Universidad Central “Marta Abreu” de las Villas.

Facultad Matemática, Física y Computación

TRABAJO DE DIPLOMA

Sistema para el control de la carga docente en el departamento de Ciencia de la Computación

Autor: Oscar Amado Castro Sarduy

Tutor: Dr. Abel Rodríguez Morffi

Santa Clara

2013

Declaración Jurada

El que suscribe, Oscar Amado Castro Sarduy, hago constar que el trabajo titulado **Sistema para el control de la carga de docente en el departamento de Ciencia de la Computación** fue realizado en la Universidad Central "Marta Abreu" de Las Villas como parte de la culminación de los estudios de la especialidad de Licenciatura en Ciencias de la Computación, autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

_____

Firma del autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

Firma del tutor	Firma del jefe del Laboratorio

Fecha: 25 de junio de 2013

"…Inculcar en los hombres ese sentido de que lo importante es la obra, lo importante es lo que se haga, lo importante es el contenido de lo que se haga, sin que importe si nos reconocen o no el mérito, sin que importe que nos reconozcan o no la paternidad de una idea, la paternidad de una investigación…"

Fidel Castro Ruz

A aquel que ha dado al hombre sabiduría, ciencia y dones:

Al Rey de Reyes y Señor de Señores.

A mis padres, abuelos, a mi novia y mi mayor bendición que aún está en su vientre.

Agradezco sinceramente:

A Dios por darme fuerza para seguir adelante en todo momento

A mi tutor Dr. Abel Rodríguez Morffi por hacerse cargo de mí y ser ejemplo profesional a seguir.

Al Dr. Daniel Gálvez Lio por su ayuda durante estos cinco años.

A toda mi familia por su ánimo y confianza, en especial a mi mamá, a mi papá y a mi novia que hicieron la carrera conmigo, a mis hermanos Bladimír y Tatiana, a mi prima Lorena y mi primo Rigoberto y a mis abuelos Amado Joaquín y Oria.

A los hermanos de la universidad, especialmente: Raúl, Eduardo, David, Sergio y Daril.

A los amigos de la carrera por soportarme.

A los profesores que se empeñaron en hacer de mí y de mis compañeros buenos profesionales.

RESUMEN

El manejo de la información referente a la carga docente de los profesores del Departamento de Ciencia de la Computación de la facultad de Matemática, Física y Computación en la Universidad Central "Marta Abreu" de Las Villas es de gran importancia tanto con fines administrativos, docentes, científicos o investigativos. Se necesita de un sistema de información que permita planificar y controlar las actividades en las que se encuentran involucrados los profesores del mismo a la vez que mantenga esta información histórica y poder emitir informes sobre la labor de los docentes a lo largo de diferentes cursos académicos. En aras de solucionar la situación planteada el presente trabajo describe el desarrollo de un sistema de información para la gestión de estas actividades que comprende la carga de trabajo en pregrado y postgrado, así como la asesoría o tutoría a estudiantes. El mismo fue elaborado a partir de un análisis y diseño orientado a objetos con el Lenguaje Unificado de Modelado (UML), e implementado con herramientas de software libre para microcomputadoras.

ABSTRACT

Nowadays, the management of the information about the workload of teachers in the Computer Science Department of the faculty of Mathematics, Physics and Computing at the Universidad Central “Marta Abreu” Las Villas is very important. An information system for planning and controlling the activities in which teachers are involved is needed. This application should keep historical information and issue reports on the workload of teachers along different academic courses. In order to solve this problem, this work describes the development of an information system for the management of the information about the workload in undergraduate and graduate activities. The Unified Modeling Language in analysis and design was used. For the implementation open source tools were applied.

TABLA DE CONTENIDO

INTRODUCCIÓN

La Computación es una de las ramas de la ciencia con mayor velocidad de desarrollo, y cada vez se introduce más en todas las esferas de la vida del hombre, desde los procesos científicos, tecnológicos e industriales hasta las tareas domésticas. Este acelerado proceso de expansión, como ciencia en sí y en el resto de los procesos donde el hombre está involucrado, provoca una creciente necesidad de personal altamente preparado que sea capaz de asimilar, aplicar, adaptar y desarrollar los nuevos avances de esta ciencia.

Esta necesidad a escala mundial también se refleja en Cuba. En la década del 1980 se alcanzó un cierto desarrollo en la Computación, que se retrasó en los primeros años de la década del 90, y en estos momentos ha adquirido un fuerte impulso. El Programa de Informatización de la Sociedad es una prueba del interés del Estado cubano por el desarrollo acelerado de la Computación, además existen otras manifestaciones del interés en esta área que se reflejan en el apoyo que reciben los Organismos de la Administración Central del Estado, las empresas de capital mixto radicadas en el país así como en la mayoría del resto de las empresas, para desarrollar la Computación. Estos esfuerzos se evidencian en la creación del Ministerio de la Informática y las Comunicaciones (MIC) que estimula una industria nacional de software ya en desarrollo (DCC 2007) La Universidad Central "Marta Abreu" de Las Villas (UCLV), y particularmente la actual Facultad de Matemática, Física y Computación (MFC), desarrolla docencia e investigación en Computación desde hace alrededor de 30 años. En esta facultad se han impartido cinco Planes de Estudio relacionados con la Computación: Matemática Aplicada; Plan "A" y Plan "B" de Licenciatura en Cibernética Matemática; Plan "C" y Plan C98 de Licenciatura en Ciencia de la Computación y desde hace algún tiempo ha comenzado a aplicarse el plan D; Plan "C transición a D" y Plan "D" de Ingeniería Informática.

Un hito importante en el desarrollo del Departamento de Ciencia de la Computación (DCC) se alcanza en el año 1998 cuando se subordina al Centro de Estudios de Informática (CEI) con lo cual se consolida la unión entre el aspecto laboral e investigativo con el docente de la carrera. Esta unión ha derivado un sinnúmero de logros científicos, así como ha fortalecido el claustro del departamento en cuanto a su proyección científica y profesional, con lo que se beneficia la formación del profesional.

El DCC realiza una intensa actividad de pregrado en dos carreras: Licenciatura en Ciencia de la Computación e Ingeniería Informática, y postgrado. Cada año se imparte un gran

número cursos en Cuba y en el extranjero. El programa de la maestría de Computación Aplicada en un sentido amplio, contribuye al vínculo con la producción, y los programas de la maestría de Ciencia de la Computación y el programa de formación doctoral en Computación contribuyen a la formación continua de los egresados y están muy relacionados con la superación del claustro de las carreras.

Desde 1994 se desarrolla la Maestría en Computación Aplicada, acreditada desde el año 2000. Esa maestría está diseñada con un enfoque amplio respondiendo a su objetivo general de preparar profesionales de diversas ramas para que hagan un uso efectivo de la Computación; se han efectuado 10 ediciones de la misma y actualmente se encuentra una nueva edición revisada. En esas ediciones se han atendido grupos de varias provincias; en todos estos casos ha estado dirigida a superar a los profesores que en estas asumieron la apertura de la carrera de Ingeniería Informática. También se tienen graduados extranjeros de México, Venezuela, Nicaragua y Colombia con participación mayoritaria de nuestros docentes.

Desde 1999 se inicia la Maestría en Ciencia de la Computación, orientada a elevar la preparación de los profesionales con formación en Ciencia de la Computación o con formación afín a la misma. Esta maestría se acredita de excelencia en el 2006. La misma se ha caracterizado por un claustro estable que cuenta con el 100% de doctores y por una incidencia en todo el país que se refleja en la alta demanda de solicitantes que se interesan por cursar esta maestría y la inclusión en cursos de graduados de Ingeniería Informática y Ciencia de la Computación de todas las provincias.

A partir del 2001 se inició un programa de formación doctoral en Computación, esa primera edición se desarrolló orientada a profesores de la Dirección General de los Institutos Tecnológicos de México. Hasta el momento se han desarrollado varias ediciones, en la que se han incorporado aspirantes de diversas instituciones y provincias. A partir de la tercera edición, el programa doctoral es apoyado por el Programa de Colaboración Internacional que la UCLV desarrolla con el Consejo de Universidades Flamencas de Bélgica (VLIR). Además del programa doctoral grupal, la formación doctoral tutelar se ha consolidado y ha entrado en una etapa de madurez en los últimos años.

La actividad científica del departamento está organizada en seis grupos científicos: Bioinformática, Inteligencia Artificial, Programación e Ingeniería de Software, Bases de Datos, Computación Gráfica e Informática Educativa. La atención a estudiantes del Grupo

de Trabajo Científico Estudiantil (GTCE) y trabajos de diploma se realiza en estos grupos establecidos. El claustro se ha destacado en el plano científico, y en particular, en los últimos años se han alcanzado numerosos resultados, muchos de los cuales se han aplicado en diferentes empresas y centros de investigación del país. El trabajo antes resumido ha sido realizado por un colectivo de profesores relativamente joven que tuvo como embrión un reducido claustro en el seno de la carrera de Licenciatura en Matemática; este colectivo fue nutriéndose de los propios egresados con un alto espíritu de superación, cohesionado, con políticas internas que estimulaban la superación y con una elevada exigencia. También se han consolidado los programas de postgrados y en general ha creado un ambiente de trabajo en que se logra una integración de lo académico y lo científico, tanto en el nivel de pregrado como en el postgrado. Inmerso en todo este intenso trabajo ha funcionado con estabilidad los colectivos de las dos carreras.

El DCC, por tanto, debe planificar y controlar el trabajo de los docentes que pertenecen al mismo en el sinnúmero de actividades en que participan. Para esto se ha apoyado en determinado grado en las Tecnologías de la Información y la Comunicación (TIC). Actualmente se cuenta con un Sistema de Bases de Datos (SBD) muy sencillo que sólo abarca la asignación de los profesores a las actividades docentes dentro de un semestre sin abordar las tutorías y asesorías, ni las responsabilidades en los colectivos de disciplina, ni la historia en el trabajo docente y metodológico.

En los últimos tiempos se ha producido un auge notable de los Sistemas de Información (SI) debido, entre otros factores, a la gran cantidad de información que cada día se va generando, a los avances en la tecnología de las comunicaciones y al decremento en los costos de las computadoras. Ante las limitaciones mencionadas en el sistema actual y atendiendo las amplias posibilidades que ofrecen los SI actuales, se justifica la creación de un nuevo SI que permita planificar y controlar las actividades del DCC eliminando las limitaciones del sistema actual.

Por todo lo anterior se formula el siguiente **problema de investigación**: El DCC necesita de un SI que permita planificar y controlar las actividades en las que se encuentran involucrados los profesores del mismo a la vez que mantenga esta información histórica y poder emitir informes sobre la labor de los docentes a lo largo de varios cursos académicos, así como información útil para evaluaciones internas y externas, acreditaciones, etc.

En aras de solucionar la situación planteada, se establece la siguiente **hipótesis**: Un SI que

permita llevar a cabo la planificación y control de la actividad docente a lo largo de los cursos académicos, así como las asesorías y tutorías de alumnos de pregrado y postgrado brindará información de gran utilidad para los procesos de balance de carga y evaluación de profesores, asignaturas, disciplinas y carreras así como alimentar el torrente de información requerida para los proceso de evaluación que realiza la Junta de Acreditación Nacional (JAN).

Para dar cumplimiento a la hipótesis planteada anteriormente, se define el siguiente sistema de objetivos:

Objetivo general

Crear un SI que permita llevar a cabo la planificación y control de la actividad docente a lo largo de los cursos académicos, así como las asesorías y tutorías de alumnos de pregrado y postgrado.

Objetivos específicos

1. Diseñar la arquitectura de la información para la planificación y control de la actividad docente a lo largo de los cursos académicos, así como las asesorías y tutorías de alumnos de pregrado y postgrado.
2. Crear una base de datos conforme a la arquitectura de la información obtenida.
3. Concebir una interfaz informacional para manipular la base de datos creada.
4. Implementar un SI acorde a la interfaz concebida.

El **valor práctico** de los resultados de este trabajo está dado por la posibilidad de contar con un sistema computacional que brinde información útil para los procesos de asignación de carga docente y evaluación de profesores, asignaturas, disciplinas y carreras. Relacionado con los antecedentes antes planteados, se puede decir que este trabajo se **justifica** por su importancia desde el punto de vista práctico en la gestión del DCC.

Esta tesis se encuentra estructurada en tres capítulos:

En el capítulo 1 describe brevemente la actividad del DCC y se resumen los principales elementos de carácter teórico-práctico relacionados con el estudio de tecnologías y herramientas que permiten dar solución efectiva al problema planteado.

En el capítulo 2 se aborda la ingeniería de requisitos, análisis y diseño del sistema propuesto; así como las tecnologías computacionales que se usaron para la implementación.

En el capítulo 3 se detallan los procesos de instalación y uso del Sistema para la Planificación y Control de la Carga Docente (SPCCD) del DCC.

La tesis culmina con conclusiones, recomendaciones, referencias bibliográficas y anexos.

CAPÍTULO 1: TECNOLOGÍAS Y HERRAMIENTAS DE DESARROLLO LIBRES

En este capítulo se describe brevemente la actividad del DCC y se resumen los principales elementos de carácter teórico-práctico relacionados con el estudio de tecnologías y herramientas de software libre que permiten dar solución efectiva al problema planteado.

1.1 Breve descripción de la actividad del DCC.

El DCC se encarga de la docencia de pregrado en dos carreras, educación continua de postgrado en cursos, maestrías y un programa doctoral, a la vez que mantiene una dinámica actividad investigativa básica y aplicada. Por tanto, este departamento debe planificar y controlar el trabajo de los docentes que pertenecen al mismo en el sinnúmero de actividades en que participan.

Actualmente, el departamento cuenta con un SBD muy sencillo que sólo permite asignar carga docente dentro de un solo semestre sin registrar las tutorías y asesorías, ni las responsabilidades en los colectivos de disciplina, ni la historia en el trabajo docente y metodológico.

Para lograr los objetivos deseados, cualquier solución contemporánea debe apoyarse en tecnologías y herramientas que optimicen el proceso de planificación y control de las actividades en las que se encuentran involucrados los profesores del mismo a la vez que mantenga información histórica para poder emitir informes sobre la labor de los docentes a lo largo de varios cursos académicos, así como información útil para evaluaciones internas y externas, acreditaciones, etc. y que ayude en la tome de decisiones operacionales.

El MIC ha estimulado la consolidación de una industria nacional de software con una determinada independencia tecnológica que presupone la creación y uso de tecnologías y herramientas libres. Debido a las dificultades económicas que afronta el país y otros factores influyentes derivados del bloqueo económico y financiero que ha impuesto EEUU a Cuba, no se pueden adquirir la mayoría de las herramientas computacionales más avanzadas para el desarrollo de software por lo que existe la necesidad de buscar la vía más factible para lograr el desarrollo deseado. Dadas estas circunstancias, en este trabajo se decide analizar y desarrollar el sistema propuesto bajo el paradigma de software libre.

1.2 Tecnologías y herramientas libres

En esta sección se aborda un conjunto de tecnologías y herramientas para el desarrollo de aplicaciones que serán utilizadas para la implementación computacional del sistema propuesto.

Como es de saber, con el surgimiento de las grandes computadoras en los 60, el software no era más que un añadido a las grandes computadoras; luego, con el desarrollo de la técnica en las computadoras, surgió una nueva necesidad: los sistemas operativos, obligando a los usuarios aceptar condiciones privativas que impedían realizar modificaciones en dicho software. No fue hasta el año 1984 cuando el denominado padre del software libre, Richard Stallman, decide comenzar su proyecto llamado GNU, y un año más tarde decide fundar la Fundación del Software Libre (FSF, del inglés Free Software Foundation) (Stallman 2004)

1.2.1 Software libre

Un software se considera libre para el usuario siempre que, como usuario particular, tenga la libertad de (Stallman 2004):

1. Ejecutar el programa sea cual sea el propósito.
2. Modificar el programa para ajustarlo a sus necesidades. Para que se trate de una libertad efectiva en la práctica, deberá tener acceso al código fuente, dado que sin él la tarea de incorporar cambios en un programa es extremadamente difícil.
3. Redistribuir copias, ya sea de forma gratuita o a cambio de un determinado pago.
4. Distribuir versiones modificadas del programa, de tal forma que la comunidad pueda aprovechar las mejoras introducidas.

Debido a la gran satisfacción brindada por el software libre desde sus inicios en los años 80, es que se suma una gran cantidad de países a su consumo en el ámbito de las administraciones públicas tales como: Alemania, Argentina, Brasil, Cuba, Chile, China, Ecuador, España, Francia, México, República Dominicana y Venezuela. Existen varias tecnologías y herramientas libres que van desde entornos de desarrollo basados en lenguajes de programación, Sistemas Operativos (SO), Sistemas Gestores de Bases de Datos (SGBD), Servidores Web (SW), etc.

1.2.2 Lenguaje de programación Java

Java es un lenguaje de programación de alto nivel de última generación que se basa en el paradigma orientado a objetos. Por sus características, es un lenguaje que permite disminuir el tiempo de desarrollo y a la vez aumentar la manejabilidad del código (Herrera 2007). Fue creado en Sun Microsystems, en el año 1991. Inicialmente estaba desinado a equipos electrodomésticos. Como lenguaje de programación para computadoras, Java se introdujo a finales de 1995.

Este lenguaje se creó con cinco objetivos principales:

1. Debería usar el paradigma de la programación orientada a objetos.
2. Debería permitir la ejecución de un mismo programa en múltiples sistemas operativos.
3. Debería incluir soporte para trabajo en red de forma predeterminada.
4. Debería diseñarse para ejecutar código en sistemas remotos de forma segura.
5. Debería ser fácil de usar y tomar lo mejor de otros lenguajes orientados a objetos, como C++.

1.2.2.1 Paradigma de programación orientado a objetos

La primera característica orientada a objetos (OO) se refiere a un método de programación y al diseño del lenguaje. Aunque hay muchas interpretaciones para OO, una primera idea es diseñar el software de forma que los distintos tipos de datos que usen estén unidos a sus operaciones. Así, los datos y el código (funciones o métodos) se combinan en entidades llamadas objetos. Un objeto puede verse como un paquete que contiene el "comportamiento" (código) y el "estado" (datos). El principio es separar aquello que cambia de las cosas que permanecen inalterables. Frecuentemente, cambiar una estructura de datos implica un cambio en el código que opera sobre los mismos, o viceversa. Esta separación en objetos coherentes e independientes ofrece una base más estable para el diseño de un sistema software. El objetivo es hacer que grandes proyectos sean fáciles de gestionar y manejar, mejorando como consecuencia su calidad y reduciendo el número de proyectos fallidos. Otra de las grandes promesas de la programación orientada a objetos es la creación de entidades más genéricas (objetos) que permitan la reutilización del software entre proyectos, lo cual es una de las premisas fundamentales de la Ingeniería del Software. Un objeto genérico "cliente", por ejemplo, debería, en teoría, tener el mismo conjunto de comportamiento en diferentes proyectos, sobre todo cuando estos coinciden en cierta

medida, algo que suele suceder en las grandes organizaciones. En este sentido, los objetos podrían verse como piezas reutilizables que pueden emplearse en múltiples proyectos distintos, posibilitando construir proyectos de envergadura empleando componentes ya existentes y de comprobada calidad. Esto conduce a una reducción drástica del tiempo de desarrollo. La reutilización del software no ha experimentado los resultados esperados. Las principales dificultades son: el diseño de objetos realmente genéricos no se comprende completamente y faltan metodologías para la amplia comunicación de oportunidades de reutilización. Algunas comunidades de "código abierto" (open source) han estado ayudando en este problema ofreciendo medios a los desarrolladores para diseminar la información sobre el uso y versatilidad de objetos reutilizables y bibliotecas de objetos.

1.2.2.2 Independencia de la plataforma

La segunda característica del lenguaje de programación Java, la independencia de la plataforma, significa que programas escritos en este lenguaje pueden ejecutarse igualmente en cualquier tipo de hardware. Este es el significado de ser capaz de escribir un programa una vez y que pueda ejecutarse en cualquier dispositivo.

Para ello, se compila el código fuente escrito en lenguaje Java para generar un código conocido como "Java bytecode" que son instrucciones de máquina simplificadas específicas de la plataforma Java. Esta es una pieza intermedia entre el código fuente y el código máquina que entiende el dispositivo destino. El bytecode es ejecutado en la máquina virtual (JVM, del inglés Java Virtual Machine), un programa escrito en código nativo de la plataforma destino que entiende su hardware e interpreta y ejecuta el código. Además, se suministran bibliotecas adicionales para acceder a las características de cada dispositivo, como los gráficos, ejecución mediante hilos o threads, la interfaz de red, de forma unificada. Se debe tener presente que, aunque hay una etapa explícita de compilación, el bytecode generado es interpretado o convertido a instrucciones de máquina del código nativo por el compilador JIT (Just In Time) (véase la figura 1.1).

La portabilidad es técnicamente difícil de lograr, y los resultados de Java en ese campo han sido heterogéneos. Aunque se pueden escribir programas para la plataforma Java que actúen de forma correcta en múltiples plataformas de distinta arquitectura, el gran número de estas presentan errores o inconsistencias. El concepto de independencia de la plataforma de Java cuenta, sin embargo, con un gran éxito en las aplicaciones en el entorno del servidor, como

los Servicios Web, los Servlets, los Java Beans, así como en sistemas empotrados, usando entornos Java empotrados.

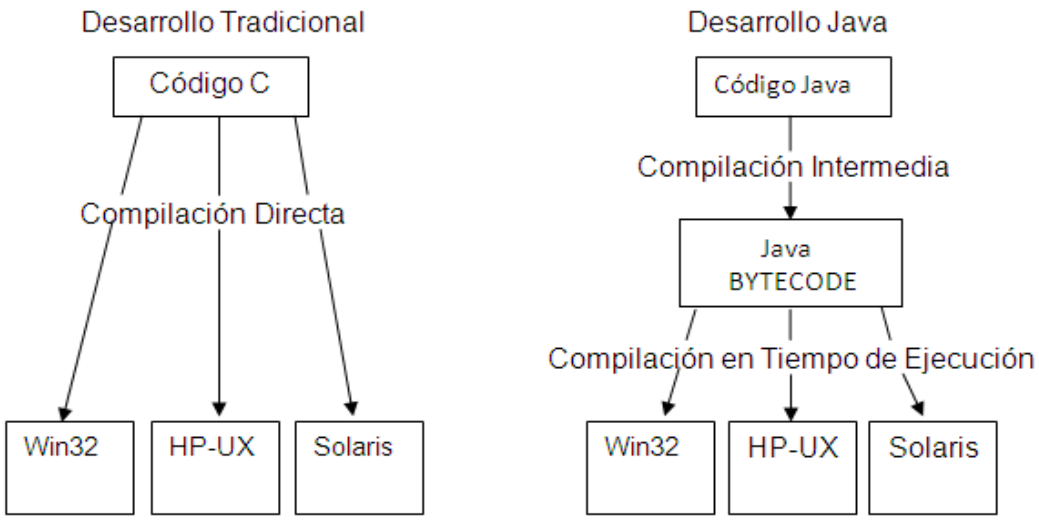

Figura 1.1 Comparación entre la compilación de un programa hecho en C y en Java.

1.2.2.3 Facilidad de uso y trabajo en red

El diseño de Java, su robustez, el respaldo de la industria y su fácil portabilidad han hecho de Java uno de los lenguajes con un mayor crecimiento y amplitud de uso en distintos ámbitos de la industria de la informática.

La sintaxis de Java se deriva en gran medida de C++. Pero a diferencia de éste, que combina la sintaxis para programación genérica, estructurada y orientada a objetos, Java fue construido desde el principio para ser completamente orientado a objetos. Todo en Java es un objeto (salvo algunas excepciones), y todo en Java reside en alguna clase a partir del cual pueden crearse varios objetos.

Hoy en día existen muchas aplicaciones gráficas de usuario basadas en Java. El entorno de ejecución Java (JRE) se ha convertido en un componente habitual en las computadoras de usuario de los sistemas operativos más usados en el mundo. Además, muchas aplicaciones Java lo incluyen dentro del propio paquete de la aplicación de modo que se ejecuten en cualquier computadora.

En las primeras versiones de la plataforma Java existían importantes limitaciones en las interfaces de programación de aplicaciones (API) de desarrollo gráfico (AWT). Desde la aparición de la biblioteca Swing, la situación mejoró substancialmente y posteriormente con la aparición de bibliotecas como SWT hacen que el desarrollo de aplicaciones de escritorio complejas y con gran dinamismo, usabilidad, etc. sea relativamente sencillo.

La compañía Sun Microsystems describe el lenguaje Java como "simple, orientado a objetos, distribuido, interpretado, robusto, seguro, de arquitectura neutra, portable, de altas prestaciones, multitarea y dinámico". Todo ello describe bastante bien el lenguaje Java, aunque en algunas de esas características el lenguaje sea todavía bastante mejorable (García de Jalón and Rodríguez 1999).

Hoy en día se puede encontrar la tecnología Java en redes y dispositivos que comprenden desde Internet y supercomputadora científicas hasta computadoras portátiles y teléfonos móviles; desde simuladores de mercado en Wall Street hasta juegos de uso doméstico y tarjetas de crédito. Esto significa que Java está en todas partes y es referenciada en entornos ubicuos de cómputo.

1.2.3 NetBeans

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto gratuito sin restricciones de uso. Esta herramienta fue desarrollada por Sun Microsystems (Cerda 2011) y está dotada de muchas facilidades y ventajas La rapidez con que se pueden desarrollar las aplicaciones es un aspecto a tener presente, pues es de vital importancia para los usuarios tener el producto final lo más rápido posible en la mayoría de los casos. Entre los lenguajes compilables en NetBeans se encuentra el Java. Este software, además de apegarse a la política seguida por un gran número de empresas en Cuba respecto al uso y desarrollo de herramientas y tecnologías de software libre, ofrece rapidez en el procesamiento de datos, tiene alta capacidad para soportar estructuras de almacenamiento complejas, da la posibilidad de construir interfaces gráficas de usuario utilizando la llamada Programación Visual. En lugar de escribir el código puro en Java, se crean aplicaciones utilizando herramientas visuales de manera que el código a escribir sea mínimo, el reuso de código fuente, así como la disponibilidad de la biblioteca Swing que dota al diseñador de la interfaz de usuario de las aplicaciones de muchas componentes que enriquecen y mejoran estéticamente dichas interfaces, además de proporcionarle a los programadores otras componentes que implementan muchas funcionalidades reusables.

Esta plataforma ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma están:

1. Administración de las interfaces de usuario con menús y barras de herramientas.
2. Administración de las configuraciones del usuario.
3. Administración del almacenamiento.
4. Administración de ventanas.
5. Marco de trabajo o framework basado en asistentes.

Entre las principales bibliotecas para el trabajo basadas en Java útiles para dar solución al problema planteado se encuentran:

- **java.sql.***

Esta biblioteca es implementada por la API de la conectividad Java para bases de datos (JDBC, del inglés Java Database Conectivity). Esta es importada para hacer consultas en bases de datos (BD) mediante conexiones básicas y simples que sirven para acceder a BD usando sentencias del lenguaje SQL a través de llamadas a métodos de la biblioteca Java.sql.*.

- **javax.swing.***

Originalmente la creación de interfaces gráficas en Java se realizaba con la biblioteca AWT (Abstract Window Toolkit). Pero esta tenía serías limitaciones como el número de pequeños programas, sus posibilidades o la gestión de eventos. Era realmente complejo el lograr funcionalidades que se obtenían de forma sencilla con otras bibliotecas.

Todo esto cambió con la llegada de Swing la cual ha mostrado una calidad y unas funcionalidades superiores a AWT. De hecho, Swing es una biblioteca repleta de funcionalidades que permiten la creación de interfaces gráficas muy ricas tanto estéticamente como en funcionalidades. Usando esta biblioteca se podrán construir los tres tipos de programas o aplicaciones que son JFRAME, JWINDOW y JAPPLET.

- **PostgreSQL-8.3-603.jdbc3.jar**

Esta biblioteca cuenta con los elementos necesarios, conocidos como manipuladores de dispositivos (drivers), para realizar las conexiones de acuerdo al SGBD al cual se desea conectar. En Cuba se estimula el uso de PostgreSQL, para el cual es necesario contar con la biblioteca PostgreSQL-8.3-603.jdbc3.jar.

1.2.4 Sistemas Gestores de Base de Datos

Un SGBD puede definirse como un paquete generalizado de software, que se ejecuta en un sistema computacional anfitrión y centraliza los accesos a los datos y actúa de interfaz entre los datos físicos y el usuario. Las principales funciones que debe cumplir un SGBD se relacionan con la creación y mantenimiento de la BD, el control de accesos, la manipulación de datos de acuerdo con las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, evitar redundancias e inconsistencias y mantener la integridad. Algunos SGBDs conocidos son: Oracle, PostgresSQL, SQLServer, MySQL, entre otros.

La elección del SGBD en un sistema de información constituye una parte importante de la solución del problema. Los puntos de vista para hacer esta elección son diversos y tienen implicaciones importantes desde la eficiencia hasta la economía. Una breve comparación entre algunos de los SGBD más conocidos en Cuba, teniendo en cuenta algunos factores como la incorporación de tecnologías modernas y el soporte, servirán de punto de partida para justificar la elección y establecer algunas recomendaciones (véase la tabla 1.1).

SGBD	**Microsoft Access**	**Microsoft SQL Server**	**Oracle**	**MySQL**	**PosgreSQL**
Licencia	Privada	Privada	Privada	Libre (GPL) y Privada	Libre (BSD)
Plataforma	Windows	Windows	Múltiple	Múltiple	Múltiple
Uso recomendado	Personal, Pequeños negocios	Grandes Empresas	Grandes Empresas	Medianas y Grandes Empresas	Medianas y Grandes Empresas
Concurrencia	Limitada	Alta	Alta	Alta	Alta
Soporte de transacciones	No	Sí	Sí	Sí	Sí
Modelo de datos	Relacional	Relacional	Relacional	Relacional	Objeto-Relacional

Tabla 1.1 Comparación entre SGBD.

Para justificar la elección para este trabajo, primero se analiza desde el punto de vista de la portabilidad. Lo ideal es que el sistema sea totalmente independiente del sistema operativo sobre el que se trabaja. Un SGBD que cumpla con este principio de independencia facilita la instalación y la migración de un sistema a otro de ser necesario. A este respecto, Cuba se encuentra en una etapa de informatización en la cual se aboga por la promoción y el uso del

software libre, y el GNU-Linux ha probado ser más eficiente para las aplicaciones de servidores de gran concurrencia que muchos sistemas operativos de carácter privado. Por tanto, los SGBD de Microsoft al no cumplir con este requisito pueden ser excluidos. Esto se une a la ventaja de permitir la definición de tipos de datos y el tratamiento de las tablas y sus relaciones desde el punto de vista estructural como objetos. Se eligió el PostgreSQL por las ventajas y la conveniencia que presenta respecto a los demás SGBD expuestos.

- **PostgreSQL**

El SGBD PostgreSQL está sistema diseñado para administrar grandes cantidades de datos y se prestigia de ser el SGBD de código abierto más avanzado actualmente. Este sistema es robusto y tiene más de quince años de desarrollo activo ganándose la reputación de ser confiable y mantener la integridad de los datos, se ejecuta en la mayoría de los SO más utilizados incluyendo Linux y varias versiones de UNIX. Su código fuente está disponible bajo la más liberal de las licencias del open source: la BSD. Bajo esta licencia se puede usar, modificar y distribuir PostgreSQL, en productos comerciales o no comerciales, sin costo alguno lo cual lo convierte en una alternativa extremadamente atractiva para las empresas que buscan un ahorro significativo de costos en activos TIC. Este provee nativamente soporte para:

- Números de precisión arbitraria.
- Texto de tamaño largo ilimitado.
- Figuras geométricas con una variedad de funciones asociadas.
- Direcciones IP v4 y v6.
- Bloques de direcciones estilo CIDR.
- Direcciones MAC.
- Arreglos.
- Herencia.
- Llaves Foráneas.
- *Tipos definidos por el usuario*: Se pueden crear tipos de datos que pueden ser indexables gracias a la infraestructura GiST de PostgreSQL. Un ejemplo son los tipos de datos GIS creados por el proyecto PostGIS.

- *Funciones definidas por el usuario*: Bloques de código que se ejecutan en el servidor. Pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos da, desde las operaciones básicas de programación, tales como bifurcaciones y bucles, hasta las complejidades de la programación orientada a objetos o la programación funcional. Se soportan funciones que retornan filas, donde la salida puede tratarse como un conjunto de valores que pueden ser tratados igual a una fila retornada por una consulta (query en inglés). Las funciones pueden ser definidas para ejecutarse con los derechos del usuario ejecutor o con los derechos de un usuario previamente definido. El concepto de funciones en otros SGBD es referido como "procedimiento almacenado" (stored procedure en inglés). Los disparadores son funciones enlazadas a operaciones sobre los datos.

1.2.5 JasperReports

JasperReports es una herramienta gratuita y open source que se compone de un conjunto de bibliotecas java para facilitar la generación de informes en nuestras aplicaciones tanto Web como de escritorio. Los informes se definen en un archivo xml que se compila por las bibliotecas jasperreport y generan un archivo .jasper que se utiliza para completar y mostrar el informe final. La salida de los informes puede ser a la impresora, pdf, cvs, xml, txt, html, xls, rtf, jasper viewer, y se prevé que a medida que se liberen nuevas versiones se generen otras más. La definición de los informes se puede realizar directamente sobre los archivos xml o se puede utilizar la herramienta ireport para diseñarlo gráficamente.

1.3 Sistemas de Información

Un SI, según diferentes autores como (Oz 2001), (Stair and Reynolds 2000), puede definirse técnicamente como un conjunto de componentes interrelacionados que permiten capturar, procesar, almacenar y distribuir la información para apoyar la toma de decisiones, la coordinación y el control en una institución.

Principalmente, se hace referencia al concepto de Sistemas Informativos considerándolo como un subconjunto de los SI en el sentido de que reducen sus objetivos a "manipular (almacenar y procesar) y distribuir información", puesto que la captura de los datos o información primaria corre en un elevado porcentaje o en su totalidad al propio desarrollador o en su defecto a un administrador en cada lugar de implantación. Entonces no queda duda que los SI no son más que sistemas automatizados que organizan los datos para producir información.

Debido a las características del trabajo se impone la implementación de un SI para gestionar, con diferentes propósitos, la información de la carga docente de los profesores del DCC y así satisfacer las necesidades informacionales del usuario.

1.4 Consideraciones parciales

- Existe la necesidad en el DCC de contar con una herramienta de fácil uso para las labores diarias de planificación, ajuste, seguimiento, evaluación y entrega de información a la dirección.
- Hoy en día el software libre se usa mundialmente y se ha observado una tendencia hacia su utilización. El hecho de ser libre y el aumento en la calidad del mismo son dos de las ventajas que lo hacen mucho más deseable; pero la causa fundamental de la utilización de herramientas de software libre en Cuba es la independencia tecnológica a la que se aspira.
- En la UCLV se tiene experiencia en el uso de Java, NetBeans y PostgresSQL, que satisfacen las necesidades para el desarrollo del SI planteado y concuerdan con las políticas de uso y desarrollo de software en Cuba.

CAPITULO 2: CONCEPCIÓN DEL SISTEMA PARA LA PLANIFICACIÓN Y CONTROL DE LA CARGA DOCENTE (SPCCD)

En este capítulo se aborda la ingeniería de requisitos, análisis y diseño del sistema propuesto; así como las tecnologías computacionales que se usaron para la implementación.

2.1 Descripción del sistema

El jefe del DCC cuenta con la información referente a los profesores que pertenecen y colaboran con el mismo así como de las distintas asignaturas y disciplinas que conforman el plan de estudios de las carreras que atiende, y de los programas y cursos de postgrado. Esta información permite asignar la carga docente a cada uno de estos profesores a la vez que se tenga un conocimiento del volumen de trabajo de los mismos.

2.1.2 Gestión de Requerimientos

La gestión de requerimientos es un punto fundamental en el proceso de creación de cualquier sistema computacional, las entrevistas con el cliente interesado en el desarrollo del sistema es la fuente de información principal para la obtención de los requisitos, siendo el analista es el responsable de obtener del cliente la información necesaria para que se complete exitosamente el proyecto.

2.1.2.1 Requerimientos no funcionales

Los requerimientos no funcionales forman una parte significativa de la especificación. Son importantes para que los clientes y los usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con toda la funcionalidad requerida, las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

2.1.2.2 Requerimientos funcionales

Los requerimientos funcionales son capacidades o condiciones que debe cumplir el sistema. En este subproceso se obtienen las actividades que serán objeto de automatización. Estas actividades no son exactamente los requerimientos funcionales, pero son el punto de partida para identificar qué debe hacer el sistema. Los requerimientos funcionales se mantienen invariables sin importar con que propiedades o cualidades se relacionen. Como resultado de las mismas se determinan que los requisitos funcionales con los cuales debe cumplir este trabajo son los siguientes:

1. Almacenar los datos referentes a las distintas asignaturas contenidas en el plan de estudio.
2. Almacenar los datos de los diferentes cursos de postgrado que se imparten por los profesores del DCC.
3. Almacenar los datos de los profesores del DCC.
4. Gestionar las asesorías o tutorías de los distintos profesores a estudiantes.
5. Modificar cualquier dato almacenado por el sistema.
6. Obtener reportes tanto de la actividad docente en pregrado como en postgrado así como la tutoría de estudiantes.

Al referirse a la gestión de los datos, el sistema se encarga de la inserción, modificación, eliminación, y muestra de los mismos, dando así al usuario facilidades de manipulación y almacenamiento de estos en una BD. A partir de estos requisitos se pueden identificar varios casos de uso del sistema. La figura 2.1 muestra el diagrama de actores y casos de uso del sistema.

2.1.3 Descripción del actor

Actor	***Descripción***
Usuario	Este actor tiene la posibilidad de realizar todas las acciones del sistema, como son administrar la base de datos y consultar todos los reportes.

2.1.2. Casos de uso

Según Larman (Larman 1999), los casos de uso conforman un documento narrativo que describe la secuencia de eventos de un actor (agente externo) que utiliza un sistema para completar un proceso. Los casos de uso son historias o casos de utilización de un sistema; no son exactamente los requerimientos ni las especificaciones funcionales, sino que ejemplifican e incluyen tácitamente los requerimientos en las historias que narran. Según Jacobson (Jacobson, Booch et al. 2000), un caso de uso es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado importante. Los casos de uso representan los requisitos funcionales. Todos los casos de uso juntos constituyen el modelo de casos de uso, el cual describe la funcionalidad total del sistema.

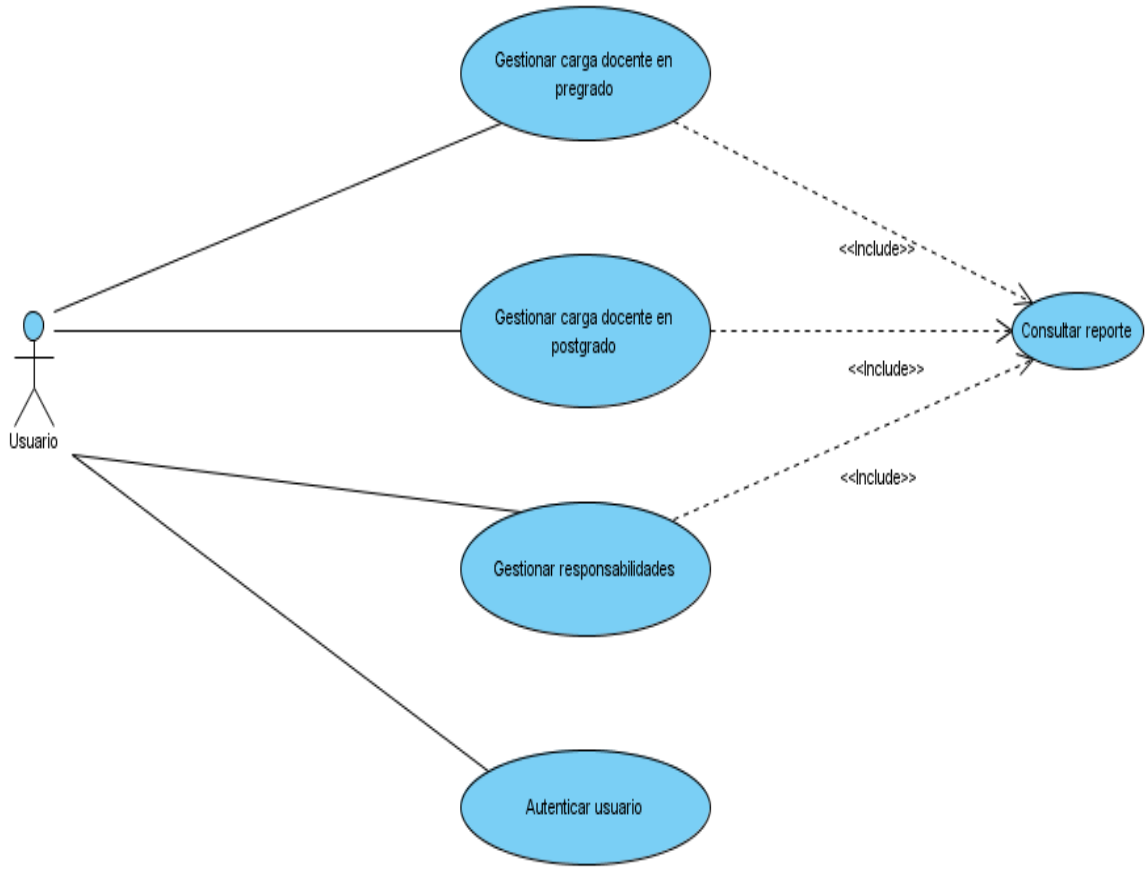

Figura 2.1. Diagrama de actores y casos de uso del sistema.

A continuación se describen los casos de uso más significativos del sistema.

Caso de uso "Gestionar carga docente en pregrado"

Caso de uso:	Gestionar carga docente de pregrado.
Actor:	Usuario.
Propósito:	Gestionar los datos referentes al trabajo de pregrado.
Resumen:	El usuario selecciona la opción Pregrado, escoge el escenario que desea, introduce los datos y se modifica la BD.
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**

1. El usuario selecciona la opción Pregrado.	2. Muestra una serie de escenarios: a. Alumno Pregrado, b. Año, c. Asignatura, d. Carrera, e. Curso Semestre, f. Disciplina, g. Facultad, h. Profesor, i. Profesor Imparte Asignatura Curso, j. Año Tiene Curso, k. Profesor Jefe Año Curso, l. Profesor Jefe Carrera Curso, m. Profesor Jefe Disciplina Curso, n. Profesor Tutor Alumno Pregrado.
3. El usuario elige el escenario a)	4. Muestra la ventana Alumno Pregrado con los siguientes campos: Id del Alumno, Nombre y Apellidos del Alumno
5. El usuario selecciona el botón de acceso rápido Insertar.	6. Muestra la ventana Insertar Alumno Pregrado con los siguientes campos: Id del Alumno, Nombre y Apellidos del Alumno.
7. El usuario llena los campos correspondientes y selecciona el botón Aceptar.	8. Comprueba que no existan campos sin datos y que en la BD no exista ya una fila con el mismo Id.
	9. Almacena en la BD los datos introducidos.
5. El usuario selecciona una fila de la tabla y luego el botón de acceso rápido Modificar.	6. Muestra la ventana Modificar Alumno Pregrado con el siguiente campo: Nombre y Apellidos del Alumno.
7. Llena el campo correspondiente y selecciona el botón Aceptar.	8. Comprueba que este campo esté lleno.
	9. Almacena en la BD los datos introducidos.
5. El usuario selecciona	6. Muestra una ventana para que el usuario ratifique la

una fila de la tabla y luego el botón de acceso rápido Eliminar.	eliminación.
7. Selecciona el botón Sí.	8. Elimina los datos de la BD.
7. Selecciona el botón No.	8. Se mantiene en la ventana.
5. El usuario selecciona el botón de acceso rápido Guardar.	6. Muestra la ventana Guardar.
7. Selecciona la carpeta donde desea guardar los datos así como el formato de documento en que los desea almacenar y el nombre del mismo.	8. Comprueba que todos los campos estén llenos.
	9. Almacena los datos en un documento con el nombre y formato seleccionado en la carpeta especificada.
3. El usuario elige el escenario h)	4. Muestra la ventana Profesor con los siguientes campos: Id_Profesor, CI, Nombre y Apellidos, Dpto. Docente, Cat. Docente, Grado Científico, Especialidad, PTP (Profesor a Tiempo Parcial)
5. El usuario selecciona el botón de acceso rápido Insertar.	6. Muestra la ventana Insertar Profesor con los siguientes campos: Id_Profesor, CI, Nombre y Apellidos, Dpto. Docente, Cat. Docente, Grado Científico, Especialidad, PTP.
7. Llena los campos correspondientes y selecciona el botón Aceptar.	8. Comprueba que no existan campos sin datos y que en la BD no exista una fila con el mismo Id.

	9. Almacena en la BD los datos introducidos.
5. El usuario selecciona una fila de la tabla y luego el botón de acceso rápido Modificar.	6. Muestra la ventana Modificar Profesor con los siguientes campos: CI, Nombre y Apellidos, Dpto. Docente, Cat. Docente, Grado Científico, Especialidad, PTP.
7. Llena los campos que desea modificar y selecciona el botón Aceptar.	8. Comprueba que no existan campos sin datos.
	9. Almacena en la BD los datos introducidos.
5. El usuario selecciona una fila de la tabla y luego el botón de acceso rápido Eliminar.	6. Muestra una ventana para que el usuario ratifique la eliminación.
7. Selecciona el botón Sí.	8. Elimina los datos en la BD.
7. Selecciona el botón No.	8. Se mantiene en la ventana
5. El usuario selecciona el botón de acceso rápido Guardar.	6. Muestra la ventana Guardar.
7. Selecciona la carpeta donde desea guardar los datos así como el formato de documento en que los desea almacenar y el nombre del mismo	8. Comprueba que todos los campos estén llenos.
	9. Almacena los datos en un documento con el nombre y formato seleccionado en la carpeta que se especificó.

Caso de uso "Gestionar carga docente en postgrado"

Caso de uso:	Gestionar carga docente de postgrado.
Actor:	Usuario.
Propósito:	Gestionar los datos referentes al trabajo de postgrado.
Resumen:	El usuario selecciona la opción Postgrado, escoge el escenario que desea, introduce los datos y se modifica la BD.
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El usuario selecciona la opción Postgrado.	2. Muestra una serie de escenarios: a. Alumno Postgrado, b. Curso, c. Programa postgrado, d. Tipo de Programa, e. Curso Semestre, f. Profesor Imparte Curso en Curso Semestre, g. Profesor, h. Profesor Tutor Alumno Postgrado Curso
3. El usuario elige el escenario a)	4. Muestra la ventana Alumno Postgrado con los siguientes campos: Id del Alumno, Nombre y Apellidos del Alumno, Fecha de Discusión.
5. El usuario selecciona el botón de acceso rápido Insertar.	6. Muestra la ventana Insertar Alumno Pregrado con los siguientes campos: Id del Alumno, Nombre y Apellidos del Alumno, Fecha de Discusión.
7. Llena los campos correspondientes y selecciona el botón Aceptar.	8. Comprueba que no existan campos sin datos que en la BD no exista una fila con el mismo Id.
	9. Almacena en la BD los datos introducidos.

5. El usuario selecciona una fila de la tabla y luego el botón de acceso rápido Modificar.	6. Muestra la ventana Modificar Alumno Postgrado con los siguientes campos: Nombre y Apellidos del Alumno, Fecha de Discusión.
7. Llena los campos correspondiente y selecciona el botón Aceptar.	8. Comprueba que los campos estén llenos.
	9. Almacena en la BD los datos introducidos.
5. El usuario selecciona una fila de la tabla y luego el botón de acceso rápido Eliminar.	6. Muestra una ventana para que el usuario ratifique la eliminación.
7. Selecciona el botón Sí.	8. Elimina los datos en la BD.
7. Selecciona el botón No.	8. Se mantiene en la ventana.
5. El usuario selecciona el botón de acceso rápido Guardar.	6. Muestra la ventana guardar.
7. Selecciona la carpeta donde desea guardar los datos así como el formato de documento en que los desea almacenar y el nombre del mismo	8. Comprueba que todos los campos estén llenos.
	9. Almacena los datos en un documento con el nombre y formato seleccionado y los guarda en la carpeta especificada.

3. El usuario elige el escenario c)	4. Muestra la ventana Programa de Postgrado con los siguientes campos: Id_Programa, Profesor Coordinador, Nombre del Programa, Edición, Tipo de Programa.
5. El usuario selecciona el botón de acceso rápido Insertar.	6. Muestra la ventana Insertar Programa de Postgrado con los siguientes campos: Id_Programa, Profesor Coordinador, Nombre del Programa, Edición, Tipo de Programa.
7. Llena los campos correspondientes y selecciona el botón Aceptar.	8. Comprueba que no existan campos sin datos y que en la BD no exista una fila con el mismo Id.
	9. Almacena en la BD los datos introducidos.
5. El usuario selecciona una fila de la tabla y luego el botón de acceso rápido Modificar.	6. Muestra la ventana Modificar Programa de Postgrado con los siguientes campos: Profesor Coordinador, Nombre del Programa, Edición, Tipo de Programa.
7. Llena los campos que desea modificar y selecciona el botón Aceptar.	8. Comprueba que no existan campos sin datos.
	9. Almacena en la BD los datos introducidos.
5. El usuario selecciona una fila de la tabla y luego el botón de acceso rápido Eliminar.	6. Muestra una ventana para que el usuario ratifique la eliminación.
7. Selecciona el botón Sí.	8. Elimina los datos en la BD.
7. Selecciona el botón No.	8. Se mantiene en la ventana

5. El usuario selecciona el botón de acceso rápido Guardar.	6. Muestra la ventana Guardar.
7. Selecciona la carpeta donde desea guardar los datos así como el formato de documento en que los desea almacenar y el nombre del mismo	8. Comprueba que todos los campos estén llenos.
	9. Almacena los datos en un documento con el nombre y formato seleccionado y los guarda en la carpeta especificada.

Caso de uso "Consultar Reportes"

Caso de Uso:	Consultar Reportes.
Actor:	Usuario.
Propósito:	Consultar los distintos reportes referentes al trabajo de los profesores.
Resumen:	El usuario selecciona la opción Reportes y escoge este escenario.
Flujo normal de eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción Reportes.	2. Muestra el escenario Reportes.
3. El usuario elige este escenario.	4. Muestra la ventana Reportes con los siguientes tipos de reportes y los campos curso y semestre para que el usuario escoja de cuál de ellos quiere conocer la información:

	Todo por Profesor, Todo un Profesor (Especificando el nombre del Profesor), Asignatura Postgrado, Carrera (Especificando la Carrera), Carga Postgrado, Carga Docente por Profesor, Servicios, Profesor Tutor Alumno
5. El usuario selecciona el reporte Todo por Profesor.	6. Muestra la ventana de dicho reporte con los siguientes campos: Nombre del Profesor, Nombre de la Asignatura, Cantidad de Horas de la Asignatura, Procedencia de la Asignatura, Fecha de Inicio, Fecha de Fin.
7. Selecciona la Opción de Guardar el reporte.	8. Muestra la ventana Guardar.
9. Selecciona la carpeta donde desea guardar los datos así como el formato de documento en que los desea almacenar y el nombre del mismo	10. Comprueba que todos los campos estén llenos.
	11. Almacena los datos en un documento con el nombre y formato seleccionado y los guarda en la carpeta especificada.
7. Selecciona la Opción de imprimir el reporte.	8. Muestra la ventana imprimir.
9. Selecciona el botón Aceptar	9. Imprime el reporte.
5. El usuario selecciona el reporte Carga de Postgrado y selecciona el curso y el semestre en el cuál desea obtener dicha información.	6. Muestra la ventana de dicho reporte con los siguientes campos: Nombre del Profesor, Nombre de la Asignatura, Cantidad de Horas, Procedencia, Fecha de Inicio, Fecha de Fin.

7. Selecciona la opción de Guardar el reporte.	8. Muestra la ventana Guardar.
9. Selecciona la carpeta donde desea guardar los datos así como el formato de documento en que los desea almacenar y el nombre del mismo	10. Comprueba que todos los campos estén llenos.
	11. Almacena los datos en un documento con el nombre y formato seleccionado y los guarda en la carpeta especificada.
7. Selecciona la opción de Imprimir el reporte.	8. Muestra la ventana Imprimir.
9. Selecciona el botón Aceptar	9. Imprime el reporte.
5. El usuario selecciona el reporte Todo un Profesor y se le especifica el nombre del profesor que desea buscar.	6. Muestra la ventana de dicho reporte con los siguientes campos: Nombre del Profesor, Nombre de la Asignatura, Cantidad de Horas, Procedencia, Fecha de Inicio, Fecha de Fin.
7. Selecciona la opción de Guardar el reporte.	8. Muestra la ventana Guardar.
9. Selecciona la carpeta donde desea guardar los datos así como el formato de documento en que los desea almacenar y el nombre del mismo	10. Comprueba que todos los campos estén llenos.

	11. Almacena los datos en un documento con el nombre y formato seleccionado y los guarda en la carpeta especificada.
7. Selecciona la opción de Imprimir el reporte.	8. Muestra la ventana Imprimir.
9. Selecciona el botón Aceptar	9. Imprime el reporte.

Tabla 2.1 Descripción de los casos de uso.

2.2 Esquema conceptual de la BD

El corazón de un sistema informacional es la BD que lo sostiene y partiendo de la arquitectura ANSI/SPARC de tres niveles. Con el modelo Entidad-relación como referencia (Chen 1976) en el esquema conceptual de la BD se representan tipos de entidades e interrelaciones que soportan los procesos que intervienen en el negocio. Un tipo de entidad puede ser un actor o la información obtenida de aplicar un proceso; mientras que las interrelaciones pueden expresar restricciones de dominio, asociaciones, especificaciones, generalizaciones, etc. La base de datos facilita la búsqueda y la navegación dentro del sistema a través de las consultas en lenguaje SQL. Las aplicaciones desarrolladas en casi todos los lenguajes contemporáneos incluyen herramientas de conexión e interacción completa con los SGBD más comunes. En la figura 2.2 se muestra el diagrama Entidad Relación para el sistema propuesto. Un buen diseño de la BD le imprime un carácter flexible a la información, pues los requerimientos de un sistema pueden cambiar fácilmente con la aparición de nuevas características. En concordancia con los casos de uso propuestos para el sistema informacional, la BD contiene tipos de entidades centrales para establecer un orden lógico en el soporte de estos procesos: Curso-Semestre, Profesor, Asignatura, etc. El tipo de entidad Profesor agrupa en su entorno la información personal, Curso-Semestre enmarca la fecha en la que se quieren gestionar los datos, etc.

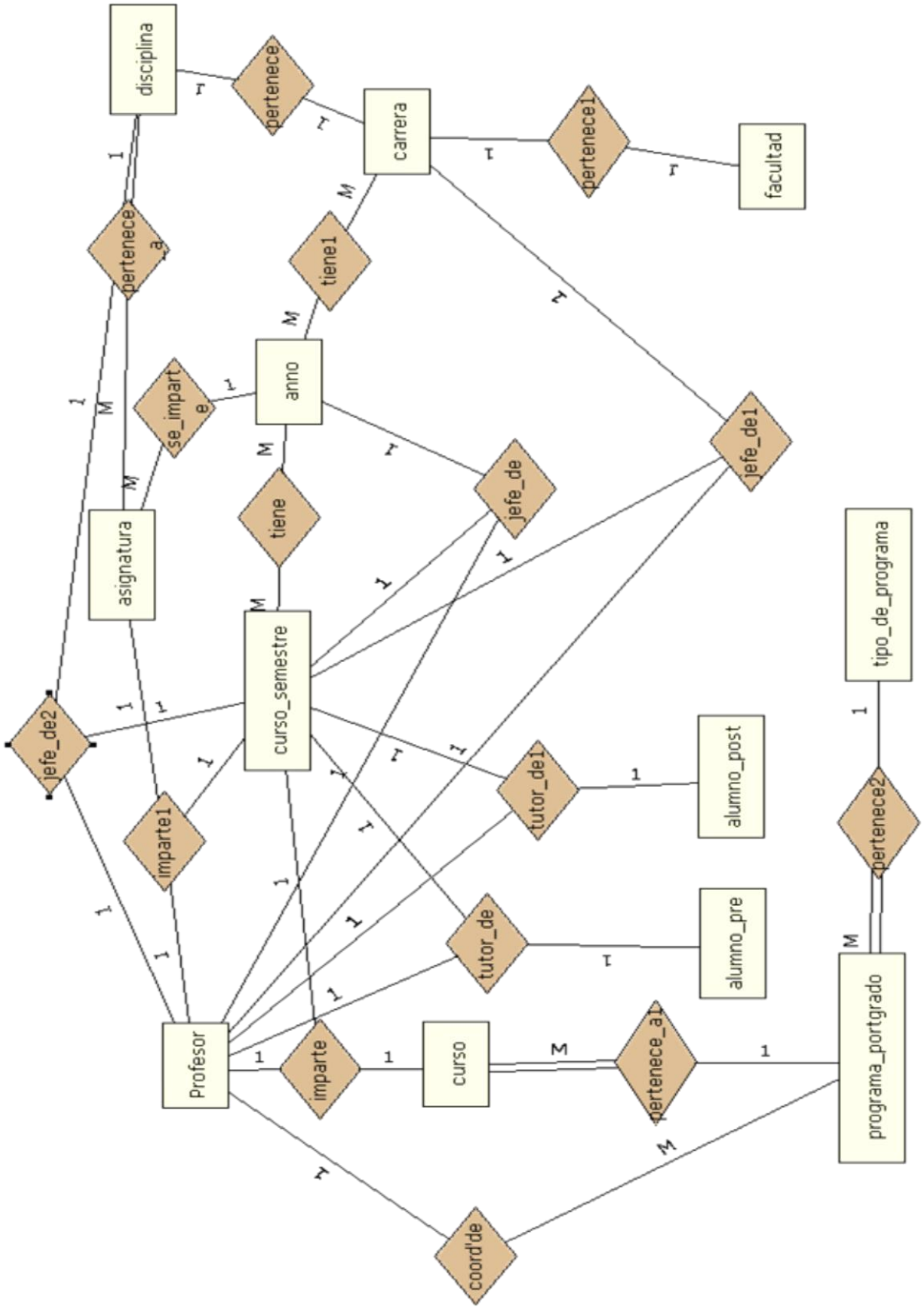

Figura 2.2 Diagrama Entidad-Relación de la BD.

Las distintas interrelaciones ternarias que se modelan en este diagrama generan tablas como es el caso de la interrelación ternaria "imparte1" que relaciona los tipos de entidades Profesor, Asignatura y Curso-Semestre en la cual se muestran los identificadores de profesores y asignaturas que se relacionan en un semestre dado de un curso. Además, las interrelaciones contribuyen a la implementación de características automáticas en el sistema al manejar la información, pues muchos componentes de los formularios se construyen siguiendo los patrones establecidos en el diseño de la BD.

2.3 Modelo lógico de la BD

El modelo lógico es resultado del análisis de requisitos. Este puede ser usado en la fase de diseño para la definición de los componentes de la aplicación, además, constituye una de las técnicas más utilizadas en el desarrollo orientado a objetos y describe formalmente la estructura de los elementos del sistema. Este diagrama describe para cada objeto su identidad y sus relaciones con otros objetos, o sea, describe un modelo general de información del sistema. Además del esquema lógico específico, se requieren otras entradas para la última etapa de la metodología de diseño de BD que es el diseño físico cuyo objetivo general es satisfacer los requisitos del sistema. Estas entradas son:

- Lista de objetivos de diseño físico con sus correspondientes prioridades y cuantificación,
- Recursos de máquina disponibles,
- Recursos de software disponibles (sistema operativo, middleware, etc.),
- Información sobre las aplicaciones que utilizarán la BD, y
- Políticas de seguridad de datos.

A partir de estas entradas, se producirán las siguientes salidas:

- Estructura interna (esquema interno),
- Especificaciones para el afinamiento (tunning) de la BD, y
- Normas de seguridad.

No existe un modelo formal general para el diseño físico, sino que depende mucho de cada producto comercial concreto.

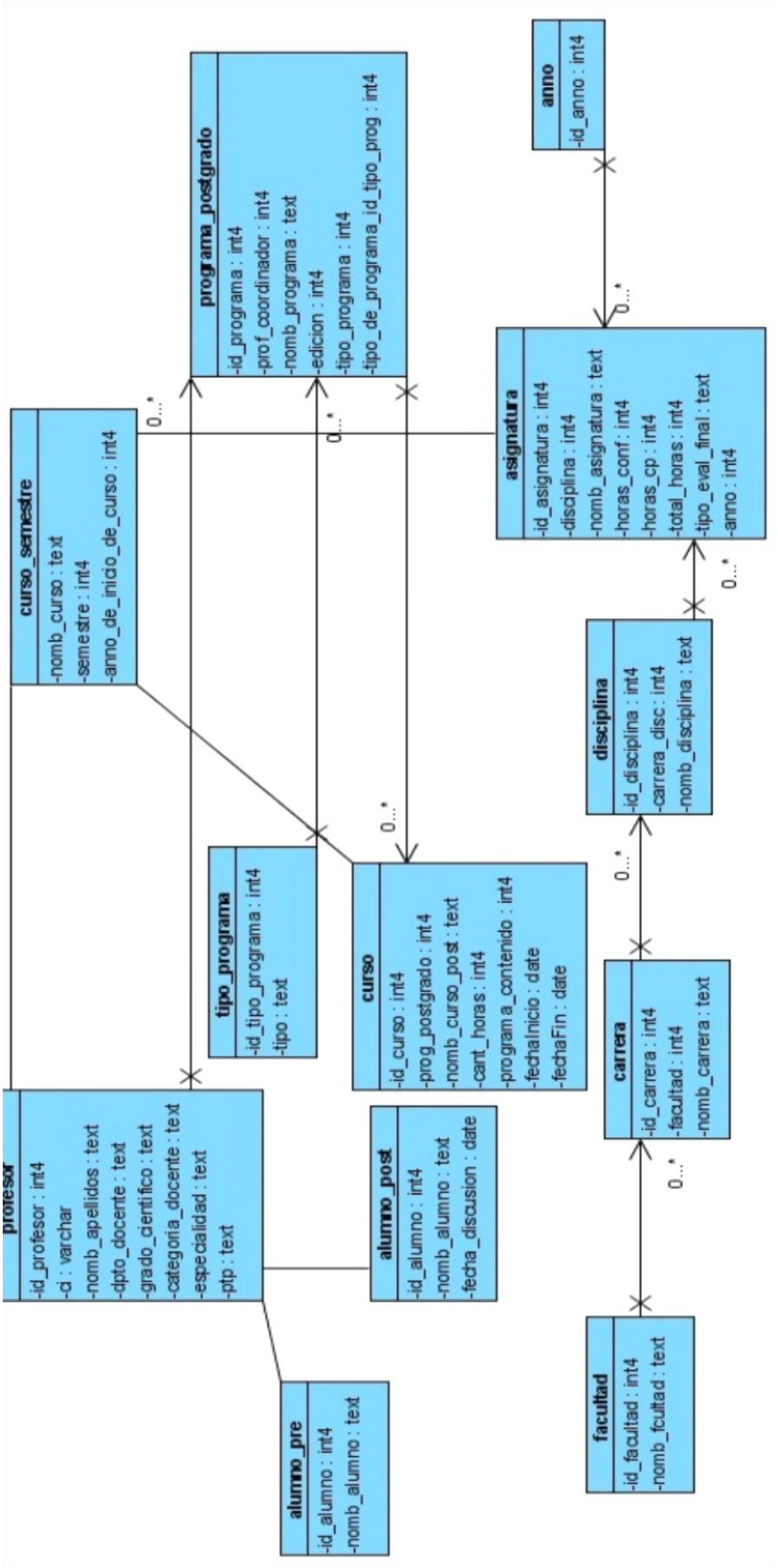

Figura 2.3 Esquema lógico de la BD.

2.4 Diagramas de despliegue

Según (Booch, Rumbaugh et al. 1999), los diagramas de despliegue son un tipo de diagrama del UML que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Algunos de los usos que se les da a los diagramas de despliegue son para modelar:

- Sistemas empotrados: Un sistema empotrado es una colección de hardware con una gran cantidad de software que interactúa con el mundo físico.
- Sistemas cliente-servidor: Los sistemas cliente-servidor son un extremo del espectro de los sistemas distribuidos y requieren tomar decisiones sobre la conectividad de red de los clientes a los servidores y sobre la distribución física de los componentes software del sistema a través de nodos.
- Sistemas completamente distribuidos: En el otro extremo se encuentran aquellos sistemas que son amplia o totalmente distribuidos, y que normalmente incluyen varios niveles de servidores. Tales sistemas contienen, a menudo, varias versiones de componentes software, alguno de los cuales pueden incluso migrar de un nodo a otro. El diseño de tales sistemas requiere tomar decisiones que permitan un cambio continuo de la topología del sistema.

Según (Wikipedia 2009) el modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. El modelo de despliegue se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño. Se puede observar lo siguiente sobre el modelo de despliegue:

- Cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo hardware similar.
- Los nodos poseen relaciones que representan medios de comunicación entre ellos, tales como Internet, Intranet, y similares.
- El modelo de despliegue puede describir diferentes configuraciones de red, incluidas las configuraciones para pruebas y para simulación.

- La funcionalidad (los procesos) de un nodo se define por los componentes que se distribuyen sobre ese nodo.
- El modelo de despliegue en sí mismo representa una correspondencia entre la arquitectura software y la arquitectura del sistema (el hardware).

En la figura 2.4 se muestra el diagrama de despliegue del sistema propuesto.

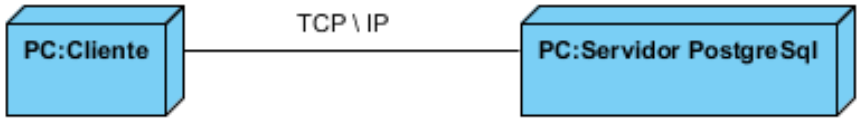

Figura 2.4 Diagrama de despliegue de la aplicación.

De este diagrama se puede decir que en la PC:Cliente es donde reside la aplicación, la cual a través de la conexión TCP/IP se conecta a la PC:Servidor PostgreSQL donde se encuentra la BD que utiliza la aplicación.

2.5 Patrón de diseño Modelo-Vista-Controlador

Modelo-Vista- Controlador es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. El patrón Modelo Vista Controlador se utiliza frecuentemente en aplicaciones Web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el modelo es el sistema de gestión de base de datos y la lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista (Díaz-Vellón and Gónzalez-Mena 2008). A continuación se ofrece una breve descripción del patrón:

- Modelo: Es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos.
- Vista: Presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- Controlador: Responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

Aunque se pueden encontrar diferentes implementaciones del patrón Modelo-Vista-Controlador, el flujo que sigue el control generalmente es el siguiente:

- El usuario interactúa con la interfaz (por ejemplo pulsa un botón de enlace)
- El controlador recibe (por parte de los objetos de la interfaz-vista) la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos (handler) o callback.
- El controlador accede al modelo actualizándolo de acuerdo a la solicitud del usuario. Los controladores complejos están a menudo estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.
- El controlador delega a los objetos de la vista, la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar una interfaz apropiada para el usuario donde refleja los cambios en el modelo.
- La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

La figura 2.5 muestra, mediante un sencillo diagrama, la relación entre el modelo, la vista y el controlador.

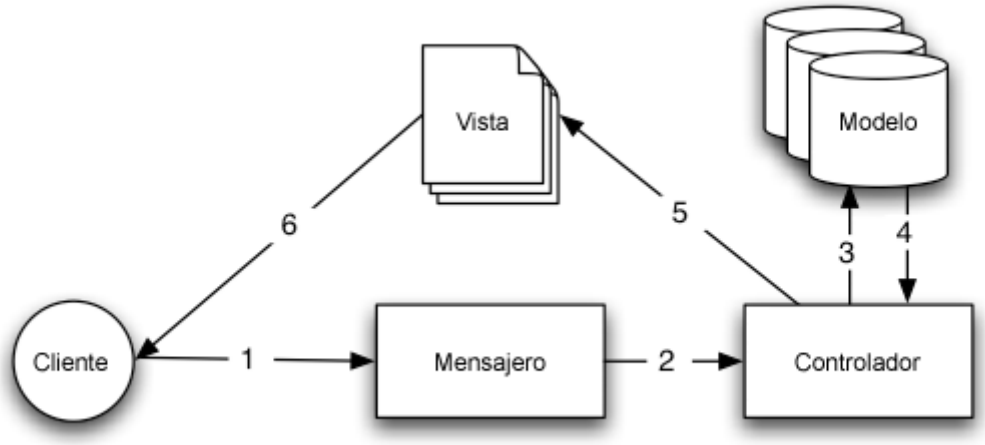

Figura 2.5 Representación del patrón Modelo-Vista Controlador.

2.7 Herramientas para el desarrollador

Para la creación del SPCCD se utilizó el lenguaje de programación Java y se conecta a un servidor de BD PostgreSQL versión 9.2. El entorno de desarrollo utilizado es la versión 7.1.2 del IDE NetBeans (NetBeans 2013).

Para continuar extendiendo el sistema se debe agregar al proyecto un conjunto de bibliotecas que fueron utilizadas en este desarrollo:

Bibliotecas	Uso de la Biblioteca
AbsoluteLayout.jar	Para el trabajo con las imágenes.
Jcalendar.jar	Para el trabajo con los campos de tipo fecha.
Jxl.jar	Para el trabajo con las tablas Excel a la hora de guardar los reportes.
nimrodlf-1.2.jar	Para modificar la parte visual del proyecto.
PostgreSQL-9.2-1002.jdbc3.jar	Para la conexión con el SGBD PostgreSQL.

Tabla 2.2 bibliotecas utilizadas en el desarrollo del software.

2.8 Conclusiones parciales

- Se parte de los requerimientos para realizar el análisis y diseño del sistema mediante el modelado de sus características principales utilizando UML.
- Se describieron el actor y los casos de uso más importantes.
- Se modeló la arquitectura de la información en sus niveles conceptual y lógico y se caracterizó el físico.
- Se empleó el patrón de diseño basado en el Modelo-Vista-Controlador para enfrentar la solución al problema siguiendo la estrategia divide-y-vencerás.

CAPITULO 3: USO DEL SISTEMA PARA LA PLANIFICACIÓN Y CONTROL DE LA CARGA DOCENTE (SPCCD).

En este capítulo se detallan los procesos de instalación y uso del Sistema para la Planificación y Control de la Carga Docente (SPCCD) del DCC.

3.1 Requerimientos del sistema

Para instalar cada uno de los componentes del sistema se deben tener en cuenta cada uno de los requisitos específicos de cada uno de ellos por separado, ya que los mismos no necesariamente tienen que estar instalados en la misma computadora.

3.1.1 Instalación de la BD

Para crear la BD y lograr un correcto funcionamiento de la misma se deben seguir los siguientes pasos:

- Tener un servidor de BD PostgreSQL 9.2.
- Crear una nueva BD en el servidor.
- Restaurar la BD creada en el paso 2) usando el archivo "BACKUP RESTAURED" que se encuentra en la carpeta "SPCCD" de la instalación.

3.1.2 Requerimientos e Instalación del "SPCCD.jar"

Antes de pasar a la instalación del SPCCD.exe, se deben revisar los siguientes requisitos:

- Procesador Pentium o superior.
- Al menos 256 Megabytes de memoria RAM.
- Microsoft Windows XP o superior.
- Máquina Virtual de Java jre7.
- PostgreSQL 9.2.

Si ya se han verificado los requerimientos anteriores, entonces se puede pasar a la instalación del SPCCD.jar.

3.2 Uso del SPCCD.exe

Al ejecutar el software SPCCD, se muestra su portada inicial como se observa en la figura 3.1.

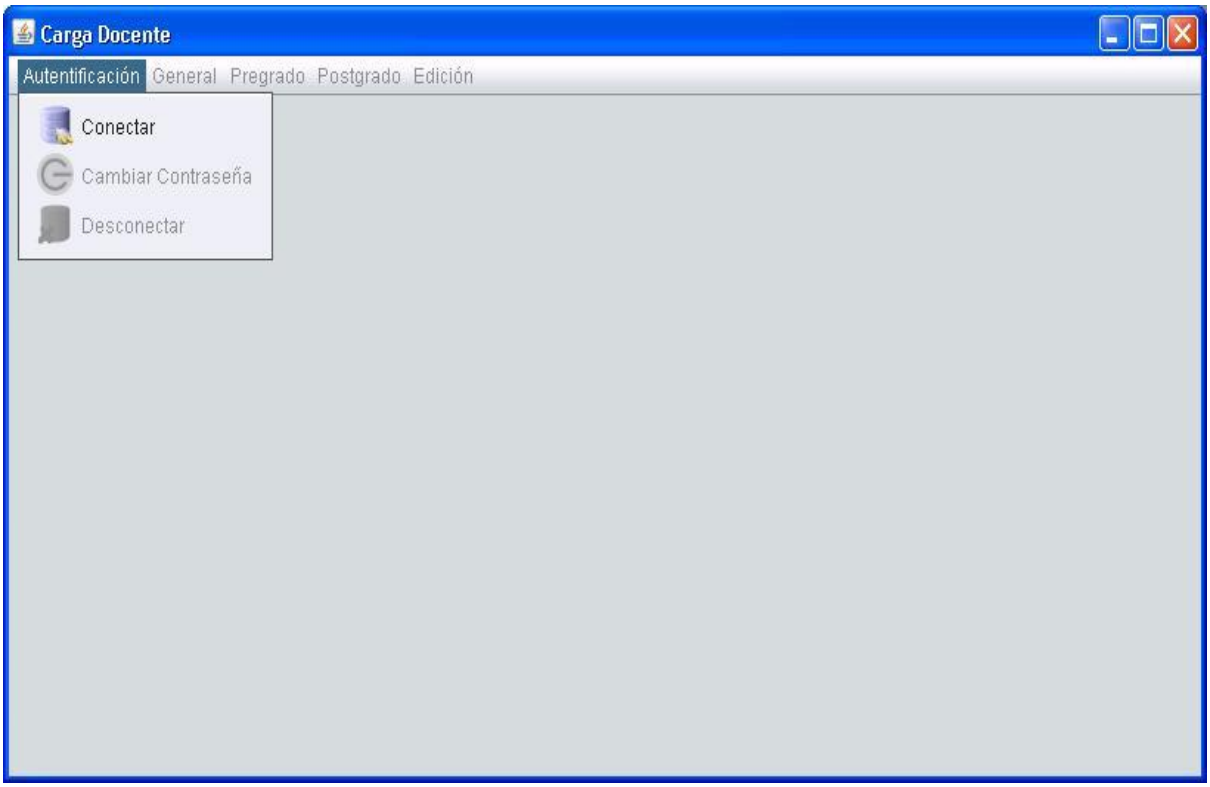

Figura 3.1 Portada de SPCCD.

Al mostrarse la misma, se habilita el menú Autentificación y la opción Conectar, dando clic en esta se muestra la ventana de la figura 3.2.

Figura 3.2 Ventana para la autentificación de los usuarios.

Como se puede observar, en la figura 3.2, para lograr la conexión de la aplicación con la BD sólo se necesita conocer la dirección IP del servidor de BD, el nombre y la contraseña del usuario; si todos los datos son correctos, se establece la conexión y se puede operar con la

BD. El sistema le brinda la posibilidad al usuario de cambiar su contraseña como se muestra en la figura 3.3.

Figura 3.3 Ventana para cambiar la contraseña de usuario.

El usuario introduce la nueva contraseña y la confirma para que el sistema pueda validarla. Ya conectado al servidor, el sistema habilita las demás opciones de menú excepto el de Edición ya que no se ha seleccionado manipular ningún dato en la BD. Si el usuario selecciona el menú General (véase la figura 3.4) se pueden manipular los datos referentes a los profesores y a los cursos, que como su nombre lo indica, son datos generales que se comportan de la misma manera para la actividad de postgrado y pregrado.

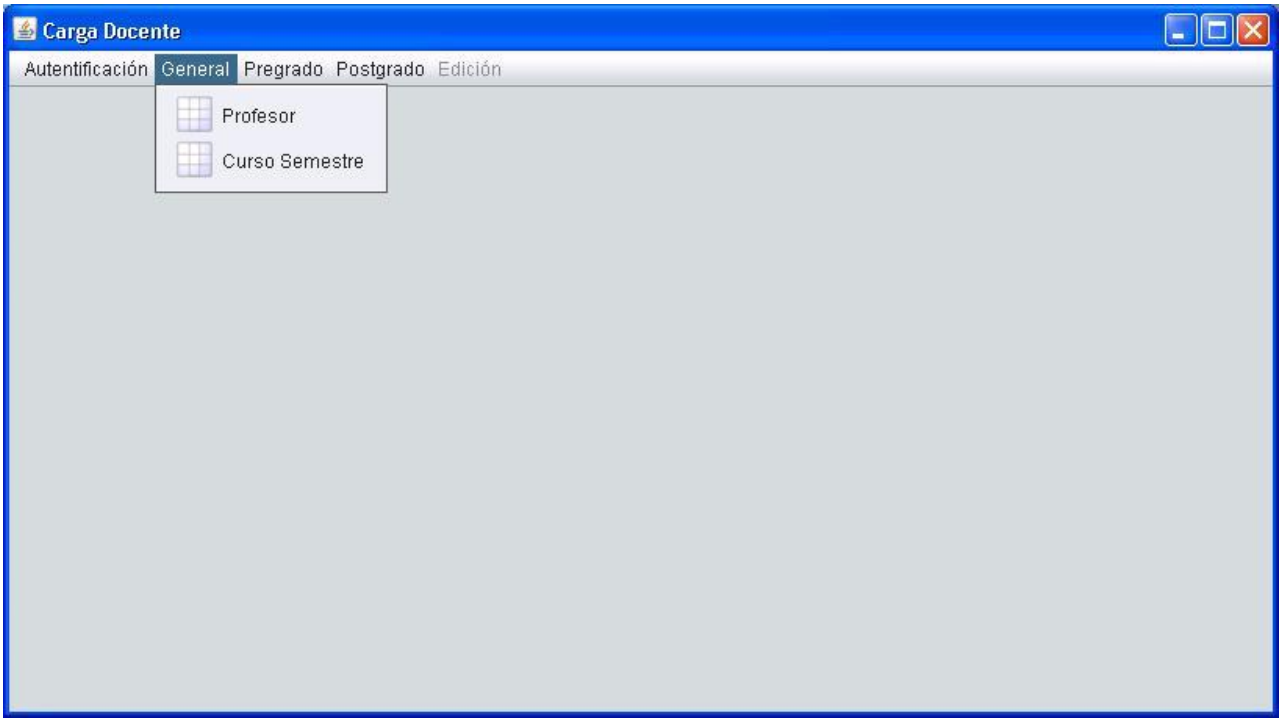

Figura 3.4 Menú General.

Si se accede a la opción Profesor se muestra una ventana como se observa en la figura 3.5, con una tabla en cuyos campos se muestra la información que se maneja en el subsistema de

los profesores así como cinco botones de acceso rápido, que de izquierda a derecha posibilitan:

- Insertar los datos de un nuevo profesor al sistema.
- Modificar los datos de un profesor dado.
- Eliminar los datos de un profesor dado.
- Actualizar los campos de la tabla que se muestra.
- Exportar los datos que se muestran en la tabla a formato de Microsoft Excel.

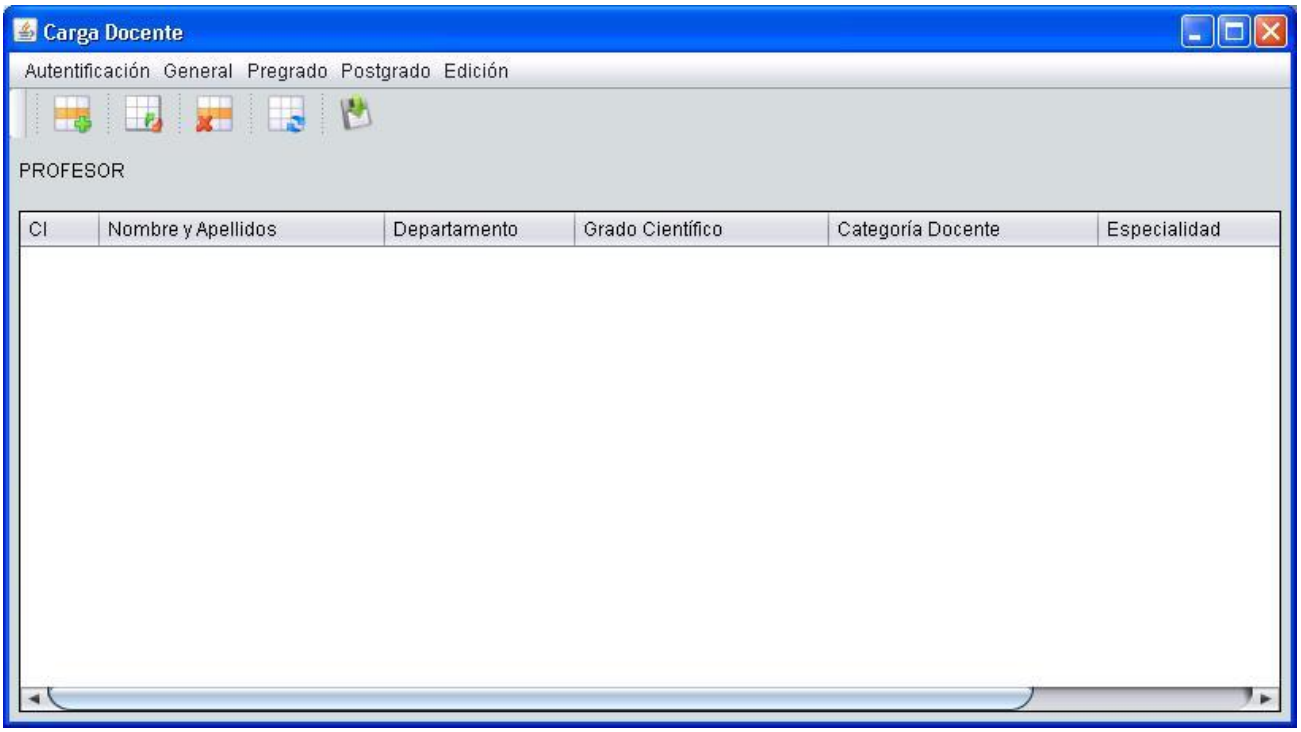

Figura 3.5 Ventana para el trabajo con los datos sobre los profesores.

Estos botones se muestran de manera análoga al seleccionar las demás opciones de menú para facilitar el trabajo con los datos; además se puede acceder a los mismos a través del menú Edición, como se muestra en la figura 3.6. Si se elige la opción Insertar se muestra la ventana que se observa en la figura 3.7 donde se ingresan los datos del profesor y estos se almacenan en la BD; así como para modificar los datos de un profesor determinado se selecciona éste en la tabla y se accede al botón de acceso rápido Modificar y se muestra una ventana similar a la de inserción de datos, con la diferencia de que sus campos están llenos con la información del docente actual, y el usuario tiene la posibilidad de editar los cambios que quiera realizar en los mismos; para eliminar los datos de un profesor se selecciona éste en la tabla y se accede al botón de acceso rápido Eliminar y el sistema le emite un mensaje

de confirmación para posteriormente eliminar dichos datos de la BD. De manera general, se procede de igual forma en todas las opciones que se seleccionen para gestionar los datos las cuales se pueden observar en las figuras 3.8 y 3.9.

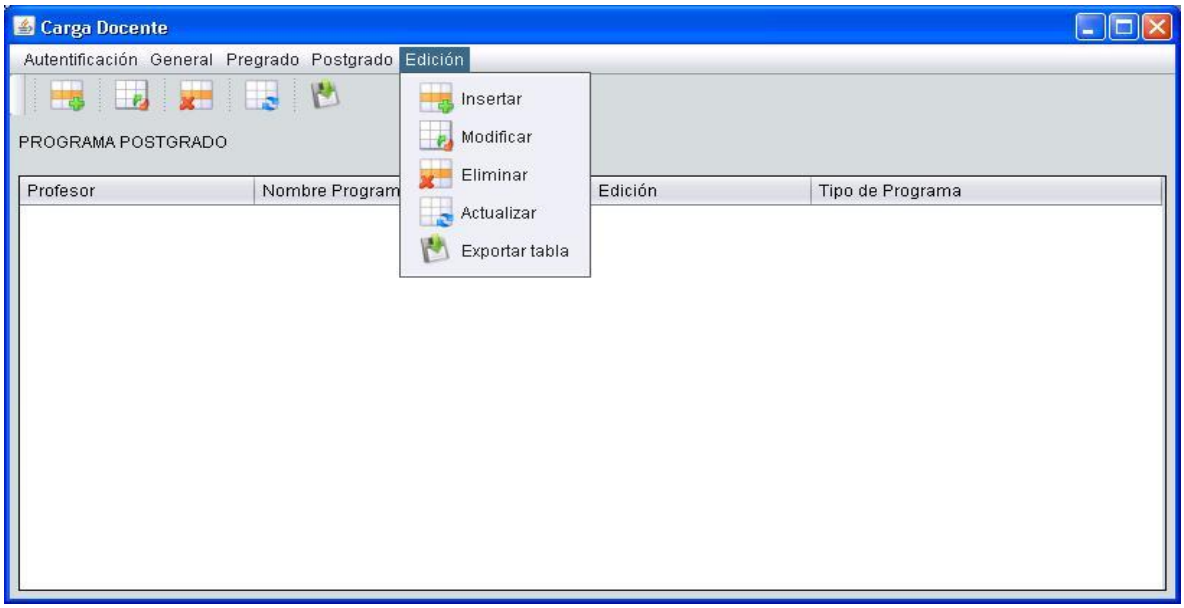

Figura 3.6 Menú Edición.

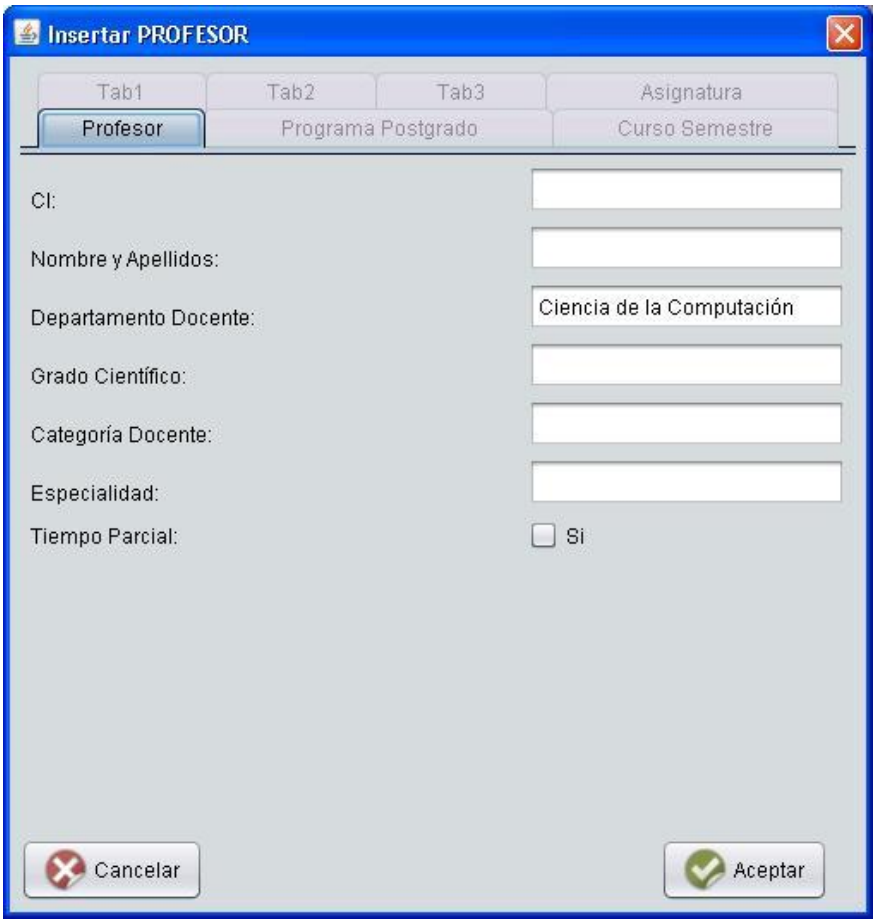

Figura 3.7 Captura de datos sobre los profesores.

En esta imagen se aprecian entre las pestañas deshabilitadas las llamadas *Tab1* que abarca las pestañas de Alumno de Postgrado, Año, Facultad, Tipo de Programa de Postgrado y Curso; *Tab2* que incluye las pestañas Profesor Imparte Asignatura Curso, Profesor Imparte Curso Curso Semestre, Profesor Jefe Año Curso, Profesor Jefe Carrera Curso, Profesor Jefe Disciplina Curso y Tutor mientras *Tab3* corresponde a las pestañas Alumno de Pregrado, Carrera y Disciplina.

Estas pestañas cuando el usuario seleccione una opción de menú distinta a las referidas por las mismas se mantendrán deshabilitadas y mantendrán esta nomenclatura pero si por el contrario el usuario seleccionase una de las opciones de menú correspondientes a las mismas se habilita la pestaña con este nombre específico como se muestra en las figuras siguientes

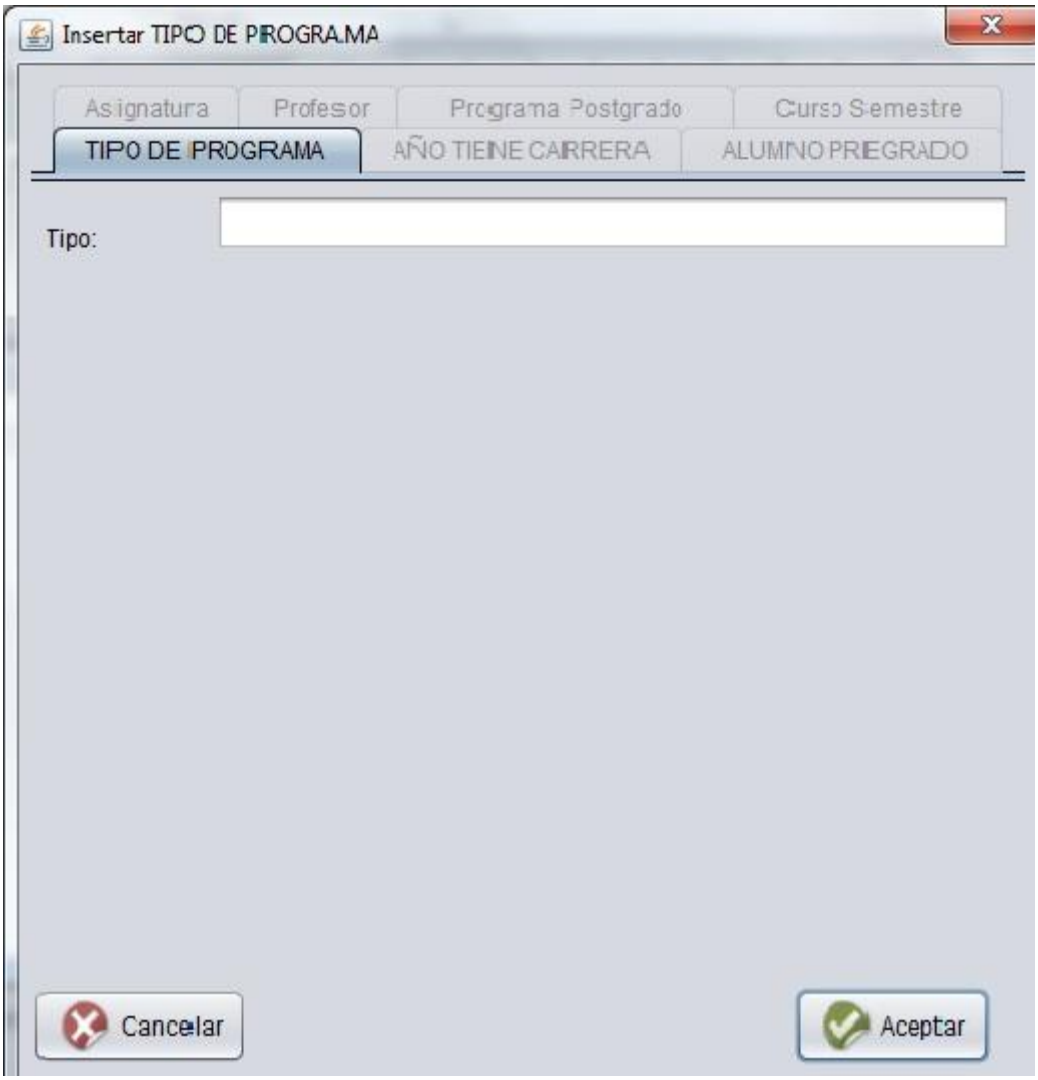

Figura 3.8 Ventana de inserción de un nuevo Tipo de Programa.

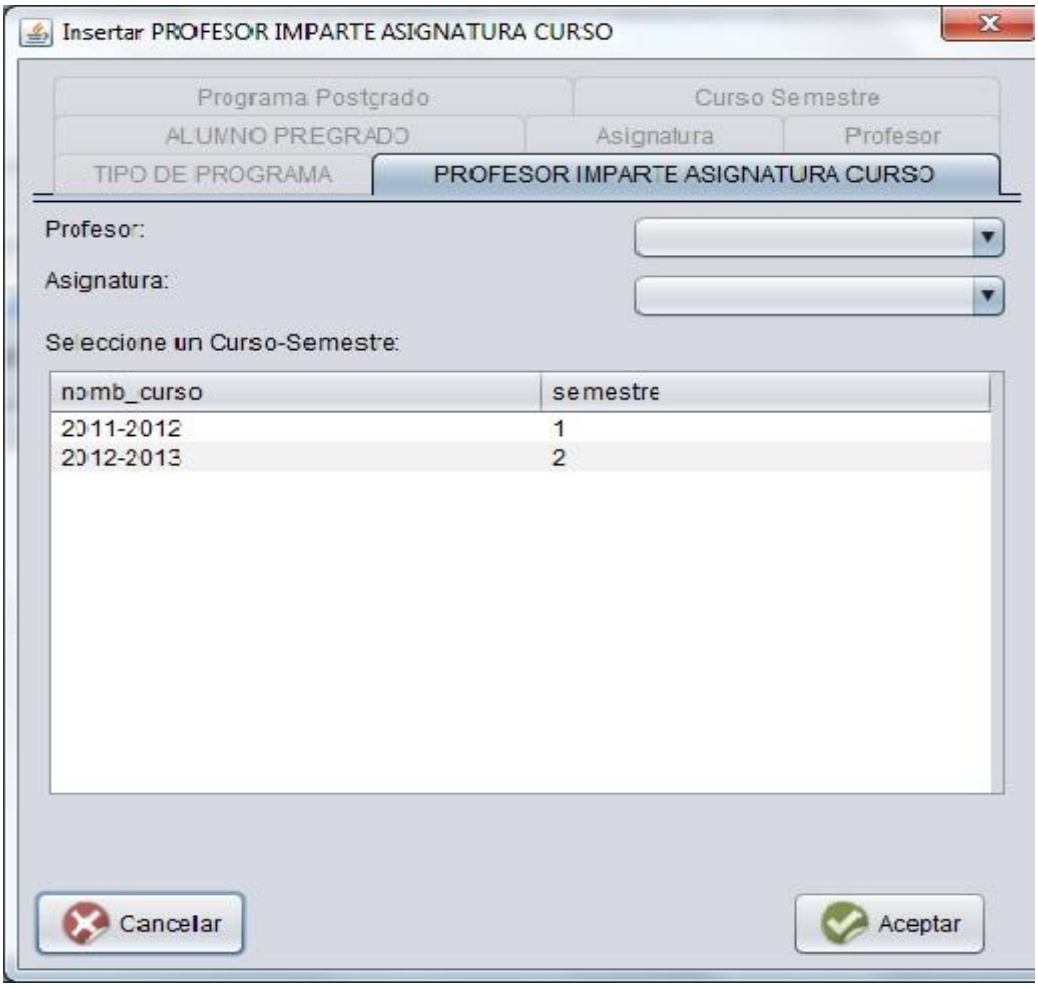

Figura 3.9 Ventana de inserción de una nueva relación Profesor-Asignatura

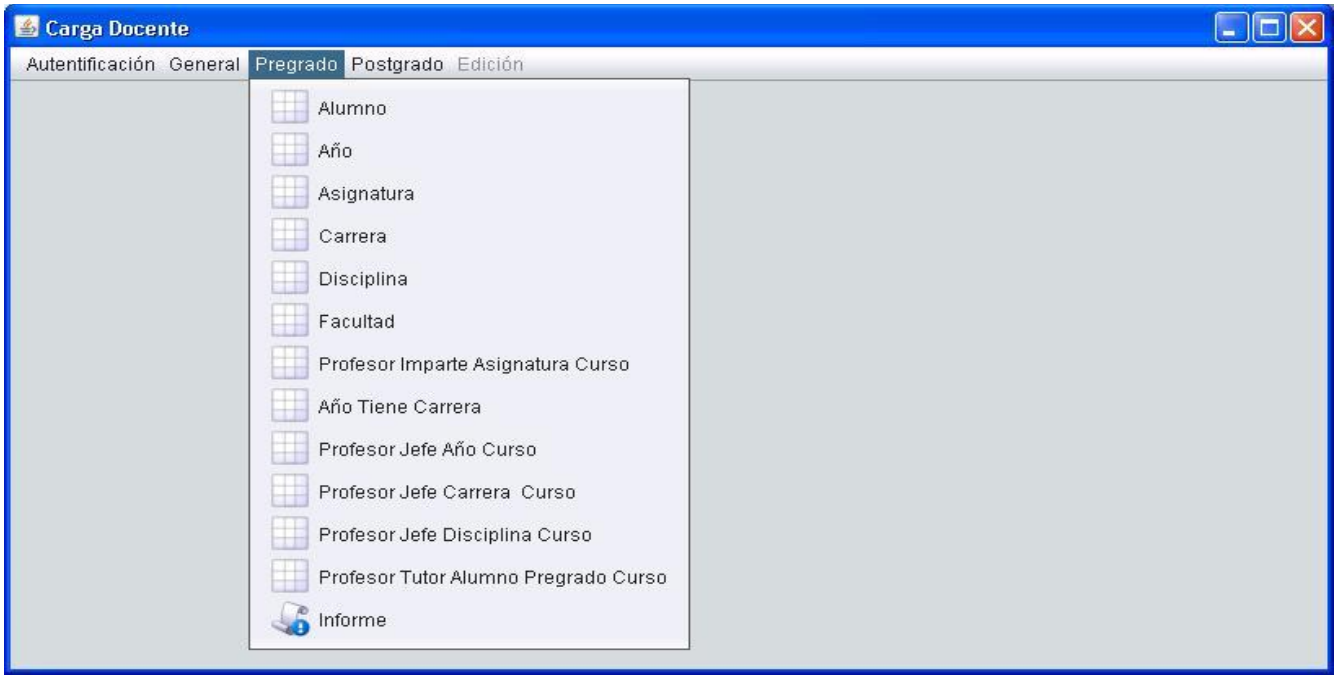

Figura 3.10 Menú Pregrado.

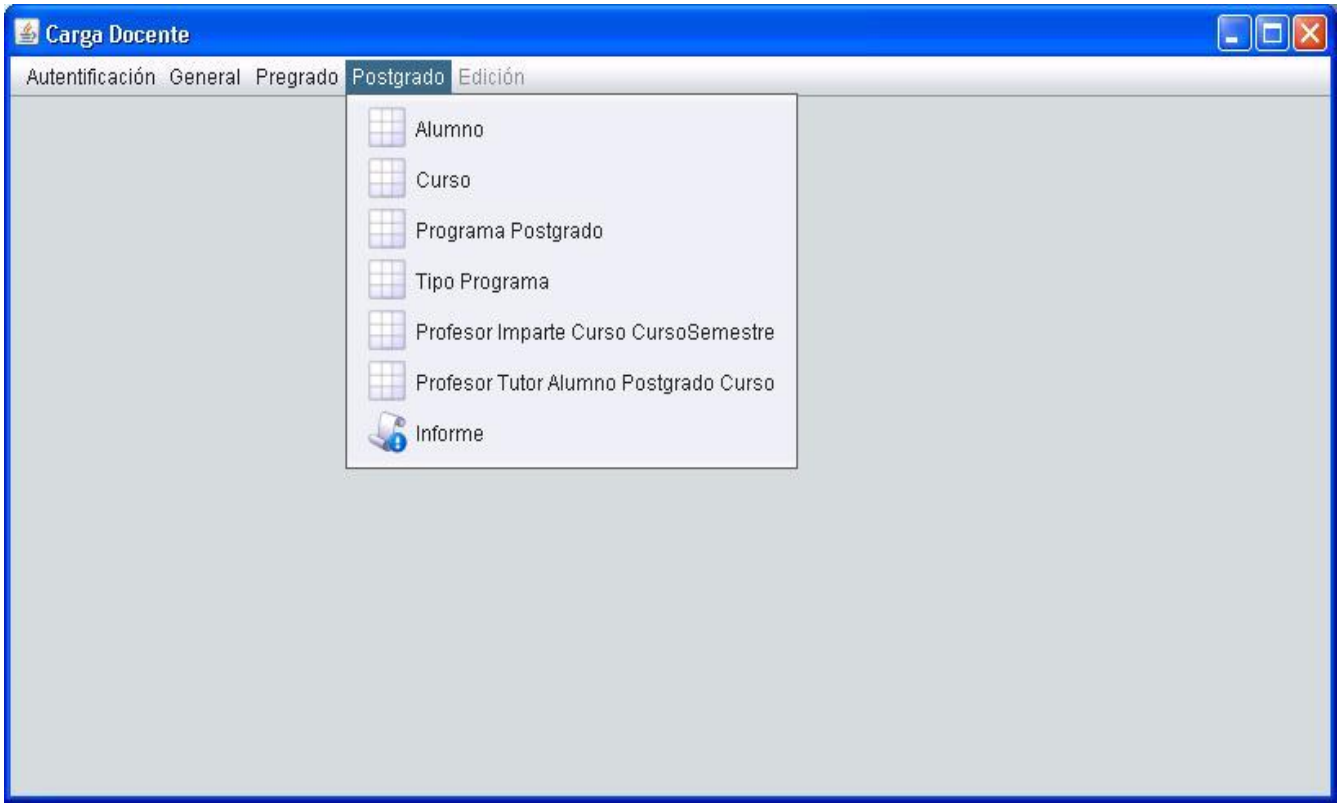

Figura 3.11 Menú Postgrado

Si se selecciona la opción Informe en el menú Pregrado o Postgrado se puede observar una ventana como la de la figura 3.10. donde se habilitan seis opciones que representan diferentes tipos de reportes que se le brindan al usuario acerca de la labor docente de los profesores; para acceder a los mismos se debe con seleccionar cuál desea observar y el curso y semestre en el cual desea analizar los datos anteriormente referidos.

Si se selecciona la opción Todo Un Profesor se muestra la ventana referida en la figura 3.11 donde se especifica cuál es el profesor que se desea analizar y el curso y semestre en el que se quiere ubicar este análisis. Posteriormente, se abre una nueva ventana donde se observa esta información resaltando el nombre y los apellidos del profesor, así como la(s) asignatura(s) que imparte con la cantidad de horas que tiene la misma y la carrera a la que pertenece; se brinda la posibilidad al usuario de imprimirla, exportarla a otros formatos como son PDF, DOC, HTML, etc. De manera similar se puede manejar la información referida en los demás tipos de reporte.

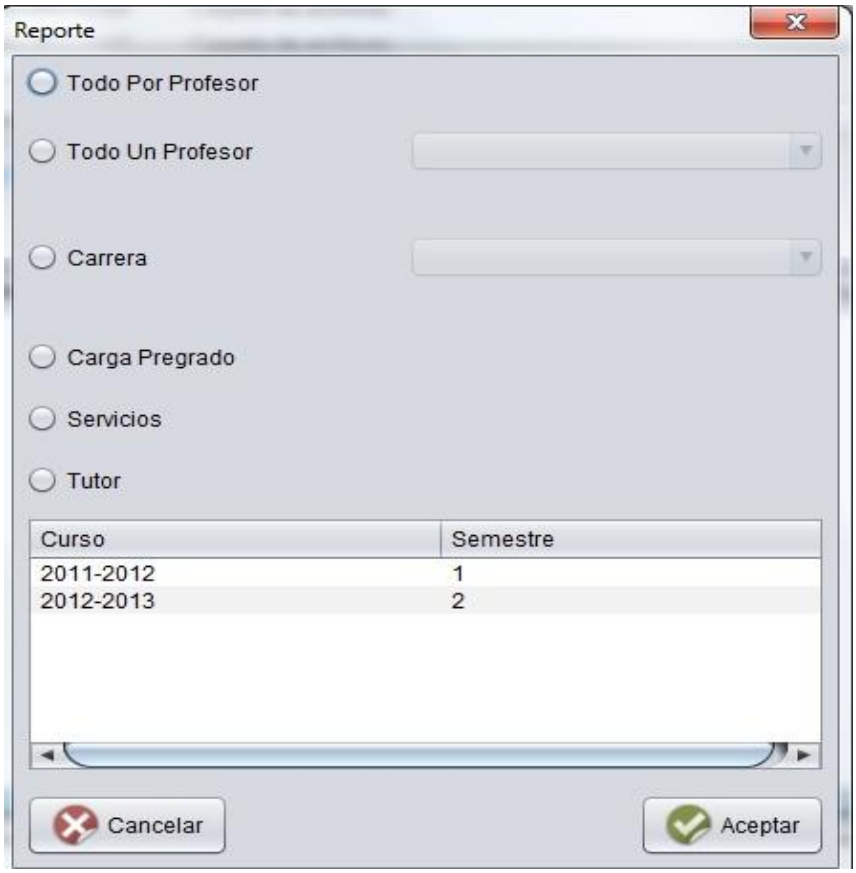

Figura 3.12 Selección de los reportes.

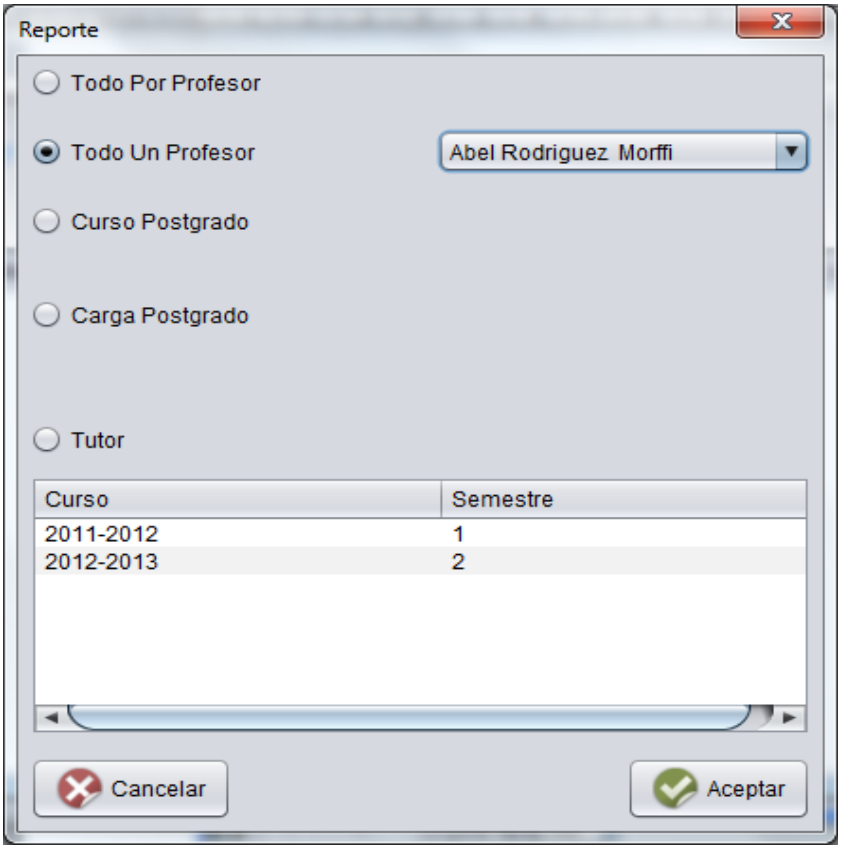

Figura 3.13 Selección de reporte Todo Un Profesor.

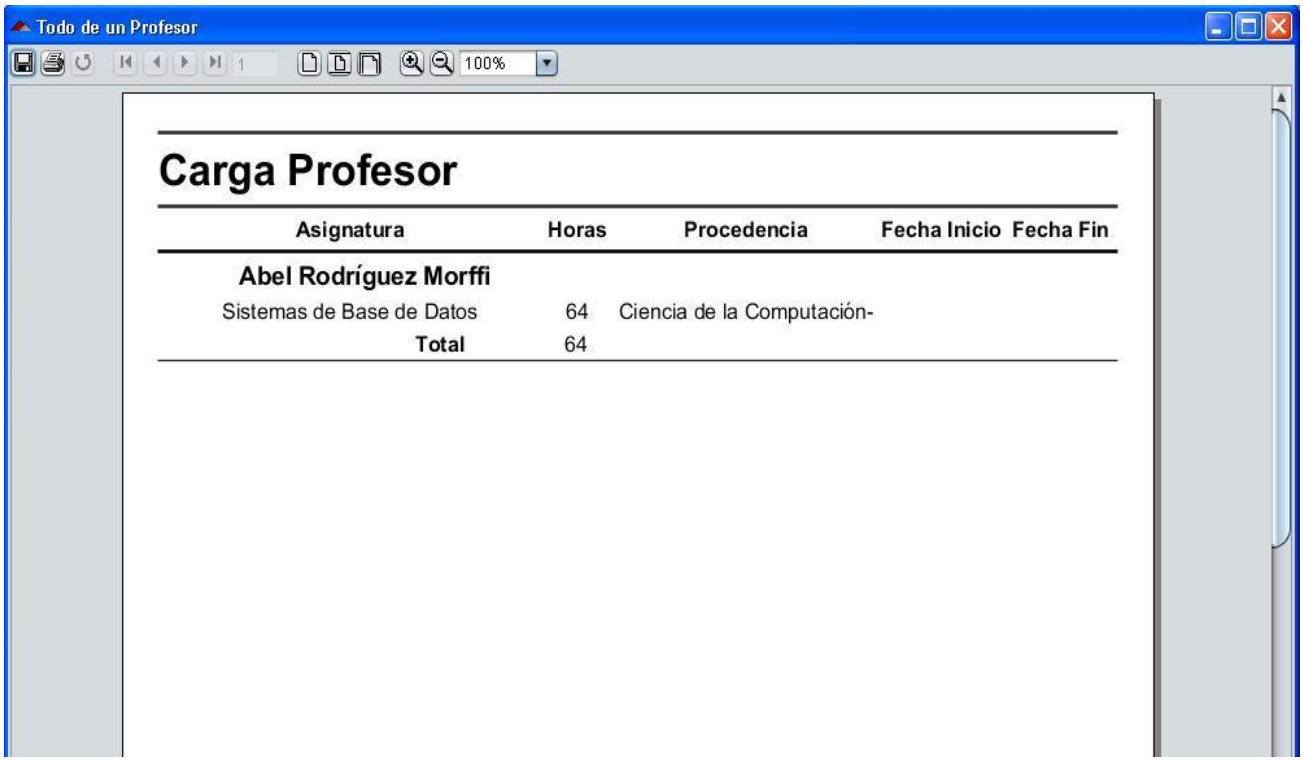

Figura 3.14 Reporte de la carga docente de un profesor.

En la figura 3.13 imagen se muestra la vista del reporte Tutor Pregrado donde se observan las relaciones de tutoría o asesoría de estudiantes por parte de los profesores, señalándose el nombre y los apellidos del profesor y el nombre y los apellidos del alumno y la carrera a la que pertenece así como el laboratorio al que pertenece el trabajo.

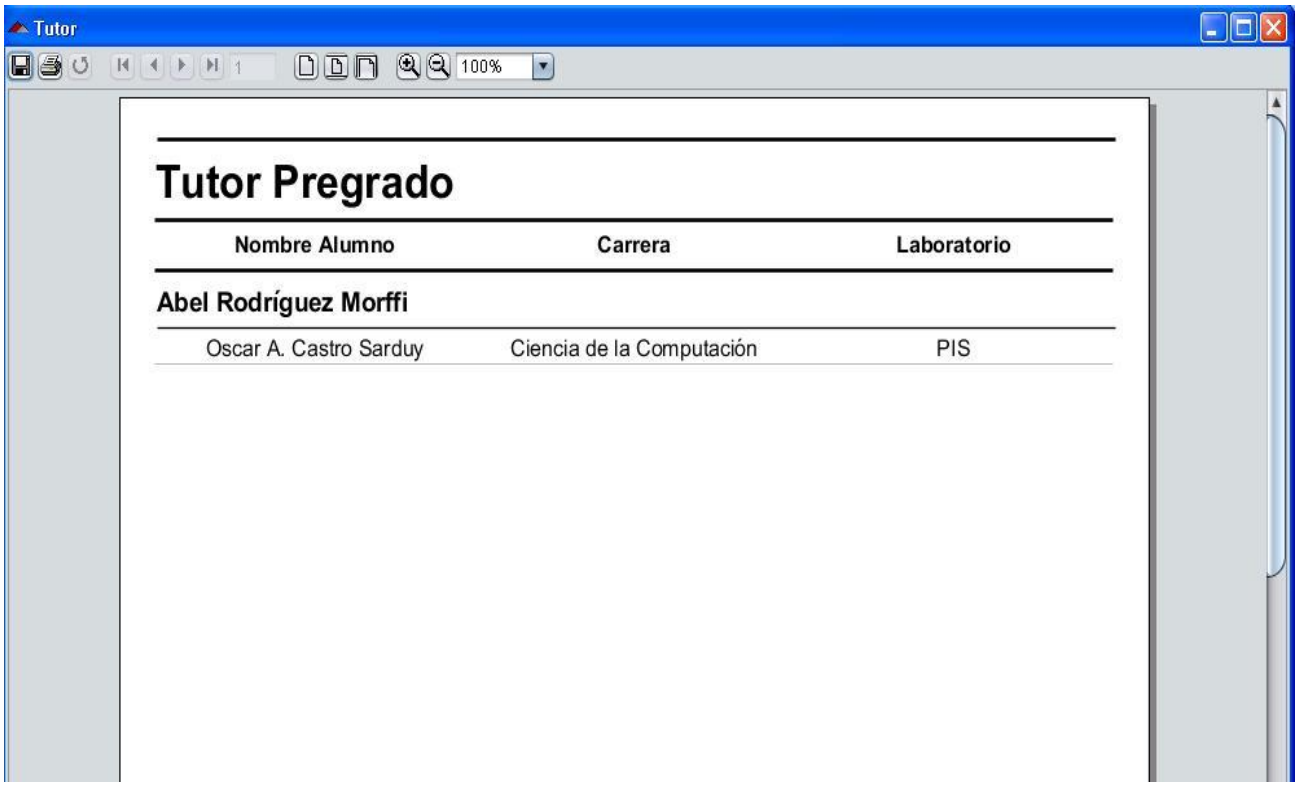

Figura 3.15 Reporte sobre tutorías.

3.3 Conclusiones parciales

- Se determinaron los requerimientos para la instalación y uso del sistema propuesto.
- Se diseñó una interfaz informacional para manipular la base de datos creada lográndose así un SI que permite llevar a cabo la planificación y control de la actividad docente a lo largo de varios cursos académicos, así como las asesorías y tutorías de alumnos de pregrado y postgrado.
- El sistema creado brinda toda información para los procesos de asignación de carga docente, seguimiento al proceso docente desde el punto de vista de carga y responsabilidades útil para el análisis del cumplimiento del plan de trabajo y evaluación de profesores, asignaturas, disciplinas y carreras.

CONCLUSIONES

1. Se pudo diseñar la arquitectura de la información para la planificación y control de la actividad docente a lo largo de diferentes cursos académicos, así como las asesorías y tutorías de alumnos de pregrado y postgrado. Para ello se utilizaron los estándares de diseño y se siguió, desde el punto de vista metodológico, la arquitectura ANSI/SPARC de tres niveles: conceptual, lógico y físico.
2. Se creó la base de datos conforme a la arquitectura de la información obtenida en el SGBD PostgreSQL, cumpliendo con las indicaciones acerca del uso del software libre en Cuba y por ser este gestor más usado internacionalmente dentro de la categoría de libres y open source.
3. Se concibió una interfaz informacional para manipular la base de datos creada lográndose un SI que permite llevar a cabo la planificación y control de la actividad docente a lo largo de varios cursos académicos, así como las asesorías y tutorías de alumnos de pregrado y postgrado.
4. Este sistema brinda información útil para los procesos de asignación de carga docente y evaluación de profesores, asignaturas, disciplinas y carreras.

RECOMENDACIONES

Derivadas del estudio realizado, así como de las conclusiones generales emanadas del mismo, se recomienda:

- Propiciar la implantación del sistema en otros Departamentos Docentes de la Universidad

REFERENCIAS BIBLIOGRÁFICAS

Booch, G., J. Rumbaugh, et al. (1999). El Lenguaje Unificado de Modelado. Addison-Wesley. Addison Wesley.

Cerda, F. (2011). NetBeans 6.5.

Chen, P. P.-S. (1976). "The entity-relationship model-toward a unified view of data." ACM Transactions on Database Systems (TODS) **1**(1): 9-36.

DCC (2007). Informe de autoevaluación para la acreditación de la carrera de Licenciatura en Ciencia de la Computación. Departamento de Ciencia de la Computación, Universidad Central "Marta Abreu" de Las Villas**:** 2-5.

Díaz-Vellón, M. and J. L. Gónzalez-Mena (2008). Modelación de reglas de negocio como apoyo para sistemas de información en el área de Nefrología. Departamento de Computacion. Santa Clara, Villa Clara, Universidad Central "Marta Abreu"de las Villas.

García de Jalón, J. and J. I. Rodríguez (1999). Aprenda Java como si estuviera en primero. San Sebastián.

Herrera, I. G. (2007). Incorporación de técnicas de visualización al sistema Weka. Ciencia de la Computación. Santa Clara, Universidad Central "Marta Abreu" de Las Villas.

Jacobson, I., G. Booch, et al. (2000). El proceso unificado de desarrollo de software. Madrid.

Larman, C. (1999). UML y Patrones. Introducción al análisis y diseño orientado a objetos. México.

NetBeans (2013). "Sitio Oficial." Retrieved 28 de abril del 2013, from http://www.NetBeans.org.

Oz, E. (2001). Administración de Sistemas de Información, Thomson Learning.

Stair, R. and G. Reynolds (2000). Principios de Sistemas de Información. México, Thomson.

Stallman, R. M. (2004). Software libre para una sociedad libre. Madrid, Traficantes de Sueños.

Wikipedia (2009). "Diagramas de Despliegue.". Retrieved 30 de abril del 2013, from http://es.wikipedia.org/wiki/DiagramasdeDespliegue