Universidad de las Ciencias Informáticas

TRABAJO DE DIPLOMA

Título: *Desarrollo de un sistema informático para la gestión del uso correcto de las computadoras en los laboratorios de producción de la facultad 9.*

Autores:

Carlos Costa Sánchez

Yuliexis Reyes Chávez

Tutor: Ing. Liester Cruz Castro

Cotutor: Msc. Manuel Enrique Puebla Martínez.

DEDICATORIA

De Carlos:

A mis padres por siempre preocuparse y confiar en mí.

A mi hermana, que tome mi ejemplo y sea mejor que yo.

A mi querida novia, Leibys, que fue mi gran amiga y compañera en todo momento.

A todos mis familiares que me sirvieron de apoyo.

A todas las personas que de una forma u otra se preocupó por mis estudios.

De Yuliexis:

A mi hija Yaliannis que ha sido mi fuente de inspiración y es lo que más quiero en el mundo.

A mis padres por siempre estar presente en mi vida y confiar en mí.

A mi abuelo Chávez por estar siempre a mi lado dándome consejos y en especial a mi abuelo Celso (chicholo), a pesar de no poderme ver formar como ingeniero, siempre estuvo al tanto de mis estudios.

A mi otra madre, mi abuela Edelmira (nené) por preocuparse siempre por mí.

A mis primos por confiar en mí y sobre todo a Edilberto (papachongo) e Idianis que son como hermanos para mí y siempre han estado conmigo apoyándome en todo momento.

A mi querida novia, Neris, que ha sido mi compañera y amiga durante todo este tiempo y ha estado a mi lado dándome fuerzas para seguir adelante.

A toda mi familia por confiar siempre en mí.

AGRADECIMIENTOS

A nuestros padres, hermanos y familiares más queridos.

A nuestros amigos que siempre han estado presentes cuando los necesitamos.

A la Revolución Cubana por garantizarnos todos los medios necesarios para nuestra educación.

A la Universidad de Ciencias Informáticas por formarnos como buenos ingenieros.

A nuestro tutor Liester Cruz Castro por su paciencia y dedicación.

A nuestro amigo Yuniesky O. Vasconcelo Mir por su ayuda desinteresada y su disposición de ayudarnos en todo momento.

A Julio A. Leyva Durán por toda la ayuda prestada.

A nuestras novias, Leibys y Neris, que han demostrado una gran paciencia y apoyo en todo momento.

A todas las personas que en un momento u otro nos brindaron su ayuda.

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) a hacer uso del mismo en su beneficio.

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

Carlos Costa Sánchez	Yuliexis Reyes Chávez	Ing. Liester Cruz Castro
_____	_____	_____
Firma del autor	Firma del autor	Firma del tutor

DATOS DE CONTACTO

Tutor: Ing. Liester Cruz Castro.

Email: lcruz@uci.cu

Cotutor: Msc. Manuel Enrique Puebla Martínez.

Email: mpuebla@uci.cu

OPINIÓN DEL TUTOR

Título:" Desarrollo de un sistema informático para la gestión del uso correcto de las computadoras en los laboratorios de Producción de la Facultad 9".

Autores:

Carlos Costa Sánchez.

Yuliexis Reyes Chávez.

El tutor del presente Trabajo de Diploma considera que durante su ejecución los estudiantes mostraron las cualidades que a continuación se detallan.

Por todo lo anteriormente expresado considero que los estudiantes están aptos para ejercer como Ingenieros Informáticos; y propongo que se le otorgue al Trabajo de Diploma la calificación de ____.

Ing. Liester Cruz Castro.

_____ _____

Firma Fecha

RESUMEN

El presente trabajo muestra el resultado de una investigación realizada con el objetivo de desarrollar un sistema informático que gestione el uso correcto de las computadoras en los laboratorios de producción de la facultad 9. Para un mejor entendimiento de la materia se realizó un estudio profundo de los temas relacionados con los procesos de gestión y control, apoyándose para el estudio de dichos temas en los diferentes autores, y empleándose diferentes métodos: tantos teóricos como empíricos, que posibilitaron una mayor comprensión del problema actual.

Se exponen las principales debilidades diagnosticadas al proceso del negocio a través de diferentes técnicas, identificándose las principales tendencias y tecnologías actuales para la modelación y desarrollo del sistema. La aplicación diseñada como producto final además de maximizar el control de los ordenadores durante su uso es de gran ayuda para los líderes de proyectos en la toma de decisiones. El documento recoge los distintos requisitos que hacen posible el éxito y limitaciones de la misma, los resultados tangibles y no tangibles mediante un estudio de factibilidad realizado, además se demuestra que el sistema desarrollado cumple con la calidad requerida gracias a un conjunto de pruebas efectuadas.

PALABRAS CLAVES

Control, monitoreo, gestión, herramienta de gestión, procesos, lista de procesos, tecnologías, herramientas, factibilidad, diseño, implementación.

KEYWORDS

Control, monitoring, management, management tool, Process List, technologies, tools, feasibility, design, implementation.

DATOS EN INGLÉS

This work shows the results of investigation with the aim of developing a computer system to manage the proper use of computers in production laboratories of the nine faculty. For a better understanding of the matter is made a thorough study of issues related to management and control processes, relying on various authors, and employ different methods: theoretical and empirical, which enabled a better understanding of the current problem.

It sets out the main weaknesses of the business process diagnosed through several techniques, identifying key trends and current technologies for modeling and system development. The final product application designed as well to maximize control over their use of computers is helpful for project leaders in making decisions. It reflects the different requirements that make possible the success and limitations of it, the tangible and intangible results through a feasibility study carried out also shows that the system developed meets the required quality, through a range of tests carried out.

Índice

Índice de Tablas

Índice de Figuras

Introducción:

En la actualidad la sociedad está inmersa en un profundo avance en el tema de la tecnología, la gestión de la información y las comunicaciones. La información para cualquier país es un aspecto muy importante, más aún si se quiere lograr y mantener un mayor desarrollo económico-social. A partir de las últimas décadas del siglo XX el mundo ha alcanzado un gran desarrollo Científico-Técnico, que ha traído consigo la evolución de la informática y las nuevas tecnologías, teniendo ambas un fuerte impacto social.

Dentro de los notorios adelantos de la ciencia y las nuevas tecnologías, la producción de software juega un papel fundamental en el desarrollo de la economía y la sociedad del mundo contemporáneo. El gobierno cubano no está ajeno al contexto anteriormente descrito y dedica cuantiosos recursos para lograr la informatización del país. La creación de la Universidad de las Ciencias Informáticas (UCI) refuerza las acciones implementadas en materia de desarrollo de la informática en Cuba.

La UCI tiene el propósito de formar profesionales capacitados para la fabricación de software, con el objetivo de desarrollar productos con los cuales se contribuya a la informatización del país y se logre insertar a Cuba en el mercado mundial. Esta institución vincula la docencia a la producción, con el objetivo de capacitar a los estudiantes de forma práctica en el desarrollo de software. Para una mayor organización en la producción esta universidad está conformada por 9 facultades en su sede central y por 3 facultades regionales, donde cada una responde a diferentes perfiles productivos.

La facultad 9 actualmente cuenta con un Centro de Producción, dividido a su vez en dos departamentos, Geoinformática y Señales Digitales. Cada uno de estos departamentos está compuesto por varios proyectos, donde se realizan actividades en función de la producción de software. Estas actividades son llevadas a cabo en laboratorios dedicados exclusivamente a la docencia y la producción.

El cumplimiento del horario docente-productivo en los laboratorios de producción en numerosas ocasiones no se realiza de la forma establecida, existen deficiencias reflejadas principalmente en que gran cantidad de estudiantes no aprovechan al máximo el tiempo de máquina asignado para cumplir con las tareas productivas y en su lugar realizan otras actividades que no representan beneficio para la producción de software. El uso incorrecto de los laboratorios de producción influye de forma negativa en el desarrollo de los proyectos productivos, así como en el cumplimiento de las fechas fijadas en los cronogramas de dichos proyectos; elevando además el consumo energético en dichos locales.

La facultad 9 no cuenta con un mecanismo factible para monitorizar y controlar las actividades que se realizan en los laboratorios de producción. En la actualidad esta acción se ejecuta de forma manual mediante un profesor designado por el jefe de proyecto, el mismo controla el laboratorio en distintos horarios y guarda la información en un documento digital. El proceso presenta las deficiencias que se enuncian a continuación: en numerosas ocasiones el horario de control es conocido por los estudiantes y estos toman las medidas pertinentes para no realizar en ese momento acciones incorrectas en cuanto al cumplimiento de las tareas que se les han orientado, no se controla con la seriedad necesaria por parte de los responsables de realizar esta tarea, los estudiantes conocen quien es el que controla y cuando llega el momento realizan las actividades correctamente.

La situación problemática planteada anteriormente conduce al siguiente **problema de investigación**: La utilización indebida de las computadoras en los laboratorios de producción de la facultad 9 afecta en el cumplimiento de las tareas productivas.

Para darle solución a dicho problema se plantea como **objeto de estudio**: Los procesos de gestión del uso de las computadoras de los laboratorios de la facultad 9 y como **campo de acción**: La automatización de los procesos de gestión del uso de las computadoras en los laboratorios de producción de la Facultad 9.

El **objetivo general** de este trabajo es: Desarrollar un sistema informático que gestione el uso correcto de las computadoras en los laboratorios de producción de la facultad 9.

La **Idea a defender planteada** es: El desarrollo de un sistema informático para la gestión de la correcta utilización de las computadoras en los laboratorios de producción de la facultad 9, permitirá mejoras en el cumplimiento de las tareas productivas.

Para cumplir el objetivo se tienen las siguientes **tareas de investigación**:

- ✓ Actualizar los logros y limitaciones en los enfoques existentes de los procesos de gestión del uso de computadoras.
- ✓ Realizar un diagnóstico de las tendencias y tecnologías actuales sobre este proceso.
- ✓ Identificar las metodologías actuales sobre el ámbito del problema en cuestión.
- ✓ Modelar los procesos de Negocio, Análisis, Diseño e Implementación del sistema.
- ✓ Implementar y Probar el sistema.
- ✓ Validar la solución propuesta.

En el desarrollo de la aplicación se aplicaron diferentes **métodos científicos:**

Teóricos:

- ✓ **Analítico-Sintético:** Permitió el análisis de los documentos y la bibliografía con el objetivo de obtener información sintetizada y detallada relacionada con el objeto de estudio.

- ✓ **Análisis Histórico-Lógico:** Este método posibilitó el análisis histórico-lógico del proceso de gestión de la información del uso de las computadoras en los laboratorios de producción de la facultad 9.

Empíricos:

- ✓ **Observación:** Se empleó para comprobar cómo se realiza el proceso de gestión de la información del uso de las computadoras y así se obtuvo un mejor entendimiento de este fenómeno.

Los **posibles resultados**:
Un sistema informático que garantice la gestión del uso correcto de las computadoras en los laboratorios de producción de la facultad 9, además debe contar con un manual de usuario para mayor información y ayuda a los usuarios finales.

Este trabajo de diploma presenta 5 capítulos en total. En el primer capítulo se aborda el tema de la **Fundamentación teórica,** en el cual se explican los aspectos teóricos del trabajo y elementos relacionados con los principales conceptos del problema en cuestión. Algunos conceptos a tratar son: control, gestión, monitoreo, entre otros, garantizando así una mejor comprensión del negocio.
El segundo capítulo, **Tendencias y Tecnologías actuales** se aborda el tema de las tecnologías más usadas y se escogen las necesarias para la realización del sistema informático, se aborda el tema del lenguaje de programación, la metodología de desarrollo de software a utilizar, así como los gestores de base de datos entre otras tecnologías a escoger.
El tercer capítulo titulado **Presentación de la solución propuesta**, resume la descripción de los procesos del negocio y una representación de los casos de usos del sistema resultantes. Se da una explicación de cómo se debe desarrollar el sistema, tratando temas relacionados con el negocio, la representación del sistema, dígase los actores y casos de uso más significativos.
En el cuarto capítulo, **Construcción de la solución propuesta** están descritos los modelos de Análisis, Diseño e Implementación, así como el tema de los estándares de codificación y la realización de las pruebas correspondientes al sistema resultante.
Capítulo cinco, **Estudio de Factibilidad**, se realiza un estudio para ver si la solución final es factible, realizando la estimación por puntos de casos de uso, donde se obtiene como resultado si la aplicación es factible de construir.

CAPÍTULO 1: Fundamentación Teórica.

En el presente capítulo aparece todo lo referente al estado de un conjunto de conocimientos que proporciona el marco teórico-conceptual referente al dominio de los sistemas de gestión y control de las computadoras en los laboratorios de producción, la descripción de las características fundamentales, estructura, componentes y funciones que cumple el objeto de estudio de interés para el investigador. Las relaciones de causa y efecto que genera la situación problemática, un análisis crítico a las soluciones dadas a problemas presentados en áreas y momentos diferentes, ofreciéndose un resumen de los principales aspectos tratados en los tópicos relacionados en este capítulo.

1.1 Conceptos asociados al dominio del problema

Existen un conjunto de conceptos asociados al problema planteado anteriormente, a los cuales se le dará una explicación en los siguientes acápites.

1.1.1 Control.

Todo estudio que se encamine a reducir las brechas existentes en los mecanismos de control de cualquier proceso en las organizaciones en Cuba debe tener como antecedentes la Resolución Económica del V Congreso del Partido Comunista de Cuba donde se consigna:"… En las nuevas condiciones que opera la economía,…," se precisa, "… el control oportuno y eficaz de la actividad económica"… y cultural, "es esencial para la dirección a cualquier nivel…" (Precios., 2003)

Son numerosos los autores que han reflexionado sobre el control y gestión de distintos procesos, observándose una evolución hacia formas superiores que garanticen la eficiencia futura de los mismos. Para tener una mayor comprensión de lo planteado anteriormente se invita a dar un breve recorrido por los restantes acápites, de forma tal que permita formar un mayor juicio acerca del tema de estudio y darle la solución más apropiada al problema de investigación.

Muchas son las definiciones que han sido consultadas en distintas obras de diferentes autores entre ellas se cita: “control, es el proceso para asegurar que las actividades reales se ajustan a las actividades planificadas” (STONER, 1999)

El industrial francés Henry Fayol consideraba que el control “consistía en verificar si todo marcha de acuerdo con el plan adoptado, las instrucciones dadas y los principios establecidos. Tiene por objeto subrayar las debilidades y los errores para que puedan rectificarse y evitar que se repitan” (Fayol, 1946), en otras palabras señala que el control es la medición y la corrección de las actividades de los subordinados para asegurar que los hechos se ajusten a los planes establecidos.

Controlar en el sentido general es el término anglosajón que significa, dominar o conducir en la dirección deseada, asociándose a los vocablos regular, dirigir y verificar. El control entraña los siguientes elementos básicos que no deben ser omitidos para controlar un proceso:

- ✓ Establecer estándares de desempeño.
- ✓ Medir los resultados presentes.
- ✓ Comparar estos resultados con las normas establecidas.
- ✓ Tomar medidas correctivas cuando se detecten desviaciones.
- ✓ Premiar, felicitar, remunerar y disciplinar.

1.1.2 Proceso

El control surge por necesidad de los procesos de mantenerse y autogenerarse, pero primero se debe definir que es un proceso el cual según (DRUCKER, 2003)“es el método sistemático para manejar actividades”. Otra definición de proceso “es el sistema de actividades que utiliza recursos para transformar elementos de entrada en elementos de salida” (ISO/900, 2000)

“Las actividades de cualquier organización pueden ser concebidas como integrantes de un proceso determinado. Desde este punto de vista, una organización cualquiera puede ser considerada como

un sistema de procesos, más o menos relacionados entre sí". (Valentín, José Manuel Jiménez., 2008)

Entre otras palabras el proceso no es más que un conjunto de actividades que se relacionan entre sí, dependiendo de una o más entradas las convierte en salidas ó resultados.

1.1.3 Gestión.

Si se analiza esta última definición que aparece en las normas ISO se hace necesario intervenir con distintas acciones para que se obtengan resultados de salidas positivas y satisfactorias para el proceso, lo cual implica que no se obtienen por sí solo, si no hay que gestionar.

"Gestión, proviene de la acepción latina gesti-onis, acción del verbo género que significa acción y efecto de gestionar, o sea, hacer diligencias que conduzcan al logro de un negocio o deseo cualquiera" (Llanes Delgado, 2004), también se puede observar la gestión "como la actividad coordinada para dirigir y controlar una organización" (ISO/900, 2000), que puede ser utilizada en el dominio previsto porque se trata de controlar el proceso que las personas realizan en una actividad estructurada. Gestión "consiste en seleccionar ciertas acciones, partiendo de diversas informaciones" (Rolhud, 1992), las cuales permiten tomar las decisiones más adecuadas.

1.1.4 Herramienta de gestión.

La gestión garantiza la efectiva y eficiente utilización de los recursos materiales, humanos y financieros puestos a disposición de la universidad para llevar a cabo su objetivo básico para el cual fue creada, para ello es necesario desarrollar como principal una herramienta de gestión de control, la cual se define como "una herramienta cuyo soporte es un conjunto de informes sistemáticos que permiten a la organización analizar la captación, transformación y utilización de los recursos". (Hernández Regalado, 2007)

Para elaborar una herramienta de gestión es necesario establecer la relación existente entre las variables cuantitativas y cualitativas que conforman el modelo en cuestión, como resultado, sobre las variables claves elegidas permitirá:

- ✓ Observar la situación pasada, presente y futura.
- ✓ Monitorear las tendencias y cambios generales.
- ✓ Tomar decisiones oportunas y deseablemente acertadas.
- ✓ Adoptar las medidas correctivas.
- ✓ Controlar en el tiempo las principales variables.

La herramienta entonces permitirá analizar el cumplimiento de los objetivos y el desempeño de cada computadora en los laboratorios de producción. La herramienta de control moderna busca las principales vías y medios para alcanzar la eficiencia de los procesos en los laboratorios de producción, renunciando así a la forma tradicional de inspeccionar y corregir la no eficiencia.

La función de control surge como un requisito obligado para evaluar el resultado de las decisiones delegadas por los responsables de los proyectos y esto contribuye a mejorar la cultura y el entorno de gestión caracterizado por estimular y aunar los esfuerzos individuales de los estudiantes que participan en la ejecución del proyecto. Este último aspecto pocas veces se asocia al control, sin embargo es uno de los aspectos fundamentales que promueven los sistemas modernos de gestión, reflejando el tránsito de personal controlado a involucrado, dando el éxito a la herramienta de gestión.

1.1.5 Monitoreo.

Un elemento que se debe tener en cuenta es el proceso de monitoreo, el cual constituye un componente básico para cualquier herramienta de gestión. Monitoreo no es más que “el proceso continuo y sistemático mediante el cual se verifica la eficiencia y la eficacia de un proyecto mediante la identificación de sus logros y debilidades y en consecuencia, se recomienda medidas correctivas para optimizar los resultados esperados del proyecto. Es por tanto, condición para la rectificación o profundización de la ejecución y para asegurar la retroalimentación entre los objetivos, presupuestos teóricos y las lecciones aprendidas a partir de la práctica”. (CINTERFOR, 1996)

Para que el proceso de monitoreo tenga éxito debe tener un mecanismo de recopilación de datos e información sobre las actividades que se desean monitorear, para ayudar a la toma de decisiones y realizar acciones al respecto. El modelo que se propone permite avanzar del enfoque retrospectivo al proactivo, orientado al futuro, de la poca implicación del estudiante a la alta implicación, del intensivo en mano de obra a la optimización de los recursos, para lograr finalmente salir de la información redundante y tediosa a la automatización de la información y servicios compartidos, consolidando así la imagen de la UCI como líder en la informatización de la sociedad cubana.

1.2 Objeto de Estudio

1.2.1 Descripción General

A nivel mundial la producción de software es un aspecto muy importante en el desarrollo económico, social y científico-técnico para todos los países, Cuba no está ajena a este proceso y dando pasos agigantados trata de convertir la industria de software en la principal rama de su economía. La UCI juega un papel fundamental en este tema, como universidad su objetivo principal es formar profesionales capacitados para el desarrollo de software y elevar el país a niveles superiores en el mercado mundial.

Esta institución está conformada actualmente en su sede central por 9 facultades, donde la producción de software es una tarea muy importante para la universidad. Este proceso logrará aumentar la informatización de las diferentes esferas de la economía del país y alcanzar un mayor desarrollo tecnológico.

La informatización de las facultades y junto con esto, la automatización de los procesos de gestión y control del uso correcto de las computadoras en los laboratorios de producción, contribuirá a aumentar la calidad de los software y lograr un monitoreo constante de las actividades que se realizan en cada uno de estos laboratorios.

1.2.2 Descripción actual del dominio del problema

Los jefes de departamentos, junto con el Jefe del Centro de Producción de la facultad son los máximos responsables de las tareas que se llevan a cabo en cada uno de los laboratorios de producción. Los estudiantes son vinculados a los diferentes proyectos productivos, se le asignan determinadas actividades y son controlados por profesores designados para verificar su correcto desempeño en las mismas.

Los laboratorios de producción son destinados principalmente para el desarrollo de software y también son utilizados por los estudiantes para realizar actividades docentes. Dichos laboratorios son controlados, con el objetivo de lograr un mayor aprovechamiento del uso de la tecnología y en la ejecución de las tareas productivas para las cuales están destinados.

Durante el proceso de control se realizan reportes por parte de los encargados de desempeñar esta tarea, la información gestionada es entregada al líder del proyecto, esta es revisada y almacenada en el repositorio del proyecto o en otro sitio destinado para este fin; verificando de esta forma, la existencia o no de anormalidades por parte del comportamiento de los estudiantes en su puesto de trabajo.

1.2.3 Situación Problemática

Todos los estudiantes que están vinculados a proyectos productivos tienen un horario docente-productivo en los laboratorios de producción, en numerosas ocasiones este horario es violado y no se realizan las actividades orientadas por la dirección del proyecto, esta deficiencia se refleja claramente, pues existe una gran cantidad de estudiantes que no aprovechan al máximo su tiempo de máquina asignado para producir y en su lugar practican otras actividades que no están autorizadas, como es el caso del juego, las series y las películas entre muchas otras que no representan ningún tipo de beneficio para la facultad.

La utilización de las computadoras de estos laboratorios de forma incorrecta influye de forma negativa en la producción, así como también en el cumplimiento de las fechas fijadas en los

cronogramas de dichos proyectos y también afectan muy de cerca a la economía cubana, pues la tecnología que el gobierno y el país dedica a la producción en general no está siendo aprovechada al máximo, existe también una elevación en el consumo energético en todos los laboratorios de forma innecesaria.

La facultad 9 no cuenta con un mecanismo factible para monitorizar y controlar las actividades que se realizan en los laboratorios de producción. En la actualidad esta acción se ejecuta de forma manual mediante un profesor designado por el jefe de proyecto, este profesor controla el laboratorio en distintos horarios y guarda la información en un documento digital.

Existen muchas deficiencias en la ejecución de esta labor, pues en numerosas ocasiones los estudiantes conocen al profesor o a la persona encargada de realizar el control, además también es conocido con bastante precisión el horario en que el profesor pasa por los laboratorios y se hace muy difícil tener un verdadero registro de cuáles estudiantes verdaderamente aprovechan el horario asignado para tareas productivas. Otra de las deficiencias es que no se controla con la mayor seriedad por parte del personal encargado y estas deficiencias dan al traste con la calidad de los productos que son desarrollados, pues no son de la mayor calidad posible.

1.3 Análisis de otras soluciones existentes

Actualmente en la UCI el proceso de gestión del uso de las computadoras en los laboratorios de producción se realiza de forma manual, lo cual resuelve el problema, pero de forma parcial y con muchas deficiencias, en el país no se ha construido ninguna aplicación informática que pueda resolver el proceso anteriormente planteado, y a nivel mundial existen varias aplicaciones que resuelven parcialmente el objetivo propuesto de la investigación:

Ejemplo de ellas está: PC **Activity Monitor (PC Acme)**, realiza un monitoreo de todos los usuarios que interactúan con la computadora, guardando esta información en un archivo log encriptado. Reporta la actividad que realizan los usuarios como los datos de los programas que se están

ejecutando en ese momento. Los archivos log pueden ser convertidos a formato HTML[1] para obtener una vista más conveniente con el navegador. Esta herramienta es fácil de burlar por parte de los usuarios, ya que brinda una opción en una ventana mediante la cual se puede desactivar la opción de monitoreo, no realiza un control de toda la red, solo de la computadora donde esté instalada.

Otra aplicación encontrada es: **iMonitorPC BussinesPlus**, registra la actividad de los usuarios, los programas usados, sitios web visitados, capturas de pantalla, permite bloquear sitios web, realiza informes de la actividad de los usuarios en la computadora. Esta herramienta consume muchos recursos, se necesita una máquina con más de 1 GB de memoria RAM, es propietaria y requiere que se actualice constantemente la licencia.

La aplicación **NetSupport Manager** es una solución de control remoto de PCs que permite la gestión remota de ordenadores y servidores a través de redes locales LAN e Internet. Combina el control remoto con herramientas para realizar inventario de software y hardware de las máquinas remotas, ejecutar y terminar aplicaciones de forma remota, gestionar los servicios de Windows y transferir ficheros a múltiples ordenadores de manera simultánea.

Uno de los problemas que presentan todas estas aplicaciones es que son privativas, actualmente en el país y en la UCI se está abogando fuertemente por la utilización de software libre, ya que el software propietario incluye grandes gastos en soporte técnico y licencias. Otro de los problemas es que la mayoría de estas aplicaciones consumen muchos recursos de hardware en general y eso delimita el trabajo de los usuarios, principalmente en los laboratorios docentes-productivos.

El principal problema que presentan estas aplicaciones es que de una forma u otra realizan la gestión y monitorización de algunas funcionalidades con respecto a los ordenadores, pero ninguna

[1] **HTML**, siglas de ***HyperText Markup Language*** (*Lenguaje de Marcado de Hipertexto*), es el lenguaje de marcado predominante para la elaboración de páginas web.

logra tener un control total de los procesos que se requieren para gestionar el uso de computadoras en los laboratorios de producción de la facultad 9.

1.4 Conclusiones Parciales

El capítulo **Fundamentación Teórica**, ofrece información general de cómo ocurre el proceso descrito en el objeto de estudio, visualizando el posible problema y variables a estudiar, también permite la búsqueda y estudio de la información referente al problema, consulta a expertos y revisión bibliográfica en todas las fuentes disponibles adquiriendo así mayor dominio del problema científico a investigar, el establecimiento de un marco teórico ayuda a prevenir y a minimizar posibles errores que pueden presentarse durante el estudio, al formular este capítulo queda orientado el modo de conducir la investigación. Además se brinda un mayor entendimiento de los restantes capítulos y se realiza un análisis de otras soluciones existentes de este tipo, explicando sus desventajas con respecto a la aplicación que se quiere desarrollar y resaltando la importancia que representa los procesos de gestión del uso de las computadoras de los laboratorios de la facultad 9.

"…*hoy la falta de control es uno de los problemas más agudos a resolver*…"

Cr. Armando Pérez Betancourt

Secretario Ejecutivo del Grupo Gubernamental

Comité Ejecutivo del Consejo de Ministros.

CAPÍTULO 2: Tendencias y Tecnologías actuales

Durante los últimos años se ha reflejado un elevado avance en el tema de las tendencias y tecnologías actuales en el desarrollo de la informática, lo que ha traído consigo una gran variedad de estas para el desarrollo de software. Por esta razón se debe hacer un profundo análisis para poder escoger correctamente todas las herramientas que se van a utilizar, siempre pensando en las necesidades del cliente y tratando de utilizar las tecnologías más novedosas.

2.1 ¿Cuál escoger? ¿Software libre ó Software Privativo?

Ante el grado de libertad del software actualmente existe un conflicto en cuanto cuál se va a utilizar entre software libre o privativo, es por eso que se debe hacer un minucioso estudio de ambas tendencias y así determinar cual se va a utilizar, atendiendo también a las necesidades de los clientes. A continuación se presentará una breve explicación de ambas tendencias.

2.1.1 Software Libre

Primeramente se debe aclarar que software libre no es lo mismo que software gratuito, es decir, este último no cuesta nada, lo que no lo hace software libre, pues el tema no se trata sobre el precio sino sobre la libertad. El software libre puede ser distribuido, modificado, copiado y usado sin ningún tipo de problema y además viene con el código fuente incluido lo que caracteriza su libertad. Por lo tanto cualquier usuario puede modificarlo según sus necesidades, redistribuirlo e incluso hasta publicar esta aplicación ya modificada.

Richard Stallman se refiere a cuatro clases especiales de libertad para los diferentes usuarios de este tipo de software los cuales son importantes y se enumeran a continuación:

- ✓ Libertad para ejecutar sea cual sea nuestro propósito.
- ✓ Libertad para estudiar el funcionamiento del programa y adaptarlo a tus necesidades.
- ✓ Libertad para redistribuir copiar y ayudar así a tu vecino.

- ✓ Libertad para mejorar el programa y luego publicarlo para el bien de toda la comunidad.

2.1.2 Software Propietario:

El software propietario ó privativo como también es llamado está caracterizado por varias limitaciones que les presenta a sus usuarios en el tema de la reproducción (ya sea con o sin modificación alguna), modificación y el uso del mismo, también otra dificultad que exhibe es que su código fuente no es accesible o no está disponible, es decir este restringido solo a sus autores o compañías que lo construyen.

En el software propietario su autor o las instituciones que lo construyen son los que poseen todo el derecho sobre él, siendo así los únicos en poder modificarlos o redistribuirlos e impidiéndoles a los usuarios que los utilizan para realizar otras acciones, tales como estudiar su funcionamiento (ver código fuente), modificarlo y mejorarlo a sus necesidades y publicar estas mejoras.

2.1.3 Tipo de Software escogido: Software Libre

Reflejadas estas características se va a optar por la opción del software libre, principalmente porque en estos momentos el país está defendiendo fuertemente la utilización de este tipo de tecnología, además se adapta perfectamente a las exigencias del cliente y por otra parte va a permitir desarrollar en conjunto con varios entornos de desarrollos que admiten la realización de aplicaciones eficientes en los distintos sistemas operativos, aunque se debe tener en cuenta que posiblemente también se utilice herramientas de software propietario, pues algunas de estas son las más completas, se ajustan con una mayor aceptación y su trabajo es más cómodo y eficiente.

2.2 Metodologías de desarrollo de Software.

Una metodología es un proceso que engloba procedimientos, técnicas, documentación y herramientas que se utilizan en la creación de un producto de software. Se va indicando paso a paso lo que se debe hacer para lograr un producto informático. Además se debe especificar las personas que van a participar en el proceso así como el papel que van jugar en él. Actualmente están definidas en dos grandes grupos, las metodologías estructuradas y las no estructuradas.

En las metodologías estructuradas se encuentran las orientadas a procesos, orientadas a datos y las mixtas, en la otra clasificación se pueden ver las orientadas a objetos y los sistemas de tiempo real.

También existe otra clasificación, están las metodologías pesadas, son aquellas con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado (RUP)[2], del mismo modo existen las metodologías ágiles, estas más orientadas a la generación de código con ciclos muy cortos de desarrollo, se dirigen a equipos de desarrollo pequeños, hacen especial hincapié en aspectos humanos asociados al trabajo en equipo e involucran activamente al cliente en el proceso (XP)[3].

2.2.1 Proceso Unificado de Software (RUP)

El proceso unificado conocido como RUP, es un modelo de software que permite el desarrollo de software a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad. Aunque con el inconveniente de generar mayor complejidad en los controles de administración del mismo. Sin embargo, los beneficios obtenidos recompensan el esfuerzo invertido en este aspecto. Este proceso de desarrollo de software lo conforman el conjunto de actividades necesarias para trasformar los requisitos funcionales de un usuario en un producto de software.

Lo que sustenta el proceso de desarrollo de software son: el proyecto, las personas, el producto y el proceso, existe una estrecha relación entre ellas. Es conocido como las cuatro P en el desarrollo del software. RUP define para cada etapa: el flujo de trabajo, los trabajadores que intervienen, las actividades que realizan y los artefactos que se necesitan o producen. Su meta es asegurar la producción de software con la más alta calidad, que cumpla con las necesidades de los usuarios dentro del cronograma planeado y la inversión prevista.

---

[2] Rational Unified Process: en idioma español, Proceso Unificado de Rational.
[3] Extreme Programming: en idioma español, Programación Extrema.

Además, el Proceso Unificado utiliza el Lenguaje Unificado de Modelado (UML)[4] para expresar gráficamente todos los esquemas de un sistema software y los aspectos más importantes que se definen en esta metodología: es iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura.

RUP se divide en 4 fases:

Inicio: El objetivo de esta etapa es determinar la visión del proyecto.

Elaboración: El objetivo es determinar la estructura óptima.

Construcción: El objetivo es obtener la capacidad operacional inicial.

Transición: El objetivo es obtener la primera versión del proyecto (release).

2.2.2 Programación Extrema (XP)

La metodología XP es una de las más populares actualmente. Sus principales objetivos son muy sencillos de entender, primeramente se trata de dar al cliente el software que él necesita y cuando lo necesita, por tanto se debe responder muy rápido a las necesidades del cliente, por tanto se tiene mucho coraje para enfrentar los cambios, incluso cuando sean a finales del ciclo de programación. Como segundo se tiende a fomentar grandemente el trabajo en equipo, tanto los jefes de proyectos, los clientes y los desarrolladores son parte del grupo y están fuertemente involucrados en el desarrollo de software.

2.2.3 Metodología de Desarrollo seleccionada: RUP

La metodología que más se adapta a las condiciones de la aplicación que se quiere desarrollar es RUP, puesto que recoge un grupo de buenas prácticas de muchas otras metodologías y la organización del trabajo es un aspecto fundamental, además la documentación que se genera es de

[4] UML :Lenguaje de Modelado Unificado

gran apoyo a las siguientes iteraciones que se van a desarrollar en un futuro. Esta metodología está preparada para desarrollar proyectos complejos gracias a las características que presenta, como por ejemplo:

Dirigido por casos de uso: Basándose en los casos de uso, los desarrolladores crean una serie de modelos de diseño e implementación que los llevan a cabo. Además, estos modelos se validan para que sean conformes a los casos de uso. Finalmente, los casos de uso también sirven para realizar las pruebas sobre los componentes desarrollados.

Centrado en la arquitectura: En la arquitectura de la construcción, antes de construir un edificio éste se contempla desde varios puntos de vista: estructura, conducciones eléctricas, fontanería, etc. Cada uno de estos aspectos está representado por un gráfico con su notación correspondiente. Siguiendo este ejemplo, el concepto de arquitectura software incluye los aspectos estáticos y dinámicos más significativos del sistema.

Iterativo e incremental: Todo sistema informático complejo supone un gran esfuerzo que puede durar desde varios meses hasta años. Por lo tanto, lo más práctico es dividir un proyecto en varias fases. Actualmente se suele hablar de ciclos de vida en los que se realizan varios recorridos por todas las fases. Cada recorrido por las fases se denomina iteración en el proyecto en la que se realizan varios tipos de trabajo (denominados flujos). Además, cada iteración parte de la anterior incrementado o revisando la funcionalidad implementada. (Jacobson, y otros, 2000)

2.3 Lenguaje Unificado de Modelado (UML)

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y

medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. (James Rumbaugh, 1997)

UML incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos, además es el lenguaje de modelaje de sistemas software más utilizado y conocido en la actualidad.

2.4 Sistema Gestor de Base de Datos (SGBD)

De forma sencilla un sistema de gestión de bases de datos (SGBD) se puede definir como un conjunto de programas dedicados para acceder a una colección de datos muy bien interrelacionadas. “Los sistemas de Gestión de Bases de Datos, son aplicaciones que permiten a los usuarios definir, crear y mantener la base de datos y proporciona un acceso controlado a la misma”. (Elmasri, 1997)

2.4.1 MySQL

El sistema de base de datos operacional MySQL es hoy en día uno de los más importantes. Cuenta con millones de aplicaciones y aparece en el mundo informático como una de las más utilizadas por usuarios del medio. El programa MySQL se usa como servidor a través del cual pueden conectarse múltiples usuarios y utilizarlo al mismo tiempo. Las plataformas que utiliza son de variados tipos y entre ellas se puede mencionar LAMP, MAMP, SAMP, BAMP y WAMP (aplicables a Mac, Windows, Linux, BSD, Open Solaris, Perl y Phyton entre otras). (Victoria, 2009)

MySQL es un software de código abierto, licenciado bajo la GPL de la GNU[5], aunque MySQL AB distribuye una versión comercial, en lo único que se diferencia de la versión libre, es en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de otra manera, se vulneraría la licencia GPL.

Las principales características de este gestor de bases de datos son las siguientes:

- ✓ Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
- ✓ Soporta gran cantidad de tipos de datos para las columnas.
- ✓ Dispone de API's[6] en gran cantidad de lenguajes (C, C++, Java, PHP, etc.).
- ✓ Gran portabilidad entre sistemas.
- ✓ Soporta hasta 32 índices por tabla.
- ✓ Gestión de usuarios y contraseñas, manteniendo un muy buen nivel de seguridad en los datos. (Pecos, 2010)

2.4.2 PostgreSQL

PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas características que tradicionalmente sólo se podían ver en productos comerciales de alto calibre. Es un Sistema Gestor de Base de Datos que ha sido desarrollado desde 1977. Comenzó como un proyecto en la Universidad Bekerley de California. El proyecto PostgreSQL sigue actualmente un activo proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuidores de código abierto. Está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas

[5] La **Licencia Pública General de GNU** o más conocida por su nombre en inglés ***GNU General Public License*** o simplemente su acrónimo del inglés **GNU GPL**, es una licencia creada por la Free Software Foundation en 1989 (la primera versión), y está orientada principalmente a proteger la libre distribución, modificación y uso de software.

[6] Interfaz de programación de aplicaciones (API por sus siglas en inglés) Conjunto de funciones residentes en ficheros que permiten que una aplicación corra bajo determinado sistema operativo.

características que tradicionalmente sólo se podían ver en productos comerciales de alto calibre. (eAprende, 2008)

PostgreSQL puede funcionar en múltiples plataformas (en general, en todas las modernas basadas en Unix) y, a partir de la versión 8.0 también en Windows de forma nativa.

Tiene amplias características, como son:

- ✓ DBMS Objeto-Relacional:
  PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transacciones, optimización de consultas, herencia, y arreglos.
- ✓ Altamente Extensible:
  PostgreSQL soporta operadores, funciones métodos de acceso y tipos de datos definidos por el usuario.
- ✓ Soporte SQL Comprensivo:
  PostgreSQL soporta la especificación SQL99[7] e incluye características avanzadas tales como las uniones (joins) SQL92.
- ✓ Cliente/Servidor:
  PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

[7] Open Database Connectivity, es un estándar de acceso a Base de Datos.

2.4.3 SGBD seleccionado: PostgreSQL

Por la investigación realizada anteriormente se propone como sistema gestor de base de datos, PostgreSQL 8.2, dado que sus ventajas y características se ajustan al problema presentado, además es un sistema muy potente y multiplataforma, destacando que es de código abierto, publicado bajo la licencia BSD, que pertenece al grupo de licencias de software libre. Además utiliza control de versionado concurrente y soporte para la implementación de consultas complejas. Con este SGBD se podrá mantener una base de datos actualizada, confiable y configurable.

2.5 Herramienta CASE[8]

El concepto de CASE es muy amplio; y una buena definición genérica, que pueda abarcar esa amplitud de conceptos, sería la de considerar a la Ingeniería de Software Asistida por Computación (CASE), como la aplicación de métodos y técnicas a través de las cuales se hacen útiles a las personas comprender las capacidades de las computadoras, por medio de programas, de procedimientos y su respectiva documentación. (Perissé, 2001)

Algunos de los componentes de las herramientas CASE permiten:

- ✓ Confeccionar la definición de requerimientos de los usuarios.
- ✓ Mejorar el diseño de los sistemas.
- ✓ Mejorar la eficiencia en la programación (por su generación automática de códigos).
- ✓ Otorgar a la administración un mejor soporte en la documentación.

2.5.1 Visual Paradigm

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones

[8] CASE es una sigla, que corresponde a las iniciales de: **C**omputer **A**ided **S**oftware **E**ngineering; y en su traducción al Español significa Ingeniería de Software Asistida por Computación.

con calidad, a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

Algunas características de esta herramienta:

- ✓ Soporte de UML.
- ✓ Ingeniería de ida y vuelta.
- ✓ Ingeniería inversa - Código a modelo, código a diagrama.
- ✓ Generación de código - Modelo a código, diagrama a código.
- ✓ Editor de Detalles de Casos de Uso - Entorno todo-en-uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso.
- ✓ Generación de bases de datos - Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- ✓ Ingeniería inversa de bases de datos - Desde Sistemas Gestores de Bases de Datos (DBMS) existentes a diagramas de Entidad-Relación.

2.5.2 Rational Rose Enterprise

Rational Rose Enterprise es el producto más completo de la familia Rational Rose. Todos los productos Rational Rose incluyen soporte (UML™). Rational Rose Enterprise es una buena elección para el ambiente de modelado que soporte la generación de código a partir de modelos en Ada, ANSI C++, C++, CORBA, Java™/J2EE™, Visual C++ y Visual Basic. Como todos los demás productos Rational Rose, proporciona un lenguaje común de modelado para el equipo que facilita la creación de software de calidad más rápidamente. (Grupo Soluciones GSInnova, 2007)

Algunas de las principales características que tiene incluida esta herramienta:

- ✓ Soporte para análisis de patrones ANSI C++, Rose J y Visual C++ basado en "Design Patterns: Elements of Reusable Object-Oriented Software".

- ✓ Característica de control por separado de componentes modelo que permite una administración más granular y el uso de modelos.
- ✓ Soporte de ingeniería Forward y/o reversa para algunos de los conceptos más comunes de Java 1.5.
- ✓ La generación de código Ada, ANSI C ++, C++, CORBA, Java y Visual Basic, con capacidad de sincronización modelo- código configurables.
- ✓ Soporte Enterprise Java Beans™ 2.0.
- ✓ Capacidad de análisis de calidad de código.
- ✓ El Add-In para modelado Web provee visualización, modelado y las herramientas para desarrollar aplicaciones de Web.
- ✓ Modelado UML para trabajar en diseños de base de datos, con capacidad de representar la integración de los datos y los requerimientos de aplicación a través de diseños lógicos y físicos.

2.5.3 Herramienta CASE seleccionada: Visual Paradigm

La herramienta Rational Rose es una herramienta muy completa y de gran aceptación en el mundo actual, pero al ser una herramienta privativa hace imposible escogerla para la realización de esta aplicación, por tanto se escoge el Visual Paradigm, herramienta verdaderamente potente, como lo demuestran sus características anteriormente mencionadas, además es una herramienta que actualmente presenta una gran aceptación mundial y especialmente por los estudiantes y desarrolladores de la UCI.

Esta herramienta es multiplataforma y presenta una licencia comercial porque se utiliza para modelar varias metodologías de desarrollo, favorece mucho la agilización del trabajo de los desarrolladores y otra razón de la selección es que actualmente existen una gran cantidad de información, tutoriales y videos que ayudan a su aprendizaje fácilmente, específicamente la versión Visual Paradigm para UML 6.4 Enterprise Edition.

2.6 Lenguajes de Programación

Para posibilitar la comunicación entre las computadoras y los humanos se hace necesario de la existencia de los lenguajes de programación para lograr tal interactividad. Un lenguaje de programación está conformado por una serie de reglas sintácticas y semánticas que serán utilizadas por el programador y a través de las cuales creará un programa o subprograma; las instrucciones que forman dicho programa son conocidas como código fuente. (Lanzillotta, 2004)

Los lenguajes de programación son clasificados en tres grupos según su nivel de abstracción o complejidad:

- ✓ Lenguajes bajo nivel: Es el único que entiende directamente la computadora, utiliza el alfabeto binario que consta de los dos únicos símbolos 0 y 1, denominados bits (abreviatura inglesa de dígitos binarios). Fue el primer lenguaje utilizado en la programación de computadoras, pero dejó de usarse por su nivel de dificultad, se conoce como lenguaje de máquina.
- ✓ Lenguajes de medio nivel: Es una composición entre el lenguaje de máquina y el lenguaje natural.
- ✓ Lenguajes de alto nivel: Logran la independencia del tipo de máquina y se acercan al lenguaje natural. (http://www.todo-programacion.com.ar, 2005)

Los lenguajes de alto nivel permiten la ejecución de programas creados por lenguajes próximos al habla natural y se puede enviar la ejecución de sentencias a los ordenadores de forma fácil y entendible. Los principales lenguajes de este tipo son C++, C#, Java, Python, entre otros.

A continuación se realizará una caracterización de los lenguajes más utilizados actualmente.

2.6.1 C++

Es un lenguaje hibrido basado en C, que se puede compilar y su principal característica es el soporte para programación orientada a objetos (abstracción, encapsulación y polimorfismo), pero añade otras propiedades como el soporte de plantillas o programación genérica (templates), la

posibilidad de redefinir operadores y la identificación de tipos en tiempo de ejecución. (Almeida Rodríguez, 2002)

Una de las particularidades del C++ es la sobrecarga de operadores. La sintaxis de clases y objetos permite manipular diversas estructuras de datos y operaciones; las excepciones permiten procesar de un modo claro los casos de errores, elevando de esta forma su expresividad. (AmericaTI.com, 2006)

C++ es un lenguaje popular, utilizado actualmente para resolver diferentes tipos de problemas desde los más simples hasta los más complejos, su código puede ser copilado en varias plataformas. Incorpora una nueva característica que permite aumentar la eficiencia en cuanto a la llamada de funciones, dicha característica es la posibilidad de escribir funciones en línea (inline); inline es una petición al compilador para sustituir la llamada a la función directamente por su código, es decir, genera un código en línea.

Los constructores y destructores son funciones miembros usadas para crear y destruir instancias. Este lenguaje permite la conversión de tipo implícito, la función inlining, darle nombres a las funciones, tener argumentos de funciones por defectos y pasarlos por referencias. (Bjarne Stroustrup, Addison-Wesley, 1991)

En C++ una clase es sintácticamente una estructura con funciones de miembro de un tipo definida por el usuario. Abarca los tres paradigmas de programación: la programación orientada a objetos, la programación genérica y la programación estructurada.

2.6.2 C SHARP (C#)

Es un lenguaje desarrollado y estandarizado por Microsoft, diseñado para la construcción de una amplia gama de aplicaciones empresariales que se ejecutan en .NET Framework. C# es introducido como Visual C# en Visual Studio .NET suite, soporte para Visual C# incluye plantillas de proyectos, diseñadores, páginas de propiedades, asistentes de código, un modelo de objetos, y otras

características del entrono de desarrollo. La biblioteca de Visual C# es el de .NET Framework. (Microsoft Corporation, 2010)

El código de C# se compila como código administrado, lo cual significa que utiliza los beneficios de los servicios de Common Language Runtime, estos servicios incluyen el idioma interoperabilidad, la mejora de la seguridad, el apoyo y la mejora de versiones.

C# es un lenguaje de programación orientado a objetos que coge las mejores características de los lenguajes preexistentes como Visual Basic, Java o C++ y las combina como uno solo.

Entre las principales características se encuentra la facilidad de uso, el ambiente de trabajo es muy cómodo, amigable y clásico en las aplicaciones de Windows. Permite la administración de memoria, la seguridad en cuanto al manejo de datos, es compatible con otros lenguajes y permite el uso de operadores. C# fue diseñado para combinar el control a bajo nivel de lenguajes como C y la velocidad de programación de lenguajes como Visual Basic.

2.6.3 Java

Es un lenguaje de programación desarrollado por Sun Microsystems a principios de los años 90; soporta las tres características del paradigma de programación orientado a objetos: herencia, encapsulamiento y polimorfismo. Trabaja con sus datos como objetos y con interfaces a esos objetos. Una interfaz de red permite que el código compilado sea enviado a través de Internet y ejecutado remotamente, lo que permite a los usuarios añadir sus programas a páginas web. Utiliza técnicas de autenticación basada en cifrado de clave pública. (Gosling, 1995)

La tecnología java está constituida por dos componentes básicos: el lenguaje java y su plataforma, la máquina virtual de java (Java Virtual Machine), fue diseñado para construir software altamente seguro, proporciona muchas comprobaciones en compilación y en tiempo de ejecución. (Álvares Marañón, Gonzalo, 2006)

Dentro de sus principales características se encuentra que es un lenguaje orientado a objetos, es independiente su plataforma, es decir puede ejecutarse en cualquier sistema operativo que posea

su máquina virtual instalada. Además es extremadamente robusto ya que no permite el manejo directo de la memoria ni del hardware.

Sun Microsystems liberó gran parte de sus tecnologías java bajo la licencia GNU GPL entre noviembre 2006 y mayo 2007, de acuerdo con las especificaciones de Java Community Process, de manera tal que casi todo el Java de Sun es software libre.

2.6.4 Lenguaje de programación seleccionado: C++

Para la selección del lenguaje a utilizar en el desarrollo del sistema con las características requeridas, se tuvieron en cuenta las siguientes funcionalidades que ofrece el lenguaje de programación C++:

- ✓ Es un lenguaje de programación muy utilizado para el desarrollo de aplicaciones de escritorio.
- ✓ La posibilidad de orientar la programación a objetos permite al programador diseñar aplicaciones desde un punto de vista más cercano a la vida real; permite la reutilización del código de una forma más lógica y productiva.
- ✓ Eficiencia: Es uno de los lenguajes más rápidos en cuanto a ejecución. (Luján Mora, 2006)
- ✓ C++ contiene una extensa cantidad de librerías que evitan la re implementación de operaciones comunes.
- ✓ Herramientas: Existen multitud de compiladores para C++ en los sistemas operativos y en todos los casos existe más de un ejemplar con licencia libre. (Luján Mora, 2006)
- ✓ Portabilidad: Es un lenguaje multiplataforma sus aplicaciones solamente tienen que ser recompiladas nuevamente para adaptarse al sistema operativo, a diferencia de otros lenguajes como Java que necesita de una máquina virtual, lo que hace el proceso mucho más lento. El lenguaje está estandarizado y un mismo código fuente se puede compilar en diversas plataformas. (Luján Mora, 2006)

2.7 Entrono de Desarrollo Integrado (Integrated Develoment Enviroment IDE)

Los IDEs generalmente están compuestos por un compilador, un depurador, un editor de código y un constructor de interfaz de usuario, son creados para el desarrollo de aplicaciones en un solo lenguaje de programación; sin embargo, existen algunos, en los que se puede desarrollar en más de un lenguaje. Un IDE no es más que un conjunto de herramientas de desarrollo de software.

Cuando el lenguaje es orientado a objetos, los IDEs incluyen navegador de clases, inspector de objetos y un diagrama de jerarquías de clases. En los siguientes acápites se dará una breve descripción de algunos de los IDEs que permiten el desarrollo de aplicaciones en C++.

2.7.1 Qt Creator

Es un IDE para la realización de aplicaciones con las bibliotecas Qt, creado por la compañía noruega Trolltech (actualmente QT Software). En marzo del 2009 liberó su primera versión estable, teniendo gran aceptación, esta versión posee la licencia GPL permitiendo el desarrollo de aplicaciones con fines comerciales. Puede ser ejecutado en los sistemas operativos desde Windows 98 hasta Windows XP y Vista, requiriendo el compilador MinGW incluido en la instalación del IDE, Linux 2.6.x, para versiones de 32 y 64 bits con Qt 4.x instalado, además hay una versión para Linux con gcc[9] 3.3, Solaris, HP-UX, Mac OS X 10.4 o superior, requiriendo Qt 4.x; puede ser soportado por otras versiones de Unix con X11(X- sistemas de ventanas para dotar a los sistemas Unix de interfaz gráfica, 11-version 11).

2.7.2 Microsoft Visual Studio

Visual Studio es un conjunto de herramientas de desarrollo para la generación de aplicaciones web ASP.NET, Servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Visual Basic, Visual C# y Visual C++ utilizan todos el mismo entorno de desarrollo integrado, que habilita el uso compartido de herramientas y hace más sencilla la creación de soluciones en varios lenguajes.

[9] gcc es un comando usado para ejecutar el compilador de C.

Asimismo, dichos lenguajes utilizan las funciones de .NET Framework, las cuales ofrecen acceso a tecnologías clave para simplificar el desarrollo de aplicaciones web ASP y Servicios Web XML. (Microsoft Corporation, 2007) Es un IDE desarrollado por Microsoft, su última versión estable registrada es Visual Studio 2008.

2.7.3 IDE seleccionado: Qt Creator

Se decidió utilizar Qt Creator para el desarrollo del sistema, es una buena práctica para la elaboración de aplicaciones de escritorio y da soporte al lenguaje de programación C++ (seleccionado para el desarrollo). Ofrece gran compatibilidad con varios sistemas operativos, posee riquezas en sus funciones y librerías permitiendo el desarrollo de interfaces amigables y por su alto rendimiento con C++.

Qt tiene la característica de presentar el código fuente disponible, posee gran documentación y calidad en cuanto a soporte técnico y una comunidad de desarrollo que la da mantenimiento, ha sido adaptado a las características de los entornos de desarrollo actuales y está pensado para sistemas operativos recientes.

2.8 Conclusiones Parciales

En el capítulo se realizó un completo análisis de las principales tendencias y tecnologías actuales, definiendo aquí las herramientas que serán utilizadas para desarrollar la aplicación planteada. Teniendo siempre en cuenta que la investigación fue orientada al software libre y por su puesto a las tendencias y tecnologías más adecuadas con respecto a las necesidades del cliente. Terminado el presente capítulo se comenzará a desarrollar la propuesta del sistema para la aplicación informática anteriormente mencionada.

CAPÍTULO 3: Presentación de la solución propuesta

En el presente capítulo se mostrará una propuesta de la solución, así como lo referente a las principales clases y conceptos relacionados con el modelo del dominio, los actores del sistema, una descripción de los casos de uso más significativos, todo esto a partir de un estudio inicial del problema, con el objetivo de que exista un mejor entendimiento del producto que se desea desarrollar.

3.1 Modelo del Dominio

El Modelo del Dominio captura los tipos de objetos más importantes en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabaja el sistema; con el objetivo de comprender y describir las principales clases dentro del contexto del sistema. (Jacobson, y otros, 2000)

A continuación se describen las clases más importantes del sistema para obtener una mayor comprensión de los eventos que suceden en el entorno en el que se trabaja.

3.1.1 Glosario de términos asociados al dominio

Con el objetivo de lograr un mayor conocimiento del modelo que se plantea, se hace necesario de la utilización de un glosario de términos donde se definen los principales conceptos a utilizar.

Laboratorio: es el local constituido por un grupo de computadoras destinadas a la producción de software.

Usuario: persona que interactúa con una o varias computadoras en los laboratorios.

Profesor: persona encargada de controlar a los usuarios y de verificar la disponibilidad de las computadoras en los laboratorios de producción.

Computadora: medio físico utilizado por los usuarios para realizar tareas docente-productivas en los laboratorios.

3.2 Requisitos Funcionales y No Funcionales

En este acápite se muestra una lista de requisitos funcionales y no funcionales que presenta el sistema planteado.

3.2.1 Requisitos Funcionales

Un requerimiento funcional especifica una acción que debe ser capaz de realizar el sistema, sin considerar restricciones físicas; especifica además comportamiento de entrada/salida de un sistema. (Jacobson, y otros, 2000)

A continuación se listan los referentes al sistema a desarrollar:

R.1 Gestionar instrucciones.

R.1.1 Recibir instrucciones.

El sistema cliente debe recibir instrucciones de la aplicación servidora.

R1.2 Enviar Instrucciones.

El sistema servidor debe enviar Instrucciones a la aplicación cliente.

R.1.3 Ejecutar instrucciones.

El sistema cliente debe ejecutar las funcionalidades.

R.2 Gestionar datos.

R.2.1 Adicionar Datos.

El sistema cliente debe adicionar datos al servidor de Base de Datos.

R.2.2 Eliminar Datos.

El sistema servidor debe eliminar los datos en el servidor.

R.2.3 Mostrar Datos

El sistema servidor debe mostrar todos los datos guardados en la Base de Datos.

R.3 Gestionar usuario.

R.3.1 Insertar usuario.

El sistema servidor debe permitir insertar usuarios en el servidor de Base de Datos.

R.3.2 Modificar usuario.

El sistema servidor debe permitir modificar usuarios en el servidor de Base de Datos.

R.3.3 Eliminar usuario.

El sistema servidor debe permitir eliminar usuarios en el servidor de Base de Datos.

R.3.4 Mostrar Usuarios.

El sistema servidor debe permitir mostrar los usuarios registrados en el servidor de Base de Datos.

R.4 Autenticar usuario.

El sistema servidor permitirá que los usuarios se autentiquen de forma segura accediendo a las funcionalidades que les son permitidas según su rol.

R.4.1 Establecer permisos de usuario.

El sistema servidor debe permitir establecer permisos o roles a los usuarios registrados en el servidor.

R.5 Consultar Datos PC.

R.5.1 Mostrar datos guardados según un criterio de búsqueda.

El sistema servidor debe mostrar datos guardados en la Base de Datos según un criterio de búsqueda.

R.6 Gestionar funcionalidades.

El sistema cliente debe realizar todas las funcionalidades.

R.6.1 Enviar resultados al servidor.

El sistema cliente de mandar los resultados a la aplicación servidora.

R.6.2 Ejecutar otras funcionalidades.

El sistema cliente debe realizar otras funcionalidades.

3.2.2 Requisitos No Funcionales

Un requerimiento no funcional especifica propiedades del sistema, como restricciones del entorno o de implementación, rendimiento, dependencias de las plataformas, extensibilidad o fiabilidad. Este puede especificar restricciones físicas sobre un requisito funcional. (Jacobson, y otros, 2000)

A continuación se listan los referentes al sistema a desarrollar.

Requisitos No funcionales Aplicación Cliente

- ✓ **Usabilidad**
  - El sistema será usado por cualquier persona que tenga un conocimiento intermedio de computación e informática.
- ✓ **Confiabilidad y Disponibilidad**

- El sistema debe estar disponible durante todo el horario de trabajo de los usuarios finales.

✓ **Seguridad**

- La aplicación debe estar corriendo como un servicio de Windows.

✓ **Rendimiento**

- Tanto el tiempo de respuesta como la velocidad de procesamiento de la información no debe exceder de los 10 segundos.
- El tiempo de procesamientos de los ficheros no debe exceder de los 5 segundos.

✓ **Software**

- PC[10] Cliente: Tendrá sistema operativo de la plataforma WINDOWS cualquiera de sus versiones.

✓ **Hardware**

- PC Cliente: La PC cliente deberá contar con un microprocesador con una velocidad de procesamiento superior a un 1 GHz[11], una memoria RAM de 1 GB o superior, más de 40 GB de disco duro y una tarjeta de red.

Requisitos No Funcionales Aplicación Servidora

✓ **Usabilidad**

[10] Personal Computer: en idioma español, Computadora Personal
[11] Gigahercio: es un múltiplo de la unidad de medida de frecuencia hercio (Hz) y equivale a 109 Hz.

- El sistema será usado por cualquier persona que tenga un conocimiento intermedio de computación e informática.

✓ **Confiabilidad y Disponibilidad**

- El sistema debe estar disponible durante todo el horario de trabajo de los usuarios finales.

✓ **Seguridad**

- El sistema debe tener restricciones de acceso de acuerdo con los permisos establecidos para cada usuario.

✓ **Soporte**

- Deberá contar con un manual de usuario.

✓ **Rendimiento**

- Tanto el tiempo de respuesta como la velocidad de procesamiento de la información no debe exceder de los 10 segundos.

✓ **Software**

- PC Servidora: Tendrá sistema operativo de la plataforma WINDOWS cualquiera de sus versiones.

✓ **Hardware**

- PC Servidora: El servidor deberá contar con un microprocesador con una velocidad de procesamiento superior a un 2GHz, una memoria RAM de 1GB o superior, más de 100 GB de disco duro y una tarjeta de red.

- PC Servidor de Base de Datos: Debe contar con un microprocesador con una velocidad de procesamiento superior a un 2GHz, una memoria RAM de 1GB o superior, más de 100 GB de disco duro y una tarjeta de red.

3.3 Descripción del Sistema Propuesto

La propuesta de solución está compuesta por los modelos de análisis y diseño de un sistema informático de tipo escritorio, que va a controlar y gestionar las actividades que se realizan en cada computadora de los laboratorios de producción de la Facultad 9. Este sistema será operado por un administrador que va a permitir gestionar toda la información que se realiza en cada puesto de trabajo. Estará dividida en dos aplicaciones, una aplicación cliente corriendo como un proceso o servicio en cada ordenador de los laboratorios de producción y otra aplicación servidora, encargada de la gestión de toda esta información, ayudando así a tener un mayor control y conocimiento de todas las actividades que se realizan en cada computadora de dichos laboratorios.

3.3.1 Descripción de los actores del sistema

Un actor es la idealización de una persona externa, un proceso o cosa que interactúa con un sistema, subsistema, o clase. Caracteriza a las interacciones que pueden tener los usuarios con el sistema. Se puede incluir que un usuario puede llegar a desempeñarse como varios actores en tiempo de ejecución dentro del sistema, además que diferentes usuarios pueden estar reflejados en un mismo actor, y por lo tanto, representan distintas instancias de la misma definición de actor. Cada actor participa en uno o más casos de uso. (Jacobson, y otros, 2000)

Se muestran a continuación en la **Tabla 1** los actores del sistema con su respectiva descripción.

Tabla 1: Actores del Sistema y su descripción

Actor	Descripción
Administrador	Es el encargado de controlar y gestionar toda la información en los laboratorios de producción de la Facultad 9.
Usuario	Es el actor, dígase administrador que está encargado de mostrar toda la información de los laboratorios y de registrarse en el sistema.
Reloj ó Aplicación Cliente	Es un actor ficticio que cada cierto tiempo dispara el caso de uso, el cual es el encargado de gestionar las funcionalidades que van a ser ejecutadas, además de enviar los datos al servidor de Base de Datos. También debe ser capaz de mantenerse en espera de peticiones del servidor y dar respuesta a ellas.

3.4 Conclusiones Parciales

En el presente capítulo se realizó la propuesta de solución a través de un modelo del dominio, identificándose los principales conceptos involucrados en los procesos de control y gestión de las computadoras en los laboratorios de producción de la facultad 9. Se identificaron los requerimientos tanto funcionales como no funcionales del sistema. Se modeló el sistema que se va a desarrollar basado en casos de usos, lo que trae consigo que el sistema pueda marchar sin problemas por los siguientes flujos de trabajo como está expuesto en la metodología de desarrollo utilizada (RUP).

CAPÍTULO 4: Construcción de la solución propuesta

En el presente capítulo se modelará los procesos de análisis y diseño de la aplicación propuesta, basándose en los distintos patrones de diseño y arquitectura, dirigidos mediante los casos de uso identificados en la propuesta de solución del capítulo anterior. También se presentará el modelo de implementación de las aplicaciones, así como los elementos referentes al diseño de la base de datos.

4.1 Modelo de Análisis

El modelo de análisis muestra una visión general del sistema, especificando de una forma más precisa los requisitos capturados para el desarrollo de la aplicación. Durante la elaboración del modelo de análisis se identificaron las clases más importantes del análisis que describen la realización de los casos de uso. Las clases del análisis muestran unos de los tres estereotipos básicos: control, interfaz y entidad.

4.2 Modelo de Diseño

En el modelo de diseño se crean las bases para realizar las actividades de la fase de implementación, siguiendo la estructura trazada para el desarrollo del sistema en el modelo de análisis y se obtiene de tal forma que soporte tanto los requisitos funcionales como no funcionales. Se realiza la modelación del sistema siguiendo el modelo arquitectónico n-capas y cliente-servidor. La arquitectura cliente-servidor consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta, esta idea se puede aplicar a programas que se ejecuten sobre una sola computadora, pero es más ventajoso utilizar varias, donde la aplicación este distribuida que es el caso de la aplicación a desarrollar. En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

Por otra parte el objetivo principal de la programación por capas es lograr separar la lógica del negocio de la lógica de diseño, por ejemplo separar la capa de datos de la capa de presentación.

Unas de las ventajas que presenta este estilo es poder desarrollar una aplicación en diferentes niveles y en caso que ocurra algún error, solo se afecta el nivel requerido y no es necesario revisar el código involucrado en toda la aplicación.

Este modelo consta con tres capas principales:

Capa de presentación: es la que ve el usuario (también se la denomina "capa de usuario"), se encarga de recoger la información proveniente del usuario y realiza un filtrado previo para comprobar que no hay errores de formato. Esta capa se comunica únicamente con la capa de negocio. También es conocida como interfaz gráfica y debe tener la característica de ser amigable para el usuario.

Capa de negocio: es donde residen los programas que se ejecutan, aquí se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio ó lógica del negocio, porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación y con la capa de datos, con esta última para solicitar al gestor de base de datos almacenar o recuperar datos de él.

Capa de datos: es donde radican los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Además se utiliza un conjunto de patrones de diseño para tener mejor estructurado todo el sistema.

"Una arquitectura orientada a objetos bien estructurada está llena de patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles". (Jacobson, y otros, 2000)

Uno de los patrones que se utiliza es el GRASP (General Responsibility Assignment Software Patterns), en español, patrones de asignación de responsabilidades generales.

Una de los temas más complicados en la programación Orientada a Objetos consiste en elegir las clases adecuadas y decidir como estas clases deben interactuar. Es inevitable elegir cuidadosamente las responsabilidades de cada clase en la primera codificación y, fundamentalmente, en la refactorización del sistema.

Dentro del patrón GRASP se destacan 5 patrones principales, experto, creador, alta cohesión, bajo acoplamiento y controlador. Cada uno de estos patrones tiene distintas responsabilidades por separado, el patrón experto consiste en asignar una responsabilidad al experto en información, la clase que tiene la información necesaria para la realización de la asignación, un ejemplo se observa claramente en la clase LN_Fichero, donde esta clase es la que presenta toda la información necesaria para el trabajo con los ficheros, por tanto esta es la clase experta en este tema y es donde realiza todo lo referente a ficheros.

El patrón bajo acoplamiento es el encargado de garantizar que el acoplamiento de las clases permanezca bajo, lo que trae consigo un incremento en la reutilización de código y además el impacto al cambio es bajo por mantener una baja dependencia, este patrón se utiliza en la aplicación cuando se aplica la arquitectura en capas, en cada capa hay presente un conjunto de clases encargadas de realizar acciones por separado y tratando de tener la menor dependencia posible.

El patrón creador es uno de los más importantes, pues es el encargado de asignar la responsabilidad de que una clase pueda crear instancias de otras, otro de los patrones principales es alta cohesión, el encargado de mantener la complejidad de la aplicación manejable, es decir que la aplicación esté bien cohesionada y funcione toda unida, por último y no menos importante el patrón controlador que se encarga de gestionar todos los eventos que suceden en el sistema.

En la aplicación existen varias modalidades de la arquitectura n-capas, pues en ocasiones solo se necesitan 2 capas, en el caso de la aplicación cliente, no se encuentra presente la capa de presentación, esta capa no es necesaria en dicha aplicación porque nunca se va a interactuar con el usuario directamente, el sistema cliente va estar corriendo como un servicio del sistema operativo, especialmente en modo consola y cada cierto tiempo se va a ejecutar de modo silencioso y automático, con la tarea de ejecutar todas las funcionalidades programadas y enviará directamente los datos recogidos al servidor de Base de Datos.

También existe la posibilidad de que se relacione con la otra arquitectura propuesta (cliente-servidor), puesto que la aplicación servidora es capaz de mandar instrucciones, dígase comandos del sistema (CMD) hacia la aplicación cliente, la cual tiene la propiedad de ejecutar estas instrucciones y enviar una respuesta a la aplicación servidora y mostrarla mediante la capa de presentación, por esta razón queda clara la arquitectura cliente-servidor.

4.3 Principios de Diseño

4.3.1 Estándares de interfaz de la aplicación

Con el objetivo de lograr una mayor aceptación por parte de los usuarios, es necesario de la creación de aplicaciones con interfaces amigables que mejoren su usabilidad. Para tener las interfaces de comunicación con el usuario de una manera estándar y sencilla se utilizaron colores claros y en ningún momento sobrepasaría la suma de 4 colores para que el usuario fuera capaz de trabajar con la aplicación sin muchos problemas.

Para diferenciar las interfaces se tuvo en cuenta los roles de cada usuario en especifico, con el objetivo de no presentar demasiados elementos a todos los usuarios, de esta forma solo se mostraría la información referente al rol que presenta el usuario autenticado. Además con la información se tuvo en cuenta una factible organización en cada interfaz, los nombres de los elementos son sencillos y sugerentes, para así lograr una aplicación de fácil entendimiento y comprensión.

4.3.2 Estándares de codificación

Los estándares de codificación que se utilizó deben servir para lograr una mayor organización y fácil entendimiento del código, ya que la aplicación debe pasar por otras iteraciones en el futuro.

Declaraciones:

Las variables deberán tener un nombre sencillo y si es compuesto debe estar unido por guion bajo, ejemplo:

QString nombre_fichero.

Las clases deben empezar con minúscula y además si es compuesta será con guión bajo, ejemplo:

Class cargar_DLL

Para las clases de acceso a datos y lógica del negocio se debe empezar con las siglas AD y LN respectivamente, seguido de un guión bajo y el nombre de la clase, ejemplo:

Class AD_Cliente.

4.3.3 Concepción general de la ayuda

Como se establece en los requisitos no funcionales, los usuarios que van a utilizar la aplicación cuentan con conocimientos intermedios de informática, se propone insertar una ayuda de forma simple y concisa con el objetivo de explicar cómo se realiza cada proceso en el sistema y así ayudar a una mejor interacción con el mismo.

4.4 Diseño de la Base de Datos

El correcto funcionamiento de la aplicación depende también del buen diseño y funcionamiento de la base de datos a utilizar. En este epígrafe se muestran el Diagrama de Entidad Relación (DER), **figura 1** y el Diagrama de Clases Persistentes (DCP), **figura 2**, correspondiente al diseño de la BD.

El DER representa la realidad de la problemática identificada a través de las entidades y de los enlaces que rigen la unión de las mismas y que constituyen la relación del modelo.

Figura 1: Diagrama de Entidad Relación (DER)

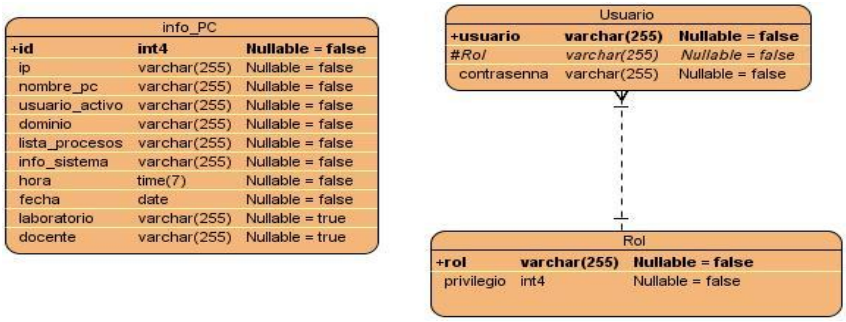
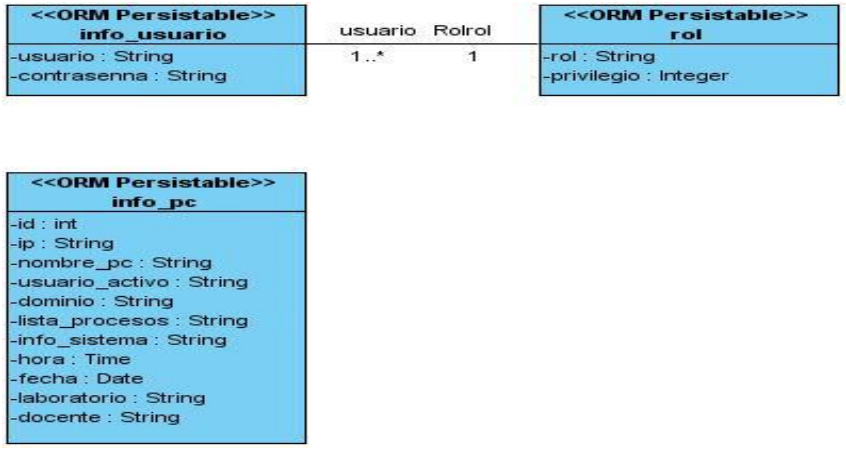


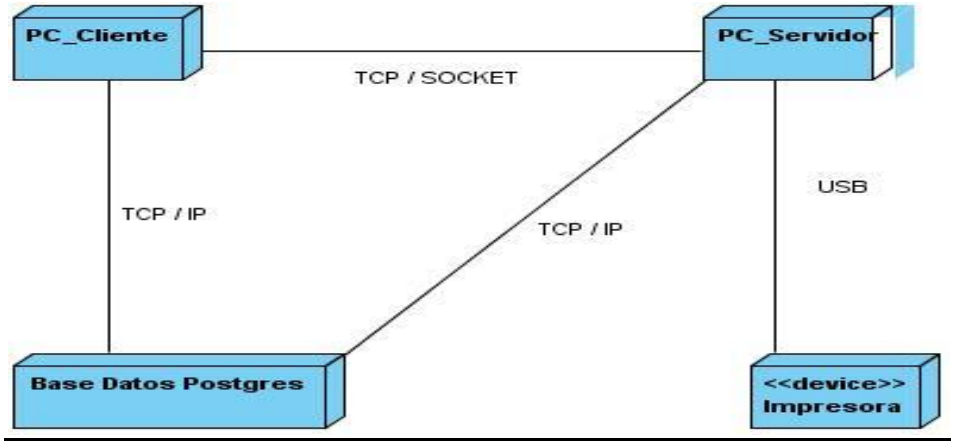
Figura 2: Diagrama de Clases Persistentes (DCP)



4.5 Modelo de Despliegue

El modelo de despliegue es un modelo conformado por objetos donde se describe la distribución física que presenta el sistema. A continuación se muestra en la **figura 3** el diagrama de despliegue para la aplicación a desarrollar:

Figura 3: Diagrama de Despliegue



4.5.1 Descripción de componentes

A continuación se describen los componentes del Modelo de Despliegue.

Nodo PC_Cliente

Este nodo representa las computadoras clientes, que son las PC donde la aplicación cliente está corriendo como un servicio y es la encargada de guardar toda la información en el Servidor de Base de Datos, además de recibir y enviar mensajes al nodo PC_Servidor mediante el protocolo TCP.

Nodo PC_Servidor

Este es el nodo que representa la computadora donde se encuentra la aplicación servidora, la cual está encargada de gestionar toda la información guardada en la Base de Datos y además de comunicarse con el nodo PC_Cliente mediante el protocolo TCP.

Nodo Base de Datos PostgreSQL

En este nodo se estará ejecutando el servidor PostgreSQL con la base de datos del sistema.

Dispositivo Impresora

Es el dispositivo encargado de imprimir los reportes de la aplicación.

4.6 Modelo de Implementación

El modelo de implementación describe como los elementos del modelo de diseño, como las clases, se implementan en términos de componentes, ficheros de código fuente, ejecutables, etc. También se organizan los componentes de acuerdo a los mecanismos de estructuración disponibles en el entorno de implementación y el lenguaje de programación utilizado. Además se representan las relaciones de dependencias entre los componentes existentes en el sistema.

4.7 Pruebas del sistema propuesto

El modelo de pruebas es una colección de casos de prueba, procedimientos de prueba y componentes de prueba, que permite probar los componentes ejecutables en el modelo de implementación. Entre los casos de prueba se puede distinguir dos tipos comúnmente utilizados: las llamadas pruebas de “caja negra” y las de “caja blanca”.

Una prueba de “caja negra” es una prueba del comportamiento observable externamente del sistema, mientras que una prueba de “caja blanca” prueba la interacción interna entre los componentes del sistema.

En el caso de dicho sistema, se realizó pruebas de “caja negra” para cada caso de uso, con el objetivo de probar la interacción entre el usuario y el sistema, que se satisfagan las precondiciones

y poscondiciones, y que se siga la secuencia de acciones intermedias especificadas por el caso de uso.

4.7.1 Diseño de Casos de pruebas

El principal objetivo de las pruebas es la intención de descubrir errores y por su puesto una prueba tiene éxito cuando se descubre un error no detectado hasta entonces, pero en gran parte las pruebas dependen del diseño de casos de pruebas.

Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar la mayor cantidad de errores con el mínimo esfuerzo posible, en menor tiempo y su principal objetivo es cubrir todas las posibilidades sin realizar demasiados casos de pruebas.

4.8 Conclusiones Parciales

En el presente capítulo se llevó a cabo la modelación de los diferentes procesos que presenta la aplicación, dentro de los cuales se encuentra el proceso de Análisis y Diseño, teniendo en cuenta su importancia para la realización del modelo de Implementación. Se hace referencia a los principales artefactos generados, así como los patrones arquitectónicos y del diseño utilizados. También se realiza el diseño de la Base de Datos y una breve explicación de las pruebas realizadas.

CAPÍTULO 5: Estudio de Factibilidad

La necesidad que representa saber si el desarrollo de un software resultará factible se manifiesta desde los comienzos de la concepción del mismo, por ello es que se hace muy importante realizar un estudio detallado de los principales beneficios y costos que traerá consigo su ciclo de desarrollo. En este capítulo se realizará un estudio detallado de la factibilidad que representa el sistema modelado hasta el momento.

5.1 Método de Estimación por Puntos de Casos de Uso.

La estimación mediante el análisis de Puntos de Casos de Uso es un método propuesto originalmente por Gustav Karner, y posteriormente refinado por muchos otros autores. Se trata de un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores. A continuación se despliega detalladamente dicho método.

Para realizar dicha estimación se efectúan 4 pasos fundamentales:

- ✓ Calcular los puntos de caso de uso sin ajustar.
- ✓ Calcular los puntos de casos de uso ajustados.
- ✓ La estimación de esfuerzo por a través de los puntos de de caso de uso.
- ✓ Costo del Proyecto.

Paso 1 Calcular los puntos de caso de uso sin ajustar (UUCP):

Para el cálculo de los puntos de caso de uso sin ajustar se hace necesario calcular también el factor de peso de los actores sin ajustar y el factor de peso de los casos de uso sin ajustar. El factor de peso de los actores sin ajustar (**UAW**), se calcula mediante un análisis de la cantidad de actores presentes en el sistema y la complejidad de cada uno de ellos.

Tabla 2: Factor de Peso de los Actores sin ajustar.

Tipo de Actor	Cantidad de Actores	Factor de Peso	Total
Simple	0	0*1	0
Medio	1	1*2	2
Complejo	2	2*3	6
Total			8

UAW = Σ cantidad de actores * peso

UAW = 0*0 + 1*2 + 2*3 = 8

El factor de peso de los casos de uso sin ajustar (**UUWC**) se calcula mediante un análisis de la cantidad de casos de uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo.

Tabla 3: Factor de Peso de los Casos de Uso sin ajustar.

Tipo de Actor	Factor de Peso	Cantidad de Casos de Uso	Total
Simple	5	1*5	5
Medio	10	2*10	20
Complejo	15	3*15	45
Total			70

UUCW = Σ cantidad de caso de uso * peso

UUCW = 5*1 + 10*2 + 15*3 = 70

Después de obtenidos los valores de **UAW** y el **UUCW** se puede calcular el valor de los puntos de caso de uso sin ajustar.

UUCP = UAW + UUCW

UUCP = 8 + 70

UUCP = 78

Paso 2 Cálculo los puntos de casos de uso ajustados:

Para el cálculo de los puntos de caso de uso ajustados se hace necesario calcular también el factor de complejidad técnica y el factor de ambiente.

El factor de complejidad técnica (**TCF**): se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5. Se cumple que para el valor:

- ✓ 0-No presenta influencia
- ✓ 1-Influencia incidental
- ✓ 2-Influencia moderada
- ✓ 3-Influencia media
- ✓ 4-Influencia significativa
- ✓ 5-Influencia fuerte

Tabla 4: Factor de Complejidad Técnica.

Factor	Descripción	Peso	Valor Asignado	Observación	Total
T1	Sistema Distribuido	2	5	Sistema Distribuido	10
T2	Objetivos de performance o tiempo de respuesta	1	4	El tiempo de respuesta debe ser lo más rápido posible.	4
T3	Eficiencia del usuario final.	1	1	Escasas restricciones de eficiencia.	1
T4	Procesamiento interno complejo.	1	3	Existen temas de mediana complejidad en el sistema	3
T5	El código debe ser reutilizable	1	4	El código si tiene como objetivo reutilizarse.	4
T6	Facilidad de instalación.	0.5	2	Hay que tener en cuenta algunos factores para la instalación.	1
T7	Facilidad de Uso	0.5	3	Normal	1.5
T8	Portabilidad	2	1	No es un objetivo primordial que el sistema sea portable.	2
T9	Facilidad de cambio.	1	1	El sistema es un poco resistente al cambio.	1
T10	Concurrencia	1	4	El sistema presenta una alta concurrencia.	4
T11	Incluye objetivos especiales de	1	3	Seguridad Normal	3

	seguridad				
T12	Provee acceso directo a terceras partes	1	0	No presenta acceso a terceros.	0
T13	Se requieren facilidades especiales de entrenamiento a los usuarios	1	2	Se necesita una breve explicación a los usuarios.	2

Finalmente el factor de complejidad técnica es:

TCF = 0.6 + 0.01 * Σ (peso * valor asignado)

TCF = 0.6 + 0.01 * 36.5

TCF = 0.965

El factor ambiente (**EF**): se calcula mediante la cuantificación de un conjunto de factores como son las habilidades y el entrenamiento que debe tener el grupo involucrado en el desarrollo del proyecto.

Tabla 5: Factor Ambiente.

Factor	Descripción	Peso	Valor Asignado	Comentario	Total
E1	Familiaridad con el modelo de proyecto utilizado	1.5	2	El equipo no tiene mucha experiencia con el modelo utilizado.	3
E2	Experiencia en la	0.5	2	El equipo hace poco tiempo	1

	aplicación			trabaja en dicha aplicación.	
E3	Experiencia en orientación a objetos	1	4	El equipo trabaja hace bastante tiempo orientado a objeto.	4
E4	Capacidad del analista líder	0.5	4	Existe un analista líder con experiencia en el equipo.	2
E5	Motivación	1	5	La motivación es muy alta.	5
E6	Estabilidad de los requerimientos	2	2	Se esperan cambios en los requerimientos.	6
E7	Personal part-time	-1	0	Todo el equipo es a tiempo completo.	0
E8	Dificultad del lenguaje de programación	-1	3	El lenguaje a utilizar es C++	-3
Total					**18**

EF = 1.4 – 0.03 * ∑ (peso * valor asignado)

EF = 1.4 – 0.03*18

EF = 0.86

Después de obtener los valores de **UUCP, TCF y EF** se calcula el valor de los puntos de casos de uso ajustados (**UCP**):

UCP = UUCP * TCF * EF

UCP = 78 * 0.965 * 0.86

UCP = 64.73

Paso 3 De los Puntos de Casos de Uso a la estimación del esfuerzo:

El esfuerzo en horas-hombre viene dado por:

E = UCP x CF

Donde:

E: Esfuerzo estimado en horas-hombre

UCP: Puntos de Casos de Uso ajustados

CF: Factor de conversión

Este método proporciona una estimación del esfuerzo en horas-hombre contemplando sólo el desarrollo de la funcionalidad (programación) especificada en los casos de uso. Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por debajo del valor medio (3), para los factores E1 a E6. Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por encima del valor medio (3), para los factores E7 y E8.

CF = 20 horas-hombre (si Total $_{EF} \leq 2$)

CF = 28 horas-hombre (si Total $_{EF} = 3$ ó Total $_{EF} = 4$)

CF = abandonar o cambiar proyecto (si Total $_{EF} \geq 5$)

Total $_{EF}$ = Cant EF < 3 (entre E1 –E6) + Cant EF > 3 (entre E7, E8)

Total $_{EF}$ = 2+0 = 2

CF = 20 horas-hombre (porque Total $_{EF} \leq 2$)

UCP =64.73

CF = 20 horas-hombre.

E = UCP x CF= 64.73* 20 horas-hombre =1294.6~1295 horas-hombre.

En la siguiente tabla se muestra la distribución del esfuerzo horas-hombres en las diferentes etapas del proyecto.

Tabla 6: Distribución del esfuerzo horas-hombre por etapas.

Actividad	Esfuerzo (%)	Valor Esfuerzo
Análisis	15%	485 horas-hombre
Diseño	20%	647 horas-hombre
Implementación	40%	1295 horas-hombre
Prueba	10%	323 horas-hombre
Sobrecarga	15%	485 horas-hombre
Total	100%	3238 horas-hombre

ET = 3238 horas-hombre y se estima que cada mes tiene como promedio 240 horas laborables eso daría un **ET**= 13.5 mes-hombre.

Se estima que una persona podrá realizar el problema analizado en 14 meses aproximadamente.

Paso 4 Costo del Proyecto:

Se asume como salario promedio mensual $100.00

CH: Cantidad de hombres.

T: Tiempo total del proyecto.

CHM: Costo de hombre por mes

CH = 2 hombres

CHM = 2 * Salario Promedio = 200.00 $/mes

Costo = CHM * ET / CH

Costo = 200.00 * 13.5 / 2 = $ 1350.00

Tiempo = **ET / CH**

Tiempo =13.5 / 2 = 6.75 meses.

De los resultados obtenidos se estima que con dos hombres trabajando en el proyecto, el mismo se desarrolla en poco menos de 7 meses, con un costo total estimado en $1350 pesos.

5.2 Beneficios Tangibles e intangibles.

5.2.1 Beneficios Tangibles

La solución final del proyecto no constituye un producto comercial, por tanto, no aporta directamente ganancias económicas al país. El principal beneficio que brinda es la posibilidad de un mejor uso de las computadoras en los laboratorios de producción de la facultad 9 y la automatización del proceso de control y gestión de las actividades que se realizan en dichos laboratorios, además contribuye de forma directa en el cumplimiento de las tareas docentes-productivas, proporcionando una fuente confiable de información para ayudar en la toma de decisiones con respecto a las indisciplinas existentes en estos laboratorios.

5.2.2 Beneficios intangibles

La aplicación posibilitará que el encargado de controlar el uso correcto de las computadoras en los laboratorios de producción tenga la premisa de lo que está sucediendo en cada ordenador del laboratorio que se le haya asignado, así podrá gestionar de forma ágil y segura la información correspondiente a las incidencias en dichos laboratorios. Por tanto, los recursos asignados por la Revolución serán utilizados de la mejor forma posible y así se contribuirá al ahorro energético,

influenciando de forma positiva en el desarrollo exitoso de los proyectos productivos en la facultad 9.

5.2.3 Análisis de Costos y Beneficios

El desarrollo del sistema no requerirá gastos significativos ya que todas las herramientas que se utilizaron en su confección son libres y de código abierto, con la excepción de la herramienta Case Visual Paradigm. No será necesario contratar ninguna empresa para la realización de algunos servicios indispensables en la aplicación. Por todos los beneficios analizados anteriormente y debido a que sus costos son mínimos se puede concluir que es factible desarrollar la aplicación.

5.3 Conclusiones Parciales

Con el estudio detallado de la factibilidad que se realizó se llegó a la conclusión que el sistema modelado será factible en muchos sentidos, analizándose los beneficios tangibles e intangibles que se desprenden del desarrollo y la utilización de la aplicación, tanto así como los costos que se tendrán en su desarrollo. Finalmente el análisis de costos contra los beneficios realizado es otra de las razones que condujo a la idea final de este estudio.

CONCLUSIONES GENERALES

- ✓ La aplicación informática desarrollada permite gestionar el uso correcto de las computadoras en los laboratorios de producción de la Facultad 9 desde un controlador virtual en tiempo real.

- ✓ El modelo de software diseñado sobre la base de las tecnologías y herramientas actuales garantizan la calidad y el funcionamiento del sistema, así como su mantenimiento y perduración, satisfaciendo de esta forma las necesidades del cliente, en este caso la Facultad 9.

- ✓ La factibilidad del proyecto realizado ha sido demostrada a través de la estimación por puntos de casos de usos, donde los resultados tangibles e intangibles obtenidos son mayores a los costos incurridos.

- ✓ El sistema garantiza un mayor control de las actividades docente-productivas ayudando en gran medida a la toma de decisiones por parte de los líderes de proyecto e influye directamente en el ahorro energético de la Universidad.

RECOMENDACIONES

- ✓ Desplegar en los laboratorios de producción de la facultad 9 la primera versión del sistema desarrollado para automatizar los procesos de gestión del uso de las computadoras.
- ✓ Recoger las principales deficiencias que se generen durante el despliegue del sistema y utilizarlas como premisas para el desarrollo de próximas iteraciones.
- ✓ Para el correcto funcionamiento del sistema desarrollado, los laboratorios de producción deben cumplir con las políticas de seguridad de la universidad.
- ✓ En estudios posteriores relacionados con el tema, se recomienda ampliar la arquitectura propuesta para desarrollar una aplicación lo más reutilizable posible y la realización de un conjunto de pruebas de caja blanca con el objetivo de obtener un producto de mayor calidad.

BIBLIOGRAFÍA REFERENCIADA

CINTERFOR. 1996. *Centro Interamericano para el Desarrollo del Conocimiento en la Formación del Profesional. [En línea] 1996. [Citado el: 18 de Enero de 2010.] http://www.cinterfor.org.uy/public/spanish/region/ampro/cinterfor/temas/gender/em_ca_eq/m_eva.h.*

Almeida Rodríguez, Francisco. 2002. *Herramientas y Lenguajes Infórmaticos. Herramientas y Lenguajes Infórmaticos. [En línea] 15 de 2 de 2002. [Citado el: 12 de 2 de 2010.] http://nereida.deioc.ull.es/~pcgull/hli02/index.html.*

Álvares Marañón, Gonzalo. 2006. *Características del lenguaje Java. [En línea] 2006. [Citado el: 14 de 2 de 2010.] http://www.iec.csic.es/CRIPTONOMICON/java/quesjava.html.*

AmericaTI.com. 2006. *Ventajas y Desventajas: Comparación de los lenguajes C,C++ y Java. [En línea] 11 de 11 de 2006. [Citado el: 12 de 2 de 2010.] http://www.americati.com/doc/ventajas_c/ventajas_c.html#id2486750.*

Bjarne Stroustrup, Addison-Wesley. 1991. *The C++ Programming Language. 1991.*
eAprende. 2008. eAprende.com. [En línea] 2008. [Citado el: 22 de febrero de 2010.] http://www.eaprende.com/gestor-de-basededatos-mysql-postresql-sqlite.html#post.

DRUCKER, PETER F. 2003. *GESTION DEL CONOCIMIENTO. Bilbao : DEUSTO S.A. EDICIONES, 2003. 9788423420230 .*

Eclipse Foundation. 2010. *Eclipse. [En línea] 2010. [Citado el: 10 de 2 de 2010.] http://www.eclipse.org/org/.*

Elmasri, R. y Navathe, S. 1997. "Sistemas de Bases de Datos conceptos y fundamentos". 1997.

Fayol, Henry (1841-1925). 1946. *Administración industrial y general. . s.l. : El Ateneo, 1946. 950-02-3540-4.*

Gosling, James. 1995. *Sun Microsystems. 1995.*

Grupo Soluciones GSInnova. 2007. *Grupo Soluciones GSInnova. [En línea] 2007. [Citado el: 23 de febrero de 2010.] http://www.rational.com.ar/herramientas/roseenterprise.html.*

Hernández Regalado, Rafael. 2007. *Las MIpimes en Latinoamerica.1era. p18. Mexico : s.n., 2007. 978-968-864-482-9.*

http://www.todo-programacion.com.ar. 2005. *Lenguaje de máquina. tp Todo sobre programación web & más. [En línea] 19 de 4 de 2005. [Citado el: 10 de 2 de 2010.] http://www.todo-programacion.com.ar/archives/2005/04/lenguaje_de_maq.html.*

ISO/900, 2000. 2000. *International Standards for Quality Manegement. s.l. : 2000, 2000.*

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software. Madrid : Pearson Educación , 2000. 84-7829-036-2.*

Lanzillotta, Analía. 2004. *Definición de Lenguajes de Programación. Master Magazine. [En línea] 2004. [Citado el: 10 de 2 de 2010.] http://www.mastermagazine.info/termino/5560.php.*

Llanes Delgado, W. 2004. *Fundamentos de la dirección y gestión. Ciudad Habana : s.n., 2004.*

Luján Mora, Sergio. 2006. *C++ paso a paso. [En línea] 19 de 12 de 2006. [Citado el: 15 de 2 de 2010.] http://gplsi.dlsi.ua.es/~slujan/materiales/cpp-muestra.pdf.*

Microsoft Corporation. 2007. *Visual Studio 2008. [En línea] 11 de 2007. [Citado el: 16 de 2 de 2010.] http://msdn.microsoft.com/es-es/library/aa187917.aspx.*

Pecos, Daneil. 2010. *danielpecos.com. danielpecos.com. [En línea] 2010. [Citado el: 20 de febrero de 2010.] http://www.netpecos.org/docs/mysql_postgres/index.html#AEN11.*

Perissé, Marcelo Claudio. 2001. *http://www.cyta.com.ar/. [En línea] febrero de 2001. [Citado el: 23 de febrero de 2010.] http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c5/c5.htm. 987-43-2947-5.*

Precios., Ministerio de Finanzas y. 2003. *R.No. 297/2003. Ciudad Habana : Gaceta Oficial de la Republica de Cuba, 2003.*

Python Software Fundation. 2010. *Python Programming Language. [En línea] 2010. [Citado el: 15 de 2 de 2010.] http://www.python.org/.*

Rolhud, Yues Dupuy and Gerard. 1992. *Manual de control de gestión. 1992.*

STONER. 1999. *ADMINISTRACION, 6ª. Mexico : PRENTICE HALL.S.A, 1999.*

Valentín, José Manuel Jiménez. 2008. *Gestión Empresarial. [En línea] 2008. [Citado el: 9 de Febrero de 2010.] http://www.gestionempresarial.info/VerItemProducto.asp?Id_Prod_Serv=28&Id_Sec=8..*

Victoria. 2009. *Definición de ABC. Definición de ABC. [En línea] 25 de feberro de 2009. [Citado el: 28 de febrero de 2010.] http://www.definicionabc.com/tecnologia/mysql.php.*

BIBLIOGRAFÍA CONSULTADA

CINTERFOR. 1996. *Centro Interamericano para el Desarrollo del Conocimiento en la Formación del Profesional. [En línea] 1996. [Citado el: 18 de Enero de 2010.] http://www.cinterfor.org.uy/public/spanish/region/ampro/cinterfor/temas/gender/em_ca_eq/m_eva.h.*

ABC Datos. 1999. ABC Datos. *[En línea] 1999. [Citado el: 21 de Enero de 2010.] http://www.abcdatos.com/programas/programa/l7933.html..*

Álvares Marañón, Gonzalo. 2006. *Características del lenguaje Java. [En línea] 2006. [Citado el: 14 de 2 de 2010.] http://www.iec.csic.es/CRIPTONOMICON/java/quesjava.html.*

Bjarne Stroustrup, Addison-Wesley. 1991. *The C++ Programming Language. 1991.*

DRUCKER, PETER F. 2003. *GESTION DEL CONOCIMIENTO. Bilbao : DEUSTO S.A. EDICIONES, 2003. 9788423420230 .*

eAprende. 2008. *eAprende.com. [En línea] 2008. [Citado el: 22 de febrero de 2010.] http://www.eaprende.com/gestor-de-basededatos-mysql-postresql-sqlite.html#post.*

Elmasri, R. y Navathe, S. 1997. *"Sistemas de Bases de Datos conceptos y fundamentos". 1997.*

Fayol, Henry (1841-1925). 1946. *Administración industrial y general. . s.l. : El Ateneo, 1946. 950-02-3540-4.*

Gonzalez, Michelena Fernandez and Gonzalez. 2004. 2004. *Gestion de la calidad. Ciudad Habana : ISPJE, 2004, 2004.*

Gosling, James. 1995. *Sun Microsystems. 1995.*

Hernández Regalado, Rafael. 2007. *Las MIpimes en Latinoamerica.1era. p18. Mexico : s.n., 2007. 978-968-864-482-9.*

ISO/900, 2000. 2000. *International Standards for Quality Manegement. s.l. : 2000, 2000.*

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software. Madrid : Pearson Educación , 2000. 84-7829-036-2.*

Lanzillotta, Analía. 2004. *Definición de Lenguajes de Programación. Master Magazine. [En línea] 2004. [Citado el: 10 de 2 de 2010.] http://www.mastermagazine.info/termino/5560.php.*

Llanes Delgado, W. 2004. *Fundamentos de la dirección y gestión. Ciudad Habana : s.n., 2004.*

Lopez Rodriguez, M. *El control en las organizaciones. Ciudad Habana : UH, 2004.*

Luján Mora, Sergio. 2006. *C++ paso a paso. [En línea] 19 de 12 de 2006. [Citado el: 15 de 2 de 2010.] http://gplsi.dlsi.ua.es/~slujan/materiales/cpp-muestra.pdf.*

Microsoft Corporation. 2007. Visual Studio 2008. *[En línea] 11 de 2007. [Citado el: 16 de 2 de 2010.] http://msdn.microsoft.com/es-es/library/aa187917.aspx.*

Pecos, Daneil. 2010. *danielpecos.com. danielpecos.com. [En línea] 2010. [Citado el: 20 de febrero de 2010.] http://www.netpecos.org/docs/mysql_postgres/index.html#AEN11.*

Perissé, Marcelo Claudio. 2001. *http://www.cyta.com.ar/. [En línea] febrero de 2001. [Citado el: 23 de febrero de 2010.] http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c5/c5.htm. 987-43-2947-5.*

Precios., Ministerio de Finanzas y. 2003. *R.No. 297/2003. Ciudad Habana : Gaceta Oficial de la Republica de Cuba, 2003.*

Rolhud, Yues Dupuy and Gerard. 1992. *Manual de control de gestión. 1992.*

STONER. 1999. *ADMINISTRACION, 6ª. Mexico : PRENTICE HALL.S.A, 1999.*

Valentín, José Manuel Jiménez. 2008. *Gestión Empresarial. [En línea] 2008. [Citado el: 9 de Febrero de 2010.] http://www.gestionempresarial.info/VerItemProducto.asp?Id_Prod_Serv=28&Id_Sec=8..*

Victoria. 2009. *Definición de ABC. Definición de ABC. [En línea] 25 de feberro de 2009. [Citado el: 28 de febrero de 2010.] http://www.definicionabc.com/tecnologia/mysql.php.*

GLOSARIO DE TÉRMINOS

Control: control, es el proceso para asegurar que las actividades reales se ajustan a las actividades planificadas" (STONER, 1999)

Monitoreo: el proceso continuo y sistemático mediante el cual se verifica la eficiencia y la eficacia de un proyecto mediante la identificación de sus logros y debilidades

Herramienta de Gestión: una herramienta cuyo soporte es un conjunto de informes sistemáticos que permiten a la organización analizar la captación, transformación y utilización de los recursos". (Hernández Regalado, 2007)

Laboratorios de Producción: Lugar donde se encuentran un conjunto de computadoras destinadas exclusivamente a realizar tareas docente-productivas.

UCI: Universidad de las Ciencias Informáticas.

IDE: Entrono de Desarrollo Integrado

CASE: es una sigla, que corresponde a las iniciales de: Computer Aided Software Engineering; y en su traducción al Español significa Ingeniería de Software Asistida por Computación.

Caso de Uso: Caso de Uso: Es una descripción de un conjunto de secuencias de acciones, incluyendo variaciones, que un sistema lleva a cabo y que conduce a un resultado observable de interés para un actor determinado.

Lista de Procesos: Listado de los programas o ejecutables que están corriendo actualmente en la PC.

CMD: Consola del Sistema Operativo Windows y se utiliza para ejecutar distintos comandos.

Aplicación Cliente: El cliente es una aplicación informática que ejecuta las funcionalidades descritas y envía el resultado al servidor de Base de Datos.

Aplicación Servidora: Aplicación que gestiona toda la información guardada en la Base de datos y envía instrucciones a la aplicación cliente.

Factibilidad: El concepto de factibilidad debe entenderse como la posibilidad y la conveniencia de llevar a cabo un proyecto. Para el desarrollo de sistemas los proyectos son evaluados a través de tres tipos de estudios de factibilidad: técnica, económica y operacional.