Universidad Católica Andrés Bello

Facultad de Ingeniería

Escuela de Ingeniería Informática

Ingeniería del Software

Profesor Jacinto Fung

Informe Proyecto

Ivrea Network

**Integrantes:**

Nelly Fernández

C.I. 18223442

Juan Mirelles

C.I. 18546848

Antonio Sánchez

C.I. 19045403

Caracas, 07 de Mayo de 2010

**Índice**

Introducción.....................................................................................3

Alcance …………………….........................................................................4

Análisis…………………………………...............................................................5

Diseño Lógico....................................................................................8

Diseño Físico....................................................................................10

Mejoras al diseño actual....................................................................15

Defectos del diseño………....................................................................15

Conclusión.......................................................................................16

Bibliografía……………………………………………………………………………………………………17

**Introducción**

“IVREA Network” es una aplicación de tipo cliente-servidor, desarrollado para el respaldo y manipulación de la data de cada uno de los departamentos pertenecientes a la empresa “IVREA Company”. Los departamentos de la empresa “IVREA Company” están constituidos por: Administración de Inventarios, Gestión de Proyectos y Recursos Humanos.

Para satisfacer los requisitos del cliente, y cumplir con los objetivos planteados, Se ejecutaran los siguientes puntos:

* Utilización de archivos xml y archivos planos txt como medios de respaldo para la data de los departamentos.
* Se implementará un servidor para la manipulación (búsquedas, eliminaciones consultas y modificaciones) de la data de los departamentos.
* Como medio de comunicación entre el/los cliente/es se utilizará “Sockets”.
* Se crearan ventanas (tipo Windows forms que representarán el/los clientes) con el objetivo que los usuarios puedan interactuar con el servidor para realizar operaciones sobre la data.

Para el desarrollo de los apartados anteriores, se utilizara como entorno de trabajo, Visual Studio C#.Net 08., pues nos ofrece programación orientado a objeto, que nos facilita el diseño y desarrollo de la aplicación con las herramientas que nos ofrece la POO (Encapsulamiento, abstracción, Modularidad, etc.).

Pero tener todas las tecnologías disponibles, no es suficiente para el desarrollo y obtención de la aplicación lo más eficiente posible. Se necesitan ejecutar planes y estrategias convenientemente elaboradas para poder resolver los problemas de la manera más optima, que conllevan el desarrollo del proyecto.

En el presente informe se especificarán todas las técnicas, estrategias y criterios vistas en clases (Elegidas según las necesidades del proyecto) y usadas durante las fases de desarrollo del mismo.

**Alcance**

El proyecto Ivrea Network consiste en el diseño y desarrollo de tres servidores:

* Administración de Inventario (ADI)
* Gerencia de Proyecto (GP)
* Recursos Humanos (RRHH)

Mediante la lógica del desarrollo de la aplicación, el usuario tiene la posibilidad de:

* Registrar un nuevo:

- Usuario

- Empleado

- Activo

- Proyecto

* Eliminar:

- Usuario

- Empleado

- Activo

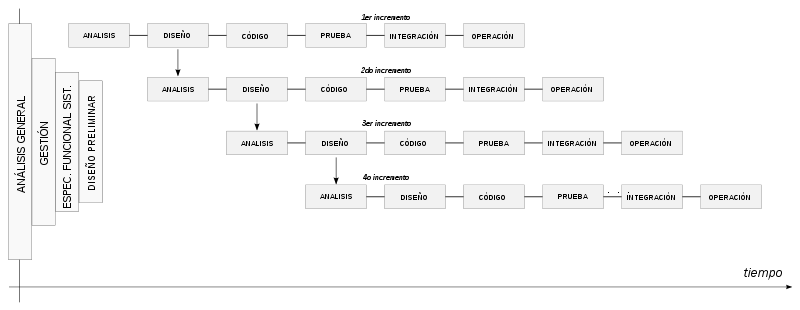
- Proyecto

* Guardar las modificaciones realizadas en la base de datos en archivos planos Txt y archivos Xml, permitiendo así obtener una fuente confiable de almacenamiento de información.
* Tener un acceso de seguridad controlado que no permite a usuarios externos del departamento modificar información almacenada.
* Consultar información perteneciente a otros archivos externos a su departamento.
* Conocer cualquier error ocurrido durante la ejecución de la aplicación, ya que es reflejado en pantalla y en ciertos casos da la posibilidad de corregirlos y realizar nuevamente la solicitud.
* Obtener un manejo sencillo de la información y de las posibles opciones a realizar sobre la base de datos, manejando una interfaz gráfica de eficaz y sencillo funcionamiento.

**Análisis**

**Modelo de desarrollo aplicado:** Modelo Incremental.

“El modelo incremental combina elementos del modelo en cascada aplicado en forma iterativa. El modelo incremental aplica secuencias lineales de manera escalonada conforme avanza el tiempo en el calendario. Cada secuencia lineal produce “incrementos” del software.”



*Figura de desarrollo aplicado Modelo Incremental*

* Comunicación: En esta primera etapa se estableció el análisis de los requerimientos necesarios para este incremento; nos aseguramos de comprender y manejar cada uno de los requerimientos asignados. Esta fase es de vital importancia para evitar ambigüedades o desarrollo de aspectos innecesarios.
* Planeación: En esta fase se estableció las técnicas y estrategias para llevar a cabo esta fase de incremento. Se asignó cada tarea a cada integrante de forma equitativa de acuerdo a los requerimientos planteados en este incremento para lograr la finalización de las mismas en el plazo de tiempo acordado.
* Modelado: Se analizaron las estrategias de solución de cada tarea asignada, se diseño la solución posible para un desarrollo optimo del requerimiento. Se diseño un pequeño diagrama con el comportamiento de los métodos a implementar y su funcionalidad dentro de la lógica del proyecto.
* Construcción: Se procedió a realizar el código (desarrollo) de los modelos o tareas planeadas, luego de finalizada la etapa de desarrollo del código se derivó a las pruebas de las respectivas tareas, para verificar funcionalidad en un ambiente optimo.
* Despliegue: Para finalizar esta fase se procedió a unir las tareas asignadas a cada integrante al proyecto desarrollado hasta este incremento, luego de verificar su correcto acoplamiento y optimo funcionamiento mediante reiteradas pruebas , se concluyo los avances realizadas con éxito y el resultado final de este incremento.

Luego de concluido todos los pasos anteriores se analiza el resultado final del incremento y se procede nuevamente al inicio de los pasos( comunicación) para la realización del siguiente y así sucesivamente hasta lograr concluir exitosamente la totalidad de l proyecto planteado.

Decidimos aplicar este modelo de desarrollo debido a las facilidades que nos proporciona, permitiéndonos desarrollar el proyecto comenzando por la parte básica y luego ir incrementando la dificultad para lograr cubrir la totalidad de los requerimientos propuestos. Además nos proporcionaba un perfecto acoplamiento con la estructura de capas utilizada para la implementación del código (explicado mas adelante), ya que se logro realizar o desarrollar cada capa o gran parte de ella en cada incremento realizado con éxito.

**Ventajas observadas:**

* Resolución de problemas de alto riesgo en tiempos tempranos del proyecto.
* Visión de avance en el desarrollo desde las etapas iniciales del desarrollo.
* Obtención del feedback del usuario lo antes posible, para orientar el desarrollo al cumplimiento de sus necesidades y realizar todas las adaptaciones identificadas para cumplir con los objetivos planteados.
* Menor tasa de fallo del proyecto, mejor productividad del equipo, y menor cantidad de defectos.
* Permite manejar la complejidad del proyecto, apuntando a la resolución de los problemas por partes, y no caer en la inanición del “súper análisis” del producto.
* El aprendizaje y experiencia del equipo iteración tras iteración, mejora exponencialmente el trabajo, aumenta la productividad y permite optimizar el proceso en el corto plazo.
* El trabajo iterativo deja una experiencia en el equipo que permite ir ajustando y mejorando las planificaciones, logrando menores desvíos en la duración total del proyecto.

**Desventajas Observadas**

Hasta el momento se podría decir que no observamos grandes desventajas, pero sí hay puntos a manejar:

* El uso de un desarrollo iterativo e incremental no garantiza por sí solo el éxito de su uso.
* Hay costos ocultos en su implementación, ya que se incorporan varias actividades a realizar por el equipo, y hay que saber medir ese impacto para no fracasar en el intento.
* Es necesario [disponer de técnicas y herramientas que permitan hacer cambios fácilmente](http://www.proyectosagiles.org/requisitos-de-scrum#facilidad-cambios) en el producto, de manera que pueda crecer en cada iteración de manera incremental sin hacer un gran esfuerzo adicional, manteniendo su complejidad minimizada y su calidad.

**Diseño Lógico**

En este punto se llega al entendimiento de cómo funciona el sistema. Al usuario ingresar deberá registrarse luego esté le proporcionará una serie de opciones que le permitirán realizar el manejo de la base de datos de acuerdo a su área, cada consulta o modificación del usuario es manejado mediante un servidor que se encarga de realizar las solicitudes requeridas mediante un protocolo de comunicación que le permitirá realizar los respectivos cambios en la base de datos o obtener la información requerida, asegurándose que se conserve en forma optima la información almacenada.

Cabe destacar que el usuario posee un nivel de acceso a la información, esté se maneja de acuerdo al departamento al que pertenezca; ya que solo puede modificar, eliminar o crear nuevos ítems que pertenezcan a su área; aún así puede consultar la información de otros departamentos si así lo desea. El departamento de cada usuario se conoce y almacena en base de dato mediante la información proporcionada por él mismo al momento de realizar su registro (crear una cuenta).

**Diseño de formas y reportes**

Las entradas y salidas del sistema son mediante ventanas dinámicas que le permiten al usuario acceder, consultar y modificar la base de datos correspondiente de acuerdo al nivel de acceso que posea.

Son manejadas adecuadamente para cuidar los siguientes aspectos:

* Control de la [Calidad](http://www.monografias.com/trabajos11/conge/conge.shtml) de Entrada
* Evitar los Retrasos
* Evitar los errores en los datos
* Evitar los pasos adicionales
* Mantener la Sencillez del Proceso

En estas ventanas se mostraran mediante un menú las posibles opciones proporcionadas por el sistema para el manejo de la información, el usuario realiza la opción de su preferencia y el sistema la lleva a cabo retornando un mensaje con el estado final de dicha opción, permitiéndole conocer al usuario si su petición se logro o no con éxito.

**Diseño de interfaces y diálogos**

El patrón de interacción entre el sistema y el usuario es por medio de una ventana principal que posee un menú que presenta las siguientes opciones a realizar:

1. *Ingresar*

Que permite al usuario acceder al sistema luego de proporcionar y ser validado su Login y Password, este es un requisito indispensable ya que asegura que el usuario esta registrado en la base de datos para así acceder a ella y conocer su nivel de acceso antes mencionado. Luego de ingresar logrará realizar las consultas o modificaciones de la información que requiera.

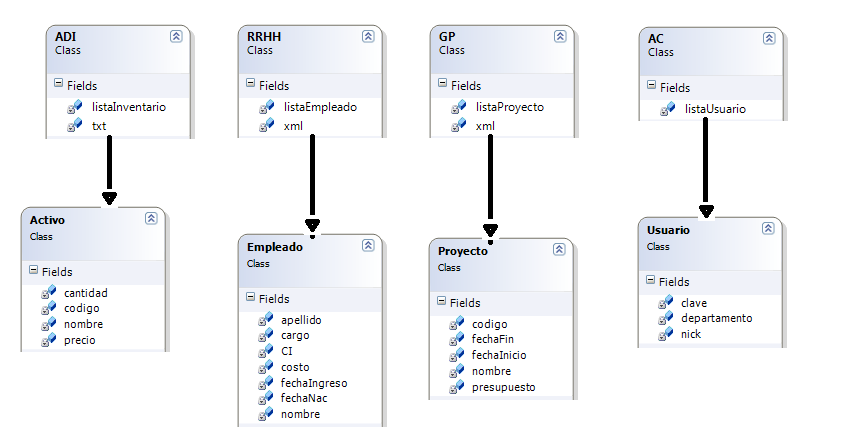
1. *Crear Cuenta*

Esta opción proporciona a un nuevo usuario la forma de registrarse en el sistema. Debe proporcionar un Login que debe ser único y un Password, con el cual será manejado el control de acceso para certificar la seguridad de la base de datos que contiene la aplicación.

1. *Eliminar Cuenta*

Acción que le permite a un usuario ya registrado eliminar su cuenta. Para llevar a cabo esta opción solo se requiere su Login y Password, luego de verificada su información se procede a eliminar de la base de datos al usuario impidiendo su acceso a la aplicación en futuros intentos de ingreso.

**Diseño lógico de la base de datos:**



En la figura anterior se muestra como se manejará la base de datos en la cual se manejan los siguientes elementos:

**Diseño Físico**

En esta etapa se especifican todas las características tecnológicas del sistema. Se especifican las estructuras para datos y programas que nos proporcionan un sistema con trabajo eficiente y de manera segura.

**Diseño de archivos físicos y base de datos**

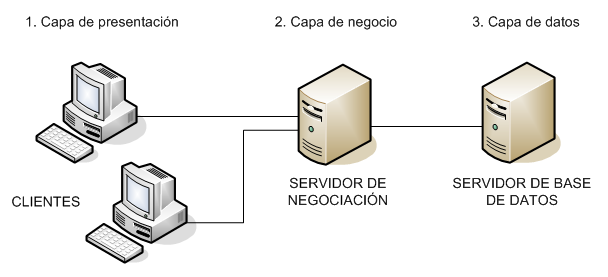
En este sistema los datos serán almacenados en archivos planos Txt y archivos Xml. El medio de acceso a dichos archivos es mediante las herramientas proporcionadas por las librerías System.IO y System.Xml contenidas en el ambiente de desarrollo de Microsoft Visual Studio 2008; que nos permiten realizar creaciones de archivo y posteriormente las respectivas modificaciones de la información contenida en cada uno de ellos.

Los requerimientos con respecto a la base de datos almacenada en la memoria del computador serán a través de solicitudes realizadas por el usuario mediante la interfaz grafica (detallada más adelante), estas solicitudes son procesadas por la capa Lógica desarrollada para luego ser almacenadas en memoria mediante la capa de Acceso a datos implementada.

Las validaciones fundamentales de la estructura adecuada de cada tipo de dato son realizadas parcialmente en la ventana de interfaz del usuario y principalmente en la capa Lógica, en ella mediante métodos desarrollados con tal fin, se verifica que no existan discrepancias en los tipos de datos proporcionados por el usuario, por ejemplo, que la cedula no posea algún tipo de carácter diferente de números o que le fecha este en el formato adecuado establecido. Todo esto con el fin de asegurar la calidad de los datos almacenados.

**Diseño de estructuras del sistema y del programa**

Para la estructura del sistema decidimos emplear Programación por Capas (cuatro capas); ya que cubría nuestros requerimientos al permitirnos primordialmente la separación de la lógica de negocios de la lógica de diseño (GUI); proporcionándonos una ventaja principal como es el poder llevar a cabo la aplicación en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado.



*Modelo básico de programación por capas*

Además, al permitirnos distribuir el trabajo de creación de una aplicación por niveles esto nos hace más fácil el desarrollo, ya que se acopla al modelo de desarrollo aplicado (Modelo incremental); logrando obtener en cada incremento una capa o parte de una capa del modelo.

Cabe destacar que la estructura cuatro capas se divide en:

* Interfaz gráfica de usuario: posee las pantallas que interactúan con el usuario.
* Lógica: posee el manejo de validaciones, procedimientos y acciones para llevar a cabo la optimización y correcto funcionamiento del sistema.
* Acceso a datos: encargado del manejo de los métodos para realizar el acceso, consulta o modificación a los archivos y lograr cumplir los requerimientos solicitados por el usuario
* Dominio: en él se definen las clases bases que posee el sistema, son entidades atómicas fundamentales para lograr llevar a cabo la estructura deseada.

Es importante mencionar que la capa Interfaz gráfica de usuario y Lógica no tienen contacto directo, más bien se comunican mediante un protocolo de servidor TCP/IP tipo Socket de Microsoft.

En la siguiente figura se refleja el diagrama alusivo a la estructura antes mencionada aplicada

Interfaz Gráfica de Usuario (GUI)

TCP/IP

Lógica

Dominio

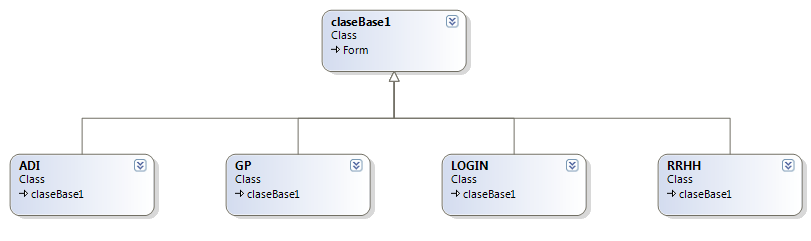
Acceso a datos

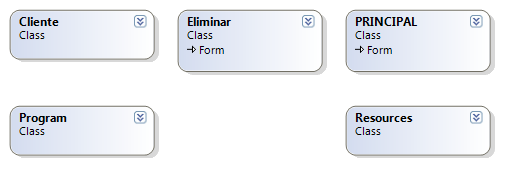
Xml

Txt

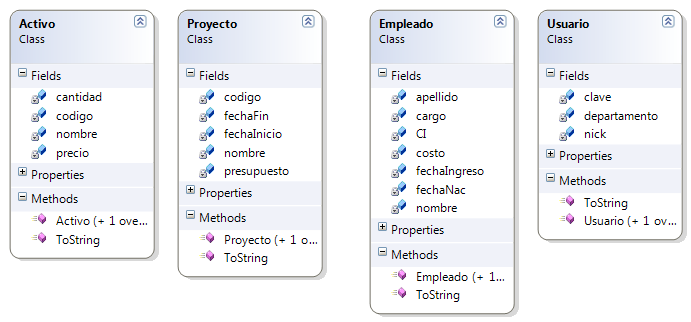
A continuación se presentará una breve descripción de los elementos que posee cada capa de desarrollo.

* **Interfaz gráfica de usuario:** Posee cuatro clases: ADI, GP, LOGIN, RRHH que heredan de la clase claseBase1, ya que comparten los mismos métodos y se realizo de esta forma para una mejor optimización y reutilización del código desarrollado. Además posee la clase Cliente, Eliminar y Principal. Todas contienen las propiedades y acciones manejadas en cada pantalla.

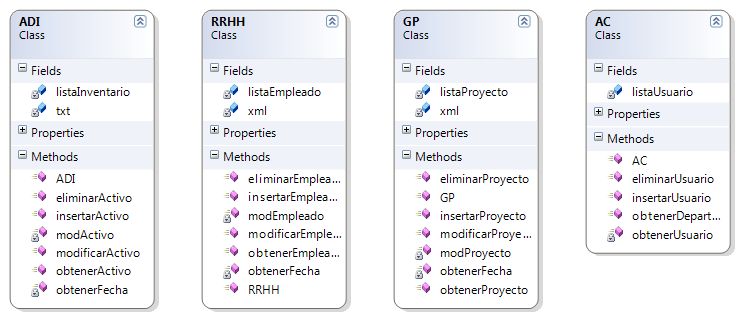




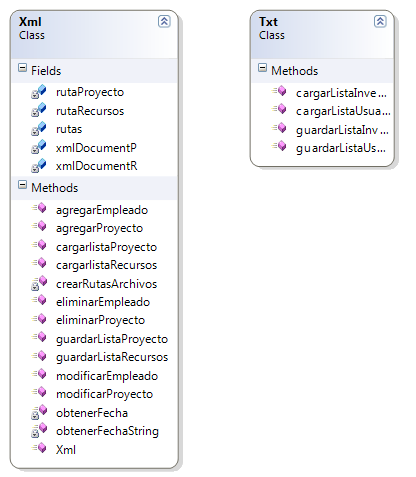
* **Dominio:** Posee las clases entidades o bases de la estructura planteada para la solución del proyecto, solo poseen la definición de los atributos manejados en cada área o departamento de la compañía.

****

* **Lógica:** Contiene cuatro clases que representan los tres departamentos de la empresa y una clase adicional para el manejo del acceso de seguridad de los usuarios registrados. Estas clases (clases Controladoras) contienen el manejo de las listas con los ítems respectivos a su departamento.



* **Acceso a Datos:** Contiene dos clases Txt y Xml que son las encargadas del manejo de los archivos (consulta y modificación) que contienen la información de la base de datos de los usuarios, activos, empleados y proyectos almacenados.



* **Txt y Xml:** representan los archivos físicos respectivos de cada departamento de la compañía.

**Defectos del diseño**

El proyecto presenta los siguientes defectos:

* Al agregar en cualquiera de las aéreas de trabajo (RRHH, GP, ADI), si el campo nombre posee un espacio (Ejemplo: “RELAMPAGO AZUL”) el sistema no funciona debidamente, ya que por tener el espacio AZUL será tomado como otro atributo.

Nota: Esto NO ocurre cuando se registra un usuario al sistema (quien posee el control del proyecto según su área), para esté el Error fue solventado.

* Si se agrega en RRHH o GP (los cuales poseen el atributo Fecha), dependiendo del tipo de computadora donde se agrega la información el formato no será el mismo del servidor trayendo como consecuencia información incorrecta al almacenarlo.

* La información de las distintas bases de datos las tenemos tanto en archivos físicos (Planos TXT, Dinámicos XML) como en memoria del software, esto resulta una desventaja si la empresa contiene mucha información ya que al modificar, agregar, eliminar un campo de algún objeto se modifica tanto en el archivo como en la lista. Pero a su vez resulta una ventaja ya que ir en cada momento al archivo físico haría que la operación fuera lenta y si por alguna razón se va la luz existe grandes posibilidades de perder la información.

**Mejoras al diseño Actual**

* Eliminación en el protocolo de la posición en bytes (4-7) “Cantidad de campos a enviar”. Debido a que si usamos la función Split() de la tecnología Microsoft C# no es indispensable saber la cantidad de campos que envía el Cliente (ya que se podría calcular si es necesario).

Cabe destacar que aunque lo señalamos como una mejora, el proyecto Ivrea Network cumple con el protocolo (aunque no es usado tanto por Cliente como por Servidor).

**Conclusión**

Los equipos de desarrollos deben ser cuidadosos durante la planificación, análisis y diseño del proyecto (Elección del modelo de desarrollo, asignaciones de tares, motivación, etc.). Ya que el software resultante depende en gran parte de las estrategias utilizadas y del aprovechamiento de los recursos disponibles (tiempo, tecnología, etc.).

Aunque “IVREA Network” es un proyecto “moderadamente corto” y no está en juego recursos económicos, todos los requerimientos del mismo (Cliente, servidor, protocolo de comunicación y bases de datos) tanto a nivel físico como lógico son dependientes entre sí, y en consecuencia, elegimos dividir las tareas por cada integrante del equipo, obligándonos a desarrollar técnicas para maximizar el trabajo en equipo (mini-Exposiciones entre los integrantes , Presentaciones en Power Point, Documentos escritos explicativos acerca de las necesidades de cada intégrate, mapas).

Dichas técnicas nos facilitaron y nos ahorraron trabajo, en cuanto al desarrollo de las tareas de cada integrante, así como el ensamblaje final del proyecto.

“IVREA Network” no solamente nos permitió poner en prácticas las estrategias y las metodologías vistas en clases necesarias para el desarrollo de la aplicación, sino que también nos permitió indagar (quizás no muy profundos) en temas como las redes (Sockets, protocolos de comunicación, servidores, puertos, etc.) y en nuevas

Tecnologías (VS c#.Net 08).

En esencia, no existen una planificación ni estrategias específicas para resolver un determinado proyecto y los “pequeños proyectos” no necesariamente resultan ser “fácilmente” desarrollados sin una buena planificación o estrategias.

**Bibliografía**

<http://upload.wikimedia.org/wikipedia/commons/e/ea/Tres_capas.PNG>

<http://bdigital.eafit.edu.co/bdigital/PROYECTO/P005.1M516/capitulo2.pdf>

<http://www.proyectosagiles.org/desarrollo-iterativo-incremental>

Pressman, Roger S. Ingenieria del SoftWare. Un enfoque Practico Mc Graw Hill.