Universidad de las Ciencias Informáticas

Facultad 2

Sistema para el control de tiempo de producción en los proyectos productivos de la universidad de las Ciencias Informáticas

Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autores: Onel Roselló Reyes

Carlos Sotolongo Alonso

Tutor: Ing. Yohamnes Hernández Báez

Co-tutor: Ing. Arian Zulueta Casal

Julio de 2007

DECLARACIÓN DE AUTORÍA

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

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

Onel Rosello Reyes Arian Zulueta Casal

Carlos Sotolongo Alonso Yohamnes Hernández Báez

_____ _____

Firma del Autor Firma del Tutor

_____ _____

Firma del Autor Firma del Tutor

AGRADECIMIENTOS

Un agradecimiento especial a:

Nuestro tutor Yohamnes Hernández Báez por ser el creador de esta idea, por estar siempre pendiente de nosotros y por trasmitir el conocimiento para la realización de este trabajo. Eres un espejo para nosotros. Arian Zulueta Casal nuestro cotutor, que se sumó inesperadamente pero asumió la tarea y nos acogió con la voluntad que lo caracteriza de ayudar y brindar su apoyo a quien lo necesite. La Revolución y Fidel que fue el creador de esta magnífica idea y siempre confió en nosotros.

Carlos y Onel

A todos aquello que de una forma u otra han contribuido al desarrollo de este trabajo. A todos los profesores que a lo largo de estos 5 años han sabido cultivar en nosotros un sin número de valores, que han luchado tanto para convertirnos en lo que hoy somos. A nuestros compañeros que siempre pusieron su granito de arena, dándonos mucho ánimo y apoyándonos en todo momento. A Daniel García Fernández que fue el motor impulsor, siempre estuvo ahí, aportando sus ideas y brindándonos su ayuda. A Maidelis Milanés, Alberto Tamayo, Ronny Zamora y Yokiro Valdés Melo fuero como un dios, siempre dándome mucho apoyo y confianza.

Carlos

Un agradecimiento especial Al Ing. Arian Zulueta Casal que siempre estuvo en los momentos difíciles, que siempre supo darme el consejo correcto. A todos que en un momento de crisis tendieron su mano amiga, le gente de porcyon, y sus agregados, la gente de manzanillo, leyvis…... A dos amigos Yoandris S Pacheco y Daniel M Garcia que siempre han sabido estar en todo momento, a ese par de HP que tanto apoyo me dieron y por formar Dyos junto a mí. A mis padres que a pesar de todo son los culpables de ser lo que soy jejejejeje. Quiero dar las gracias a Maybel Marrero una indeseable que conocí que en todo memento me ofreció una sincera amistad incondicional.

Onel

DEDICATORIA

Le dedico este trabajo a mis seres más queridos. En especial a mis Padres que tantos sacrificios hicieron para que hoy yo pueda estar escribiendo estas palabras. Que siempre supieron guiarme por el mejor camino A mis amigos Daniel, Pacheco, Alexis que más que hermano siempre fue un amigo.
Onel

Este trabajo va dedicado a toda mi familia, por estar todos siempre pendientes de mí. A todos mis tíos y primos, en especial a Tonito, Juan Pedro, Papurri y mi abuela Nelia por sus consejos y su apoyo. A mis padrinos Yanay y Leobanis que siempre han estado ahí, gracias a ellos puedo decir que tengo 4 padres. A mis hermanos Giced y Alexander, que han sido siempre mi inspiración y ejemplo para triunfar. A todos los compañeros que de una forma u otra me han brindado su mano en cualquier momento. A mi tutor Yohamnes que siempre me ayudó y guió mucho en todos los sentidos. Y un agradecimiento muy especial a Bernardo Rivas y su esposa Isabel, sin ellos nunca hubiese llegado hasta aquí. A mis padres que son 3lo más grande que tengo y sin ellos mi vida no tendría sentido, el orgullo más grande que tengo es la confianza que depositaron en mí, este es el fruto de su sacrificio. Y me disculpan los que no menciono saben que los tengo presentes pero son muchos, se me hace imposible expresar tanto en tan pocas líneas.

Carlos.

RESUMEN

Con el Presente trabajo titulado “Sistema para el control del tiempo de producción en los proyectos productivos de la Universidad de las Ciencias Informáticas” se pretende llevar a cabo un control estricto de las actividades que realizan los estudiantes en el tiempo de máquina asignado a la producción. Este sistema permitirá brindar la información relacionada con cada usuario y computadora vinculada a determinado proyecto productivo, permitiendo conocer realmente en que se está invirtiendo el tiempo y los recursos asignados a la producción en la UCI.

La aplicación contará con 2 módulos, un módulo Cliente y otro Servidor, entre estos existirá una comunicación que permitirá garantizar el tráfico y el almacenamiento de la información referente a cada computadora que se desee monitorear vía remota. Además el sistema contará con una base de datos capaz de almacenar de manera organizada toda la información que se maneja durante este proceso.

El sistema dará la posibilidad de emitir en cualquier momento una serie de reportes de vital interés para los directivos de los proyectos, permitiendo mostrar datos estadísticos relacionados a las actividades desarrolladas en los proyectos productivos. Estos reportes además brindaran la opción de imprimirlos en diferentes formatos, en caso de necesitar una copia dura.

TABLA DE CONTENIDOS

INTRODUCCIÓN

En la actualidad en las empresas modernas es común que exista una completa dependencia de los medios de cómputo. Esto trae consigo que el trabajo en muchas ocasiones sea operado mediante el uso de una computadora por parte de los trabajadores. Teniendo en cuenta que en las computadoras actuales un gran por ciento de su propósito de construcción está basado en la búsqueda de nuevos mecanismos de ocio y entretenimiento, lo que nos lleva a la siguiente interrogante: ¿cuánto tiempo de máquina es empleado en propósitos ajenos al plan de trabajo de una empresa que base su funcionamiento en el desarrollo de productos informáticos?

Estudios realizados sobre el tema han llegado a la conclusión que como mínimo cerca de dos horas por cada jornada de trabajo de ocho horas son empleadas en acciones ajenas al trabajo de las empresas, el resultado de este estudio no incluye horario de almuerzo y de merienda, que es común en todos los centros de trabajo.

Por una cuestión de práctica las empresas siempre asumen una pérdida de tiempo a la hora de calcular el salario del trabajador, asumiendo tiempos de descansos permitidos, pero estos estudios demuestran que lo asumido está muy por debajo de lo que realmente se gasta. Un ejemplo es en los Estados Unidos de América donde el monto de perdidas esta aproximadamente alrededor de 759 mil millones de dólares por año.

Nuestro país ha comenzado a integrarse a este mundo de las nuevas tecnologías, por lo que no está ajeno a este tipo de pérdidas. Actualmente nos vemos inmersos en un proceso de racionalización y ahorro en pos de la optimización y el desarrollo, lo que nos incita a tomar medidas urgentes con el objetivo aprovechar más los tiempos de producción. Estas medidas posibilitarían una mejora constante de nuestros productos que a su vez permiten lograr un uso eficiente del tiempo dedicado a la producción.

En la Universidad de las Ciencias Informáticas (UCI) se está realizando una importante inversión tecnológica formada por sistemas computarizados, redes y aplicaciones, permitiendo la implantación de proyectos productivos los cuales permitirán al país un incremento en la economía, reportando una importante suma de divisas.

Para dichos proyectos productivos se cuenta con un cierto tiempo de desarrollo el cual se está obligado a cumplir para contribuir a lograr una mayor seguridad para con los clientes, alcanzar una gran reputación en el mercado del software y lograr una mayor integración Cliente-Entidad.

Actualmente no existe alguna forma que permita llevar a cabo un control del aprovechamiento del tiempo de máquina dedicado a la producción. Debido a ello es imposible calcular hasta el momento cuanto tiempo es realmente aprovechado en función de la producción en los proyectos productivos. Y en los mejores casos solo se tiene en cuenta un estimado muchas veces realizado por el mismo programador que no refleja para nada la realidad. Todo esto implica que un gran por ciento del tiempo de máquina destinado al trabajo del proyecto no se esté aprovechando debidamente.

Hasta el momento es imposible para el líder del proyecto obtener información detallada sobre el trabajo desarrollado por un programador, a fin de poder realizar una valoración real de la tarea que pudo haberse entregado satisfactoriamente o no. Lo que podría servir como base para el análisis de los fallos que pudieron ocasionarse o de lo que pudo haberse hecho mejor.

En pos de facilitarle al jefe de proyecto el control del tiempo dedicado a la producción se encuentra un grupo de estadísticas que le serian útiles como por ejemplo: conocer el usuario que está trabajando en la computadora en cada momento, que aplicaciones ha ejecutado, el tiempo que estuvieron activas, así como el control de la hora en que se inicien y cierren las mismas contribuyendo a un mejor análisis del flujo de información manejado en estos procesos.

El principal objetivo de este trabajo consiste en el análisis, diseño e implementación de un prototipo funcional capaz de llevar a cabo el control de las tareas antes mencionadas. Además que se convierta en una herramienta útil no solo para nuestra universidad sino que su uso pueda ser extensible a todas las empresas que necesitan tener un control sobre el tiempo dedicado a desarrollar productos de software. De esta forma podría convertirse en línea base o punto de partida de un software con la calidad suficiente para ser exportado.

Problema Científico.

¿Cómo gestionar los tiempos de las actividades de los proyectos productivos en la UCI?

Objeto de Investigación.

Procesos automatizados de gestión de los tiempos de producción.

Objetivo de la Investigación.

Realizar el análisis, diseño e implementación de un sistema informático para gestionar los tiempos de producción en los proyectos productivos de la UCI.

Campo de acción

Sistema de gestión de los tiempos de producción de proyectos productivos en la UCI.

Tareas de Investigación

1. Realizar un estudio exhaustivo para recopilar la mayor cantidad de información referente al tema.
2. Llevar a cabo el análisis, diseño e implementación del prototipo funcional capaz de dar solución al problema planteado.
3. Someter la aplicación a un riguroso plan de pruebas, con el objetivo de comprobar su correcto funcionamiento.

Posibles resultados de la investigación

1. Estudio de las principales herramientas de control o monitoreo existentes a nivel nacional e internacional.
2. El desarrollo de una herramienta de monitoreo para controlar los tiempo de producción.
3. El sistema desarrollado podrá ser sometido a un plan de pruebas para detectar en qué grado cumple con los objetivos de su desarrollo.
4. El sistema permitirá ser desplegado en las redes de los proyectos productivos de la Universidad de las Ciencias Informáticas para dar cumplimiento a los objetivos para los cuales fue diseñado.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 INTRODUCCIÓN

La Universidad de las Ciencias informáticas en la actualidad carece de un sistema capaz de llevar a cabo por parte de los jefes de proyectos el control del tiempo dedicado a los proyectos productivos para los cuales en la mayoría de las veces es dedicado a otro fin que no incluye el correcto empleo de los medios de cómputos.

En el presente capítulo se realiza un estudio del estado del arte a cerca de las técnicas de programación existentes, además de un análisis de las tendencias, tecnologías y metodologías utilizadas en la actualidad a nivel internacional, en el desarrollo de aplicaciones de este tipo, las principales herramientas existentes con el fin de llevar a cabo el control del tiempo, así como las plataformas de desarrollo que las soportan, fundamentándose las seleccionadas para la solución que se propone.

1.2 ESTADO DEL ARTE

La informatización de herramientas de control de tiempo ha sido objeto de estudio de numerosos proyectos de software, cada uno con algunas características peculiares, que brindan innumerables funcionalidades.

Existen en la actualidad disímiles aplicaciones que nos brindan información relacionada con lo que está ocurriendo en nuestro sistema. Algunos ejemplos de esta información sería, que usuario se encuentra trabajando en cada momento, que procesos se están ejecutando, el uso del CPU (Unidad Central de Procesamiento) de cada proceso, el tiempo de inicio y de fin de cada proceso, la ventana que se encuentra activa entre otras cosas. Además brindan la posibilidad de obtener esta misma información desde una computadora remota, poniendo en práctica la tecnología cliente-servidor. Pero con esta aplicación se pretende profundizar más en el tema, porque a partir de esa información obtenida se realizará un profundo análisis para obtener datos concretos

referentes al tiempo destinado a cada actividad y consideramos que su uso es indispensable para los líderes de proyectos dedicados al desarrollo de software.

En el ámbito internacional podemos contar con numerosos proyectos dedicados al control de los tiempos de producción cada uno con sus particularidades. Entre las más comunes se encuentran.

TimeSheet: Es un sistema de control de tiempo especialmente creada para medianas y pequeñas empresas de servicios. Permite al personal de una empresa registrar el tiempo empleado en las diferentes actividades asociadas a los proyectos de esta, y obtener información sobre tiempos de ejecución, costo de los proyectos y rendimiento del personal.

Entre sus principales características están:

Clientes, proyectos, tareas de usuarios: El sistema permite tener múltiples clientes donde a cada uno de estos se les asignan proyectos, que a su vez tienen asociados tareas. Esta herramienta de administración permite definir proyectos y sus tareas, asociando usuarios a ambos. Los usuarios, entonces, registran sus actividades en las tareas asignadas.

Revisión y alerta de usuario morosos: Con un periodo de tiempo definido por el administrador, el sistema revisa los usuarios que no hayan llenado su plantilla de tiempos y envía un correo electrónico a estos recordándole hacerlo. También es enviada dicha información al administrador general.

Soporte detallado de las horas trabajadas: Los administradores cuentan con varias herramientas para observar los proyectos y emitir reportes que dan respuesta a preguntas del tipo:

1. ¿En qué proyectos ha trabajado mi empleado N este mes?
2. ¿Cuánto me costó cada una de las actividades del proyecto Z?

3. ¿Cuánto he trabajado en horas para el cliente X en los últimos 3 meses?
4. ¿Qué tipo de proyectos es más rentable para mi empresa?

Vista de Actividades: El sistema permite a los usuarios observar y editar sus actividades realizadas en diferentes modalidades de tiempo (Mensual, Semanal, Diaria, Simple). El administrador puede ver sus actividades y las de los demás usuarios. (1)

Otra de las herramientas utilizadas por las pequeñas y medianas empresas en el ámbito internacional es el Jourmyx TimeSheet.

Journyx TimeSheet (Ver Anexo 1) es una herramienta especialmente creada para medianas y pequeñas empresas de servicios. Permite al personal de una empresa registrar el tiempo empleado en las diferentes actividades asociadas a los proyectos de ésta, y obtener información sobre tiempos de ejecución, costos de los proyectos, rendimiento del personal. (2)

Como otro de los proyecto distinguido dedicado a lograra obtener mejor controles de los tiempos de trabajos en algunas de las empresas se encuentra el Time Tracker. (Ver Anexo 2) esta es una aplicación WEB desarrollada con Ajax que nos permite gestionar el tiempo que debemos dedicar a una tarea. El funcionamiento es bien sencillo: nos registramos en la aplicación, y una vez estemos dentro, veremos un panel en el que aparecerán todas las tareas que vayamos añadiendo.

Cuando se añade una tarea hay que decir cuál es la hora de inicio y la hora en que ha de terminar. Así, cuando se cumpla ese tiempo, se nos avisará de que ya debería de haber terminado dicha tarea. Naturalmente, para que esto funcione, debemos dejar abierto el navegador con la página cargada.

Sé sabe que existen muchos sistemas que permiten hacer esto, pero la ventaja es que con Time Tracker podemos organizar las tareas de un día entero incluso aunque no estemos en nuestro ordenador. (3)

Otras de las variantes menos funcionales aplicadas para el control de tiempo de las actividades dentro de una empresa dedicada al desarrollo de software es la utilización de una hoja de cálculo. Ventajas que ofrece: rápido de poner en marcha.

Desventajas: podemos perder mucho tiempo preparando la hoja de cálculo y preparando un informe de seguimiento de cada proyecto. Y el método más simple es la utilización de documentos en formato duro. Este puede ser quizás el más sencillo para el control del tiempo.

Su ventaja: si son pequeños volúmenes de información es rápido de operar y podemos llevarlo siempre con nosotros.

La desventaja: cuando el cúmulo de información es muy grande se torna imposible de operar, además de ser entre papeles ¿por qué siempre tienden a desaparecer? (4)

Todas estas herramientas y técnicas mencionadas anteriormente son las más usadas en la actualidad para el control de los tiempos de producción en las pequeñas, medianas y grandes empresas dedicadas al desarrollo de software. Pero estas prácticas traen consigo planificaciones irreales de los tiempos estimados a las actividades asignadas. Ya que en muchas de estas herramientas son los propios usuarios lo que se reportan y planifican el tiempo que le debe dedicar a cada actividad.

Otra herramienta de gran importancia para las empresas e instituciones con grandes recursos y Tecnologías de la Información (TI) es el DameWare NT Utilities (DNTU) que es una aplicación para la administración de sistemas empresariales, que le permite al administrador ir más allá de las limitaciones de la Consola de Administración de Microsoft (MMC). DNTU provee una gran colección de utilidades integradas en una interfaz sencilla de utilizar para la administración remota de plataformas Windows. Solo que este se limita a funcionalidades puramente de control no ofrece registros de tareas

para llevar u registro y visualizar las tareas realizadas en los ordenadores monitoreadas. (5)

Principales características:

1. Vista de Drives de Discos
2. Vista de Log de Eventos
3. Vista de Grupo Global
4. Vista de Grupo Local
5. Vista de los Miembros
6. Vista de los Archivos Abiertos
7. Vista de las Impresoras
8. Vista de los Procesos
9. Vista de las Propiedades
10. Vista de Administración de RAS
11. Vista del Registro
12. Vista del Comando Remoto
13. Vista de Replicación
14. Vista de Schedule
15. Vista de Send Message
16. Vista de Search
17. Vista de Sesiones
18. Vista de Servicios/Dispositivos
19. Vista de Sesiones
20. Vista de Compartidos
21. Vista de Shutdown
22. Herramientas de Sistema
23. Utilitarios TCP
24. Vista de Terminal Server
25. Vista de Domain Users
26. Vista de Usuarios Vista Wake On LAN

Además de:

1. Administración de usuarios de Microsoft Exchange.
2. Soporte para Windows Terminal Server.
3. Despliegue de dominios de la red de Microsoft Windows de manera jerárquica.
4. Posibilidades de agregar Dominios NT a la selección de Favoritos.
5. Control Remoto de sus quipos con plataformas W95/98/ME/NT/2000/XP.

DameWare posee la funcionalidad Mini Remote Control:

1. Herramienta para tomar el control remoto de equipos bajo ambientes Windows
2. Puede ejecutarse como aplicación o como servicio.

DameWare Exporter:

1. Permite exportar la información de los dispositivos lógicos y físicos de las redes Windows.
2. Genera reportes dinámicos que pueden ser importados desde una hoja de cálculos o desde una base de datos. (6)

En las pequeñas y medianas empresas de software cubanas no existen proyectos dedicados al desarrollo de tales herramientas, por lo que se hace casi imposible llevar un control estricto de las actividades asignadas en los proyectos dedicados al desarrollo de software. Lo mismo pasa en nuestra universidad, donde el proceso productivo tiene un ritmo acelerado y tampoco cuenta con tan importante herramienta. Aunque algunos proyectos productivos en la UCI utilizan la herramienta de código abierto *Time Tracker* para el control de los tiempos de producción de sus desarrolladores, es bueno señalar que dicha herramienta no cumple con todas las características que tendrá nuestro sistema.

1.3 METODOLOGÍAS Y HERRAMIENTAS.

1.3.1 LA TECNOLOGÍA .NET.

.NET es un proyecto de Microsoft para crear una nueva plataforma de desarrollo de software con énfasis en transparencia de redes, con independencia de plataforma y que permita un rápido desarrollo de aplicaciones. Basado en esta plataforma, Microsoft

intenta desarrollar una estrategia horizontal que integre todos sus productos, desde el sistema operativo hasta las herramientas de mercado.

.NET podría considerarse una respuesta de Microsoft al creciente mercado de los negocios en entornos WEB, como competencia a la plataforma Java de Sun Microsystems.

A largo plazo Microsoft pretende reemplazar la API (Application Programming Interface*)* Win32 (Conjunto de APIs) o Windows API con la plataforma .NET. Esto debido a que l*a* API Win32 o Windows API fue desarrollada sobre la marcha, careciendo de documentación detallada, uniformidad y cohesión entre sus distintos componentes, provocando múltiples problemas en el desarrollo de aplicaciones para el sistema operativo Windows. La plataforma .NET pretende solventar la mayoría de estos problemas proveyendo un conjunto único y expandible con facilidad, de bloques interconectados, diseñados de forma uniforme y bien documentados, que permitan a los desarrolladores tener a mano todo lo que necesitan para producir aplicaciones sólidas.

Debido a las ventajas que la disponibilidad de una plataforma de este tipo puede darle a las empresas de tecnología y al público en general, muchas otras empresas e instituciones se han unido a Microsoft en el desarrollo y fortalecimiento de la plataforma .NET, ya sea por medio de la implementación de la plataforma para otros sistemas operativos aparte de Windows (Proyecto Mono de Ximian/Novell para Linux/MacOS X/BSD/Solaris), el desarrollo de lenguajes de programación adicionales para la plataforma (C Estándar de la Universidad de Princeton, Net COBOL de Fujitsu, Delphi de Borland, entre otros) o la creación de bloques adicionales para la plataforma (como controles, componentes y bibliotecas de clases adicionales); siendo algunas de ellas software libre, distribuibles bajo la licencia GPL.(Licencia Pública General)

Con esta plataforma Microsoft incursiona de lleno en el campo de los servicios WEB y establece el XML (Extensible Markup Language) como norma en el transporte de

información en sus productos y lo promociona como tal en los sistemas desarrollados utilizando sus herramientas.

.NET intenta ofrecer una manera rápida y económica pero a la vez segura y robusta de desarrollar aplicaciones o como la misma plataforma las denomina, soluciones permitiendo a su vez una integración más rápida y ágil entre empresas y un acceso más simple y universal a todo tipo de información desde cualquier tipo de dispositivo.

1.3.1.1 NET FRAMEWORK

El "framework" o marco de trabajo, constituye la base de la plataforma .NET y denota la infraestructura sobre la cual se reúnen un conjunto de lenguajes, herramientas y servicios que simplifican el desarrollo de aplicaciones en entornos de ejecución distribuidos.

Bajo el nombre .NET Framework o Marco de trabajo .NET se encuentran reunidas una serie de normas impulsadas por varias compañías además de *Microsoft* (como Hewlett-Packard, Intel, IBM, Fujitsu Software, Plum Hall, la Universidad de Monash), entre las cuales se encuentran:

La norma que define las reglas que debe seguir un lenguaje de programación para ser considerado compatible con el marco de trabajo .NET (ECMA (European Computer Manufacturers Association)-335, ISO/IEC 23271).

Por medio de esta norma se garantiza que todos los lenguajes desarrollados para la plataforma ofrezcan al programador un conjunto mínimo de funcionalidad, y compatibilidad con todos los demás lenguajes de la plataforma.

La norma que define el lenguaje C# (ECMA-334, ISO/IEC 23270)

Este es el lenguaje insignia del marco de trabajo .NET, y pretende reunir las ventajas de lenguajes como C/C++ y Visual Basic en un solo lenguaje.

La norma que define el conjunto de funciones que debe implementar la librería de clases base (BCL) incluido en ECMA-335, ISO/IEC 23271.

Tal vez esta norma es el más importante de los componentes de la plataforma, puesto que define el conjunto funcional mínimo que debe implementarse para que el marco de trabajo sea soportado por un sistema operativo. Aunque Microsoft implementó esta norma para su sistema operativo Windows, la publicación de la norma abre la posibilidad de que sea implementada para cualquier otro sistema operativo existente o futuro, permitiendo que las aplicaciones se ejecuten sobre la plataforma independientemente del sistema operativo para el cual haya sido implementada. El Proyecto Mono emprendido por Ximian pretende realizar la implementación de la norma para varios sistemas operativos adicionales bajo el marco del software libre o código abierto.

Los principales componentes del marco de trabajo son:

El conjunto de lenguajes de programación.
La Biblioteca de Clases Base o BCL.
El Entorno Común de Ejecución para Lenguajes o CLR

Debido a la publicación de la norma para la infraestructura común de lenguajes (CLI por sus siglas en inglés), el desarrollo de lenguajes se facilita, por lo que el marco de trabajo .NET soporta ya más de 20 lenguajes de programación y es posible desarrollar cualquiera de los tipos de aplicaciones soportados en la plataforma con cualquiera de ellos, lo que elimina las diferencias que existían entre lo que era posible hacer con uno u otro lenguaje. Algunos de los lenguajes desarrollados para el marco de trabajo .NET son: C#, Visual Basic, Turbo Delphi para .NET C++, J#, Perl, Python, Fortran y Cobol.NET.

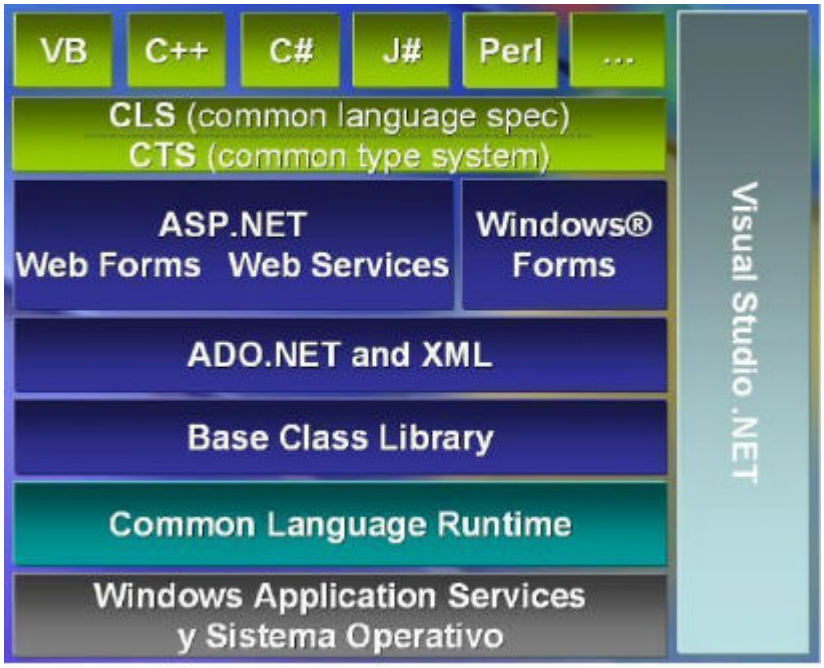

Figura 1Diagrama detallado del Marco de Trabajo .NET

1.3.1.2 VISUAL STUDIO .NET

Visual Studio .NET es un IDE (Entorno Integrado de Desarrollo) desarrollado por Microsoft a partir de 2002. Es para el sistema operativo Microsoft Windows y está pensado, principal pero no exclusivamente, para desarrollar para plataformas Win32

La característica más notable del IDE es su soporte de los nuevos lenguajes .NET. Los programas desarrollados en esos lenguajes no se compilan a código máquina ejecutable (como por ejemplo hace C++) sino que son compilados a algo llamado CIL. Cuando los programas ejecutan la aplicación CIL, ésta es compilada en ese momento al código de máquina apropiado para la plataforma en la que se está ejecutando. Mediante este método, Microsoft espera poder soportar varias implementaciones de sus sistemas operativos Windows (como Windows CE). Los programas compilados a CIL pueden ejecutarse sólo en plataformas que tengan una implementación de .NET framework. Es

posible ejecutar programas CIL en Linux o en Mac OS X utilizando algunas implementaciones .NET que no pertenecen a Microsoft, como Mono y DotGNU.

1.3.1.3 COMMON LANGUAGE RUNTIME

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

La herramienta de desarrollo compila el código fuente de cualquiera de los lenguajes soportados por .NET en un código intermedio (MSIL, Microsoft Intermediate Lenguaje), similar al BYTECODE de Java. Para generar dicho código el compilador se basa en el Common Language Specification (CLS) que determina las reglas necesarias para crear ese código MSIL compatible con el CLR.

Para ejecutarse se necesita un segundo paso, un compilador JIT (Just-In-Time) es el que genera el código máquina real que se ejecuta en la plataforma del cliente.

De esta forma se consigue con .NET independencia de la plataforma y del hardware.

La compilación JIT la realiza el CLR a medida que el programa invoca métodos, el código ejecutable obtenido, se almacena en la memoria caché del ordenador, siendo recompilado de nuevo sólo en el caso de producirse algún cambio en el código fuente.

1.3.1.4 BIBLIOTECA DE CLASES BASE DE .NET

La Biblioteca de Clases Base maneja la mayoría de las operaciones básicas que se encuentran involucradas en el desarrollo de aplicaciones, incluyendo entre otras:

Interacción con los dispositivos periféricos.
Manejo de datos (ADO.NET).

Administración de memoria.
Cifrado de datos.
Transmisión y recepción de datos por distintos medios (XML, TCP/IP).
Administración de componentes WEB que corren tanto en el servidor como en el cliente (ASP.NET).
Manejo y administración de excepciones.
Manejo del sistema de ventanas.
Herramientas de despliegue de gráficos (GDI+).
Herramientas de seguridad e integración con la seguridad del sistema operativo.
Manejo de tipos de datos unificado.
Interacción con otras aplicaciones.
Manejo de cadenas de caracteres y expresiones regulares.
Operaciones aritméticas.
Manipulación de fechas, zonas horarias y periodos de tiempo.
Manejo de arreglos de datos y colecciones.
Manipulación de archivos de imagines.
Aleatoriedad.
Generación de código.
Manejo de idiomas.
Auto descripción de código.
Interacción con el API Win32 o Windows API.
Compilación de código.
Esta funcionalidad se encuentra organizada por medio de espacios de nombres jerárquicos.
La Biblioteca de Clases Base se clasifica, en tres grupos clave:
ASP.NET y Servicios WEB XML.
Windows Forms.
DO.NET.

1.3.1.5 ENSAMBLADOS

Los ensamblados son ficheros con forma de EXE o DLL (Bibliotecas de Enlace Dinámico) que contienen toda la funcionalidad de la aplicación de forma encapsulada. Con los ensamblados ya no es necesario registrar los componentes de la aplicación. (7)

1.3.2 JAVA ENTERPRISE EDITION

Java Platform, Enterprise Edition o Java EE (anteriormente conocido como Java Platform, Enterprise Edition (J2EE) hasta la versión 1.4), es una plataforma de programación (parte de la plataforma Java) para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura de n niveles distribuida, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. La plataforma Java EE está definida por una especificación. Similar a otras especificaciones del Java Community Process, Java EE es también considerada informalmente como un estándar debido a que los suministradores deben cumplir ciertos requisitos de conformidad para declarar que sus productos son conformes a Java EE; no obstante sin un estándar de las organizaciones internacionales para la estandarización ISO o ECMA.

Java EE incluye varias especificaciones de API, tales como JDBC, RMI, e-mail, JMS, Servicios WEB, XML, etc., y define como coordinarlos. Java EE también configura algunas especificaciones únicas para Java EE para componentes. Estas incluyen Enterprise JavaBeans, servelets, portlests (siguiendo la especificación de Portlets Java), JavaServer Pages y varias tecnologías de servicios WEB. Esto permite al desarrollador crear una Aplicación de Empresa que es portable entre plataformas y escalable, mientras integramos con tecnologías de legado. Otros beneficios añadidos son, por ejemplo, que el servidor de aplicaciones puede manejar las transacciones, seguridad, escalabilidad, concurrencia y gestión de los componentes que son desplegados, significando que los desarrolladores pueden concentrarse más en la lógica de negocio de los componentes en lugar de las tareas de mantenimiento de bajo nivel. (8)

1.3.3 CRYSTAL REPORT

Crystal Report permite la elaboración de reportes en .NET. Esta herramienta incluye diversos controles para graficar, un soporte de visualización múltiple paginado, un panel de tabla de contenidos con una nueva pestaña de miniaturas, posibilidad de búsqueda de informes y personalización completa de su barra de herramientas.

Se integra con Visual Studio .NET permitiendo la utilización desde C# o Vb, incluye filtros de exportación a los formatos Adobe PDF, Microsoft Excel, Microsoft Word, Rich Text Format (RTF), lenguaje de marcación de hipertexto (HTML) y HTML Dinámico (DHTML). También se incluye un visor WEB que aprovecha los manejadores de ASP.NET para permitirle visualizar los informes sin necesidad de escribir código personalizado para exportar.

Las versiones disponibles son:

Crystal Reports Server: Novedad entre las diferentes ediciones, al proporcionar una solución completa para diseñar, crear, gestionar y distribuir todos los informes de la empresa. Viene a rellenar el vacío existente para la gestión de informes corporativos en empresas que no podían adquirir Crystal Enterprise por quedar fuera de sus presupuestos.

Crystal Reports Developer Edition: Esta edición permite integrar capacidades de visualización, exportación e impresión de informes en las aplicaciones.

Crystal Reports Professional Edition: Diseñado para crear y mantener informes con los requisitos de diseño más exigentes. Además incorpora la posibilidad de acceder a fuentes de datos corporativas.

La versión actual de Crystal Reports es la versión XI. Y está diseñado para conectarse a cualquier base de datos. Los productos pueden ser en idioma inglés o español. Cada paquete incluye una media de instalación y la llave de activación para una licencia.

La forma de licenciamiento que tiene Crystal Reports: Cada usuario que diseñe reportes, modifique reportes debe tener una licencia instalada, las personas que vean la pantalla del reporte no necesitan tener instalada una licencia. (9)

1.3.4 ACTIVE REPORT PARA .NET V 2.0

La segunda versión de ActiveReports para .NET incorpora diversas y nuevas características para aumentar las capacidades de generación de informes alabadas por los desarrolladores que han utilizado la versión anterior. Entre ellas podemos destacar la incorporación de un nuevo control de gráficos comerciales muy potente, las miniaturas de páginas en el visor de Windows, el soporte HTML y el soporte mejorado para tablas en el control RichTextBox, un editor de guiones mejorado con resaltado de sintaxis, enlace a datos contra clases personalizadas que soporten la interfaz IList y mejoras en la detección de tiempos límites de respuesta en conexiones a bases de datos (7).

ActiveReports para .NET está escrito completamente en C# y ofrece integración con Visual Studio .NET. Esto permite a los desarrolladores de Visual Studio .NET utilizarlo desde su lenguaje favorito (C# o Visual Basic .NET). Esta herramienta se licencia por desarrollador y es de libre distribución. El producto incluye un asistente de informes y un asistente de conversión de informes de Microsoft Access para trabajar rápidamente. Además incluye filtros de exportación a los formatos Adobe PDF, Microsoft Excel, RTF, HTML, Texto y TIFF, y un visor Windows con soporte para la visualización simultánea de múltiples páginas, un panel de Tabla de Contenidos con una nueva pestaña de miniaturas, posibilidad de búsqueda de informes y personalización completa de su barra de herramientas.

La edición Professional de ActiveReports para .NET ofrece un Diseñador de Informes para el usuario final que le permite embeber el diseñador en sus propias aplicaciones para que sus usuarios puedan crear y modificar sus propios informes. También se incluye un visor WEB que aprovecha los manejadores de ASP.NET para permitirle visualizar los informes sin necesidad de escribir código personalizado para exportar a los formatos más utilizados como HTML y PDF.

Novedades en versión 2.0 (ambas ediciones):

1. Control de gráficos comerciales totalmente integrado.
2. Control RichText con soporte para tablas RTF y etiquetas HTML.
3. Soporte para el enlace a datos contra clases que implementan IList.
4. Exportación mejorada a formato PDF.

Edición Estandar

1. Diseñador de informes integrado.
2. Soporte para fuentes de datos OleDB, SQL Server y XML, además de ADO.NET (lectores de datos, tablas y conjuntos de datos).
3. Incluye un visor Windows personalizable.
4. Utilidades de importación de informes Crystal y Microsoft Access.
5. Exportación a HTML, PDF, Excel, RTF, TIFF y Texto.

Edición Professional

1. Incluye todas las características de la edición Standard.
2. Incluye un control Diseñador de Informes para usuarios finales.
3. Incluye un visor WEB ASP.NET. (10)

1.3.5 LENGUAJE DE PROGRAMACIÓN C++

C y C++ son dos de los lenguajes más utilizados en el campo de la ingeniería y la programación de sistemas. La principal razón es que C y C++ proporcionan el nivel de abstracción preciso para construir una aplicación compleja, pero, al mismo tiempo, ofrecen mecanismos de bajo nivel que permiten a los programadores hacer uso de las características más avanzadas de las plataformas sobre las que se ejecutan sus programas. Microsoft ha creado C# que combina algunas de las características más avanzadas de Java con algunas de las más potentes de C y C++ con el objetivo de convertirlo en el nuevo lenguaje de Internet y, por supuesto, en el lenguaje nativo para acceder a todos los servicios que en el futuro brindará .NET. (11)

1.3.6 LENGUAJE DE PROGRAMACIÓN C#

C# es el lenguaje de .NET construido especialmente para adaptarse de manera natural al framework y aprovechar al máximo todas sus características. Al igual que C y C++, permite programar fácilmente a bajo nivel, este incorpora características encontradas en los lenguajes industriales y de investigación más habituales. Gracias a esto, acceder a las características avanzadas de la plataforma sobre la que se trabaje, crear código muy eficiente en aquellos puntos de la aplicación que son críticos y acceder a las Interfaces de Programación de Aplicaciones (Apis) existentes es perfectamente posible. Muchos dicen que si Java se puede considerar un C++ mejorado en cuestiones de seguridad y portabilidad, C# debe entenderse como un Java mejorado en todos los sentidos. (11)

1.3.7 SISTEMA GESTORES DE BASE DE DATOS

Una Base de Datos (BD) es un conjunto de datos interrelacionados, almacenados con carácter más o menos permanente en la computadora, puede ser considerada una colección de datos relacionados, variables en el tiempo [MATOS 99]. En la actualidad existen numerosos sistemas gestores de bases de datos, entre ellos el Microsoft Access, Oracle, MySQL, Visual Fox Pro, etc. Microsoft SQL Server 2000.

1.3.7.1 MICROSOFT SQL *SERVER* 2000

Microsoft SQL Server 2000 es un Servidor de Base de Datos y herramienta de Análisis de la información. Proporciona seguridad, fiabilidad y escalabilidad para poner en marcha cualquier aplicación en un tiempo pequeño, destacando sus sencillas tareas de administración y su capacidad de analizar la información.

SQL Server permite la creación de procedimientos almacenados, los cuales consisten en instrucciones que se almacenan dentro de una base de datos de SQL Server, realizados en lenguaje SQL. Son procedimientos que se guardan semicompilados en el servidor y que pueden ser invocados desde el cliente. Se ejecutan más rápido que instrucciones SQL independientes [SQL].

1.3.7.2 MICROSFT ACCESS SERVER 2003

Microsoft Access es un sistema de gestión de bases de datos creado y modificado por Microsoft (DBMS) para uso personal o de pequeñas organizaciones. Su principal función es ser una potente base de datos, capaz de trabajar en sí misma o bien con conexión hacia otros lenguajes de programación, tales como Visual Basic 6.0, Visual Basic .NET o Visual C++ 6.0. Pueden realizarse consultas directas a las tablas contenidas mediante instrucciones SQL. Internamente trae consigo el lenguaje Visual Basic para Aplicaciones (VBA) el cual es similar en forma a VB6.

Permite el ingreso de datos de tipos: Numéricos, Texto, Fecha, Sí/No, OLE, Moneda, Memo y Booleano. Pueden desarrollarse aplicaciones completas basadas en Microsoft Access, pues trae consigo las herramientas necesarias para el diseño y desarrollo de formularios para el ingreso y trabajo con datos e informes para visualizar e imprimir la información requerida.

Su funcionamiento se basa en un motor llamado Microsoft Jet, y permite el desarrollo de pequeñas aplicaciones autónomas formadas por formularios Windows y código VBA (Visual Basic para Aplicaciones). Una posibilidad adicional es la de crear ficheros con bases de datos que pueden ser consultados por otros programas. (12)

1.3.8 PROGRAMACIÓN EXTREMA

Programación Extrema (XP) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración. El ciclo de vida ideal de XP consiste de seis fases: Exploración, Planificación de la Entrega (Versión), Iteraciones, Producción, Mantenimiento y Muerte del Proyecto.

La metodología se basa en:

1. Pruebas Unitarias: se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándonos en algo hacia el futuro, podamos hacer pruebas de las fallas que pudieran ocurrir. Es como si nos adelantáramos a obtener los posibles errores.
2. Re fabricación: se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
3. Programación en pares: una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Es como el chofer y el copiloto: mientras uno conduce, el otro consulta el mapa. (13)

1.3.9 PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE

RUP es un proceso de desarrollo de software dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. Pretende implementar las mejores

prácticas en ingeniería de software, con el objetivo de asegurar la producción de software de calidad, dentro de plazos y presupuestos predecibles.

1.3.9.1 PRINCIPALES CARACTERÍSTICAS

1. Desarrollo iterativo: Permite una comprensión creciente de los requerimientos, a la vez que se va haciendo crecer el sistema. Rational Unified Process (RUP) sigue un modelo iterativo que aborda las tareas más riesgosas primero. Así se logra reducir los riesgos del proyecto y tener un subsistema ejecutable tempranamente.
2. Administración de requerimientos: RUP describe cómo obtener los requerimientos, cómo organizarlos, cómo documentar los requerimientos de funcionalidad y restricciones, cómo rastrear y documentar las decisiones, y cómo captar y comunicar los requerimientos del negocio.
3. Arquitecturas basadas en componentes: El proceso se basa en diseñar tempranamente una arquitectura base ejecutable. Esta arquitectura debe ser: flexible, fácil de modificar, intuitivamente comprensible, y debe promover la reutilización de componentes.
4. Modelamiento visual: RUP propone un modelamiento visual de la estructura y el comportamiento de la arquitectura y las componentes. En este esquema, los bloques de construcción deben ocultar detalles, permitir la comunicación en el equipo de desarrollo, y permitir analizar la consistencia entre las componentes, entre el diseño y entre la implementación. UML es el lenguaje utilizado para sustentar la documentación del modelamiento visual de RUP.
5. Verificación de la calidad del software: No sólo la funcionalidad es esencial, también el rendimiento y la confiabilidad. RUP ayuda a planificar, diseñar, implementar, ejecutar y evaluar pruebas que verifiquen estas cualidades.
6. Control de cambios: Los cambios son inevitables, pero es necesario evaluar si éstos son necesarios y también es necesario rastrear su impacto. RUP indica cómo controlar, rastrear y monitorear los cambios dentro del proceso iterativo de desarrollo.

RUP divide el proceso de desarrollo en ciclos, donde se obtiene un producto al final de cada ciclo. Cada ciclo se divide en cuatro Fases: Concepción, Elaboración,

Construcción, y Transición. Cada fase concluye con un hito bien definido donde deben tomarse ciertas decisiones.

1.3.9.2 FASE DE CONCEPCIÓN

En esta fase se establece la oportunidad y alcance del proyecto. Se identifican todas las entidades externas con las que se trata (actores) y se define la interacción en un alto nivel de abstracción: se deben identificar todos los casos de uso, y se deben describir algunos en detalle. La oportunidad del negocio incluye: definir los criterios de éxito, identificación de riesgos, estimación de recursos necesarios, y plan de las fases incluyendo hitos.

1.3.9.3 FASE DE ELABORACIÓN

Definir y validar una arquitectura estable. Se hace un refinamiento de la visión del sistema, basándose en nueva información obtenida durante esta fase, se establece una sólida comprensión de los casos de uso más críticos que definen las decisiones arquitectónicas y de planificación. Creación de los planes de desarrollo detallados para las iteraciones de la fase de construcción.

1.3.9.4 FASE DE CONTRUCCIÓN

Gestión de los recursos, optimización y control de los procesos de construcción del software.

Se completa el desarrollo de los componentes y/o subsistemas, probándolos contra un conjunto definido de criterios aprobados al inicio del proyecto.

1.3.9.5 FASE DE TRANSICION

Ejecución de los planes de implantación. Se finalizan los manuales de usuario y mantenimiento. Pruebas del sistema en el entorno de explotación. Creación de una versión del sistema. Validación del sistema por los usuarios. Ajuste fino del sistema

según la validación con el usuario. Se facilita la transición del sistema al personal de mantenimiento. Se pone el producto a disposición del usuario. (14)

1.3.10 LENGUAJE DE MODELACIÓN

El Lenguaje de Modelación Unificado (UML - Unified Modeling Language) es un lenguaje gráfico para visualizar, especificar, construir y documentar cada una de las partes que comprende el desarrollo de software. Posee formas de modelar conceptos como por ejemplo las funciones del sistema, además de otras particularidades como la de escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables. Usa procesos de otras metodologías, aprovechando la experiencia de sus creadores [UML 03]. Es desde finales de la década del 90, un lenguaje de modelación orientado a objetos estándar, de acuerdo con el Object Management Group, siendo utilizado diariamente por grandes organizaciones como: Microsoft, Oracle, Rational, etc. Existen herramientas Case de trabajo visuales como el Análisis, el Diseño, el Rational Rose, que permiten realizar el modelado del desarrollo de los proyectos. En la actualidad una de las mejores y más utilizadas en el mercado mundial que provee el modelado visual basado en UML es Rational Rose por lo que es la usada en la modelación de este proyecto. (15)

Una vez analizadas todas estas herramientas y tecnologías hemos decidido emplear para la aplicación servidor el lenguaje de programación C++ mezclado con C# ya que C++ permite programar a bajo nivel con mayor facilidad que otros lenguajes lo que posibilita explotar al máximo las características que brinda la tecnología sobre la cual estemos desarrollando, y algunas de estas funcionalidades son exportadas a C# por las facilidades que ofrece. Además brinda un gran poder de abstracción convirtiéndolo así en el principal candidato para desarrollar aplicaciones muy complejas y difíciles de modelar, como la que se pretende desarrollar. Para el desarrollo de la aplicación cliente se ha elegido el lenguaje C#, puesto que brinda muchas facilidades a la hora de interactuar con los gestores de bases de datos antes mencionados. También se ha elegido porque brinda la posibilidad de programar eficientemente en lugares de la

aplicación que son críticos, dando la posibilidad de acceder a las interfaces de programación de aplicaciones (Apis) existentes en la plataforma .NET.

1.4 ANALISIS CRÍTICO DE LAS FUENTES BIBLIOGRAFICAS UTILIZADAS

Para el cumplimiento de los objetivos de la investigación, las fuentes bibliográficas consultadas hasta el momento son insuficientes. Ya que es un tema poco tratado tanto en el ámbito nacional como internacional. Los elementos fundamentales de la investigación están sustentados en pequeños artículos y foros existentes en la red. Se hace muy difícil encontrar libros o artículos de autores o compañías de prestigio, ya que al estar trabajando sobre tecnología propietaria, la documentación acerca de la misma está restringida.

1.5 CONCLUCIONES

En este capítulo se hizo un análisis del estudio del arte de las principales tecnologías y herramientas utilizadas, de los procesos que se llevan a cabo en el mismo, así como de los paradigmas de programación existentes. Se definieron las tecnologías y herramientas a utilizar en el desarrollo de aplicaciones de escritorio. A partir de este análisis llegamos a las siguientes conclusiones:

1. Utilizar la plataforma Microsoft .NET para el desarrollo de la solución propuesta, especialmente Microsoft Visual Studio .NET 2003 y como lenguaje de programación Microsoft Visual C# .NET para la aplicación cliente, teniendo en cuenta que estos ofrecen una gran productividad componente ineludible por el poco tiempo que se tiene para la entrega de la aplicación.
2. Para la aplicación servidor utilizaremos como lenguaje de programación C++ ya que ofrece mayores prestaciones para el desarrollo a bajo nivel que es el centro de la solución propuesta.
3. Usar el Proceso Unificado de Desarrollo de Software como metodología de desarrollo de software.

4. Utilizar el Crystal Reports como herramienta para el diseño y visualización de reportes en Microsof*t* .NET, ya que se integra a este permitiendo su utilización desde C# y permitirle visualizar los informes sin necesidad de escribir código personalizado para exportar.
5. Usar SQL Server 2000 para la aplicación cliente y Microsoft ACCES 2000 para la aplicación servidor.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 INTRODUCCIÓN

En el presente capítulo se explica cómo es el actual control del uso de los tiempos de máquina para la producción en la Universidad de la Ciencias Informáticas, se identifican las principales necesidades de la institución y se describe el proceso que será automatizado. Además de plasmar el estudio de los diferentes sistemas de control de los tiempos de producción que persiguen el mismo objetivo que el propuesto.

Se presenta además la propuesta del sistema y se especifican los requerimientos funcionales y no funcionales. Se realiza la definición de los casos de uso, de los actores que intervienen en ellos y se determinan los casos de uso del primer ciclo de iteración.

2.2 SITUACIÓN PROBLÉMICA.

La utilización de los medios de cómputo en los proyectos de la Universidad de las Ciencias Informáticas, excepto en algunos proyectos puntuales, así como el control de los tiempos de producción de los mismos es algo que no se conoce con exactitud o prácticamente es nulo. No se pueden precisar con exactitud las actividades que realizan los estudiantes en el horario dedicado al desarrollo.

Actualmente el control de tiempo de producción es casi imposible. Aunque en algunos casos se utilizan algunas herramientas para controlar los estereotipos creados o en los que se está trabajando como es el caso del TimeTracker. Por tal motivo se desea automatizar el control de las actividades reales que realizan los estudiantes en los tiempos de máquina asignados.

El objetivo del presente trabajo es automatizar las actividades relacionadas al empleo del tiempo en los medios de cómputo de la UCI, a través de una aplicación que poseerá funcionalidades que faciliten el control de estas actividades y además contará con una interfaz amigable para los usuarios finales del sistema (Jefes de Proyectos).

2.3 OBJETIVOS ESTRATÉGICOS DE LA ORGANIZACIÓN Y PROCESO DE NEGOCIO QUE LOS SOPORTAN

Como una de las metas estratégicas de la Universidad de las Ciencias Informáticas se encuentra la formación científico-técnica y algunas disciplinas como la investigación científica o el trabajo de desarrollo técnico de los estudiantes en actual formación; así como la superación gradual de los actuales profesionales con que cuenta la institución.

Para lograr que todos los objetivos vinculados con esta tarea se cumplan la universidad ha trazado una serie de estrategias que facilitan el trabajo y que permite a los involucrados hacer un uso eficiente y consciente de los medios de cómputo puestos a su disposición. Además el centro tiene grandes aspiraciones de automatizar todos los procesos que en él se desarrollan, pues se ha trabajado bastante en la infraestructura productiva, con el fin de sustituir las labores manuales por potentes herramientas capaces de automatizar todo cuanto se pueda.

En la actualidad los procesos de control de tiempo de producción o tiempo de máquina, cosas que no se realizan o se realizan muy poco en nuestro centro, pues las pocas herramientas existentes no son capaces de satisfacer todas las necesidades reales de la institución. Esto provoca que no se le esté dando un uso racional a los recursos en los cuales nuestro país ha invertido grandes sumas de dinero. Incluso que no se utilicen con los objetivos para los cuales fueron concebidos y se pierda la potencialidad que brinda toda la infraestructura con que cuenta la universidad.

2.4 OBJETO DE AUTOMATIZACIÓN

En determinado período de tiempo por necesidades de la automatización de algún proceso o negocio de la universidad; o por convenios contraídos con alguna institución nacional o extranjera, la UCI se ve en la necesidad de formar algunos grupos de desarrollo o de incorporar fuerzas de trabajo en caso de que estos grupos existan.

Una vez conformado y consolidadas las bases necesarias del grupo de desarrollo se le entrega un grupo de medios de cómputo con el fin de realizar las tareas asignadas. Según la importancia o necesidad del mismo se pueden asignar más de un estudiante a una misma computadora para darle mayor utilidad a los medios asignados.

Cada estudiante tiene una cuenta o sesión de trabajo en la computadora desde la cual podrá realizar las tareas que le fueron asignadas. Muchas veces, por motivos justificados o injustificados las entregas no se realizan en el tiempo planificado. Para estos atrasos o incumplimientos de entregas, el jefe de proyecto no sabe con precisión cuales fueron los motivos reales que provocaron tal situación. Es decir, si fueron malas planificaciones de proyectos o por motivos de actividades ajenas al mismo, realizadas en los tiempos de producción planificados para este fin. Por consiguiente no existe manera alguna de controlar con exactitud las actividades desarrolladas por los estudiantes.

En proyectos puntuales se utiliza una herramienta para tratar de controlar estos tiempos de producción llamada TimeTracker, ésta en sus diferentes variantes es lo más cercano con que se cuenta en la UCI para controlar las actividades que realizan los estudiantes en los tiempos de máquina asignados. Además existen proyectos que de forma interna cuentan con una herramienta capaz de llevar este control, así como de acercarse a las necesidades reales del jefe de proyecto.

Este proceso brinda una breve descripción de cómo es que se desarrolla el control de los tiempos de producción y el uso de los medios de cómputo en los proyectos productivos de la Universidad de las Ciencias Informáticas.

Por todo lo antes expuesto se decidió implementar en la UCI, un sistema de control de los tiempos de producción, el cual será capaz de controlar las actividades que realizan los estudiantes, así como de garantizar con mayor precisión los tiempos de entrega de los proyectos y verificar en que los estudiantes invierten realmente los tiempos de máquina asignados.

Para desarrollar todo lo que se propone en el sistema se cuenta con las herramientas necesarias, así como con el apoyo de la institución y el soporte técnico necesario.

2.5 INFORMACIÓN QUE SE MANEJA

El sistema que se pretende desarrollar manipulará toda la información referente a los procesos de Windows, es decir el identificador del proceso (PID), el nombre del proceso, la ventana del proceso en caso de tenerla, el tiempo de inicio, tiempo de fin, el tiempo de activación, el usuario que creó el proceso, etc.

2.6 PROPUESTA DE SISTEMA

2.6.1 DESCRIPCIÓN GENERAL DE LA PROPUESTA DEL SISTEMA

El sistema que se propone presenta funcionalidades que automatizará el control del horario asignado a las labores productivas de la UCI, pues el mismo contará con una interfaz fácil y amigable donde la institución como mayor interesada tendrá las herramientas de trabajo necesarias para la gestión de las actividades desarrolladas por los estudiantes. Brindará información importante para los directivos, pues estos podrán conocer en todo momento las actividades que se están realizando, el tiempo utilizado para una actividad, la fecha y hora en que comenzó, la fecha y hora en que finalizó, así como otro tipo de información de importancia para el control de tales actividades.

2.7 MODELO DE DOMINIO.

2.7.1 DIAGRAMA DEL MODELO DE DOMINIO

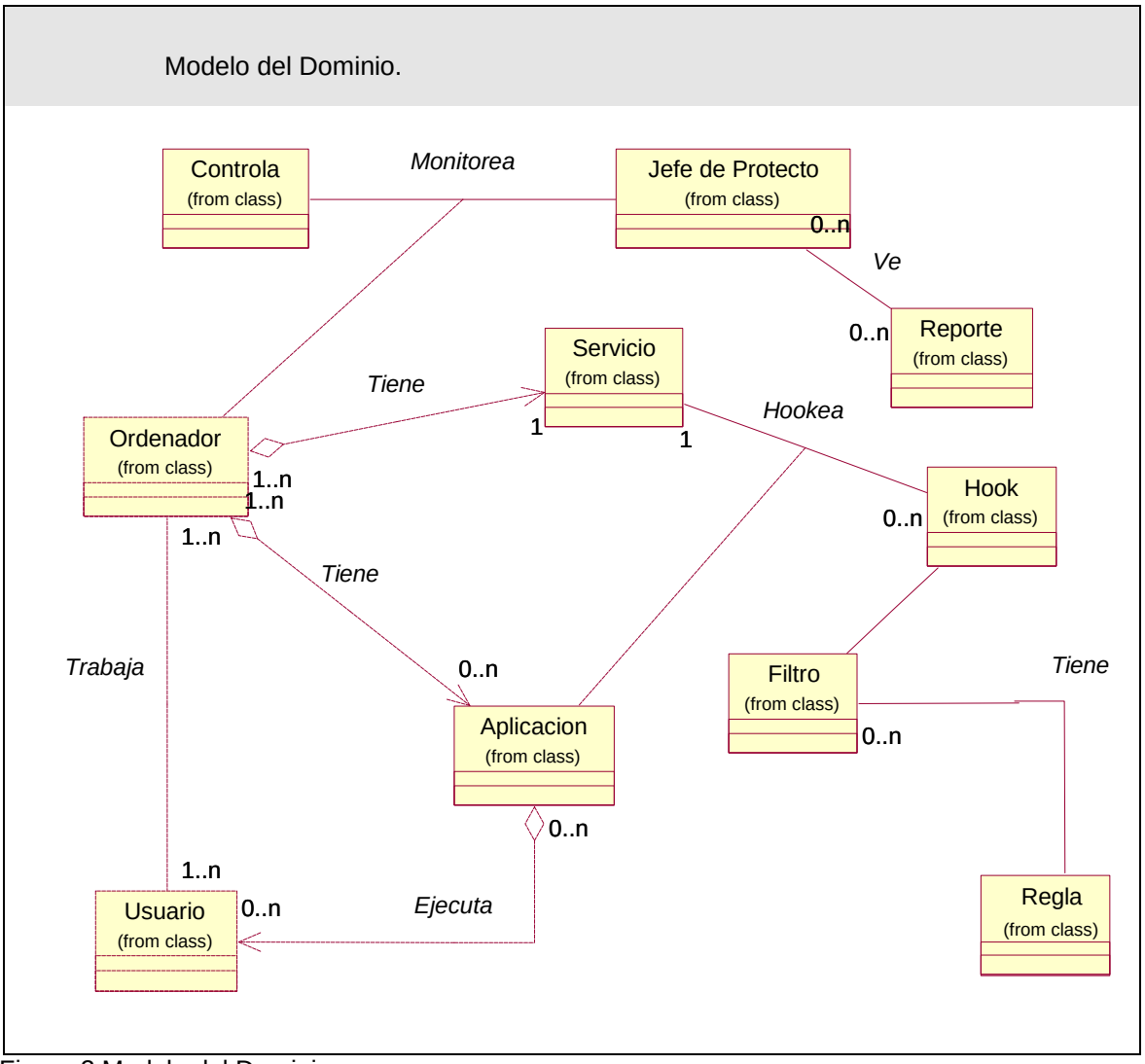

Figura 2 Modelo del Dominio

2.7.2 GLOSARIO DE TÉRMINOS DEL MODELO DE DOMINIO

Controlar: Acción por aquel o aquellos individuos que se encuentren como máximos responsables del proyecto.

Jefe de Proyecto: Individuo que tiene derecho, actual o futuro para tomar decisiones sobre las tareas en el proyecto.

Ordenador: Objeto o estación de trabaja a utilizar por los miembros del proyectos, las cuales serán objeto de chequeo o control.

Servicio:

Reporte: Conjunto de datos organizados que serán mostrados a aquellos individuos los cuales tengan la responsabilidad de controlar la ejecución de un proyecto de software.

Hook*:* Acción que establecería la ejecución de actividades en determinados eventos del sistema.

Usuario: Individuo que será objeto de chequeo o control durante su jornada de trabajo o tiempo de máquina para el desarrollo.

Aplicación: Tareas o programas que son ejecutados por un individuo en una ordenador durante un período de tiempo.

Filtro: Operación que permitirá extraer del flujo de datos, toda aquella información que no le interese realmente al jefe de proyecto.

Reglas: Conjunto de predicados o precondiciones que conformarían todos los filtro.

2.8 RELACIÓN DE LOS REQUERIMIENTOS

2.8.1 DEPENDENCIAS Y RELACIONES CON OTRAS APLICACIONES.

La aplicación dependerá en gran medida de los gestores de base datos SQL Server 2000 y Microsoft Office Access, del framework de la plataforma .NET, así como del correcto funcionamiento del sistema operativo Windows.

2.8.2 REQUERIMIENTOS FUNCIONALES

RF 1. Autentificar usuario.

RF 2. Adicionar una nueva PC para monitorear.

RF 3. Establecer conexión con el servidor.

RF 4. Instalar servicio.

RF 5. Hookear procesos del esclavo.

RF 6. Obtener información de cada proceso del usuario.

RF 7. Almacenar información en la base de datos local.

RF 8. Actualizar información en la base de datos local.

RF 9. Atender peticiones del cliente.

RF 10. Enviar la información de los procesos solicitada al cliente.

RF 11. Limpiar base de datos local.

RF 12. Almacenar la información de los procesos en la base de datos del sistema.

RF 13. Mostrar reportes según determinados filtros (IP, Usuario, Procesos).

RF 14. Eliminar una PC que se estaba monitoreando.

RF 15. Configurar la aplicación para iniciar su funcionamiento.

2.8.3 REQUERIMIENTOS NO FUNCIONALES

2.8.4 REQUERIMIETOS DE INTERFAZ O APARIENCIA

La aplicación propuesta poseerá una interfaz sencilla, intuitiva y amigable para el usuario a quién va dirigida, o sea los jefes de proyecto de la universidad. En general, fácil de usar y agradable a la vista del usuario.

2.8.5 REQUISITO DE USABIIDAD

Los jefes de proyectos que serán los usuarios de esta aplicación, no necesitarán contar con una preparación previa para operar la misma, pues la mayoría de las operaciones se llevan a cabo de forma automática. Las operaciones no automatizadas tampoco requerirán de la intervención de un experto pues en su mayoría se trata de opciones de configuración y visualización de reportes fácilmente entendibles por cualquier persona.

2.8.6 REQUISITO DE RENDIMIENTO

El sistema optimizará la utilización de los recursos de las estaciones de trabajo. Además de optimizar la utilización de la red para la comunicación cliente-servidor para la entrega de las peticiones de información de uno hacia otro.

2.8.7 REQUISITOS DE SOPRTE

El sistema estará integrado, de manera que se mantenga la colaboración entre los subsistemas que lo componen. El medio de almacenamiento estará soportado por el sistema gestor de base de datos SQL Server 2000 para posibilitar mejores resultados de rendimientos.

2.8.8 REQUISITOS DE SEGURIDAD

Garantizar e identificar con certeza que la información sea accedida y modificada por el personal debidamente autorizado para manipular la misma. Además asegurar la manipulación de la información correspondiente para mantener su integridad.

2.9 MODELO DE CASO DE USO DEL SISTEMA

2.9.1 DEFINICIÓN DE LOS ACTORES DEL SISTEMA

Actor	Descripción
Jefe de Proyecto.	Representa un usuario que juega el rol de jefe de proyecto, que a su vez es el único autorizado a modificar información que se genera del flujo de trabajo de los estudiantes.

Tabla 1 Definición del actor del sistema.

2.9.2 DESCRIPCIÓN DE LOS CASOS DE USO

Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto.

CU - 1	Autentificarse
Actor	Jefe de Proyecto.
Descripción	El usuario introduce los datos personales de acceso para ingresar al sistema, el sistema verifica la validez de los datos; de ser correctos se

	le permite acceder al sistema de lo contrario lo envía a la pantalla de autentificación.
Referencia	RF 1

Tabla 2 Descripción del Caso de Uso Autentificarse.

CU - 2	Actualizar Información.
Actor	Jefe de Proyecto.
Descripción	El jefe de proyecto solicita actualizar la información de los ordenadores que se están controlando, este se conecta a cada uno de los ordenadores o al ordenador seleccionado y actualiza toda la información de las actividades detectadas por el sistema.
Referencia	RF 10, RF 12

Tabla 3 Descripción del Caso de Uso Actualizar Información.

CU - 3	Gestionar Reportes.
Actor	Jefe de Proyecto.
Descripción	El jefe de proyecto solicita gestionar reportes, el sistema genera un reporte con toda la información necesaria del ordenador o el usuario seleccionado.
Referencia	RF 13

Tabla 4 Descripción del Caso de Uso Gestionar Información.

CU - 4	Gestionar Ordenador.
Actor	Jefe de Proyecto.
Descripción	El jefe de proyecto selecciona una de las opciones a ejecutar, ya sea adicionar un nuevo ordenador a la red de control o eliminar uno

	existente.
Referencia	

Tabla 5 Descripción del Caso de Uso Gestionar Control de Ordenadores.

CU - 5	Gestionar Información.
Actor	Sistema Operativo.
Descripción	Cuando un usuario interactúa con la computadora el sistema operativo crea, termina o cambia el estado de determinados procesos, la aplicación detecta cualquiera de estos eventos y captura la información referente a cada proceso que haya sido creado, modificado o destruido. Esta información se almacena en una base de datos local para posteriormente enviarla a la aplicación cliente y almacenar la misma en la base de datos general del sistema.
Referencia	RF 5, RF 6, RF 7, RF 8, RF 9, RF 11

CU - 5	Configurar Aplicación.
Actor	Jefe de Proyecto.
Descripción	El jefe de proyecto creará el usuario para acceder al sistema, además de configurar la conexión a la base de datos con que cuenta el mismo, también podrá crear nuevos usuarios, y cambiar su contraseña.
Referencia	RF 15

2.9.3 DIAGRAMA DE CASO DE USO DEL SISTEMA A AUTOMATIZAR

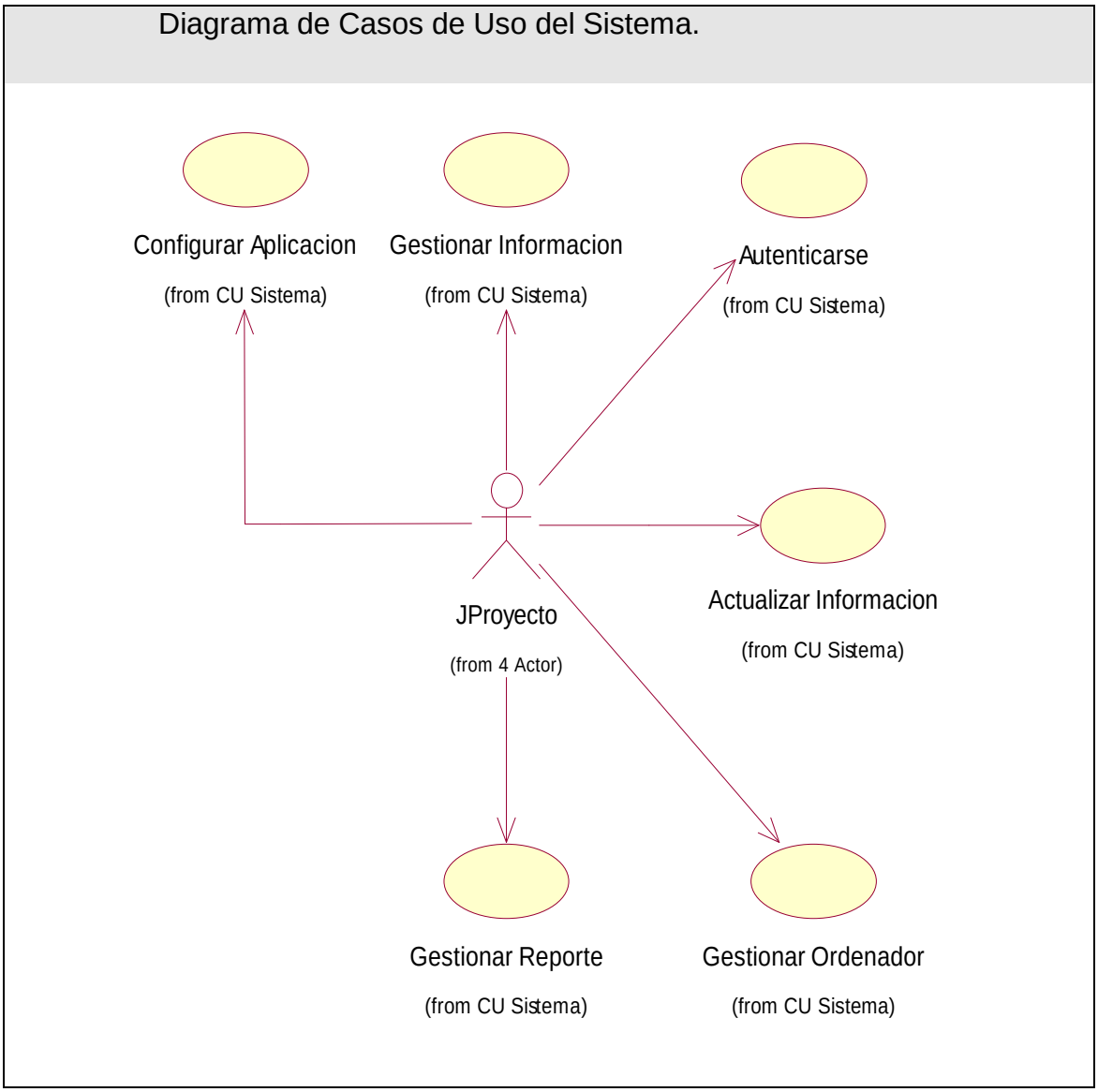

Figura 3 Diagrama de Casos de Uso del Sistema.

2.9.4 DESCRIPCIÓN DE LOS CASOS DE USO DEL SISTEMA

Nombre: Caso de Uso Autentificarse.	
Propósito	Permitir acceder al sistema solo al personal autorizado.

Actores	Jefe de Proyecto
Resumen: El Jefe de Proyecto ingresa sus datos personales y el sistema verifica la validez de los mismos, de ser correctos se le da acceso a las funcionalidades, en caso contrario se solicitan nuevamente sus datos personales (usuario y contraseña).	
Referencias	RF 1

Acción del actor	Respuesta del sistema
El jefe de proyecto ejecuta la aplicación. 3. El jefe de proyecto introduce sus credenciales. 5. En caso de ser correctos los datos se le permite al jefe de proyecto acceder al sistema.	2. El sistema le pide las credenciales al jefe de proyecto. 4. El sistema verifica si los datos son correctos. 6. El sistema inicia todas sus funcionalidades.
Flujo alternativo Si no son válidos los datos proporcionados por el jefe de proyecto se le vuelven a pedir para su posterior verificación.	
Acción del actor	Respuesta del sistema
1. El jefe de proyecto vuelve a ingresar sus datos personales. 3. De ser correctos se le da acceso al sistema.	2. El sistema vuelve a verificar la validez de los datos proporcionados por el jefe de proyecto. 4. El sistema inicia todas sus funcionalidades.
Puntos de extensión.	

Nombre: Caso de Uso Gestionar Información.	
Propósito	Actualizar la información referente a cada proceso en la base de datos local, esto se produce una vez finalizado el proceso.
Actores Sistema Operativo.	

Resumen: Cada vez que se crea un nuevo proceso se almacena la información referente a este en una base de datos local, después cuando el proceso termina se vuelve a actualizar toda la información en la base de datos para posteriormente ser enviada al cliente.	
Referencias	RF 8, RF 11.
Acción del actor	Respuesta del sistema
1. Iniciar un nuevo proceso en el ordenador. 3. Terminar la ejecución de un proceso determinado.	2. Monitorear esos procesos y obtener la información referente a los mismos. 4. Actualizar la información referente al proceso que acaba de terminar. 5. El establece comunicación con el cliente. 6. El servicio envía la información de los procesos al cliente y este la almacena en la base de datos.
Flujo alternativo El sistema operativo no inicia ningún proceso.	
Acción del actor	Respuesta del sistema
1. El sistema operativo no crea ningún proceso nuevo.	2. La aplicación se mantiene a la espera de un nuevo evento en el sistema.
Puntos de extensión.	

Nombre: Caso de Uso Gestionar Reporte.	
Propósito	Mostrarle al jefe del proyecto un reporte con la información que este desee revisar atendiendo a una serie de filtros tales como #ip, #mac, usuario, nombre del proceso, etc.
Actores Jefe de Proyecto	
Resumen: El jefe de proyecto una vez registrado en el sistema recibirá un reporte con la información solicitada, haciendo uso de los filtros antes mencionados.	

Referencias	RF 13
Acción del actor	Respuesta del sistema
1. El jefe de proyecto solicita un reporte. 3. El jefe de proyecto guarda el reporte en caso que lo desee.	2. Mostar el reporte según las especificaciones que haya hecho el jefe de proyecto.
Flujo alternativo En caso de contar con la información solicitada el sistema emitirá un mensaje al jefe de proyecto avisándole de la situación.	
Acción del actor	Respuesta del sistema
1. El jefe de proyecto solicita un nuevo reporte. 3. El jefe de proyecto guarda el reporte en caso que lo desee.	2. Mostrar el reporte según las especificaciones que haya hecho el jefe de proyecto.
Puntos de extensión.	

Nombre: Caso de Uso Gestionar Ordenador	
Propósito	Permitir adicionar nuevas computadoras para monitorear, así como eliminar algunas que ya no requieran ser monitoreadas.
Actores Jefe de Proyecto	
Resumen: Una vez iniciada la aplicación el jefe de proyecto podrá adicionar un nuevo ordenador para ser monitoreado, así como eliminar alguno que ya no necesite monitorear. En todo momento el jefe de proyecto podrá ver la lista de ordenadores que actualmente está monitoreando.	
Referencias	RF 2, RF 14, CU 6, CU 7
Acción del actor	Respuesta del sistema
1. El jefe de proyecto introduce los datos del ordenador que desea adicionar o eliminar.	2. El sistema elimina o añade el ordenador especificado.
Flujo alternativo En caso de no ser válida la información introducida por el jefe de proyecto el sistema emitirá un mensaje avisándole de la situación.	

Acción del actor	Respuesta del sistema
Puntos de extensión.	

Nombre: Caso de Uso Actualizar Información.		
Propósito	El objetivo es actualizar correctamente la información referente a cada proceso proveniente de los distintos ordenadores que estemos monitoreando.	
Actores Jefe de Proyecto.		
Resumen: El jefe de proyecto seleccionará el ordenador que desea actualizar en la base de datos solicitándole la información referente al mismo. De esta forma se establece la comunicación con el servidor y este envía la información solicitada.		
Referencias	RF 12, CU 2.	
Acción del actor		Respuesta del sistema
1. El jefe de proyecto selecciona el ordenador que desea actualizar. 3. El jefe de proyecto actualiza la información en la base de datos.		2. El sistema establece comunicación con dicho servidor. 4. El sistema ejecuta la acción especificada.
Flujo alternativo		
Acción del actor		Respuesta del sistema
Puntos de extensión.		

Nombre: Caso de Uso Configurar Aplicación.	
Propósito	El objetivo es configurar la aplicación para que funcione correctamente.
Actores Jefe de Proyecto.	
Resumen: El jefe de proyecto creará el usuario para acceder al sistema, además de configurar la conexión a la base de datos con que cuenta el mismo, además también podrá crear nuevos usuarios, y cambiar su contraseña.	
Referencias	
Acción del actor	Respuesta del sistema
1. El jefe de proyecto selecciona la opción que desea configurar. 3. El jefe de proyecto configura el sistema.	2. El sistema le mostrará la interfaz correspondiente. 4. El sistema comenzará a funcionar.
Flujo alternativo	
Acción del actor	Respuesta del sistema
Puntos de extensión.	

2.10 CONCLUSIONES

En este capítulo se trataron los aspectos del problema, la situación polémica que da origen al desarrollo de la solución, el objeto de automatización, una breve descripción de la solución del problema, los requerimientos funcionales y no funcionales que se deben cumplir para el desarrollo de la aplicación y el diagrama de los casos de uso del sistema.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

3.1 INTRODUCIÓN

El análisis forma parte del proceso de desarrollo de software, cuyo propósito primario es diseñar la arquitectura y el modelo del dominio del problema. Así como establecer las primeras pautas para su implementación, durante el análisis, se analizan los requisitos que se describieron en la captura de los mismos, refinándolos y estructurándolos. El objetivo de hacer esto es conseguir una mejor comprensión de las funcionalidades, de forma tal que el sistema sea fácil de mantener y que ayude a estructurarlo. Proporciona además una estructura centrada en el mantenimiento, en aspectos tales como la flexibilidad ante los cambios y la reutilización.

El diseño es el centro de atención final de la fase de elaboración y el comienzo de las iteraciones de construcción, contribuye a una arquitectura estable y sólida, y a crear un plano del modelo de implementación. Más tarde durante la fase de construcción, cuando la arquitectura es estable y los requisitos están bien definidos, el centro de atención se desplaza a la implementación.

En el presente capítulo se procede a representar la realización de los casos de uso, los diagramas de secuencia y la descripción de los contratos correspondientes a cada acción de dichos diagramas, utilizando para su modelado el Lenguaje Unificado de Modelación (UML).

3.2 REALIZACIÓN DE LOS CASO DE USO

Una realización de caso de uso de diseño es una colaboración en el modelo de diseño que describe como se realiza un caso de uso especifico, y como se ejecuta, en términos de clases de diseño y sus objetos. Tiene una descripción de flujo de eventos textual, diagramas de clases que muestran sus clases de diseño participantes, y diagramas de interacción que muestran la realización de un flujo o escenario concreto de un caso de uso en términos de interacción entre objetos del diseño. Existen dos tipos de diagramas de interacción: el diagrama de colaboración y el diagrama de secuencia.

El diagrama de secuencia es más adecuado para observar la perspectiva cronológica de las interacciones, muestra la secuencia explícita de mensajes y son mejores para especificaciones de tiempo real y para escenarios complejos. El diagrama de colaboración ofrece una mejor visión espacial mostrando los enlaces de comunicación entre objetos, muestra las relaciones entre objetos y son mejores para comprender todos los efectos que tiene un objeto y para el diseño de procedimientos. El diagrama de colaboración puede obtenerse automáticamente a partir del correspondiente diagrama de secuencia (o viceversa). En este caso se realizarán los diagramas de colaboración, además de los diagramas de clases para cada caso de uso. El diagrama de clases es una vista gráfica de un modelo estructural estático. Describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación. Además de las asociaciones y los atributos básicos incluye fundamentalmente los métodos de cada clase, la información sobre el tipo de los atributos, su visibilidad y la navegación entre los objetos.

3.3 DEFINICIÓN DE DISEÑO QUE SE APLICA

3.3.1 ARQUITECTURA BASE

Los patrones de diseño describen un problema que ocurre repetidas veces en algún contexto determinado de desarrollo de software, y entregan una buena solución ya probada. Esto ayuda a diseñar correctamente en menos tiempo, ayuda a construir problemas reutilizables y extensibles, y facilita la documentación. Los patrones enseñan a aplicar de manera eficaz que hacer con la herencia, el polimorfismo, y todas las ventajas que posee la POO.

Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno y describe también el núcleo de la solución al problema, de forma que puede utilizarse un millón de veces sin tener que hacer dos veces lo mismo.

Por otro lado los patrones de arquitectura: Expresan un paradigma fundamental para estructurar u organizar un sistema de software. Proporcionan un conjunto de

subsistemas o módulos predefinidos, con reglas y guías para organizar las relaciones entre ellos.

En la propuesta de solución se utilizó como patrón de arquitectura de software, el Modelo Vista Controlador (MVC), ya que este patrón divide la estructura de la aplicación cliente en tres clases distintas.

1. Modelo (o Estado): Gestiona el comportamiento y los datos de la aplicación, responde a las peticiones que realizan las vistas sobre su estado y permite su actualización normalmente desde el controlador. En este caso en la propuesta de solución, el modelo es el paquete de acciones.
2. Controlador: Interpreta las acciones del usuario, accediendo a las operaciones de negocio de la aplicación y modificando a partir de sus resultados el estado del modelo y la navegación entre vistas. En este caso dentro de la solución propuesta, el controlador es el gestor, el cual es el encargado de gestionar los tipos de conexiones que se utilizan, así como los métodos auxiliares de gestión de datos.
3. Vista: Muestra el estado al usuario de la aplicación, redirigiendo las acciones que realiza sobre la interfaz al controlador. En nuestro caso la vista, está conformada por los diferentes formularios de la aplicación.
4. Se hizo una adaptación del patrón Modelo Vista Controlador, basándose en las acciones no en las vistas, utilizando el patrón comando para implementar las acciones.

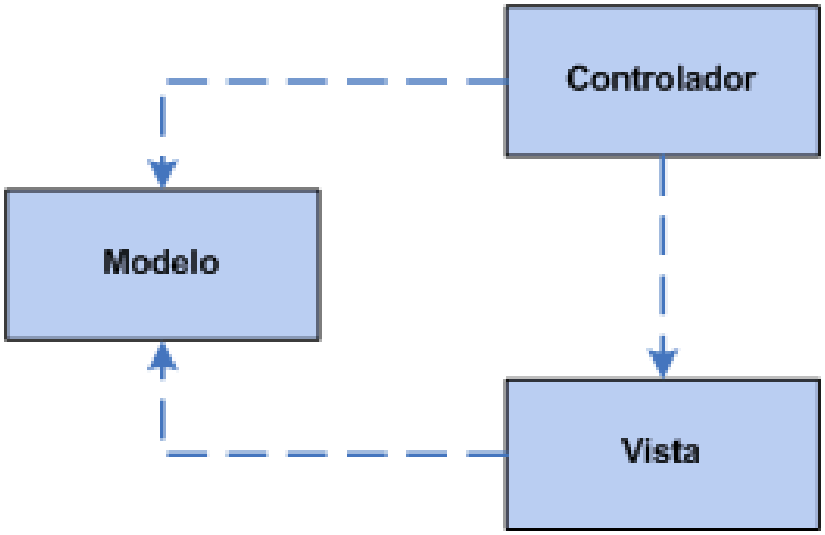

Figura 4 Esquema simplificado del patrón MVC

3.3.2 MODELO MULTICAPAS

El desarrollo de la solución responde a un modelo multicapas. Un ejemplo de la interacción de estas capas se muestra en la figura que se encuentra debajo.

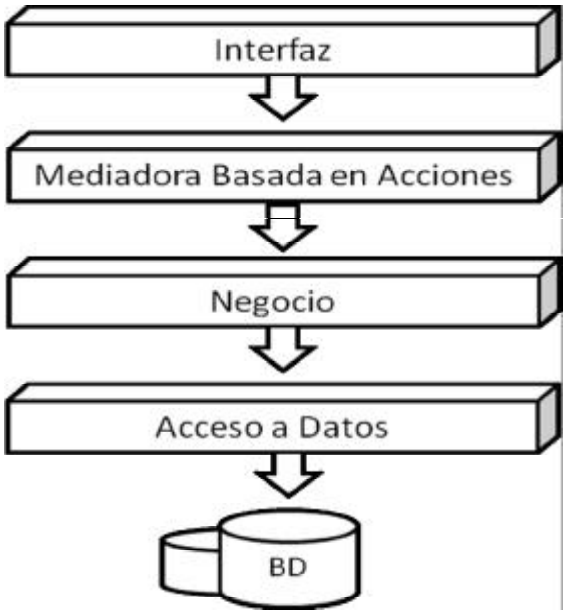

Figura 5 Modelo Multicapas

Los tipos y clases comunes son todos los tipos de datos que deben ser comunes a varias capas, por tanto deben ser vistos por varias capas. El sentido de las flechas indica que la capa origen conoce la entidad que está en el destino de la misma.

La capa de interfaz responde a un patrón estándar. Su uso se basa en la explotación de la componente interfaz que realmente provee el framework .NET facilitando el desarrollo del sistema.

La capa mediadora interfaz basada en acciones. Su uso se basa en la explotación del componente interfaz que provee el framework de .NET. Cada acción debe tener sentido semántico propio y completo. Deben ser atómicas. Las acciones básicamente definen un bloque de código reusable por varias operaciones sobre el sistema. Las acciones pueden estar asociadas a vistas del sistema. Cada una puede representar un estado del sistema que puede o no persistir. Una acción es una clase que representa precisamente la ejecución de alguna tarea concreta. Estas tareas no deben ser excesivamente complejas y la clase debe:

1. Heredar de la clase acción.
2. Opcionalmente tener una forma visual asociada.
3. Propiedades en función del objetivo específico de la misma.
4. Se le debe reescribir el método CrearForma con vistas a mostrar lo que se desee.

En el caso de la capa lógica de negocio, el diseño de esta capa depende directamente del negocio específico al que se refiera.

Por último está la capa de datos que es desarrollada por un equipo en común. Esta capa está compuesta por dos subcapas. La primera se corresponde con una capa compuesta por clases que constituyen factorías de las entidades del negocio que deben persistir. La segunda subcapa es la encargada de persistir los datos, todo el manejo de la información que requiera el negocio del problema en cuestión y la conexión con la base de datos.

3.4 ANÁLISIS

3.4.1 DEFINICIÓN DEL MODELO DEL ANÁLISIS

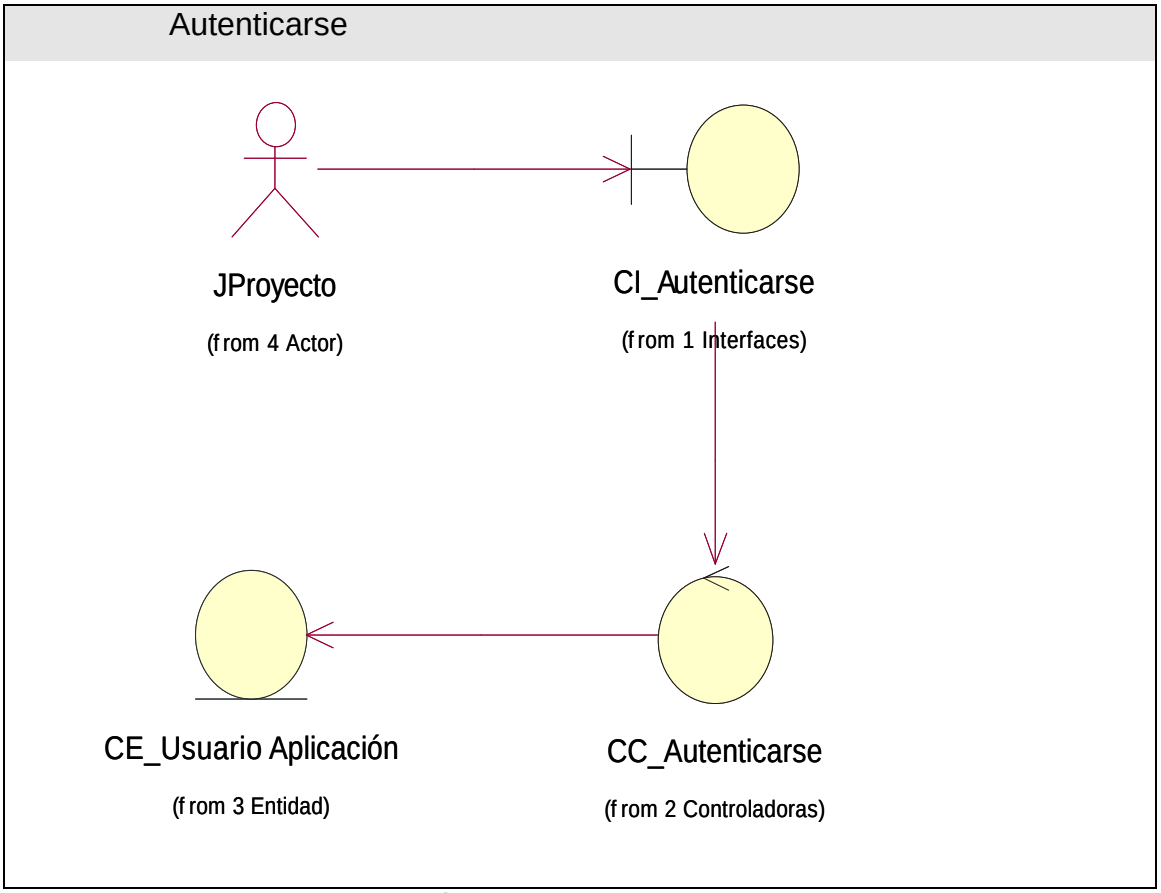

Figura 6 Diagrama de Clases del Análisis Autenticarse

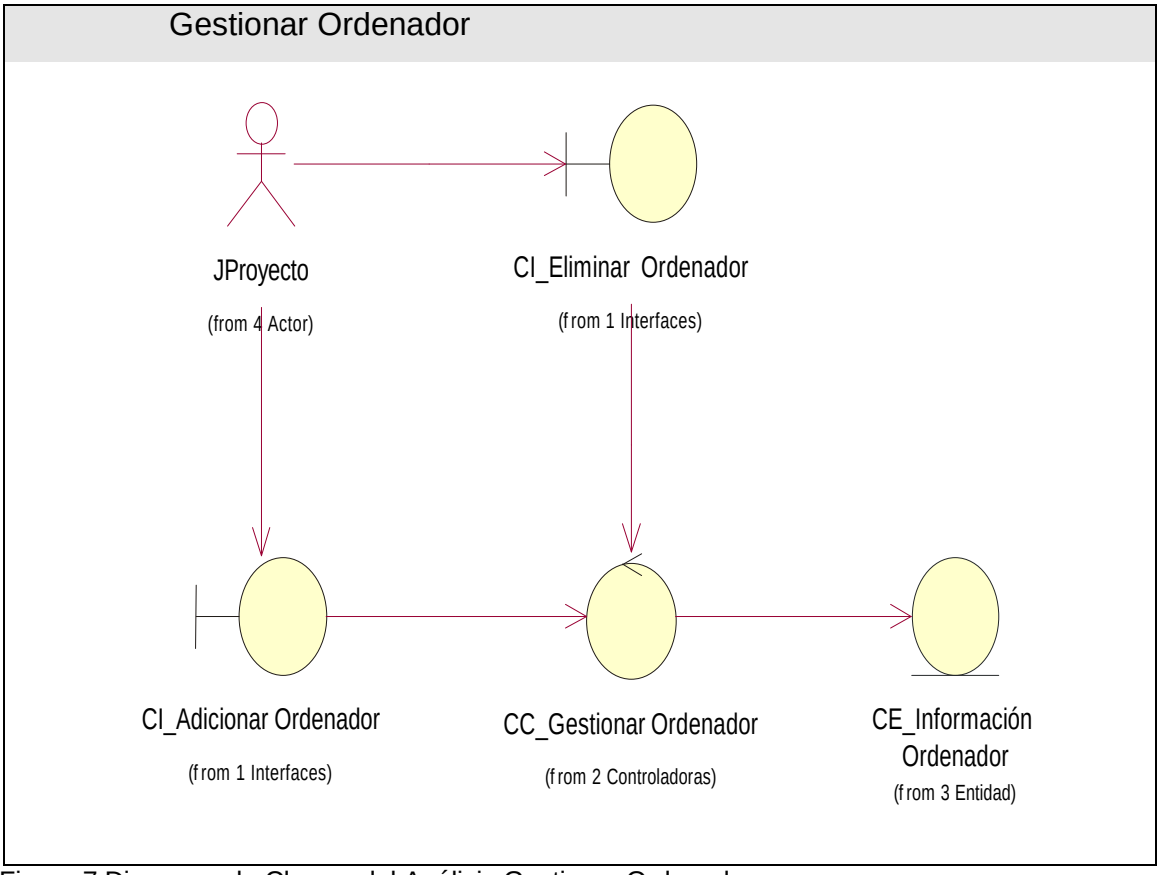

Figura 7 Diagrama de Clases del Análisis Gestionar Ordenador

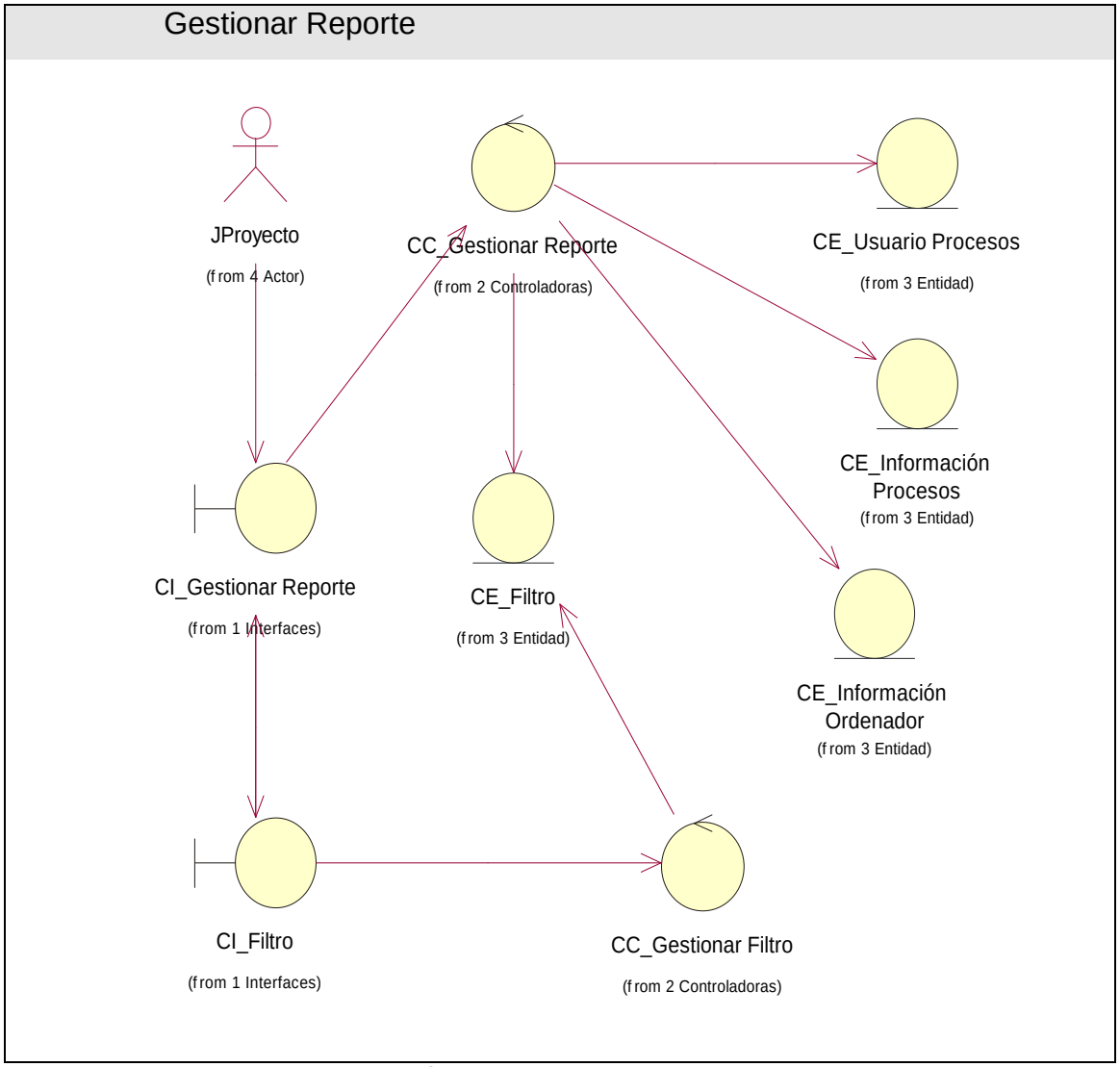

Figura 8 Diagrama de Clases del Análisis Gestionar Reporte

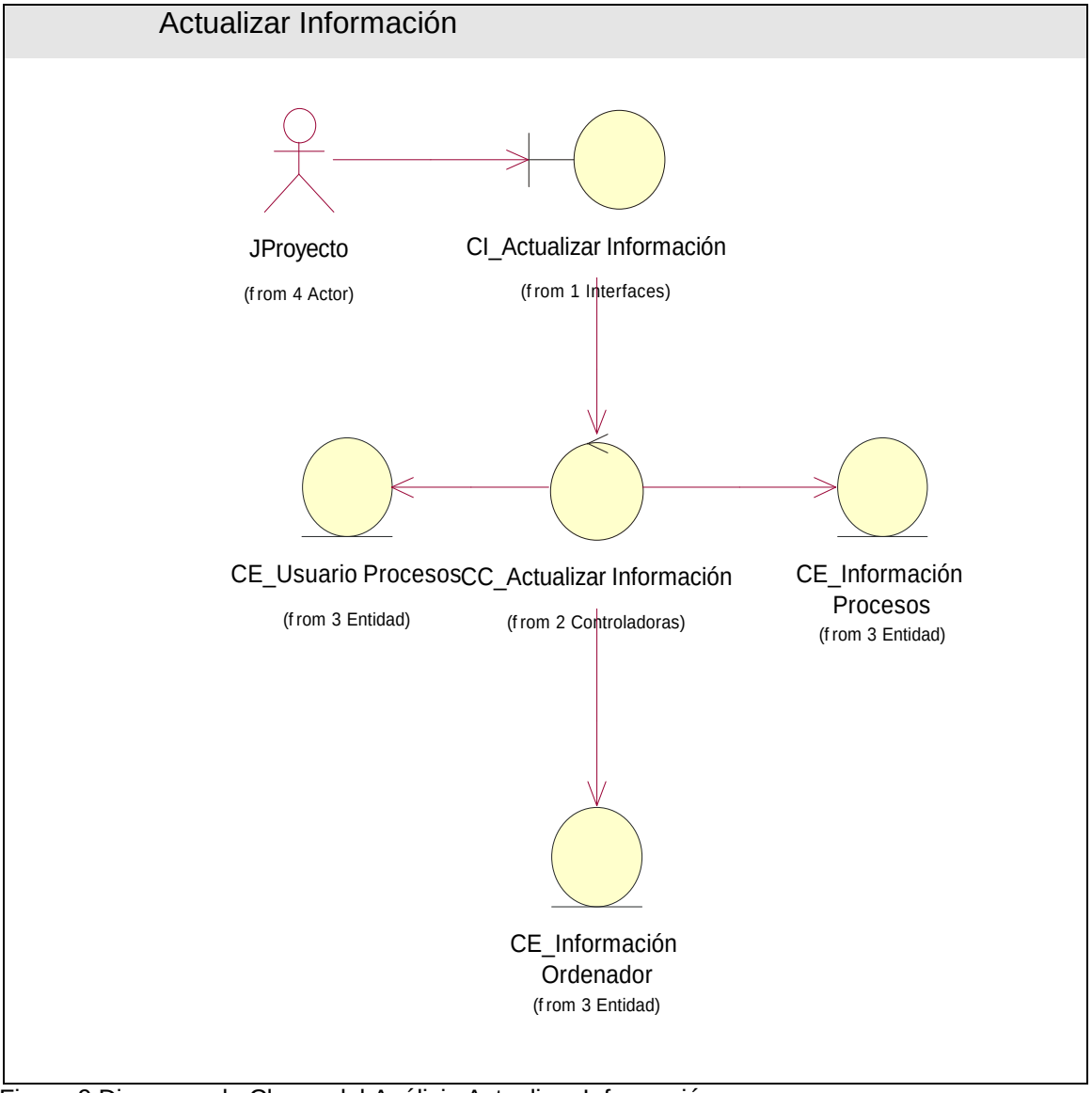

Figura 9 Diagrama de Clases del Análisis Actualizar Información

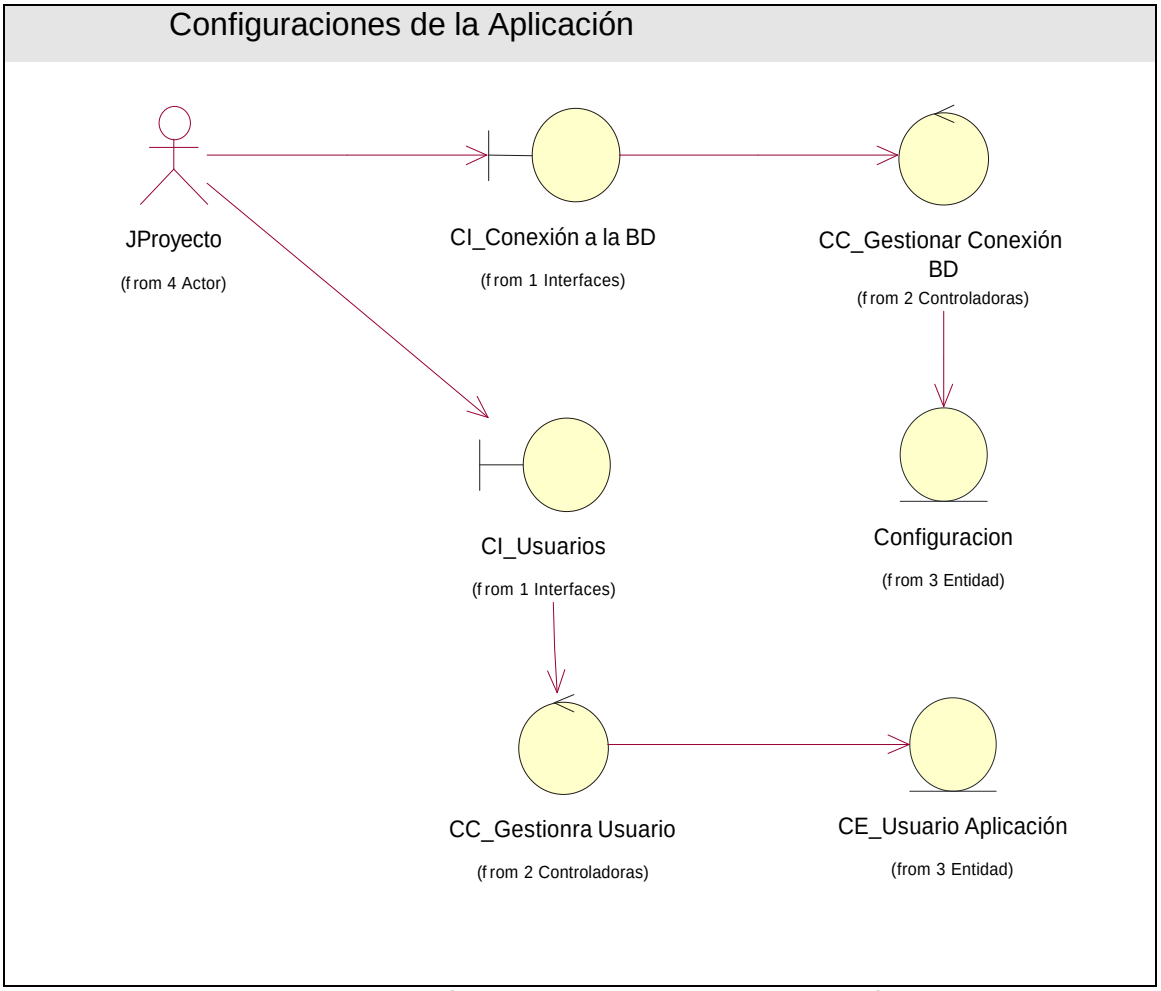

Figura 10 Diagrama de Clases del Análisis Configuraciones de la Aplicación

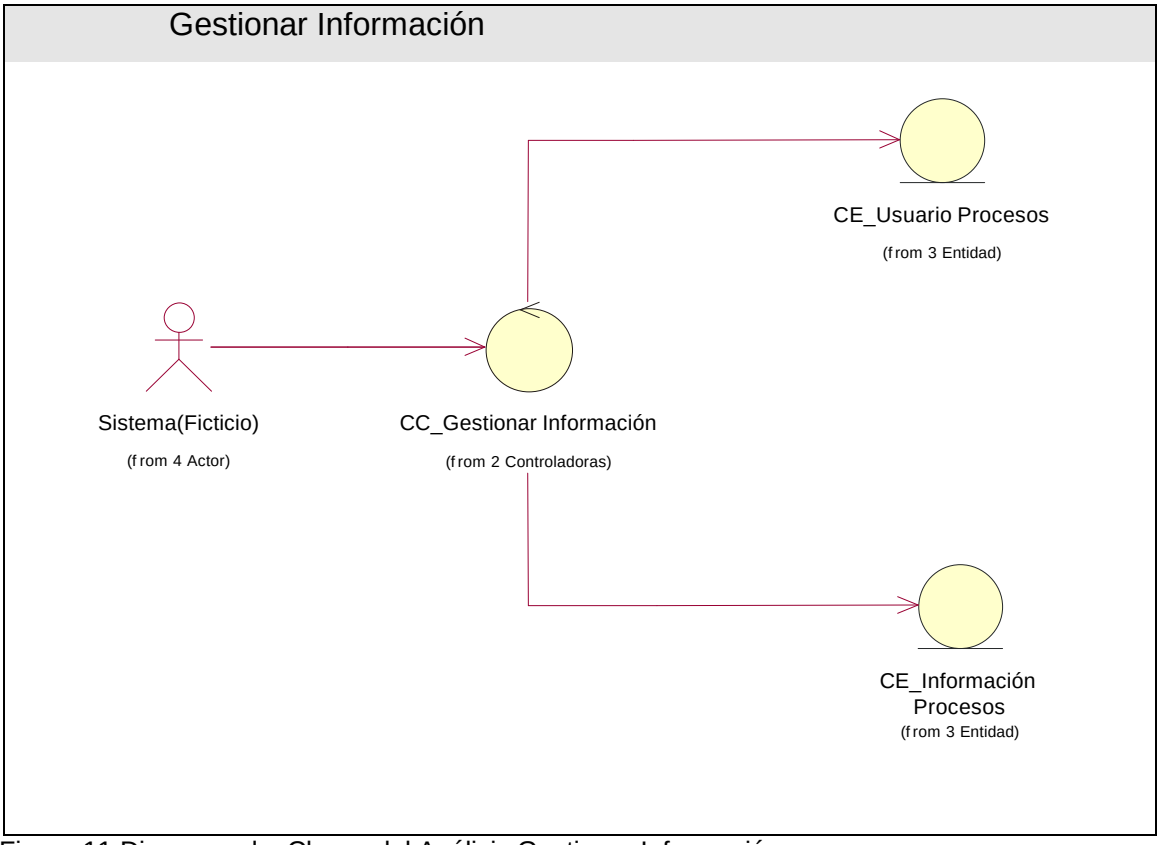

Figura 11 Diagrama de Clases del Análisis Gestionar Información

3.4.2 DIAGRAMAS DE INTERACCIÓN VER ANEXO 1

3.5 DISEÑO

3.6 DIAGRAMA DE CLASES DEL DISEÑO

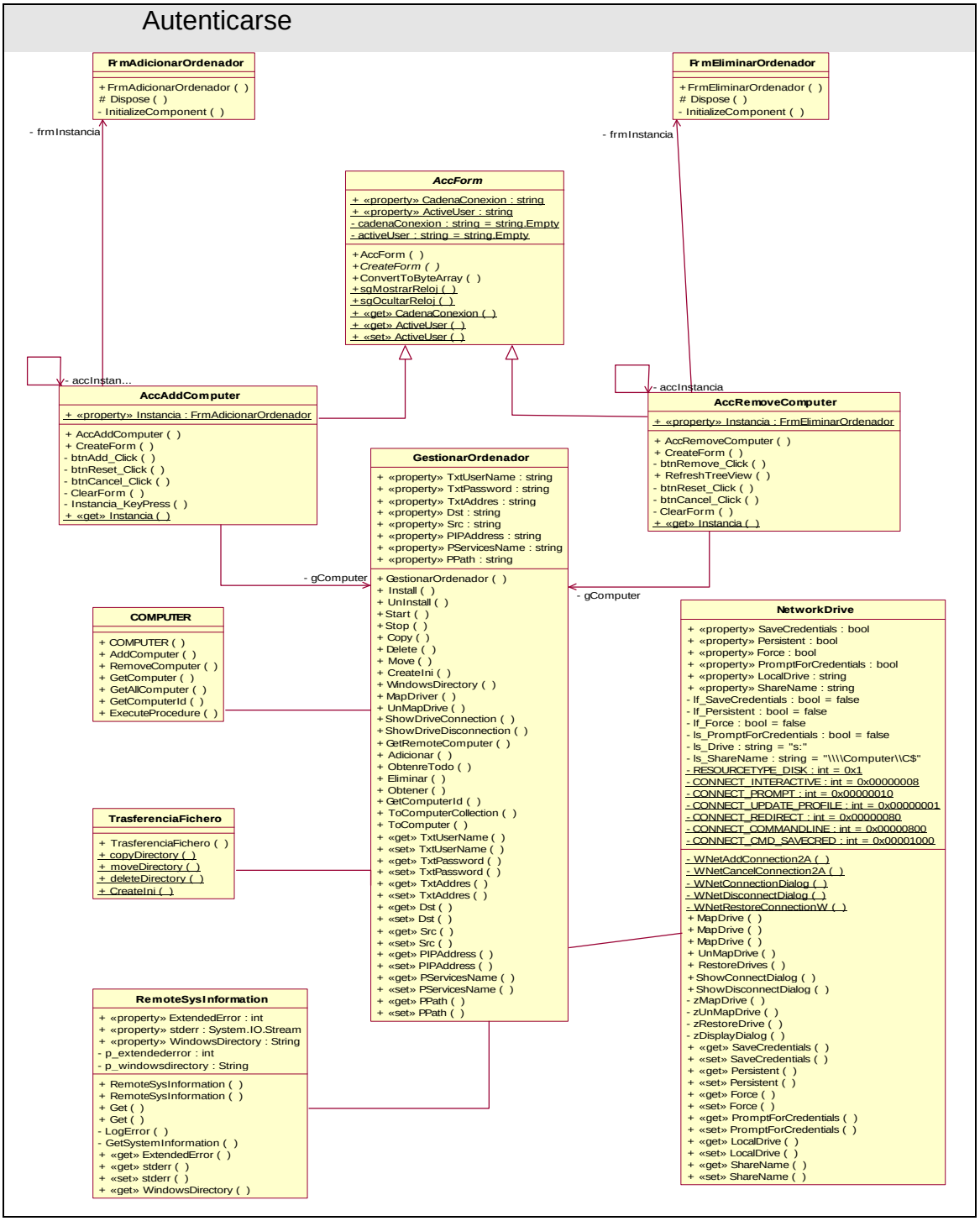

Figura 12 Diagrama de Clases del Diseño Autenticarse

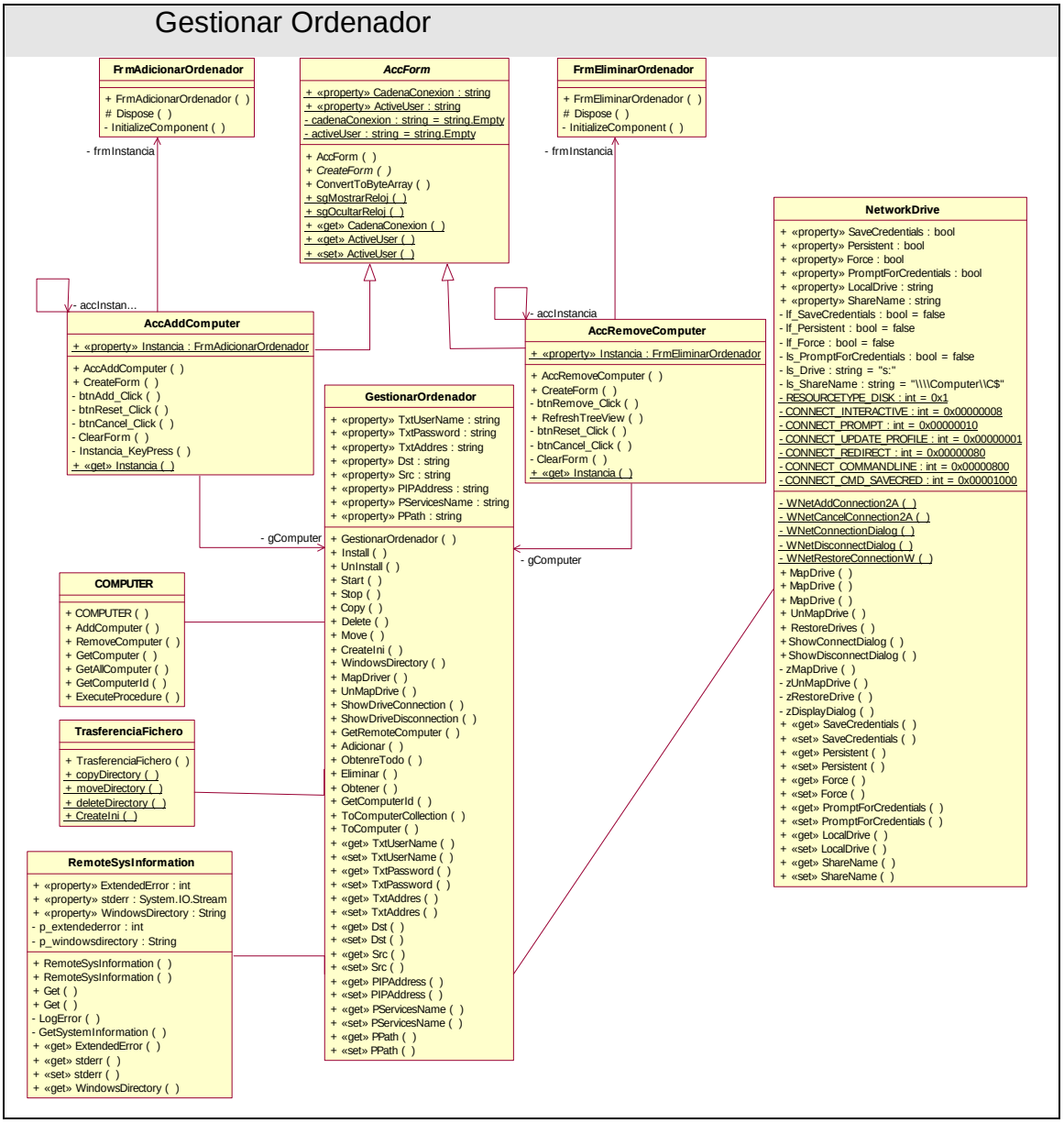

Figura 13 Diagrama de Clases del Diseño Gestionar Ordenador

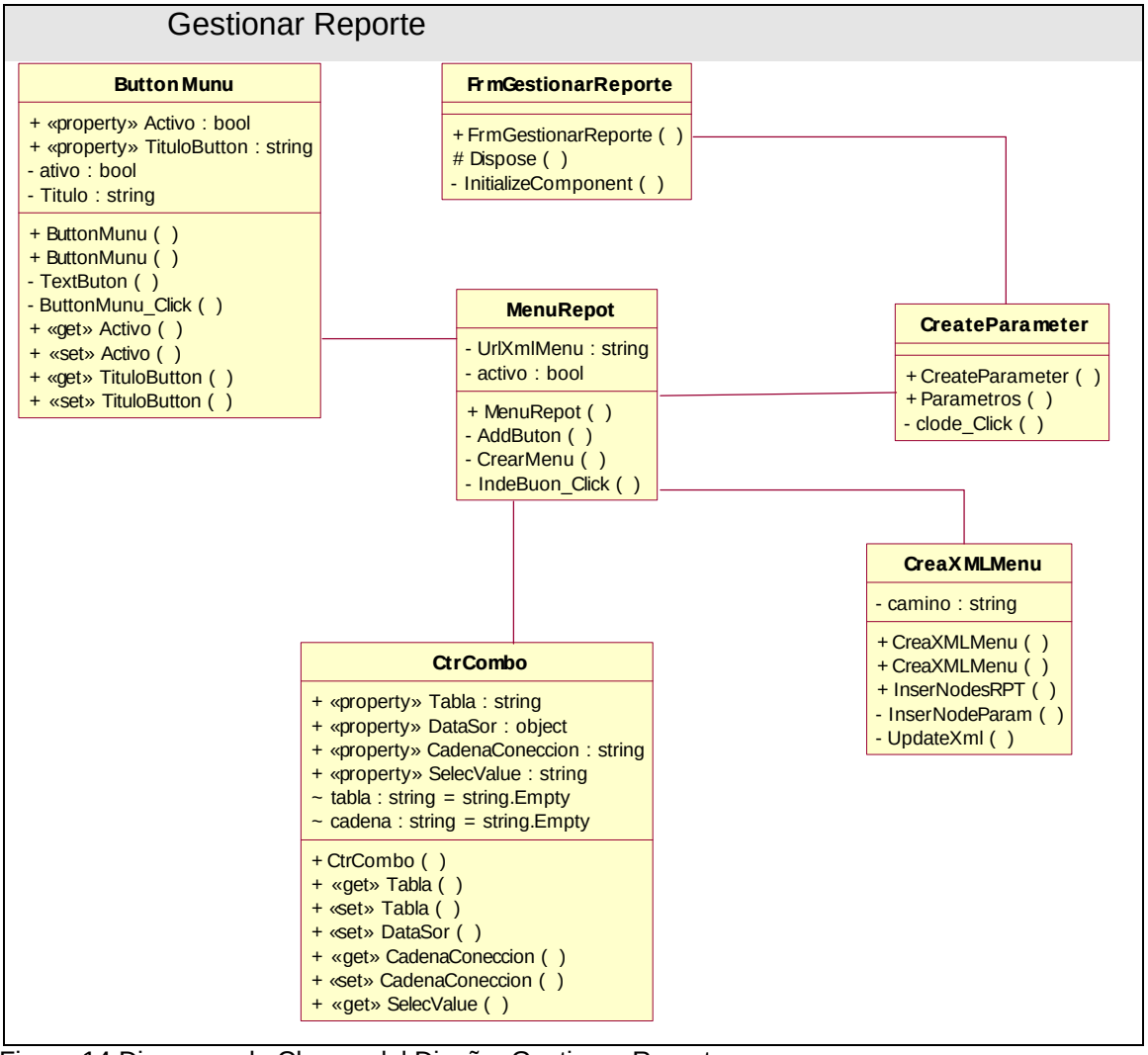

Figura 14 Diagrama de Clases del Diseño Gestionar Reporte

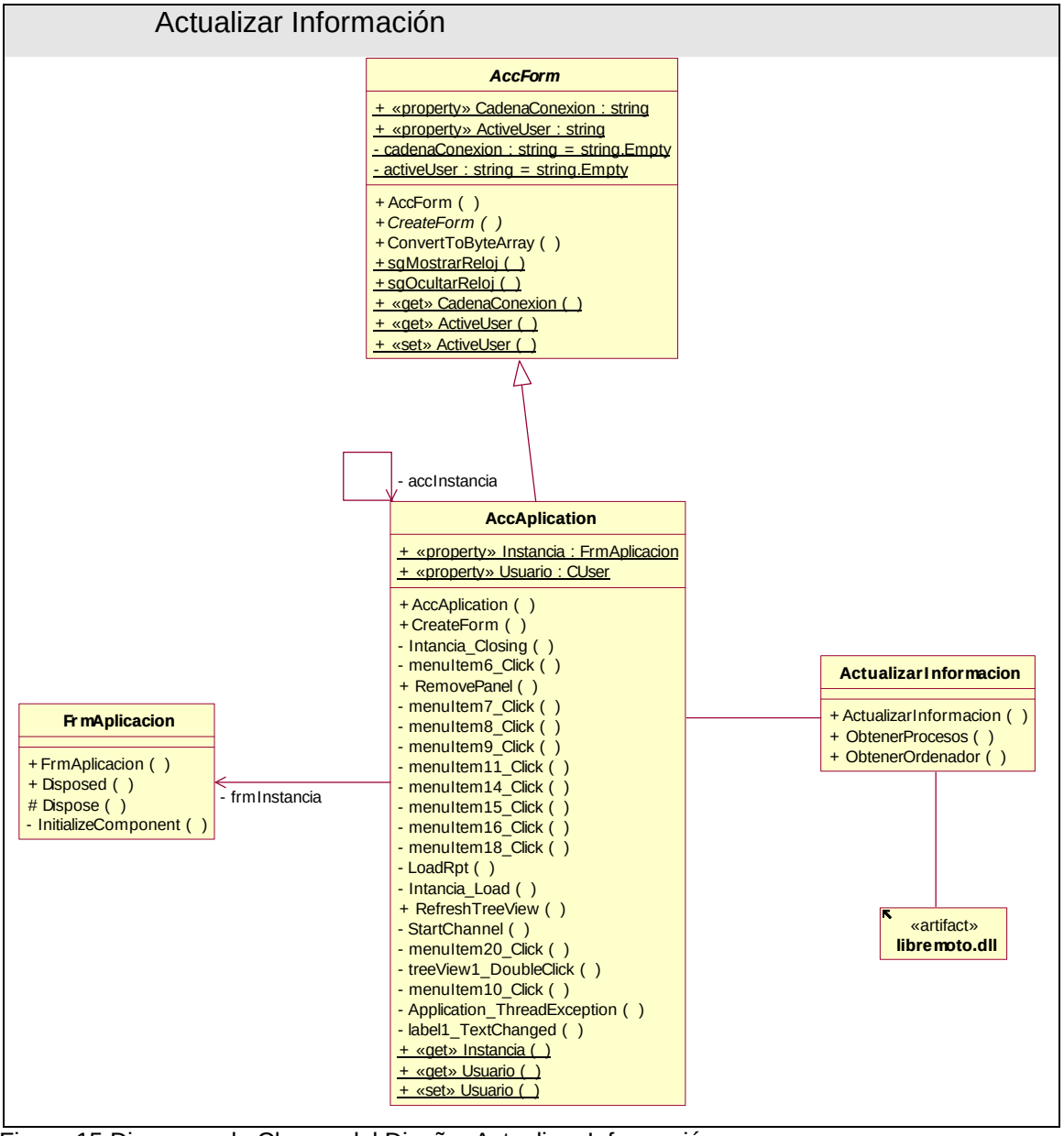

Figura 15 Diagrama de Clases del Diseño Actualizar Información

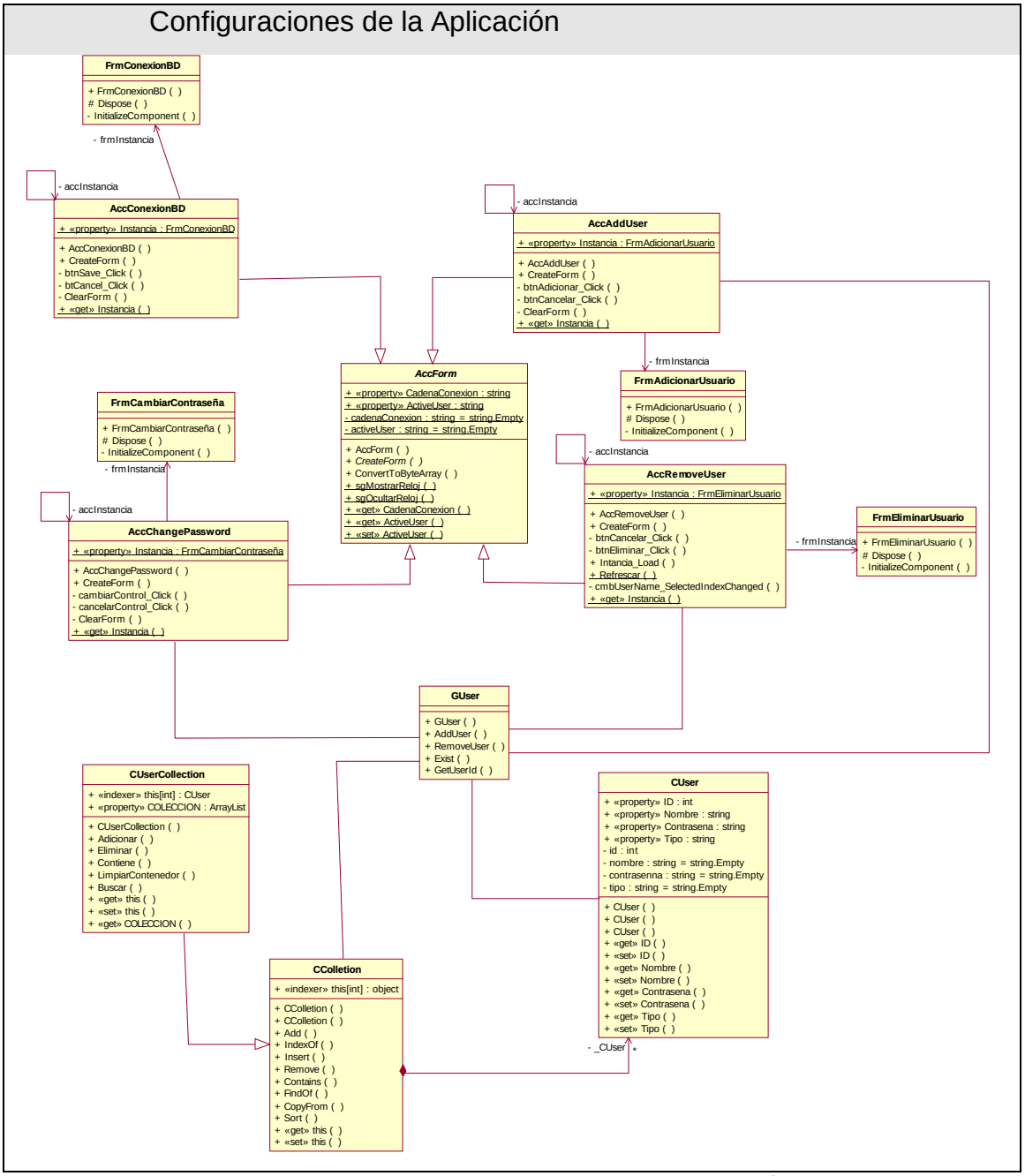

Figura 16 Diagrama de Clases del Diseño Configuraciones de la Aplicación

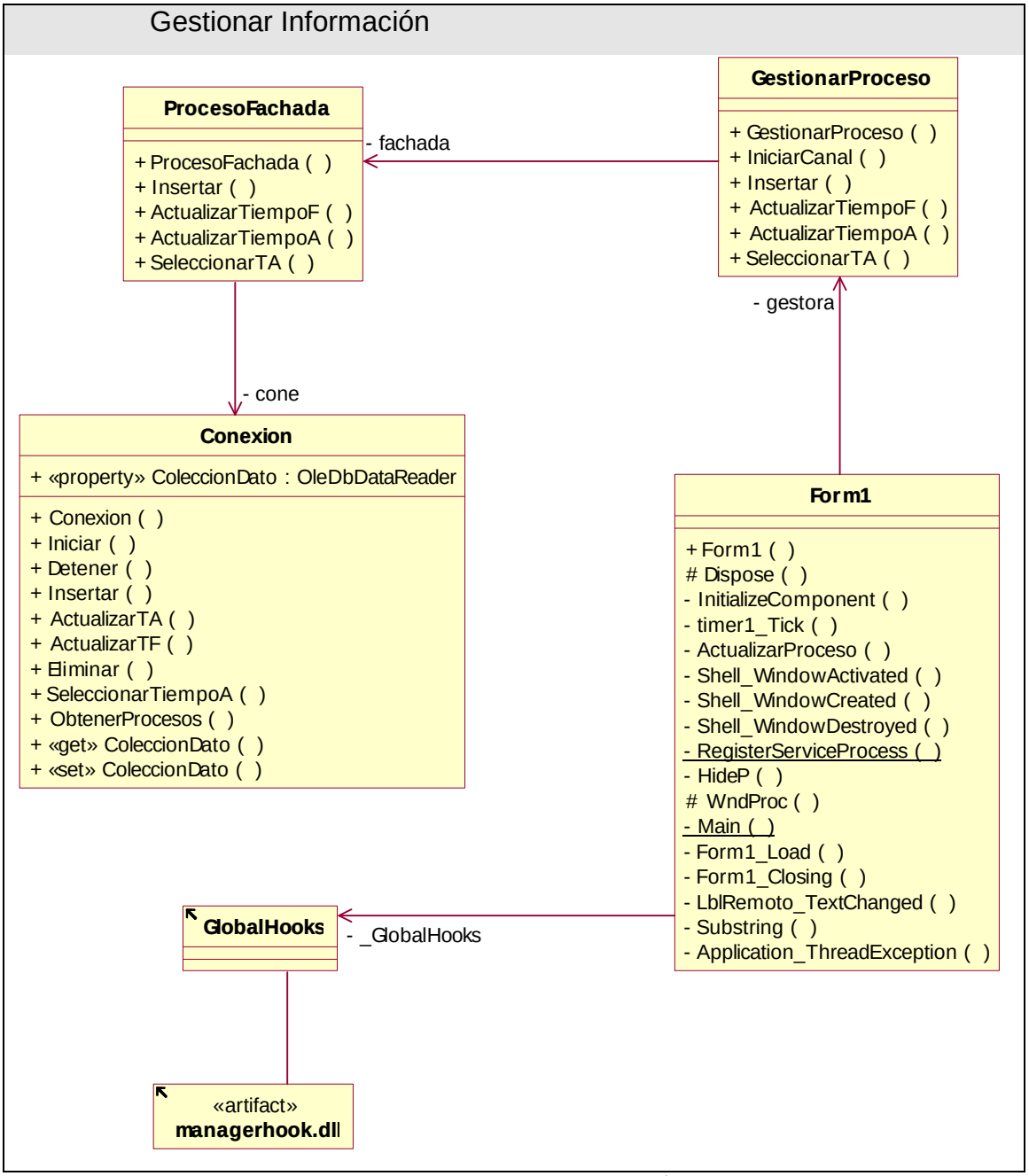

Figura 17 Diagrama de Clases del Diseño Gestionar Información

3.7 DISEÑO DE LA BASE DE DATOS

3.7.1 MODELO LÓGICO DE DATOS

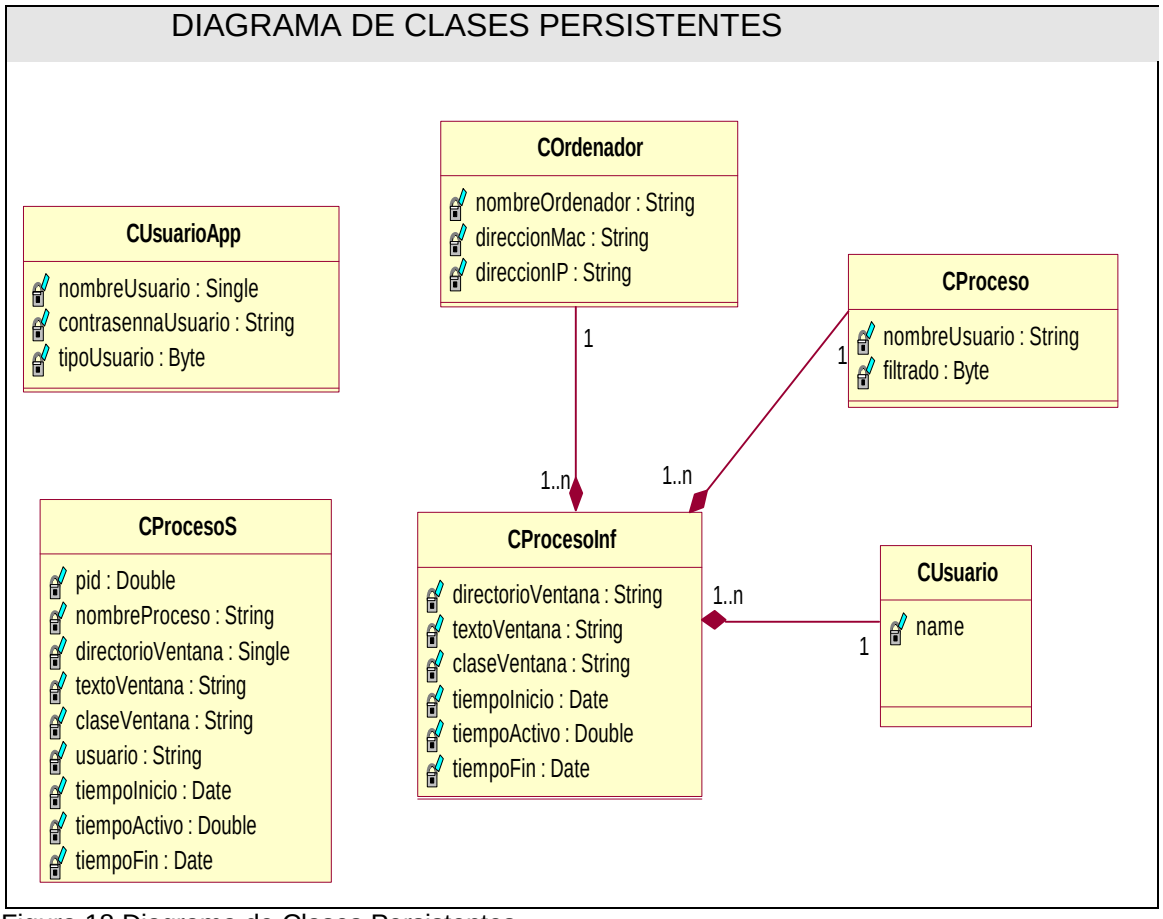

Figura 18 Diagrama de Clases Persistentes

3.7.2 MODELO FÍSICO DE DATOS

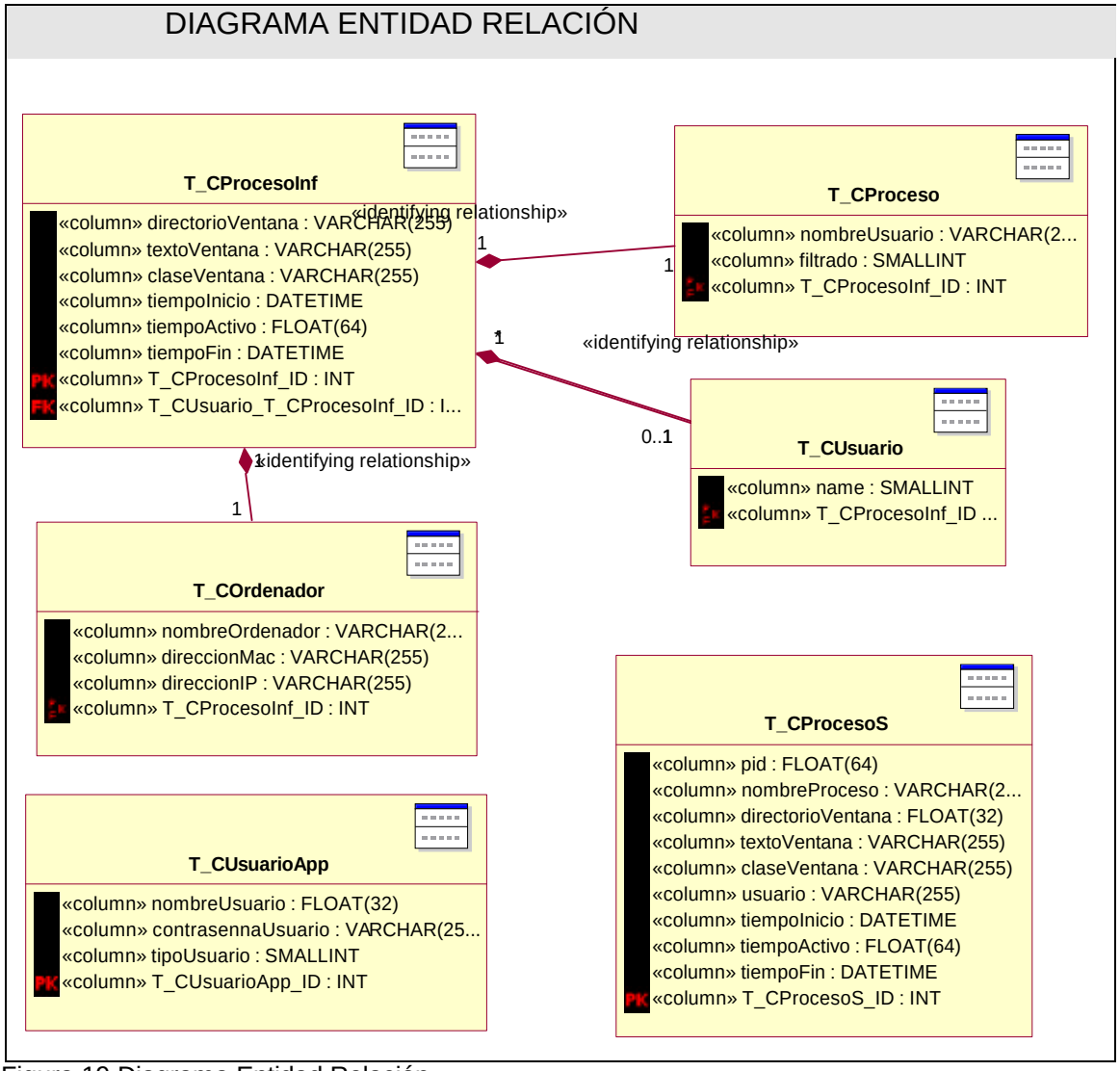

Figura 19 Diagrama Entidad Relación

3.8 CONCLUSIONES

En este capítulo se ha descrito de forma detallada la estructura del sistema propuesto, así como las clases con que contará el mismo. Se ha analizado

profundamente su estructura y el modelo de datos a utilizar, además de presentar su diseño.

CAPÍTULO 4: IMPLEMENTACIÓN

4.1 INTRODUCCIÓN

La fase de implementación en el desarrollo de un producto de software, es el mecanismo donde se ponen en práctica todas las descripciones y arquitecturas propuestas en las fases de análisis y diseño, es el complemento del trabajo de las fases que lo preceden dentro del proceso unificado de software. Se consideran más cercanos al acceso de datos y a la arquitectura física de la aplicación.

La implementación ofrece una materialización precisa de los requisitos, poniendo en práctica lo ya antes descrito. Y como resultado se obtiene componentes de código que se compilan e integran en versiones ejecutables del mismo. Este capítulo aborda el modelo de implementación de la aplicación que se propone, así como la especificación del los componentes que integrarán el despliegue de la aplicación.

4.2 MODELO DE DESPLIEGUE

El modelo de despliegue describe la distribución física de de los componentes lógicos desarrollados. Es decir, se sitúa el software en el hardware que lo contiene. Teniendo en cuenta que la solución propuesta está compuesta por dos subsistemas, el diagrama de despliegue del sistema, se representan tres nodos y un dispositivo. Uno de los nodos es la PC Servidor, que representa los ordenadores de los usuarios que están siendo controladas por el sistema, a las cuales se podrán acceder (utilizando el protocolo TCP/IP). Otro de los nodo es la PC cliente que es la encargada de realizar las peticiones del flujo de información. Y en el último nodo se encuentra ubicado el servidor de base de datos, se podrá acceder a este mediante el protocolo ADO. El dispositivo mencionado es la impresora, que se encuentra conectada a la PC del jefe de proyecto utilizando el puerto USB.

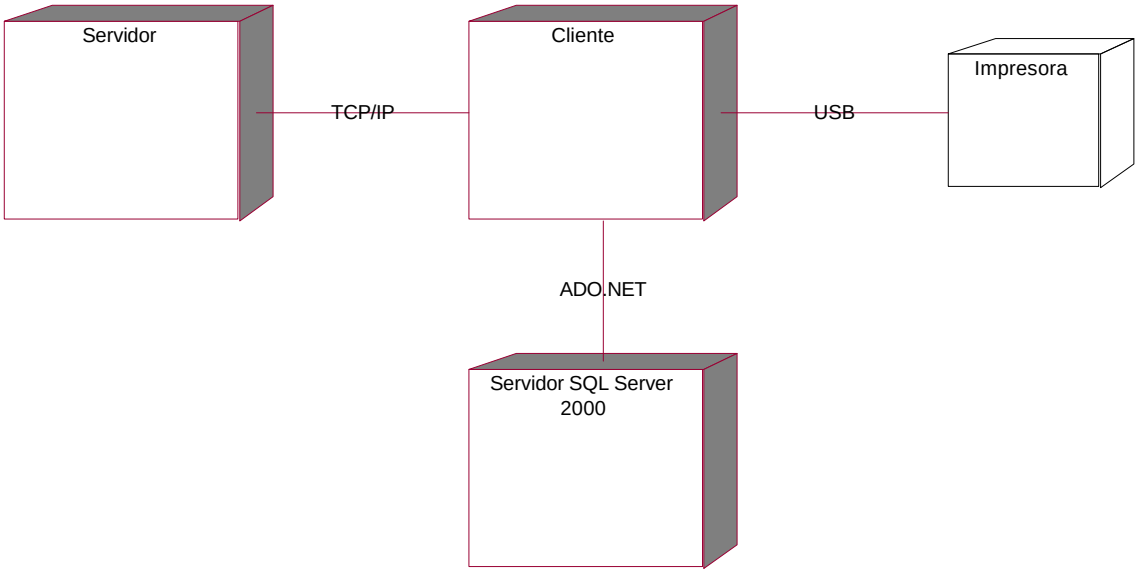

Figura 20 Diagrama de Despliegue

4.3 DIAGRAMA DE COMPONENTES.

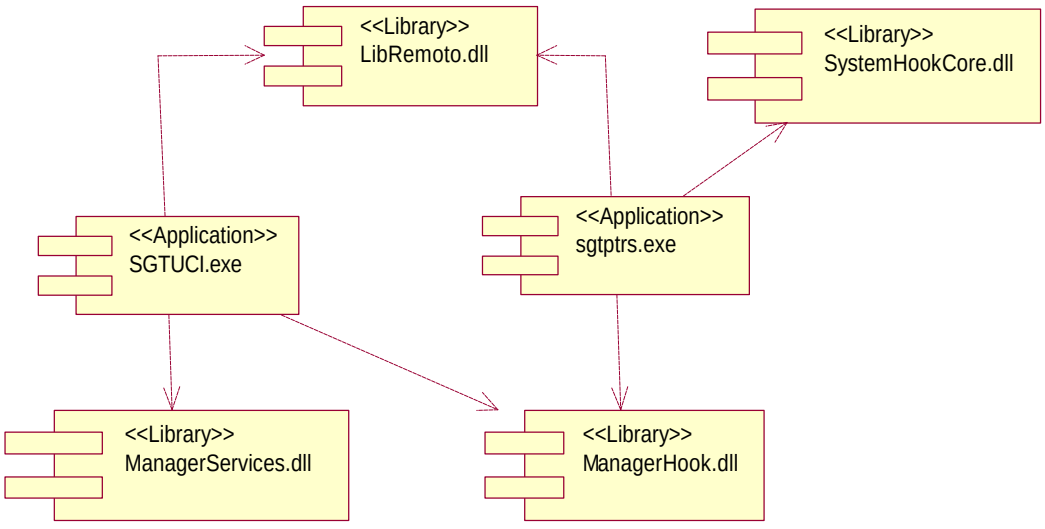

Figura 21 Diagrama de Componente de Código Ejecutable

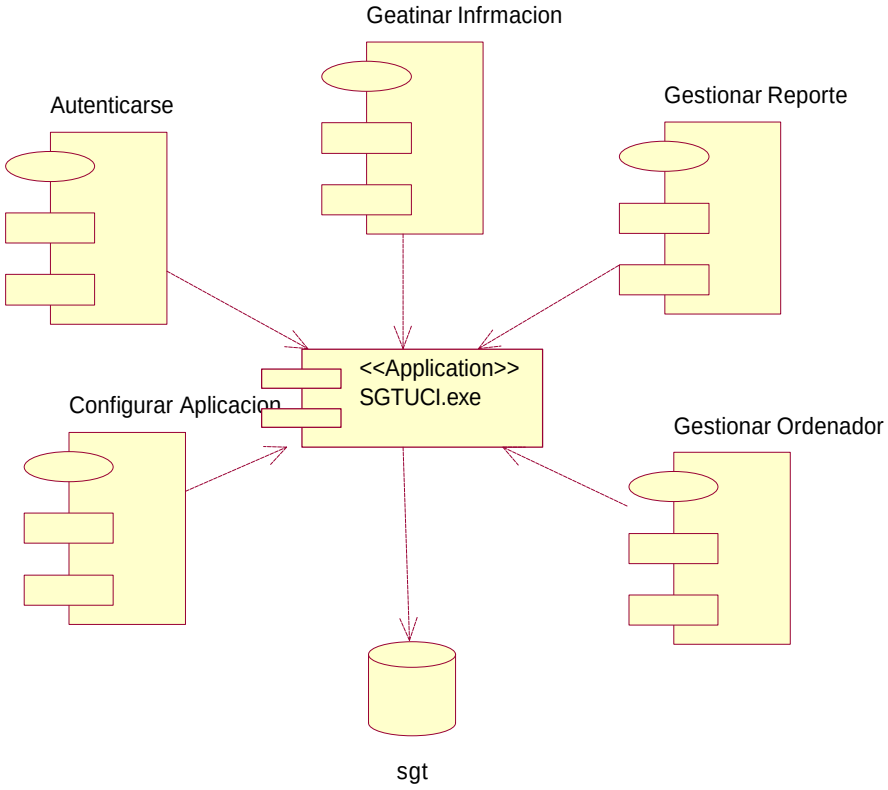

Figura 22 Diagrama de Componente de Código Fuente

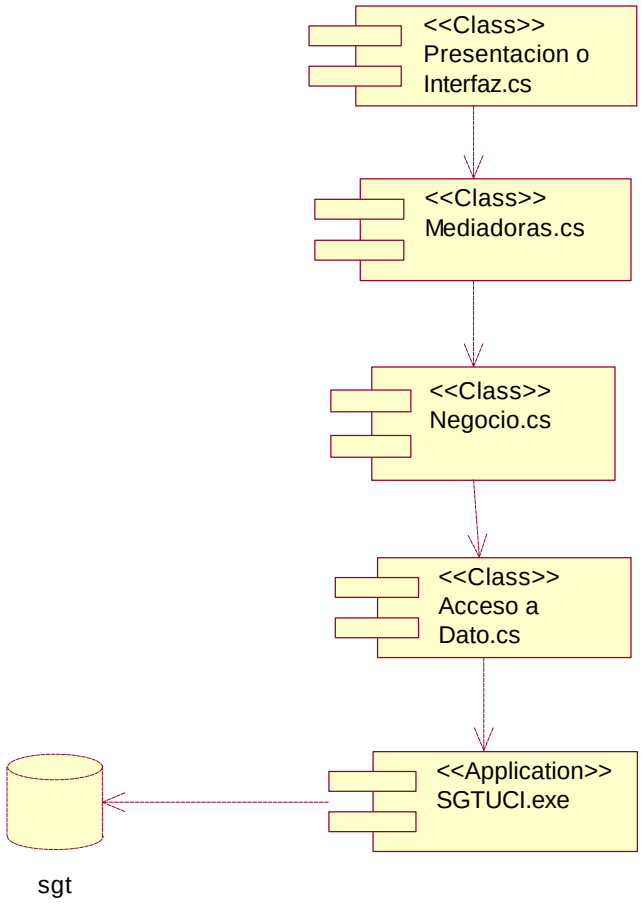

Figura 23 Diagrama de Componente de Código Fuente II

4.4 CONCLUSIONES

En este capítulo fueron realizados los diagramas de despliegue y componentes, conformando de esta manera el modelo de implementación, donde se representan el diagrama de componente de base de datos, el diagrama de componente de código fuente y el diagrama de componente de código ejecutable

CAPÍTULO 5: ESTUDIO DE FACTIBILIDAD

5.1 INTRODUCCIÓN

Para llevar a cabo el desarrollo de un proyecto de software es sumamente importante analizar el costo y los beneficios que este nos reportará. A partir de este análisis se podrá obtener el tiempo de desarrollo del proyecto, así como la cantidad de personas que intervendrán y el costo que tendrá el mismo.

En el siguiente capítulo se abordará la estimación de costos del sistema a desarrollar además de los costos y beneficios que aportará el mismo.

5.2 PLANIFICACIÓN BASADA EN CASOS DE USO

Paso 1. Cálculo de los puntos de casos de uso desajustados.

$$UUCP ? UAW ? UUCW$$

Donde:

UUCP: Puntos de casos de uso sin ajustar.

UAW: Factor de peso de los actores sin ajustar.

UUCW: Factor de peso de los casos de uso sin ajustar.

Tipo de actor	Descripción	Factor de peso	Actores	Total
Simple	Sistema con sistema a través de interfaz de programación.	1	0	0
Medio	Sistema con sistema mediante protocolo de interfaz basada en texto.	2	0	0
Complejo	Persona que interactúa con el sistema mediante interfaz gráfica.	3	1	3
Total			1	3

Tabla 5.1 Factor de peso de los actores sin ajustar.

$$UAW = \sum cant\ actores \times peso$$

$$UAW = 3$$

Tipo de CU	Descripción	Peso	Cantidad de CU	Total
Simple	El caso de uso tiene de 1 a 3 transacciones.	5	4	20
Medio	El caso de uso tiene de 4 a 7 transacciones.	10	0	0
Complejo	El caso de uso tiene más de 8 transacciones.	15	2	30
Total			4	50

Tabla 5.2 Factor de peso de los casos de uso sin ajustar.

$$UUCW = \sum cant\,CU \times Peso$$

$$UUCW = 50$$

$$UUCP = 3 + 50$$

$$UUCP = 53$$

Paso 2. Cálculo de los Puntos de casos de uso ajustados.

$$UCP = UUCP \times TCF \times EF$$

Donde:

UCP: Puntos de casos de uso ajustados.

UUCP: Puntos de casos de uso sin ajustar.

TCF: Factor de complejidad técnica.

EF: Factor de ambiente.

El factor de complejidad técnica (TCF) se determina mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada factor se cuantifica en un valor que va desde 0 (Aporte Irrelevante) hasta 5 (Aporte muy Relevante).

Factor	Descripción	Peso	Valor asignado	Total
T1	Sistema distribuido	2	5	0
T2	Tiempo de respuesta	1	4	4
T3	Eficiencia del usuario final	1	3	2
T4	Funcionamiento Interno complejo	1	5	5
T5	El código debe ser reutilizable	1	5	4
T6	Facilidad de instalación	0,5	4	2.5
T7	Facilidad de uso	0,5	5	2,5
T8	Portabilidad	2	5	10
T9	Facilidad de cambio	1	4	4
T10	Concurrencia	1	3	5
T11	Incluye objetivos especiales de seguridad	1	3	3
T12	Provee acceso directo a terceras partes	1	0	0
T13	Se requieren facilidades especiales de entrenamiento de usuarios	1	2	2
	Total			55.5

Tabla 5.3 Factor de complejidad técnica.

$$TCF\ ?\ 0.6\ ?\ 0.01?\ ?\ (peso?\ valor\ asignado)$$

TCF = 0.6 + 0.01 * 55.5

TCF = 0.6 + 0.555

$TCF \approx 1.135$

El factor de ambiente (EF) está relacionado con las habilidades y entrenamiento del grupo de desarrollo que realiza el sistema. Cada factor se cuantifica con un valor que va desde 0 (Aporte Irrelevante) hasta 5 (Aporte muy Relevante).

Factor	Descripción	Peso	Valor asignado	Total
E1	Familiaridad con el modelo de proyecto utilizado	1,5	3	4.5
E2	Experiencia en la aplicación	0,5	1	0.5
E3	Experiencia en la orientación a objetivos.	1	4	4
E4	Capacidad del analista líder.	0,5	4	2
E5	Motivación.	1	5	5
E6	Estabilidad de requerimientos	2	5	10
E7	Personal Part–Time	-1	0	0
E8	Dificultad del lenguaje de programación	-1	5	-5
	Total			21

Tabla 5.4 Factor de ambiente.

$$EF \approx 1.4 - 0.03 * \sum (peso * valor\ asignado)$$

EF = 1.4 – 0,03 * 21

EF = 1.4 – 0.63

$EF \approx 0.77$

UCP = UUCP * TCF * EF

UCP = 53 *1.135 * 0.77

$UCP ? 46.32$

Paso 3. Estimación de esfuerzo a través de los puntos de casos de uso.

$$E ? UCP * CF$$

Donde:

E: Esfuerzo estimado en horas hombres.

UCP: Punto de casos de usos ajustados.

CF: Factor de conversión.

Para obtener el factor de conversión (CF) se cuentan cuantos valores de los que afectan el factor ambiente (E1...E6) están por debajo de la media (3), y los que están por arriba de la media para los restantes (E7, E8). Si el total es 2 o menos se utiliza el factor de conversión 20 Horas-Hombre / Punto de Casos de uso. Si el total es 3 o 4 se utiliza el factor de conversión 28 Horas-Hombre / Punto de Casos de uso. Si el total es mayor o igual que 5 se recomienda efectuar cambios en el proyecto ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

En este caso se puede decir que:

CF = 20 Horas-Hombre / Punto de Casos de uso.

$E ? 46.32 * 20$

$E ? 926.387$ Horas-Hombre

Actividad	Porcentaje %	Horas-Hombres
Análisis	15	277.9161
Diseño	15	277.9161
Implementación	50	926.387

Pruebas	10	185.2774
Sobrecarga (otras actividades)	10	185.2774
Total	100	1852.774

Paso 4. Calcular esfuerzo de todo el proyecto.

Tabla 5.5 Esfuerzo del proyecto.

Si E_T = 1852.774 horas-hombre y se estima que cada mes tiene como promedio 192 horas laborables, eso daría un E_T = 9.65 mes-hombre.

Esto quiere decir que 1 persona puede realizar el problema analizado en poco más de 9 meses y medio aproximadamente.

5.3 COSTO DEL PROYECTO

Se asume como salario promedio mensual $270.50

CH: Cantidad de hombres.

Tiempo: Tiempo total del proyecto.

CH = 2 hombres

CHM = 2 * Salario Promedio

CHM = 200.00 $/mes

Costo = CHM * E_T / CH

Costo = 200.00 * 9.65

Costo = $1929.97292˜ $1930

Tiempo = E_T / CH

Tiempo = 9.64986458 / 2

Tiempo = 4.82493229 ˜ 4.8

De los resultados obtenidos se interpreta que con 2 hombres trabajando en el proyecto el mismo se desarrolla en 4,8 meses, aproximadamente 5 meses y su costo total se estima que sea $1930.

5.4 BENEFICIO TANGIBLE E INTANGIBLE

El sistema para la gestión del tiempo en los proyectos productivos de la UCI no es un producto con fines comerciales, su objetivo fundamental es resolver los problemas existentes en la universidad con respecto al aprovechamiento del tiempo de máquina en los proyectos productivos e inclusive en los laboratorios docentes, aunque en un futuro se pudieran hacer mejoras y considerar su comercialización.

El beneficio fundamental es contar con un sistema capaz de controlar en que están empleando realmente el tiempo de máquina los estudiantes, además de poder contar con información correctamente almacenada para arribar a conclusiones a partir de los reportes emitidos por el sistema.

En este caso los beneficios inmediatos son intangibles:

1. Rápido acceso a la información solicitada, ya sea de un usuario determinado o un ordenador.
2. Reportes con información actualizada de cada ordenador o usuario que queramos monitorear.

Es bueno señalar que de forma indirecta también se obtendrán beneficios tangibles ya que una vez desplegado el sistema en la red se podrán controlar con mayor facilidad las actividades desarrolladas en los proyectos trayendo consigo una mayor eficiencia y reduciendo drásticamente los tiempos de entrega de los proyectos, lo que propicia mayores ganancias en el orden económico.

5.5 ANÁLISIS DE COSTO Y BENEFICIO

Es importante señalar que el desarrollo de cualquier producto informático es muy caro. Por lo tanto determinar si se lleva a cabo o no depende de varios factores como por ejemplo, el factor económico, en dependencia de los aportes y beneficios que reportaría y el factor social que también es no menos importante. En este caso el sistema desarrollado está pensado para usarse en la Universidad de las Ciencias Informáticas, aunque pudiera ser extensivo a otros lugares, y su objetivo está centrado en controlar el tiempo de máquina en una red con tecnología basada en Windows, por lo que los beneficios del mismo serán en el orden social, pero con cierta influencia desde el punto de vista económico ya que se podrá controlar con mayor precisión las actividades de proyectos que si aportan mucho económicamente a nuestro país.

Con los beneficios antes mencionados que traerá el sistema y analizando los costos del mismo se deduce claramente que su implementación es totalmente factible.

5.6 CONCLUSIONES

En este capítulo se describió el estudio de factibilidad realizado correspondiente al sistema propuesto, teniendo en cuenta el costo estimado y los beneficios que reportará al ser implantado, lo cual mejorará el proceso de desarrollo en los proyectos de la universidad.

CONCLUSIONES

Con el siguiente trabajo se espera que haya sido comprendido el problema planteado, así como la solución propuesta, además de todo el proceso de desarrollo que se llevó a cabo para su análisis e implementación.

Con la investigación llevada a cabo se logró el objetivo propuesto ya que cumple con las expectativas y las metas trazadas. Además se alcanzó el objetivo en cuestión que era implementar un sistema con una interfaz amigable que permita llevar un control estricto de las actividades que se desarrollan en los proyectos productivos de la UCI, para conocer realmente en que se está empleando el tiempo destinado a la producción, y de esta forma garantizar las entregas de los mismo en el tiempo planificado.

Se ha demostrado la eficacia de los lenguajes y tecnologías utilizadas para el desarrollo del sistema. Se realizó una base de datos, donde se almacena toda la información necesaria de los proyectos, para de esta forma garantizar la veracidad y centralización de la misma.

Se realizó el análisis, diseño e implementación del sistema. La solución propuesta ha sido acertada, los requerimientos soportan al sistema y los casos de uso satisfacen las necesidades funcionales. Se han seguido los principios básicos de diseño descritos para el desarrollo del sistema. Se logra una seguridad y protección de los datos consecuente con el nivel de seguridad requerido.

RECOMENDACIONES

1. Poner a prueba el sistema durante un período de tiempo determinado, para comprobar sus potencialidades y verificar que funciona como se esperaba.
2. Continuar investigando en el tema con el objetivo de añadirle nuevas funcionalidades.
3. Una vez probado y aprobado el sistema ponerlo en funcionamiento en los proyectos que desarrolla la universidad.

REFERENCIA BIBLIOGRÁFICA

1. http://www.vrweb.cl/vrc/producto/art/id_8.html

2. IceWALKERS. IceWALKERS. [Online] Abril 15, 2007.
http://www.icewalkers.com/Linux/Software/521240/journyx-timesheet.html

3. GENBETA. GENBETA. [Online] [Cited: Abril 11, 2007.]
http://www.genbeta.com/2005/09/23-time-tracker-controla-tu-tiempo-de-la-mejor-maner

4. Nideaderedes. Nideaderedes. [Online] Noviembre 24, 2006. [Cited: Mayo 5, 2007.]
http://nideaderedes.urlansoft.com/2006/11/24/control-del-tiempo-empleado-en-un-proyecto/.

5. ZMA. ZMA. [Online]
http://www.zma.com.ar/inicio/productos/gestion/index.htm

6. Technologypartners. Technologypartners. [Online] Mayo 3, 2007.
http://www.techno-partners.com/soluciones/administracion/dameware/index.html

7. http://es.wikipedia.org/wiki/.NET

8. http://es.wikipedia.org/wiki/Java_2_Enterprise_Edition

9. http://www.crystalsolutions.com.ar/productos/crystalreports.html

10. xtras.net. ActiveReport. [En línea] [Citado el: 12 de 04 de 2006.]
http://www.xtras.net.

11. http://www.programatuweb.com/manuales/net/introduccion.php

12. http://es.wikipedia.org/wiki/Microsoft_Access

13. Escibano, Gerardo Fernández**.** Extreme Programing

14. Pressman, Roger S**.** Ingenieria del Software . La Habana : Félix Varela, 2005.

15. http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/multiple-html/c12.html

BIBLIOGRAFÍA

Crystalintelligence. Crystal Report. [En línea] [Citado el: 15 de 04 de 2006.] http://www.crystalintelligence.com

Escibano, Gerardo Fernández**.** Extreme Programing

FERGUSON, JEFF y PATTERSON, BRIAN y BERES, JASON**.** La Biblia de C#. Mayo 12, 2007.

GENBETA. GENBETA. [Online] [Cited: Abril 11, 2007.] http://www.genbeta.com/2005/09/23-time-tracker-controla-tu-tiempo-de-la-mejor-maner

GENBETA. GENBETA. [Online] [Cited: Abril 11, 2007.] http://www.genbeta.com/2005/09/23-time-tracker-controla-tu-tiempo-de-la-mejor-manera.

gsKinner. gsKinner. [Online] Abril 14, 2007. http://www.gskinner.com/blog/archives/2007/03/apollo_time_tra.html

http://www.vrweb.cl/vrc/producto/art/id_8.html

http://es.wikipedia.org/wiki/.NET

http://es.wikipedia.org/wiki/Java_2_Enterprise_Edition

http://www.crystalsolutions.com.ar/productos/crystalreports.html

http://www.crystalsolutions.com.ar/productos/crystalreports.html

http://www.programatuweb.com/manuales/net/introduccion.php

http://es.wikipedia.org/wiki/Microsoft_Access

http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/multiple-html/c12.html

http://www.solocodigo.com/

http://www.codeproject.com/

IceWALKERS. IceWALKERS. [Online] Abril 15, 2007

Journyx. Journyx. [Online] Abril 14, 2007. http://www.journyx.com/.

http://www.icewalkers.com/Linux/Software/521240/journyx-timesheet.html

MSDN. [Online] Abril 7, 2004. [Cited: Abril 11, 2007.] http://www.microsoft.com/spanish/msdn/matrix/Tracker.asp.

Nideaderedes. Nideaderedes. [Online] Noviembre 24, 2006. [Cited: Mayo 5, 2007.] http://nideaderedes.urlansoft.com/2006/11/24/control-del-tiempo-empleado-en-un-proyecto/.

Pressman, Roger S**.** Ingenieria del Software . La Habana : Félix Varela, 2005.

Sun. Java 2 Enterprise Edition. [En línea] [Citado el: 06 de 12 de 2006.] http://es.sun.com

Technologypartners. Technologypartners. [Online] Mayo 3, 2007.
Technologypartners. Technologypartners. [Online] Mayo 3, 2007. http://www.techno-partners.com/soluciones/administracion/dameware/index.html.
VrWeb. [Online] [Cited: Mayo 4, 2007.] http://www.vrweb.cl/vrc/producto/art/id_8.html.
WinTotal. WinTotal. [Online] Abril 10, 2007. [Cited: Abril 10, 2007.] http://www.wintotal.de/.
xtras.net. ActiveReport. [En línea] [Citado el: 12 de 04 de 2006.] http://www.xtras.net.
ZMA. ZMA. [Online] http://www.zma.com.ar/inicio/productos/gestion/index.htm

GLOSARIO

A

API

Del inglés (Application Programming Interface - Interfaz de Programación de Aplicaciones) es un conjunto de especificaciones de comunicación entre componentes software. Se trata del conjunto de llamadas al sistema que ofrecen acceso a los servicios del sistema desde los procesos y representa un método para conseguir abstracción en la programación, generalmente entre los niveles o capas inferiores y los superiores del software.

ANSI

Instituto de Estándar Nacional Americano.

ADO

Objeto de datos de Activex.

ASP

Tecnología desarrollada por Microsoft para sus servidores de páginas Web. Una página ASP es una página html que incluye uno o más scripts (pequeños programas).

B

BCL

Librerías de clases Base

BYTECODE

El bytecode es un código intermedio más abstracto que el código máquina. Habitualmente se lo trata como a un fichero binario que contiene un programa ejecutable similar a un módulo objeto, que es un fichero binario que contiene código máquina producido por el compilador.

C

CLR

Entorno Común de Ejecución para Lenguajes

CLI

Infraestructura común de lenguajes

CPU

Se denomina CPU (siglas de Central Processing Unit) o Unidad Central de Proceso (UCP) a la unidad donde se ejecutan las instrucciones de los programas y se controla el funcionamiento de los distintos componentes del ordenador. Suele estar integrada en un chip denominado microprocesador.

D

DLL

Es el acrónimo de Dynamic Linking Library (Bibliotecas de Enlace Dinámico), término con el que se refiere a los archivos con código ejecutable que se cargan bajo demanda del programa por parte del sistema operativo. Esta denominación se refiere a los sistemas operativos Windows siendo la extensión con la que se identifican los ficheros, aunque el concepto existe en prácticamente todos los sistemas operativos modernos

DHTML

El HTML Dinámico o DHTML (del inglés Dynamic HTML) designa el conjunto de técnicas que permiten crear sitios web interactivos utilizando una combinación de lenguaje HTML estático, un lenguaje interpretado en el lado del cliente (como JavaScript) y el lenguaje de Hojas de estilo en cascada (CSS).

E

ECMA

Asociación europea de los fabricantes de computadora.

F

FRAMEWORK

En el desarrollo de software, un Framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un

framework puede incluir soporte de programas, librerías y un lenguaje de scripting entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

G

GPL

Del ingles (General Public License o licencia pública general) es una licencia creada por la Free Software Foundation a mediados de los 80, y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

H

HTML

El HTML, acrónimo inglés de Hyper Text Markup Language (lenguaje de marcación de hipertexto), es un lenguaje de marcas diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas web. Gracias a Internet y a los navegadores del tipo Explorer o Netscape, el HTML se ha convertido en uno de los formatos más populares que existen para la construcción de documentos.

HOOK

Gancho que se establece a un elemento del sistema (teclado, mouse, sheel, cbt).

I

ISO

Organización Internacional para la estandarización

IDE

Un entorno de desarrollo integrado o en inglés Integrated Development Environment ('IDE') es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios. Es un entorno de programación que ha sido empaquetado

como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.

J

JIT

Es una técnica para mejorar el rendimiento de sistemas de programación que compilan a bytecode, consistente en traducir el bytecode a código máquina nativo en tiempo de ejecución. La compilación en tiempo de ejecución se construye a partir de dos ideas anteriores relacionadas con los entornos de ejecución: la compilación a bytecode y la compilación dinámica.

JDBC

Es el acrónimo de Java Database Connectivity, un API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java independientemente del sistema de operación donde se ejecute o de la base de datos a la cual se accede utilizando el dialecto SQL del modelo de base de datos que se utilice.

M

MSIL

Lenguaje intermedio de Microsoft

MAC

En redes de computadoras Media Access Control Address cuyo acrónimo es MAC es un identificador físico -un número, único en el mundo, de 48 bits- almacenado en fábrica dentro de una tarjeta de red o una interfaz usada para asignar globalmente direcciones únicas en algunos modelos OSI (capa 2) y en la capa física del conjunto de protocolos de interne.

N

.NET

.NET es un proyecto de Microsoft para crear una nueva plataforma de desarrollo de software con énfasis en transparencia de redes, con independencia de plataforma y que permita un rápido desarrollo de aplicaciones.

NT

Windows Nueva Tecnología. Versión de Windows diseñada para entornos profesionales.

R

RTF

RTF es el acrónimo inglés de Rich Text Format, lenguaje de descripción desarrollado por Microsoft para intercambiar información entre programas multiplataforma de edición de texto. El RTF es un pobre formato estandarizado con diversas incompatibilidades incluso entre distintas aplicaciones de Microsoft, y rara vez se usa para guardar documentación.

RUP

El Proceso Racional Unificado (RUP, el original inglés Rational Unified Process) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP es en realidad un refinamiento realizado por Rational Software del más genérico Proceso Unificado.

RM

Java Remote Method Invocation, es un mecanismo ofrecido en Java para invocar un método remotamente. Al ser RMI parte estándar del entorno de ejecución Java usarlo provee un mecanismo simple en una aplicación distribuida que solamente necesita comunicar servidores codificados para Java.

T

TCP/IP

La familia de protocolos de Internet es un conjunto de protocolos de red que implementa la pila de protocolos en la que se basa Internet y que permiten la transmisión de datos entre redes de computadoras. En ocasiones se la denomina conjunto de protocolos TCP/IP, en referencia a los dos protocolos más importantes que la componen: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP)

TIFF

Formato de fichero para almacenar gráficos de alta calidad.

TI

Tecnologías de la información

U

UML

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modelling Language) es el lenguaje de modelado de sistemas de software más conocido en la actualidad; aún cuando todavía no es un estándar oficial

UCI

Universidad de las Ciencias Informáticas.

W

WIN32

Significa "Windows 32 bits". Hace referencia a todas las plataformas de 32 bits del sistema operativo Windows: Windows NT, Windows 95, Windows 98, Windows CE.

X

XML

XML es el acrónimo del inglés Extensible Markup Language (lenguaje de marcado ampliable o extensible) desarrollado por el World Wide Web Consortium (W3C).