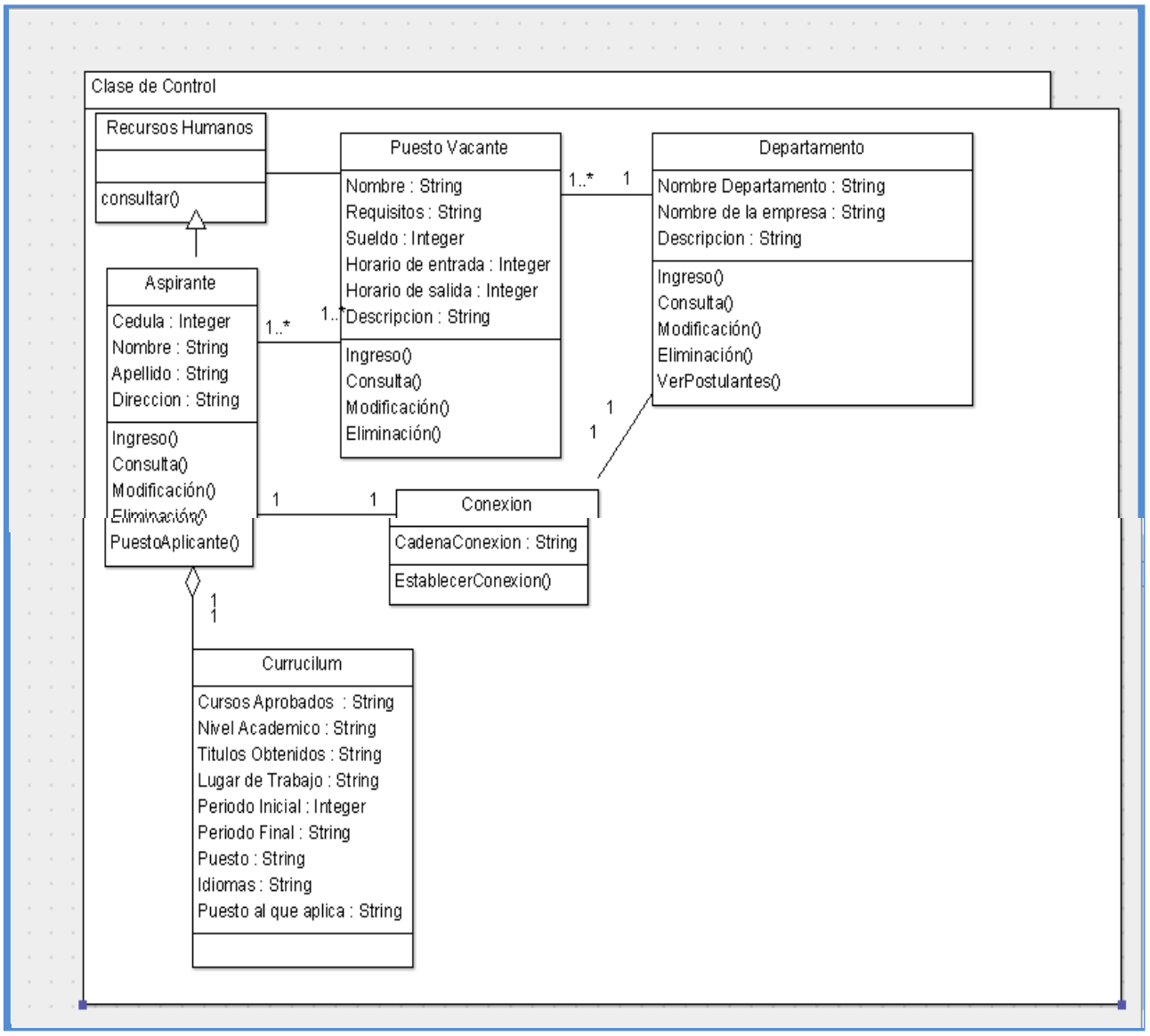


Figura N° 5.21: Diagrama de Clase de Control

El diagrama de clase muestra los pasos a seguir en un puesto vacante de la empresa. La clase general Recursos Humanos se puede especializar en Puesto vacante por lo que es una relación de herencia. La clase aspirante tiene relación de agregación con la clase currículo, casa aspirante tiene un currículo la clase puesto vacante está asociada con las clases departamento y aspirante.

Clases de Interfaz

Después de identificar las clases de control se procede a identificar las clases de interfaz de usuario. Las clases de interfaz se podrán implementar con diversas tecnologías como serán ventanas, código HTML, etc. que se deciden en el diseño.

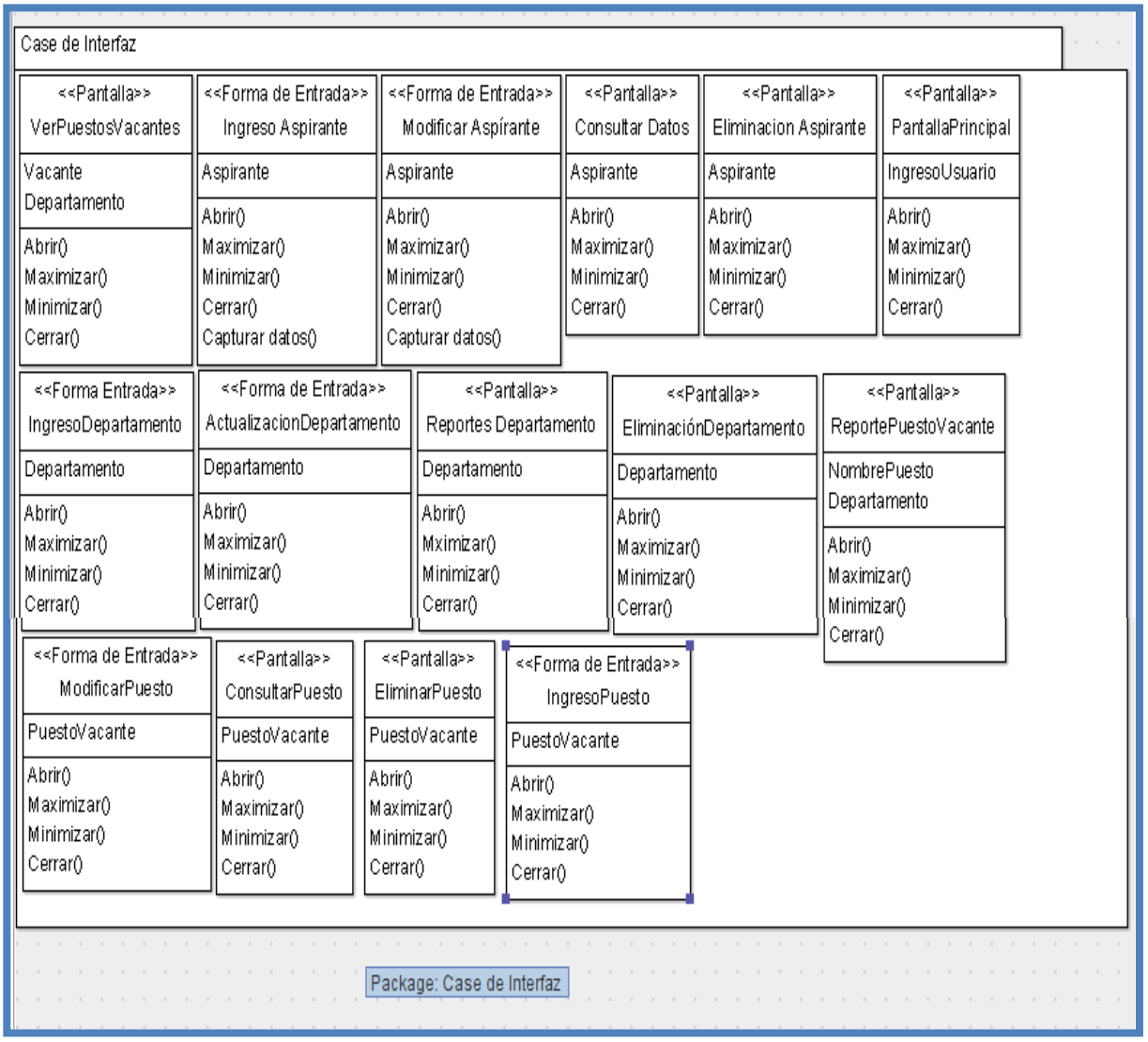


Figura N° 5.22: Diagrama de Clase de Interfaz

Clase de Entidad

Para identificar las clases de identidad se parte de las clases de control, estas clases de dibujan de tipo entidad, que resguardan los atributos en una base o un archivo.

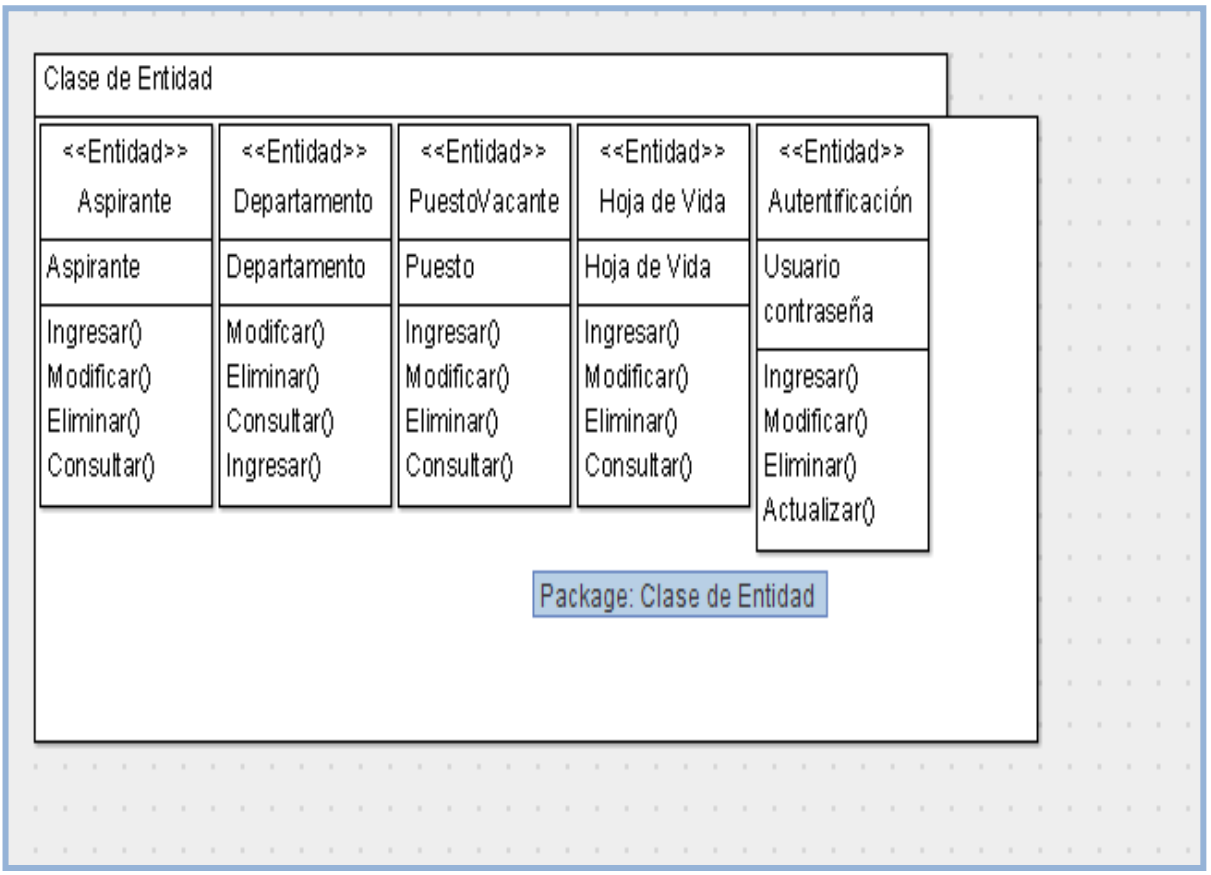


Figura N° 5.23: Diagrama de Clase de Entidad

5.3.3 Vista Dinámica

La vista dinámica describe cómo interactúan los objetos para entregar la funcionalidad requerida del sistema. Existen algunos diagramas que permiten esta interacción es decir para representar las vistas, que se describen a continuación.

5.3.3.1 Diagrama de Secuencia

Los diagramas de secuencia muestra la interacción entre los objetos de las clases como una secuencia de envió de mensajes.

- **Objetos:** Son instancia de las clases, se los representa por un rectángulo con el nombre subrayado de la clase a la que pertenece.

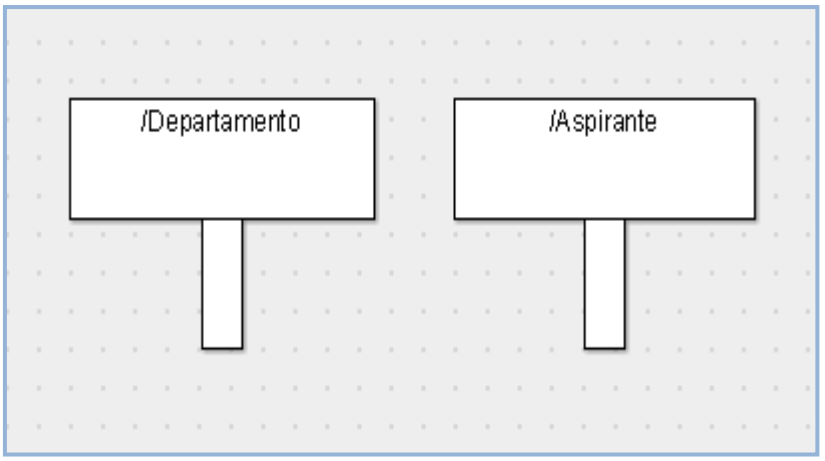


Figura N° 5.24: Diagrama de Secuencia de Objeto

- **Mensaje:** Son enviados de un objeto fuente a otro receptor, se representa con una línea con flecha del objeto fuente al receptor.

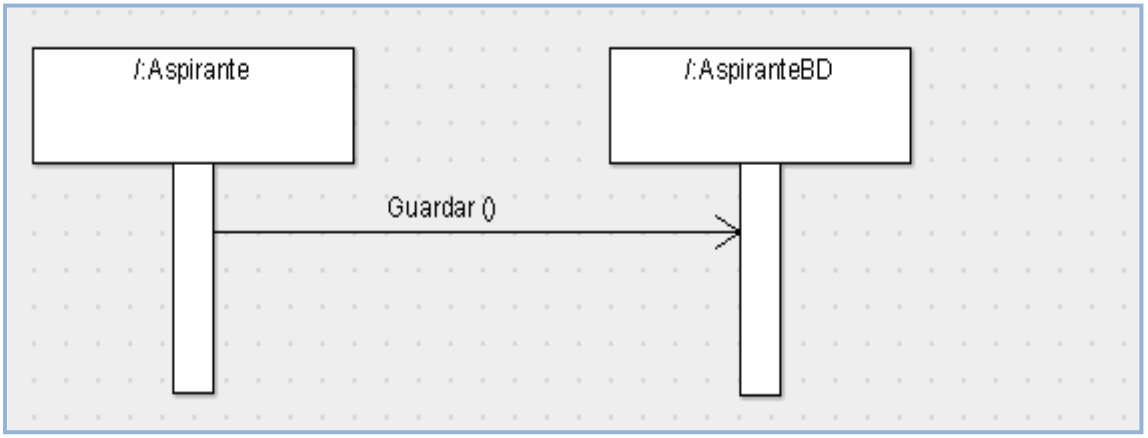


Figura N° 5.25: Diagrama de Secuencia de Mensaje

- **Actor:** Es el indicador de la invocación al método en la secuencia de mensajes.

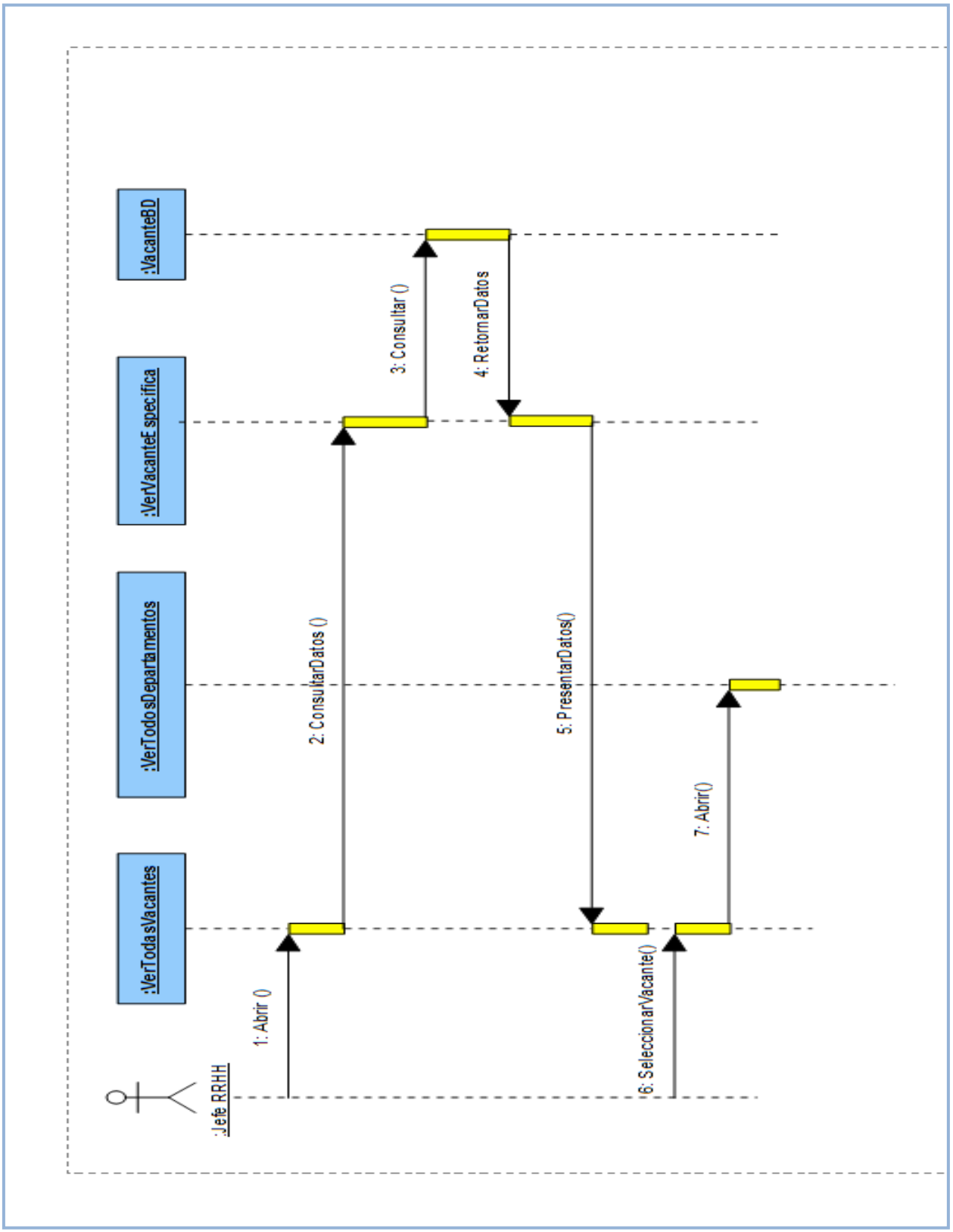


Figura N° 5.26: Diagrama de secuencia

5.3.3.2 Diagrama de Estado

Modela una maquina de estados finitos, que enfatiza el flujo de control de un estado a otro, se representan con nodos que significa estados.

- **Estado:** Es la condición de una entidad del sistema, puede caracterizar a un objeto o representar una pantalla, se representa por rectángulos con esquinas redondeadas con el nombre del estado.

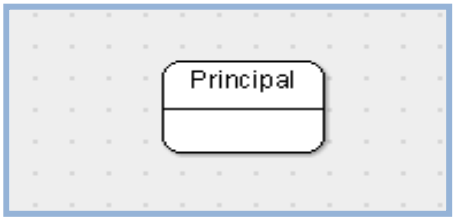


Figura N° 5.27: Diagrama de Estado (Estado)

- **Evento:** Significa que ocurre en un momento y que causa el cambio de estado
- **Transición:** modela el cambio de estado a causa de un evento, se representa por una flecha que puede etiquetar el nombre del evento y va de un estado a otro.

Figura N° 5.28: Diagrama de Estado (Transición)

- **Estado Inicial:** Se representa con un punto negro

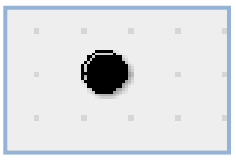


Figura N° 5.29: Diagrama de Estado (Estado Inicial)

- **Estado Final:** Se representa por un circulo negro en el centro, puede haber varios estados finales

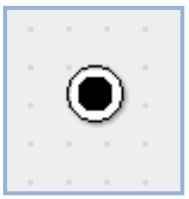


Figura N° 5.30: Diagrama de Estado (Estado Final)

Construcción del Diagrama de Estado

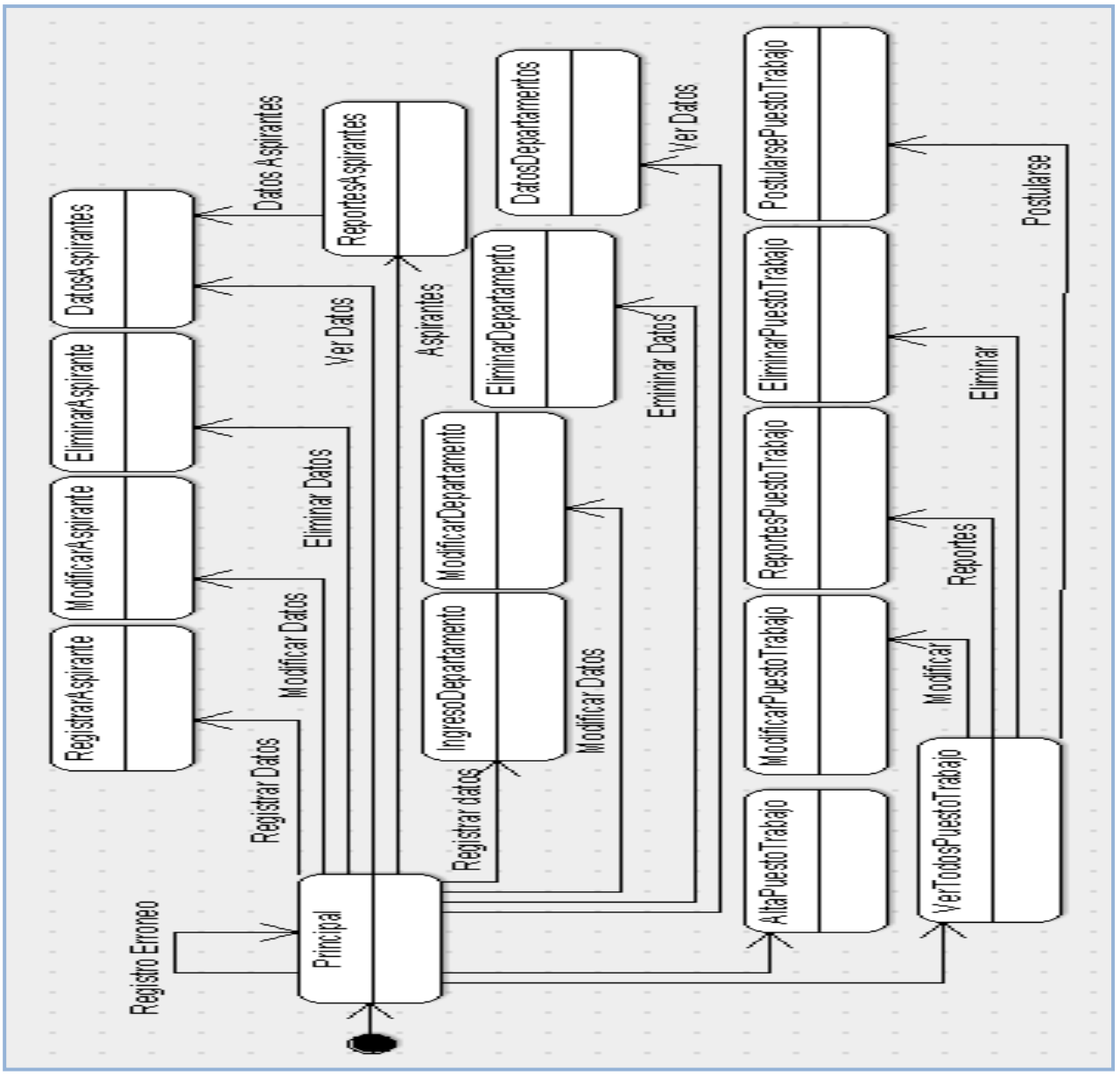


Figura N° 5.31: Diagrama de estado

El estado inicial representa a la pantalla principal, se procede a los estados.

5.4 Diseño

5.4.1 Introducción al Diseño

El diseño del software es un proceso creativo para construir el producto del software. Los objetivos del diseño son:

- Identificar
- Caracterizar los componentes principales del software
- Definir su interacción e integración en el producto

5.4.1.1 Principios del Diseño

Diseñar para el Cambio: Hoy en día el software cambia constantemente, por lo que es fundamental anticiparse a los cambios, esto quiere decir que el diseño debe ser flexible para que permita realizar cambios.

Diseñar para facilitar el Uso del Software: Es importante tener en mente al momento de diseñar a los usuarios de software y sus aptitudes quienes serán los que van administrar el software se debe formar una naturaleza agradable para el trabajo.

Diseñar para facilitar la Prueba: Esta enfocado en el desarrollo que probará el sistema, se identifican los componentes del sistema como unidades que se puedan probar sin necesidades de incluir otros componentes, se le realizar pruebas en cada modulo para revisar el avance del software.

Diseñar para la Reutilización: Una manera de mejorar la productividad del equipo en proyecto futuros o en ciclos siguientes, se define partes genéricas que puedan volver a usarse, esto incluye no solo el nivel del diseño sino de código, casos de

pruebas modelos o diagramas.

El software está abierto a modificaciones tanto en código como en los diagramas.

Cohesión: Es el grado de relación entre los elementos que pertenecen a un componente. El grado de cohesión de un componente se mide al escribir una frase que describa el objetivo del componente:

- Si la frase tiene un solo verbo el grado de cohesión en fuerte
- Si la frase compuesta contiene más de un verbo o contiene comas tiene una cohesión débil y habrá que definir un componente para cada verbo. Ingresar, Consultar, Modificar, etc.

Acoplamiento: Es el grado de relación entre los componentes. Un buen diseño tiene un acoplamiento débil entre sus componentes

5.4.2 Arquitectura del Software

Se les denominada arquitectura de software, porque a semejanza de los planos de un edificio o construcción, estas indican la estructura, funcionamiento e interacción entre las partes del software.

Consiste en producir un modelo o representación técnica del software que se va a desarrollar, la arquitectura indica los elementos más importantes del sistema como son las relaciones, podemos tener una visión global de todo el sistema, el diseño arquitectónico comienza con el diseño de datos que facilita la representación de los componentes de datos y el diseño arquitectónico se centra en la representación de la estructura de los componentes

La arquitectura de software es el diseño del más alto nivel, se define cuales serán los componentes que formarán el software. La arquitectura debe favorecer el cumplimiento de los requerimientos funcionales y no funcionales que fueron detallados en los diagrama de casos de usos.

La arquitectura del software se ha conformado por los componentes de aspirante departamento puesto vacante y se ha cumplido los requerimientos funcionales y no funcionales.

Las cualidades que debe tener la arquitectura son:

- Sencillez es fácil de comprender y de implementar, el software es fácil de entender el código de programación y su implementación no es difícil, solo es necesario una PC con acceso a internet.
- Extensión la posibilidad de agregar nuevos componentes, el software tiene la posibilidad de adaptarse a más componentes como puede ser sucursales, empresas para utilizar a otras empresas.
- Cambio los requerimientos no afecten mucho a la arquitectura.

Arquitectura Orientada a Objetos

Los componentes de un sistema encapsulan los datos y las operaciones que se deben realizar para manipular los datos, la comunicación y la coordinación entre los componentes se consigue a través del paso de mensajes.

Arquitectura Cliente-servidor

La arquitectura cliente-servidor consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta.

Aunque esta idea se puede aplicar a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma.

Una disposición muy común son los sistemas multicapa en los que el servidor se descompone en diferentes programas que pueden ser ejecutados por diferentes computadoras aumentando así el grado de distribución del sistema.

La arquitectura cliente-servidor sustituye a la arquitectura monolítica en la que no hay distribución, tanto a nivel físico como a nivel lógico. La red cliente-servidor es aquella red de comunicaciones en la que todos los clientes están conectados a un servidor, en el que se centralizan los diversos recursos y aplicaciones con que se cuenta; y que los pone a disposición de los clientes cada vez que estos son solicitados. Esto

significa que todas las gestiones que se realizan se concentran en el servidor, de manera que en él se disponen los requerimientos provenientes de los clientes que tienen prioridad, los archivos que son de uso público y los que son de uso restringido, los archivos que son de sólo lectura y los que, por el contrario, pueden ser modificados, etc. Este tipo de red puede utilizarse conjuntamente en caso de que se esté utilizando en una red mixta.

Características Cliente Servidor

En la arquitectura C/S el remitente de una solicitud es conocido como cliente. Sus características son:

- Es quien inicia solicitudes o peticiones, tienen por tanto un papel activo en la comunicación (dispositivo maestro o amo).
- Espera y recibe las respuestas del servidor.
- Por lo general, puede conectarse a varios servidores a la vez.
- Normalmente interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.
- Al contratar un servicio de redes, se tiene que tener en la velocidad de conexión que le otorga al cliente y el tipo de cable que utiliza , por ejemplo : cable de cobre ronda entre 1 ms y 50 ms.

Al receptor de la solicitud enviada por el cliente se conoce como servidor. Sus características son:

- Al iniciarse esperan a que lleguen las solicitudes de los clientes, desempeñan entonces un papel pasivo en la comunicación (dispositivo esclavo).
- Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.

- Por lo general, aceptan conexiones desde un gran número de clientes (en ciertos casos el número máximo de peticiones puede estar limitado).
- No es frecuente que interactúen directamente con los usuarios finales.

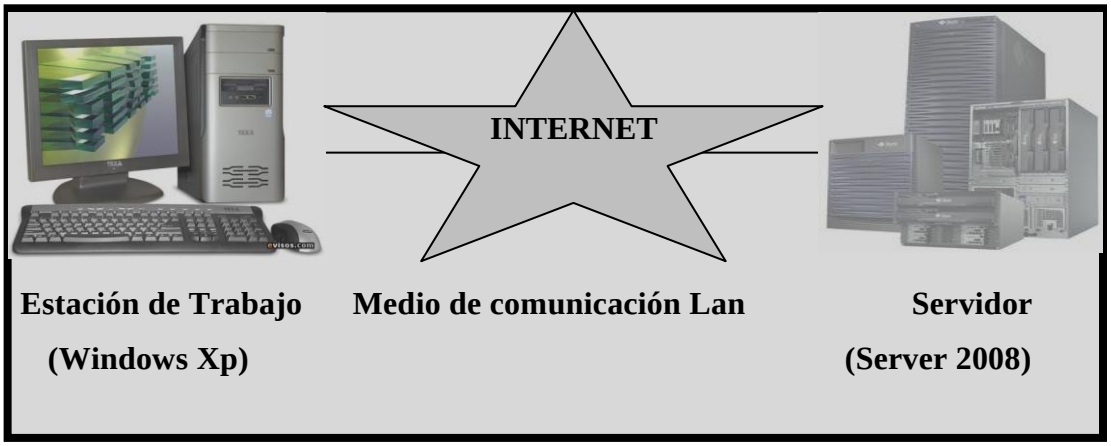


Figura N° 5.32: Arquitectura Cliente Servidor

Programación por Capas

La programación por capas es un estilo de programación en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario.

El modelo de capas es la base de la arquitectura de un sistema. Una capa es una abstracción que toma el resultado de la capa inferior y efectúa la función de entregar resultados a la parte superior. En el proyecto las capas tomas datos de la base de datos y los lleva a la aplicación para su presentación, se define la arquitectura del sistema.

La ventaja principal es que el desarrollado del software se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado.

Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles, de forma que basta con conocer la API que existe entre niveles.

En el diseño de los sistemas informáticos actuales se suelen usar las arquitecturas multinivel o Programación por capas. En dichas arquitecturas a cada nivel se le confía una misión simple, lo que permite el diseño de arquitecturas escalables (que pueden ampliarse con facilidad en caso de que las necesidades aumenten).

El diseño utilizado es el diseño en tres niveles (o en tres capas). Tenemos las siguientes capas:

- **Capa de presentación:** En esta capa se ponen los elementos con los que interactúan directamente el usuario del software: las ventanas de ingreso, consulta, reportes, informes, páginas.
- **Capa lógica de la aplicación:** Son los elementos que implementan las reglas del negocio y la lógica de programación que fueron explicados, detallados en el capitulo anterior.
- **Capa de almacenamiento:** Contiene los elementos que guardan la información para asegurar su persistencia tales como base de datos archivos. Se utilizara la base de datos SQL server para el almacenamiento de la información.

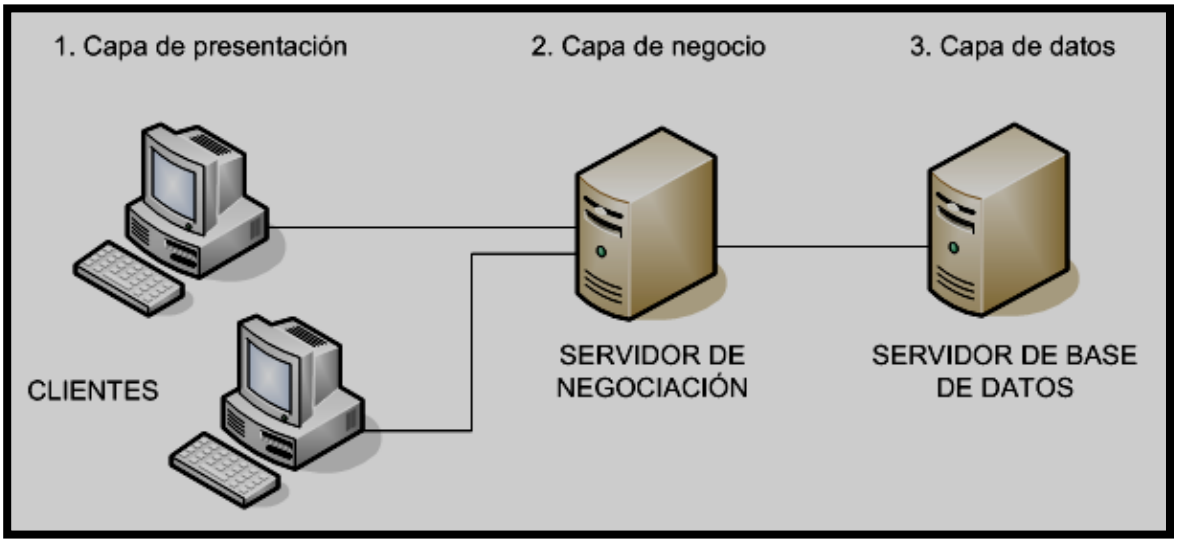


Figura N° 5.33: Programación por Capas

5.4.2.1 Diagrama de Paquetes

Sirven para representar las capas de la arquitectura, los paquetes pueden contener otros paquetes, clases, interfaces, etc., además contienen los paquetes y las relaciones:

- **Dependencia:** Se representan con una flecha punteada, un paquete B depende de otro paquete A. Ejemplo: aspirante necesita de vacante.
- **Asociación:** Se representa por una línea continua establece una relación entre dos paquetes, esta relación es bidireccional, departamento necesita de vacante y vacante necesita de departamento.
- **Generalización:** Se representa por línea continua y un triángulo transparente que apunta hacia el paquete mas general esto hace relación de herencia. Ejemplo: empresa
- **Realización:** Se representa por una línea punteada con un triangulo transparente.

Las capas que representan por paquetes son:

- **Capa de presentación:** Esta capa debe contener uno o varios paquetes que contiene la interfaz de usuario en la aplicación los prefijos que se utilicen en el servidor serán los mismos que se utilizara en la interfaz del cliente txt, cb.
- **Capa lógica de la aplicación:** Esta capa contiene varios paquetes uno para cada funcionalidad del sistema, por ejemplo departamento, aspirante, etc.
- **Capa de almacenamiento:** Es la base de datos o archivos que se utilizara para la aplicación.

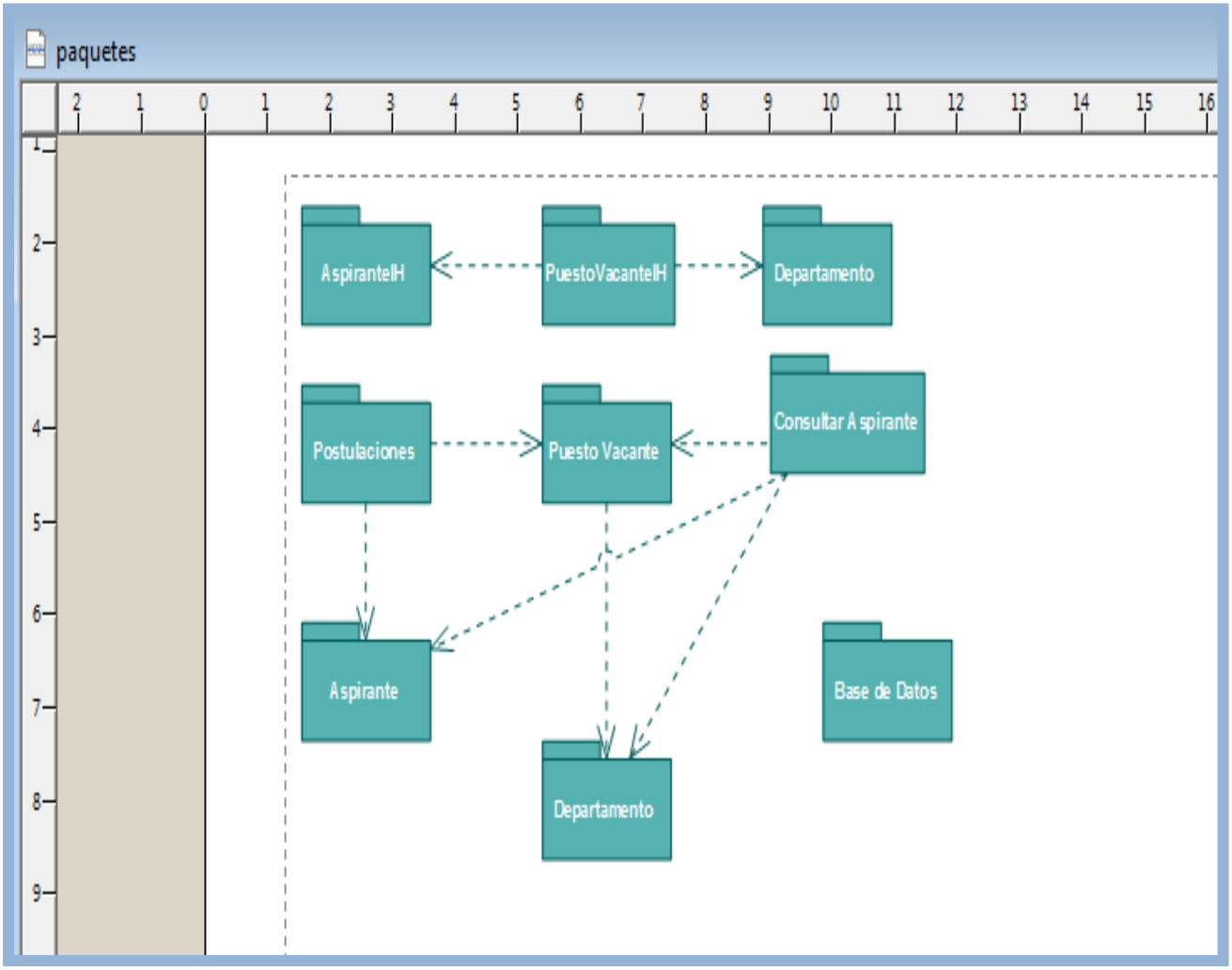


Figura N° 5.34: Diagrama de Paquetes

5.4.2.2 Diagrama de Distribución

Es un sistema distribuido por se ejecuta en más de una maquina y sus componentes se encuentran distribuidos en ellas. El sistema es distribuido porque se puede instalar en cualquier maquina que tenga internet y pueden usar más de un usuario a la vez pero la finalidad del proyecto es solo la matriz por lo que será un poco difícil la navegación habrá pérdida de tiempo es un proyecto a futuro implementar a as sucursales

Los elementos de estos diagramas son: los nodos y las conexiones entre ellos.

- Un nodo es un elemento físico que existe y representa un recurso computacional, se representa por un cubo que debe ser nombrado (computador, monitor, teclado, etc.).
- Las conexiones representan las comunicaciones entre los nodos. Puede etiquetarse como nodo de comunicación (redes LAN, WAN, etc.).

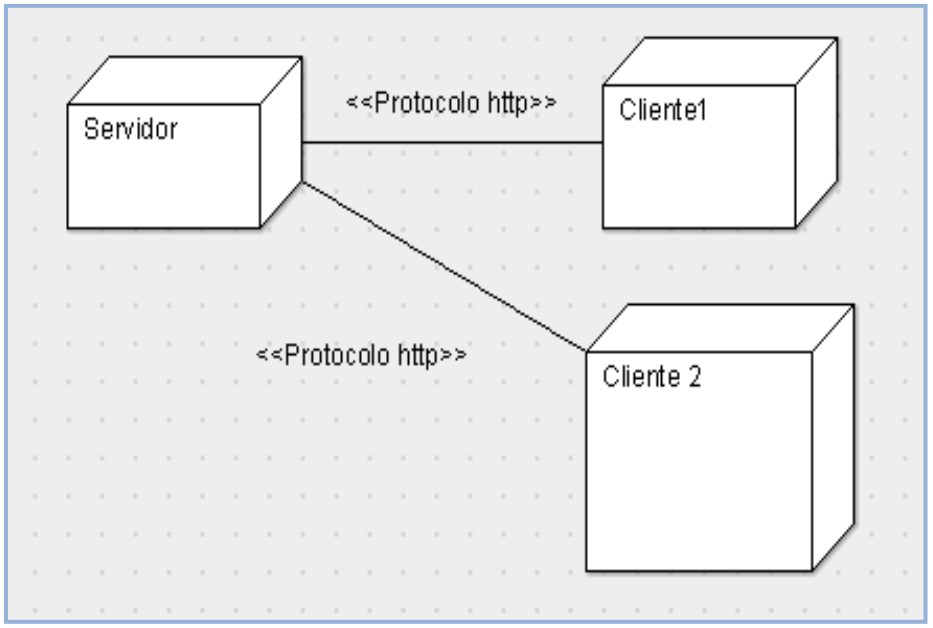


Figura N° 5.35: Diagrama de Distribución

Proceso para definir la Arquitectura

- Seleccionar el tipo de arquitectura de software según la aplicación. Se puede elegir la arquitectura de tres capas como sé lo ha hecho en el ejemplo del proyecto, existen otras arquitectura.
- Identificar los paquetes para la arquitectura del sistema, departamento, empresa, aspirante, etc.
- Construir el diagrama de paquetes.
- Identificar si es un sistema distribuido, identificar la lógica de distribución, el proyecto si es un sistema distribuido porque permite la navegación de varios usuarios.
- Construir el diagrama de distribución con los nodos y sus conexiones.

5.4.3 Construcción de Componentes

Un componente es como una caja negra con un comportamiento bien definido que encapsula una cierta parte del diseño y que proporciona para interaccionar con él. Los componentes tienen:

- **Interfaces de entrada:** Especifican lo que se debe proporcionar para la ejecución en el proyecto las entradas son los datos que el usuario desee consultar, también es el ingreso de información.
- **Interfaces de salida:** Especifican lo que entrega al ejecutarse son los reportes, consultas que el usuario necesite obtener información

Un diagrama de componentes muestra la organización y las dependencias entre un conjunto de componentes, cubre la vista estática y se relaciona con los diagramas de clase puesto que un componente puede contener una o más clase.

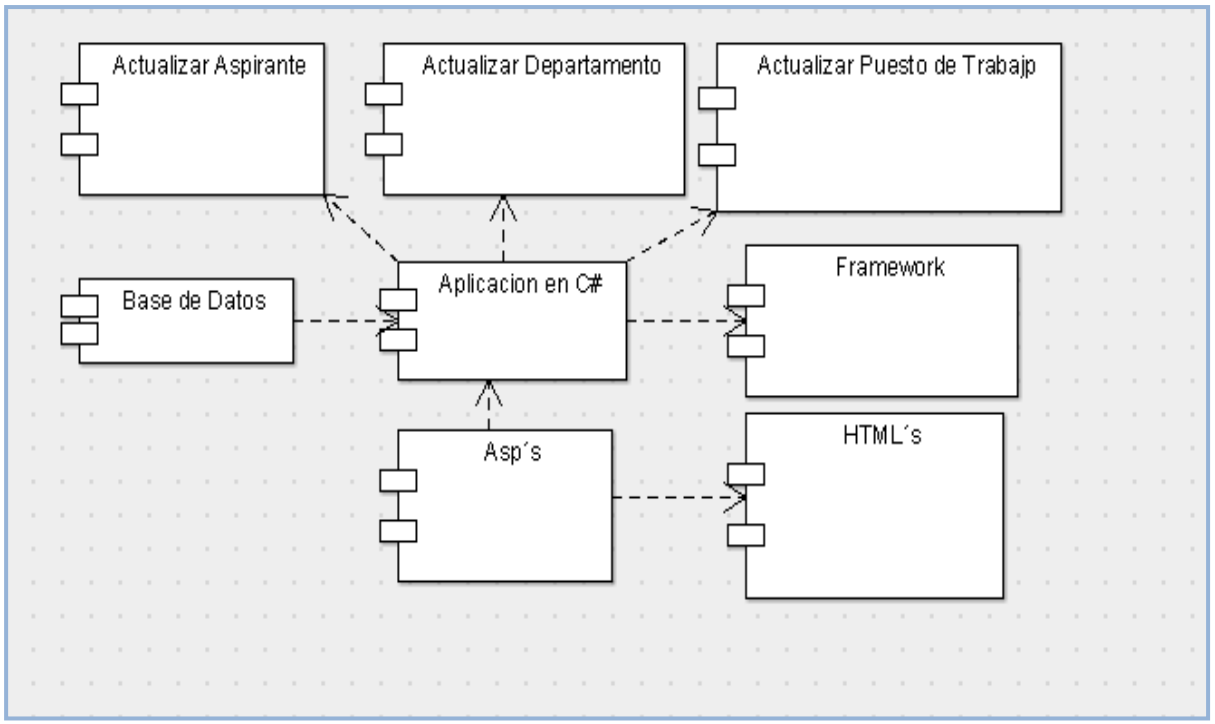


Figura N° 5.36: Diagrama de Componentes

El desarrollo de software basado en componentes favorece la reutilización de componentes y facilita el cambio.

5.4.4 Diseño de la Base de Datos

Después de las clases de tipo de la capa de almacenamiento se diseña la base de datos.

En el diagrama Entidad-Relación, se mapea cada clase persistente a una entidad, los atributos de las clases se convierten en los atributos de las entidades, las relaciones entre las entidades se construye a partir de las relaciones entre las clases.

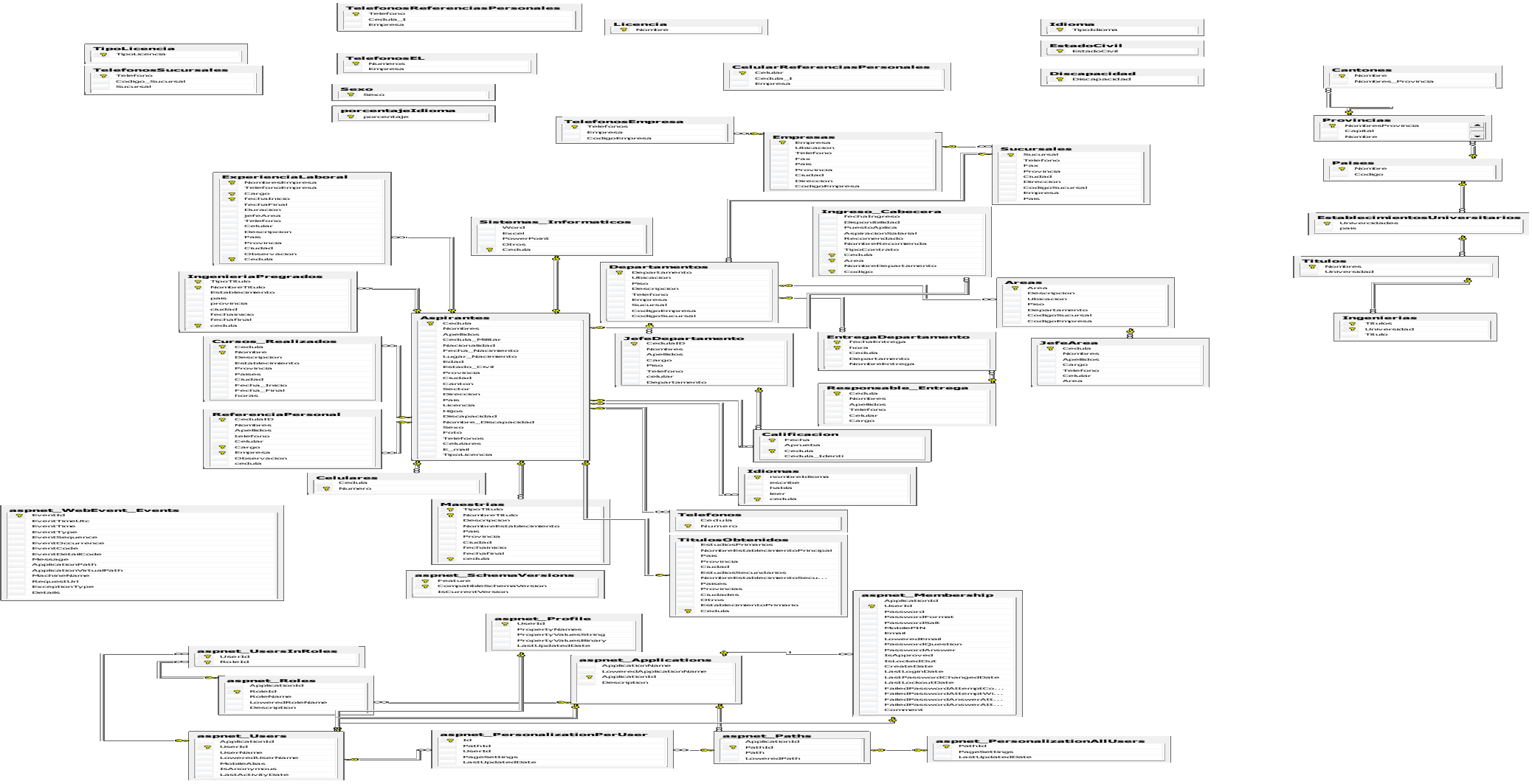


Figura N° 5.37: Modelo Entidad Relación

5.4.4.1 Conversión del Diagrama de Clase al Modelo de Datos de una Base de Datos Relacional

La estructura de una base de datos relacional es muy sencilla, pero esta sencillez dificulta el proceso de representar objetos completos.

Cada clase de diagrama de clase se convierte en el esquema de una tabla en la base de datos relacional, se puede utilizar el mismo nombre de la clase para la tabla, aunque es recomendables que el nombre este en plural, los campos de los atributos de la clase se convierte en una columna de la tabla.

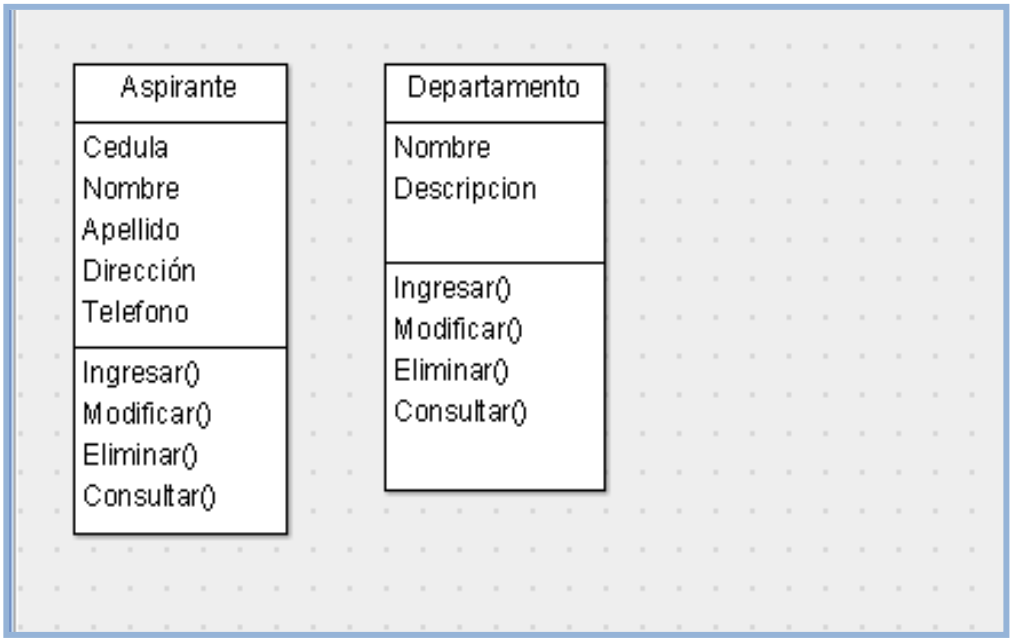


Figura N°: 5.38 Entidades de Clases

Para realizar el ejemplo del proyecto se trabajara con la clase aspirante

Nombre del esquema → **Aspirante**

Atributos →

Cedula	Nombre	Apellido	Dirección	Teléfono

Tabla: N°: 5.3 Relación de Entidades de Clases

En los sistemas manejadores de bases de datos relacionales la visibilidad de cada atributo es siempre pública. Para el dominio de los atributos se tiene que relacionar los tipos UML con los datos de SQL.

Atributo UML	Columna SQL
Char	char
String	char(254)
Short	smallint
Int	integer
Float	float
Date	date

Tabla N°: 5.4 Equivalencia de tipos entre UML y SQL

Los manejadores relacionales dependen completamente de identidad explícita, por lo que no puede existir un identificador de objeto que no sea un valor de una columna de la tabla, transforma la identidad explicita de los atributos significa poner los atributos como columnas y establecer las restricciones para la llave primaria sobre las columnas.

Conversión de Interfaces

Después de que la interfaz contenga operaciones únicamente es necesario implementar los procedimientos almacenados correspondientes, siempre y cuando sean adecuados para ejecutarse en el servidor de base de datos.

Conversión de Asociaciones

La base de datos relacional contiene las tablas y no asociaciones por lo que las asociaciones deben integrarse a las tablas a través de

columnas especiales. Las tablas son las clases descritas por ejemplo: aspirante, departamento, currículo, etc. Las asociaciones serán los campo, cédula, nombre, dirección. En la conversión de una asociación binaria a un esquema relacional se debe utilizar el concepto de llave externa. Una llave externa es un conjunto de columnas cuyo valor debe estar en la llave primaria de la tabla con la que se está conectado.

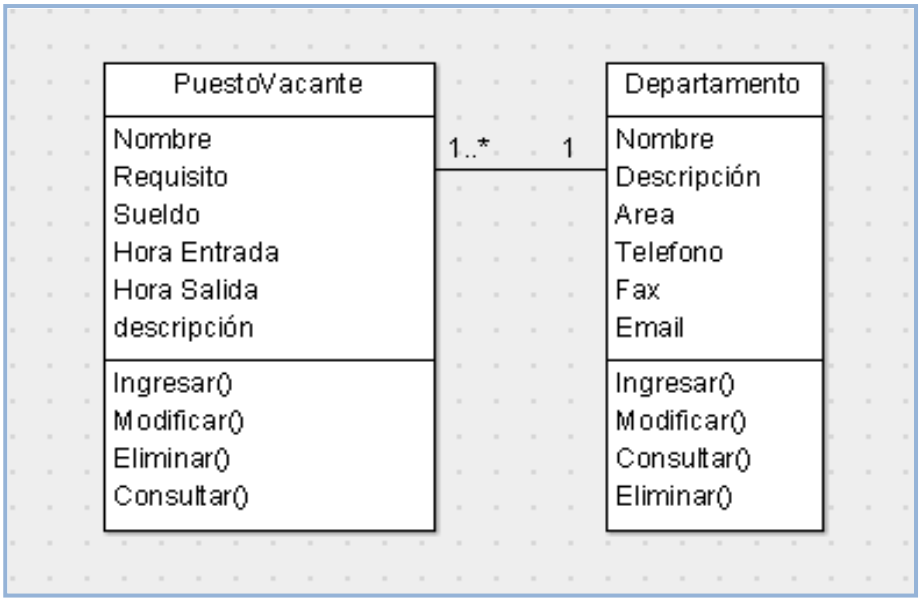


Figura N° 5.39: Relación de Asociaciones entre dos Clases

En la relación del grafico se puede apreciar que un departamento de la empresa tiene 1 o varias puesto vacantes y esa vacante es solo para ese departamento y no puede ser para otro.

Las características de la multiplicidad en la clase relacional se dividen en dos aspectos de interés para la transformación relacional:

- La tabla genera columnas y restricciones de nulidad sobre las columnas
- La multiplicidad contiene un máximo de 1 entonces la tabla correspondiente a esa clase tendrá columnas como llave

primaria que se incluirán en la tabla correspondiente a las clases asociadas como la llave externa.

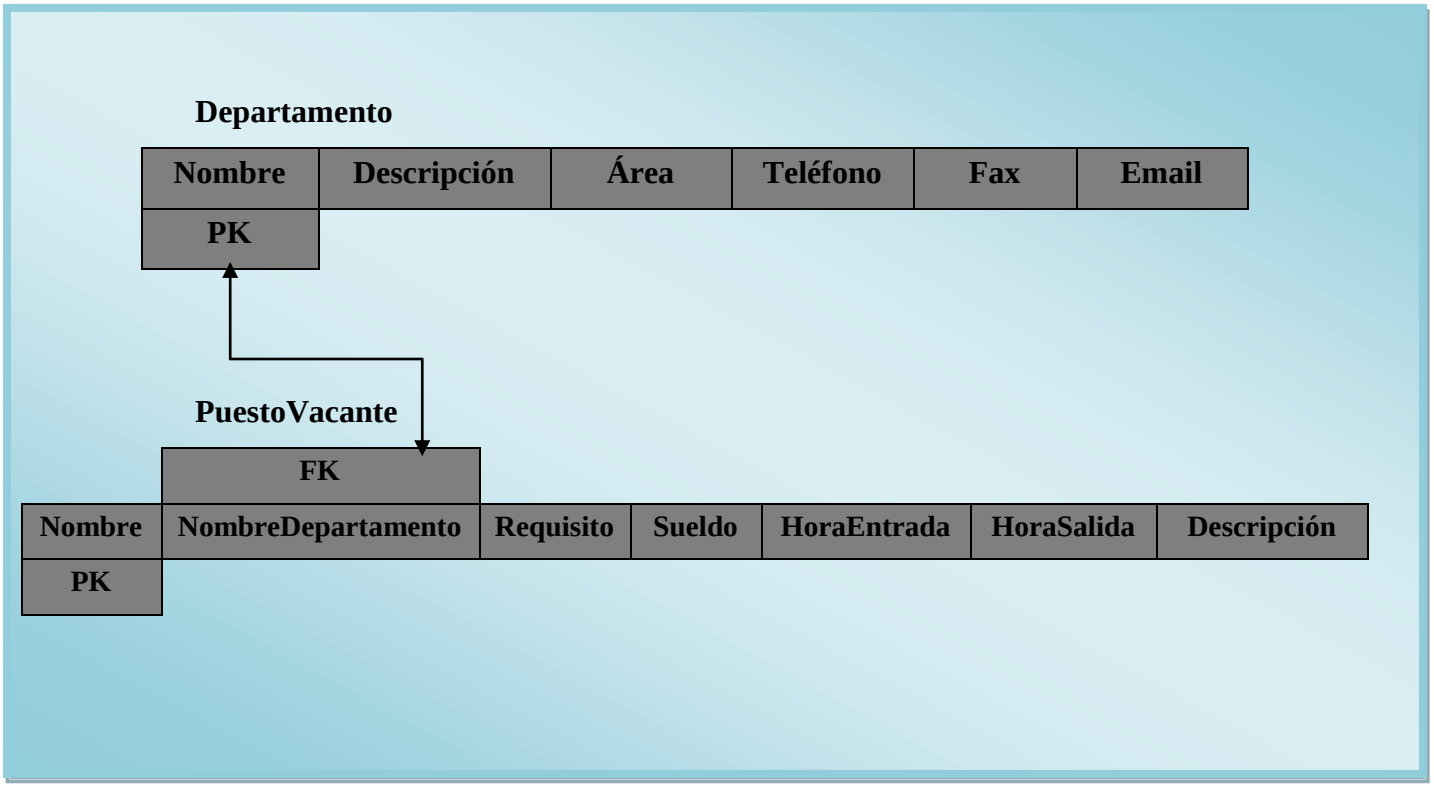


Tabla N° 5.5: Llaves Externas

En esta relación de la tabla departamento se define como llave primaría (PK) y en la tabla puesto vacante se tiene como nombre departamento como llave externa (FK) en donde se especifica que el nombre departamento debe ser en donde de algún departamento perteneciente a la Quito Motors. Estas tupas están relacionadas. En cada puesto vacante solo hay un departamento es decir cada puesto es colocada por un departamento.

Si la multiplicidad es de varios a varios no se puede representar directamente en el esquema relacional, es necesario crear una tabla en la base de datos para esta asociación.

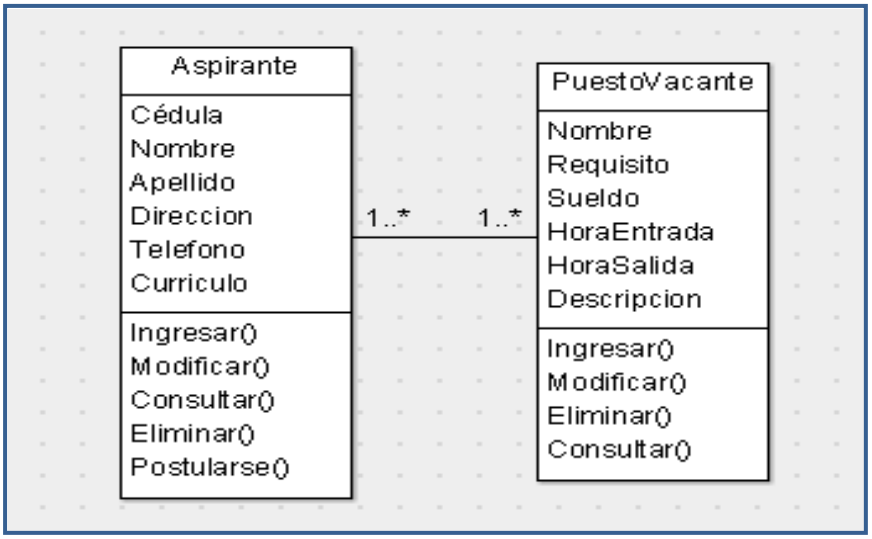


Figura N° 5.40: Asociación n: n

La asociación varios a varios aspirantes y puesto vacante se convierte en una relación uno a varios entre aspirante y aspirante puesto vacante más una relación muchos a uno entre aspirante puesto vacante y puesto vacante.

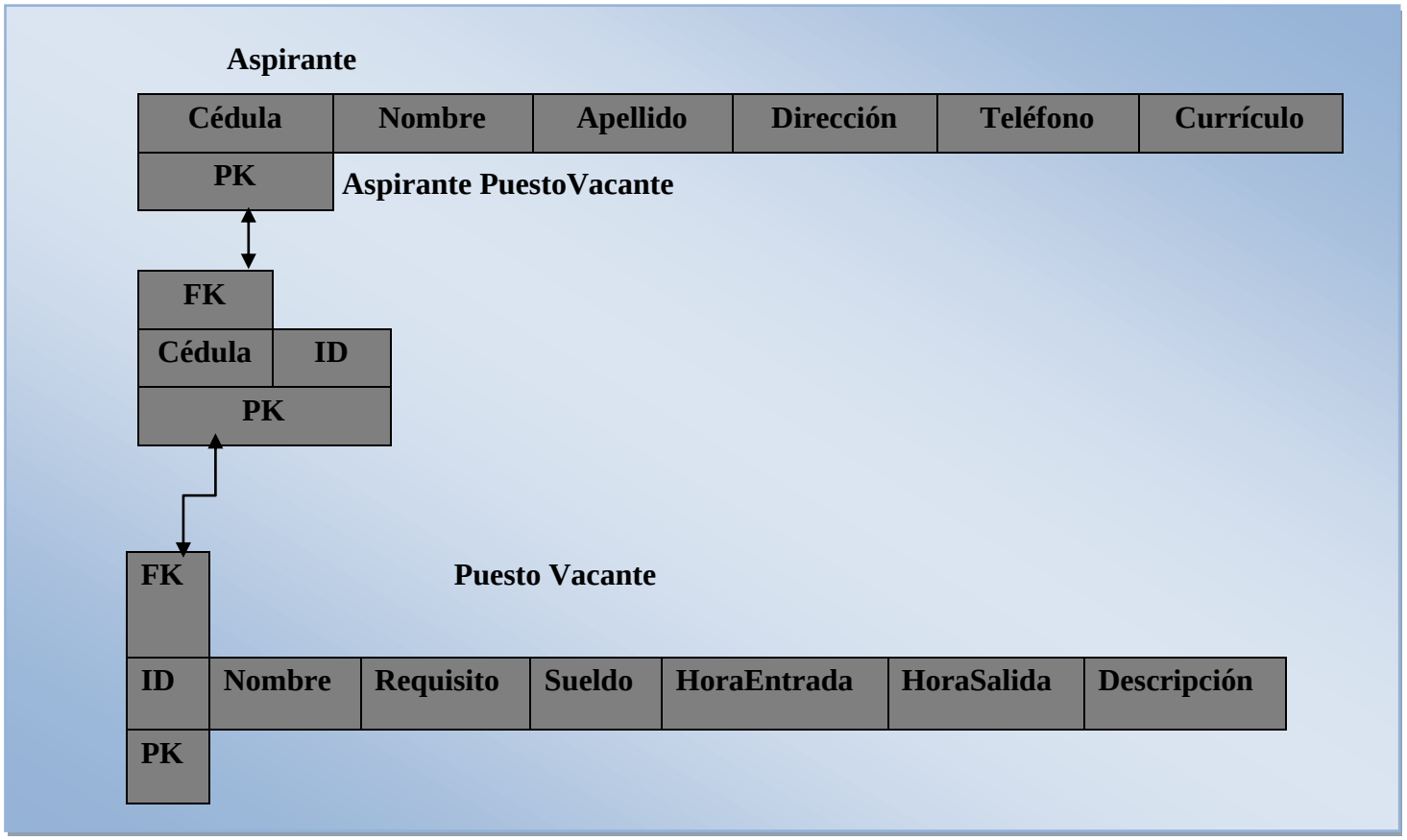


Tabla N° 5.6: Tablas para la Asociación n: n

Conversión de Relaciones de Agregación

La agregación corresponde directamente a una llave externa en la tabla dependiente con acciones de actualización y borrado. Por ejemplo cuando se borra un reglón en la tabla dueña, debe borrarse los renglones asociados a su llave primaria en todas las tablas dependientes, esto es un borrado en cascada.

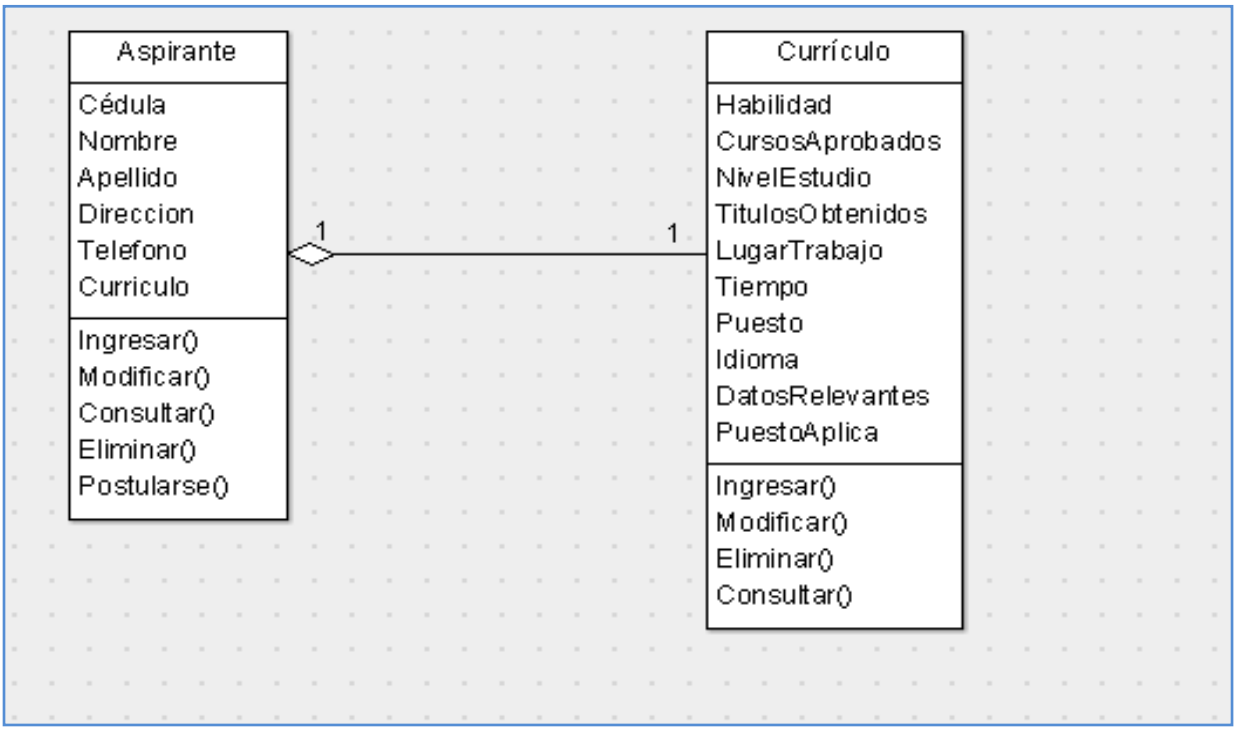


Figura N° 5.41: Relaciones de Agregación entre Clases

Todo aspirante tiene una hoja de vida y la información de éste se encuentra en las tablas currículo. Se creara la tabla para la información de la hoja de vida debido a que en una base de datos no puede haber columnas con atributos no-atómicos.

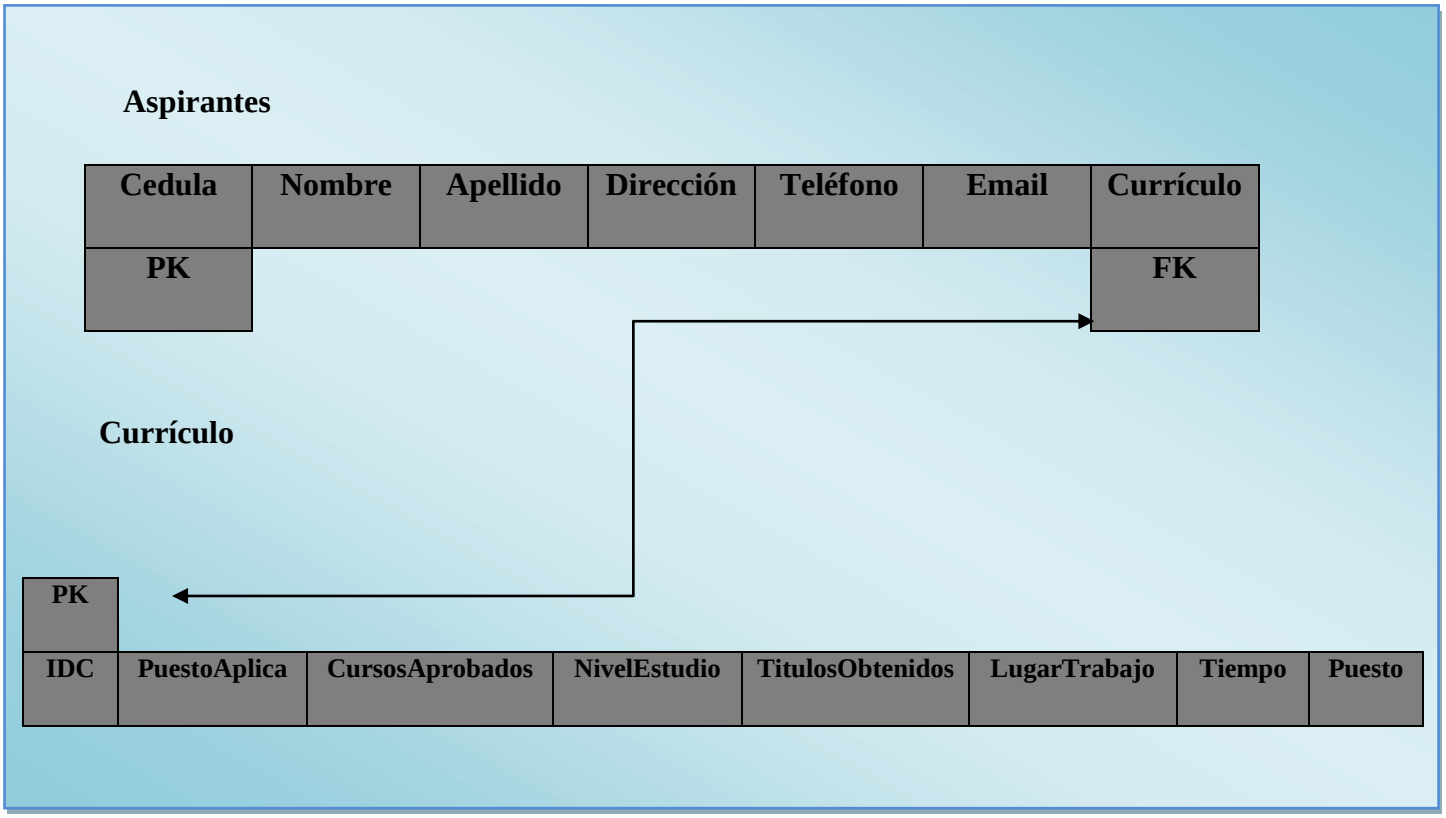


Tabla N° 5.7: Tablas para la relación de Agregación entre Clases

Se agregó a la tabla currículo un identificador para facilitar la manipulación de las llaves. De esta manera la tabla aspirante tiene un identificador de currículo como llave externa que debe corresponder con el identificador de algún currículo de la tabla de ellos.

Conversión de Relaciones de Generalización

Se crea una tabla para cada clase en la jerarquía, incluyendo las clases abstractas, después se crean las llaves externas para cada relación de generalización. Si la llave primaria de la superclase consta de varias columnas, se deben crear esas mismas columnas en cada subclase.

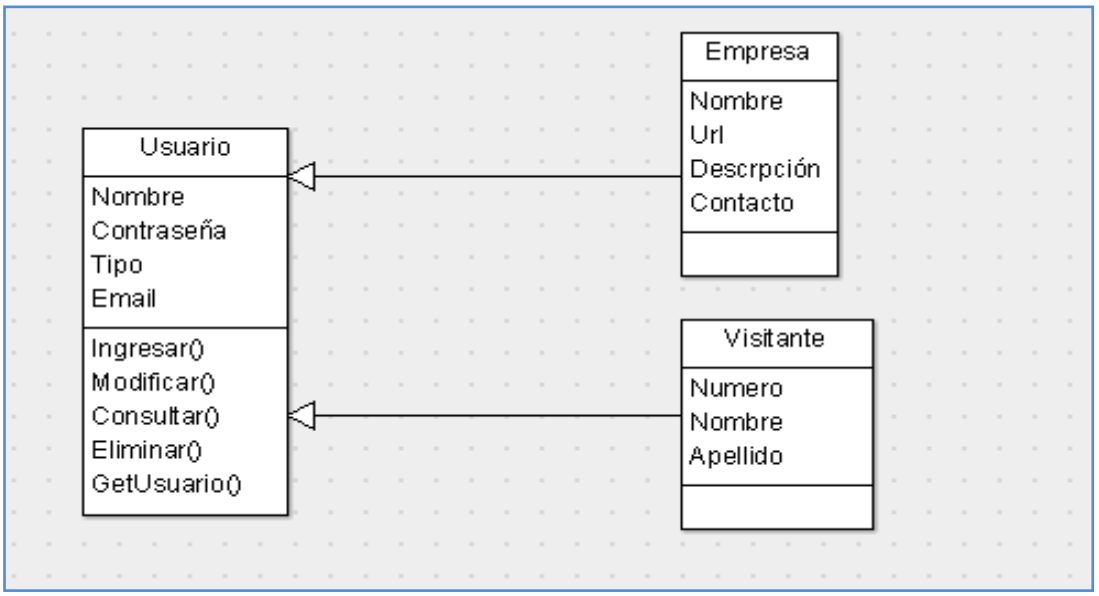


Figura N° 5.42: Jerarquía de Clases

La tabla usuario tiene como llave primaria un identificador para cada usuario, éste se utiliza como llave externa en cada una de las tablas que representa a casa una de las subclases.

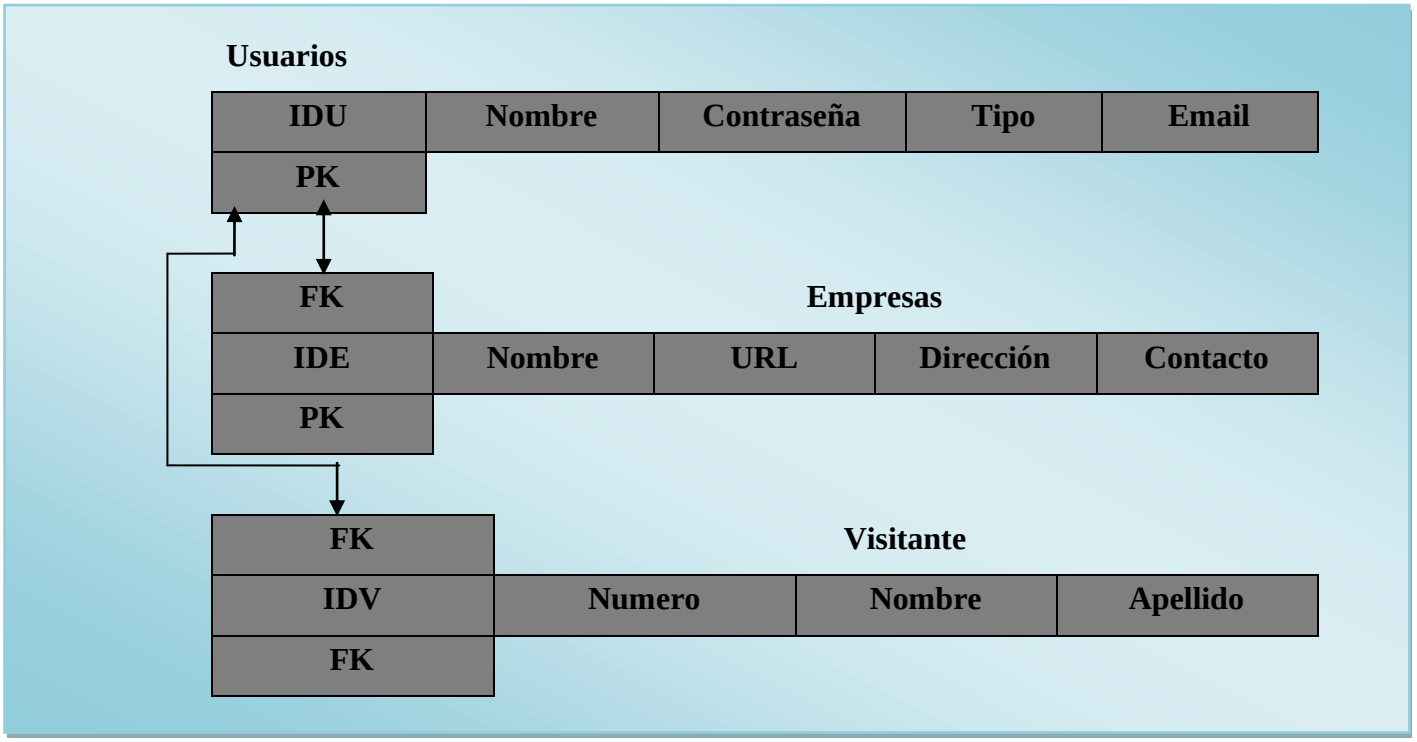


Tabla N° 5.8: Tablas para una Jerarquía de Clases

Lo importante en la conversión del modelo de datos UML a la esquena relacional es ser muy sistemático en el método.

5.5 Construcción

5.5.1 Diseño detallado de la Clase

En el diseño detallado se completan los diagramas de clases incluyendo los detalles necesarios para su codificación. El nivel al que se detallan estos diagramas debe ser suficiente para que se construya su código.

El detalle de las clases de control incluye: nombres, y tipos de todos los atributos, para los métodos se especifican los parámetros con sus tipos y el valor de retorno. Si algún método requiere para su implantación de un algoritmo más complejo, este se especifica en seudocódigo.

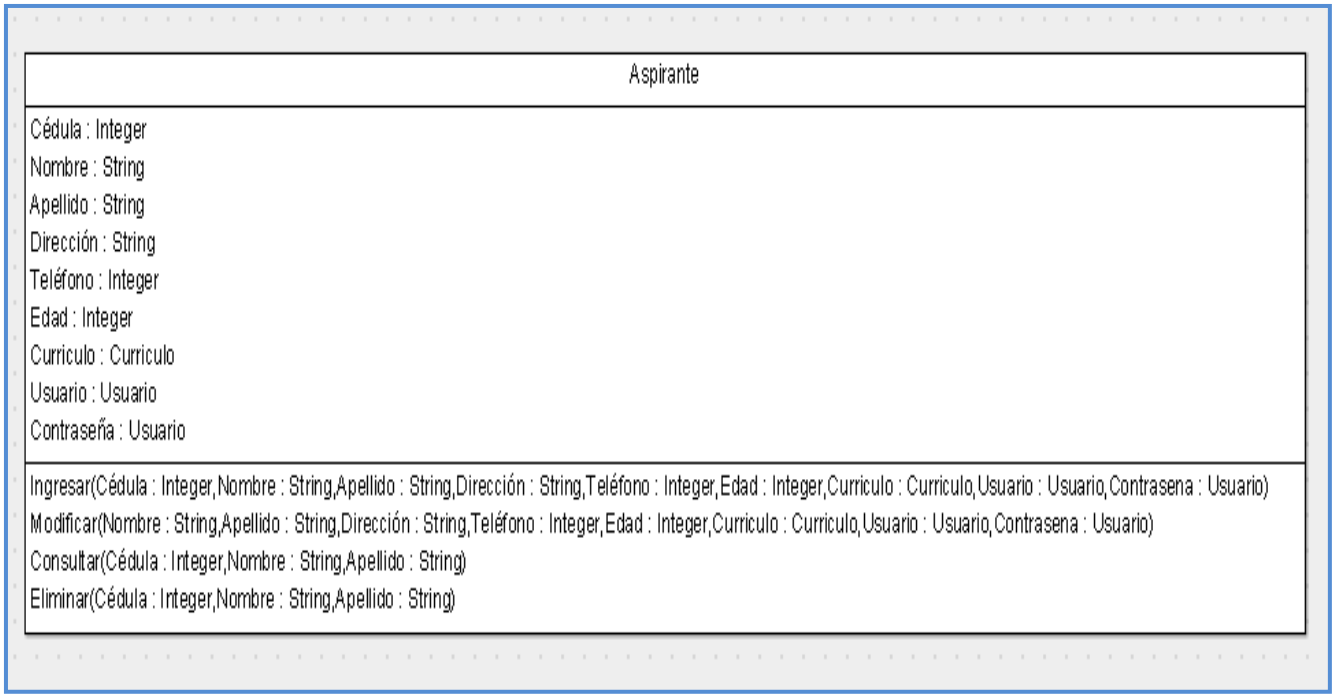


Figura N° 5.43: Ejemplo del Diseño detallado de la Clase

No todas las clases se codifican en el lenguaje de programación, esto quiere decir en una aplicación web, las clases de interfaz pueden implementar como código HTML. Pero a partir de las clases de entidad se

diseña la base de datos, y en los paquetes se incluyen las clases necesarias del ambiente de programación o las bibliotecas disponibles para la construcción de las clases.

Algunas herramientas de los diagramas UML ayudan a completar el diseño detallado a partir de los diagramas de clases, generando un esqueleto de código, esto consiste en detallar los diagramas y completar los esqueletos de código. Una ventaja de estas herramientas es que se mantiene la integridad entre los diagramas y el código.

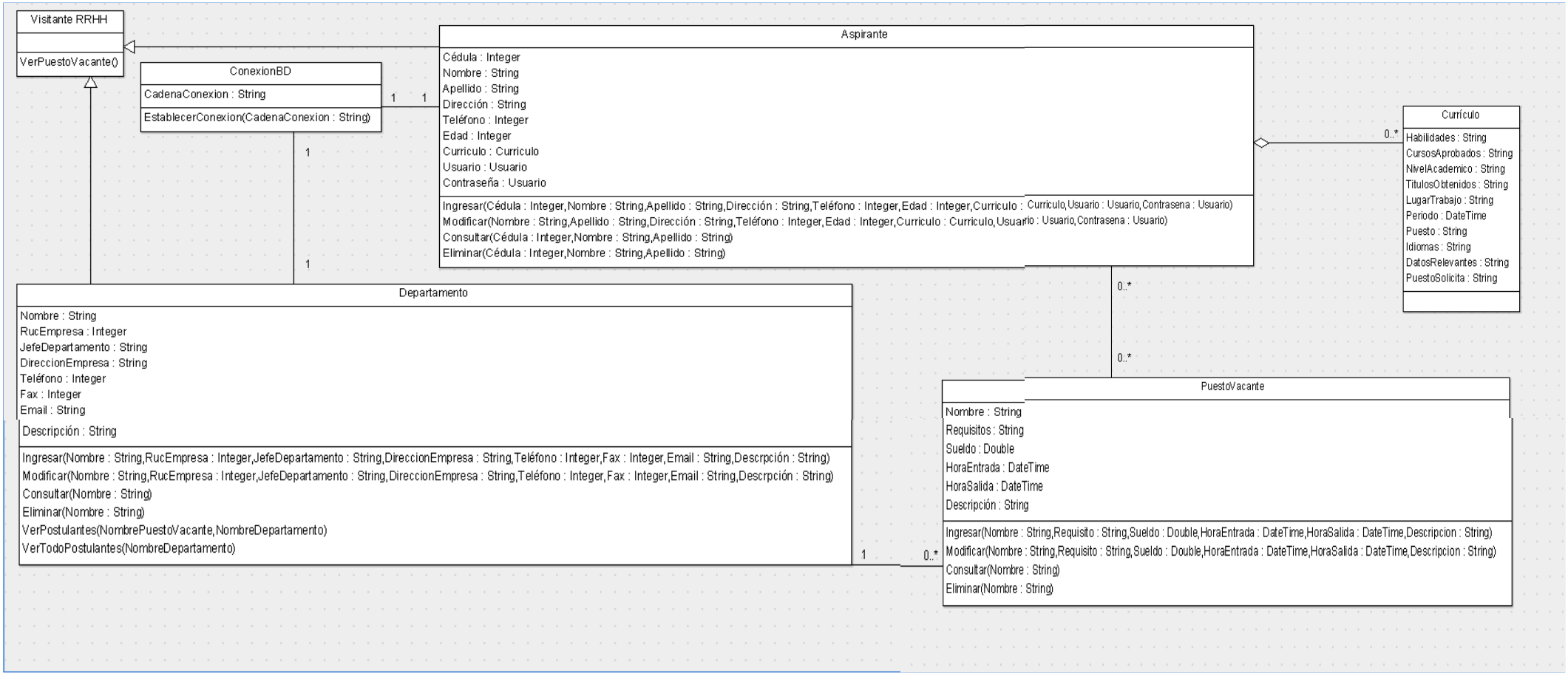


Figura N° 5.44: Diagrama de Clase detallado

5.5.2 Estándares de Codificación

El código de un lenguaje de programación se genera con el fin de que sea compilado y ejecutado por la máquina, pero este código también debe ser comprendido por los seres humanos, por ende debe ser claro y entendible para la persona que lo desarrolla dure el tiempo que dure siempre debe ser claro y no solo para la persona que se cree el software sino para cualquier otro desarrollador que formen o parte del equipo de trabajo del proyecto para su mantenimiento y establecer los estándares de codificación y documentación del sistema.

El código de programación del software es el mismo código de Asp net con clases, métodos, módulos, etc. Por ende no hay ningún problema en el código y en su mantenimiento en un futuro.

Un estándar de codificación es un conjunto de normas para hacer los programas más legibles. Define las convenciones para escribir:

- Los encabezados de de los paquetes o clases
- Para nombrar clases
- Atributos
- Métodos
- Parámetros
- Para estructurar de manera legible las instrucciones de control,
- Maneja de errores

Por ejemplo, un estándar de comentario de encabezado de una clase puede pedir que éste contenga los siguientes elementos

- El propósito de la clase
- Sus autores
- La fecha de creación
- Su versión actual
- Referencias cruzadas a otras clases o archivos

Se recomienda usar estándares de codificación ya existentes para el lenguaje de programación que se va a usar.

El estándar que se empleara para el siguiente proyecto es:

- Las variables se crearan siempre con el tipo de datos que corresponda por ejemplo:
  - Para declarar un capo tipo texto se utilizara la nomenclatura **txt** seguido el nombre del campo (**txtApellido**)
  - Para declarar un campo tipo entero se utilizara la nomenclatura **int** seguido el nombre del campo (**intAño)**
  - Para declara un campo de tipo cadena la nomenclatura seria **string** seguido el nombre del campo (**stringNombre)**
  - Para declarar un texto que será que será utilizado como boolean de utilizara la nomenclatura **lbl** seguidos del nombre del campo (**lblAprueba**)
  - Para declarar los botones se utilizara la nomenclatura **btn** seguido en nombre de botón (**btnGuardar**).

5.5.3 Revisión de Código y Programación entre Pares

El código se genera a partir del diseño detallado, proponer leer el código antes de su compilación para eliminar defectos de forma temprana es poco practicada por las prisas de compilar. Pero al compilador se le escapan los defectos de la lógica de programación, la lectura detenida del código por una persona antes de compilar puede detectar esa clase de defectos y corregirlos de manera oportuna.

Una alternativa para generar el código y leerlo a la vez por dos personas, es la programación entre pares, la idea principal es que el código se escribe entre dos personas sentadas frente de una sola máquina, una persona tiene el control del teclado y del ratón y es el que esta codificando, la otra persona lee lo que se va escribiendo, analiza el código, revisa la lógica y la sintaxis y comenta posible alternativas y errores de su pareja.

Con esta práctica se hacen revisiones constantes al código debido que una persona desarrolla el código y la otra analiza el desarrollo, aunque con esto la programación avanza más lento, la practica ha demostrado que esto no necesariamente es verdad y que el código que se obtiene está más libre de errores lo que hace que tenga una mejor calidad.

Para el proyecto solo existirá una persona para el desarrollo y el análisis de la programación en el desarrollo del proyecto pero habrá un tutor el encargado de revisar el software según el modulo que se vaya avanzando.

Clase Aspirante

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Conectividad;
using System.Data;
using System.Data.SqlClient;

namespace RegladeNegociosGCA
{
  public class Aspirantes
   {
      #region Zona de Datos
      private ConexionesSQLServer conexionData = new
ConexionesSQLServer();
      private SqlConnection ConexionGCA = new SqlConnection();
      #endregion

      #region Zona de Medotos

      #region Zona de Ingreso de Datos
```

```
public void InsertAspirante(string cedula, string nombres, string
apellidos, string cedulaMilitar, string Nacionalidad,
DateTime fechaNacimiento)
                {
                  try
                  {
               ConexionGCA = conexionData.SqlConexion();
               SqlCommand Insert = new SqlCommand();
               Insert.CommandText = procedimientoAlmacenado;
               Insert.CommandType = CommandType.StoredProcedure;
               Insert.Connection = ConexionGCA;
 SqlParameter parametro1 = new SqlParameter("@cedula",
 SqlDbType.NVarChar, 10);
 SqlParameter parametro2 = new SqlParameter("@nombres",
 SqlDbType.NVarChar, 50);
SqlParameter parametro3 = new SqlParameter("@apellidos",
SqlDbType.NVarChar,50);
SqlParameter parametro4 = new SqlParameter("@Cedula_Militar",
SqlDbType.NVarChar, 10);
SqlParameter parametro5 = new SqlParameter("@nacionalidad",
SqlDbType.NVarChar, 30);
SqlParameter parametro6 = new SqlParameter("@Fecha_Nacimiento",
SqlDbType.Date);
               Insert.Parameters.Add(parametro1);
               Insert.Parameters.Add(parametro2);
               Insert.Parameters.Add(parametro3);
               Insert.Parameters.Add(parametro4);
               Insert.Parameters.Add(parametro5);
               Insert.Parameters.Add(parametro6);

               Insert.Parameters["@Cedula"].Value = cedula;
               Insert.Parameters["@Nombres"].Value = nombres;
               Insert.Parameters["@Apellidos"].Value = apellidos;
```

```
            Insert.Parameters["@Cedula_Militar"].Value = cedulaMilitar;
            Insert.Parameters["@Nacionalidad"].Value = Nacionalidad;
            Insert.Parameters["@Fecha_Nacimiento"].Value =
fechaNacimiento;

            ConexionGCA.Open();
            Insert.ExecuteNonQuery();
            ConexionGCA.Close();
            ConexionGCA.Dispose();
        }
        catch (SqlException error)
        {
            throw error;
        }
        catch (Exception error)
        {
            throw error;
        }
        finally
        {
            ConexionGCA.Dispose();
            ConexionGCA.Close();
        }
    }
    #endregion

    #region Zona de Actualizacion de Datos
public void upDateAspirante(string cedula, string nombres, string
apellidos, string cedulaMilitar, string Nacionalidad,
    DateTime fechaNacimiento procedimientoAlmacenado)
    {
        try
        {
```

```
            ConexionGCA = conexionData.SqlConexion();
SqlCommand cmdUpdate = new
SqlCommand(procedimientoAlmacenado,     ConexionGCA);
            cmdUpdate.CommandType = CommandType.StoredProcedure;

             cmdUpdate.Parameters.Add("@cedula",
SqlDbType.NVarChar, 10);
             cmdUpdate.Parameters.Add("@nombres",
SqlDbType.NVarChar, 50);
             cmdUpdate.Parameters.Add("@apellidos",
SqlDbType.NVarChar, 50);
             cmdUpdate.Parameters.Add("@Cedula_Militar",
SqlDbType.NVarChar, 10);
             cmdUpdate.Parameters.Add("@nacionalidad",
SqlDbType.NVarChar, 30);
             cmdUpdate.Parameters.Add("@Fecha_Nacimiento",
SqlDbType.Date);

             cmdUpdate.Parameters["@Cedula"].Value = cedula;
             cmdUpdate.Parameters["@Nombres"].Value = nombres;
             cmdUpdate.Parameters["@Apellidos"].Value = apellidos;
             cmdUpdate.Parameters["@Cedula_Militar"].Value =
cedulaMilitar;
             cmdUpdate.Parameters["@Nacionalidad"].Value =
Nacionalidad;
             cmdUpdate.Parameters["@Fecha_Nacimiento"].Value =
fechaNacimiento;

             ConexionGCA.Open();
             cmdUpdate.ExecuteNonQuery();
             ConexionGCA.Close();
             ConexionGCA.Dispose();
            }
```

```
            catch (SqlException error)
            {
                throw error;
            }
            catch (Exception error)
            {
                throw error;
            }
            finally
            {
                ConexionGCA.Dispose();
                ConexionGCA.Close();
            }
        }
        #endregion

        #region Zona de Borrado de Datos
        public void DeleteAspirante(string cedula, string
procedimientoAlmacenado)
        {
            try
            {
             ConexionGCA = conexionData.SqlConexion();
SqlCommand cmdDelete = new
SqlCommand(procedimientoAlmacenado, ConexionGCA);
                cmdDelete.CommandType =
CommandType.StoredProcedure;
                cmdDelete.Parameters.Add("@Cedula",
SqlDbType.NVarChar, 10);
                cmdDelete.Parameters["@cedula"].Value = cedula;
                ConexionGCA.Open();
                cmdDelete.ExecuteNonQuery();
                ConexionGCA.Dispose();
```

```
            ConexionGCA.Close();
        }
        catch (SqlException error)
        {
            throw error;
        }
        catch (Exception error)
        {
            throw error;
        }
        finally
        {
            ConexionGCA.Dispose();
            ConexionGCA.Close();
        }
    }
    #endregion
    #endregion
  }
}
```

En la interfaz

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;

//DLL de Usuario
```

```
using RegladeNegociosGCA;
using ReglasConsultasGCA;
using ComponestesWindows;
using TratamientoImagen;
using Excepciones;

namespace GCA.Administrador
{
   public partial class Aspirante : System.Web.UI.Page
   {
      #region Zona de Datos
      private DataView ConsultaDatos = new DataView();
      private ArreglodeBytesDeImagenes Imagen = new
ArreglodeBytesDeImagenes();
      private Aspirantes aspirante = new Aspirantes();
      public static string valor;
      private static string valordevuelto;
      private int suma = 0;
      byte[] fotoEnviar;
      string sSavePath;
      #endregion

      #region Zona de Metodos

      #region Zona de Habilitar Controles
      public void HabilitarControles()
      {
txtCedula.Enabled = true;
         txtNombres.Enabled = true;
         TxtApellidos.Enabled = true;
         txtCedulaMmilitar.Enabled = true;
         txtNacionalidad.Enabled = true;
         txtFechadeNacimiento.Enabled = true;
```

```
}
#endregion

#region Zona de Limpieza de los Controles
public void LimpiarComtroles()
{
   txtCedula.Text = "";
   txtNombres.Text = "";
   TxtApellidos.Text = "";
   txtCedulaMmilitar.Text = "";
   txtNacionalidad.Text = "";
   txtFechadeNacimiento.Text = "";
}
#endregion

#region Zona de Desavilitar Controles
public void DesavilitarControles()
{
   txtCelular.Enabled = false;
   txtNombres.Enabled = false;
   TxtApellidos.Enabled = false;
   txtCedulaMmilitar.Enabled = false;
   txtNacionalidad.Enabled = false;
   txtFechadeNacimiento.Enabled = false;
}
#endregion

protected void Page_Load(object sender, EventArgs e)
{
   try
   {
      if (!Page.IsPostBack)
      {
```

```
                #region Zona de Cargado de Pais
                ConsultaPais ConsultaPais = new ConsultaPais();
                ConsultaDatos = ConsultaPais.ConsultaPaisGeneral();
                int cual = ConsultaDatos.Count;
                cbrPais.DataSource = ConsultaDatos;
                cbrPais.DataTextField = "Nombre";
                cbrPais.DataValueField = "Nombre";
                cbrPais.DataBind();
this.cbrPais.Items.Insert(0, new ListItem("Seleccione un Pais"));
                #endregion

                #region Zona de Cargado de Tipo de Sexo

                ConsultaTipoSexo consultaSexo = new
ConsultaTipoSexo();
                ConsultaDatos = consultaSexo.ConsultaSexoGeneral();
                cbSexo.DataSource = ConsultaDatos;
                cbSexo.DataTextField = "Sexo";
                cbSexo.DataValueField = "Sexo";
                cbSexo.DataBind();
this.cbSexo.Items.Insert(0, new ListItem("Seleccione el Tipo de
Sexo"));
                #endregion
            }
        }

        catch (SqlException error)
        {
            ExcepcionesWebBaseDeDatosCliente.Web(error);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
        catch (Exception errores)
```

```
        {
            ExcepcionesWebBaseDeDatosCliente.Web(errores);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
    }

    protected void btnINuevo_Click(object sender,
ImageClickEventArgs e)
    {
        if (suma == 0)
        {
            HabilitarControles();
            valor = "Insertar";
            OpcionHacer enviar = new OpcionHacer();
            enviar.AsignarValorBandera(valor);
            suma = 1;
            txtCedula.Focus();
            txtNombreDiscapacidad.Text = "Ninguna";
        }
    }

    protected void btnIGuardar_Click1(object sender,
ImageClickEventArgs e)
    {
        try
        {
            OpcionHacer devolver = new OpcionHacer();
            valordevuelto = devolver.DevolverBanderaEmpresa();

            if (valordevuelto == "Insertar")
            {
```

```
fotoEnviar = Imagen.RegresoInterfazArregloBytes();
aspirante.InsertAspirante(txtCedula.Text.Trim(),
txtNombres.Text.Trim(), TxtApellidos.Text.Trim(),
txtCedulaMmilitar.Text.Trim(),
txtNacionalidad.Text.Trim(),
Convert.ToDateTime(txtFechadeNacimiento.Text.Trim())
"sp_InsertAspirantes");
                suma = 0;
                LimpiarComtroles();
                DesavilitarControles();
            }
            else
            {
                if (valordevuelto == "Actualizar")
                {
fotoEnviar = Imagen.RegresoInterfazArregloBytes();
aspirante.upDateAspirante(txtCedula.Text.Trim(),
txtNombres.Text.Trim(), TxtApellidos.Text.Trim(),
txtCedulaMmilitar.Text.Trim(),
txtNacionalidad.Text.Trim(),
Convert.ToDateTime(txtFechadeNacimiento.Text.Trim())
"sp_UpdateAspirantes");
                suma = 0;
                LimpiarComtroles();
                DesavilitarControles();
                    suma = 0;
                    LimpiarComtroles();
                    DesavilitarControles();
                }
            }
        }
        catch (SqlException error)
        {
```

```
            ExcepcionesWebBaseDeDatosCliente.Web(error);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
        catch (Exception errores)
        {
            ExcepcionesWebBaseDeDatosCliente.Web(errores);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
    }

    protected void btnIActualizar_Click(object sender,
ImageClickEventArgs e)
    {
        try
        {
            if (suma == 0)
            {
                valor = "Actualizar";
                OpcionHacer enviar = new OpcionHacer();
                enviar.AsignarValorBandera(valor);
                txtCedula.Enabled = true;
                txtCedula.Focus();
                suma = 1;
            }
        }
        catch (SqlException error)
        {
            ExcepcionesWebBaseDeDatosCliente.Web(error);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
```

```
            catch (Exception errores)
            {
                ExcepcionesWebBaseDeDatosCliente.Web(errores);
                lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
            }
        }

        protected void cbrProvincia_TextChanged(object sender,
EventArgs e)
        {
            try
            {
             ConsultaCiudades consultaCiudades = new
ConsultaCiudades();
ConsultaDatos =
consultaCiudades.ConsultaCiudadesporProvinciaspor(cbrProvincia.Text
);
                int sa = ConsultaDatos.Count;
                cbrCiudad.DataSource = ConsultaDatos;
                cbrCiudad.DataTextField = "Nombre";
                cbrCiudad.DataValueField = "Nombre";
                cbrCiudad.DataBind();
this.cbrCiudad.Items.Insert(0, new ListItem("Seleccione un Cuidad"));
            }
            catch (SqlException error)
            {
                ExcepcionesWebBaseDeDatosCliente.Web(error);
                lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
            }
            catch (Exception errores)
            {
```

```
            ExcepcionesWebBaseDeDatosCliente.Web(errores);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
    }

    protected void btnSubirImagen_Click(object sender, EventArgs e)
    {
        if (Page.IsValid)
        {
            sSavePath = "~/Fotos/";
            string carpetaVirtual = this.Server.MapPath(sSavePath);
        HttpPostedile myFile = upFoto.PostedFile;
         string nombre = upFoto.FileName;
string fileExt =
System.IO.Path.GetExtension(myFile.FileName).ToLower();
if (fileExt == ".jpg" || fileExt == ".png" || fileExt == ".gif" || fileExt ==
".mp" || fileExt == ".jpeg")
          {
              int len = upFoto.PostedFile.ContentLength;
              Imagen.ImageToByteArreglo(len);
               string imgContentType = upFoto.PostedFile.ContentType;
               fotoEnviar = new byte[len];
               upFoto.PostedFile.InputStream.Read(fotoEnviar, 0, len);
               upFoto.SaveAs(carpetaVirtual + upFoto.FileName);
               ImFoto.ImageUrl = sSavePath + upFoto.FileName;
            }
        }
    }
    protected void txtCedula_TextChanged(object sender, EventArgs
e)
    {
        try
```

```
        {
            if (Convert.ToInt32(txtCedula.Text.Length) == 10)
            {
                OpcionHacer devolver = new OpcionHacer();
                valordevuelto = devolver.DevolverBanderaEmpresa();
                if (valordevuelto == "Insertar")
                {
                    txtNombres.Focus();
                }
                else
                {
                    if (valordevuelto == "Actualizar")
                    {
ConsultaAspirante consultaAspirantes = new ConsultaAspirante();
ConsultaDatos =
consultaAspirantes.ConsultaAspirantesPorCondicion(txtCedula.Text.Tr
im());
                        LimpiarComtroles();
txtCedula.Text = ConsultaDatos.Table.Rows[0][0].ToString();
txtNombres.Text = ConsultaDatos.Table.Rows[0][1].ToString();
TxtApellidos.Text = ConsultaDatos.Table.Rows[0][2].ToString();
txtCedulaMmilitar.Text = ConsultaDatos.Table.Rows[0][3].ToString();
txtNacionalidad.Text = ConsultaDatos.Table.Rows[0][4].ToString();
                    txtCedula.Focus();
                }
                catch (SqlException error)
                {
                    ExcepcionesWebBaseDeDatosCliente.Web(error);
                    lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
                }
                catch (Exception error)
                {
```

```
            ExcepcionesWebBaseDeDatosCliente.Web(error);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
        txtCedula.Focus();
    }

    protected void btnIEliminar_Click(object sender,
ImageClickEventArgs e)
    {
        try
        {
aspirante.DeleteAspirante(txtCedula.Text.Trim(),
"sp_DeleteAspirantes");
        }
        catch (SqlException error)
        {
            ExcepcionesWebBaseDeDatosCliente.Web(error);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
        catch (Exception error)
        {
            ExcepcionesWebBaseDeDatosCliente.Web(error);
            lbError.Text =
ExcepcionesWebBaseDeDatosCliente.Mensaje;
        }
    }
  }
}
```

5.5.4 Pruebas Unitarias

Las pruebas unitarias consisten en probar la lógica de código que se va terminando en cada modulo. En la orientación a objetos la lógica de pruebas es la clase por ende se debe asegurar la calidad de las clases probándoles detenidamente.

Las clases se prueban a través de la invocación de sus métodos, los métodos de una clase se definen casos de prueba, un caso de prueba de un método consiste en definir un conjunto representativo de valores para los parámetros del método y el valor esperado para ese conjunto, para cada método se puede definir uno o más casos de prueba según la complejidad del método.

Para realizar pruebas unitarias cada desarrollador define su plan de las pruebas unitarias para las clases que está constituyendo, las pruebas se le han realizado con los datos de la clases requeridas con datos reales, para ver el resultado obtenido.

El plan de pruebas unitarias se define:

- Las clases que se probaran: aspirante, departamento, puesto, currículo
- Métodos a probar: ingresar, modificar, eliminar, consultar
- Los casos de prueba con los conjuntos de valores para los parámetro para cada método
- El resultado esperado de cada caso de prueba.

Se ha implantado el sistema para realizar sus respectivas pruebas, se lo ha instalado en el departamento correspondiente, realizando el ingreso de los aspirantes obteniendo como resultado los datos esperados, el sistema está funcionando en su totalidad. Las pruebas obtenidas son positivas, se puede observar que el ahorro de tiempo es bastante además evita que existan datos repetidos.

Clase	Método	Caso de Prueba	Resultado Esperado
UsuarioRegistrado	setNombre(nombre)	miNombre	Guarda el nombre
UsuarioRegistrado	SetDireccion(calle)	miCalle	Guarda los campos de la dirección

Tabla N° 5.9: Ejemplo de Plan de Pruebas Unitarias

Para realizar la prueba unitaria de una clase, se crea un objeto de esa clase y se invoca al método a probar con los parámetros definidos en sus casos de prueba. Si el resultado obtenido coincide con el esperado el método pasa esa prueba. Si no coincide se revisa el código del método para encontrar la causa del defecto y corregirlo. Se vuelve aplicar el mismo case de prueba hasta que pase. Cuando una clase pasa todas las pruebas definidas en el plan de unitarias, es aceptada.

Técnicas para definir los Casos de Pruebas

Para definir los casos de prueba se usan dos técnicas:

- Caja blanca se toma en cuenta la estructura del código de un método y se busca que durante las pruebas cada instrucción se ejecute al menos una vez.
- Caja negra consideran al código del método como un todo oculto, verificando que para cada conjunto de valores de los parámetros se obtenga el resultado esperado.

5.5.4.1 Pruebas de Caja Blanca

También se las conocen como pruebas estructurales, revisa la estructura lógica de la unidad a probar. Una técnica que se usa para saber cuántos casos de prueba se deben definir es la complejidad ciclo matica que se define como el número de condiciones encontradas en el código más 1. Una condición es un punto de decisión, como por ejemplo if, while, for, etc. Para el proyecto utilizaremos si una tabla de la base y el valor devuelvo

es de ingresar o actualizar

- El valor devuelvo es actualizar (la condición if es falsa)
- El valor devuelto es insertar (la condición if es verdadera)
- Las datos ingresados ya existen y no puede volver a grabad debe actualizar (la condición if es falsa)
- No se puede actualizar datos que no se han ingresado (la condición if es verdadera)

```
if (valordevuelto == "Insertar")
{       txtNombres.Focus();      }
else
{
        if (valordevuelto == "Actualizar")        {
                    ConsultaAspirante consultaAspirantes = new
                    ConsultaAspirante();
                    ConsultaDatos =
                    consultaAspirantes.ConsultaAspirantesPorCondicion
                    (txtCedula.Text.Trim ());
                    LimpiarComtroles();
                    txtCedula.Text =
                    ConsultaDatos.Table.Rows[0][0].ToString();
                    txtNombres.Text =
                    ConsultaDatos.Table.Rows[0][1].ToString();
                    TxtApellidos.Text =
                    ConsultaDatos.Table.Rows[0][2].ToString();
                    txtCedulaMmilitar.Text =
                    ConsultaDatos.Table.Rows[0][3].ToString();
                    txtNacionalidad.Text =
                    ConsultaDatos.Table.Rows[0][4].ToString();
                    txtCedula.Focus();
          }  }
```

Figura N° 5.45: Ejemplo de Seudocódigo

El plan de prueba para este método es:

#	Casos de prueba	Resultados esperados
1	No hay datos que actualizar (la condición del if es falsa)	Falso
2	Se ha ingresado datos (la condición del if es verdadera)	Verdadero

Tabla N° 5. 10: Ejemplo de Caja Blanca

5.5.4.2 Pruebas de Caja Negra

También se las conoce como prueba funcional, revisa que la unidad cumpla con la funcionalidad esperada sin considerar el detalle del código. Para definir los casos de prueba, se establecen los posibles resultados esperados de la unidad y se identifican los conjuntos de valores de los parámetros, para que se generen estos resultados.

Una técnica para diseñar los casos de prueba de caja negra son las tablas de decisión, ayudan a describir combinaciones de datos de entrada que generan diferentes salidas.

Una tabla de decisión tiene 2 secciones:

- **Condiciones de parámetros de entrada:** En la parte superior de la tabla se define la lista de condiciones de los parámetros de entrada y sus posibles combinaciones de valores verdaderos y falsos.
- **Resultados esperados:** Se marca con la X el resultado esperado de cada posible combinación de valores de parámetros.

Condiciones de parámetros de entrada			
Insertar es guardar datos	verdadero	verdadero	falso
Modificar es actualizar información	verdadero	falso	falso
Resultados esperado			
El método ingresar sea verdadero	X		
El método actualizar sea falso		X	X

Tabla N° 5.11: Ejemplo de Tabla de Decisión

El plan de pruebas unitarias de caja negra, creado a partir de la tabla de decisión, es una tabla que tiene casos de prueba la combinación de condiciones de valores verdadero y falso para los parámetros de entrada y los resultados esperados.

#	**Casos de prueba**	**Resultados esperados**
1	Se han ingresados los datos y se puede modificar.	Verdadero
2	Se quiere actualizar pero no se ha ingresado.	Falso
3	No se ha ingresado datos y no se puede actualizar.	Falso

Tabla N° 5.12: Plan de prueba Caja Negra

Las tablas de decisión es una técnica que se puede emplear durante el desarrollo del software, en diferentes etapas, cuando hay combinaciones de condiciones para ayudar en la toma de decisiones.

Efectuar las Pruebas

Para efectuar las pruebas unitarias se recomienda:

- Ejecutar los casos de prueba.
- Se agregan otros casos de prueba que permitan revisar la estructura, según la técnica de caja blanca.

Cuando se prueba una operación o método que requieren de otro que no están disponibles, para esto es necesario simular el comportamiento se declaran métodos sustitutos, estos métodos no hacen nada excepto regresar el valor esperado, por lo que se puede programar una pequeña interfaz por el programador quien simula que se invoco el método y que eventualmente proporciona el resultado esperado.

En el proyecto estos métodos son:

```
catch (SqlException error)
{
   throw error;
}
catch (Exception error)
{
   throw error;
}
```

Se ejecuta para asegurar, liberar la memoria de la variable que contiene la cadena de conectividad y después salta y cierra la conectividad.

```
finally
{
   ConexionGCA.Close();
   ConexionGCA.Dispose();
}
```

Otro tipo de sustituto es un drive que es un programa que inicializa variables no locales y los parámetros para activar a la unidad que se está probando.

5.6 Integración y Prueba del Sistema

5.6.1 Integración al Sistema

El objetivo de la integración es unir todos los componentes construidos y probados para comprobar que funcionen bien juntos. Para hacer la integración se usa como guía el diseño arquitectónico. La integración se planea identificando el orden en que se unirán los paquetes de la arquitectura, a esto se le conoce como integración del sistema.

Estrategias para la Integración del Sistema

Las posibles estrategias son:

- **Estrategias de paquetes:** La idea es hacer las integraciones según los paquetes del diseño. La ventaja es que se integran los componentes de un paquete y posteriormente se unen a otros paquetes ya integrados, siguiendo la arquitectura. Ejemplos: se integran todos los elementos del paquete de interfaz de usuario para un actor.
- **Estrategia para la funcionalidad o caso de uso:** La idea es integrar las tres capas que corresponden a la realización de un caso de uso.

No existe una estrategia adecuada para todos los sistemas, decidir cuál usar es decisión del proyecto y el nivel de diseño efectuado. Para el proyecto se utilizara la estrategia de paquetes debido a que no depende de otros paquetes que se integren, debido a que una clase depende de otra si invoca algunos de sus métodos.

Al integran los componentes se identifican los métodos que se invocan entre clases. Estos métodos deben ejecutarse para comprobar que el paso de parámetros y los resultados devueltos son los adecuados.

5.6.2 Prueba del Sistema

En la prueba del sistema ya integrado, se trata de comprobar que el sistema cumple con todos los requisitos establecidos, tantos las funcionales como los no funcionales. El software ha cumplido todos sus requerimientos que han sido planteados en el inicio del proyecto.

Un producto de software es muy difícil probar al cien por ciento todos los aspectos del sistema, se debe planear a que se dará prioridad.

- Asegurar que cumple con sus funciones
- Evaluar su usabilidad
- Comprobar el sistema bajo condiciones de estrés y medir el desempleo

En el proyecto se aseguro que cumpla con las funciones que se estableció o las metas para las cuales fue desarrollado y luego evaluamos su usabilidad que tan usable es el software y es útil para el departamento de recursos humanos, después se comprobó el sistema que funciona bien bajo una tensión de trabajó muy alta.

Se probaron los siguientes aspectos definidos por los casos de prueba:

- La instalación del sistema
- Sus funcionales según los casos de uso
- La usabilidad del sistema con la participación de usuarios reales
- Se deben incluir pruebas de los otros requerimientos no funcionales como:
    - Rendimiento
    - Robustez (Recuperación a los errores de hardware)
    - Tiempo de respuesta
    - Capacidad de carga (memoria, acceso a internet y a la bases de datos)

Técnicas para las pruebas funcionales del Sistema

Al probar el sistema se debe comprobar que cumplan con los requerimientos y asegurar que todas las funcionalidades están cubiertas.

Una técnica para documentar lo que se quiere documentar son las tablas cruzadas estas contiene en los reglones los requerimientos y en las columnas los componentes del sistema. En el interior de la tabla se pone con una X si es requerimiento y i si está cubierta en el componente la j para indicar en qué componente están cubiertos los requerimientos.

Requerimientos/componente	Componente1	Componente2	Componente3
Req.1	X		X
Req.2		X	
Req.3		X	
Req.4	X		

Tabla N° 5.13: Ejemplo de Tabla Cruzada

De esta manera se tiene una lista de aprobaciones que se están cumpliendo todos los requerimientos, debido a que ningún renglón deberá ir vacio, lo que indica que al menos algún componente del sistema cubre este requerimiento.

Si se revisa la columna de los componentes se puede notar que ninguna columna esta en blanco lo que significa que los componentes cooperan con al menos un requerimiento. En la tabla se documenta que los componentes cubren los requerimientos, lo que es útil para mantener el sistema al hacer cambios a los requerimientos.

Planeación de las Pruebas del Sistema

El plan de pruebas que debe incluir:

- Que se va a probar, el software que se ha desarrollado
- En qué orden se lo va hacer primero con pequeños datos
- El material de prueba esto es, el componente a probar, los datos reales de prueba, el software que se necesita para su correcto funcionamiento
- El resultado esperado para cada dato de prueba
- Responsable de la prueba el desarrollador del software, los participantes en la prueba los digitadores y recursos humanos, la fecha y lugar en que se realizara la prueba, que se hará en la empresa.
- Plantilla para el informe de los resultados obtenidos indicando el número de defectos encontrados.

Efectuadas las pruebas, se corrigen los defectos encontrados. Se vuelven a aplicar pruebas, conocidas como pruebas de regresión estas sirven para detectar defectos al hacer cambios a componentes ya probados. Al iniciar un nuevo ciclo de desarrollo, se generan nuevos componentes o se modifican los que se tenían para incluir nuevas funcionalidades o características. Estos componentes pueden ocasionar fallas en lo que ya funcionaba por efectos laterales o nuevas interacciones.

El sistema ha pasado todas las pruebas y se ha instalado en la matriz para su funcionamiento.

5.6.3 Manuales

Cuando se desarrolla un sistema, siguiendo el ciclo de desarrollo también se genera la documentación del sistema para que sirva de guía en los siguientes ciclos o mantenimientos esto es para los desarrolladores. Sin embargo, generalmente se requieren los manuales para otros involucrados en el sistema, principalmente para los usuarios. Los

manuales deberán estar escritos en el lenguaje del lector a quien está dirigido debe ser entendible, comprensible.

Manual de Usuario

Este manual debe contiene información del sistema que se puede usar para los usuarios. Puede ser un documento electrónico o impreso que describe la forma de uso del software, que está organizado en base a la interfaz de usuario. Este manual debe contener:

- Cómo se entra, debe contener en la interfaz, en cada opción o campo del sistema debe ser útil para resolver problemas a la hora de ejecución

Este manual también contiene información sobre la instalación que pueda ser útil al usuario.

Manual de Operación o Instalación

Debe estar escrito de forma entendible para el momento de la instalación del software, puede estar en formato electrónico o impreso. Su contenido será:

- La versión que está instalado
- Necesidades de hardware indicando las capacidades y velocidades mínimas.
- Necesidades de software con versiones mínimas para su funcionamiento
- Proceso para la instalación
- A quien recurrir en caso de problemas mayores que no se pueda solucionar con el manual

A los manuales también se aplican pruebas para asegurarse que la escritura es clara, precisa, completa y entendible.

Estos manuales han sido entregados a la empresa para su administración correcta.

5.7 Conclusiones y Recomendaciones

Conclusiones:

- Al aplicar las técnicas de seguridad informática, la información podrá ser almacenada de forma segura y confiable para la empresa, se protege contra cualquier daño o acceso indebido.
- Con un servidor de directorio se podrá organizar y administrar mejor los recursos informáticos de la empresa.
- La implementación de políticas de seguridad informática facilita la administración de la red informática de la empresa, reduce errores de los usuarios por manejo inadecuado de los computadores.
- Con la realización de copias de seguridad periódicas de la información, se podrá recuperación la información tal vez no toda pero si la mayoría de la información respaldada.
- Con un correcto manejo de control de asignación de permisos de acceso, se garantiza que los recursos informáticos de la empresa sean manipulados y utilizados por las personas autorizadas.
- Con un servidor de actualizaciones, se consigue que los equipos de la empresa se mantengan correctamente actualizados mejorando su rendimiento, seguridad y productividad.
- Un punto de distribución de software, facilita la tarea del administrador de sistemas, además se consigue que las instalaciones se realicen de manera simultánea en varios equipos y de forma rápida.

Recomendaciones

- Se debe tener en cuenta la utilización de aplicaciones y el acceso a recursos de los usuarios de la red antes de organizarlos en grupos y unidades organizativas.
- Por lo menos se debe tener levantados los servicios de un controlador de dominios de respaldo en caso se suscite algún daño con el principal.
- Considerar el alcance y funcionalidad de las directivas de grupo, antes de habilitar o no alguna opción de configuración.
- Se debe configurar el servidor de actualizaciones para que descargue sólo las actualizaciones que las estaciones de la empresa requieran, evitando que colapsen los medios de almacenamiento y minimizando el acceso a internet.
- Identificar la información importante y vulnerable para la empresa, a fin de realizar las copias de seguridad, además se debe probar que la restauración se realice correctamente simulando pérdida o daño de esto.
- Elaborar un plan de contingencia que proteja los recursos informáticos de la empresa, en caso de presentarse algún tipo de desastre.
- Instruir a los usuarios en el uso adecuado de los computadores, para que se complemente con la configuración del ambiente trabajo mediante directivas de grupo.
- Capacitar al personal para que puedan resolver problemas comunes y leves que se presenten durante sus labores diarias, sin requerir la presencia del personal del área de Sistemas.

5.8 Bibliografía

5.8.1 Libros

- Charte, F. (2003). Manual Avanzado Windows Server2003. Grupo Anaya SA. Primera edición. Madrid-España.
- Staneck, W. (2003). Manual Del Administrador Windows Server2003. McGraw Hill. Primera edición. Madrid-España.
- Simmons, C.(2000). Windows 2000 Server Configuration. Prentice Hall.
- Primera edición. Madrid-España.
- Sallings, W. (2001). Sistemas Operativos. Prentice Hall. Primera edición. Madrid-España.
- Karanjit, S. (2001). Windows 2000 TCP/IP. Prentice Hall. Primera edición. Madrid-España.

5.8.2 Referencias Web

- http://es.wikipedia.org
- http://hispasec.com
- http://www.monografías.com
- www.mailxmail.com/curso/informatica/mantenerordenador/capitulo44.htm
- www.juntadeandalucia.es/empleo/orienta/menuInicioguia/glosario.asp
- feda.es/leyenda/glosario.html
- http://www.slideshare.net/jose_rob/diseo-de-la-arquitectura-del-software
- http://blogparaentregartareas.blogspot.com/2011/04/ingenieria-de-software-manual-de.html
- http://ayudaeninternet.blogspot.com/2007/03/introduccion-internet-y-las- redes.html

5.9 Anexos

Encuesta Quito Motors

Nombre: _____

Puesto: _____

1. ¿Cuál opina usted que sería el un problema fundamental por lo que la empresa está pasando?

   _____

   _____

   _____

2. ¿Cuál cree usted que es la principal causa del origen del problema?

   _____

   _____

   _____

3. ¿Qué opinión podría ofrecer para encontrar una solución optima al problema?

   _____

   _____

   _____

4. ¿Estaría usted de acuerdo en la utilización de recursos informáticos para la solución como puede ser un sistema de automatización?

   _____

   _____

Anexo N° 1: Encuesta

Manual De Usuario

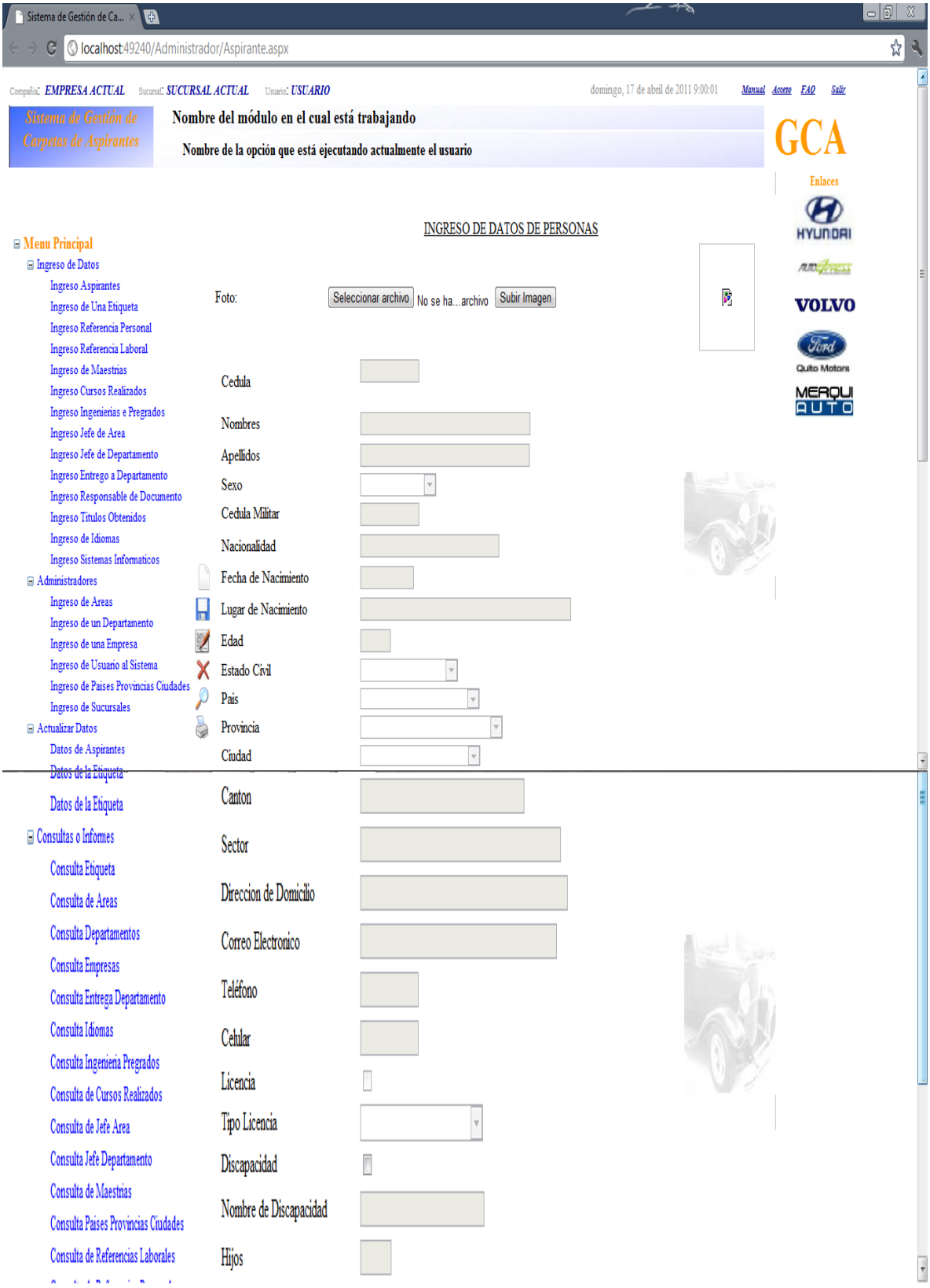
Un manual de usuario es una guía que ayudara a entender y comprender de mejor manera el funcionamiento del software CGA (Desarrollo de un sistema de información de gestión de las carpetas de aspirantes a la Quito Motors.

Es un documento de comunicación técnica que busca brindar asistencia de ayuda a los empleados del Departamento de Recursos Humanos que usan el sistema o software.

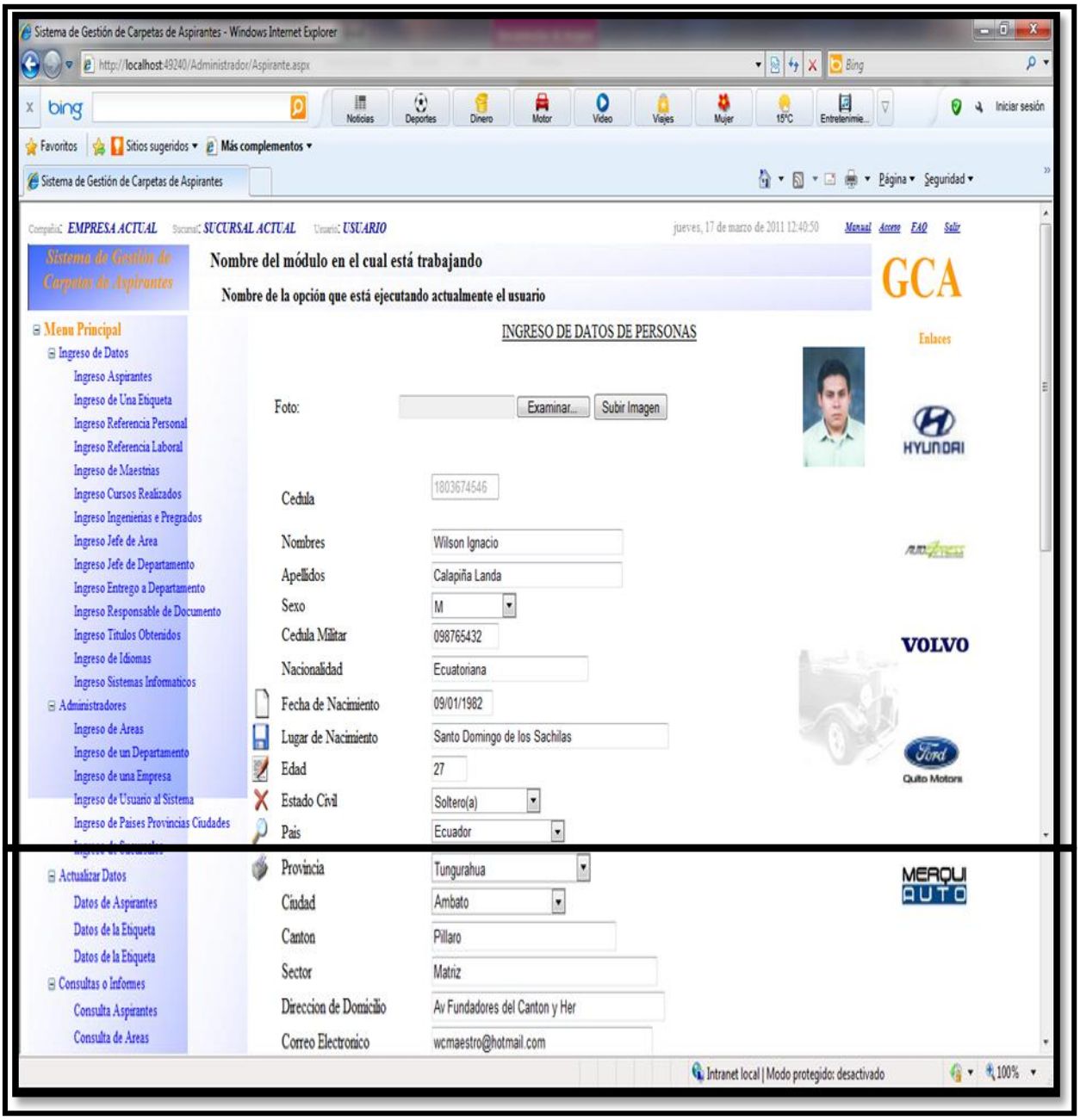
Pasos del manual del usuario:

1. **Portada**: El documento está elaborado y dirigido para el personal de la empresa que tenga que trabajar con el sistema, que tengan dudas en el funcionamiento puede utilizar el manual de ayuda.
2. **Introducción**: El uso del documento describe paso a paso como puede acceder y navegar en el sistema, sirve para consultar algún problema que se puede presentar como puede ser los pasos para realizar un reporte, etc.
3. **Análisis y requerimientos del sistema:** Para poder utilizar el sistema solo es necesario una PC Pentium IV que tenga acceso a internet para poder trabajar con el sistema y poder navegar desde cualquier lugar.
4. **Explicación del funcionamiento:** Para poder acceder al sistema debemos ingresar a internet a la dirección http://

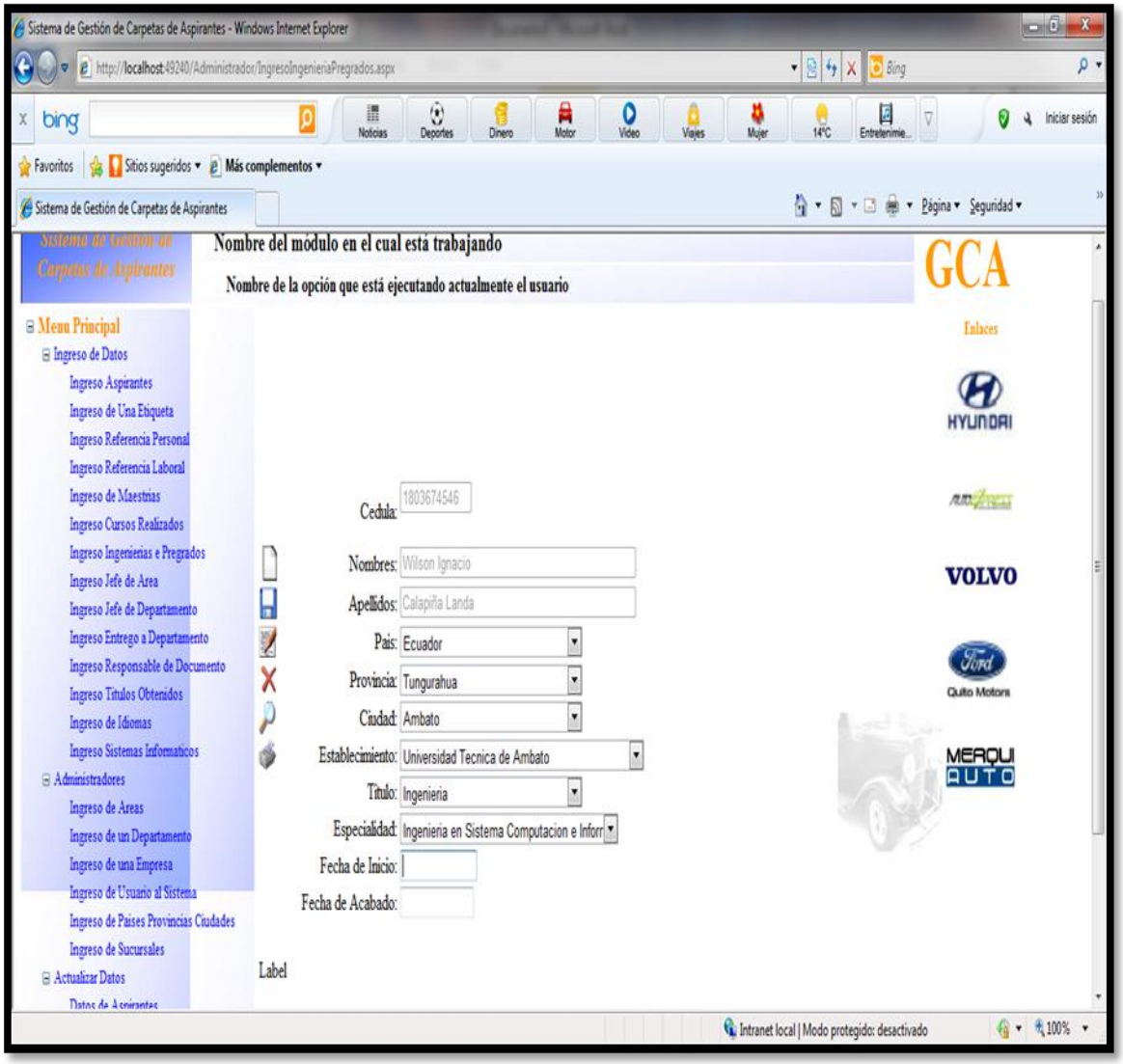
Después registrarse como usuario y contraseña que se le haya destinado. Ya en el sistema se puede observar que tiene muchas funciones se irá explicando paso a paso como esta en el menú de opciones de sistema.


Sistema de Gestión de Ca...
localhost:49240/Administrador/Aspirante.aspx
Compañía: EMPRESA ACTUAL Sucursal: SUCURSAL ACTUAL Usuario: USUARIO
domingo, 17 de abril de 2011 9:00:01
Manual Acceso FAQ Salir
Sistema de Gestión de Carpetas de Aspirantes
Nombre del módulo en el cual está trabajando
Nombre de la opción que está ejecutando actualmente el usuario
GCA
Enlaces
HYUNDAI
VOLVO
Quito Motors
MERQUI AUTO
INGRESO DE DATOS DE PERSONAS
Menu Principal
Ingreso de Datos
Ingreso Aspirantes
Ingreso de Una Etiqueta
Ingreso Referencia Personal
Ingreso Referencia Laboral
Ingreso de Maestrias
Ingreso Cursos Realizados
Ingreso Ingenierias e Pregrados
Ingreso Jefe de Area
Ingreso Jefe de Departamento
Ingreso Entrego a Departamento
Ingreso Responsable de Documento
Ingreso Titulos Obtenidos
Ingreso de Idiomas
Ingreso Sistemas Informaticos
Administradores
Ingreso de Areas
Ingreso de un Departamento
Ingreso de una Empresa
Ingreso de Usuario al Sistema
Ingreso de Paises Provincias Ciudades
Ingreso de Sucursales
Actualizar Datos
Datos de Aspirantes
Datos de la Etiqueta
Consultas o Informes
Consulta Etiqueta
Consulta de Areas
Consulta Departamentos
Consulta Empresas
Consulta Entrega Departamento
Consulta Idiomas
Consulta Ingenieria Pregrados
Consulta de Cursos Realizados
Consulta de Jefe Area
Consulta Jefe Departamento
Consulta de Maestrias
Consulta Paises Provincias Ciudades
Consulta de Referencias Laborales
Foto:
Seleccionar archivo
No se ha...archivo
Subir Imagen
Cedula
Nombres
Apellidos
Sexo
Cedula Militar
Nacionalidad
Fecha de Nacimiento
Lugar de Nacimiento
Edad
Estado Civil
Pais
Provincia
Ciudad
Canton
Sector
Direccion de Domicilio
Correo Electronico
Teléfono
Celular
Licencia
Tipo Licencia
Discapacidad
Nombre de Discapacidad
Hijos

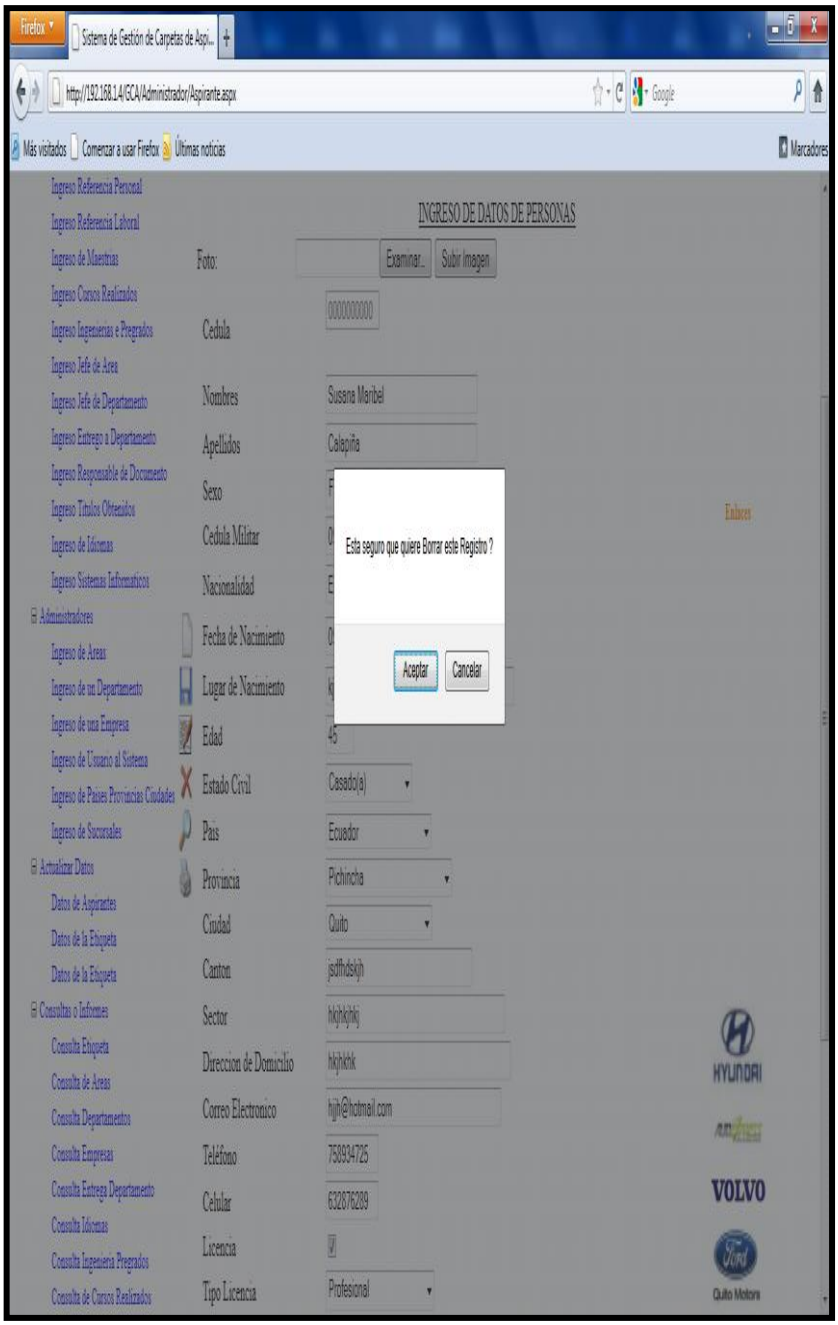
- **Ingreso de Datos** En el ingreso del sistema tenemos un nuevo aspirantes llenamos todos los datos y se procede a grabar.



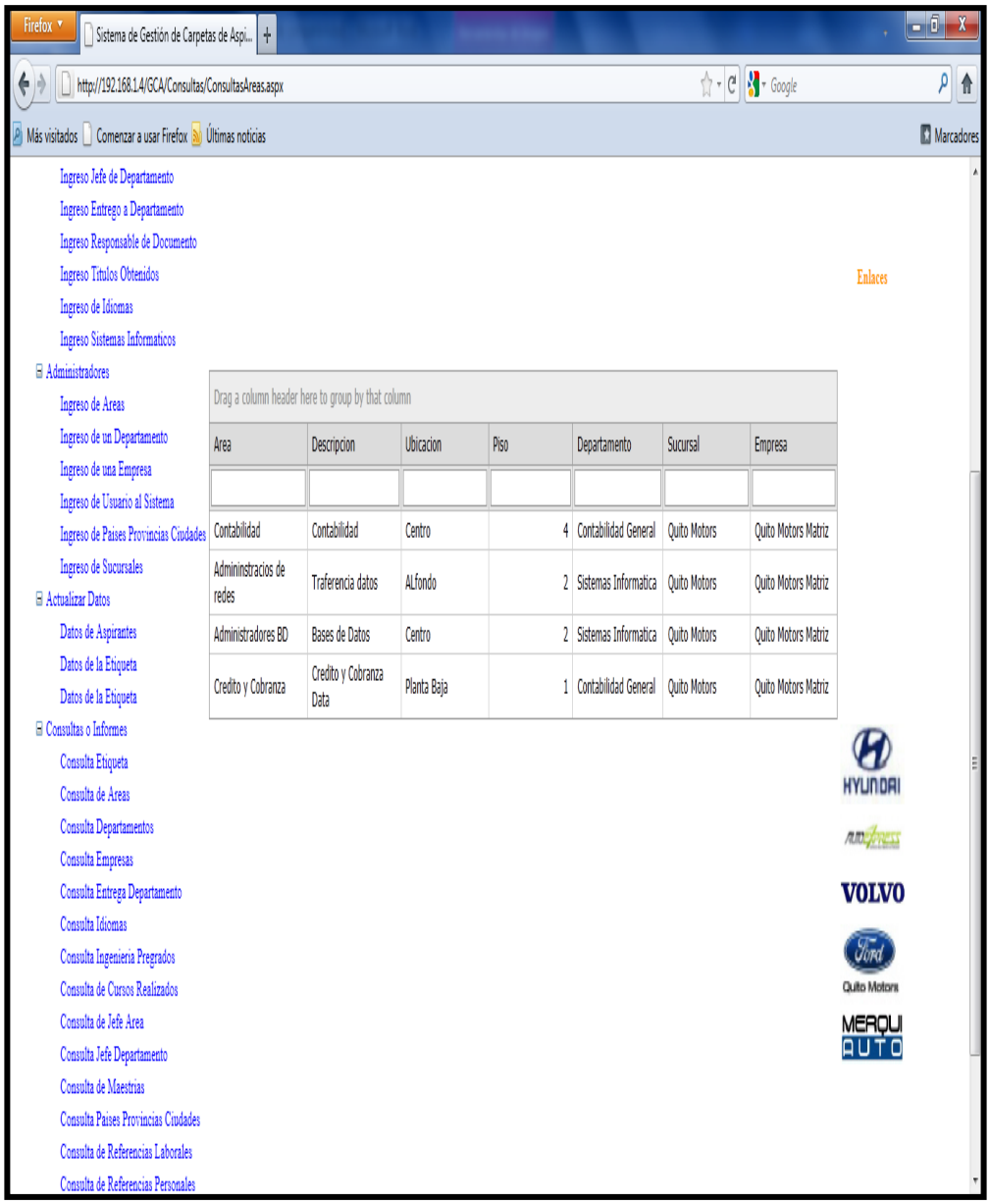
- **Modificación de Datos:** Para modificar la información se puede realizar haciendo una consulta de cedula donde se habilitaran solo los campos que se puede modificar y los demás serán bloqueados, cuando los cambios sean realizados se procede a guardar.



- **Eliminación de Datos** Para eliminar los datos se envía el archivo a consultar a eliminar, todos los datos de los campos deben salir bloqueados y solo el campo que realiza la consulta estar desbloqueada.



- **Reportes de Datos:** Los reportes se puede obtener en diferente archivos en Word, Excel pdf de acuerdo como el usuario necesite



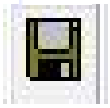
- **Glosario**


Para ingresar nueva información


Para eliminar información


Para Actualizar información


Para realizar reportes


Para guardar información


Para buscar un registro

Los ejemplos de lo ha realizado con un aspirante, pero el procedimiento es el mismo para cualquier clase que necesite o desee trabajar

El documento está escrito y redactado de tal manera, que cualquier persona pueda entenderlo con la menor dificultad posible. Debido a que esta detallado todas las tareas que están implementadas en el sistema.

Los alcances y las limitaciones que tiene el programa.