Universidad de las Ciencias Informáticas

Facultad 8

Título: Sistema de Gestión de Información de la Facultad 8. Módulo para la Gestión de la Residencia Estudiantil

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

AUTOR:

Fidel Alberto Curbelo Rosell

TUTOR:

Ing. Sandra Gutiérrez Secades

Ciudad de La Habana, julio de 2007

"Año del 49 Aniversario de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma con carácter exclusivo.

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

_____	_____
Firma del Autor	Firma del Tutor
Fidel Alberto Curbelo Rosell	Ing. Sandra Gutiérrez Secades

Pensamiento

"Al venir a la tierra, todo hombre tiene derecho a que se le eduque, y luego en pago, el deber de contribuir a la educación de los demás."

José Martí

Dedicatoria

A mi madre, por constituir este trabajo uno de sus más anhelados sueños, al igual que el resto de mi familia, a los cuales siempre tengo muy presentes, porque constituyen para mí, lo más preciado que posee todo ser humano en la vida.
A Fidel Castro, por haber sido el promotor e impulsor, junto a muchos otros, de este Proyecto Futuro, gracias a lo cual tengo la posibilidad hoy día de graduarme como Ingeniero Informático.
A la persona más especial que he conocido en toda mi vida, y que cada día me hace sentir muy feliz, por su amor, dedicación y entrega: Daylén.
A todos aquellos que siempre confiaron en mí y que me brindaron su apoyo constante en todo momento.

Agradecimientos

- A la **revolución** en primer lugar, por haberme dado la posibilidad de alcanzar uno de mis mayores sueños en la vida, el de formarme como profesional, lo cual no hubiese sido posible si no existiese la misma.
- A **toda mi familia**, por su apoyo y preocupación constante, especialmente a mi tío Roberto por haberme ayudado de la manera en que lo hizo siempre que lo necesité.
- A un **grupo de personas muy especiales** que constituyen para mi, mi segunda familia, por el cariño y el sin número de atenciones que me ofrecieron durante casi toda la carrera, ellos son: Maxorita, Dióscora, Miguelito, Camilo, Cori, Fina y demás familiares.
- A **todos los profesores** que tuve a lo largo de estos años, por todo lo bueno pude aprender de cada uno de ellos y por el apoyo que recibí de muchos de estos en circunstancias en las que lo necesité.
- A **todos mis amigos y compañeros de aula**, por la ayuda imprescindible que recibí de todos, entre los que se encuentran: Ariel, Sandy, Julio, Abduly, Arcel, Damir y muy en especial a Dayron por su constante disposición para ayudarme ante cualquier situación que se me presentara.
- A **Sandra** por confiar en que lo lograría, y por su ayuda en la etapa final de la tesis.

A todos, muchísimas gracias por haber contribuido a que yo obtuviese este logro, que también es de ustedes y que siempre confiaron en que lo alcanzaría.

RESUMEN

En el presente trabajo se pretende realizar el análisis, diseño e implementación del Módulo para la Gestión de la Residencia Estudiantil de la Facultad 8 de la Universidad de las Ciencias Informáticas (UCI), con el objetivo de automatizar varios de los procesos de esta índole que allí tienen lugar, como es el caso de: llevar un control sobre los edificios y apartamentos con los que cuenta la facultad y por ende, de los estudiantes que han sido ubicados en cada uno de estos, además de controlar los partes de la guardia estudiantil que se realizan a diario. Por lo cual, primero se analiza cómo se llevan a cabo varias de las actividades que se desarrollan en esta importante área de la universidad y luego se exponen las características del sistema que se propone, además de cómo esta construido el mismo, siguiendo para ello, los estándares de la metodología de desarrollo de software RUP, utilizada a lo largo de este trabajo. Una vez culminado el mismo, se obtendrá una versión del sistema capaz de mejorar la realización de una buena parte de los procesos que hoy día se llevan a cabo en esta Facultad en el área anteriormente mencionada, los cuales se realizan de una forma un tanto engorrosa, debido a la gran cantidad de información que se maneja, lo cual se implementa de forma manual.

Tabla de Contenidos

INTRODUCCIÓN 1

FUNDAMENTACIÓN TEÓRICA 5

1.1 INTRODUCCIÓN 5

1.2 SISTEMAS AUTOMATIZADOS SIMILARES EXISTENTES VINCULADOS AL CAMPO DE ACCIÓN 5

1.3 TENDENCIAS Y TECNOLOGÍAS ACTUALES 6

1.3.1 Lenguajes de Programación 6

1.3.2 Servidor Web Apache 12

1.3.3 Proceso de desarrollo de software 13

1.3.4 Metodologías de desarrollo de software 14

1.3.5 Herramientas CASE de Modelado con UML 18

1.3.6 NuSphere PHPEd 20

1.3.7 Smarty 22

1.3.8 Arquitectura 22

1.4 CONCLUSIONES 25

CARACTERÍSTICAS DEL SISTEMA 27

2.1 INTRODUCCIÓN 27

2.2 OBJETO DE ESTUDIO 27

2.3 PROCESOS OBJETO DE AUTOMATIZACIÓN 30

2.4 PROPUESTA DE SISTEMA 30

2.5 MODELO DEL NEGOCIO ACTUAL 30

2.6 REGLAS DEL NEGOCIO A CONSIDERAR 31

2.7 ACTORES DEL NEGOCIO 32

2.8 DIAGRAMA DE CASOS DE USO DEL NEGOCIO 32

2.9 TRABAJADORES DEL NEGOCIO 33

2.10 ESPECIFICACIÓN DE LOS CASOS DE USO DEL NEGOCIO 34

2.11 DIAGRAMA DE CLASES DEL MODELO DE OBJETOS 38

2.12 DEFINICIÓN DE LOS REQUISITOS FUNCIONALES 39

2.13 DEFINICIÓN DE LOS REQUISITOS NO FUNCIONALES 40

2.14 ACTORES DEL SISTEMA 42

2.15 DIAGRAMA DE CASOS DE USO DEL SISTEMA 43

2.15.1 Descripción de los casos de uso del sistema *43*
2.16 CONCLUSIONES 60
ANÁLISIS Y DISEÑO DEL SISTEMA **61**
3.1 INTRODUCCIÓN 61
3.2 DIAGRAMA DE CLASES DEL ANÁLISIS 61
3.3 DIAGRAMA DE CLASES DEL DISEÑO 64
3.4 DESCRIPCIÓN TEXTUAL DE LAS CLASES WEB 67
3.5 DIAGRAMA DE CLASES PERSISTENTES 79
3.6 PRINCIPIOS DE DISEÑO 80
3.6.1 Interfaz de Usuario *80*
3.6.2 Formato de salida de los Reportes *82*
3.6.3 Ayuda *83*
3.6.4 Tratamiento de errores *84*
3.6.5 Seguridad *85*
3.7 CONCLUSIONES 86
IMPLEMENTACIÓN Y PRUEBAS **87**
4.1 INTRODUCCIÓN 87
4.2 MODELO DE IMPLEMENTACIÓN 87
4.3 MODELO DE DESPLIEGUE 90
4.4 PRUEBA 91
4.5 CONCLUSIONES 99
CONCLUSIONES **100**
RECOMENDACIONES **101**
REFERENCIAS BIBLIOGRÁFICAS **102**
GLOSARIO DE TÉRMINOS Y SIGLAS **103**
ANEXOS **105**

INTRODUCCIÓN

En la actualidad, la cantidad de información que se maneja en cada uno de los centros de enseñanza existentes es bastante amplia, debido a que en todo momento se desea conocer una serie de datos referentes a todos aquellos involucrados en el proceso docente-educativo, ya sean estudiantes, profesores o trabajadores; así como de las actividades que cada uno de ellos realiza, por lo que en la mayoría de los casos, se hace un poco difícil tener toda esta información organizada y almacenada de forma adecuada, para posteriormente hacer un uso eficiente de la misma.

Como parte también, de todos los centros de enseñanzas se encuentran las universidades, en las cuales se maneja un gran cúmulo de información, no solo relacionada con aspectos docentes, sino también con aspectos tales como: **la residencia estudiantil**.

La Universidad de las Ciencias Informáticas (UCI), es uno de esos centros universitarios en los que la cantidad de información que se maneja en todo momento es muy grande, debido a que la misma se divide en 10 Facultades y cada una de estas a su vez, está compuesta por una gran cantidad de estudiantes de todos los años de la carrera, lo que trae consigo que la información que se necesite obtener de cada uno de estos y de las actividades que realizan a cada momento sea considerable, por lo que se requiere de que la misma esté lo más organizada posible para ser procesada de forma eficiente.

Dentro de estas 10 Facultades se encuentra, la Facultad 8, la cual no está exenta de ello y en la que hoy día, la gestión de toda la información, específicamente la relacionada con los procesos de ubicar estudiantes en la residencia estudiantil y controlar los parte de la guardia que se emiten diariamente, se hace un tanto engorrosa, lo que puede llevar a que se vea afectado en determinado momento la toma de decisiones, por no contar con toda la información necesaria o debidamente organizada para ello, por lo cual se han identificado los siguientes problemas:

- La información se manipula y organiza de forma manual, trayendo consigo que existan demoras a la hora de realizar una búsqueda de la misma, teniendo en cuenta que en muchas ocasiones, la información que se necesita aparece en más de un documento y

en más de un lugar de almacenamiento, por lo que si se desea hacer una recopilación de la información proveniente de diferentes fuentes, el proceso se torna un poco lento.

- Toda la información que se manipula en la facultad, relacionada con las cuestiones relativas a la ubicación de cada estudiante en la residencia estudiantil, además de los partes de la guardia que se emiten a diario, se almacena en muchos casos en papel, documentos Word y Excel, lo cual puede provocar con el paso del tiempo, pérdidas de información, ya sea por deterioro de la documentación que se manipula constantemente o por la aparición de imprevistos tales como: Virus, Problemas en el Sistema Operativo, entre otros.
- No existe una centralización de la información de manera que se pueda acceder a esta en todo momento y desde cualquier parte, lo que puede incidir notablemente en demoras a la hora de recopilar información proveniente de diversas personas, que para hacerle llegar la misma a la persona o las personas que la necesitan, tienen que trasladarse de un lugar a otro, o en el mejor de los casos, mandarla por correo electrónico, lo cual en definitiva, influye en que no se tenga la información en tiempo y con la calidad requerida.

De todo lo anteriormente expresado se deriva el **Objeto de Estudio** de este trabajo que son los procesos que se llevan a cabo en la residencia estudiantil de la Facultad 8 de la UCI.

Esto conlleva a que el **Campo de Acción** estará determinado por los procesos relativos a la ubicación de los estudiantes en la residencia estudiantil, además de los partes de la guardia estudiantil que se emiten a diario de dicha Facultad.

Por esta razón se plantea para este trabajo como **Hipótesis** que, si se realiza el análisis, diseño e implementación de un sistema que gestione la información que se maneja en la Facultad 8 de la UCI, con relación a la ubicación de los estudiantes en la residencia estudiantil y a los partes de la guardia que se emiten diariamente, se piensa que los procesos de esta índole que allí se llevan a cabo, se realizarán con mayor organización y rapidez.

Los **aportes prácticos** que se esperan obtener son:

- Integridad de la información relacionada con los procesos de la residencia estudiantil que tienen lugar en la Facultad.
- Mayor rapidez en la búsqueda de cualquier información, a partir de la automatización de varios de los procesos que se llevan a cabo en la Facultad en el área anteriormente

mencionada, con el objetivo de humanizar el trabajo de las personas involucradas en los mismos.

- Centralización de toda la información de esta índole, con el propósito de lograr una mayor accesibilidad a la misma.

Como **Objetivo General** de este trabajo se tiene: Análisis, Diseño e Implementación de una Aplicación Web que permita Gestionar la información relacionada con la ubicación de los estudiantes en la residencia estudiantil y los partes de la guardia emitidos a diario en la Facultad 8 de la UCI.

Los **Objetivos Específicos** que se persiguen son:

1. Informatizar los procesos que lleva a cabo la residencia estudiantil de la Facultad 8 de la UCI, concernientes a la ubicación de los estudiantes en los apartamentos y edificios que han sido asignados a la facultad, además de los partes de la guardia que se emiten a diario.
2. Proponer y desarrollar una versión del sistema, que de solución a gran parte de los problemas existentes.
3. Conformar la documentación del sistema con vista al desarrollo de posteriores versiones.

Para darle cumplimiento a los objetivos trazados se ha decidido desarrollar las siguientes **tareas**:

1. Entrevistar al Vicedecano de Extensión y Residencia para conocer todos los procesos que tienen lugar en esta área.
2. Realizar un estudio de otros sistemas similares existentes dentro y fuera de la universidad.
3. Realizar un estudio de las tendencias y tecnologías actuales y seleccionar las más apropiadas para la elaboración del sistema.
4. Modelar el negocio actual que tiene lugar en la residencia estudiantil de la Facultad.
5. Determinar y conformar los requisitos que debe cumplir el sistema.
6. Proponer una solución que contenga las funcionalidades principales con el objetivo de eliminar en gran medida, los problemas existentes.

Este trabajo está estructurado en 4 capítulos, en los cuales se tratan las siguientes cuestiones:

Capítulo 1. Fundamentación Teórica: Este capítulo trata acerca de algunos elementos teóricos que sirven de soporte para la realización de todo el trabajo en general, tales como: algunos sistemas similares existentes vinculados al campo de acción, además de una descripción y selección de las herramientas a utilizar.

Capítulo 2. Características del Sistema: En dicho capítulo se realiza una descripción del objeto de estudio, además de que se describen todos los procesos, actores, casos de usos y trabajadores del negocio actual. También se muestra el diagrama de casos de usos del negocio y el modelo de objetos del negocio. Por otro lado, se definen cuáles son los requerimientos funcionales y no funcionales, a la vez que se presentan los actores del sistema y diagrama de casos de uso del mismo, acompañado de la descripción textual de cada uno de estos.

Capítulo 3. Análisis y Diseño del Sistema: En este capítulo se presentan los diagramas de clases del análisis, lo cual influye notablemente a la hora de concebir el diseño del sistema, además de los diagramas de clases Web, que reflejan de una forma más clara cómo va a funcionar dicho sistema y qué clases están presentes en el mismo, acompañado también de la descripción de cada una de estas.

Capítulo 4. Implementación y Pruebas: En este capítulo se describe como está implementado el sistema, a través de los diagramas de componentes y el diagrama de despliegue, además de que se exponen y detallan las diferentes pruebas que se le realizan al mismo.

Al final se podrán observar las conclusiones generales, además de las recomendaciones, citas bibliográficas hechas a lo largo del trabajo, así como, un glosario de los diferentes términos y siglas utilizadas. Por último, se podrán observar los anexos, los cuales contienen información de apoyo a algunos aspectos tratados durante el trabajo.

Capítulo 1 Fundamentación Teórica

1.1 Introducción

El siguiente capítulo aborda la fundamentación teórica del Módulo para la Gestión de la Residencia Estudiantil de la Facultad 8 de la UCI, perteneciente al Sistema de Gestión de Información de la Facultad, en el cual se realiza un análisis de sistemas similares existentes, vinculados al campo de acción, además de que se presentan las tendencias y tecnologías actuales y se realiza una selección de aquellas que serán utilizadas durante el desarrollo del proyecto.

1.2 Sistemas automatizados similares existentes vinculados al campo de acción

En la actualidad se encuentra disponible para todas las facultades de la UCI un sistema de gestión de la información académica llamado Akademos, implementado sobre la plataforma .NET y del cual la Facultad 8 hace uso, al igual que el resto de las facultades, con el objetivo de obtener información relacionada con cuestiones académicas de gran utilidad, pero dicho sistema no brinda la posibilidad de gestionar información relativa a los procesos no docentes que se llevan a cabo en las facultades, tales como: Ubicar a los estudiantes en la residencia estudiantil, tener un control sobre los apartamentos y edificios que le fueron asignados a la Facultad, conocer en qué edificio, paso de escalera y apartamento de la residencia se encuentra ubicado un estudiante de la facultad o un grupo de estudiantes, conocer cómo se ha comportado la guardia estudiantil de una brigada específica hasta el momento, a través de los partes emitidos de la misma, o consultar los partes de la guardia estudiantil de cualquier día.

En la UCI no se cuenta con un sistema que facilite la gestión de la información no docente de cada una de las facultades y que brinde la posibilidad de contar con algunas de las funcionalidades anteriormente mencionadas. En algunas facultades se ha pensado en realizar sistemas que ofrezcan la posibilidad de gestionar alguna que otra información relativa a la residencia estudiantil, como es el caso de la Facultad 10, en la cual se está trabajando en la realización de un sistema sobre la plataforma Linux que brinde la posibilidad de gestionar información docente que Akademos no facilita, además de que, se está pensando en un futuro agregarle funcionalidades relativas a los procesos no docentes que se llevan a cabo dentro de dicha Facultad, pero hasta el momento solo se está trabajando en los aspectos relacionados

con la parte docente. Por otro lado, tampoco se cuenta con un sistema de este tipo en muchas de las universidades del país.

1.3 Tendencias y tecnologías actuales

A continuación se describen algunas de las tendencias y tecnologías actuales posibles a utilizar para darle solución a los problemas planteados anteriormente de manera eficiente, teniendo en cuenta las necesidades existentes y el entorno donde se aplicará el sistema qué se va a construir.

1.3.1 Lenguajes de Programación

Un lenguaje de programación es un medio que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente.

Un lenguaje de programación permite a un programador especificar de *manera precisa*: sobre qué datos una computadora debe operar, cómo deben ser estos almacenados y transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. Todo esto, a través de un lenguaje que intenta estar *relativamente* próximo al lenguaje humano o natural, tal como sucede con el lenguaje léxico. [1]

PHP: Es un lenguaje de programación usado generalmente para la creación de contenido para sitios Web. PHP es un acrónimo recurrente que significa "PHP Hypertext Pre-processor" (inicialmente PHP Tools, o, Personal Home Page Tools), y se trata de un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios Web. [2]

El fácil uso y la similitud con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores experimentados crear aplicaciones complejas con una curva de aprendizaje muy suave. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones y prácticas. [3]

Su interpretación y ejecución se da en el servidor Web, en el cual se encuentra almacenado el script, y el cliente sólo recibe el resultado de la ejecución. Cuando el cliente hace una petición al servidor para que le envíe una página Web, generada por un script PHP, el servidor ejecuta el intérprete de PHP, el cual procesa el script solicitado que generará el contenido de manera

dinámica, pudiendo modificar el contenido a enviar, y regresa el resultado al servidor, el cual se encarga de regresárselo al cliente. Además es posible utilizar PHP para generar archivos PDF, Flash, así como imágenes en diferentes formatos, entre otras cosas. [3]

Permite la conexión a diferentes tipos de servidores de base de datos tales como: MySQL, Postgre, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird, SQLite; lo cual facilita la creación de aplicaciones Web muy robustas. [3]

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos tales como UNIX (y de ese tipo, como Linux), Windows y Mac, y puede interactuar con los servidores de Web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI. [3]

Los principales usos del PHP son los siguientes:

- Programación de páginas Web dinámicas, habitualmente en combinación con el motor de base datos MySQL, aunque cuenta con soporte nativo para otros motores, incluyendo el estándar ODBC, lo que amplía en gran medida sus posibilidades de conexión.
- Programación en consola, al estilo de Perl o Shell scripting.
- Creación de aplicaciones gráficas independientes del navegador, por medio de la combinación de PHP y GTK (GIMP Tool Kit), lo que permite desarrollar aplicaciones de escritorio en los sistemas operativos en los que está soportado. [4]

Ventajas de PHP

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL.
- Leer y manipular datos desde diversas fuentes, incluyendo datos que pueden ingresar los usuarios desde formularios HTML.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.
- Permite crear los formularios para la Web.

- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables ni manejo detallado del bajo nivel. [4]

Otros lenguajes como Perl (Practical Extraction and Report Language), ASP (Active Server Pages) y JSP (Java Server Pages) tienen características similares al PHP aunque poseen rasgos que los marcan y por ello los distingue, entre ellos se pueden encontrar:

- **Características multiplataformas:** Menos el ASP, que es solamente soportado por la plataforma Windows, los demás lenguajes están soportados en múltiples plataformas.
- **Velocidad de ejecución:** La velocidad es mayor en PHP, seguidos por PERL y JSP.
- **Disponibilidad de recursos:** actualmente los más utilizados en la Internet son el PHP y el JSP, siendo más utilizado en la publicación de artículos y códigos de ejemplos. PHP tiene una de las comunidades más grandes en Internet, al igual que la de Java.
- **Familiaridad con el lenguaje:** En las universidades los lenguajes más utilizados por los programadores es el ASP y el PHP. [4]

Java: Supone un significativo avance en el mundo de los entornos software, y esto viene avalado por tres elementos claves que diferencian a este lenguaje desde un punto de vista tecnológico:

- Es un lenguaje de programación que ofrece la potencia del diseño orientado a objetos con una sintaxis fácilmente accesible y un entorno robusto y agradable.
- Proporciona un conjunto de clases potentes y flexibles.
- Pone al alcance de cualquiera la utilización de aplicaciones que se pueden incluir directamente en páginas Web (aplicaciones denominadas *applets*).

Java aporta a la Web una interactividad que se había buscado durante mucho tiempo entre usuario y aplicación. [3]
Las principales características de Java son descritas a continuación:

Potente

Orientado a Objetos: En este aspecto Java fue diseñado partiendo de cero, no siendo derivado de otro lenguaje anterior y no tiene compatibilidad con ninguno de ellos.
En Java el concepto de objeto resulta sencillo y fácil de ampliar. Además se conservan elementos "no objetos", como números, caracteres y otros tipos de datos simples.

Riqueza Semántica: Pese a su simpleza se ha conseguido un considerable potencial, y aunque cada tarea se puede realizar de un número reducido de formas, se ha conseguido un gran potencial de expresión e innovación desde el punto de vista del programador.

Robusto: Java verifica su código al mismo tiempo que lo escribe, y una vez más antes de ejecutarse, de manera que se consigue un alto margen de codificación sin errores. Se realiza un descubrimiento de la mayor parte de los errores durante el tiempo de compilación, ya que Java es estricto en cuanto a tipos y declaraciones, y así lo que es rigidez y falta de flexibilidad se convierte en eficacia. Respecto a la gestión de memoria, Java libera al programador del compromiso de tener que controlar especialmente la asignación que de ésta hace a sus necesidades específicas. Este lenguaje posee una gestión avanzada de memoria llamada gestión de basura, y un manejo de excepciones orientado a objetos integrados. Estos elementos realizarán muchas tareas, antes tediosas a la vez que obligadas para el programador.

Modelo de objeto rico: Existen varias clases que contienen las abstracciones básicas para facilitar a los programas una gran capacidad de representación. Para ello se contará con un conjunto de clases comunes que pueden crecer para admitir todas las necesidades del programador.

Además la biblioteca de clases de Java proporciona un conjunto único de protocolos de Internet.

El conjunto de clases más complicado de Java son sus paquetes gráficos AWT (*Abstract Window Toolkit*) y *Swing*. Estos paquetes implementan componentes de una interfaz de usuario gráfica básica común a todos los ordenadores personales modernos. [3]

Simple

Fácil Aprendizaje: El único requerimiento para aprender Java es tener una comprensión de los conceptos básicos de la programación orientada a objetos. Así se ha creado un lenguaje simple (aunque eficaz y expresivo) pudiendo mostrarse cualquier planteamiento por parte del programador sin que las interioridades del sistema subyacente sean desveladas.

Java es más complejo que un lenguaje simple, pero más sencillo que cualquier otro entorno de programación. El único obstáculo que se puede presentar es conseguir comprender la programación orientada a objetos, aspecto que, al ser independiente del lenguaje, se presenta como insalvable.

Completado con utilidades: El paquete de utilidades de Java viene con un conjunto completo de estructuras de datos complejas y sus métodos asociados, que serán de inestimable ayuda para implementar *applets* y otras aplicaciones más complejas. Se dispone también de estructuras de datos habituales, como *pilas y tablas hash*, como clases ya implementadas. [3]

Interactivo y orientado a red

Interactivo y animado: Uno de los requisitos de Java desde sus inicios fue la posibilidad de crear programas en red interactivos, por lo que es capaz de hacer varias cosas a la vez sin perder rastro de lo que debería suceder y cuándo. Se da soporte a la utilización de múltiples hilos de programación (*multithread*).

Las aplicaciones de Java permiten situar figuras animadas en las páginas Web, y éstas pueden concebirse con logotipos animados o con texto que se desplace por la pantalla. También pueden tratarse gráficos generados por algún proceso. Estas animaciones pueden ser interactivas, permitiendo al usuario un control sobre su apariencia.

Arquitectura neutral: Java está diseñado para que un programa escrito en este lenguaje sea ejecutado correctamente independientemente de la plataforma en la que se esté actuando (Macintosh, PC, UNIX…). Para conseguir esto utiliza una compilación en una representación intermedia que recibe el nombre de *códigos de byte*, que pueden interpretarse en cualquier sistema operativo con un intérprete de Java. La desventaja de un sistema de este tipo es el rendimiento; sin embargo, el hecho de que Java fuese diseñado para funcionar razonablemente bien en microprocesadores de escasa potencia, unido a la sencillez de traducción a código máquina hacen que Java supere esa desventaja sin problemas.

Applets: Una *applet* (mini aplicación) es un pequeño programa en Java transferido dinámicamente a través de Internet. Presentan un comportamiento inteligente, pudiendo reaccionar a la entrada de un usuario y cambiar de forma dinámica. Sin embargo, la verdadera novedad es el gran potencial que Java proporciona en este aspecto, haciendo posible que los programadores ejerzan un control sobre los programas ejecutables de Java que no es posible encontrar en otros lenguajes. [3]

Otros

Seguridad: Existe una preocupación lógica en Internet por el tema de la seguridad: virus, caballos de Troya, y programas similares que navegan de forma usual por la red, constituyendo una amenaza palpable. Java ha sido diseñado poniendo un énfasis especial en el tema de la

seguridad, y se ha conseguido lograr cierta inmunidad en el aspecto de que un programa realizado en Java no puede realizar llamadas a funciones globales ni acceder a recursos arbitrarios del sistema, por lo que el control sobre los programas ejecutables no es equiparable a otros lenguajes. [3]

Los niveles de seguridad que presenta son:

- Fuertes restricciones al acceso a memoria, como son la eliminación de punteros aritméticos y de operadores ilegales de transmisión.
- Rutina de verificación de los *códigos de byte* que asegura que no se viole ninguna construcción del lenguaje.
- Verificación del nombre de clase y de restricciones de acceso durante la carga.
- Sistema de seguridad de la interfaz que refuerza las medidas de seguridad en muchos niveles.

Lenguaje basado en C++: Java fue desarrollado basándose en C++, pero eliminando rasgos del mismo poco empleados, optándose por una codificación comprensible. Básicamente, encontramos las siguientes diferencias con C++:

- Java no soporta los tipos *struct, union* ni punteros*.*
- No soporta *typedef* ni *#define.*
- Se distingue por su forma de manejar ciertos operadores y no permite una sobrecarga de operadores.
- No soporta herencia múltiple.
- Java maneja argumentos en la línea de comandos de forma diversa a como lo hacen C o C++.
- Tiene una clase *String* que es parte del paquete *java.lang* y se diferencia de la matriz de caracteres terminada con un nulo que usan C y C++.
- Java cuenta con un sistema automático para asignar y liberar memoria, con lo que no es necesario utilizar las funciones previstas con este fin en C y C++.

Gestión de la entrada/salida: En lugar de utilizar primitivas como las de C para trabajar con ficheros, se utilizan primitivas similares a las de C++, mucho más elegantes, que permiten tratar los ficheros, sockets, teclado y monitor como flujos de datos. [3]

De este modo se pueden utilizar dichas primitivas para cualquier operación de Entrada/Salida.

Diferentes tipos de aplicaciones: En Java podemos crear los siguientes tipos de aplicaciones:

- Aplicaciones: Se ejecutan sin necesidad de un navegador.
- *Applets*: Se pueden descargar de Internet y se observan en un navegador.
- *JavaBeans*: Componentes software Java, que se puedan incorporar gráficamente a otros componentes.
- *JavaScript*: Conjunto del lenguaje Java que puede codificarse directamente sobre cualquier documento HTML
- *Servlets:* Módulos que permiten sustituir o utilizar el lenguaje Java en lugar de programas CGI (Common Gateway Interface) a la hora de dotar de interactividad a las páginas Web.

1.3.2 Servidor Web Apache

Un servidor de páginas Web es un programa que permite acceder a páginas Web alojadas en un ordenador. Hoy en día Apache es el servidor Web más utilizado del mundo, encontrándose muy por encima de sus competidores, tanto gratuitos como comerciales. Es un software de código abierto que funciona sobre cualquier plataforma. Desde su origen ha evolucionado hasta convertirse en uno de los mejores servidores en términos de eficiencia, funcionalidad y velocidad, surgió en abril de 1996 y ya en julio del 2002 era utilizado por el 57% de los sitios Web de Internet. [4]

Tiene capacidad para servir páginas tanto de contenido estático, para lo que nos serviría sencillamente un viejo ordenador 486, como de contenido dinámico a través de otras herramientas soportadas que facilitan la actualización de los contenidos mediante bases de datos, ficheros u otras fuentes de información, además de que es muy potente y altamente configurable. [4]

Los servidores Web suministran páginas Web a los navegadores que lo solicitan. En términos más técnicos, los servidores Web soportan el Protocolo de Transferencia de Hipertexto como HTTP (HyperText Transfer Protocol), el estándar de Internet para comunicaciones Web. Usando HTTP, un servidor Web envía páginas Web en HTML y Common Gateway Interface (CGI), así como otros tipos de scritps a los navegadores o browsers cuando éstos los requieren. Cuando un usuario hace clic sobre un enlace a una página Web, se envía una solicitud al servidor Web para localizar los datos nombrados por ese enlace. El servidor Web recibe esta solicitud y suministra los datos que le han sido solicitados o bien devuelve un mensaje de error. [4]

El servidor Apache es un software que está estructurado en módulos, es decir, está dividido en muchas porciones de código que hacen referencia a diferentes aspectos o funcionalidades del servidor Web. Esta modularidad es intencionada ya que la configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo. Los módulos del Apache se pueden clasificar en tres categorías: [4]

- Módulos Base: Módulo con las funciones básicas del Apache.
- Módulos Multiproceso: Son los responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos a atender a las peticiones.
- Módulos Adicionales: Cualquier otro módulo que le añada una funcionalidad al servidor.

Las funcionalidades más elementales se encuentran en el módulo base, siendo necesario un módulo multiproceso para manejar las peticiones. Se han diseñado varios módulos multiprocesos para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código. [4]

El resto de funcionalidades del servidor se consigue por medio de módulos adicionales que se pueden cargar. Para añadir un conjunto de utilidades al servidor, simplemente hay que añadirle un módulo, de forma que no es necesario volver a instalar el software. [4]

1.3.3 Proceso de desarrollo de software

Con el de cursar de los tiempos la producción de software ha ido creciendo en cuanto a cantidad, calidad y complejidad, debido a que cada vez son mayores las exigencias de los usuarios, por lo que cada vez se hace mayor la necesidad de un proceso que integre todas las fases del desarrollo del software y que sea perfectamente entendible por todos aquellos que participan en su elaboración.

Lenguaje Unificado de Modelado (UML)

UML (Unified Modeling Language) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema de software. [5]

Sus creadores pretendieron con este lenguaje, unificar las experiencias acumuladas sobre técnicas de modelado e incorporar las mejores prácticas en un acercamiento estándar.

El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas y proporciona un estándar que permite al analista de sistemas generar un anteproyecto de varias facetas que sean comprensibles para los clientes, desarrolladores y

todos aquellos que estén involucrados en el proceso de desarrollo. Un modelo UML indica que es lo que supuestamente hará el sistema pero no como lo hará. [5]

El UML permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma convencional y fácil de comprender para comunicarlas a otras personas que estén involucradas en el proceso de desarrollo de los sistemas, esto se lleva a cabo mediante un conjunto de símbolos y diagramas. [6]

Existen varias herramientas CASE (Computer-Aided Systems Engineering), que dan asistencia a analistas, ingenieros de software y desarrolladores durante el ciclo de vida de desarrollo de un software, pero es Rational Rose líder en el modelado del desarrollo de los proyectos. La herramienta fue desarrollada por los creadores de UML, utilizando la notación estándar en la arquitectura de software. Esta herramienta integra todos los elementos que propone la metodología RUP para cubrir el ciclo de vida de un proyecto. [4]

1.3.4 Metodologías de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y pasos a seguir para construir un software. Por otra parte, una metodología de desarrollo de software define quién debe hacer qué, cuándo y cómo para alcanzar un determinado objetivo. También podemos decir que una metodología es un proceso, y en su modelación se definen como elementos principales los siguientes:

- **Trabajadores (quién):** Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.
- **Actividades (cómo):** Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.
- **Artefactos (qué):** Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.
- **Flujo de actividades (cuándo):** Secuencia de actividades realizadas por trabajadores y que producen un resultado de valor observable.

Rational Unified Process (RUP)

La metodología RUP, llamada así por sus siglas en inglés (Rational Unified Process), divide en 4 fases el desarrollo del software (*Ver Anexo 1*):

- Inicio (El objetivo de esta etapa es determinar la visión del proyecto)
- Elaboración (En esta etapa el objetivo es determinar la arquitectura)
- Construcción (En esta etapa el objetivo es llegar a alcanzar una funcionalidad operativa)
- Transición (El objetivo de esta etapa es llegar a obtener el release del producto)

En cada una de estas fases tienen lugar iteraciones de varios flujos de trabajos (*Ver Anexo 1*), que son:

- **Modelo del negocio:** Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
- **Requerimientos:** Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
- **Análisis y Diseño:** Describe cómo el sistema será construido a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo qué se debe programar.
- **Implementación:** Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
- **Prueba (Testeo):** Busca los defectos a los largo del ciclo de vida.
- **Instalación o despliegue:** Produce release del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.
- **Administración del proyecto:** Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
- **Administración de configuración y cambios:** Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.
- **Ambiente:** Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

Características de RUP

- Creado por Jacobson, Rumbaugh y Booch.
- Unifica los mejores elementos de metodologías anteriores.

- Preparado para desarrollar grandes y complejos proyectos.
- Orientado a Objetos.
- Utiliza el UML como lenguaje de representación visual.

El ciclo de vida de RUP se caracteriza por:

1. **Dirigido por casos de uso:** Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso (cómo se llevan a cabo).
2. **Centrado en la arquitectura:** La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. RUP se desarrolla mediante iteraciones, comenzando por los CU relevantes desde el punto de vista de la arquitectura. El modelo de arquitectura se representa a través de vistas en las que se incluyen los diagramas de UML (Vista de Casos de Uso, Vista Lógica, Vista de Procesos, Vista de Implementación, Vista de Despliegue).
3. **Iterativo e Incremental:** Aunque el *Anexo 1* puede sugerir que los flujos de trabajo se desarrollan en cascada, la “lectura” de este gráfico tiene que ser vertical y horizontal. RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Por ejemplo, una iteración de elaboración centra su atención en el análisis y diseño, aunque refina los requerimientos y obtiene un producto con un determinado nivel, pero que irá creciendo incrementalmente en cada iteración.

Extreme Programing (XP): Es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizada para proyectos de corto plazo y equipo de desarrollo pequeño.

La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. [7]

Esta metodología de desarrollo de software se basa en: [7]

- **Pruebas Unitarias:** Consiste en las pruebas realizadas a los principales procesos, de tal manera que adelantándose en algo hacia el futuro, puede hacer pruebas de las fallas que pudieran ocurrir. Es como si se adelantara a obtener los posibles errores.
- **Refabricación:** Se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- **Programación en pares:** Una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Es como el chofer y el copiloto: mientras uno conduce, el otro consulta el mapa. [7]

¿Qué es lo que propone XP?

- Empieza en pequeño y añade funcionalidad con retroalimentación continua.
- El manejo del cambio se convierte en parte sustancial del proceso.
- El costo del cambio no depende de la fase o etapa.
- No introduce funcionalidades antes que sean necesarias.
- El cliente o el usuario se convierte en miembro del equipo. [7]

Derechos del Cliente

- Decidir que se implementa.
- Saber el estado real y el progreso del proyecto.
- Añadir, cambiar o quitar requerimientos en cualquier momento.
- Obtener lo máximo de cada semana de trabajo.
- Obtener un sistema funcionando cada 3 o 4 meses. [7]

Derechos del Desarrollador

- Decidir cómo se implementan los procesos.
- Crear el sistema con la mejor calidad posible.
- Pedir al cliente en cualquier momento aclaraciones de los requerimientos.
- Estimar el esfuerzo para implementar el sistema.
- Cambiar los requerimientos en base a nuevos descubrimientos. [7]

Lo fundamental en este tipo de metodología es:

- La comunicación, entre los usuarios y los desarrolladores.
- La simplicidad, al desarrollar y codificar los módulos del sistema.

- La retroalimentación, concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales. [7]

1.3.5 Herramientas CASE de Modelado con UML

A medida que los sistemas que hoy se construyen se tornan más y más complejos, las herramientas de modelado con UML ofrecen muchos beneficios para todos los involucrados en un proyecto, por ejemplo, administrador del proyecto, analistas, arquitectos, desarrolladores y otros. Las herramientas CASE (***C**omputer **A**ided **S**oftware **E**ngineering*, Ingeniería de Software Asistida por Ordenador) de modelado con UML nos permiten aplicar la metodología de análisis y diseño orientados a objetos y abstraernos del código fuente, en un nivel donde la arquitectura y el diseño se tornan más obvios y más fáciles de entender y modificar. Cuanto más grande es un proyecto, es más importante utilizar una herramienta CASE. Por otro lado, al usar las herramientas CASE:

- Los Analistas de Negocio/ Sistemas pueden capturar los requisitos del negocio/sistema con un modelo de casos de uso.
- Los Diseñadores/Arquitectos pueden producir el modelo de diseño para articular la interacción entre los objetos o los subsistemas de la misma o de diferentes capas (los diagramas UML típicos que se crean son los de clases y los de interacción).
- Los Desarrolladores pueden transformar rápidamente los modelos en una aplicación funcionando, y buscar un subconjunto de clases y métodos y asimilar el entendimiento de cómo lograr interfaces con ellos. [8]

El modelo actúa como el plano y guiará finalmente la construcción del sistema. De manera semejante, la administración es capaz de ver, puntualmente y desde un alto nivel, una representación del diseño y comprender lo que está sucediendo.

Por estas razones, las herramientas CASE de UML acompañadas con metodologías, nos brindan una forma de representar sistemas demasiados complejos para comprenderlos a través de su código fuente subyacente y nos permiten desarrollar la solución de software correcta, más rápido y más económicamente.

Sin embargo, las herramientas CASE varían con respecto a las capacidades de modelado con UML, el soporte del ciclo de vida del proyecto, las ingenierías directa y reversa, el modelado de datos, la performance, el precio, el soporte, la facilidad de uso, etc. [8]

Rational Rose

Rational Rose es la herramienta CASE desarrollada por los creadores de UML (Booch, Rumbaugh y Jacobson), que cubre todo el ciclo de vida de un proyecto: Concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables.

El navegador UML de Rational Rose nos permite establecer una trazabilidad real entre el modelo (análisis y diseño) y el código ejecutable.

Facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información (vista de Casos de Uso, vista Lógica, vista de Componentes y vista de Despliegue), pero utilizan un lenguaje común para comprender y comunicar la estructura y la funcionalidad del sistema en construcción. [9]

Esta herramienta propone la utilización de cuatro tipos de modelos para realizar el diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas, creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software.

Rational Rose utiliza un proceso de desarrollo iterativo controlado (Controlled Iterative Process Development), donde el desarrollo se lleva a cabo en una secuencia de iteraciones. Cada iteración comienza con una primera aproximación del análisis, diseño e implementación para identificar los riesgos del diseño, los cuales se utilizan para conducir la iteración, primero se identifican los riesgos y después se prueba la aplicación para que estos se hagan mínimos.

Cuando la implementación pasa todas las pruebas que se determinan en el proceso, esta se revisa y se añaden los elementos modificados al modelo de análisis y diseño. Una vez que la actualización del modelo se ha modificado, se realiza la siguiente iteración.

Rational Rose permite que hayan varias personas trabajando a la vez en el proceso iterativo controlado, para ello posibilita que cada desarrollador opere en un espacio de trabajo privado que contiene el modelo completo y tenga un control exclusivo sobre la propagación de los cambios en ese espacio de trabajo.

También es posible descomponer el modelo en unidades controladas e integrarlas con un sistema para realizar el control de proyectos que permite mantener la integridad de dichas unidades. [9]

Enterprise Architect (EA)

Enterprise Architect es una herramienta comprensible de diseño y análisis UML, cubriendo el desarrollo de software desde el paso de los requerimientos a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento. EA es una herramienta multi-usuario, basada en Windows, diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad. [10]

Algunas de las principales características de EA son las siguientes: [10]

Velocidad, estabilidad y rendimiento: El Lenguaje Unificado de Modelado (UML) provee beneficios significativos para ayudar a construir modelos de sistemas de software rigurosos y donde es posible mantener la trazabilidad de manera consistente. Enterprise Architect soporta este proceso en un ambiente fácil de usar, rápido y flexible.

Trazabilidad de extremo a extremo: Enterprise Architect provee trazabilidad completa desde el análisis de requerimientos hasta los artefactos de análisis y diseño, a través de la implementación y el despliegue. Combinados con la ubicación de recursos y tareas incorporados, los equipos de administradores de proyectos y calidad están equipados con la información que ellos necesitan para ayudarles a entregar proyectos en tiempo.

Construido sobre UML 2.1: Las bases de Enterprise Architect están construidas sobre la especificación de UML 2.0 - pero no se detiene ahí. Usa perfiles UML para extender el dominio de modelado, mientras que la validación del modelo asegura integridad.

Por otra parte EA soporta generación e ingeniería inversa de código fuente para muchos lenguajes populares, incluyendo C++, C#, Java, Delphi, VB.Net, Visual Basic y PHP. Con un editor de código fuente con "resaltador de sintaxis" incorporado, EA le permite navegar y explorar su modelo de código fuente en el mismo ambiente. Las plantillas de generación de código le permiten personalizar el código fuente generado a las especificaciones de su compañía. [10]

1.3.6 NuSphere PHPEd

PHPEd es un editor para programadores con soporte para múltiples formatos, similar a otras aplicaciones como PHP Coder.

PHPEd facilita el trabajo de programación con numerosas características de gran utilidad entre las que destacan: [11]

- Completo sistema de ayuda

- Plantillas de documentos y de fragmentos de código frecuentes.
- Código de colores para comandos en PHP, Perl, Javascript, SQL, HTML y más.

Además, esta herramienta incluye un cliente de FTP y un servidor Web integrados, totalmente configurables según tus necesidades de trabajo.

Zend Studio

Se trata de un programa de la casa Zend, impulsores de la tecnología de servidor PHP, orientado a desarrollar aplicaciones Web, en lenguaje PHP. El programa, además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código. [12]

El programa entero está escrito en Java, lo que a veces supone que no funcione tan rápido como otras aplicaciones de uso diario. Sin embargo, esto ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para Windows, Linux y MacOS, aunque el desarrollo de las versiones de este último sistema se retrase un poco más.

Zend Studio consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Las dos partes se instalan por separado, la del cliente contiene el interfaz de edición y la ayuda. Permite además hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración habrá que disponer de la parte del servidor, que instala Apache y el módulo PHP o, en caso de que estén instalados, los configura para trabajar juntos en depuración. [12]

Lo más destacable de este editor es que contiene una ayuda contextual con todas las librerías de funciones del lenguaje que asiste en todo momento ofreciendo nombres de las funciones y parámetros que deben recibir. Aunque esta ayuda contextual no solo se queda en las funciones definidas en el lenguaje, sino que también reporta ayudas con las funciones que vayamos creando nosotros, incluso en páginas que tengamos incluidas con la función include(). [12]

También dispone de herramientas para gestionar los proyectos, muy útiles para mejorar la productividad en la programación. Los proyectos permiten guardar mucha más información al programa sobre los archivos, discos, servidores, etc. que se gestionen en estas aplicaciones PHP. [12]

Sin duda, más de una vez los programadores de PHP se han visto en un duro problema por no encontrar un error en algún script que está dando resultados inesperados.

Zend Studio dispone de una herramienta muy interesante de debug o depuración. Gracias a ella se pueden ejecutar páginas y conocer en todo momento el contenido de las variables de la aplicación y las variables del entorno, como las cookies; las recibidas por formulario o en la sesión. Se pueden colocar puntos de parada de los scripts y realizar las acciones típicas de depuración.

Además de la ventana para visualizar el contenido de las variables, dispone de otras donde muestra la salida del script según se va generando, y otra donde se pueden ver las alertas y errores. Las posibilidades se completan con distintos tipos de depuración, en local, en remoto o a partir de una URL. [12]

1.3.7 Smarty

Smarty es un motor de plantillas para PHP, cuyo objetivo es separar el contenido de la presentación en una página Web, se encuentra bajo la licencia GPL por lo que puede ser usado libremente. [13]

Es común que en grandes proyectos el rol de diseñador gráfico y el de programador sean cubiertos por personas distintas, sin embargo la programación en PHP tiene la tendencia a combinar estas dos labores en una persona y dentro del mismo código lo que trae consigo grandes dificultades a la hora de cambiar alguna parte del diseño de la página, pues se tiene que escarbar entre los scripts para modificar la presentación del contenido, Smarty tiene como objetivo solucionar este problema. [13]

Existen más sistemas de plantillas para PHP pero éste parece ser la más avanzada y con más frecuencia de desarrollo. También hay detractores de estas técnicas que alegan que las mismas hacen en cierta medida un grado más complejo el desarrollo Web, por la necesidad de aprender un seudo lenguaje más. [13]

1.3.8 Arquitectura

"Una Arquitectura Software, también denominada *Arquitectura lógica*, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información. [4]

Se selecciona y diseña con base en unos objetivos y restricciones. Los objetivos son aquellos prefijados para el sistema de información, pero no solamente los de tipo funcional, también otros objetivos como la mantenibilidad, auditabilidad, flexibilidad e interacción con otros sistemas de información. Las restricciones son aquellas limitaciones derivadas de las tecnologías disponibles para implementar sistemas de información. Unas arquitecturas son más

recomendables de implementar con ciertas tecnologías mientras que otras tecnologías no son aptas para determinadas arquitecturas. Por ejemplo, no es viable emplear una arquitectura software de tres capas para implementar sistemas en tiempo real. [4]

La arquitectura software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación ente ellos. Toda arquitectura software debe ser implementable en una *arquitectura física*, que consiste simplemente en determinar qué computadora tendrá asignada cada tarea de computación. [4]

La arquitectura de software, tiene que ver con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requerimientos no funcionales, como la confiabilidad, escalabilidad, portabilidad, y disponibilidad. —Kruchten, Philippe" [4]

Generalmente, no es necesario inventar una nueva arquitectura software para cada sistema de información. Lo habitual es adoptar una arquitectura conocida en función de sus ventajas e inconvenientes para cada caso en concreto. Así, las arquitecturas más universales son: [4]

Monolítica. Donde el software se estructura en grupos funcionales muy acoplados.

Cliente-servidor. Donde el software reparte su carga de cómputo en dos partes independientes pero sin reparto claro de funciones.

Arquitectura de tres niveles. Generalización de la arquitectura cliente-servidor donde la carga se divide en tres partes *(ver Anexo 2)* con un reparto claro de funciones: una capa para la presentación, otra para el cálculo y otra para el almacenamiento. Una capa solamente tiene relación con la siguiente.

Modelo Vista Controlador (MVC): 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 MVC se ve frecuentemente en aplicaciones Web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página.

- **Modelo**: Esta es la representación específica del dominio de la información sobre la cual funciona la aplicación. El modelo es otra forma de llamar a la capa de dominio. La lógica de dominio añade significado a los datos; por ejemplo, calculando si hoy es el cumpleaños del usuario o los totales impuestos en un carrito de compra.
- **Vista**: Este presenta el modelo en un formato adecuado para interactuar, usualmente un elemento de interfaz de usuario.

- **Controlador**: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.
- Muchas aplicaciones utilizan un mecanismo de almacenamiento persistente (como puede ser una base de datos) para almacenar los datos. MVC no menciona específicamente esta capa de acceso a datos.

Es común pensar que una aplicación tiene tres capas principales: presentación (IU), dominio, y acceso a datos. En MVC, la capa de presentación está partida en controlador y vista. La principal separación es entre presentación y dominio; la separación entre vista y controlador es menos clara.

Arquitectura n-Capas. La que más comúnmente se usa es la de cuatro capas, la capa que se agrega es la que surge de separar definitivamente las reglas de negocio de la de Acceso a Datos. Esta arquitectura trae consigo la ventaja de aislar definitivamente la lógica de negocios de todo lo que tenga que ver con el origen de datos, ya que desde el manejo de la conexión, hasta la ejecución de una consulta, la manejará la capa de Acceso a Datos. De este modo, ante cualquier eventual cambio, solo se deberá tocar un módulo específico, así como al momento de plantear la escalabilidad de este sistema, si se han respetado las reglas básicas de diseño, entonces no se deberán afrontar grandes modificaciones. [4]

*Abstracción total acerca del origen de datos***.** Las distintas capas se especializan absolutamente en la funcionalidad que deben brindar (procesamiento en las reglas de negocios o presentación de datos en la capa cliente) sin importar cual es el origen de los datos procesados.

Bajo costo de desarrollo y mantenimiento de las aplicaciones. Si bien al momento del diseño podemos observar una mayor carga de complejidad, la utilización de esta arquitectura brinda un control más cercano de cada componente, así como también la posibilidad de una verdadera reutilización del código.

Estandarización de las reglas de negocio. Las reglas de negocio se encuentran encapsuladas en un set de rutinas comunes y pueden ser llamadas desde diversas aplicaciones sin necesidad de saber cómo esta funciona o ha sido diseñada.

Mejor calidad en las aplicaciones. Como las aplicaciones son construidas en unidades separadas, estas pueden ser testeadas independientemente y con mucho más detalle, esto conduce a obtener un producto mucho más sólido.

Reutilización de código. La concepción natural de un sistema desarrollado con esta arquitectura, promueve la reutilización de sus componentes en varias partes del propio desarrollo y de futuros sistemas.

1.4 Conclusiones

Teniendo en cuenta las necesidades que posee la facultad 8 de la UCI, a partir de los problemas existentes en la ejecución de los procesos que tienen lugar en la residencia estudiantil, y habiendo hecho un previo análisis de las principales características de las tecnologías y herramientas más usadas en el mundo en la actualidad, se decidió seleccionar, para la construcción y desarrollo del sistema propuesto, las siguientes herramientas y tecnologías: Como lenguaje de programación para implementar el sistema, el PHP, debido a que, a pesar de poseer muchas ventajas al igual que Java, este lenguaje posee algunas característica que lo distinguen, como es el caso de la simplicidad en su código. Además de que este lenguaje, posee una de las comunidades más grandes de desarrolladores en el mundo y aquí en la propia UCI, lo cual es muy beneficioso a la hora de consultar alguna duda con alguien, además de que se tiene un poco más de experiencia en el uso de este lenguaje. Por otro parte, el hecho de usar PHP, justifica el uso del servidor Web Apache, los cuales poseen una gran compatibilidad, además de que este último, brinda algunas ventajas, como son: una gran modularidad, es gratuito y multiplataforma al igual que el PHP, además de que es muy configurable. Como editor de PHP se decidió utilizar el NuSphere PHPEd, debido a que, a pesar de poseer diversas características semejantes al Zend Studio, se tiene más experiencia en el trabajo con el primero. Se decidió también, el uso de Smarty, debido a que es un motor de plantillas para PHP, que tiene como finalidad, separar la capa de presentación de la capa lógica, lo cual trae grandes ventajas, a la hora de cambiar en cualquier momento el diseño de las páginas Web sin que afecte la lógica del negocio. En lo que respecta a la metodología de software a utilizar, se decidió escoger RUP, debido a que la misma abarca todo el ciclo de vida del software de manera organizada, dividiéndolo en fases, en las cuales, se realizan varias iteraciones que traen consigo que se vaya desarrollando el producto de forma incremental, además de que utiliza UML para la representación visual, el cual constituye un estándar a nivel internacional, que trae consigo un entendimiento común del sistema por parte de los diferentes desarrolladores. El hecho de haber escogido la metodología de desarrollo de software RUP, justifica en gran medida la decisión de seleccionar como herramienta CASE a Rational Rose, debido a que esta integra todos los elementos que propone dicha metodología para cubrir el

ciclo de vida de un proyecto, además de que es la herramienta líder del mundo en este sentido. En cuanto a la arquitectura, se decidió el uso de la de 3 capas, debido a que permite trabajar de una forma más organizada, lo cual trae grandes beneficios a la hora de hacer cambios y corregir errores.

Capítulo 2 Características del Sistema

2.1 Introducción

En el presente capítulo se enuncia y describe el objeto de estudio, además de que se dan a conocer cuáles son los procesos que serán automatizados. Por otra parte se exponen las reglas que debe cumplir el negocio en cuestión, además de describir los actores y trabajadores que intervienen en el mismo, y se muestra el modelo de objetos. También se identifican cuáles son los requisitos funcionales y no funcionales del sistema, que darán solución a los problemas existentes; a la vez que se presentan cuáles son las funcionalidades que brinda este sistema y quienes son los actores que interactúan con el mismo.

2.2 Objeto de Estudio

La Facultad 8 de la Universidad de las Ciencias Informáticas (UCI), constituye una de diez facultades por las que está compuesta dicha institución, en la que el objeto de estudio son los procesos que tienen lugar en la residencia estudiantil de la misma.

2.2.1 Objetivos estratégicos de la organización

La UCI es un centro universitario de reciente creación, que tiene como objetivo fundamental darles una formación integral a los jóvenes que cursan estudios en la misma. La premisa central de esta formación debe ser la de formar jóvenes altamente comprometidos con la patria y portadores de los ideales más puros que se necesitan de un revolucionario de estos tiempos, a la vez que se les da una sólida preparación académica para que una vez graduados, sean profesionales competentes que se pronuncien por resolver los problemas existentes en su entorno laboral, así como participar en la construcción de productos de software que impulsen el desarrollo económico del país.

Por otro lado, esta institución pretende ser un espacio no solo para fomentar la creación de sistemas informáticos, sino también para promover la cultura en todas sus manifestaciones, así como la práctica frecuente del deporte como complemento sano al gran cúmulo de actividades docentes que allí se realizan durante todo el curso.

Se pretende también, formar jóvenes que estén constantemente actualizados en cuanto a tecnología informática se refiere, además de que sean capaces de asimilar los constantes

cambios que se están produciendo en esta área del saber, así como, aplicar dichos conocimientos en cualquier lugar donde se encuentren, es decir, que sean profesionales que posean sobre todo, una alta disposición para cumplir cualquier tarea que le sea encomendada, por el tiempo que sea necesario.

Por ello, la ejecución de todos los procesos no docentes que tiene lugar en la facultad, revisten una importancia crucial, debido a que repercuten en la formación integral que se quiere dar a estos jóvenes, además de la importancia de tener un control de las actividades que se realizan fuera del marco docente, como es el caso de la residencia estudiantil y todo lo referente a la ubicación que posee cada uno de los estudiantes en la misma, además de tener un control sobre los partes de la guardia que se emiten a diario.

2.2.2 Flujo actual de los procesos

Analizar el flujo actual de los procesos, trae consigo que se pueda conocer como funciona en realidad el negocio en cuestión y de ahí obtener los posibles resultados. Estos últimos pueden ser: una información determinada, un servicio, un producto o combinaciones de ellos. El análisis de cómo se llevan a cabo estos procesos permite identificar un grupo de problemas que tienen lugar en la entidad donde se desarrollan y que dificultan el buen desenvolvimiento de las actividades que allí se realizan, como por ejemplo: demoras en la búsqueda de información, pérdida de información, duplicación de información o incorrecto formato en que se presenta la misma, entre otras.

Los procesos no docentes que tienen lugar en la facultad, referentes a la residencia estudiantil se describen a continuación:

Vicedecano de Extensión y Residencia (VDER): Es uno de los directivos de la facultad, el cual tiene la responsabilidad de ubicar a los estudiantes en cada uno de los edificios y apartamentos con los que cuenta la facultad, con el objetivo de tener organizado dónde se encuentran cada uno de los mismos, para facilitar con ello la búsqueda posterior de cualquier información relativa a la residencia estudiantil, como por ejemplo: en cuáles apartamentos se encuentran ubicados los estudiantes pertenecientes a una brigada determinada y de qué año son, quiénes son los estudiantes jefes de edificio, jefes de paso de escalera y jefes de apartamento, cuántos estudiantes hay ubicados en cada edificio, paso de escalera y apartamento de la facultad. Por otro lado, es también el encargado de llevar un control sobre la brigada que realizó la guardia estudiantil en la noche anterior, además de tener registrado cómo se han efectuado las mismas

en días pasados, en cuanto al cumplimiento o no de dicha actividad por parte de los estudiantes.

Secretaria del Vicedecano: Es la persona encargada de suministrar al VDER todas las informaciones que este solicite como por ejemplo: partes de la guardia estudiantil de un día determinado o de una brigada determinada. Además de ser la encargada de digitalizar una buena parte de las informaciones que el VDER le solicita a manera de reporte, entre las cuales se encuentran la mencionada anteriormente. Además de ayudar de alguna forma al VDER, a mantener un control sobre algunas tareas programadas por el mismo.

2.2.3 Análisis crítico de la ejecución de los procesos

En la actualidad, los procesos que se llevan a cabo en la facultad, específicamente en la residencia estudiantil, no se realizan de forma eficiente. En ello influyen algunos factores, como son:

- Demoras en la búsqueda de información
- Insuficiencia a la hora de recopilar información proveniente de fuentes diferentes.
- No centralización de la información.
- Pérdidas de información.
- No contar con la información que se necesita en tiempo.

Estos factores provocan lo siguiente: si el Vicedecano de Extensión y Residencia necesita en un momento determinado un reporte de todos los estudiantes que fueron ubicados en un edificio, paso de escalera u apartamento específico, tiene que acudir a varios documentos donde se encuentra dicha información. El proceso de ubicar a los estudiantes en cada uno de los edificios y apartamentos que fueron asignados a la facultad se torna un tanto engorroso, por cuanto, se realiza de forma manual, haciendo uso también de información proveniente de varias fuentes, lo cual provoca que la realización de esta actividad se tarde considerablemente. Por otro lado, el Vicedecano de Extensión y Residencia necesita conocer cómo se realizó la guardia estudiantil, no solo del día anterior, sino de todos los días, por lo que los reportes que recibe a diario de la realización de dicha actividad son almacenados, y con el paso del tiempo, provoca que exista un cierto cúmulo de información, trayendo consigo que a la hora de realizar alguna búsqueda de cómo se llevó a cabo la guardia estudiantil en un determinado día o de una brigada específica el proceso se dificulte en gran medida.

2.3 Procesos objeto de automatización

Los procesos que se desean automatizar son aquellos que tienen lugar en la residencia estudiantil de la facultad, y que de alguna forma u otra contribuirán a resolver los problemas que existen durante la realización de los mismos.

Por ejemplo, se quiere automatizar los siguientes procesos:

- Control de los partes de la guardia estudiantil que se realizan a diario.
- Control de la ubicación de los estudiantes en la residencia.
- Control sobre cada uno de los edificios y apartamentos que fueron asignados a la Facultad

2.4 Propuesta de Sistema

El sistema que se propone, pretende agilizar en gran medida la realización de varios procesos que tienen lugar en la residencia estudiantil de la Facultad 8 de la UCI, es decir, que se pueda acceder a la información requerida de manera casi instantánea, lo cual le reportaría grandes beneficios a aquellos personas que necesiten de la misma. Como es el caso de definir cuáles van a ser los edificios, pasos de escaleras y apartamentos con los que va a contar la Facultad para ubicar a cada uno de sus estudiantes, lo cual incluye, definir las características de cada uno de los apartamentos, en cuanto a: Capacidad, Teléfono y Sexo de las personas que serán ubicadas en los mismos, además de establecer también, una vez ubicados los estudiantes en estos, quien será el jefe de apartamento, así como jefe de paso de escalera y de edificio. Por otro lado se pretende también que el sistema brinde la posibilidad de realizar la ubicación de todos los estudiantes de la facultad en cada uno de los apartamentos definidos previamente, así como eliminar dicha ubicación, ya sea de un edificio completo, o de un paso de escalera u apartamento específico. Se pretende también que el sistema que se propone brinde la opción de generar reportes en documentos Excel sobre la ubicación de los estudiantes, ya sea por edificio, paso de escalera o apartamento. Y finalmente, que dicho sistema brinde también la posibilidad de conformar partes de la guardia estudiantil, además de almacenarlos para posteriormente hacer búsquedas de algún parte específico, ya sea en una fecha determinada o de una brigada específica.

2.5 Modelo del negocio actual

El **Vicedecano de Extensión y Residencia (VDER)** recibe diariamente un modelo con el parte de la guardia estudiantil realizada en la noche anterior, en el cual se refleja, entre otras cosas,

quienes cumplieron o no con la misma, a qué brigada pertenecen, en cuáles postas realizaron la guardia y en qué turno, además de los profesores que estaban ese día de guardia.

Por otro lado, **la secretaria docente** le entrega al inicio del curso, los listados de todas las brigadas y los estudiantes que las componen, con los cuales el VDER conforma la ubicación de los mismos en los edificios y apartamentos que le han sido asignados a la Facultad.

La **secretaria del VDER** recibe todos los meses de manos de este, los partes de la guardia estudiantil y se encarga de almacenar estos datos en la computadora para ser posteriormente utilizados por el VDER, cuando este lo solicite.

2.6 Reglas del negocio a considerar

Reglas de Restricción:

- Un estudiante no pede ser jefe de edificio, de paso de escalera y de apartamento a la vez.
- En un apartamento no pueden vivir a la vez varones y hembras.
- En un apartamento solo puede haber un número de teléfono y a su vez, un número de teléfono no puede estar en más de un apartamento.
- La cantidad de estudiantes que componen un apartamento es de 4 a 12.
- Un estudiante solo puede realizar la guardia en un turno y en una posta.
- En un día, la toca realizar la guardia a una brigada solamente.

Reglas del Modelo de Datos:

- El número que identifica a cada apartamento se conforma con el número del edificio, seguido del número de paso de escalera y seguido además del número consecutivo que posee cada apartamento en ese paso de escalera.

Reglas de Derivación:

- Cuando un estudiante no se presenta a realizar la guardia se dice que este incumplió con la misma.
- Cuando un estudiante abandona su puesto de guardia antes de culminar la misma se convierte en ausente.
- Cuando un estudiante no está en ninguno de los casos anteriores, se dice que este cumplió con la guardia.

2.7 Actores del negocio

Un actor del negocio es cualquier individuo, grupo, organización, máquina o sistema de información externo que interactúa con el negocio. El término *actor* significa el rol que algo o alguien juega cuando interactúa con el negocio para beneficiarse de sus resultados. De acuerdo con esta idea un actor del negocio representa un tipo particular de usuario del negocio más que un usuario físico, ya que varios usuarios físicos pueden realizar el mismo papel en relación al negocio, o sea, ser instancias de un mismo actor. [14]

A continuación se muestra la tabla 2.1, con los actores del negocio y la justificación de los mismos:

Actores del negocio	Justificación
Decano	Es la máxima instancia en la Facultad. Solicita información relativa a la ubicación de los estudiantes en la residencia estudiantil.
Vicedecano de Extensión y Residencia (VDER)	Es uno de los directivos de la Facultad que solicita informaciones relativas a la guardia estudiantil.

Tabla 2.1 Descripción de los actores del negocio.

2.8 Diagrama de casos de uso del negocio

El diagrama de casos de uso del negocio representa gráficamente los procesos del negocio y los actores que intervienen en el mismo, además de la relación que se establece entre estos.

A continuación se muestra la figura 2.1 correspondiente al diagrama de casos de uso del negocio.

Diagrama de casos de uso del negocio

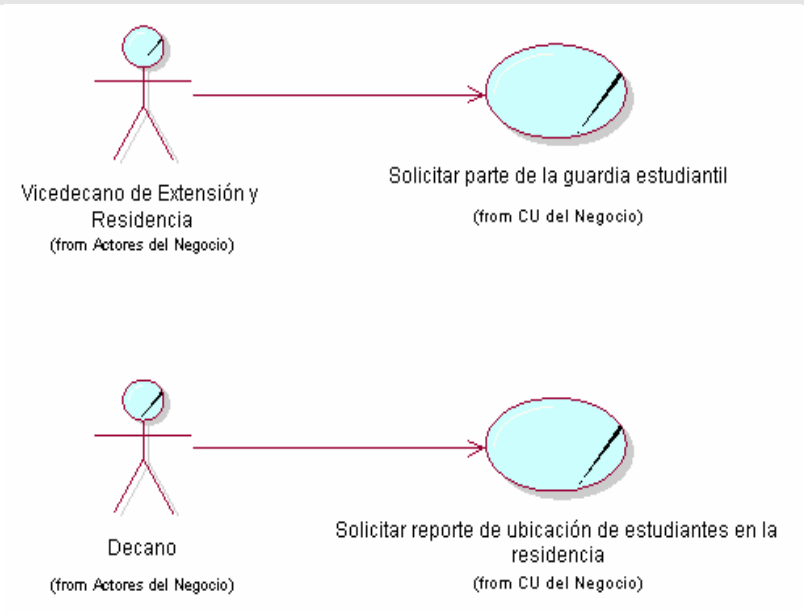


Figura 2.1 Diagrama de casos de uso del negocio

2.9 Trabajadores del negocio

Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos. [14]

A continuación se muestran en la tabla 2.2, los trabajadores del negocio y su correspondiente justificación:

Trabajadores del negocio	Justificación
Vicedecano de Extensión y Residencia	Realiza la ubicación de todos los estudiantes de la facultad en cada uno de los edificios y apartamentos que fueron asignadas a la misma.
Secretaria del Vicedecano de Extensión y Residencia.	Persona que asiste al Vicedecano de Extensión y Residencia. Es la encargada de conformar los partes diarios de la guardia estudiantil.
Secretaria Docente.	Persona que controla los listados de estudiantes que conforman cada una de las brigadas.

Tabla 2.2 Descripción de los trabajadores del negocio.

2.10 Especificación de los Casos de Uso del Negocio

La especificación de los Casos de Uso del Negocio, propicia una mayor comprensión de los procesos del negocio, tanto por parte del cliente como de los desarrolladores. Se **describe de forma textual** como se llevan a cabo las actividades dentro del negocio y quienes las realizan. Además de que se muestra el flujo de los procesos de manera gráfica, por medio de los **diagramas de actividades**, en los cuales, las actividades representan alguna acción que el actor o los trabajadores realizan, y las entidades no son más que la información con la cual estos interactúan.

A continuación se realiza la descripción textual de los Casos de Uso del Negocio y se muestran los diagramas de actividades de cada uno:

2.10.1 Descripción textual y diagrama de actividades del Caso de Uso del Negocio "Solicitar reporte de ubicación de estudiantes en la residencia"

Caso de Uso	**Solicitar reporte de ubicación de estudiantes en la residencia**
Actores	Decano (inicia).
Propósito	Conformar un reporte con la ubicación de todos los estudiantes de la Facultad en la residencia estudiantil.
Resumen	El CUN se inicia cuando el Decano solicita al inicio del curso un reporte con la ubicación de todos los estudiantes de la Facultad en la residencia estudiantil. El Vicedecano de Extensión y Residencia por su parte, conforma dicho reporte, apoyándose en la información brindada por la secretaria docente y el Decano acerca de los estudiantes que conforman cada una de las brigadas, así como, los edificios y apartamentos que dispone la Facultad respectivamente y termina el CUN.

Curso normal de los eventos	
Acción del actor	**Respuesta del Negocio**
1. El Decano solicita un reporte con la ubicación de todos lo estudiantes de la Facultad en la residencia estudiantil.	1.1 El Vicedecano de Extensión y Residencia le solicita un listado con los edificios y apartamentos que dispone la Facultad.
2. El Decano entrega el listado solicitado.	2.1 El Vicedecano de Extensión y Residencia recibe el listado solicitado.

	2.2 El Vicedecano de Extensión y Residencia solicita a la Secretaria Docente un listado con todas las brigadas de la Facultad, y los estudiantes que la conforman.
	2.3 La Secretaria Docente elabora y entrega el listado solicitado.
	2.4 El Vicedecano de Extensión y Residencia recibe el listado solicitado.
	2.5 El Vicedecano de Extensión y Residencia consulta la información contenida en los listados recibidos y conforma el reporte solicitado.
	2.6 El Vicedecano de Extensión y Residencia entrega el reporte solicitado.
3. El Decano recibe el reporte solicitado y termina el CUN.	

Tabla 2.3 Descripción textual del caso de uso del negocio “Solicitar reporte de ubicación de estudiantes en la residencia”

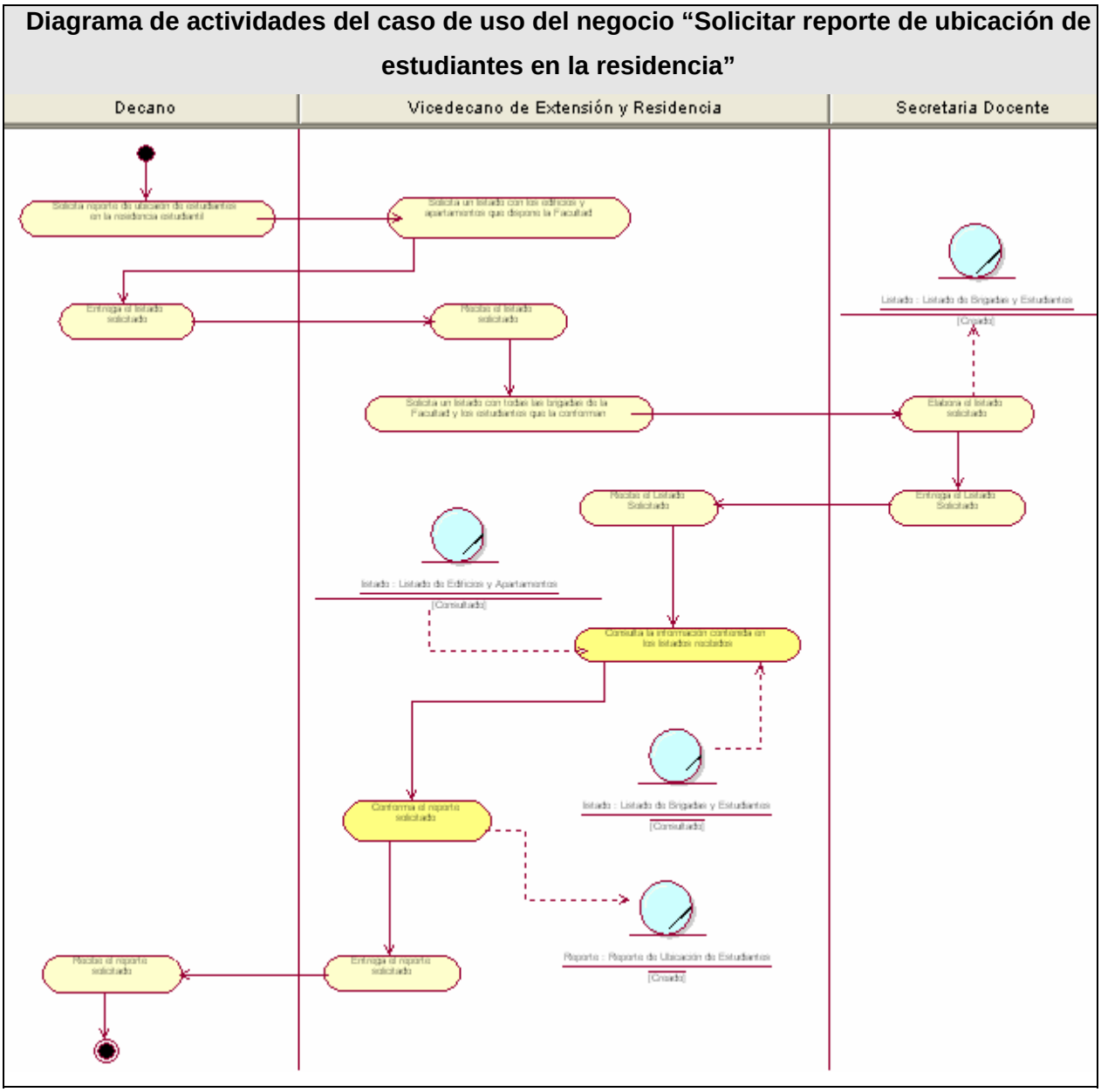


Figura 2.2 Diagrama de actividades del caso de uso del negocio “Solicitar reporte de ubicación de estudiantes en la residencia”

2.10.2 Descripción textual y diagrama de actividades del Caso de Uso “Solicitar parte de la guardia estudiantil”

Caso de Uso	**Solicitar parte de la guardia estudiantil**
Actores	Vicedecano de Extensión y Residencia (inicia).
Propósito	Conformar un parte de la guardia estudiantil.

Resumen	El CUS se inicia cuando el Vicedecano de Extensión y Residencia solicita un parte de la guardia estudiantil. Le entrega la información necesaria para ello a su Secretaria. Esta a su vez, conforma el parte solicitado y se lo entrega y termina el CUN.
Curso normal de los eventos	
Acción del actor	**Respuesta del Negocio**
1. El Vicedecano de Extensión y Residencia solicita un parte de la guardia estudiantil.	1.1 La Secretaria le solicita la información necesaria para conformar el parte solicitado.
2. El Vicedecano de Extensión y Residencia entrega la información solicitada.	2.1 La Secretaria recibe la información solicitada.
	2.2 La Secretaria consulta la información recibida y conforma el parte solicitado.
	2.3 La Secretaria entrega el parte solicitado.
3. El Vicedecano de Extensión y Residencia recibe el parte solicitado y termina el CUN.	

Tabla 2.4 Descripción textual del caso de uso del negocio “Solicitar parte de la guardia estudiantil”

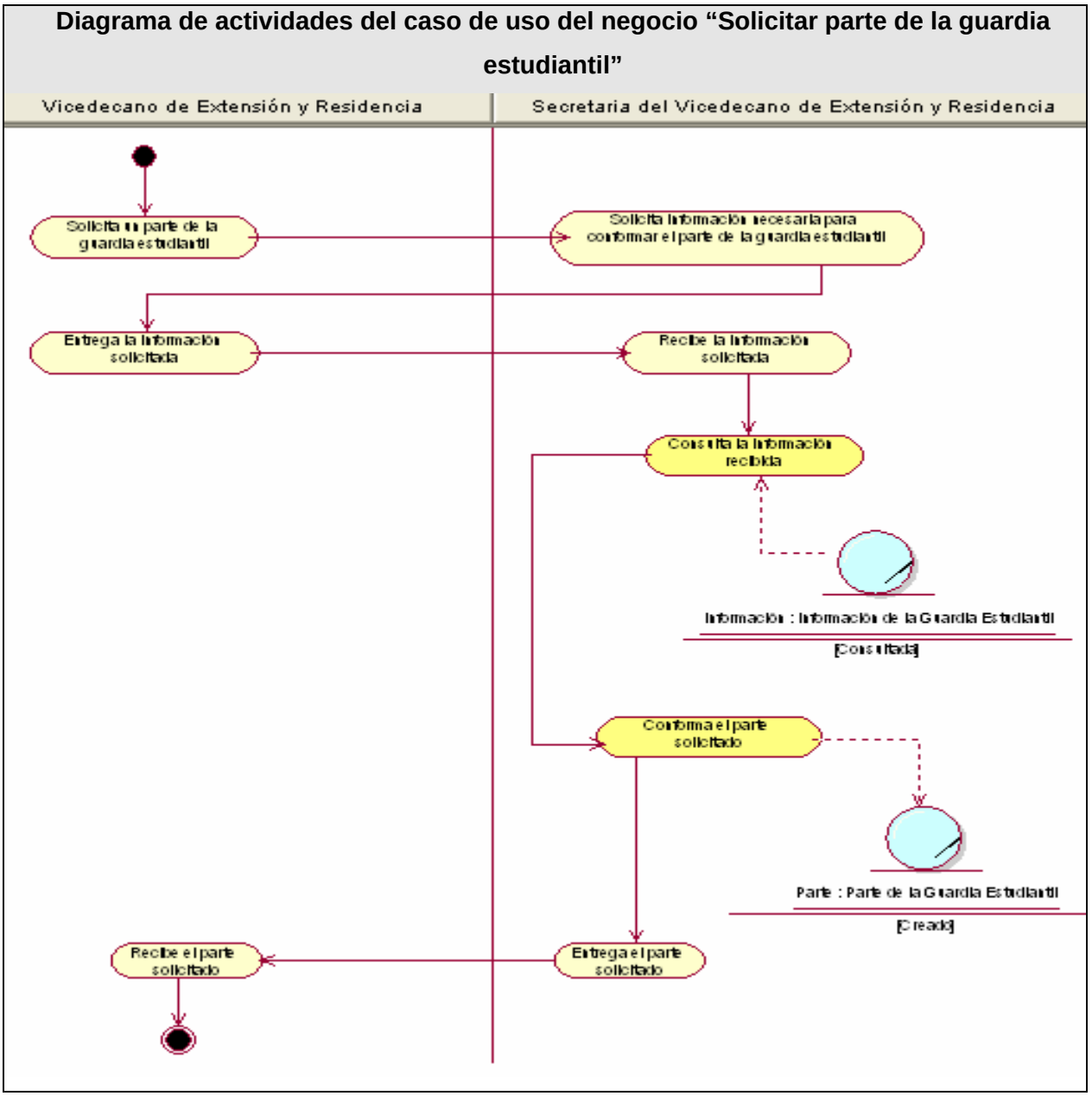


Figura 2.3 Diagrama de actividades del caso de uso del negocio “Solicitar parte de la guardia estudiantil”

2.11 Diagrama de clases del modelo de objetos

Un modelo de objetos del negocio es un modelo interno a un negocio. Describe cómo cada caso de uso del negocio es llevado a cabo por parte de un conjunto de trabajadores que utilizan un conjunto de entidades del negocio y unidades de trabajo. [14]

A continuación se muestra el diagrama de clases del modelo de objetos del negocio:

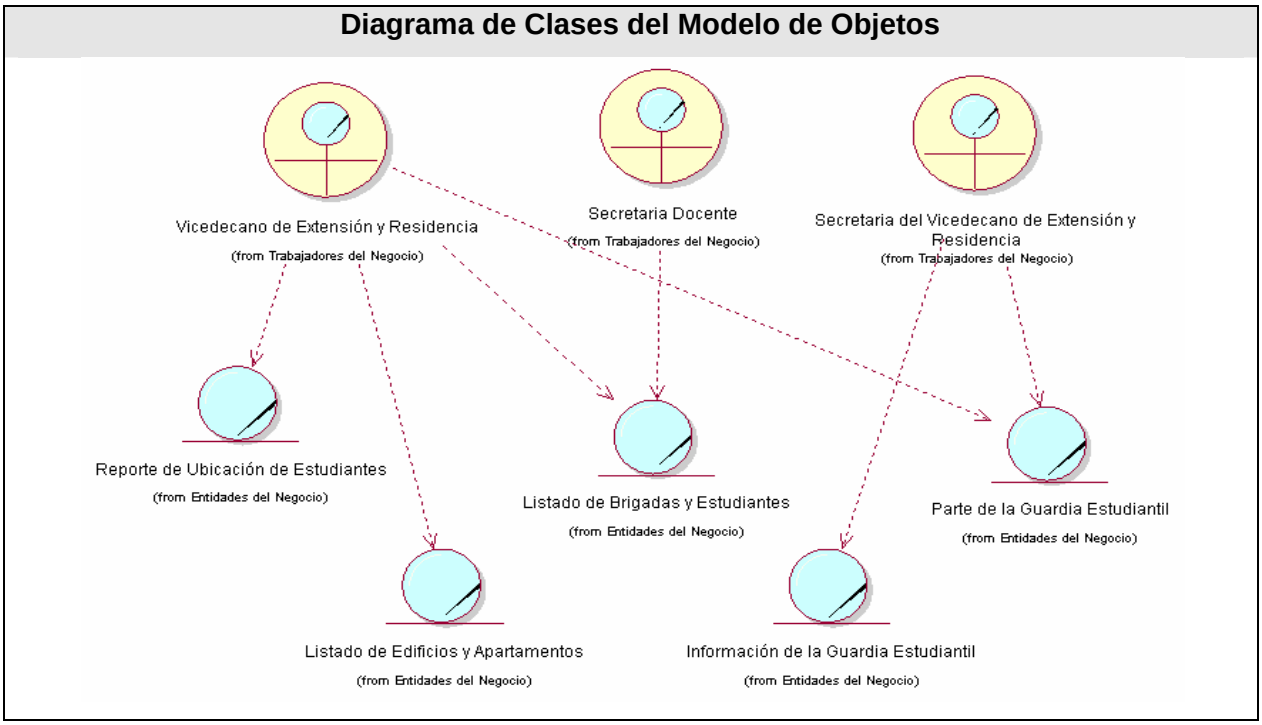


Figura 2.4 Diagrama de clases del modelo de objetos del negocio

2.12 Definición de los requisitos funcionales

Los requerimientos o requisitos funcionales, son capacidades o condiciones que el sistema debe cumplir, es decir, define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas.

A continuación se presentan las funcionalidades que debe cumplir el sistema en cuestión:

RF 1 Permitir Autenticarse

1.1 Autenticar usuarios.

1.2 Cerrar Sesión

RF 2 Gestionar Edificios

2.1 Adicionar edificios.

2.2 Listar edificios

2.2.1 Actualizar jefe de edificio.

2.2.2 Eliminar edificio.

2.2.3 Adicionar pasos de escalera

2.3 Listar pasos de escalera

2.3.1 Actualizar jefe de paso de escalera.

2.3.2 Adicionar apartamentos.

2.4 Listar apartamentos

2.4.1 Actualizar jefe de apartamento.

2.4.2 Actualizar datos del apartamento.

2.5 Generar reporte de ubicación de estudiantes

2.5.1 Generar reporte de ubicación de estudiantes por edificio.

2.5.2 Generar reporte de ubicación de estudiantes por paso de escalera.

2.5.3 Generar reporte de ubicación de estudiantes por apartamento.

RF 3 Gestionar ubicación de estudiantes en la residencia

3.1 Ubicar uno o varios estudiantes en un apartamento.

3.2 Eliminar ubicación de uno o varios estudiantes de un apartamento, paso de escalera o edificio.

RF 4 Gestionar parte de la guardia estudiantil

4.1 Adicionar parte de la guardia estudiantil.

4.2 Listar partes de la guardia estudiantil.

4.2.1 Consultar datos de un parte de la guardia estudiantil.

2.13 Definición de los requisitos no funcionales

Los requerimientos o requisitos no funcionales son propiedades o cualidades que el producto debe tener, que posibilitan que este sea más atractivo, usable, rápido, confiable, entre otras. También, puede ser alguna restricción que este debe tener para cumplir una determinada funcionalidad.

A continuación se presentan los requisitos no funcionales:

Software

RNF 1. Servidor Web Apache v1.x o superior.

RNF 2. SGBD: PostgreSQL preferiblemente v7.x en adelante.

RNF 3. Navegador Internet Explorer v4.0 o superior.

RNF 4. Adobe Acrobat Reader 7.0 en adelante.

RNF 5. Microsoft Excel.

Hardware

RNF 6. Tarjeta de red.

RNF 7. Para los servidores tanto Web como SGBD: PENTIUM II o superior con 256 MB de RAM o más.

RNF 8. Capacidad de disco duro en Gigabyte, preferiblemente mayor a los 10 GB.

Apariencia o interfaz externa

RNF 9. La interfaz no debe contener muchas imágenes que demoren las respuestas al usuario.

RNF 10. El diseño de la interfaz debe ser sencillo y claro. Debe contener elementos visibles que identifiquen cada una de sus acciones.

RNF 11. La navegabilidad debe ser sencilla.

Usabilidad

RNF 12. El sistema podrá ser usado por cualquier persona que acceda a él que tenga algún conocimiento básico de computación y trabajo en la Web.

RNF 13. Rápido acceso de búsqueda de la información, en tiempos cortos.

Rendimiento

RNF 14. El sistema deberá ser capaz de gestionar toda la información y dar respuesta a las solicitudes lo más rápido posible.

RNF 15. Debe ser eficiente a la hora de gestionar las solicitudes logrando que sin mucha navegación por el sitio se obtenga los resultados deseados.

RNF 16. Debe estar disponible las 24 horas del día.

Soporte

RNF 17. El sistema debe ser de fácil instalación y configuración, con vista a poder darle un mantenimiento asequible en caso de fallos.

Portabilidad

RNF 18. Multiplataforma. El sistema se podrá montar sobre Unix, Linux, Windows, etc. Así mismo podrá usar una serie de SGBD como PostgreSQL, MySQL, Oracle, entre otros, aunque preferiblemente se desea la portabilidad sobre software libre.

Seguridad

RNF 19. Realizar todas las validaciones pertinentes.

RNF 20. Chequeo de seguridad sobre las acciones tales como: verificación de borrado.

Políticos-culturales

RNF 21. El sistema debe tener una interfaz que esté acorde con el lugar donde se implantará, es decir, que refleje los ideales de la organización.

Legales

RNF 22. Reconocido y autorizado por instancias superiores tales como la directiva de la UCI.

RNF 23. Documentación legal de uso como Declaración de Autoría.

Ayuda y documentación en línea

RNF 24. Documentación de ayuda para uso del sistema, la cual estará asequible desde cualquier parte del mismo para satisfacer cualquier duda que el usuario presente con el manejo y uso de la aplicación.

Restricciones en el diseño y la implementación

RNF 25. Para la programación en PHP se recomienda el editor Nusphere PHPEd.

RNF 26. Se recomienda el uso de la arquitectura de n capas con el fin de tener una mayor organización del código, además de hacer el software más extensible.

2.14 Actores del sistema

Los actores del sistema definen el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que interactúan con el mismo intercambiando información con este.

A continuación se detallan los actores del sistema:

Actores del sistema	Justificación
Vicedecano de Extensión y Residencia	Es el encargado de realizar todo el proceso de ubicar a los estudiantes en la residencia, para tener posteriormente un control sobre quienes viven en cada uno de los apartamentos, pasos de escalera y edificios pertenecientes a la facultad.
Secretaria del Vicedecano	Es la encargada de registrar todos los datos relacionados con los partes de la guardia estudiantil y suministrar todas las informaciones relacionadas con esto al Vicedecano de Extensión y Residencia.
Usuario	Es todo aquel que hace uso del sistema

Tabla 2.5 Descripción de los actores del sistema.

2.15 Diagrama de casos de uso del sistema

Los Casos de Uso del sistema (CUS) son un conjunto de secuencia de acciones que un sistema ejecuta y que produce un resultado observable para un actor. Con otras palabras, son "fragmentos" de funcionalidad que el sistema ofrece a los actores que interactúan con el mismo, reportándoles algunos que otro beneficio.

A continuación se muestra la figura 2.5 correspondiente al diagrama de casos de uso del sistema:

Diagrama de casos de uso del sistema

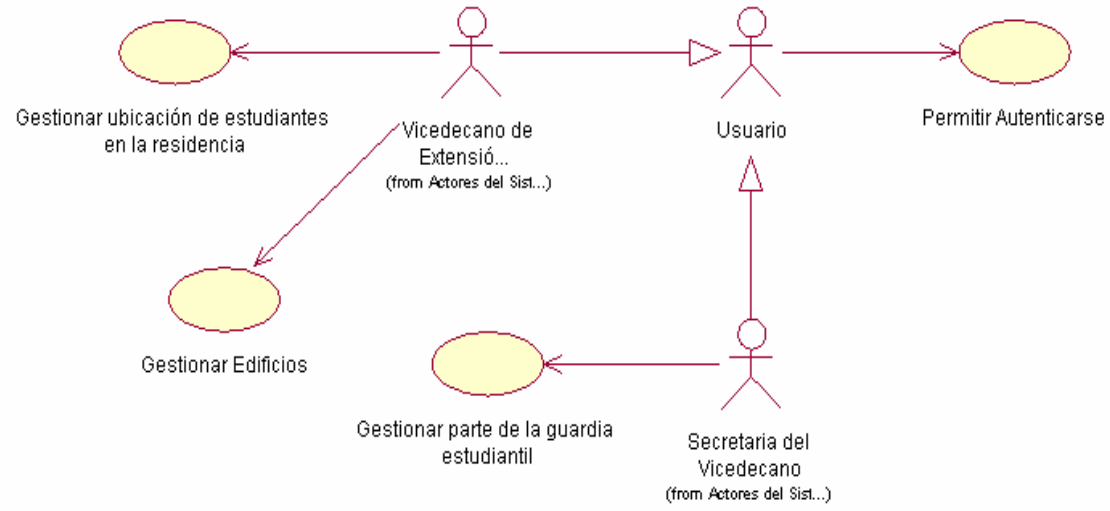


Figura 2.5 Diagrama de casos de uso del sistema

2.15.1 Descripción de los casos de uso del sistema

La descripción de los casos de uso del sistema, detallan las acciones que tienen lugar durante la interacción actor-sistema, es decir, describe el flujo de actividades que realiza el actor al hacer uso del sistema y las correspondientes respuestas del mismo. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre que es lo que el sistema debe hacer (requisitos).

2.15.1.1 Descripción del CUS "Permitir Autenticarse"

Caso de Uso	**Permitir Autenticarse**
Actores	Usuario (inicia).
Propósito	Permite a los usuarios del sistema identificarse previamente para hacer uso del sistema, según el rol que cada uno tenga, además de cerrar su

	sesión de trabajo.
Resumen	El CUS se inicia cuando un usuario del sistema desea hacer uso del mismo, para lo cual, es necesario primeramente identificarse y luego el sistema le da acceso o no a los recursos que le son permitidos, según su rol y termina el CUS.
Referencias	RF 1
Precondiciones	- Usuario registrado en la base de datos.
Poscondiciones	1. El sistema le da acceso al usuario a los recursos que le son permitidos según su rol. 2. El sistema cierra la sesión de trabajo del usuario autenticado.

Curso normal de los eventos	
Acción del actor	**Respuesta del sistema**
1. El usuario del sistema desea acceder al mismo.	1.1 El sistema muestra el formulario para ingresar los datos necesarios para ser identificado y otorgar o no el acceso al mismo.
2. El usuario introduce los datos solicitados por el sistema y presiona el botón “Aceptar”.	2.1 El sistema verifica los datos introducidos por el usuario.
	2.2 Si los datos introducidos se corresponden con los de un usuario registrado en la Base de Datos, el sistema le da acceso al mismo a los recursos que le son permitidos, según su rol y termina el CUS.
Curso alterno de los eventos	
Acción 2.2	Si los datos introducidos por el usuario, no se corresponden con los de un usuario registrado en la Base de Datos, el sistema muestra un mensaje de error e indica al usuario retornar a la acción 2.
Escenario 1: Cerrar Sesión	

1. El usuario decide cerrar su sesión de trabajo y presiona el botón: "Cerrar Sesión".	1.1 El sistema cancela todas las acciones que estaba ejecutando el usuario y lo redirecciona para la página de Autenticación y termina el CUS.
Prioridad	Crítico

Tabla 2.6 Descripción del caso de uso "Permitir Autenticarse"

2.15.1.2 Descripción del CUS "Gestionar Edificios"

Caso de Uso	**Gestionar Edificios**
Actores	Vicedecano de Extensión y Residencia (inicia).
Propósito	Permite al Vicedecano de Extensión y Residencia gestionar la información referente a un edificio (Adicionar Edificio, Adicionar Paso de Escalera, Adicionar Apartamento, Eliminar Edificio, Actualizar Datos del Apartamento, Actualizar Jefe de Edificio, Actualizar Jefe de Paso de Escalera, Actualizar Jefe de Apartamento y Generar Reporte de Ubicación de Estudiantes).
Resumen	El CUS se inicia cuando el Vicedecano selecciona la opción de Gestionar Edificios, luego selecciona el tipo de gestión, introduce los datos necesarios, el sistema realiza la acción seleccionada por el Vicedecano y termina el CUS.
Referencias	RF 2
Precondiciones	-
Poscondiciones	1. Edificio adicionado a la Base de Datos. 2. Edificio eliminado de la Base de Datos. 3. Paso de escalera adicionado a la Base de Datos. 4. Apartamento adicionado a la Base de Datos. 5. Datos de un apartamento modificado en la Base de Datos. 6. Jefe de apartamento actualizado en la Base de Datos. 7. Jefe de paso de escalera actualizado en la Base de Datos. 8. Jefe de edificio actualizado en la Base de Datos. 9. Reporte de ubicación de estudiantes mostrado al Vicedecano.
Curso normal de los eventos	

Acción del actor	**Respuesta del sistema**
1. El Vicedecano selecciona la opción de Gestionar Edificios.	1.1 El sistema muestra las opciones: Adicionar Edificio, Eliminar Edificio, Adicionar Paso de Escalera, Adicionar Apartamento, Modificar Datos de un Apartamento, Actualizar Jefe de Apartamento, Actualizar Jefe de Paso de Escalera, Actualizar Jefe de Edificio. Generar Reporte de Ubicación.
Escenario 1: Adicionar Edificio	
1. El Vicedecano selecciona la opción de Adicionar Edificio.	1.1 El sistema muestra el formulario a completar para la adición de un nuevo edificio.
2. El Vicedecano introduce los datos solicitados por el sistema y presiona el botón: “Adicionar”.	2.1 El sistema verifica los datos introducidos por el Vicedecano.
	2.2 Si los datos introducidos son correctos, el sistema adiciona dicho edificio a la Base de Datos y termina el CUS.
Curso alterno de los eventos	
Acción 2.2	Si los datos introducidos por el Vicedecano son incorrectos, el sistema muestra un mensaje de error indicando donde está el dato erróneo e indica al Vicedecano retornar a la acción 2.
Escenario 2: Eliminar Edificio	
1. El Vicedecano selecciona la opción de Eliminar Edificio.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos.
2. El Vicedecano selecciona el edificio que será eliminado y presiona el botón “Eliminar”.	2.1 El sistema muestra un mensaje de advertencia para la acción a realizar.
3. El Vicedecano confirma si desea o no	3.1 Si el Vicedecano acepta, el sistema

eliminar el edificio seleccionado.	elimina el edificio seleccionado de la Base de Datos y termina el CUS.
Curso alterno de los eventos	
Acción 3.1	Si el Vicedecano cancela la acción, se culmina el CUS sin ejecutar ninguna acción.
Escenario 3: Adicionar Paso de Escalera	
1. El Vicedecano selecciona la opción de Adicionar Paso de Escalera.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos.
2. El Vicedecano selecciona el edificio al cual le desea adicionar pasos de escalera y presiona el botón “Adicionar”.	2.1 El sistema adiciona un paso de escalera al edificio seleccionado en la Base de Datos y termina el CUS.
Escenario 4: Adicionar Apartamento	
1. El Vicedecano selecciona la opción de Adicionar Apartamento.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos.
2. El Vicedecano selecciona el edificio al que le desea adicionar un apartamento.	2.1 El sistema muestra un listado con los pasos de escalera que conforman el edificio seleccionado.
3. El Vicedecano selecciona el paso de escalera al que le desea adicionar un apartamento.	3.1 El sistema muestra el formulario a completar para la adición de un nuevo apartamento.
4. El Vicedecano introduce los datos solicitados por el sistema y presiona el botón: “Adicionar”.	4.1 El sistema verifica los datos introducidos por el Vicedecano.
	4.2 Si los datos introducidos son correctos, el sistema adiciona dicho apartamento a la Base de Datos y termina el CUS.
Curso alterno de los eventos	

Acción 4.2	Si los datos introducidos por el Vicedecano son incorrectos, el sistema muestra un mensaje de error indicando donde está el dato erróneo e indica al Vicedecano retornar a la acción 4.
Escenario 5: Modificar Datos de un Apartamento	
1. El Vicedecano selecciona la opción de Modificar Datos de un Apartamento.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos.
2. El Vicedecano selecciona el edificio que contiene el apartamento al que le desea actualizar o modificar sus datos.	2.1 El sistema muestra un listado con los pasos de escalera que conforman el edificio seleccionado.
3. El Vicedecano selecciona el paso de escalera que contiene el apartamento al que le desea modificar sus datos.	3.1 El sistema muestra un listado con los apartamentos que pertenecen al paso de escalera seleccionado.
4. El Vicedecano selecciona el apartamento al que le desea modificar sus datos.	4.1 El sistema localiza los datos del apartamento seleccionado y los muestra, listos para ser modificados.
5. El Vicedecano realiza los cambios necesarios a los datos y presiona el botón: “Actualizar”.	5.1 El sistema verifica los datos modificados por el Vicedecano.
	5.2 Si los datos están correctos el sistema actualiza los datos del apartamento seleccionado en la Base de Datos y termina el CUS.
Curso alterno de los eventos	
Acción 5.2	Si los datos introducidos por el Vicedecano son incorrectos, el sistema muestra un mensaje de error indicando donde está el dato erróneo e indica al Vicedecano retornar a la acción 5.
Escenario 6: Actualizar Jefe de Apartamento	

1. El Vicedecano selecciona la opción de Actualizar Jefe de Apartamento.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos.
2. El Vicedecano selecciona el edificio que contiene el apartamento al que le desea actualizar el jefe.	2.1 El sistema muestra un listado con los pasos de escalera que conforman el edificio seleccionado.
3. El Vicedecano selecciona el paso de escalera que contiene el apartamento al que le desea actualizar el jefe.	3.1 El sistema muestra un listado con los apartamentos que pertenecen al paso de escalera seleccionado.
4. El Vicedecano selecciona el apartamento al que le desea actualizar el jefe.	4.1 El sistema localiza los datos del actual jefe del apartamento seleccionado y los muestra, además de que muestra un listado con los estudiantes que fueron ubicados en dicho apartamento.
5. El Vicedecano selecciona al estudiante que será nuevo jefe del apartamento seleccionado y presiona el botón: "Actualizar".	5.1 El sistema verifica si el estudiante seleccionado puede o no ser jefe del apartamento seleccionado.
	5.2 Si el estudiante seleccionado puede ser jefe del apartamento seleccionado el sistema actualiza el jefe de apartamento en la Base de Datos y termina el CUS.
Curso alterno de los eventos	
Acción 5.2	Si el estudiante seleccionado no puede ser jefe del apartamento seleccionado, el sistema muestra un mensaje de error e indica al Vicedecano retornar a la acción 5.
Escenario 7: Actualizar Jefe de Paso de Escalera	
1. El Vicedecano selecciona la opción de Actualizar Jefe de paso de escalera	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos.

2. El Vicedecano selecciona el edificio que contiene el paso de escalera al que se le desea actualizar el jefe.	2.1 El sistema muestra un listado con los pasos de escalera que conforman el edificio seleccionado.
3. El Vicedecano selecciona el paso de escalera al que le desea actualizar el jefe.	3.1 El sistema localiza los datos del jefe de paso de escalera actual y los muestra, además de mostrar un listado con los estudiantes que fueron ubicados en el paso de escalera seleccionado.
4. El Vicedecano selecciona al estudiante que será nuevo jefe del paso de escalera seleccionado y presiona el botón: “Actualizar”.	4.1 El sistema verifica si el estudiante seleccionado puede o no ser jefe del paso de escalera seleccionado.
	4.2 Si el estudiante seleccionado puede ser jefe del paso de escalera seleccionado, el sistema actualiza el jefe de paso de escalera en la Base de Datos y termina el CUS.
Curso alterno de los eventos	
Acción 4.2	Si el estudiante seleccionado no puede ser jefe del paso de escalera seleccionado, el sistema muestra un mensaje de error e indica al Vicedecano retornar a la acción 4.
Escenario 8: Actualizar Jefe de Edificio	
1. El Vicedecano selecciona la opción de Actualizar Jefe de Edificio.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos.
2. El Vicedecano selecciona el edificio al cual le desea actualizar el jefe.	2.1 El sistema localiza los datos del actual jefe del edificio seleccionado y los muestra, además de que muestra un listado con los estudiantes que fueron ubicados en dicho edificio.

3. El Vicedecano selecciona al estudiante que será el nuevo jefe de edificio y presiona el botón “Actualizar”.	3.1 El sistema verifica si el estudiante seleccionado puede o no ser jefe del edificio seleccionado.
	3.2 Si el estudiante seleccionado puede ser jefe del edificio seleccionado el sistema actualiza el jefe de edificio en la Base de Datos y termina el CUS
Curso alterno de los eventos	
Acción 3.2	Si el estudiante seleccionado no puede ser jefe del edificio seleccionado, el sistema muestra un mensaje de error e indica al Vicedecano retornar a la acción 3.
Escenario 9: Generar Reporte de Ubicación de Estudiantes	
1. El Vicedecano selecciona la opción de Generar reporte de ubicación de estudiantes.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos.
2. El Vicedecano decide si lo que desea es obtener un reporte de ubicación de estudiantes por edificio, por paso de escalera o por apartamento.	
Sección 1: Generar reporte de ubicación de estudiantes por edificio	
1. El Vicedecano selecciona un edificio y decide generar un reporte de ubicación de estudiantes por edificio y presiona el botón: “Ver Reporte”.	1.1 El sistema genera un documento Excel que contiene un reporte con la ubicación de todos los estudiantes que fueron ubicados en el edificio seleccionado, organizados por paso de escalera y apartamento y termina el CUS.
Sección 2: Generar reporte de ubicación de estudiantes por paso de escalera	
1. El Vicedecano selecciona un edificio y decide generar un reporte de ubicación de estudiantes por paso de escalera.	1.1 El sistema muestra un listado con los pasos de escalera pertenecientes al edificio seleccionado.

2. El Vicedecano selecciona un paso de escalera y presiona el botón: "Ver Reporte"	2.1 El sistema genera un documento Excel que contiene un reporte con la ubicación de todos los estudiantes que fueron ubicados en el paso de escalera seleccionado, organizados por apartamento y termina el CUS.
Sección 3: Generar reporte de ubicación de estudiantes por apartamento	
1. El Vicedecano selecciona un edificio y decide generar un reporte de ubicación de estudiantes por apartamento.	1.1 El sistema muestra un listado con los pasos de escalera que componen el edificio seleccionado.
2. El Vicedecano selecciona un paso de escalera.	2.1 El sistema muestra un listado con los apartamentos que componen el paso de escalera seleccionado.
3. El Vicedecano selecciona un apartamento y presiona el botón: "Ver Reporte"	3.1 El sistema genera un documento Excel que contiene un reporte con la ubicación de todos los estudiantes que fueron ubicados en el apartamento seleccionado, organizados por nombre y apellidos y termina el CUS.
Prioridad	Crítico

Tabla 2.7 Descripción del caso de uso "Gestionar Edificios"

2.15.1.3 Descripción del CUS "Gestionar ubicación de estudiantes en la residencia"

Caso de Uso	**Gestionar ubicación de estudiantes en la residencia**
Actores	Vicedecano de Extensión y Residencia (inicia).
Propósito	Permite al Vicedecano de Extensión y Residencia ubicar o eliminar ubicación de los estudiantes en la residencia estudiantil.
Resumen	El CUS se inicia cuando el Vicedecano selecciona la opción de Gestionar ubicación de estudiantes en la residencia, luego selecciona el tipo de gestión, introduce los datos necesarios, el sistema realiza la acción seleccionada por el Vicedecano y termina el CUS.
Referencias	RF 3

Precondiciones	-
Poscondiciones	1. Ubicación de uno o varios estudiantes registrada en la Base de Datos. 2. Ubicación de uno o varios estudiantes eliminada de la Base de Datos.

Curso normal de los eventos	
Acción del actor	**Respuesta del sistema**
1. El Vicedecano selecciona la opción de Gestionar ubicación de estudiantes en la residencia.	1.1 El sistema muestra las opciones: Ubicar Estudiantes, Eliminar Ubicación de Estudiantes.
Escenario 1: Ubicar Estudiantes	
1. El Vicedecano selecciona la opción de Ubicar Estudiantes.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos, en el que se especifica la capacidad de cada uno de estos y la cantidad de estudiantes que ya han sido ubicados en los mismos.
2. El Vicedecano selecciona el edificio donde desea ubicar el o los estudiantes.	2.1 El sistema muestra un listado con los pasos de escalera que componen el edificio seleccionado, especificando la capacidad de cada uno, además de la cantidad de estudiantes que ya han sido ubicados en estos.
3. El Vicedecano selecciona el paso de escalera donde desea ubicar el o los estudiantes.	3.1 El sistema muestra un listado con los apartamentos que componen el paso de escalera seleccionado en el que se especifica, la capacidad de cada uno, la cantidad de estudiantes que ya han sido ubicados en estos, así como el sexo de las personas que viven en el mismo.

4. El Vicedecano selecciona el apartamento donde desea ubicar el o los estudiantes.	4.1 El sistema muestra un listado con las brigadas de estudiantes existentes en la Base de Datos, en el que se especifica por cada una, la cantidad de estudiantes que posee del mismo sexo del apartamento seleccionado, además de la cantidad de estos que ya han sido ubicados.
5. El Vicedecano selecciona la brigada que contiene el o los estudiantes que desea ubicar en el apartamento seleccionado.	5.1 El sistema muestra un listado con los estudiantes pertenecientes a la brigada seleccionada que aún no poseen ubicación en la residencia.
6. El Vicedecano selecciona el o los estudiantes que desea ubicar en el apartamento seleccionado y presiona el botón “Ubicar”.	6.1 El sistema comprueba si la cantidad de estudiantes seleccionados se corresponde con la capacidad disponible en el apartamento seleccionado.
	6.2 Si la cantidad de estudiantes seleccionados es menor o igual que la capacidad disponible en el apartamento seleccionado, el sistema ubica dichos estudiantes en dicho apartamento y termina el CUS.
Curso alterno de los eventos	
Acción 6.2	Si la cantidad de estudiantes seleccionados es mayor que la capacidad disponible en el apartamento seleccionado, el sistema muestra un mensaje de error e indica al Vicedecano retornar a la acción 6.
Escenario 2: Eliminar Ubicación de Estudiantes	
1. El Vicedecano selecciona la opción de Eliminar Ubicación de Estudiantes.	1.1 El sistema muestra un listado con los edificios existentes en la Base de Datos en el que se especifica la capacidad de

	cada uno de estos y la cantidad de estudiantes que ya han sido ubicados en los mismos.
2. El Vicedecano decide si lo que desea es eliminar la ubicación de un edificio completo, o paso de escalera completo o varios estudiantes de un apartamento.	
Sección 1: Edificio Completo	
1. El Vicedecano selecciona un edificio y decide eliminar la ubicación de todos los estudiantes que fueron ubicados en el mismo y presiona el botón: “Eliminar”.	1.1 El sistema muestra un mensaje de advertencia para la acción a realizar.
	1.2 Si el Vicedecano acepta, el sistema elimina la ubicación de todos los estudiantes que fueron ubicados en el edificio seleccionado y termina el CUS.
Curso alterno de los eventos	
Acción 1.2	Si el Vicedecano cancela, se culmina el CUS sin ejecutar ninguna acción.
Sección 2: Paso de Escalera Completo	
1. El Vicedecano selecciona un edificio y decide eliminar la ubicación de todos los estudiantes que fueron ubicados en alguno de los pasos de escalera que componen el mismo.	1.1 El sistema muestra un listado con los pasos de escalera que componen el edificio seleccionado, en el que se especifica la capacidad de cada uno de estos, además de la cantidad de estudiantes que han sido ubicados en los mismos.
2. El Vicedecano selecciona un paso de escalera y presiona el botón: “Eliminar”.	2.1 El sistema muestra un mensaje de advertencia para la acción a realizar.
	2.2 Si el Vicedecano acepta, el sistema elimina la ubicación de todos los estudiantes que fueron ubicados en el

	paso de escalera seleccionado y termina el CUS.
Curso alterno de los eventos	
Acción 2.2	Si el Vicedecano cancela, se culmina el CUS sin ejecutar ninguna acción.
Sección 3: Estudiantes por Apartamento	
1. El Vicedecano selecciona un edificio y decide eliminar la ubicación de todos o algunos de los estudiantes que fueron ubicados en alguno de los apartamentos que componen el mismo.	1.1 El sistema muestra un listado con los pasos de escalera que componen el edificio seleccionado, en el que se especifica la capacidad de cada uno de estos, además de la cantidad de estudiantes que han sido ubicados en los mismos.
2. El Vicedecano selecciona un paso de escalera.	2.1 El sistema muestra un listado con los apartamentos que componen el paso de escalera seleccionado, en el que se especifica la capacidad de cada uno de estos, la cantidad de estudiantes que han sido ubicados en los mismos, además del sexo de los estudiantes que viven en cada uno de estos.
3. El Vicedecano selecciona un apartamento.	3.1 El sistema muestra un listado con todos los estudiantes que fueron ubicados en el apartamento seleccionado.
4. El Vicedecano selecciona el o los estudiantes que desea eliminar su ubicación del apartamento seleccionado y presiona el botón: “Eliminar”.	4.1 El sistema muestra un mensaje de advertencia para la acción a realizar.
	4.2 Si el Vicedecano acepta, el sistema elimina la ubicación del o los estudiantes seleccionados y termina el CUS.
Curso alterno de los eventos	

Acción 4.2	Si el Vicedecano cancela la acción, se culmina el CUS sin ejecutar ninguna acción.
Prioridad	Crítico

Tabla 2.8 Descripción del caso de uso "Gestionar ubicación de estudiantes en la residencia"

2.15.1.4 Descripción del CUS "Gestionar parte de la guardia estudiantil"

Caso de Uso	**Gestionar parte de la guardia estudiantil**
Actores	Secretaria del Vicedecano (inicia).
Propósito	Permite a la Secretaria del Vicedecano gestionar (Adicionar o Consultar) la información referente a un parte de la guardia estudiantil.
Resumen	El CUS se inicia cuando la Secretaria del Vicedecano selecciona la opción de Gestionar parte de la guardia estudiantil, luego selecciona el tipo de gestión, introduce los datos necesarios, el sistema realiza la acción seleccionada por la Secretaria y termina el CUS.
Referencias	RF 4
Precondiciones	-
Poscondiciones	1. Parte de la guardia estudiantil adicionado a la Base de Datos. 2. Información del parte de la guardia estudiantil mostrado a la Secretaria del Vicedecano.

Curso normal de los eventos	
Acción del actor	**Respuesta del sistema**
1. La Secretaria del Vicedecano selecciona la opción de Gestionar parte de la guardia estudiantil.	2.1 El sistema muestra las opciones: Adicionar parte de la guardia estudiantil y Consultar parte de la guardia estudiantil.
Escenario 1: Adicionar parte de la guardia estudiantil	
1. La Secretaria del Vicedecano selecciona la opción de Adicionar parte de la guardia estudiantil.	1.1 El sistema muestra el formulario a completar para la adición de un nuevo parte de la guardia estudiantil.
2. La Secretaria del Vicedecano introduce los datos solicitados por el sistema y presiona el botón: "Adicionar".	2.1 El sistema verifica los datos introducidos por la Secretaria del Vicedecano.

	2.2 Si los datos introducidos son correctos, el sistema adiciona dicho parte de la guardia estudiantil a la Base de Datos y termina el CUS.
Curso alterno de los eventos	
Acción 2.2	Si los datos introducidos por la Secretaria del Vicedecano son incorrectos, el sistema muestra un mensaje de error indicando donde está el dato erróneo e indica retornar a la acción 2.
Escenario 2: Consultar parte de la guardia estudiantil	
2. La Secretaria del Vicedecano selecciona la opción de Consultar parte de la guardia estudiantil.	1.2 El sistema muestra las siguientes opciones para la búsqueda de un parte de la guardia estudiantil: Buscar parte por Brigada y Buscar parte por Fecha.
3. La Secretaria del Vicedecano selecciona la opción por la que desea buscar un parte de la guardia estudiantil.	
Sección 1: Buscar Parte por Brigada	
1. La Secretaria del Vicedecano selecciona la opción Buscar parte por brigada.	1.1 El sistema muestra un listado con todas las brigadas existentes en la Base de Datos.
2. La Secretaria del Vicedecano selecciona la brigada de la cual desea consultar parte de la guardia estudiantil y presiona el botón “Buscar”.	2.1 Si la brigada seleccionada posee algún parte, el sistema muestra un listado con los partes de la guardia estudiantil correspondientes a dicha brigada.
3. La Secretaria del Vicedecano selecciona el parte de la guardia estudiantil que desea consultar y presiona el botón: “Ver Parte”	3.1 El sistema localiza los datos relativos al parte seleccionado y se la muestra a la Secretaria del Vicedecano, brindando la opción de generar un documento PDF con dicha información.

4. La Secretaria del Vicedecano desea obtener los datos del parte seleccionado en un documento PDF, por lo que presiona el botón: "Ver como PDF"	4.1 El sistema genera un documento PDF que contiene la información relativa al parte seleccionado y se lo muestra a la Secretaria del Vicedecano y termina el CUS.
Curso alterno de los eventos	
Acción 2.1	Si la brigada seleccionada no posee ningún parte en la Base de Datos el sistema muestra un mensaje de error e indica a la Secretaria del Vicedecano retornar a la acción 2.
Sección 2: Buscar Parte por Fecha	
1. La Secretaria del Vicedecano selecciona la opción Buscar parte por fecha.	1.1 El sistema muestra un formulario para la selección de la fecha de la que se desea consultar el parte de la guardia estudiantil.
2. La Secretaria del Vicedecano selecciona la fecha de la cual desea consultar el parte de la guardia estudiantil y presiona el botón "Ver Parte".	2.1 Si en la fecha seleccionada existe algún parte de la guardia estudiantil, el sistema localiza la información relativa a dicho parte y se la muestra a la Secretaria del Vicedecano, brindando la opción de generar un documento PDF con dicha información.
3. La Secretaria del Vicedecano desea obtener los datos del parte seleccionado en un documento PDF, por lo que presiona el botón: "Ver como PDF"	3.1 El sistema genera un documento PDF que contiene la información relativa al parte seleccionado y se lo muestra a la Secretaria del Vicedecano y termina el CUS.
Curso alterno de los eventos	
Acción 2.1	Si en la fecha seleccionada no existe ningún parte de la guardia estudiantil el sistema muestra un mensaje de error e indica a la Secretaria del Vicedecano retornar a la acción

	2.
Prioridad	Crítico

Tabla 2.9 Descripción del caso de uso “Gestionar parte de la guardia estudiantil”

2.16 Conclusiones

Durante este capítulo fueron expuestas las características que contendrá el sistema, apoyándose para ello en el análisis de los actuales procesos de negocio, y habiendo identificado, quiénes son los actores y trabajadores que intervienen en el mismo y con cuáles actividades y entidades interactúan estos. Se identificaron además, los requisitos funcionales y no funcionales que debe cumplir el sistema en cuestión, y con ello fueron expuestos los casos de uso a tratar durante el desarrollo del mismo, con la correspondiente descripción textual de cada uno, lo cual provee de una visión general de qué es lo que el sistema debe hacer, por lo que se está en condiciones de pasar a ver cómo es que el mismo va a realizar las operaciones antes descritas y con ello, darle solución a los problemas planteados.

Capítulo 3

Análisis y Diseño del Sistema

3.1 Introducción

En este capítulo se expone la concepción general referente al análisis y diseño del sistema propuesto, donde el análisis tiene como objetivo mantener un modelo eficiente de la solución propuesta que sirva de base para el diseño. Este último tiene a su vez como objetivo, presentar cómo es que está construido el sistema, lo cual se realiza en este caso a partir de los diagramas de clases Web, que tienen la finalidad de describir la interacción entre las distintas páginas de la aplicación. Por otro lado se presenta el diagrama de clases persistentes de la base de datos del sistema, además de los principios de diseño y programación utilizados.

3.2 Diagrama de clases del análisis

En el análisis se presentan los siguientes estereotipos de clases:

Clase de frontera o interfaz: Modela la interfaz del sistema, y manejan la comunicación entre el entorno y el interior del mismo. Durante el diseño, estas clases son refinadas para tomar en consideración los mecanismos de interfaz seleccionados o implementados, además de facilitar la comunicación con otros sistemas, etc.

Clases de entidad o sistema: Representan la información manejada en el caso de uso, además de que modelan información y comportamiento asociado que generalmente es de larga duración. Reflejan entidades del mundo real, que resultan necesarias para realizar tareas internas del sistema.

Clases de control o software: Coordinan los eventos necesarios para la realización o especificación del caso de uso, con otras palabras, son las que ejecutan el caso de uso. Usualmente son dependientes de la aplicación, además de tener un control sobre todas las acciones a realizar.

A continuación se muestran los diagramas de clases del análisis para cada uno de los casos de uso del sistema:

3.2.1 Diagrama de clases de análisis del caso de uso "Permitir Autenticarse"

Diagrama de clases de análisis del caso de uso "Permitir Autenticarse"

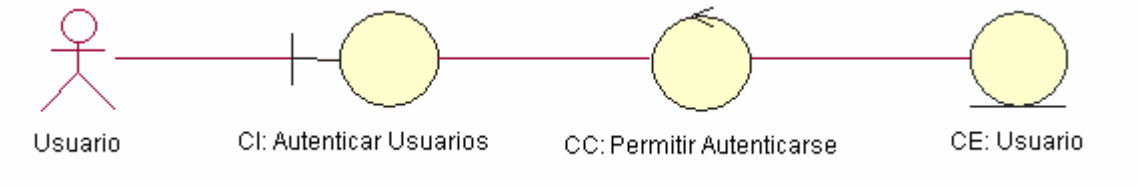


Figura 3.1 Diagrama de clases de análisis del caso de uso "Permitir Autenticarse"

3.2.2 Diagrama de clases de análisis del caso de uso "Gestionar Edificios"

Diagrama de clases de análisis del caso de uso "Gestionar Edificios"

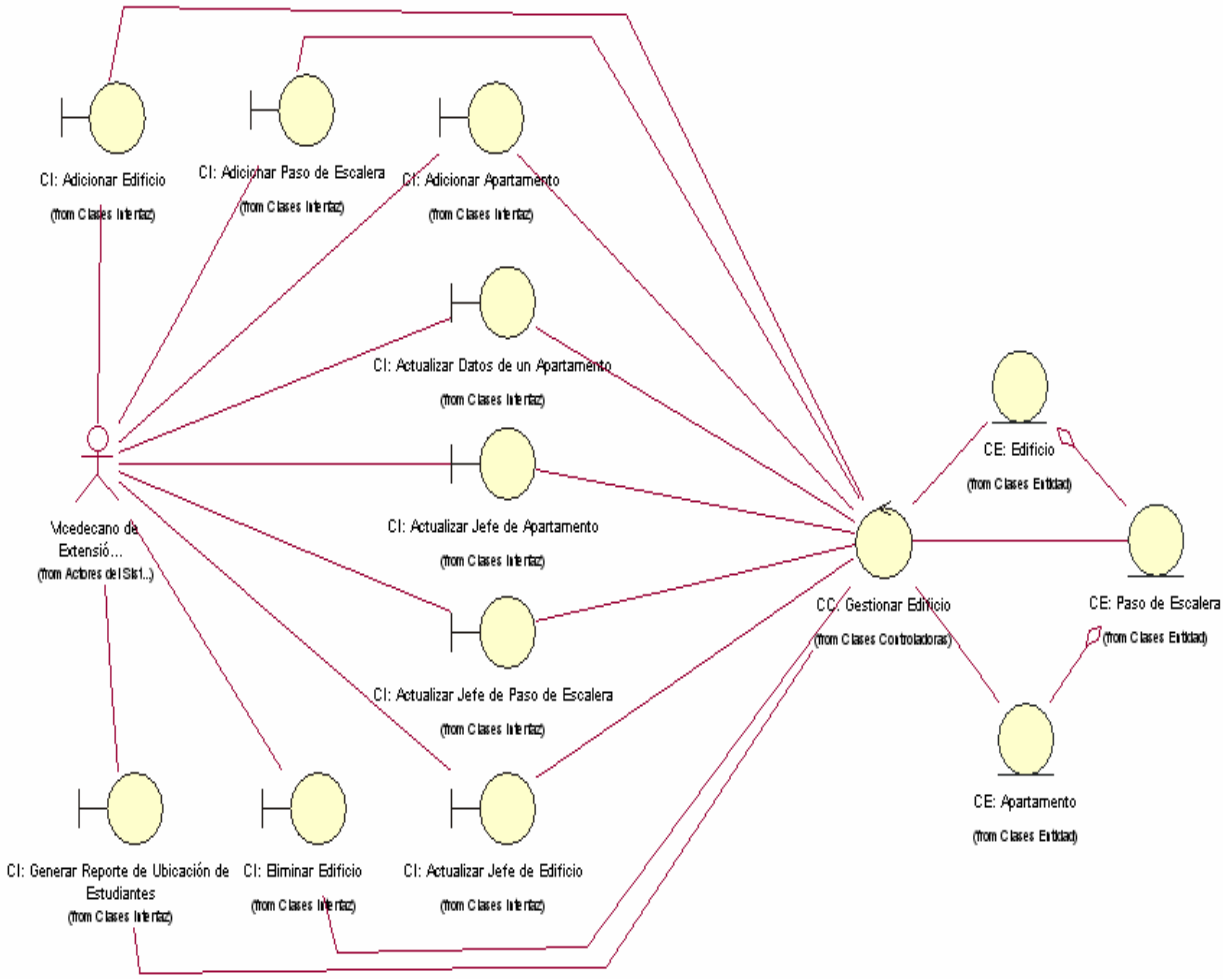


Figura 3.2 Diagrama de clases de análisis del caso de uso "Gestionar Edificios"

3.2.3 Diagrama de clases de análisis del caso de uso "Gestionar Ubicación de Estudiantes en la Residencia"

Diagrama de clases de análisis del caso de uso "Gestionar Ubicación de Estudiantes en la Residencia"

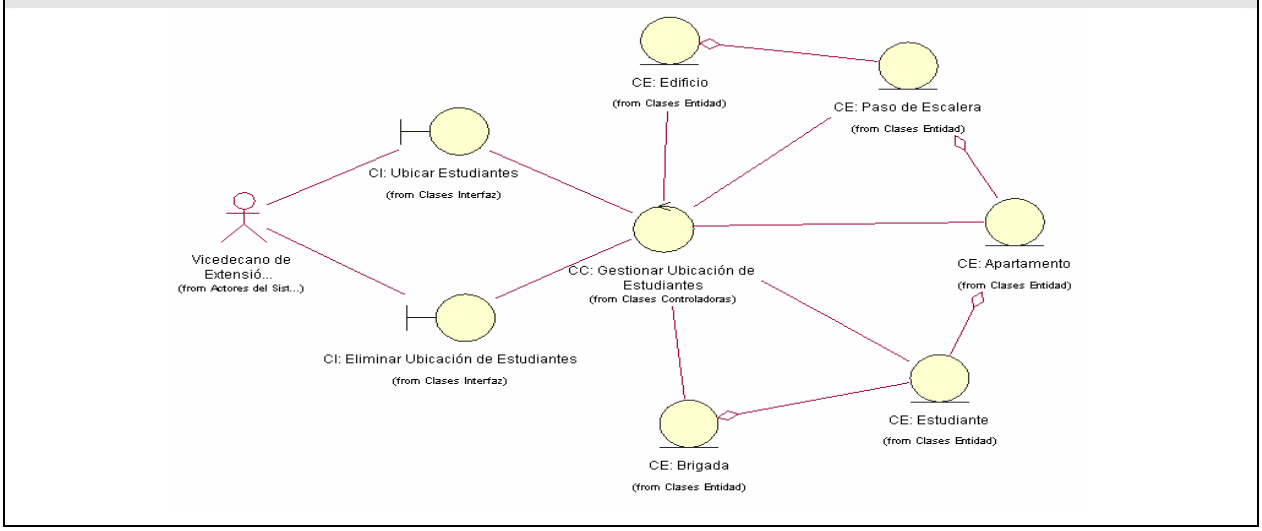


Figura 3.3 Diagrama de clases de análisis del caso de uso "Gestionar Ubicación de Estudiantes en la Residencia"

3.2.4 Diagrama de clases de análisis del caso de uso "Gestionar Parte de la Guardia Estudiantil"

Diagrama de clases de análisis del caso de uso "Gestionar Parte de la Guardia Estudiantil"

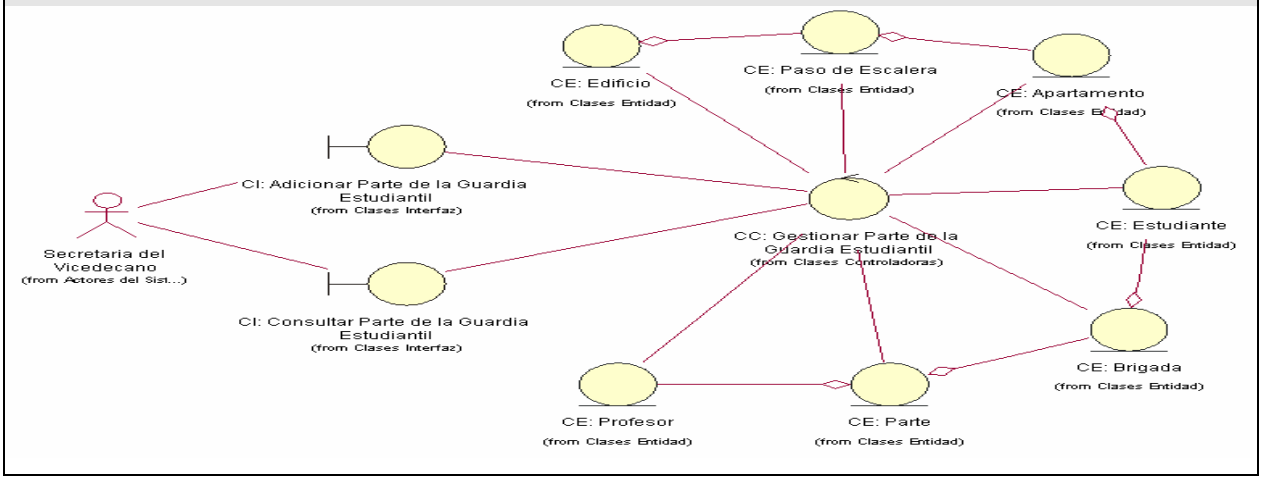


Figura 3.4 Diagrama de clases de análisis del caso de uso "Gestionar Parte de la Guardia Estudiantil"

3.3 Diagrama de clases del Diseño

Un diagrama de clases presenta las clases del sistema con sus relaciones estructurales y de herencia. En el caso de las aplicaciones Web, el diagrama de clases representa las colaboraciones que ocurren entre las páginas, donde cada pagina lógica puede ser representada como una clase.

A continuación se muestran los diagramas de clases Web para cada uno de los casos de uso del sistema:

3.3.1 Diagrama de clases Web para el caso de uso "Permitir Autenticarse"

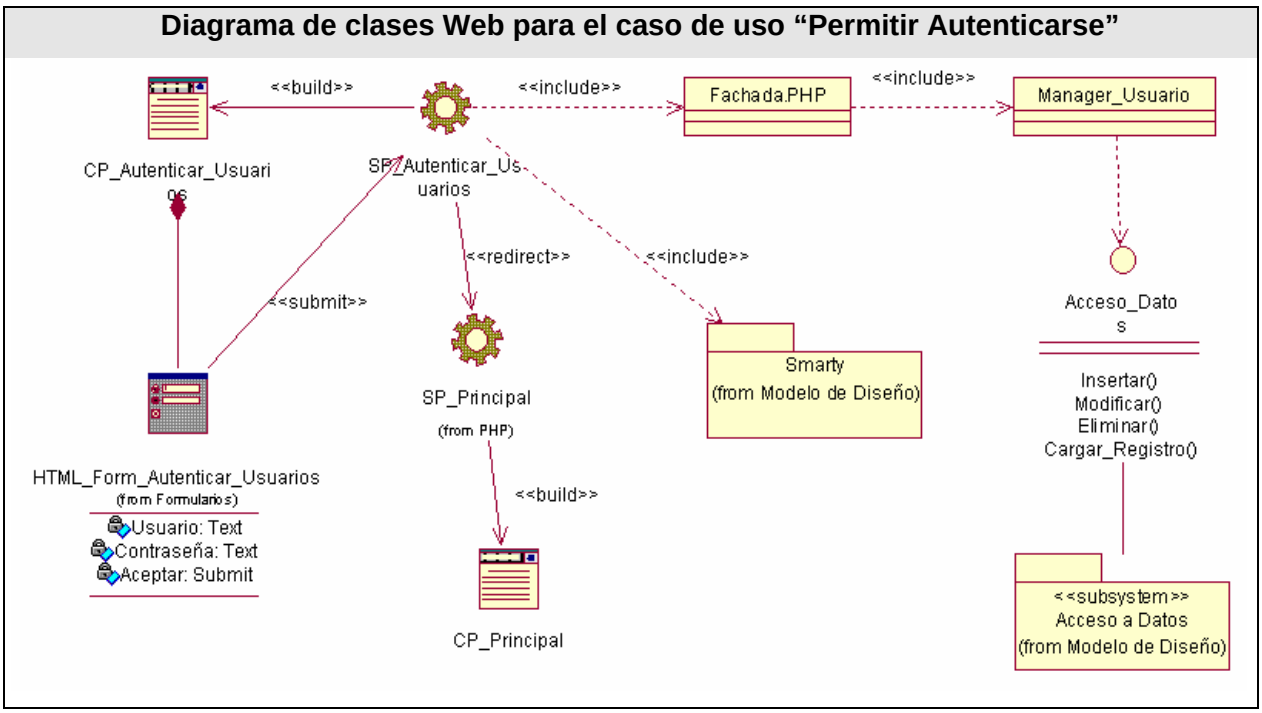


Figura 3.5 Diagrama de clases Web para el caso de uso "Permitir Autenticarse"

3.3.2 Diagrama de clases Web para el caso de uso "Gestionar Edificios"

Un caso de uso puede ser dividido en **escenarios,** los cuales representan funcionalidades específicas dentro del mismo y se utilizan a menudo cuando se tienen casos de uso que poseen varias funcionalidades, para lograr con ello una mayor comprensión de todas las actividades y procesos que este engloba.

Debido a que el caso de uso Gestionar Edificios posee varios escenarios a continuación se muestra el diagrama de clases Web para el escenario "Adicionar Edificio", el resto de los mismos se pueden encontrar en el *Anexo 3*.

3.3.2.1 Diagrama de clases Web para el escenario "Adicionar Edificio"

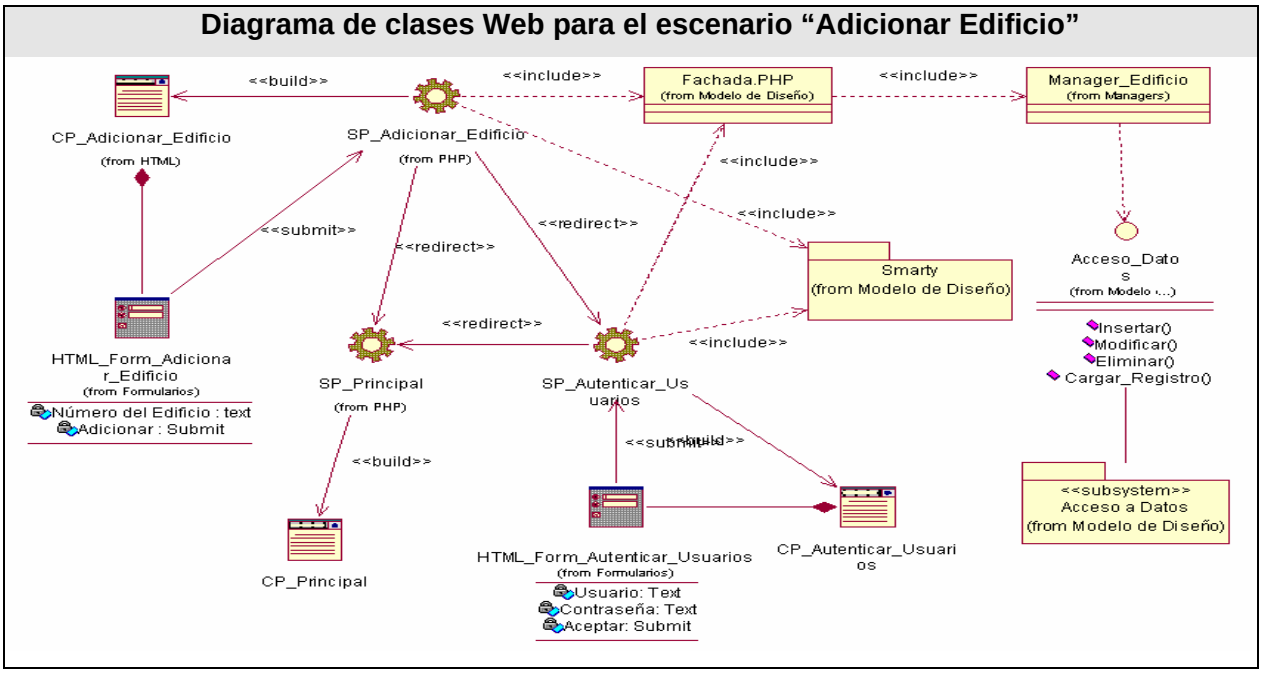


Figura 3.6 Diagrama de clases Web para el escenario "Adicionar Edificio"

3.3.3 Diagrama de clases Web para el caso de uso "Gestionar Ubicación de Estudiantes"

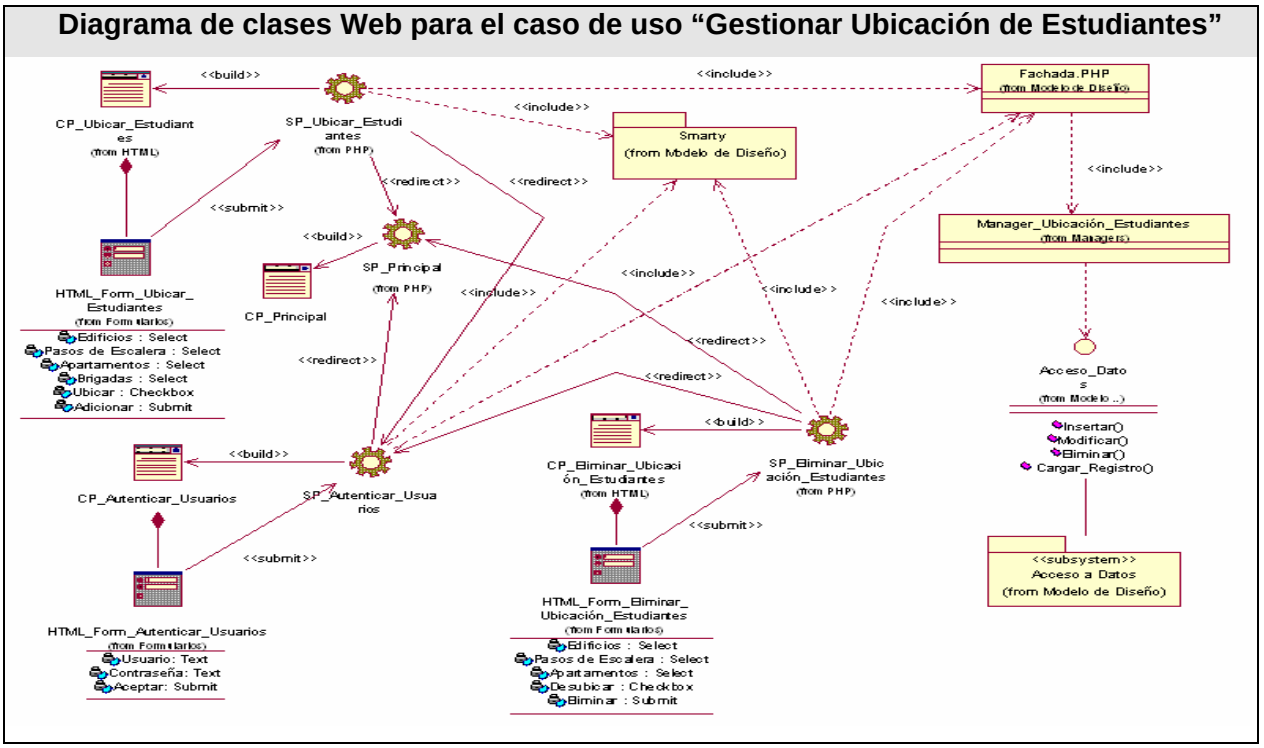


Figura 3.7 Diagrama de clases Web para el caso de uso "Gestionar Ubicación de Estudiantes".

3.3.4 Diagrama de clases Web para el caso de uso "Gestionar Parte de la Guardia Estudiantil"

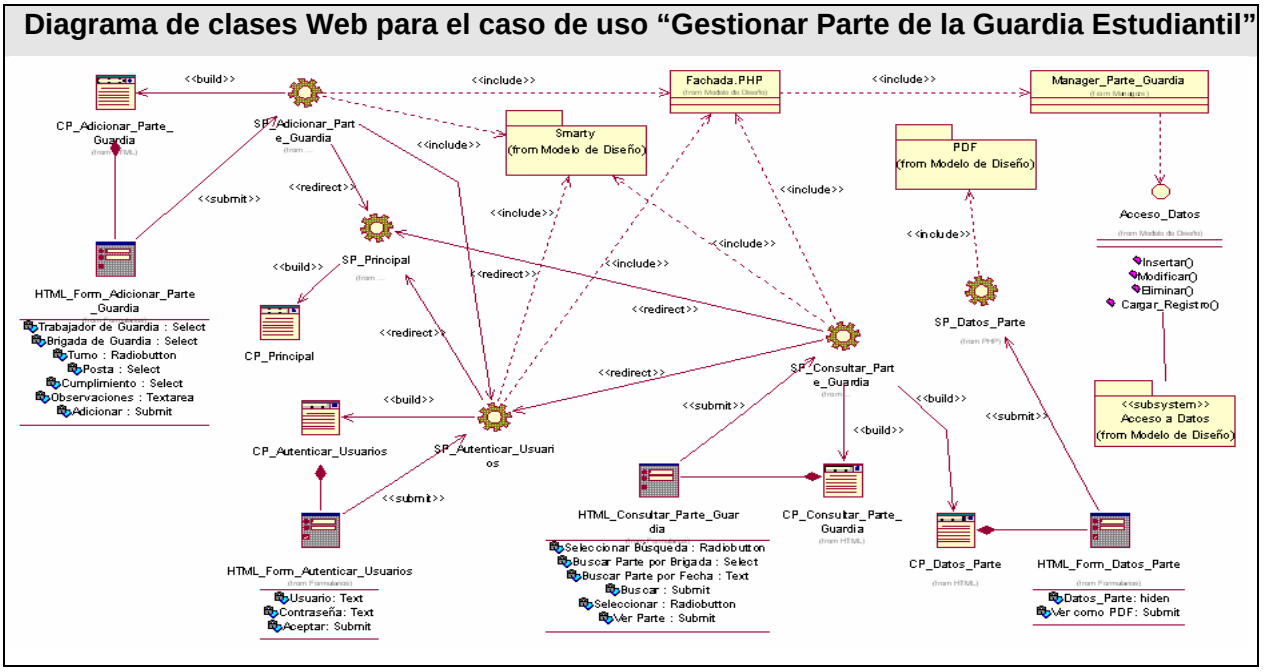


Figura 3.8 Diagrama de clases Web para el caso de uso "Gestionar Parte de la Guardia Estudiantil"

3.3.5 Diagrama de clases del subsistema de Acceso a Datos

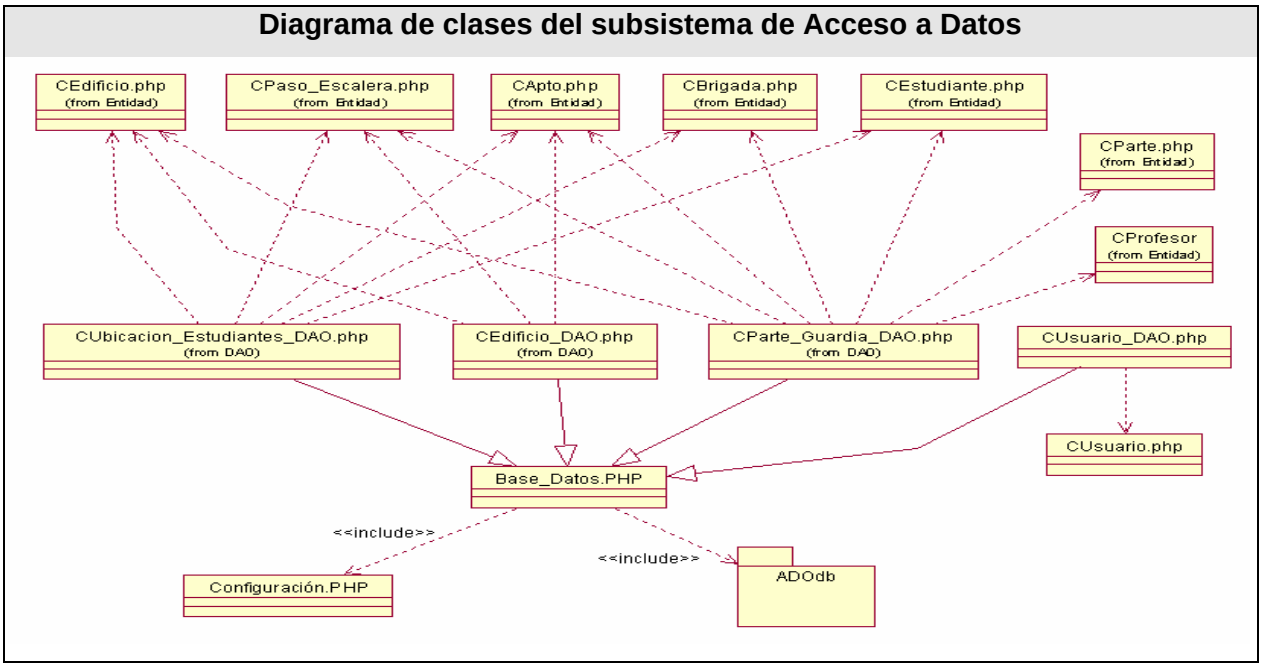


Figura 3.9 Diagrama de clases del subsistema de Acceso a Datos.

3.4 Descripción textual de las clases Web

La descripción textual de las clases del diseño ayudan a comprender el alcance y la responsabilidad de cada una de estas dentro del sistema, es decir, con dicha descripción se puede conocer qué funcionalidad específica realiza cada clase, además de la información que cada una de esta maneja.

3.4.1 Clase Controladora CManager_Edificio

Nombre	CManager_Edificio	
Tipo de Clase	Controladora	
Atributo		**Tipo**
Responsabilidad		
Nombre	Insertar _ Edificio (numero)	
	Existe _ Edificio (numero)	
	Listar _ Edificios ()	
	Cantidad_Pasos_Escalera (numero)	
	Insertar_Paso_Escalera(NumEdif, NumPasoEsc)	
	Cantidad_Aptos_Paso_Escalera(NumEdif, NumPasoEsc)	
	Insertar_Apto(NumEdif, NumPasoEsc, NumApto, Capacidad, Teléfono, Sexo)	
	Existe_Telefono(telef)	
	Datos_Apto(NumEdif, NumPasoEsc, NumApto)	
	Cantidad_Estudiantes_Apto(NumEdif, NumPasoEsc, NumApto)	
	Actualizar_Datos_Apto(NumEdif, NumPasoEsc, NumApto, Capac, NumTelef, Sexo)	
	Eliminar_Edificio(numero)	
	Buscar_Jefe_Edificio(numero)	
	Estudiantes_Edificio(numero)	
	Actualizar_Jefe_Edificio(num_edif, identif_estud)	
	Puede_ser_Jefe(identif_estud)	
	Buscar_Jefe_Paso_Escalera(NumEdif, NumPasoEsc)	
	Estudiantes_Paso_Escalera(NumEdif, NumPasoEsc)	

	Actualizar_Jefe_Paso_Escalera(NumEdif, NumPasoEsc, identif_estud)
	Buscar_Jefe_Apto(NumEdif, NumPasoEsc, NumApto)
	Estudiante_Apto(NumEdif, NumPasoEsc, NumApto)
	Actualizar_Jefe_Apto(NumEdif, NumPasoEsc, NumApto, identif_estud)
Descripción	Controla todo lo referente al manejo de los edificios con sus pasos de escalera y apartamentos.

Tabla 3.1 Descripción de la clase controladora "CManager_Edificio"

3.4.2 Clase Controladora CManager_Ubicacion_Estudiantes

Nombre	CManager_Ubicacion_Estudiantes	
Tipo de Clase	Controladora	
Atributo		**Tipo**
Responsabilidad		
Nombre	Capacidad_Ubicados_Edificio ()	
	Capacidad_Ubicados_PasoEsc (NumEdif)	
	Capacidad_Ubicados_Apto (NumEdif, NumPasoEsc)	
	Cantidad_Ubicados_Brigada (Sexo)	
	Listado_Brigada_Sexo_Desubicados (Id_Brigada, Sexo)	
	Actualizar_Ubicacion_Estudiantes (Id_Brigada, ListaEstuiantes, NumEdif, NumPasoEsc, NumApto)	
	Estudiantes_Ubicados_Apto (NumEdif, NumPasoEsc, NumApto)	
	Eliminar_Ubicacion_Edificio_Completo (NumEdif)	
	Eliminar_Ubicacion_PasoEsc_Completo (NumEdif, NumPasoEsc)	
	Eliminar_Ubicacion_Apto_Completo (NumEdif, NumPasoEsc, NumApto)	
Descripción	Controla todo lo referente al proceso de ubicación de estudiantes en la residencia, así como eliminar la ubicación de los mismos.	

Tabla 3.2 Descripción de la clase controladora "CManager_Ubicacion_Estudiantes"

3.4.3 Clase Controladora CManager_Parte_Guardia

Nombre	CManager_Parte_Guardia
Tipo de Clase	Controladora

Atributo	**Tipo**
Responsabilidad	
Nombre	Listado_Brigadas()
	Listado_Profesores()
	Datos_JefeBrigada_ProfesorGuia(brigada)
	Listado_Estudiantes_Brigada(brigada)
	Listado_Postas()
	Existe_Parte_Guardia(fecha)
	Insertar_Parte_Guardia(fecha, Id_Brigada, Id_Profesor, Observaciones, Listado_Estudiantes, Listado_Turnos, Listado_Postas, Listado_Cumplimientos)
	Buscar_Guardia_Estudiantil_Fecha(fecha)
	Buscar_Guardia_Estudiantil_Brigada(brigada)
	Estudiantes_Guardia_Seleccionada(brigada, Id_guardia)
Descripción	Controla todo lo relacionado con la adición de un nuevo parte de la guardia estudiantil, así como consultar datos de los partes ya existentes.

Tabla 3.3 Descripción de la clase controladora “CManager_Parte_Guardia”

3.4.4 Clase controladora CManager_Usuario

Nombre	CManager_Usuario
Tipo de Clase	Controladora
Atributo	**Tipo**
Responsabilidad	
Nombre	Autenticar_Usuario(usuario, contrasenna)
Descripción	Controla lo referente a la autenticación de los usuarios del sistema.

Tabla 3.4 Descripción de la clase controladora “CManager_Usuario”

3.4.5 Clase controladora CFachada

Nombre	CFachada
Tipo de Clase	Interfaz (PHP, servidor)

Atributo	**Tipo**
gestion_edif	CManager_Edificio
gestion_ubic_estud	CManager_Ubicacion_Estudiantes
gestion_parte_guardia	CManager_Parte_Guardia
gestion_usuario	CManager_Usuario
Responsabilidad	
Nombre	CFachada()
	Insertar _ Edificio (numero)
	Existe _ Edificio (numero)
	Listar _ Edificios ()
	Cantidad_Pasos_Escalera (numero)
	Insertar_Paso_Escalera(NumEdif, NumPasoEsc)
	Cantidad_Aptos_Paso_Escalera(NumEdif, NumPasoEsc)
	Insertar_Apto(NumEdif, NumPasoEsc, NumApto, Capacidad, Teléfono, Sexo)
	Existe_Telefono(telef)
	Datos_Apto(NumEdif, NumPasoEsc, NumApto)
	Cantidad_Estudiantes_Apto(NumEdif, NumPasoEsc, NumApto)
	Actualizar_Datos_Apto(NumEdif, NumPasoEsc, NumApto, Capac, NumTelef, Sexo)
	Eliminar_Edificio(numero)
	Buscar_Jefe_Edificio(numero)
	Estudiantes_Edificio(numero)
	Actualizar_Jefe_Edificio(num_edif, identif_estud)
	Puede_ser_Jefe(identif_estud)
	Buscar_Jefe_Paso_Escalera(NumEdif, NumPasoEsc)
	Estudiantes_Paso_Escalera(NumEdif, NumPasoEsc)
	Actualizar_Jefe_Paso_Escalera(NumEdif, NumPasoEsc, identif_estud)
	Buscar_Jefe_Apto(NumEdif, NumPasoEsc, NumApto)
	Estudiante_Apto(NumEdif, NumPasoEsc, NumApto)
	Actualizar_Jefe_Apto(NumEdif, NumPasoEsc, NumApto, identif_estud)

	Capacidad_Ubicados ()
	Capacidad_Ubicados_PasoEsc (NumEdif)
	Capacidad_Ubicados_Apto (NumEdif, NumPasoEsc)
	Cantidad_Ubicados_Brigada (Sexo)
	Listado_Brigada_Sexo_Desubicados (Id_Brigada, Sexo)
	Brigadas_Desubicados_Sexo (Brigada, Sexo)
	Actualizar_Ubicacion_Estudiantes (Id_Brigada, ListaEstuiantes, NumEdif, NumPasoEsc, NumApto)
	Estudiantes_Ubicados_Apto (NumEdif, NumPasoEsc, NumApto)
	Eliminar_Ubicacion_Edificio_Completo (NumEdif)
	Eliminar_Ubicacion_PasoEsc_Completo (NumEdif, NumPasoEsc)
	Eliminar_Ubicacion_Apto_Completo (NumEdif, NumPasoEsc, NumApto)
	Listado_Brigadas()
	Listado_Profesores()
	Datos_JefeBrigada_ProfesorGuia(brigada)
	Listado_Estudiantes_Brigada(brigada)
	Listado_Postas()
	Existe_Parte_Guardia(fecha)
	Insertar_Parte_Guardia(fecha, Id_Brigada, Id_Profesor, Observaciones, Listado_Estudiantes, Listado_Turnos, Listado_Postas, Listado_Cumplimientos)
	Buscar_Guardia_Estudiantil_Fecha(fecha)
	Buscar_Guardia_Estudiantil_Brigada(brigada)
	Estudiantes_Guardia_Seleccionada(brigada, Id_guardia)
	Autenticar_Usuario(usuario, contrasenna)
Descripción	Clase entidad que representa la tabla Trabajador en la base de datos

Tabla 3.5 Descripción de la clase controladora “CFachada”

3.4.6 Clase de acceso a dato Base_Datos

Nombre	Base_Datos	
Tipo de Clase	DAO	
Atributo		**Tipo**

Servidor	String
Usuario	String
Password	String
BaseDatos	String
Conexion	String
Error	String
Ado_DB	ADONewConnection
Responsabilidad	
Nombre	Base_Datos(s=SERVIDOR, u=USUARIO, p=PASSWORD, bd= BASEDATOS, mt_bd=MOTOR_BD)
	Abrir_Conexion ()
	Cerrar_Conexion ()
	Insertar (sql)
	Modificar (sql)
	Eliminar (sql)
	Ejecutar_Consulta(sql)
	Ejecutar_Insercion(sql)
	Cargar_Registro (sql)
	Empezar_Transaccion ()
	Terminar_Transaccion ()
	Ocurrio_Error_Transaccion ()
	Forzar_Transaccion ()
Descripción	Clase encargada de realizar la conexión a la base de datos y ejecutar las consultas, haciendo uso del paquete ADOdb.

Tabla 3.6 Descripción de la clase DAO “Base_Datos”

3.4.7 Clase de acceso a dato CEdificio_DAO

Nombre	CEdificio_DAO
Tipo de Clase	DAO
Atributo	**Tipo**

Responsabilidad	
Nombre	Insertar_Edificio(edif)
	Existe_Edificio(edif)
	Listar_Edificios()
	Cantidad_Pasos_Escalera(edif)
	Insertar_Paso_Escalera(pasoesc)
	Cantidad_Aptos_Paso_Escalera(pasoesc)
	Insertar_Apto(apto)
	Existe_Telefono(telef)
	Datos_Apto(apto)
	Cantidad_Estudiantes_Apto(apto)
	Actualizar_Datos_Apto(apto)
	Eliminar_Edificio(edif)
	Buscar_Jefe_Edificio(edif)
	Estudiantes_Edificio(edif)
	Actualizar_Jefe_Edificio(edif)
	Puede_ser_Jefe(identif_estud)
	Buscar_Jefe_Paso_Escalera(pasoesc)
	Estudiantes_Paso_Escalera(pasoesc)
	Actualizar_Jefe_Paso_Escalera(pasoesc)
	Buscar_Jefe_Apto(apto)
	Estudiante_Apto(apto)
	Actualizar_Jefe_Apto(apto)
Descripción	Clase encargada de realizar todas las consultas a la base de datos referentes al manejo de los edificios, pasos de escalera y apartamentos.

Tabla 3.7 Descripción de la clase DAO “CEdificio_DAO”

3.4.8 Clase de acceso a dato CUbicacion_Estudiantes_DAO

Nombre	CUbicación_Estudiantes_DAO	
Tipo de Clase	DAO	
Atributo		**Tipo**

Responsabilidad	
Nombre	Capacidad_Ubicados_Edificio ()
	Capacidad_Ubicados_PasoEsc (edif)
	Capacidad_Ubicados_Apto (pasoesc)
	Cantidad_Ubicados_Brigada (Sexo)
	Listado_Brigada_Sexo_Desubicados (brigada, Sexo)
	Actualizar_Ubicacion_Estudiantes (brigada, apto)
	Estudiantes_Ubicados_Apto (apto)
	Eliminar_Ubicacion_Edificio_Completo (edif)
	Eliminar_Ubicacion_PasoEsc_Completo (pasoesc)
	Eliminar_Ubicacion_Apto_Completo (apto)
Descripción	Clase encargada de realizar todas las consultas a la base de datos referentes al manejo de la ubicación de los estudiantes en la residencia.

Tabla 3.8 Descripción de la clase DAO “CUbicación_Estudiantes_DAO”

3.4.9 Clase de acceso a dato CParte_Guardia_DAO

Nombre	CParte_Guardia_DAO
Tipo de Clase	DAO
Atributo	**Tipo**
Responsabilidad	
Nombre	Listado_Brigadas()
	Listado_Profesores()
	Datos_JefeBrigada_ProfesorGuia(BrigadaSeleccionada)
	Listado_Estudiantes_Brigada(BrigadaSeleccionada)
	Listado_Postas()
	Existe_Parte_Guardia(parte)
	Insertar_Parte_Guardia(Parte_Guardia, Brigada)
	Buscar_Guardia_Estudiantil_Fecha(parte)
	Buscar_Guardia_Estudiantil_Brigada(BrigadaSeleccionada)
	Estudiantes_Guardia_Seleccionada(BrigadaSeleccionada, Id_guardia)

Descripción	Clase encargada de realizar todas las consultas en la base de datos para el manejo de los partes de la guardia.

Tabla 3.9 Descripción de la clase DAO “CParte_Guardia_DAO”

3.4.10 Clase de acceso a dato CUsuario_DAO

Nombre	CUsuario_DAO	
Tipo de Clase	DAO	
Atributo		**Tipo**
Responsabilidad		
Nombre	Autenticar_Usuario(user)	
Descripción	Clase encargada de verificar en la base de datos si un usuario está autenticado o no	

Tabla 3.10 Descripción de la clase DAO “CUsuario_DAO”

3.4.11 Clase entidad CEdificio

Nombre	CEdificio	
Tipo de Clase	Entidad	
Atributo		**Tipo**
Numero		int
IdentificadorJE		String
Responsabilidad		
Nombre	CEdificio(aNumero,aIdentificador=NULL)	
	Get_Numero ()	
	Get_IdentificadorJE ()	
	Set_IdentificadorJE (aIdentificador)	
Descripción	Clase entidad que representa la tabla edificio en la base de datos	

Tabla 3.11 Descripción de la clase entidad “CEdificio”

3.4.12 Clase entidad CPaso_Escalera

Nombre	CPaso_Escalera
Tipo de Clase	Entidad

Atributo	**Tipo**
NumeroEdificio	int
NumeroPasoEscalera	int
IdentificadorJPE	String
Responsabilidad	
Nombre	CPaso_Escalera(aNumeroEdificio, aNumeroPasoEscalera, aIdentificador=NULL)
	Get_NumeroEdificio ()
	Get_NumeroPasoEscalera ()
	Get_IdentificadorJPE ()
	Set_IdentificadorJPE (aIdentificador)
Descripción	Clase entidad que representa la tabla paso_escalera en la base de datos

Tabla 3.12 Descripción de la clase entidad “CPaso_Escalera”

3.4.13 Clase entidad CApto

Nombre	CApto	
Tipo de Clase	Entidad	
Atributo		**Tipo**
NumeroEdificio		int
NumeroPasoEscalera		int
NumeroApto		int
Capacidad		int
Teléfono		int
Sexo		char
IdentificadorJA		String
Responsabilidad		
Nombre	CApto(aNumeroEdificio, aNumeroPasoEscalera, aNumeroApto, aCapacidad, aTelefono, aSexo, aIdentificador=NULL)	
	Get_NumeroEdificio ()	
	Get_NumeroPasoEscalera ()	
	Get_NumeroApto ()	
	Get_Capacidad ()	

	Get_Telefono ()
	Get_Sexo ()
	Get_IdentificadorJA ()
	Set_Capacidad (aCapacidad)
	Set_Telefono (aTelefono)
	Set_Sexo (aSexo)
	Set_IdentificadorJA (aIdentificador)
Descripción	Clase entidad que representa la tabla apto en la base de datos

Tabla 3.13 Descripción de la clase entidad "CApto"

3.4.14 Clase entidad CProfesor

Nombre	CProfesor	
Tipo de Clase	Entidad	
Atributo		**Tipo**
Identificador		String
Responsabilidad		
Nombre	CProfesor (aIdentificador, aCumplimiento)	
	Get_Identificador ()	
Descripción	Clase entidad que representa la tabla Profesor en la base de datos	

Tabla 3.14 Descripción de la clase entidad "CProfesor"

3.4.15 Clase entidad CEstudiante

Nombre	CEstudiante	
Tipo de Clase	Entidad	
Atributo		**Tipo**
Identificador		String
TurnoGuardia		int
Posta		int
Cumplimiento		String
Responsabilidad		
Nombre	CEstudiante (aIdentificador, aTurnoGuardia, aPosta, aCumplimiento)	
	Get_Identificador ()	

	Get_TurnoGuardia ()
	Get_Posta ()
	Get_Cumplimiento ()
Descripción	Clase entidad que representa la tabla Guardia_Estudiante en la base de datos

Tabla 3.15 Descripción de la clase entidad “CEstudiante”

3.4.16 Clase entidad CBrigada

Nombre	CBrigada	
Tipo de Clase	Entidad	
Atributo		**Tipo**
Identificador		String
ListaEstudiantes		CEstudiante
Responsabilidad		
Nombre	CBrigada (aIdentificador)	
	Get_Identificador ()	
	Add_Estudiante (CEstudiante)	
	Obtener_Estudiante (int)	
	Get_ListadoEstudiantes()	
Descripción	Clase entidad que contiene un listado con los estudiantes que pertenecen a una brigada.	

Tabla 3.16 Descripción de la clase entidad “CBrigada”

3.4.17 Clase entidad CParteGuardia

Nombre	CParteGuardia	
Tipo de Clase	Entidad	
Atributo		**Tipo**
Fecha		Date
Brigada		CBrigada
Profesor		CProfesor
Observaciones		String
Responsabilidad		

Nombre	CParteGuardia(aFecha, aBrigada, aProfesor, aObservaciones)
	Get_Fecha ()
	Get_Brigada ()
	Get_Profesor ()
	Get_Observaciones ()
Descripción	Clase entidad que contiene toda la información referente a un parte de la guardia estudiantil.

Tabla 3.17 Descripción de la clase entidad “CParteGuardia”

3.4.18 Clase entidad CUsuario

Nombre	CUsuario	
Tipo de Clase	Entidad	
Atributo		**Tipo**
Usuario		String
Contrasenna		String
Rol		int
Responsabilidad		
Nombre	CUsuario(ausuario, acontrasenna, arol= ‘null’)	
	Get_Usuario ()	
	Get_Contrasenna ()	
	Get_Rol ()	
Descripción	Clase entidad que representa a la tabla usuario en la base de datos	

Tabla 3.18 Descripción de la clase entidad “CUsuario”

3.5 Diagrama de clases persistentes

Las clases persistentes son aquellas que necesitan ser capaz de guardar su estado en un medio permanente; lo cual está dado por el almacenamiento físico de la información de la clase, para la copia de seguridad en caso del fracaso del sistema, o para el intercambio de información.

A continuación se muestra el diagrama de clases persistentes del sistema:

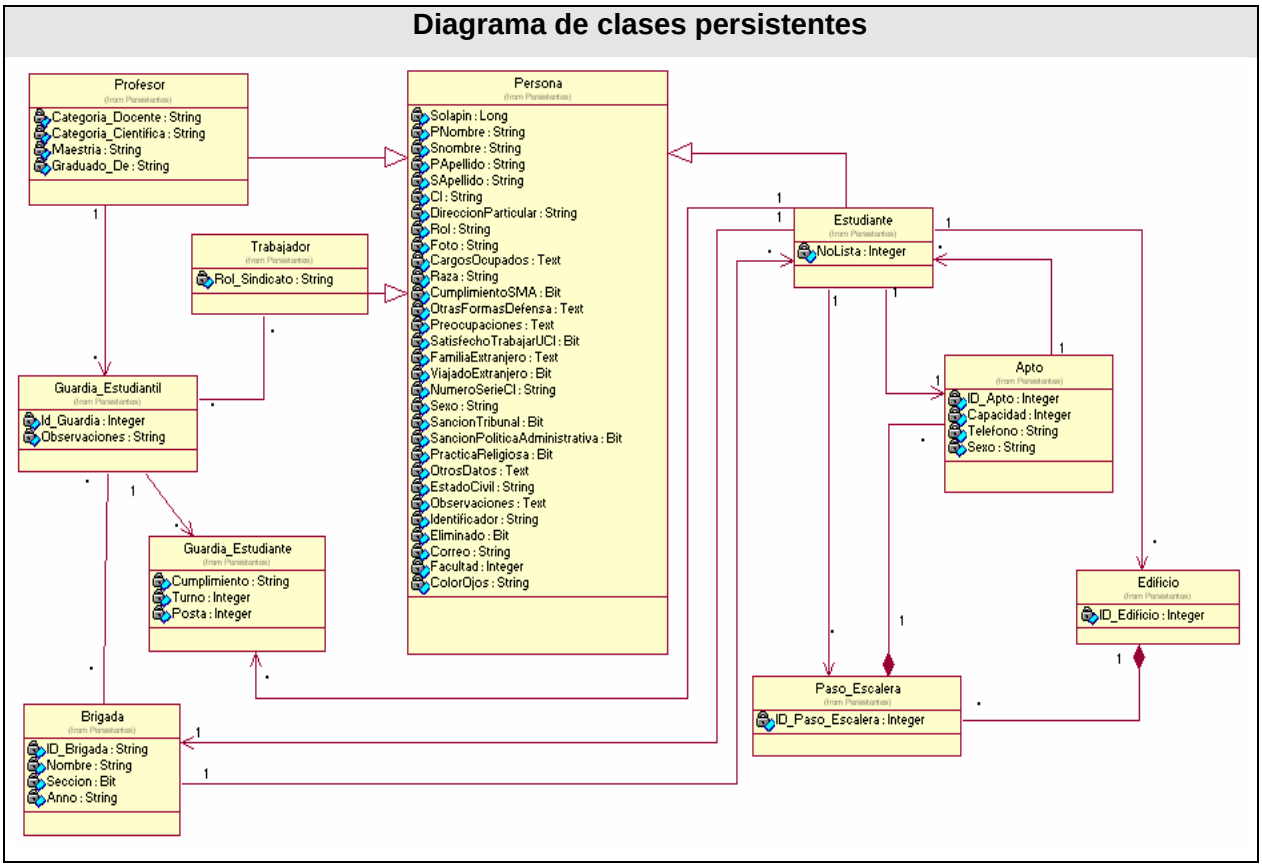


Figura 3.10 Diagrama de clases persistentes.

3.6 Principios de Diseño

Tanto el diseño de la interfaz de un sistema, como el formato de los reportes, la concepción de la ayuda y el tratamiento de excepciones tienen gran influencia en el éxito o fracaso del mismo.

A continuación se describen los principios de diseño seguidos para el desarrollo de dicho sistema.

3.6.1 Interfaz de Usuario

El diseño de la interfaz de usuario se puede definir como: “el conjunto de trabajos y pasos que seguirá el usuario, durante todo el tiempo que se relacione con el programa, detallando lo que verá y escuchará en cada momento, y las acciones que realizará, así como las respuestas que el sistema dará”. [4]

Para el diseño de la interfaz se tuvieron en cuenta los ocho principios básicos existentes, los cuales garantizan la usabilidad en los diseños para aplicaciones Web:

1. Búsqueda estable.

2. Proporcionar atajos a Usuarios expertos.
3. Ofrecer información de retroalimentación.
4. Diseñar diálogos que conduzcan a una conclusión.
5. Prever errores y manejar errores simples.
6. Permitir deshacer acciones fácilmente.
7. Favorecer la sensación de control.
8. Reducir la carga a la memoria de corto plazo.

Por otro lado, el diseño de todo sistema, debe centrarse en el usuario que va a hacer uso del mismo, que es quien determina el éxito o fracaso de este, por lo cual, además de los principios antes mencionados, también se tuvo en cuenta lo siguiente:

- La utilización de un mismo formato y estilo en cada una de las páginas.
- Facilitarle al usuario la plena navegabilidad dentro de la aplicación.
- Evitar sobrecarga de colores e imágenes.
- Proporcionar un ambiente amigable.

Se utilizó también, una hoja de estilo para guardar la configuración del diseño de todas las páginas del sistema, tanto para los botones como para el tipo y tamaño de letra, logrando una uniformidad en todas estas. Por otra parte, todas las páginas contienen el mismo menú en la parte izquierda garantizando con ello que se pueda acceder en todo momento a cada una de las opciones que brinda el sistema, lo cual propicia una alta flexibilidad en cuanto a la navegabilidad. Por otro lado, los formularios de entrada están centrados en la parte destinada para ello dentro de cada página, además de estar organizados según la prioridad de los datos que se necesitan, ver figura 3.11.

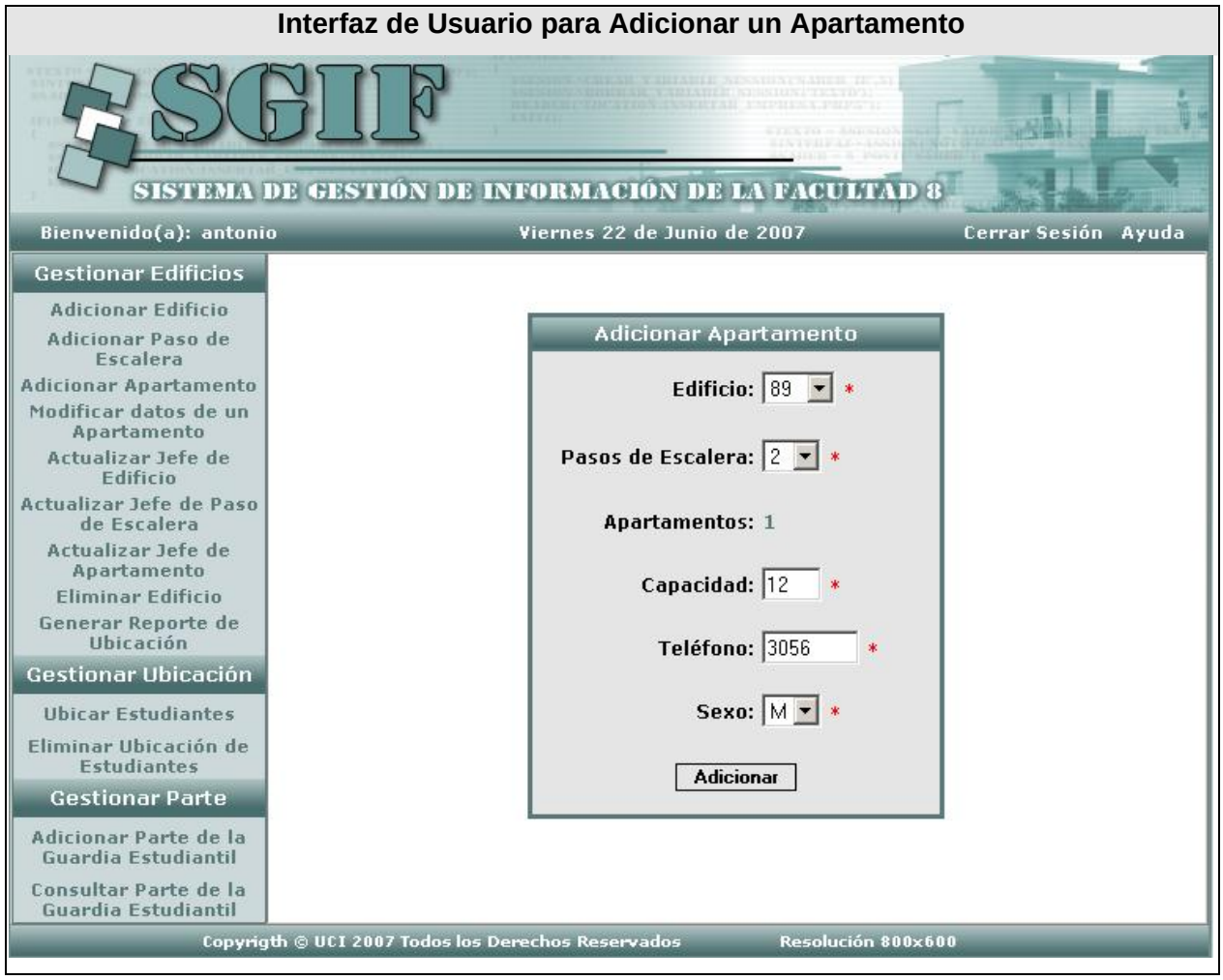
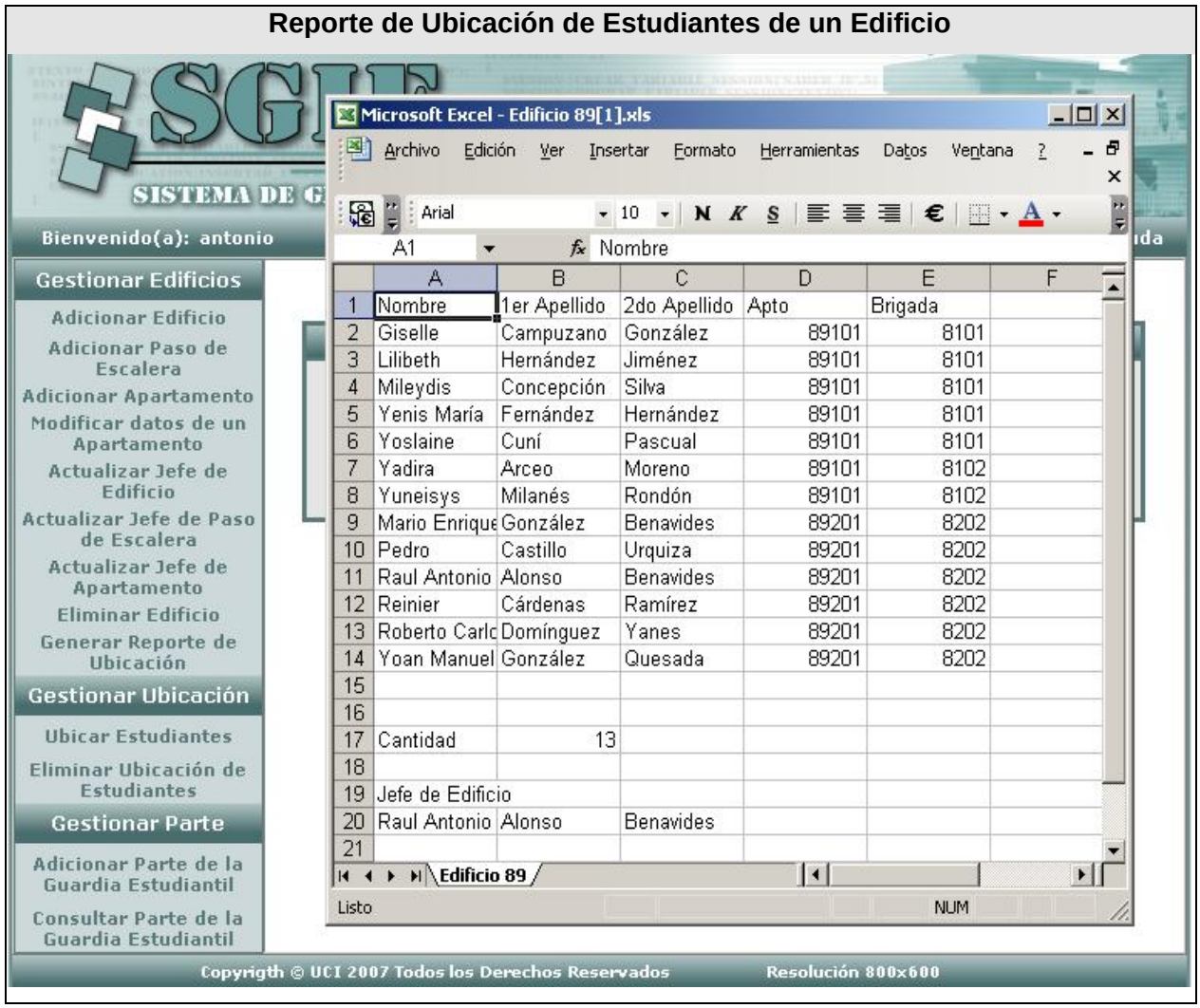


Figura 3.11 Interfaz de usuario para Adicionar un Apartamento.

3.6.2 Formato de salida de los Reportes

Los reportes generados contienen un formato estándar, por cuanto son visualizados a través de un documento Excel que genera el sistema, donde la información que está contenida en los mismos se encuentra organizada, facilitando con ello la búsqueda de algún dato en particular, ver figura 3.12.



	A	B	C	D	E	F
1	Nombre	1er Apellido	2do Apellido	Apto	Brigada	
2	Giselle	Campuzano	González	89101	8101	
3	Lilibeth	Hernández	Jiménez	89101	8101	
4	Mileydis	Concepción	Silva	89101	8101	
5	Yenis María	Fernández	Hernández	89101	8101	
6	Yoslaine	Cuní	Pascual	89101	8101	
7	Yadira	Arceo	Moreno	89101	8102	
8	Yuneisys	Milanés	Rondón	89101	8102	
9	Mario Enrique	González	Benavides	89201	8202	
10	Pedro	Castillo	Urquiza	89201	8202	
11	Raul Antonio	Alonso	Benavides	89201	8202	
12	Reinier	Cárdenas	Ramírez	89201	8202	
13	Roberto Carlo	Domínguez	Yanes	89201	8202	
14	Yoan Manuel	González	Quesada	89201	8202	
15						
16						
17	Cantidad	13				
18						
19	Jefe de Edificio					
20	Raul Antonio	Alonso	Benavides			
21						

Figura 3.12 Reporte de Ubicación de Estudiantes de un Edificio.

3.6.3 Ayuda

La ayuda estará accesible siempre en cada una de las páginas de la aplicación y mostrará solo aquella información que es de interés conocer al usuario, según la página en la que se encuentre el mismo en ese momento. Por otro lado, en cada página de ayuda se brindará información sobre qué es lo que se debe hacer para realizar cada una de las operaciones con las que el usuario esté interactuando en ese momento, así como una explicación de los posibles mensajes de error que puede darle el sistema ante cualquier acción que se ejecute, ver figura 3.13

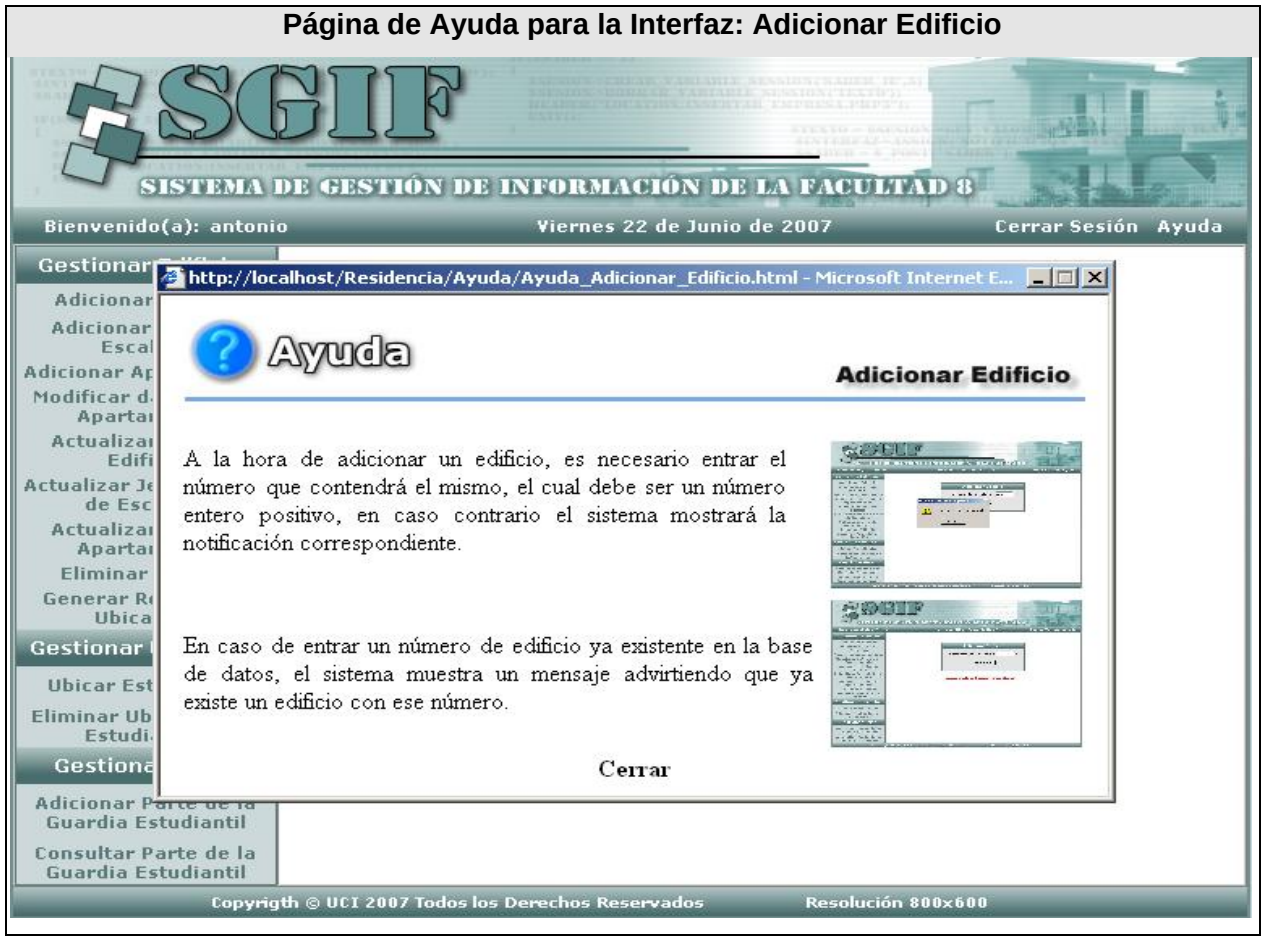


Figura 3.13 Página de ayuda para la interfaz: Adicionar Edificio.

3.6.4 Tratamiento de errores

El correcto tratamiento de los errores en un sistema, influye notablemente en el buen funcionamiento de este, a la vez que se garantiza la integridad de la información. Para lograr esto, fueron previstos todos los errores posibles que pudiera generar el sistema a partir de la interacción del usuario con este, así como de problemas inesperados que pudieran surgir, como es el caso de la pérdida de la conexión con la base de datos por falta del fluido eléctrico, etc. Muchos de estos posibles errores son tratados a partir de funciones JavaScript, y otros se obtienen a partir de la ejecución de la página en el servidor de la aplicación, estos últimos aparecen de forma textual en la misma página en la que se encuentra el usuario en ese momento. En cualquiera de los dos casos, los mensajes de error son mostrados con un texto claro, que señala de forma explícita y legible la respuesta del sistema ante cualquier acción que se ejecute. Además de esto, se muestran mensajes de confirmación ante acciones que son

irreversibles, como es el caso de la eliminación de datos, a la vez que se muestran mensajes para indicar cuándo una acción fue realizada con éxito, ver figura 3.14

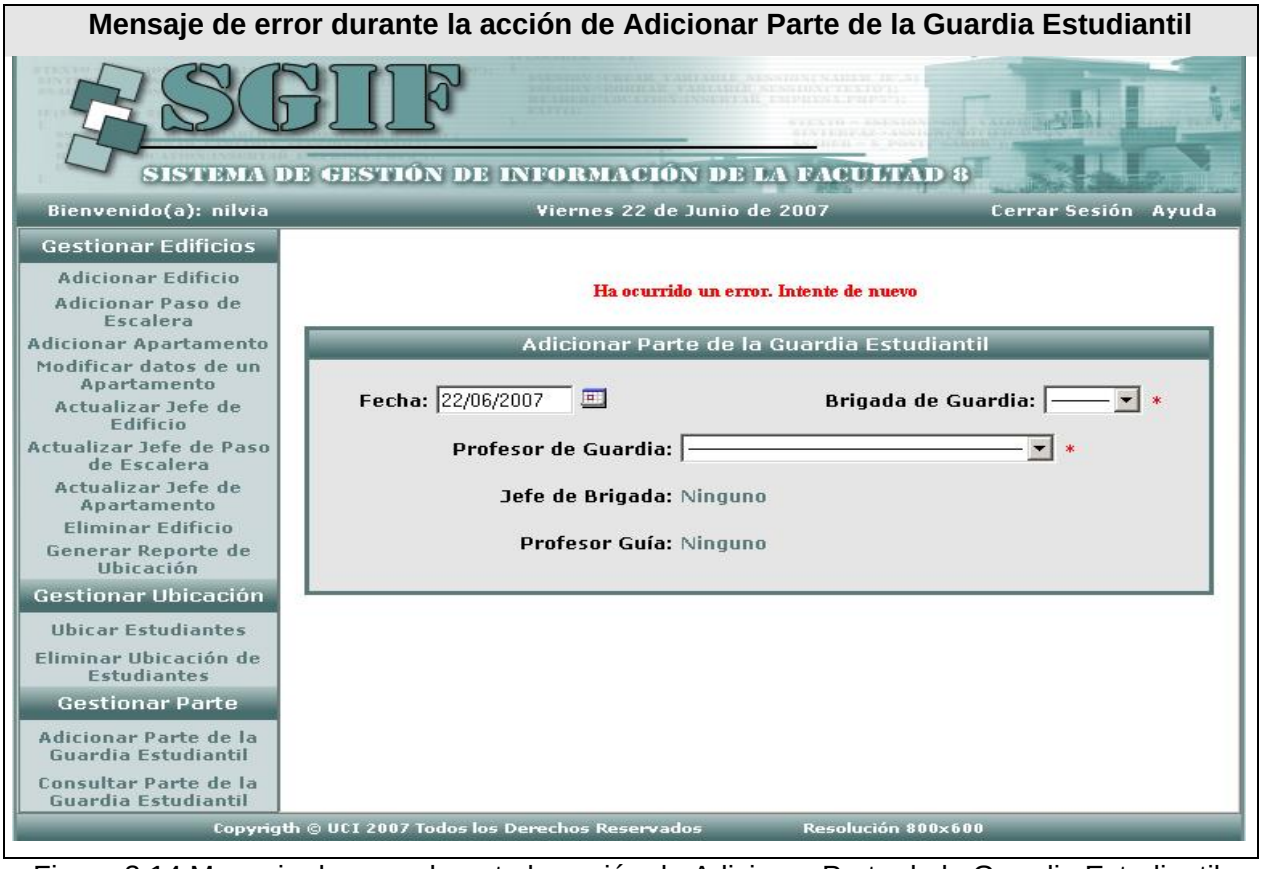


Figura 3.14 Mensaje de error durante la acción de Adicionar Parte de la Guardia Estudiantil.

3.6.5 Seguridad

La seguridad del sistema, se basa principalmente en el empleo de la autenticación como una acciona de obligatorio cumplimiento para el posterior uso del mismo, por lo que cada usuario del sistema tendrá un nombre para identificarse y una contraseña, los cuales serán verificados antes de darle acceso a las funcionalidades de sistema y si alguno de estos datos son incorrectos, se denegará dicho acceso. La contraseña pasa por un proceso de encriptación en el cliente, siguiendo el algoritmo MD5 y posteriormente se verifica en el servidor, si la misma ha sido encriptada o no, para en caso negativo encriptarla y manejarla de esta forma. Por otra parte, cada usuario posee un rol, el cual es chequeado en cada página del sistema, con vista a darle o no acceso a la misma en dependencia de cual sea este rol. También, son validados cada uno de los campos de los formularios con el objetivo de evitar que se introduzcan o

seleccionen datos no permisibles por el sistema. Además, se prevén acciones tales como la pérdida de conexión con la base de datos, o la incorrecta ejecución de alguna consulta, a la vez que se prevé que una acción de inserción de determinada información en varias tablas de la base de datos se realice incorrectamente debido a que si alguna parte de esta información no se inserta con éxito, esto puede provocar que no se cuente con la información deseada posteriormente.

Mensaje de error durante la acción de Autenticarse

Figura 3.15 Mensaje de error durante la acción de Autenticarse.

3.7 Conclusiones

En este capítulo fueron expuestos diferentes elementos que ilustran cómo está construido el sistema, en términos de clases del análisis y del diseño. Este último dio la posibilidad de comprender la lógica del sistema en general. Por otro lado, fueron detalladas textualmente cada una de las clases Web, lo cual trae consigo que se conozca la responsabilidad de cada una de estas y qué funcionalidad específica realizan, además de que fue presentado el diagrama de clases persistentes de la base de datos, que contiene la información física que se utilizó para la construcción de la aplicación. Por último fueron expuestos los principios de diseño seguidos durante la implementación del sistema, entre los que se encuentran: Interfaz de usuario, Formato de salida de los reportes, Tratamiento de errores, Ayuda y Seguridad.

Capítulo 4

Implementación y Pruebas

4.1 Introducción

En este capítulo se describe cómo los elementos del modelo de diseño son implementados en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue. Además de que se exponen las diferentes pruebas realizadas a cada caso de uso, siguiendo específicamente, el método de pruebas de caja negra.

4.2 Modelo de Implementación

En el modelo de implementación se describe cómo los elementos del diseño, es decir las clases, se implementan en términos de componentes, como son: ficheros de código fuente, ejecutables, etc. Además de que describe como están organizados los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje de programación utilizado, así como la dependencia que existe entre estos componentes.

A continuación se muestran los diferentes diagramas de componentes, asociados a cada caso de uso y al sistema en general:

4.2.1 Diagrama de componentes de la Base de Datos

En el siguiente diagrama se muestra la relación del sistema completo como un componente y la base de datos.

Diagrama de componentes de la Base de Datos

Figura 4.1 Diagrama de componentes de la base de datos

4.2.2 Diagrama de componentes del sistema

El siguiente diagrama muestra la relación que existe entre todos los paquetes de componentes del sistema.

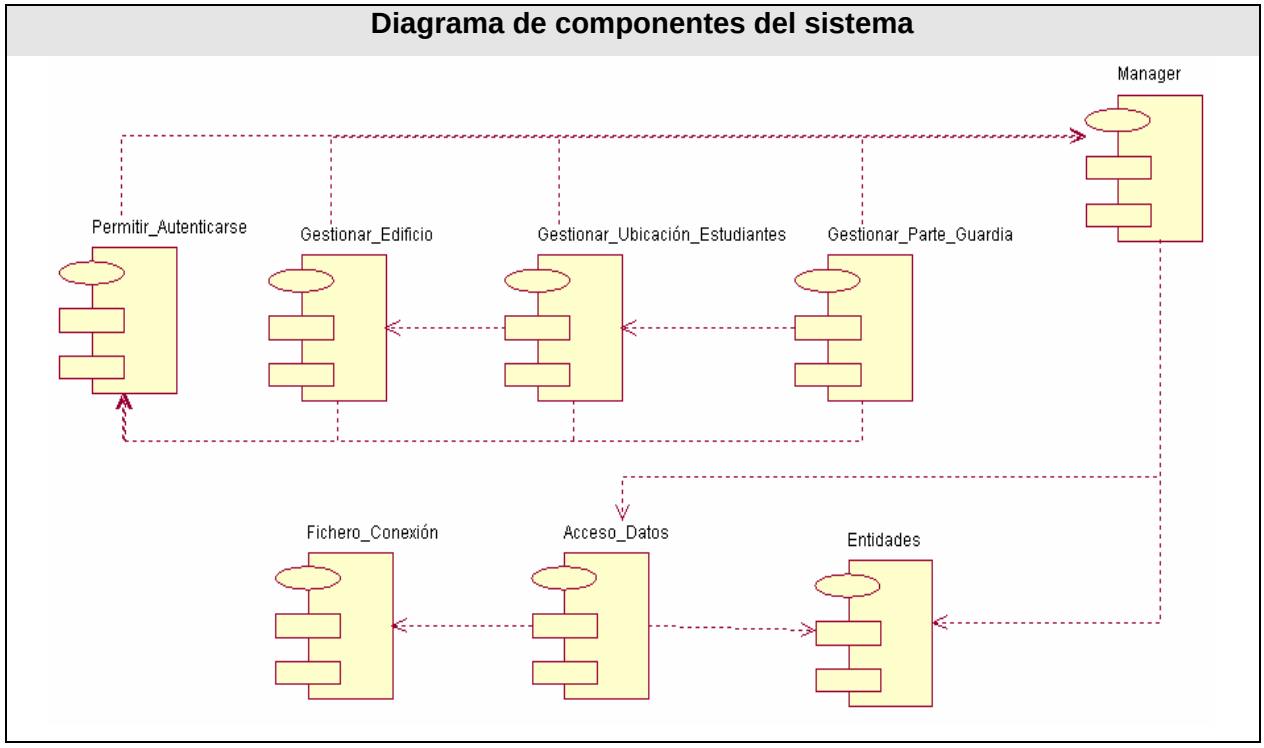


Figura 4.2 Diagrama de componentes del sistema

4.2.3 Diagrama de componentes del caso de uso "Permitir Autenticarse"

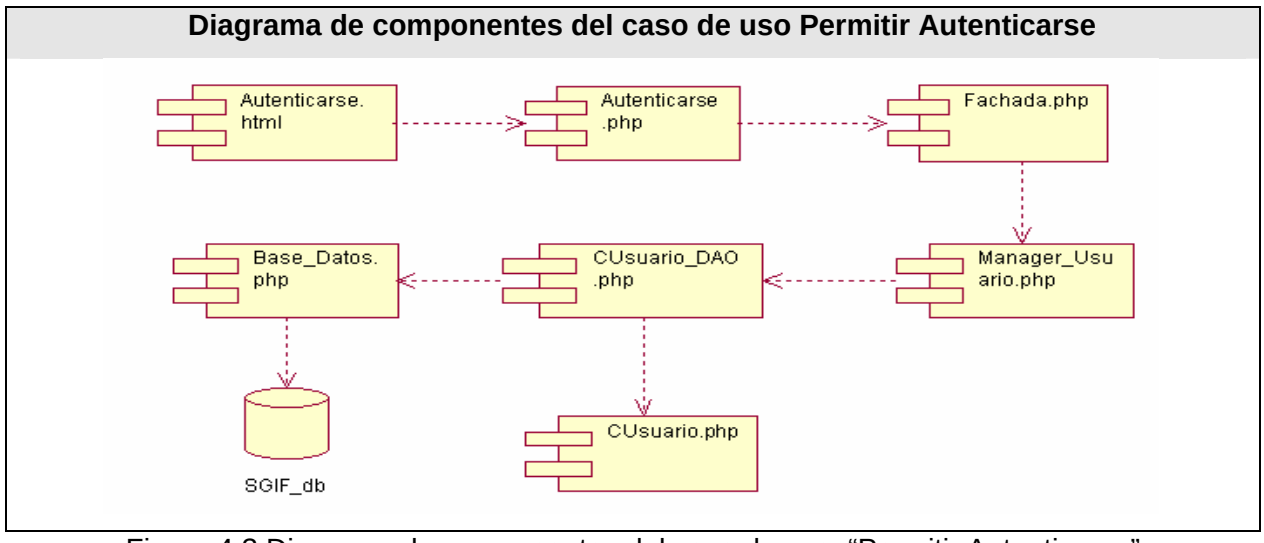


Figura 4.3 Diagrama de componentes del caso de uso: "Permitir Autenticarse"

4.2.4 Diagrama de componentes del caso de uso "Gestionar Parte de la Guardia Estudiantil"

Diagrama de componentes del caso de uso
Gestionar Parte de la Guardia Estudiantil

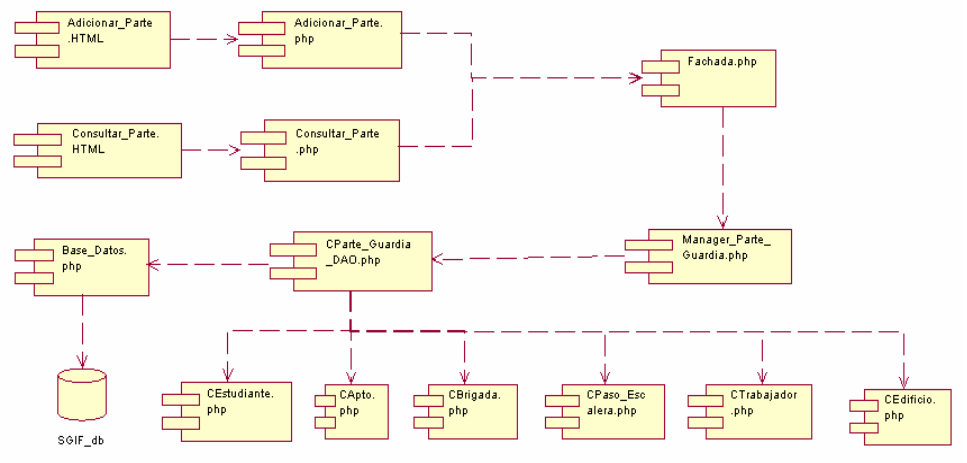


Figura 4.4 Diagrama de componentes del caso de uso: "Gestionar parte de la guardia estudiantil"

4.2.5 Diagrama de componentes del caso de uso "Gestionar Ubicación de Estudiantes en la Residencia"

Diagrama de componentes del caso de uso
Gestionar Ubicación de Estudiantes en la Residencia

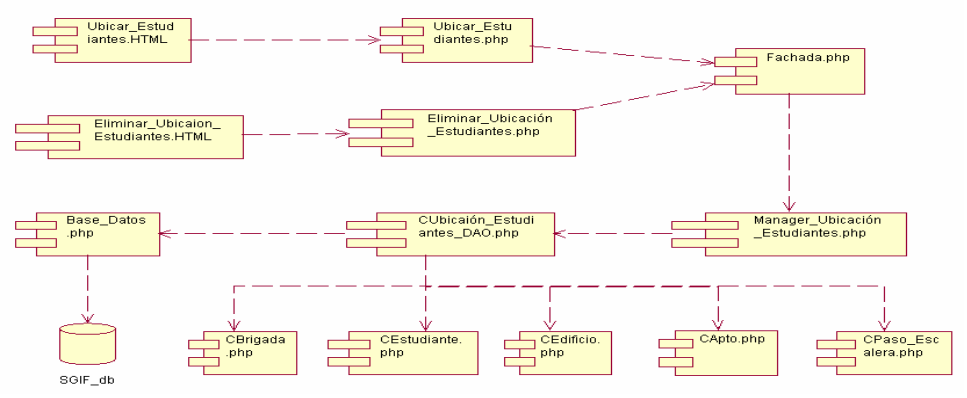


Figura 4.5 Diagrama de componentes del caso de uso: "Gestionar ubicación de estudiantes en la residencia"

4.2.6 Diagrama de componentes del caso de uso "Gestionar Edificio"

Diagrama de componentes del caso de uso Gestionar Edificio

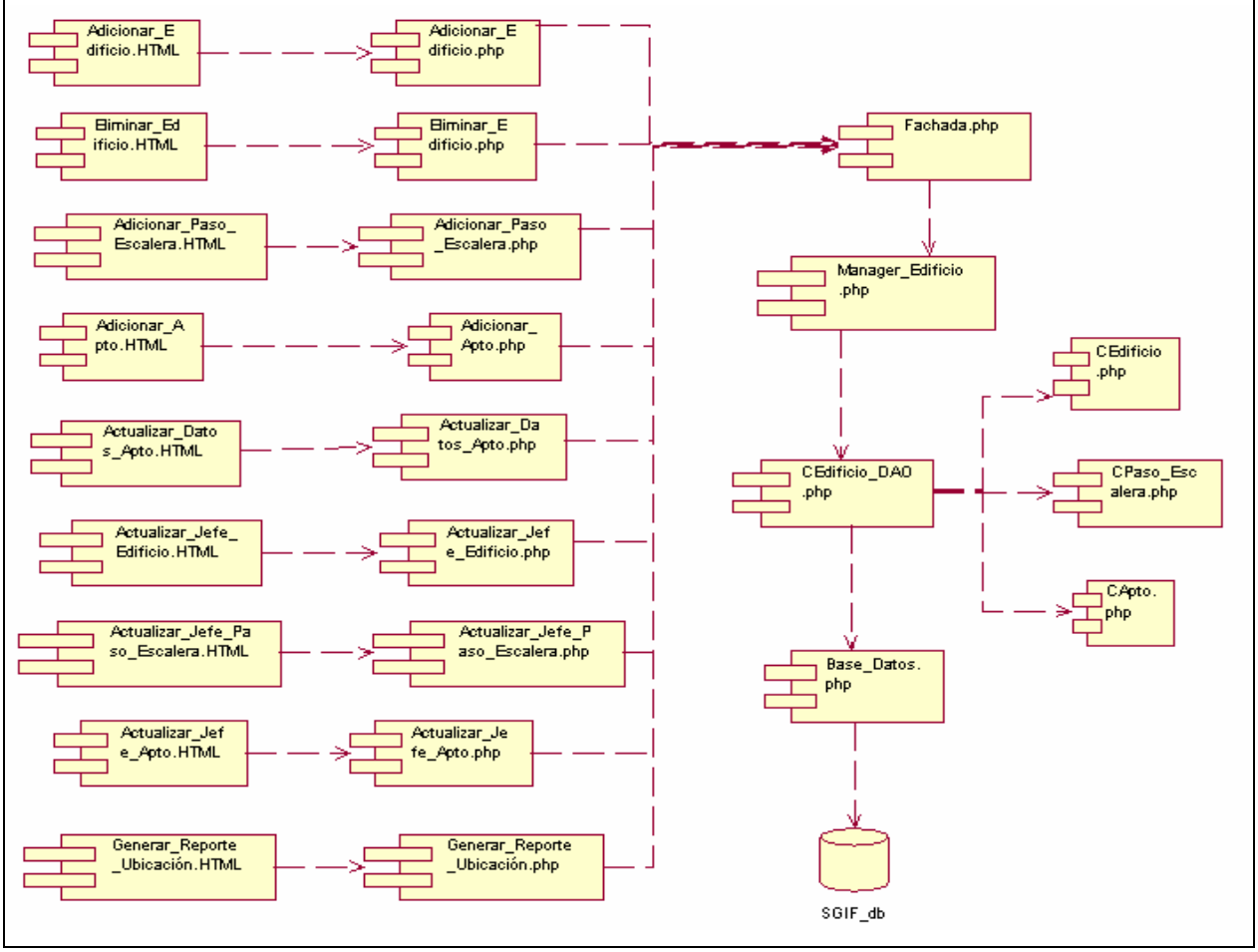


Figura 4.6 Diagrama de componentes del caso de uso: "Gestionar edificios"

4.3 Modelo de Despliegue

En un diagrama de despliegue se muestran las relaciones físicas entre los componentes de hardware y de software, es decir la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes de software, que no son más, que los procesos que se ejecutan en ellos. Se puede decir también que, un diagrama de despliegue es un grafo de nodos, unidos por conexiones de comunicación, donde un nodo puede contener instancias de componentes. En general un nodo, se entiende como una unidad de computación de algún tipo como es el caso de impresoras o la misma computadora.

A continuación se muestra el diagrama de despliegue del sistema:

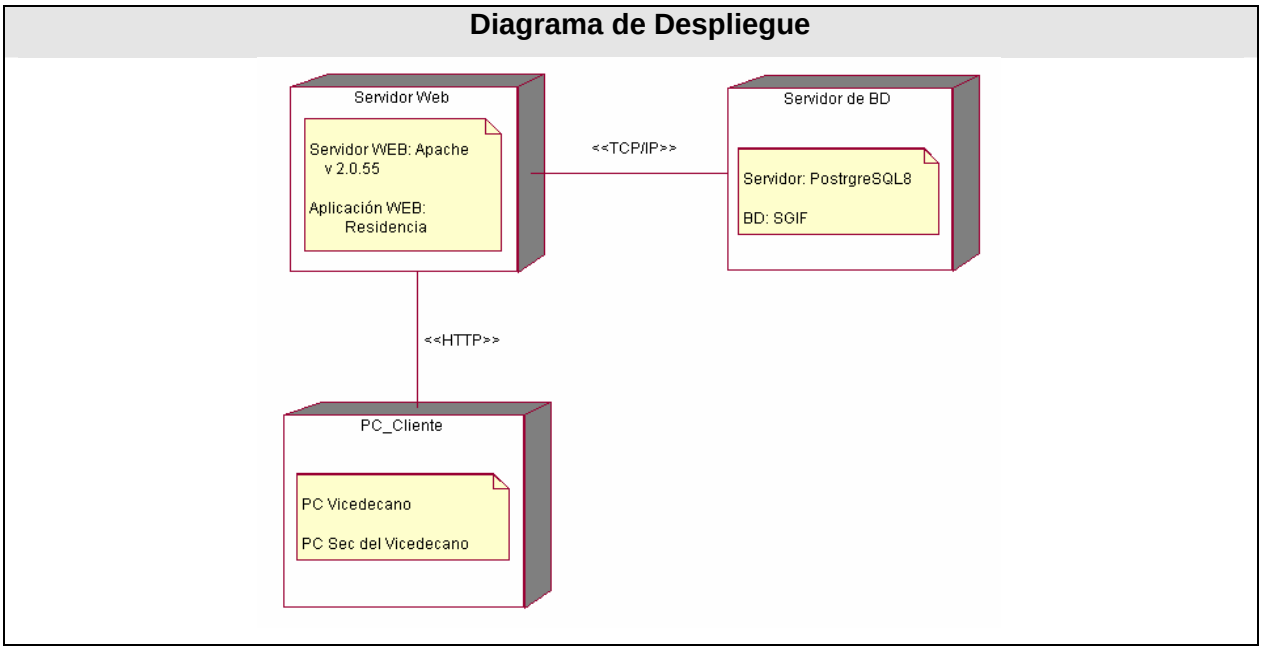


Figura 4.7 Diagrama de despliegue

4.4 Prueba

Las pruebas constituyen una actividad en la cual un sistema o componente de este, es ejecutado bajo ciertas condiciones o requerimientos específicos, en el que los resultados obtenidos son observados y registrados, para la realización posterior de alguna evaluación de dicho componente o sistema.

Dentro de los varios métodos de pruebas que existen, se encuentran las conocidas como **Pruebas de Caja Negra,** las cuales poseen las siguientes características:

- Verifican las especificaciones funcionales y no consideran la estructura interna del programa.
- No validan funciones ocultas, como es el caso de las funciones que son implementadas pero no descritas durante la fase de diseño, por lo que los errores asociados a ellas, no serán encontrados.
- Son realizadas sin el conocimiento interno del producto.

Por otro lado, las pruebas de caja negra son realizadas sobre la interfaz del sistema, es decir, controlan tanto los datos de entrada como los de salida.

A continuación se presentan las pruebas realizadas al sistema por cada caso de uso, y siguiendo el método de caja negra descrito anteriormente.

4.4.1 Pruebas de Caja Negra para el Caso de uso "Permitir Autenticarse"

A continuación se muestran las pruebas de caja negra realizadas al caso de uso Permitir Autenticarse.

Condición de Entrada	Casos Válidos	Casos no Válidos
Usuario	Cadena de caracteres	Dejar vacío el campo Usuario
Contraseña	Cadena de caracteres	Dejar vacío el campo Contraseña

Caso de Uso:	Permitir Autenticarse
Caso de Prueba:	Permitir Autenticarse a un usuario entrando correctamente los datos.
Entrada:	
El Usuario introduce correctamente los datos necesarios para acceder al sistema. Usuario: "antonio" Contraseña: "vicedecano"	
Resultado:	El sistema le da acceso al usuario autenticado y lo redirecciona para la página principal.
Condiciones:	Usuario y Contraseña introducidos correctamente.

Caso de Uso:	Permitir Autenticarse
Caso de Prueba:	Permitir Autenticarse a un usuario entrando incorrectamente alguno de los datos.
Entrada:	
El Usuario introduce incorrectamente alguno de los datos necesarios para acceder al sistema. Usuario: "Campo Omitido"	
Resultado:	El sistema muestra un mensaje de alerta notificando el error ("Olvidó introducir su Usuario").
Condiciones:	Dato introducido incorrectamente

4.4.2 Pruebas de Caja Negra para el Caso de uso "Gestionar Edificios"

El caso de uso "Gestionar Edificios", debido a su magnitud, ha sido subdividido en varios escenarios. A continuación se presentan las pruebas de caja negra realizadas al escenario "Adicionar Edificio", el resto de las mismas se pueden encontrar en el *Anexo 4*.

4.4.2.1 Escenario Adicionar Edificio

Condición de Entrada	Casos Válidos	Casos no Válidos
Número de Edificio	Números	No introducir el número de edificio. Lo que se introduce no es un número. Lo que se introduce no es un número entero positivo.

Caso de Uso:	Gestionar Edificio (Adicionar Edificio)
Caso de Prueba:	Adicionar un nuevo edificio entrando correctamente los datos.
Entrada:	
El Vicedecano introduce correctamente el dato necesario para adicionar un edificio. Número del Edificio: “91”	
Resultado:	El sistema adiciona el edificio en la base de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	El número de edificio debe estar en el rango de valores permitidos.

Caso de Uso:	Gestionar Edificio (Adicionar Edificio)
Caso de Prueba:	Adicionar un nuevo edificio entrando el dato erróneo.
Entrada:	
El Vicedecano introduce incorrectamente el dato necesario para adicionar un edificio. Número del Edificio: “Campo Omitido”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Número de Edificio Incorrecto”).
Condiciones:	Dato introducido incorrectamente

4.4.3 Pruebas de Caja Negra para el Caso de uso “Gestionar Ubicación de Estudiantes en la Residencia”

El caso de uso “Gestionar Ubicación de Estudiantes en la Residencia”, ha sido subdividido en dos escenarios. A continuación se presentan las pruebas de caja negra realizadas a cada uno de los mismos.

4.4.3.1 Escenario Ubicar Estudiantes en la Residencia

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio	No seleccionar ningún edificio. Seleccionar un edificio que posee todas sus capacidades cubiertas.
Selección de un Paso de Escalera	Seleccionar un Paso de Escalera	No seleccionar ningún paso de escalera. Seleccionar un paso de escalera que posee todas sus capacidades cubiertas.
Selección de un Apartamento	Seleccionar un Apartamento	No seleccionar ningún apartamento. Seleccionar un apartamento que posee todas sus capacidades cubiertas.
Selección de una Brigada	Seleccionar una Brigada	No seleccionar ninguna brigada. Seleccionar una brigada que posee todos sus estudiantes ya ubicados.
Selección de Estudiantes	Seleccionar al menos un estudiante	No seleccionar ningún estudiante. Seleccionar una cantidad de estudiantes mayor que la capacidad del apartamento donde se desean ubicar a estos.

Caso de Uso:	Gestionar Ubicación de Estudiantes en la Residencia (Ubicar Estudiantes en la Residencia)
Caso de Prueba:	Ubicar Estudiantes en la Residencia, habiendo seleccionado correctamente los datos para ello.
Entrada:	
El Vicedecano selecciona correctamente los datos necesarios para ubicar uno o varios estudiantes en la residencia. Edificios: “93 Cap (152) Ubic (130)” Pasos de Escalera: “1 Cap (84) Ubic (68)”	

Apartamentos: “93101 Cap (10) Ubic (8) F” Brigadas: “8502 Cant (12) Ubic (9)” Estudiante12: “Alfonso Pérez Roque”	
Resultado:	El sistema ubica a los estudiantes seleccionados en el edificio, paso de escalera y apartamento seleccionado previamente por el Vicedecano y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	Seleccionar los datos necesarios para ubicar uno o varios estudiantes correctamente.

Caso de Uso:	Gestionar Ubicación de Estudiantes en la Residencia (Ubicar Estudiantes en la Residencia)
Caso de Prueba:	Ubicar Estudiantes en la Residencia, sin haber seleccionado correctamente alguno de los datos necesarios para ello.
Entrada:	
El Vicedecano no selecciona correctamente alguno de los datos necesarios para ubicar los estudiantes deseados. Edificios: “93 C (152) U (140)” Pasos de Escalera: “1 C (84) U (84)” Apartamentos: “----------” Brigadas: “----------”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“El paso de escalera seleccionado ya posee todas sus capacidades cubiertas”).
Condiciones:	No haber seleccionado un paso de escalera correcto.

4.4.3.2 Escenario Eliminar Ubicación de Estudiantes en la Residencia

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio	No seleccionar ningún edificio.
Selección de un Paso de Escalera	Seleccionar un Paso de Escalera	
Selección de un Apartamento	Seleccionar un Apartamento	
Selección de una Brigada	Seleccionar una Brigada	

Selección de Estudiantes	Seleccionar o no Estudiantes	

Caso de Uso:	Gestionar Ubicación de Estudiantes en la Residencia (Eliminar Ubicación de Estudiantes en la Residencia)
Caso de Prueba:	Eliminar Ubicación de Estudiantes en la Residencia, habiendo seleccionado correctamente los datos para ello.
Entrada:	
El Vicedecano selecciona correctamente los datos necesarios para eliminar la ubicación de uno o varios estudiantes. Edificios: “93 C (152) U (140)” Pasos de Escalera: “1 C (84) U (80)” Apartamentos: “----------”	
Resultado:	El sistema elimina la ubicación de los estudiantes seleccionados en la base de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	Seleccionar los datos necesarios correctamente.

Caso de Uso:	Gestionar Ubicación de Estudiantes en la Residencia (Eliminar Ubicación de Estudiantes en la Residencia)
Caso de Prueba:	Eliminar Ubicación de Estudiantes en la Residencia, sin haber seleccionado correctamente el dato necesario para ello.
Entrada:	
El Vicedecano no selecciona correctamente alguno de los datos necesarios para eliminar la ubicación de los estudiantes deseados. Edificios: “----------” Pasos de Escalera: “----------” Apartamentos: “----------”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Edificio”).
Condiciones:	No haber seleccionado el dato necesario para eliminar la ubicación de los estudiantes deseados.

4.4.4 Pruebas de Caja Negra para el Caso de uso "Gestionar Parte de la Guardia Estudiantil"

El caso de uso "Gestionar Parte de la guardia Estudiantil", ha sido subdividido en dos escenarios. A continuación se presentan las pruebas de caja negra realizadas a cada uno de los mismos.

4.4.4.1 Escenario Adicionar Parte de la Guardia Estudiantil

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de una Fecha.	Seleccionar una Fecha	No seleccionar ninguna fecha.
Selección de un Trabajador	Seleccionar un Trabajador	No seleccionar ningún trabajador
Selección de una brigada	Seleccionar una brigada	No seleccionar ninguna brigada

Caso de Uso:	Gestionar Parte de la Guardia Estudiantil (Adicionar parte de la guardia estudiantil)
Caso de Prueba:	Adicionar parte de la guardia estudiantil habiendo seleccionado correctamente los datos necesarios para ello.
Entrada:	
El Vicedecano selecciona correctamente los datos necesarios para adicionar un parte de la guardia estudiantil. Fecha de Realización: "12/10/2007" Trabajador de Guardia: "José Carlos Alfonso Roque" Brigada de Guardia: "8502"	
Resultado:	El sistema adiciona el parte de la guardia estudiantil en la base de datos y muestra un mensaje notificando la acción realizada ("Proceso Satisfactorio").
Condiciones:	Haber seleccionado correctamente los datos necesarios para ello

Caso de Uso:	Gestionar Parte de la Guardia Estudiantil (Adicionar parte de la guardia estudiantil)
Caso de Prueba:	Adicionar parte de la guardia estudiantil sin haber seleccionado correctamente alguno de los datos necesarios para ello.
Entrada:	

El Vicedecano no selecciona correctamente alguno de los datos necesarios para adicionar un parte de la guardia estudiantil. Fecha: “19/5/2007” Profesor de Guardia: “----------” Brigada de Guardia: “8502”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Profesor de Guardia”).
Condiciones:	No haber seleccionado alguno de los datos necesarios para adicionar un parte de la guardia estudiantil.

4.4.4.2 Escenario Consultar Parte de la Guardia Estudiantil

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de una Fecha.	Seleccionar una Fecha	No seleccionar ninguna fecha.
Selección de una Brigada	Seleccionar una Brigada	No seleccionar ninguna brigada
Selección de un Parte	Seleccionar un Parte	No seleccionar ningún parte

Caso de Uso:	Gestionar Parte de la Guardia Estudiantil (Consultar parte de la guardia estudiantil)
Caso de Prueba:	Consultar parte de la guardia estudiantil habiendo seleccionado correctamente los datos necesarios para ello.
Entrada:	
El Vicedecano selecciona correctamente los datos necesarios para consultar un parte de la guardia estudiantil. Fecha de Realización: “5/5/2007” Parte de la Guardia: “8503 5/5/2007”	
Resultado:	El sistema muestra la información del parte seleccionado.
Condiciones:	Haber seleccionado correctamente los datos necesarios para ello

Caso de Uso:	Gestionar Parte de la Guardia Estudiantil (Consultar parte de la guardia estudiantil)
Caso de Prueba:	Consultar parte de la guardia estudiantil sin haber seleccionado correctamente alguno de los datos necesario para ello.

Entrada:	
El Vicedecano no selecciona correctamente alguno de los datos necesarios para consultar un parte de la guardia estudiantil. Brigada: “8304” Parte de la Guardia: “Campo no Seleccionado”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Parte de la Guardia Estudiantil”).
Condiciones:	No haber seleccionado alguno de los datos necesarios para consultar un parte de la guardia estudiantil.

4.5 Conclusiones

En este capítulo fue presentado cómo está construido el sistema a partir de los diagramas de componentes de cada caso de uso, así como, los diagramas de componentes de todo el sistema en general. También fue mostrado el diagrama de despliegue, el cual ilustra cuáles serán los nodos que serán usados para la implantación de la aplicación. Por último, fueron detalladas un conjunto de pruebas que se le realizaron al sistema, donde se especifica qué parámetros fueron utilizados durante la realización de las mismas, además del resultado obtenido durante la ejecución de cada una de ellas.

Conclusiones

Como resultado de este trabajo se obtuvo un sistema informático, el cual se piensa que mejore en gran medida la manera en la que se realizan varios de los procesos que tienen lugar en la residencia estudiantil de la facultad 8 de la Universidad de Ciencias Informáticas (UCI), como es el caso de tener un control sobre la ubicación de cada estudiante en los apartamentos y edificios que fueron asignados a la facultad, además de llevar un control sobre los partes de la guardia estudiantil que se emiten a diario, lo que trae consigo que la ejecución de dichos procesos se realice con una mayor organización y rapidez. Por tales razones, se piensa haberle dado cumplimiento a los objetivos trazados de forma satisfactoria, además de haberle dado solución a varios de los problemas encontrados a la hora de llevar a cabo muchos de estos procesos, como es el caso de: agilizar en gran medida la búsqueda de cualquier información y disponibilidad permanente de la misma, trayendo consigo que se pueda contar con la información requerida cada vez que se desee, además de encontrarse la misma con un mayor grado de organización.

Recomendaciones

A lo largo de este trabajo, se puede apreciar cómo se fueron cumpliendo cada uno de los objetivos trazados en el mismo, además de que se puede observar claramente el alcance y la forma en la que se le dio solución a varios de los problemas encontrados, no obstante, se realizan varias recomendaciones a aquellos que darán continuación a dicho trabajo, entre las que se encuentran las siguientes:

- Continuar con el estudio y análisis del resto de los procesos que tienen lugar en la Residencia Estudiantil de la facultad 8 de la Universidad de Ciencias Informáticas, con el objetivo de ampliar las funcionalidades del sistema que se propone.
- Perfeccionar la forma en la que se realizan todas las validaciones en el sistema, con vista a dar solución a posibles imprevistos que puedan ocurrir durante la ejecución del mismo.
- Incorporar más información a los reportes que se generan, con el objetivo de que estos tengan una mayor utilidad.
- Implementar la seguridad del sistema por módulos y no por páginas como se encuentra en estos momentos además de la posibilidad de que el sistema muestre solamente el menú que utiliza el usuario que está haciendo uso del mismo, según su rol.

Referencias Bibliográficas

1. Anónimo. *Lenguaje de programación*. 2006 [cited febrero 2007]; Available from: http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n.
2. Anónimo. *PHP*. 2006 [cited febrero 2007]; Available from: http://es.wikipedia.org/wiki/PHP#Ventajas_de_PHP.
3. Anónimo. *Características de Java*. 2006 [cited febrero 2007]; Available from: http://pisuerga.inf.ubu.es/lsi/Invest/Java/Tuto/I_3.htm.
4. Ricote, A. and M. Figueredo, *SGIF: Sistema de Gestión de la Información para la facultad*. 2006, Instituto Superior Politécnico "José Antonio Echeverría".
5. *OMG Unified Modeling Language Specification*. 2003, OMG.
6. Schmuller, J., *Aprendiendo UML en 24 horas*: Prentice Hall.
7. Anónimo. *Metodologías de Desarrollo de Software*. 2004 [cited febrero 2007]; Available from: http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html.
8. Anónimo. *Comparación de Herramientas de Modelado UML: Enterprise Architect y Rational Rose*. 2006 [cited febrero 2007]; Available from: http://www.apexnet.com.ar/index.php/news/main/38/event=view.
9. Anónimo. *Rational Rose: Procedimientos básicos para desarrollar un proyecto con UML*. 2006 [cited febrero 2007]; Available from:http://www.vico.org/TallerRationalRose.pdf.
10. Anónimo. *Enterprise Architect*. 2005 [cited febrero 2007; Available from: http://www.sparxsystems.com.ar/products/ea.html.
11. Anónimo. *NuSphere PHPEd*. 2005 [cited febrero 2007]; Available from:http://nusphere-phped.softonic.com.
12. Anónimo. *Editores para PHP, Zend Studio*. 2007 [cited febrero 2007]; Available from: http://www.adrformacion.com/cursos/php/leccion1/tutorial3.html.
13. Anónimo. *Smarty*. 2006 [cited febrero 2007; Available from: http://es.wikipedia.org/wiki/Smarty.
14. Jacobson, I., G. Booch, and J. Rumbaugh, *El Proceso Unificado de Desarrollo de software*. 2000: Addison-Wesley.

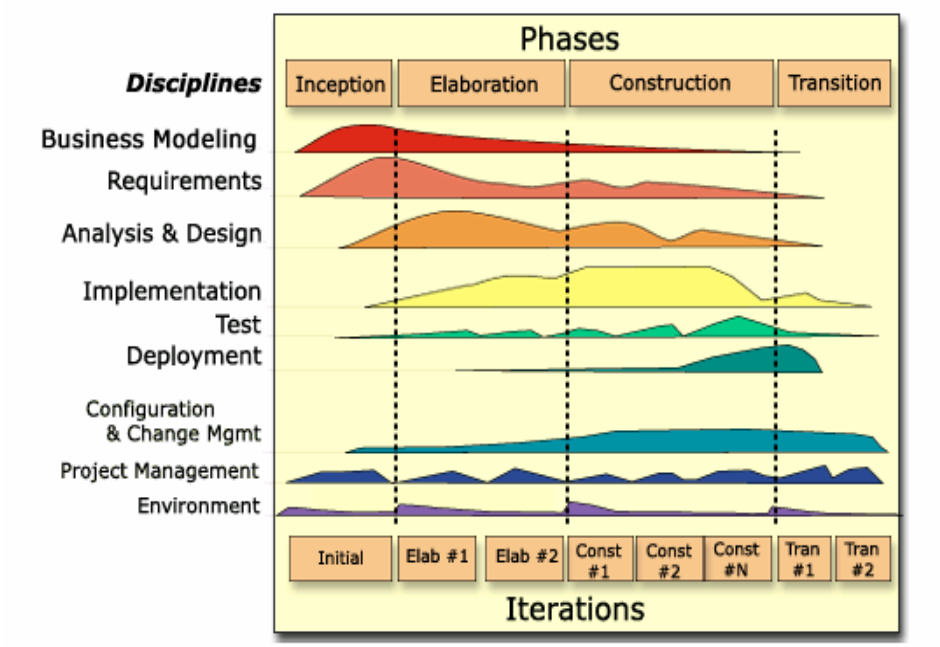
Glosario de Términos y Siglas

- ASP: *Active Server Pages.* Es una tecnología del lado servidor de Microsoft para páginas Web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Server (IIS).
- CASE: *Computer Aided Software Engineering.*
- CUN: *Caso de uso del negocio*.
- CUS: *Caso de uso del sistema*.
- HTML: *Hypertext Markup Language.* Lenguaje usado para escribir documentos para servidores World Wide Web. Es una aplicación de la ISO Standard 8879:1986. Es un lenguaje de marcas. Los lenguajes de marcas no son equivalentes a los lenguajes de programación aunque se definan igualmente como "lenguajes". Son sistemas complejos de descripción de información, normalmente documentos, que se pueden controlar desde cualquier editor ASCII.
- HTTP: *HyperText Transfer Protocol.* Protocolo de Transferencia de Hipertextos. Modo de comunicación para solicitar páginas Web.
- Herramientas CASE: Herramientas utilizadas para el desarrollo de proyectos de Ingeniería de Software.
- Hardware: Componentes electrónicos, tarjetas, periféricos y equipo que conforman un sistema de computación; se distinguen de los programas (software) porque son tangibles.
- Internet: Sistema de redes de computación ligadas entre sí, con alcance mundial, que facilita servicios de comunicación de datos como registro remoto, transferencia de archivos, correo electrónico y grupos de noticias. Internet es una forma de conectar las redes de computación existentes que amplía en gran medida el alcance de cada sistema participante.
- Java: Es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los 90.
- Microsoft: Compañía que manufactura los sistemas de operación DOS y Windows.

- MySQL: Es un sistema de gestión de bases de datos relacional que cuentan con todas las características de un motor de BD comercial: transacciones atómicas, triggers, replicación, llaves foráneas entre otras. Su ingeniosa arquitectura lo hace extremadamente rápido y fácil de personalizar.
- PHP: *PHP: Hypertext Preprocessor*. Es un ambiente script del lado del servidor que permite crear y ejecutar aplicaciones Web dinámicas e interactivas. Con PHP se pueden combinar páginas HTML y script*s*. Con el objetivo de crear aplicaciones potentes.
- PostgreSQL: es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) libre.
- RUP: *Rational Unified Process* (Proceso Unificado de desarrollo). Metodología para el desarrollo de Software.
- Samrty: Es un motor de plantillas para PHP, cuyo objetivo es separar el contenido de la presentación en una página Web.
- Software: Programas de sistema, utilerías o aplicaciones expresados en un lenguaje de máquina.
- SQL: *Structured Query Language*. Es un lenguaje declarativo de acceso a bases de datos que permite especificar diversos tipos de operaciones sobre las mismas. Aúna características del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar información de interés de una base de datos.
- SGBD: *Sistema de Gestión de Bases de Datos*. Es el software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez.
- UCI: *Universidad de las Ciencias Informáticas*.
- UML: *Unified Modeling Language*. Es una notación estándar para modelar objetos del mundo real como primer paso en el desarrollo de programas orientados a objetos. Es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema de software.
- XML: *Extensible Markup Language*. Es un lenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium. Orientado principalmente al almacenamiento, procesamiento y transmisión de mensajes.

Anexos

Anexo 1: Fases y Flujos de trabajos de RUP



Fases

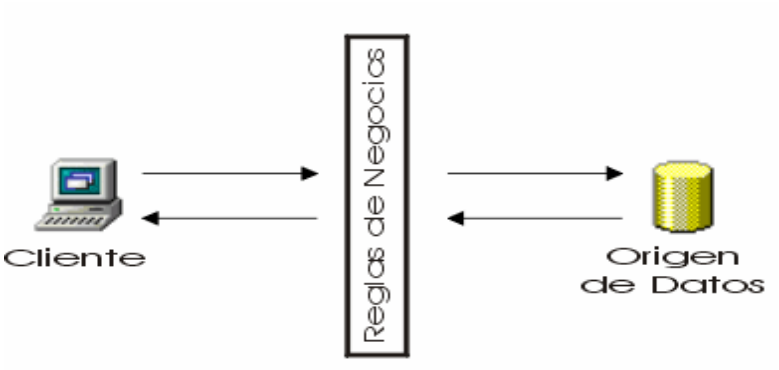
- Inicio
- Elaboración
- Construcción.
- Transición.

Flujos de trabajo

- Modelamiento del negocio.
- Requerimientos.
- Análisis y diseño.
- Implementación.

- Prueba (Testeo).
- Instalación.
- Administración del proyecto.
- Administración de configuración y cambios.
- Ambiente.

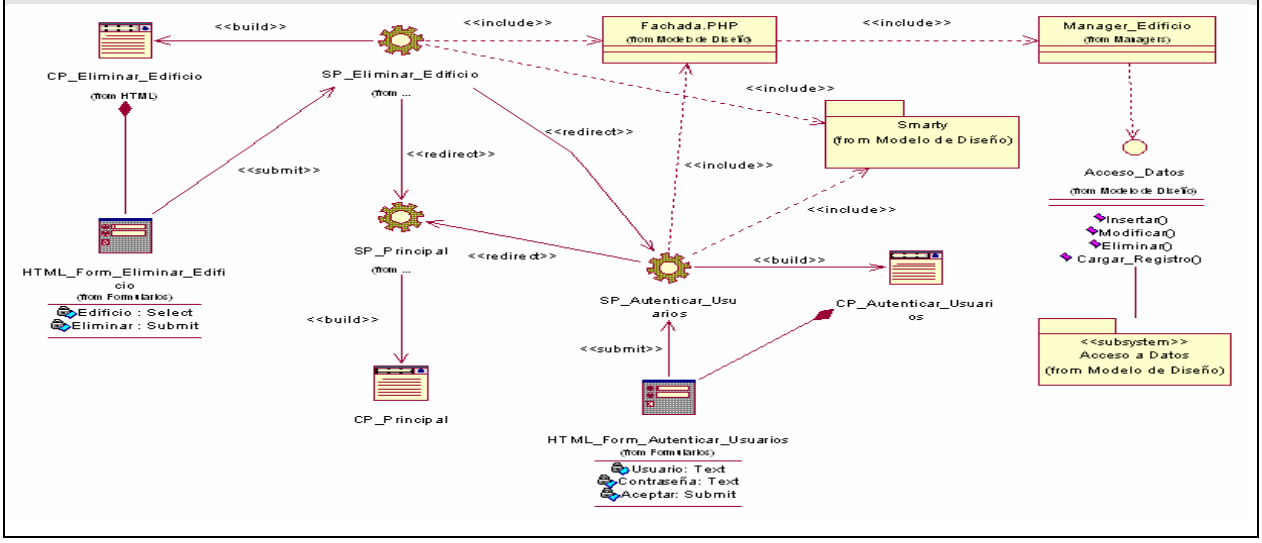
Anexo 2: Modelo de diseño en 3 capas



Anexo 3: Diagramas de clases Web correspondientes al resto de los escenarios del caso de uso "Gestionar Edificios"

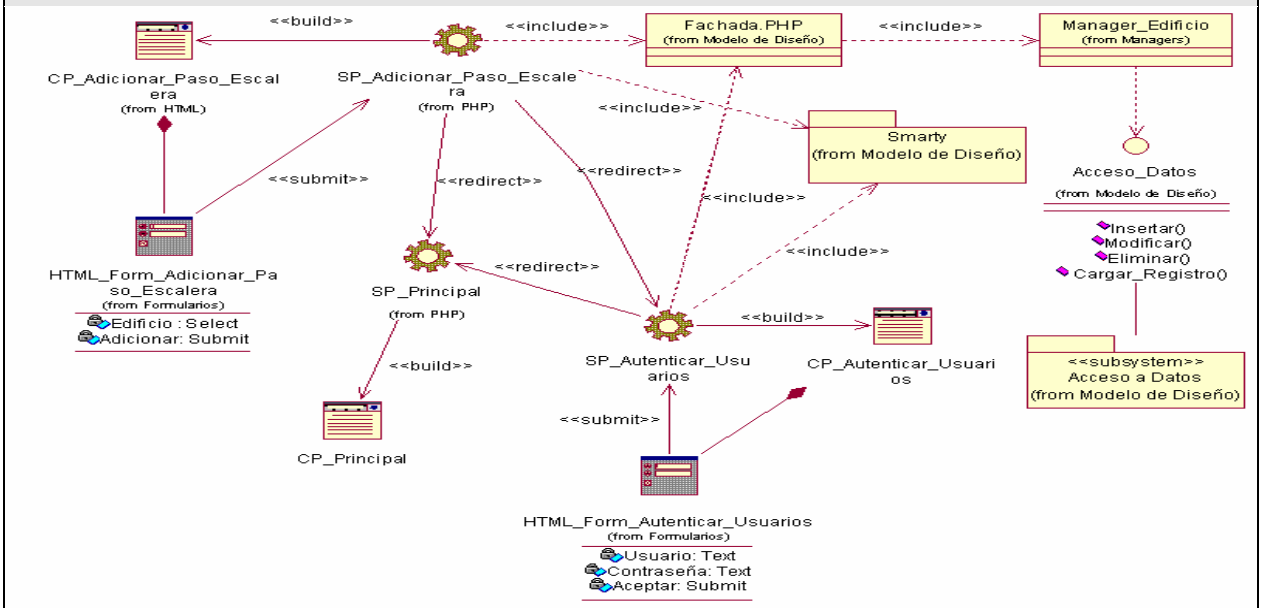
Escenario Eliminar Edificio

Diagrama de clases Web para el escenario "Eliminar Edificio"



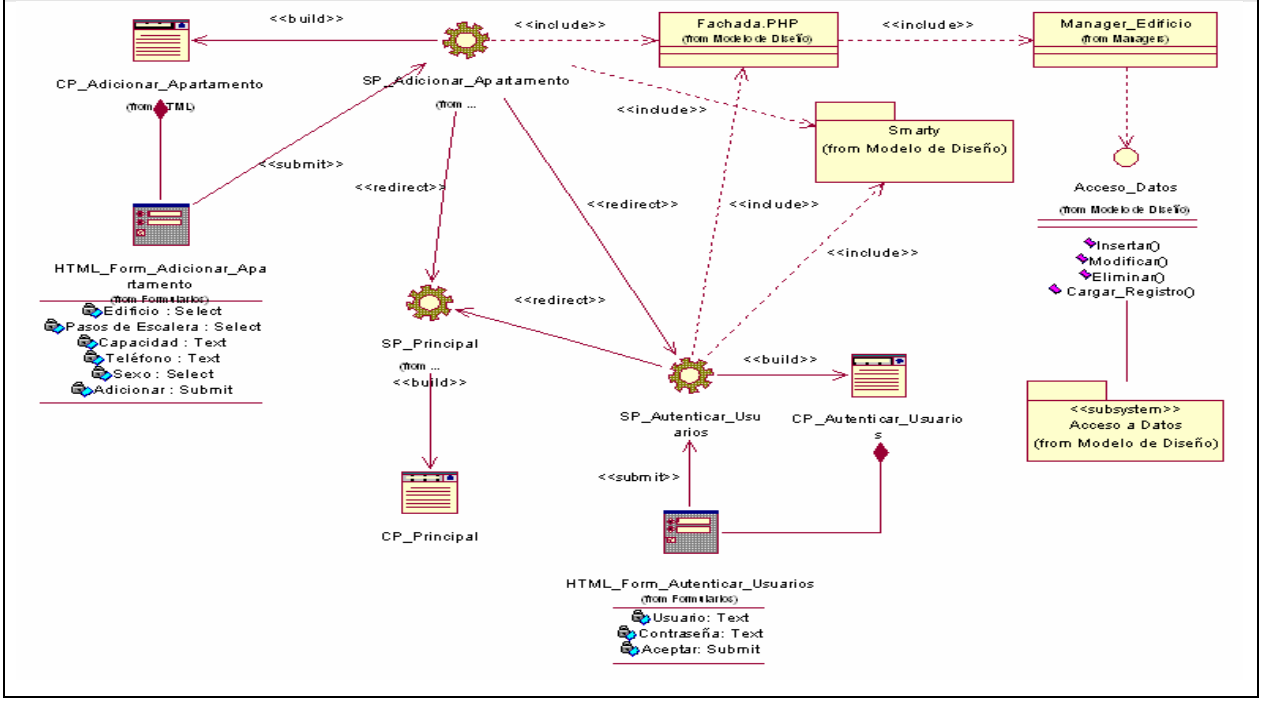
Escenario Adicionar Paso de Escalera

Diagrama de clases Web para el escenario "Adicionar Paso de Escalera"



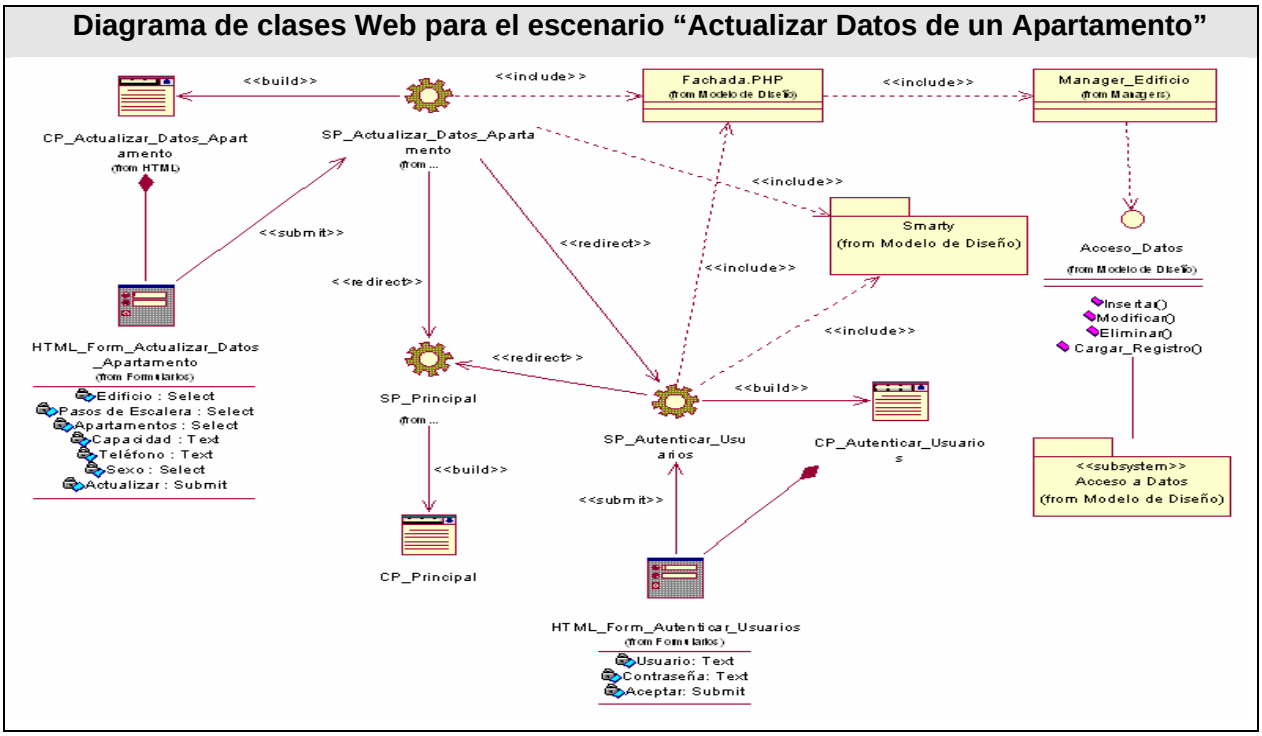
Escenario Adicionar Apartamento

Diagrama de clases Web para el escenario "Adicionar Apartamento"



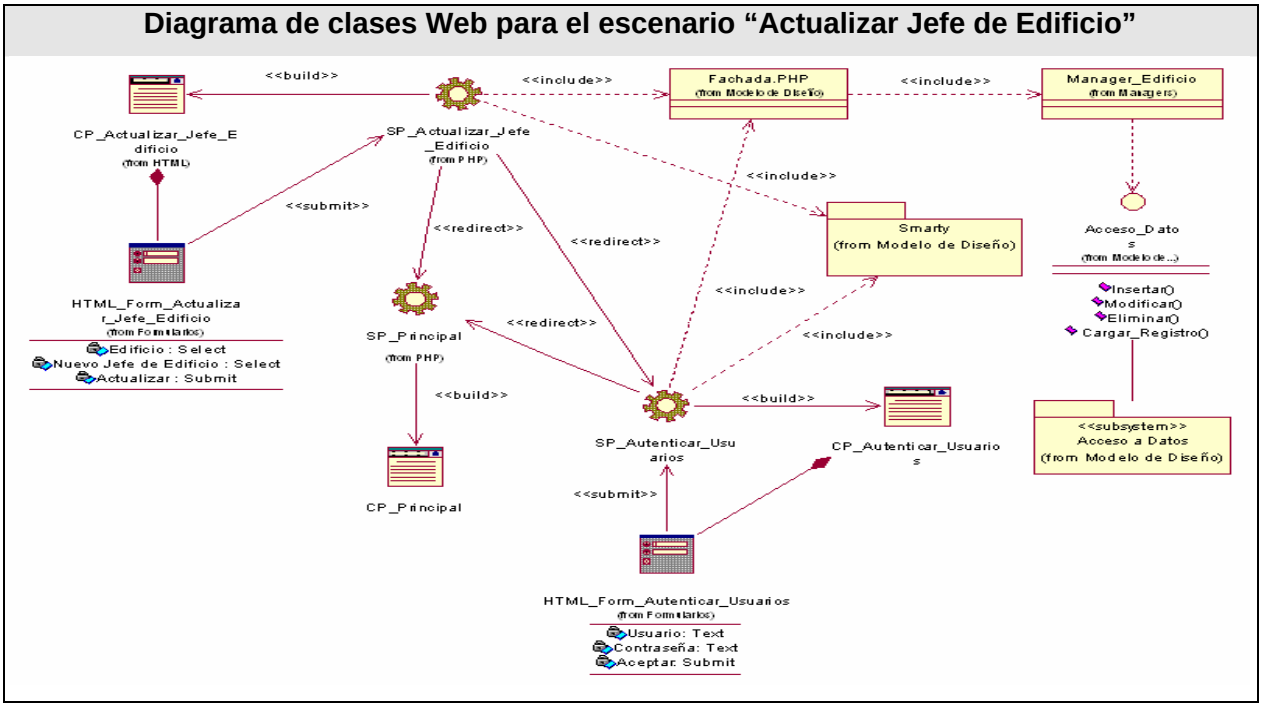
Escenario Actualizar Datos de un Apartamento

Diagrama de clases Web para el escenario "Actualizar Datos de un Apartamento"



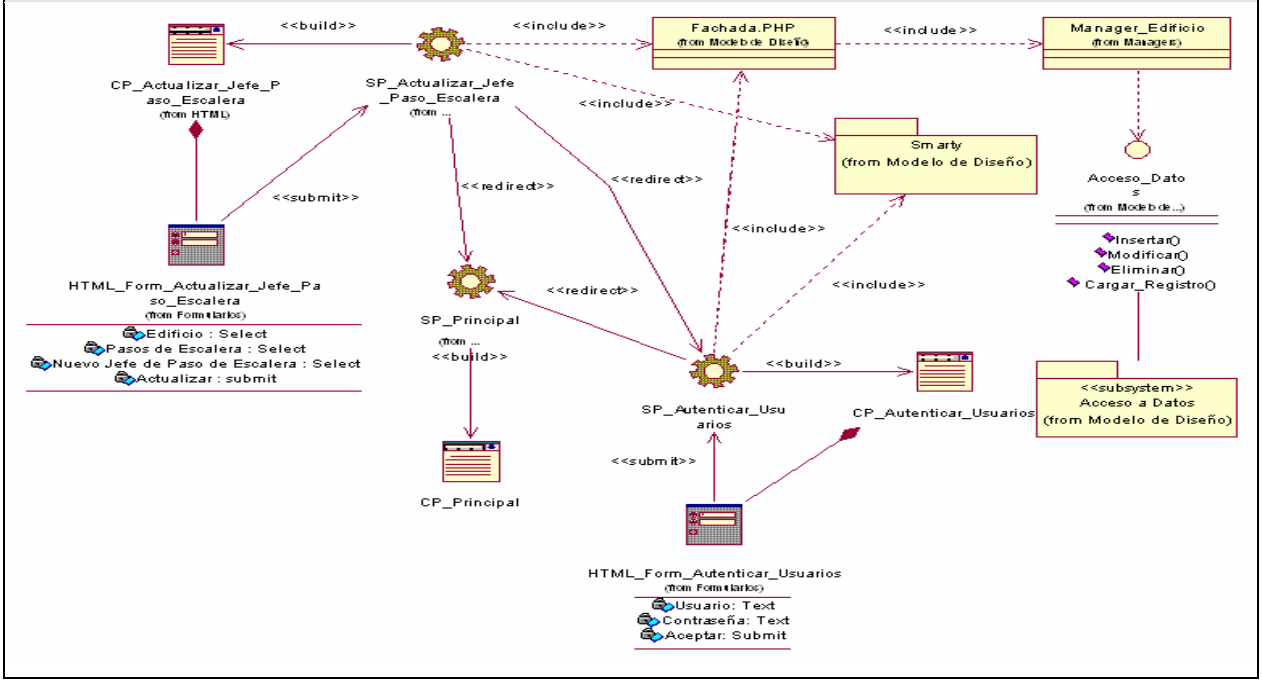
Escenario Actualizar Jefe de Edificio

Diagrama de clases Web para el escenario "Actualizar Jefe de Edificio"



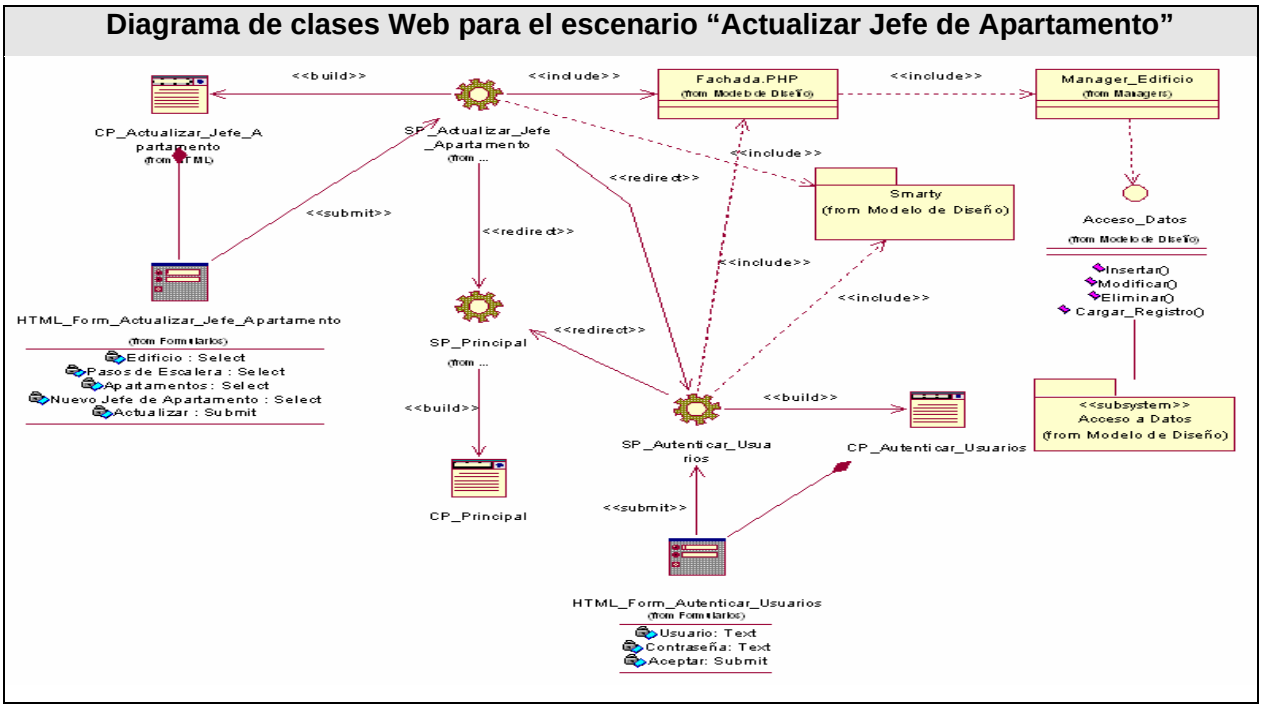
Escenario Actualizar Jefe de Paso de Escalera

Diagrama de clases Web para el escenario "Actualizar Jefe de Paso de Escalera"



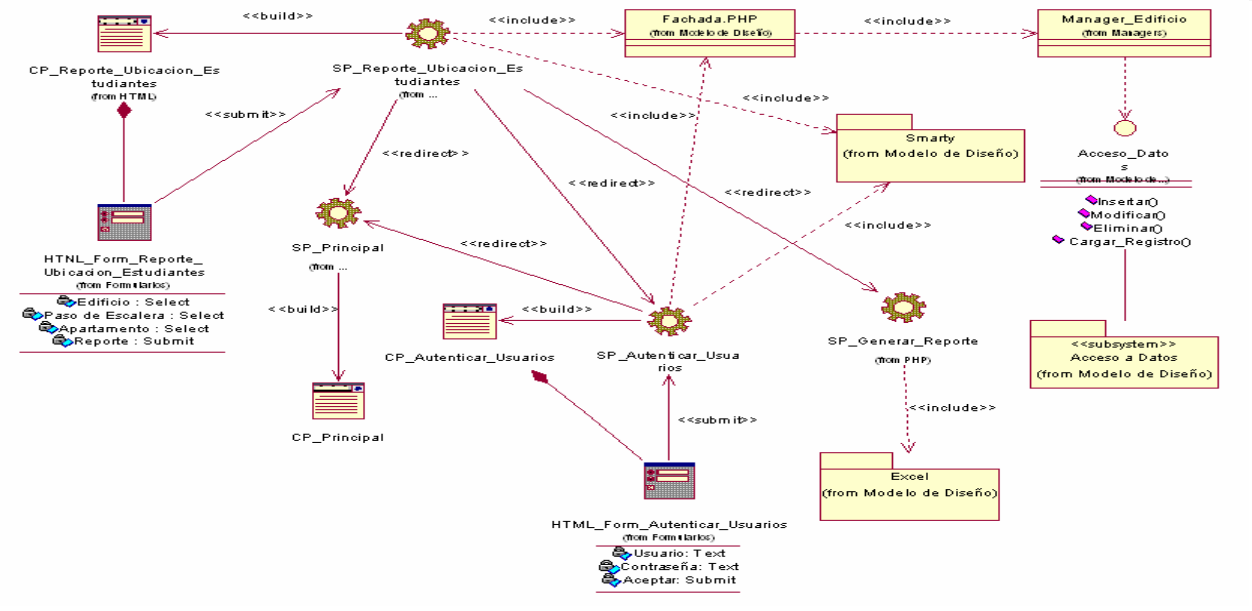
Escenario Actualizar Jefe de Apartamento

Diagrama de clases Web para el escenario "Actualizar Jefe de Apartamento"



Escenario Generar Reporte de Ubicación de Estudiantes

Diagrama de clases Web para el escenario
"Generar Reporte de Ubicación de Estudiantes"



Anexo 4: Pruebas de caja negra realizadas al resto de los escenarios del caso de uso: "Gestionar Edificios"

Escenario Adicionar Paso de Escalera

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio	Seleccionar un Edificio.	No seleccionar ningún edificio.

Caso de Uso:	Gestionar Edificio (Adicionar Paso de Escalera)
Caso de Prueba:	Adicionar un nuevo paso de escalera habiendo seleccionando correctamente un edificio.
Entrada:	
El Vicedecano selecciona correctamente un edificio para adicionarle a este un paso de escalera. Edificio: "89"	
Resultado:	El sistema adiciona el paso de escalera al edificio seleccionado en la base

	de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	Haber seleccionado un edificio.

Caso de Uso:	Gestionar Edificio (Adicionar Paso de Escalera)
Caso de Prueba:	Adicionar un nuevo paso de escalera sin haber seleccionado ningún edificio.
Entrada:	
El Vicedecano no selecciona ningún edificio. Edificio: “-----”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Edificio”).
Condiciones:	No haber seleccionado ningún edificio.

Escenario Adicionar Apartamento

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio	No seleccionar ningún Edificio.
Selección de un paso de escalera.	Seleccionar un Paso de Escalera	No seleccionar ningún Paso de Escalera.
Capacidad.	Número entero entre 4 y 12.	No entrar la capacidad. La capacidad entrada no es un número entero. La capacidad entrada es un número entero que no está en el rango de 4 a 12.
Teléfono.	Número de 4 cifras.	No entrar un Teléfono. El teléfono entrado no es un número. El teléfono entrado no es un número de 4 cifras.
Selección del sexo.	Seleccionar el Sexo	No seleccionar un sexo.

Caso de Uso:	Gestionar Edificio (Adicionar Apartamento)
Caso de Prueba:	Adicionar un nuevo apartamento habiendo entrado correctamente todos los datos.
Entrada:	
El Vicedecano introduce correctamente los datos necesarios para adicionar un nuevo apartamento. Edificio: “95” Pasos de Escalera: “1” Capacidad: “10” Teléfono: “3048” Sexo: “F”	
Resultado:	El sistema adiciona el apartamento en la base de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	Haber introducido los datos correctamente para la adición de un nuevo apartamento.

Caso de Uso:	Gestionar Edificio (Adicionar Apartamento)
Caso de Prueba:	Adicionar un nuevo apartamento habiendo entrado algún dato incorrecto.
Entrada:	
El Vicedecano introduce incorrectamente alguno de los datos necesarios para la adición de un nuevo apartamento. Edificio: “91” Pasos de Escalera: “2” Capacidad: “16” Teléfono: “2531” Sexo: “M”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“La Capacidad debe ser un número entre 4 y 12”).
Condiciones:	Dato fuera del rango de valores permitidos.

Escenario Modificar Datos de un Apartamento

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio.	No seleccionar ningún edificio.
Selección de un Paso de Escalera.	Seleccionar un Paso de Escalera.	No seleccionar ningún paso de escalera.
Selección de un Apartamento	Seleccionar un Apartamento.	No seleccionar ningún apartamento.
Capacidad.	Número entero entre 4 y 12.	No entrar la capacidad. La capacidad entrada no es un número entero. La capacidad entrada es un número entero que no está en el rango de 4 a 12.
Teléfono.	Número de 4 cifras.	No entrar un Teléfono. El teléfono entrado no es un número. El teléfono entrado no es un número de 4 cifras.
Selección del sexo.	Seleccionar el Sexo	No seleccionar un sexo.

Caso de Uso:	Gestionar Edificio (Modificar Datos de un Apartamento)
Caso de Prueba:	Modificar datos de un apartamento habiendo entrado correctamente los datos necesarios para ello.
Entrada:	
El Vicedecano introduce correctamente los datos necesarios para Modificar a su vez, los datos de un Apartamento. Edificio: “89” Pasos de Escalera: “1” Apartamentos: “89102” Capacidad: “10” Teléfono: “2712” Sexo: “F”	

Resultado:	El sistema modifica los datos del apartamento seleccionado en la base de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	Haber entrado los datos correctos para modificar a su vez los datos correspondientes al apartamento seleccionado.

Caso de Uso:	Gestionar Edificio (Modificar Datos de un Apartamento)
Caso de Prueba:	Modificar datos de un apartamento habiendo entrado algún dato erróneo.
Entrada:	
El Vicedecano introduce incorrectamente alguno de los datos que desea modificar del apartamento seleccionado. Edificio: “89” Pasos de Escalera: “1” Apartamentos: “89102” Capacidad: “10” Teléfono: “8372712” Sexo: “F”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Número de Teléfono Incorrecto”).
Condiciones:	Dato fuera del rango de valores permitidos.

Escenario Eliminar Edificio

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio	No seleccionar ningún edificio.

Caso de Uso:	Gestionar Edificio (Eliminar Edificio)
Caso de Prueba:	Eliminar un edificio habiéndolo seleccionado previamente.
Entrada:	
El Vicedecano selecciona correctamente un edificio para ser eliminado de la base de datos. Edificio: “95”	

Resultado:	El sistema elimina el edificio seleccionado de la base de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	Haber seleccionado un edificio.

Caso de Uso:	Gestionar Edificio (Eliminar Edificio)
Caso de Prueba:	Eliminar un edificio sin haber seleccionado ninguno previamente.
Entrada:	
El Vicedecano no selecciona ningún edificio. Edificio: “------”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Edificio”).
Condiciones:	No haber seleccionado ningún edificio.

Escenario Actualizar Jefe de Edificio

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio	No seleccionar ningún edificio.
Selección de un estudiante.	Seleccionar un Estudiante	No seleccionar ningún estudiante.

Caso de Uso:	Gestionar Edificio (Actualizar Jefe de Edificio)
Caso de Prueba:	Actualizar jefe de edificio, habiendo seleccionado correctamente los datos para ello.
Entrada:	
El Vicedecano selecciona correctamente los datos necesarios para actualizar el jefe de edificio. Edificio: “89” Nuevo Jefe de Edificio: “José Enríquez Pérez García (89102)”	
Resultado:	El sistema actualiza el nuevo jefe de edificio en la base de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	Haber seleccionado un edificio y un estudiante.

Caso de Uso:	Gestionar Edificio (Actualizar Jefe de Edificio)
Caso de Prueba:	Actualizar jefe de edificio sin haber seleccionado ninguno previamente o sin haber seleccionado un nuevo jefe para el mismo.
Entrada:	
El Vicedecano no selecciona alguno de los datos necesarios para actualizar el jefe de edificio. Edificio: “93” Nuevo Jefe de Edificio: “----------”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Nuevo Jefe de Edificio”).
Condiciones:	No haber seleccionado ningún estudiante.

Escenario Actualizar Jefe de Paso de Escalera

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio	No seleccionar ningún edificio.
Selección de un Paso de Escalera	Seleccionar un Paso de Escalera	No seleccionar ningún paso de escalera
Selección de un Estudiante.	Seleccionar un Estudiante	No seleccionar ningún estudiante.

Caso de Uso:	Gestionar Edificio (Actualizar Jefe de Paso de Escalera)
Caso de Prueba:	Actualizar jefe de paso de escalera, habiendo seleccionado correctamente los datos necesarios para ello.
Entrada:	
El Vicedecano selecciona correctamente los datos necesarios para actualizar el jefe de paso de escalera. Edificio: “89” Pasos de Escalera: “2” Nuevo Jefe de Paso de Escalera: “Ariel Orta Hernández”	
Resultado:	El sistema actualiza el nuevo jefe de paso de escalera en la base de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).

Condiciones:	Haber seleccionado correctamente los datos necesarios para actualizar el jefe de paso de escalera.

Caso de Uso:	Gestionar Edificio (Actualizar Jefe de Paso de Escalera)
Caso de Prueba:	Actualizar jefe de paso de escalera sin haber seleccionado algún dato necesario para ello
Entrada:	
El Vicedecano no selecciona alguno de los datos necesarios para actualizar el jefe de paso de escalera. Edificio: “94” Pasos de Escalera: “------” Nuevo Jefe de Paso de Escalera: “----------”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Paso de Escalera”).
Condiciones:	No haber seleccionado alguno de los datos necesarios para actualizar el jefe de paso de escalera.

Escenario Actualizar Jefe de Apartamento

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio	No seleccionar ningún edificio.
Selección de un Paso de Escalera	Seleccionar un Paso de Escalera	No seleccionar ningún paso de escalera
Selección de un Apartamento	Seleccionar un Apartamento	No seleccionar ningún apartamento
Selección de un Estudiante.	Seleccionar un Estudiante	No seleccionar ningún estudiante.

Caso de Uso:	Gestionar Edificio (Actualizar Jefe de Apartamento)
Caso de Prueba:	Actualizar jefe de apartamento, habiendo seleccionado correctamente los datos para ello.

Entrada:	
El Vicedecano selecciona correctamente los datos necesarios para actualizar el jefe de apartamento. Edificio: “94” Pasos de Escalera: “1” Apartamentos: “94101” Nuevo Jefe de Paso de Escalera: “Yudermis Vázquez Hernández”	
Resultado:	El sistema actualiza el nuevo jefe de apartamento en la base de datos y muestra un mensaje notificando la acción realizada (“Proceso Satisfactorio”).
Condiciones:	Haber seleccionado correctamente los datos necesarios para actualizar el jefe de apartamento.

Caso de Uso:	Gestionar Edificio (Actualizar Jefe de Apartamento)
Caso de Prueba:	Actualizar jefe de apartamento sin haber seleccionado algún dato necesario para ello
Entrada:	
El Vicedecano no selecciona alguno de los datos necesarios para actualizar el jefe de apartamento. Edificio: “94” Pasos de Escalera: “1” Apartamentos: “----------” Nuevo Jefe de Paso de Escalera: “----------”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Apartamento”).
Condiciones:	No haber seleccionado alguno de los datos necesarios para actualizar el jefe de apartamento.

Escenario Generar Reporte de Ubicación de Estudiantes

Condición de Entrada	Casos Válidos	Casos no Válidos
Selección de un Edificio.	Seleccionar un Edificio	No seleccionar ningún edificio.
Selección de un Paso de	Seleccionar un Paso de	

Escalera	Escalera	
Selección de un Apartamento	Seleccionar un Apartamento	

Caso de Uso:	Gestionar Edificio (Generar Reporte de Ubicación de Estudiantes)
Caso de Prueba:	Generar reporte de ubicación de estudiantes, habiendo seleccionado correctamente los datos para ello.
Entrada:	
El Vicedecano selecciona correctamente un edificio, y si lo desea, un paso de escalera y un apartamento. Edificio: “89” Pasos de Escalera: “1” Apartamentos: “----------”	
Resultado:	El sistema genera un documento Excel con la información solicitada por el Vicedecano.
Condiciones:	Haber seleccionado un edificio.

Caso de Uso:	Gestionar Edificio (Generar Reporte de Ubicación de Estudiantes)
Caso de Prueba:	Generar reporte de ubicación de estudiantes, sin haber seleccionado previamente un edificio.
Entrada:	
El Vicedecano no selecciona ningún edificio. Edificio: “------” Pasos de Escalera: “------” Apartamentos: “----------”	
Resultado:	El sistema muestra un mensaje de alerta notificando el error (“Debe seleccionar un Edificio”).
Condiciones:	No haber seleccionado ningún edificio para generar el reporte.