UNIVERSIDAD DE SANCTI SPÍRITUS

"JOSÉ MARTÍ PÉREZ"

FACULTAD DE INGENIERÍA

CARRERA INGENIERÍA INFORMÁTICA

Aplicación Web para la gestión de información en la Residencia Estudiantil de la Universidad de Sancti Spíritus "José Martí Pérez".

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERÍA INFORMÁTICA

Autor: Eric Acosta Puertas

Tutor: Ing. Julio Companioni Martínez

Consultante: Sonia

Sancti Spíritus, Cuba.

2014

Pensamiento

"Hay dos formas de diseñar software: la primera es hacerlo tan simple que obviamente no hay deficiencias y la segunda es hacerlo tan complicado que no hay deficiencias obvias. La primera forma es mucho más difícil."

Charles Antony Richard Hoare

Dedicatoria

A mis padres por dedicar sus vidas a nosotros, sufrir nuestros fracasos, por poner sus cubos de arena en cada uno de mis triunfos y estar siempre allí para contar con ellos.

Agradecimientos

- A mis padres porque siempre me incentivaron el deseo de terminar satisfactoriamente.
- A toda mi familia que siempre estuvo dándome ánimo y siempre creyeron que si podía.
- A mi novia Diana Hernández Hernández por darme todo su apoyo en los momentos amargos y alegres de la carrera, sin importar los que pasara siempre estaba allí para ayudarme en todo.
- A mi tutor Ing. Julio Companioni Martínez por creer en mí y brindarme todo el apoyo que les fue posible.
- A mis amigos que a pesar de no encontrarse en la universidad estudiando conmigo siempre me brindaron su apoyo de forma incondicional entre los cuales puedo mencionar Manuel A Herrera Escobar y Oscar Pérez Chaviano.
- A todos mis compañeros de estudio por estar juntos hasta el final. En especial agradecerle a Carlos A. Hernández Días, Saúl Rodríguez de León, Yadier Fuentes García, Yoel González Calero, Claudia Sánchez Prado, Amaray Utrera Gómez, Yunet Lorenzo Vega y Nuris L Lara Ramos entre otros que también aportaron su granito.
- A los profesores de la Universidad que fueron capaces de incentivar en mí el deseo de aprender, especialmente a los que me brindaron su ayuda cuando la necesité en este último año.

A todos muchas gracias.

Resumen

La revolución cubana como ejemplo de proliferación de libertad y cultura, ha venido desarrollando un amplio programa en materia de educación donde las universidades han jugado un rol decisivo en el desarrollo de todo nuestro pueblo, así como de los países amigos.

Es por ello que surgió la necesidad de diseñar e implementar un sistema para la Residencia Estudiantil de la Universidad de Sancti Spíritus "José Martí Pérez" donde se creó una aplicación web que facilita la gestión de la información dentro de esta. En el momento de concebir este producto se analizaron sistemas desarrollados con anterioridad como son: **SACBEXT, BDEDUCACION, SIBEX** y **SIREU.** En vista de los requerimientos hemos desarrollado un sistema basado en una tecnología cliente/servidor capaz de brindar soluciones que contribuyan a un mejor manejo de la información dentro de la residencia estudiantil. El flujo de la información y el trabajo de la residencia estudiantil fueron las bases para el desarrollo del sistema que se propone. El cual fue concebido bajo la utilización de la metodología RUP por su amplia difusión y facilidad de uso. Como herramienta de modelado se utilizó Visual Paradigm 8.0 y para el desarrollo se utilizó el IDE Sublime Text, el cual incluye facilidades de desarrollo para aplicaciones Web, incluyendo generación en código PHP, Java Script, HTML; lenguajes empleados en el desarrollo del software. Se utilizó como Sistema de Gestión de Base de Datos: MySql. Y como servidor web Apache.

Abstract

The Cuban Revolution as an example of proliferation of freedom and culture, has developed an extensive program in education where universities have played a decisive role in the development of all our people, as well as friendly countries.

That is why it became necessary to design and implement a system for student residence at the University of Sancti Spiritus “José Mart

Pérez " where a web application that facilitates the management of information within this was created. At the time of designing this product developed systems previously analyzed as: SACBEXT, BDEDUCACION, SIBEX and SIREU. In view of the requirements we have developed a system based on a client / server capable of providing solutions that contribute to a better management of information within the dormitory system technology. The flow of information and the work of the dorm were the basis for the development of the proposed system. Which was conceived with the use of the RUP methodology for its wide dissemination and usability. As a modeling tool Visual Paradigm 8.0 was used for development and the IDE Sublime Text, which includes facilities for developing Web applications, including code generation in PHP , Java Script , HTML was used ; languages used in software development . It was used as Management System Database: MySQL. And web server Apache.

Índice

Índice de tablas

Índice de figuras

Introducción

En el mundo contemporáneo las Tecnologías de la Información y las Comunicaciones (TIC), se han ganado un lugar importante en la vida cotidiana del hombre, al ofrecerle numerosas propuestas que garantizan el éxito en la dirección y organización de la información. Las TIC se han convertido en un medio imprescindible para lograr la eficiencia de la gestión de la información en cualquier institución donde se necesita la aplicación de las tecnologías de la información en la actividad diaria. (Leal, 2008)

Al estar presentes en todos los niveles de la actividad y ramas de la sociedad, muchas labores antes realizadas manualmente por el hombre se han sustituido por sistemas informáticos con el objetivo de disminuir el error humano y permitir el almacenamiento de una gran cantidad de información. (Leal, 2008)

Resulta innegable el auge cada vez mayor de las TIC, en las diferentes esferas de la sociedad a escala mundial. Gracias a la puesta en práctica de las mismas el hombre y la sociedad han progresado crecientemente. Tal es el punto, que el impetuoso desarrollo alcanzado incide en transformaciones sociales, económicas y culturales de la sociedad, abre espacios de búsqueda científicas dentro de los elementos que tienen que ver con el desarrollo y como centro, el conocimiento que se genera y aplica en progreso de la humanidad (Leal, 2008).

Nuestro país enfrenta el reto de informatizar su sociedad con vistas a integrarse plenamente a la infraestructura global de la información, así como a hacer uso óptimo de las tecnologías, lo que permitirá lograr incrementos sustanciales en la productividad y el mejoramiento de la calidad y la eficiencia en toda la actividad tanto industrial como de servicios (Dante, 1998).

Desde los primeros momentos del triunfo de la Revolución la Educación constituye un elemento que distinguiría el proceso revolucionario. Hoy un gran número de instituciones educacionales cuentan con sistemas automatizados que tributan al mejoramiento de la calidad en el proceso educativo (Sagaró del Campo & Jiménez, 2007).

Como parte de la política educacional que se ha desarrollado en nuestro país desde años anteriores se va evidenciando que el uso de las Tecnologías de la Información y las Comunicaciones (TIC) tienen que ganar protagonismo con el objetivo de lograr que la información sea más viable y segura(Sagaró del Campo & Jiménez, 2007).

La Universidad de Sancti Spíritus (Uniss), no está exenta de esta política nacional y en ella las TIC

han ganado espacios en la automatización de sus procesos.

La residencia de la Uniss, es la encargada de llevar un control de toda la información referente a los estudiantes becados quienes se encuentran cursando sus estudios en nuestro centro (situación de salud, comportamiento en la residencia estudiantil, etc.), sus trabajadores, así como la disponibilidad de mobiliario con que cuenta dicha entidad.

La utilización de las TIC en las acciones de la Uniss y en particular en la Residencia Estudiantil se mantiene en un nivel aun relativamente bajo, provocando el inadecuado aprovechamiento de las posibilidades que brindan dichas tecnologías. Muchos de los procesos que se llevan a cabo en la residencia, se manejan con el uso de los procesadores de textos, u hojas de cálculo; herramientas que impiden la obtención de interrelaciones entre la información allí manipulada y otras posibilidades que proporcionan los sistemas de gestión automatizada.

Toda la gestión y manipulación de la información relacionada con todos los datos de los estudiantes, ubicación del mobiliario dentro de la residencia, así como los datos personales de los trabajadores que en ella laboran, carece de inmediatez y seguridad, ya que se encuentra guardada en archivos que limitan su disponibilidad y exige además reservar espacios para su almacenamiento y consulta. Esto retarda el proceso de recopilación de información requerido para dar respuesta a todos los organismos y entidades organizativas que estén interesadas en dichos datos, como el Ministerio de Educación Superior (MES), las distintas facultades de la Uniss, así como la máxima dirección de la universidad entre otros interesados.

A partir de lo planteado anteriormente, se impone el siguiente **Problema Científico:** ¿Cómo contribuir al proceso de gestión de información en la Residencia Estudiantil de la Universidad de Sancti Spíritus “José Martí Pérez”?

En correspondencia con lo anterior, se define como **objetivo general**: Desarrollar una Aplicación Web que permita la gestión de la información relacionada con los procesos que tienen lugar dentro de la Residencia Estudiantil de la Universidad de Sancti Spíritus “José Martí Pérez”.

A partir de un análisis del objetivo general se derivan las siguientes **Preguntas Científicas:**

1. ¿Cuáles son los fundamentos teóricos y metodológicos que sustentan la elaboración de una Aplicación Web para la gestión de la información en las residencias estudiantiles universitarias?

2. ¿Cómo diseñar una Aplicación Web que ayude y facilite la gestión de la información en la Residencia Estudiantil de la Universidad de Sancti Spíritus “José Martí Pérez”?

3. ¿Cómo implementar una Aplicación Web para la gestión de la información en la Residencia Estudiantil de la Universidad de Sancti Spíritus “José Martí Pérez”?
Teniendo como **tareas de investigación** las siguientes:

1. Determinación de los fundamentos teóricos y metodológicos que sustentan la elaboración de una Aplicación Web para la gestión de la información en las residencias estudiantiles universitarias.
2. Diseño de una Aplicación Web para la gestión de información en la Residencia Estudiantil de la Universidad de Sancti Spíritus “José Martí Pérez”
3. Implementación de una Aplicación Web para la gestión de la información en la Residencia Estudiantil de la Universidad de Sancti Spíritus“ José Martí Pérez”

Para dar solución a la problemática existente la investigación se estructuró de la siguiente manera:

Una introducción, tres capítulos, conclusiones, recomendaciones, bibliografías y anexos.

A continuación se explica brevemente el contenido de los capítulos:

Capítulo 1: Fundamentación teórica.

En este capítulo se aborda los principales conceptos asociados al dominio del problema. Se describe el objeto de estudio y los sistemas existentes vinculados con el campo de acción, comparando las soluciones existentes con la propuesta. También incluye un estudio sobre las principales tendencias y tecnologías que se pueden utilizar para la solución del problema, así como las tecnologías y la justificación de las herramientas seleccionadas para el análisis, diseño e implementación de la aplicación.

Capítulo 2: Descripción de la Aplicación propuesta.

Se describe el modelo del negocio, identificando los procesos involucrados en él y las reglas que lo rigen. Se realiza la descripción del modelo de casos de uso, identificando y describiendo los actores, trabajadores y casos de uso del negocio mediante el diagrama de casos de uso y el diagrama de actividades. Además de la obtención del modelo del sistema, partiendo de los requerimientos funcionales y no funcionales, utilizando para ello el Lenguaje Unificado de Modelado (UML). Se definen los diferentes tipos de usuario y el rol que desempeña cada uno, describiendo la interacción de los mismos con el sistema a partir de los diagramas de casos de uso y su descripción textual.

Capítulo 3: Construcción de la Aplicación.

En este capítulo se describe la forma en que se realizará la implementación del sistema a través del diagrama de clases del diseño, el diagrama de clases persistentes y el modelo de datos. Se definen los principios de diseño que debe seguir la aplicación, los estándares de codificación y el modelo de implementación mediante el diagrama de despliegue y de componentes.

Capítulo 1: Fundamentación Teórica.

Introducción

El uso de las tecnologías de la información y las comunicaciones es primordial para la expansión y supervivencia de cualquier organización, pues constituye un elemento imprescindible que posibilita un continuo desarrollo; la Residencia Estudiantil de la Universidad de Sancti Spíritus "José Martí Pérez" lograría mejorar la gestión de su información con el desarrollo de una aplicación web que registre la información sobre los estudiantes, trabajadores y medios básicos entre otras de la informaciones que se manejan en dicha institución. El presente capítulo contiene la fundamentación teórica sobre el tema a desarrollar, se realiza un estudio sobre las tecnologías, lenguajes y herramientas existentes determinando cuáles van a ser las utilizadas en el desarrollo del software y que posibilitarán enfrentar exitosamente la situación problemática.

1.1- Definiciones y Conceptos asociados al dominio del problema.

1.1.1- Gestión.

Gestionar es coordinar todos los recursos disponibles para conseguir determinados objetivos, implica amplias y fuertes interacciones fundamentalmente entre el entorno, las estructuras, el proceso y los productos que se deseen obtener. (Bartle, 2009)

1.1.2- Gestión de la información.

La gestión de la información se puede definir como el conjunto de actividades realizadas con el fin de controlar, almacenar y, posteriormente, recuperar adecuadamente la información producida, recibida o retenida por cualquier organización en el desarrollo de sus actividades. (Bartle, 2009)

1.1.3- Lenguaje de Programación.

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes. (Rothberg, 2006)

1.1.4- El Directorio Activo.

El Directorio Activo es la implementación de Microsoft del servicio de directorios LDAP para ser utilizado en entornos Windows.

El AD permite: (García, 2011).

- ✓ A los administradores establecer políticas a nivel de empresa, desplegar programas en muchos ordenadores y aplicar actualizaciones críticas a una organización entera.
- ✓ Almacenar información de una organización en una base de datos central, organizada y accesible.
- ✓ Guardar información relativa a los recursos de red de un dominio.

"Un AD es una estructura jerárquica de objetos. Los objetos se enmarcan en tres grandes categorías. — recursos (p.ej. impresoras), servicios (p.ej. correo electrónico), y usuarios (cuentas, o usuarios y grupos). Proporciona información sobre los objetos, los organiza, controla el acceso y establece la seguridad." (García, 2011).

Bajo este nombre se encuentra realmente un esquema (definición de los campos que pueden ser consultados) LDAP versión 3, lo cual permite integrar otros sistemas que soporten el protocolo. En este LDAP se almacena información de usuarios, recursos de la red, políticas de seguridad, configuración, asignación de permisos, etc.

1.1.5- LDAP.

El LDAP o Protocolo de Acceso Liviano a Directorio (del inglés *Light Directory Access Protocol*) es una versión simplificada del (DAP) o Protocolo de Acceso a Directorio (del inglés *Directory Access Protocol*); es un servicio el cual provee información básica sobre personas o servicios web, así como teléfonos, direcciones de correo, organización, claves públicas de certificados digitales, hash de claves, jerarquización de grupos, etc. Está diseñado para proveer acceso al Directorio X.500 a la vez que no utiliza todos los recursos requeridos por el DAP. Es una parte definida por una base de datos y otra por un protocolo, muy similar al DNS (puede llegar a contener muchos más datos). Optimizado para buscar y soportar replicaciones de una misma información pero dividida entre varios servidores. Básicamente la habilidad de utilizar el LDAP, es realizar todas las acciones de acceso dado un usuario especifico, el cual debe y tiene permisos sobre distintas infraestructuras, es por ello que dado que se tiene la clave pública ingresada como parámetro del árbol del LDAP, fácilmente se podría utilizar este para verificar, para distintas infraestructuras la autenticación y autorización sobre recursos o datos. (Van de Graaf, 2005).

Ventajas de LDAP: (Molina & Delgado, 2008).

- ✓ Acceso al mismo desde casi cualquier plataforma de computación, desde cualquier número creciente de aplicaciones.

- ✓ Fácil personalizar tus aplicaciones internas de empresa para añadirles soporte LDAP.
- ✓ Es utilizable por distintas plataformas y basado en estándares, de ese modo las aplicaciones no necesitan preocuparse por el tipo de servidor en que se hospeda el directorio.
- ✓ La mayoría de los servidores LDAP son simples de instalar, fácilmente mantenibles, y fácilmente optimizables.
- ✓ Los servidores LDAP pueden replicar tanto algunos de sus datos como todos a través de métodos de envío o recepción, lo que permite enviar datos a oficinas remotas, incrementar tu seguridad y demás. La tecnología de replicación está incorporada y es fácil de configurar. Por contraste, muchos de los vendedores de DBMS cobran un extra por esta característica, y es bastante más difícil de gestionar.
- ✓ Permite delegar con seguridad la lectura y modificación basada en autorizaciones según tus necesidades utilizando ACIs (colectivamente, una ACL, o Lista de Control de Acceso por sus siglas en inglés).

1.1.6- Sistema de gestión de contenido (CMS).

Los sistemas de gestión de contenidos (Content Management Systems o CMS) son software que se utilizan principalmente para facilitar la gestión en la web, ya sea en Internet o en una Intranet, y por eso también son conocidos como gestores de contenidos web (Web Content Management o WCM). Hay que tener en cuenta, sin embargo, que la aplicación de los CMS no se limita sólo a la web.

1.1.7- Metodología.

Una metodología de desarrollo de software es un conjunto de pasos y procedimientos que deben seguirse para desarrollar software. Una metodología está compuesta por: ¿Cómo dividir un proyecto en etapas?, ¿Qué tareas se llevan a cabo en cada etapa?, ¿Qué restricciones deben aplicarse?, ¿Qué técnicas y herramientas se emplean? y ¿Cómo se controla y gestiona un proyecto?

1.2- Objeto de Estudio.

1.2.1- Descripción del objeto de estudio.

La Residencia Estudiantil de la Universidad de Sancti-Spíritus está encaminada a reforzar la convivencia, educación formal, cuidado de la propiedad colectiva, hábitos de orden, limpieza, disciplina y protagonismo estudiantil en la solución de problemas de carácter colectivo que contribuyan favorablemente a la formación integral de los becarios, así como elevar

cualitativamente el trabajo educativo y político-ideológico de los becarios. Todo esto protagonizado por un profundo trabajo en unión con la FEU y la dirección de la Residencia Estudiantil, para convertir a dicha entidad en una Comunidad Universitaria.

1.3.1- Objetivos estratégicos de la organización.

La residencia estudiantil tiene como principal proceso la: ***gestión de la información de los estudiantes becados, de sus trabajadores así como de la ubicación y uso de su mobiliario.***

Cuando un estudiante ingresa a la universidad en alguna de las carreras que en ellas se estudia. Se dirige a la Residencia Estudiantil. La especialista verifica sus datos generales y posteriormente se le da la ubicación en uno de los locales existentes.

El estudiante está sujeto a un proceso de control en cual es monitoreado por dichos especialistas. Estos procesos son:

- Registrar faltas cometidas por los becarios: Ante alguna falta de carácter leve, grave, menos grave o muy grave, la dirección de la Residencia Estudiantil, conjuntamente con el consejo de la FEU de la Beca, levantaran un proceso investigativo con el objetivo de efectuar un señalamiento o aplicar una sanción según el Reglamento Disciplinario para las Residencias Estudiantiles, amparado por las resoluciones No. 34/2000 y 38/2000 respectivamente.

- Registrar Evaluación en Beca del estudiante. Dicha evaluación es registrada en la Oficina de Trabajo Educativo de la Residencia Estudiantil y es solicitada por el especialista para ser registrada, puesto que reviste gran importancia, ya que la misma tributa al proceso de integralidad del estudiante.

Todos estos procesos descritos son registrados, consultados y manejados por el especialista en la elaboración de reportes que son solicitados por las instancias superiores. (Rectorado, MES, Embajadas, entre otras).

1.3- Descripción de los sistemas existentes.

A continuación se presentan los sistemas informáticos que han sido desarrollados hasta la actualidad y que están relacionados con el campo de acción

1.3.1- SACBEXT: Sistema Automatizado de control de Becarios Extranjeros.

Fue desarrollado durante los años 1998-2000, para la manipulación de información de Estudiantes Extranjeros en la ONABE. El sistema gestiona en un alcance general la

información de los estudiantes a nivel nacional, permitiendo así la gestión de esta información .SACBEXT logró mejorar el proceso de control y la rapidez en la toma de decisiones por la ONABE.

El sistema se destaca en las posibilidades brindadas para la obtención de resúmenes y reportes de la información almacenada y el envío por correo de los reportes generados por el sistema. Las exigencias presentes en ese período requirieron una gestión generalizada y resumida de la información pues SACBEXT no profundizó en los parámetros que podrían diferenciar y a la vez identificar con más precisión a los estudiantes extranjeros. La accesibilidad al sistema es limitada, ya que los Centros de Estudio donde radican los estudiantes y donde son mejor conocidos no tienen acceso directo a esta herramienta por cuestiones de seguridad que no fueron implementadas, aunque en general se puede afirmar que SACBEXT ha dado buenos resultados y ha estado en uso hasta la actualidad.

1.3.2- BDEDUCACION: Base de Datos Educación.

Proyecto que surgió con la intención de ofrecer mejoras a SACBEXT y que comenzó en el año 2003. Con el objetivo de perfeccionar el proceso de gestión, BDEDUCACION rediseñó la estructura de la Base de Datos de Estudiantes Extranjeros (BDEE), introduciendo nuevos campos de información y algunos módulos. Estas modificaciones mejorarían el funcionamiento del sistema y darían nuevos resultados que permitirían una mejora en la manipulación de la información. Sin embargo, no se logró terminar el proyecto pues se estancó el proceso de corrección de los problemas existentes en SACBEXT y el sistema no fue culminado.

1.3.3- SiBEX: Sistema de Gestión de la Información de Estudiantes Extranjeros.

Este proyecto se desarrolló en el año 2006 en la Universidad de Cienfuegos por una estudiante de la facultad de informática con el objetivo de llevar el control de los estudiantes extranjeros que cursaban estudios en dicha universidad. El sistema facilita la gestión de la información de modo que todo queda almacenado en una base de datos con la cual se puede interactuar mediante una interfaz Web con el objetivo de dar mayor seguridad y fiabilidad a la información que se procesa en la residencia estudiantil. Sibex no se encuentra en explotación por limitar las posibilidades a solo un grupo de estudiantes (estudiantes extranjeros) y no incluir otras facilidades de trabajo como es el control del mobiliario y los datos de los trabajadores.

1.3.4- SIREU: Sistema Informático para la Residencia Estudiantil Universitaria.

Este proyecto se desarrolló en el año 2009 en la Universidad de Cienfuegos por una estudiante

de la facultad de informática con el objetivo de llevar el control de los estudiantes, los trabajadores y el mobiliario que allí se encontraba, a través de una interfaz Web mediante la cual se interactuaba con la base de datos.

El sistema propuesto es superior a sus antecesores pues mejora y agiliza el proceso de gestión de información de la Residencia Estudiantil de la Universidad de Sancti-Spíritus "José Martí Pérez", almacena la información de forma segura y fiable a la cual se accede desde una interfaz Web, además está diseñada para tener un registro actualizado de la información referente a los estudiantes becados(evaluación, desempeño docente), los trabajadores y los medios básicos, entre otros datos que allí se manejan.

También en su diseño e implementación se prevé la actualización y ampliación de las utilidades del sistema sin tener que realizar grandes modificaciones en su código fuente.

1.4- Metodologías.

1.4.1- Proceso Unificado de Desarrollo (RUP).

El Proceso Unificado de Desarrollo, también conocido como Proceso Unificado de Rational (RUP, Rational Unified Process), es un proceso de desarrollo de software que utiliza UML, constituye una metodología estándar utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

Entre las características distintivas de esta metodología se pueden mencionar:

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo)
- Pretender implementar las mejores prácticas en ingeniería de Software
- Desarrollo iterativo
- Administración de requisitos
- Uso de arquitectura basada en componentes
- Control de cambios
- Modelado visual del software
- Verificación de la calidad del software

RUP es un producto de Rational (IBM). Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso). RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al concluir cada ciclo, en cada ciclo se analizan las fases (ver Figura.2) siguientes:

Inicio: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos.

Elaboración: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos

Construcción: se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario

Transición: se implementa el producto en el cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requerimientos a ser analizados. (Zaguero, "Administración de Proyectos de Software. Trabajo práctico grupal: Ciclos de Vida de proyectos. Grupo 4. Ciclo vida RUP , Mar. 2008; http://www.zohowriter.com/public/27201/38205.)

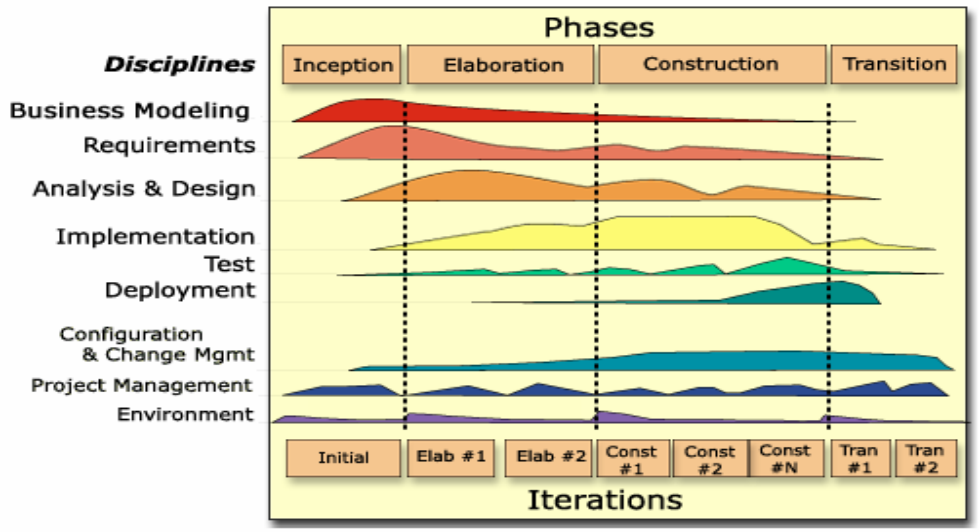


Fig 2: Fases de RUP.

Para la elaboración de los diagramas y otros modelos que propone RUP, se hará uso del Visual Paradigm, herramienta CASE 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.

1.4.2- Lenguaje de Modelación Unificado (UML).

El Lenguaje Unificado de Modelado Unificado (UML - Unified Modeling Language) es un lenguaje que permite modelar, construir y documentar los elementos que forman un producto de software que responde a un enfoque orientado a objetos. Este lenguaje fue creado por un grupo de estudiosos de la Ingeniería de Software formado por: Ivar Jacobson, Grady Booch y James Rumbaugh en el año 1995. Desde entonces, se ha convertido en el estándar internacional para definir organizar y visualizar los elementos que configuran la arquitectura de una aplicación orientada a objetos. Con este lenguaje, se pretende unificar las experiencias acumuladas sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. UML no es un lenguaje de programación sino un lenguaje de propósito general para el modelado orientado a objetos y también puede considerarse como un lenguaje de modelado visual que permite una abstracción del sistema y sus componentes. Entre sus objetivos fundamentales se encuentran:

- Ser tan simple como sea posible, pero manteniendo la capacidad de modelar toda la gama de sistemas que se necesita construir.
- Necesita ser lo suficientemente expresivo para manejar todos los conceptos que se originan en un sistema moderno, tales como la concurrencia y distribución, así como también los mecanismos de la ingeniería de software, como son el encapsulamiento y los componentes.
- Debe ser un lenguaje universal, como cualquier lenguaje de propósito general.
- Imponer un estándar mundial. (Xavier Ferrá Grau, “Desarrollo orientado a objetos con UML,” Mar. 2008; http://www.clikear.com/manuales/uml/introduccion.asp.)

1.5- Tecnologías.

1.5.1- Sistemas gestores de contenido. CMS.

En la actualidad existen varios CMS que se encargan de la gestión de contenido, además de ser muy fácil cambiar "en cuestión de minutos" todo el diseño del sitio gracias al sistema de plantillas que utiliza, las cuales se instalan y luego se seleccionan desde la interfaz administrativa, la que se desea usar por parte del administrador o del cliente. También permite publicar y administrar los

contenidos principales como: novedades, artículos, títulos, textos e imágenes, que se pueden editar desde un sencillo editor HTML, que permitirá formatear los textos con los estilos deseados en forma similar de como se haría en un editor de texto.

1.5.1.1- Joomla.

Joomla es un sistema de contenido premiado de gestión (CMS), que le permite construir sitios web y aplicaciones en líneas potentes, además de otros aspectos como su facilidad de uso y extensibilidad. Lo mejor de todo, es que Joomla es una solución de código abierto que está disponible gratuitamente para todos en la red. (7)

1.5.1.2- Drupal.

Drupal es un sistema de administración de contenido para sitios web. Permite publicar artículos, imágenes, u otros archivos y servicios añadidos como foros, encuestas, votaciones, blogs y administración de usuarios y permisos. Drupal es un sistema dinámico: en lugar de almacenar tus contenidos en archivos estáticos en el sistema de ficheros del servidor de forma fija, el contenido textual de las páginas y otras configuraciones son almacenados en una base de datos y se editan utilizando un entorno web incluido en el producto. (8)

1.5.1.3- osCommerce.

osCommerce es una aplicación web de código abierto (Open Source) que permite montar una tienda virtual en cuestión de minutos para vender en Internet. Consta de dos partes: principalmente el front y el back-end, es decir, la parte que todos pueden ver, la tienda virtual en sí y la parte de administración, donde podrás mantener tu propia tienda virtual, actualizando productos, insertando nuevas ofertas, categorías, idiomas, monedas, consultar los pedidos, los clientes y sin coste ninguno por parte del vendedor y sin necesidad alguna de saber programación. (9)

Tecnología a utilizar:

Después de estudiar cada uno de los CMS antes expuestos se escoge Drupal 7.23 por ser la herramienta líder en la creación de las aplicaciones web, por sus disímiles características y facilidades en la gestión de contenido. Es un programa de código abierto, con licencia GNU/GPL, escrito en PHP, desarrollado y mantenido por una activa comunidad de usuarios. Destaca por la calidad de su código y de las páginas generadas, el respeto de los estándares de la web, y un énfasis especial en la usabilidad y consistencia de todo el sistema, se puede usar gratuitamente, sus componentes y módulos están abiertos para reutilizar su código con fines de mejoras y nuevos aportes. Además de ir ampliando las funcionalidades de los portales con nuevas opciones tales como: galerías de imágenes y videos, tiendas virtuales, foros, multi-idioma.

1.5.2- Herramientas de modelado.

1.5.2.2- Visual Paradigm 8.0.

Visual Paradigm es una herramienta UML para el Lenguaje de Modelado de Unificado (VP-UML), soporta los últimos estándares de la notación y está diseñada para una gama muy amplia de usuarios tales como: arquitectos del sistema, ingenieros de software, analistas del sistema, analistas del negocio y cualquier otro que esté interesado en la construcción de un sistema de software, además es un lenguaje de modelado orientado a objeto.

Visual Paradigm es una herramienta CASE: Ingeniería de Software Asistida por Computación. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. El mismo ha sido concebido para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Constituye una herramienta de software libre de probada utilidad para el analista. Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque Orientado a Objetos. (17)

Características:

- Capacidades de ingeniería directa e inversa. Licencia: gratuita y comercial.
- Soporta aplicaciones Web.
- Diagramas de flujo de datos.
- Ingeniería de ida y vuelta. Diagramas de flujo de datos. Generación de base de datos -Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Ingeniería inversa de base de datos: Desde Sistemas Gestores de Base de Datos (DBMS) existentes a diagramas de Entidad-Relación.
- Editor de figuras. (17)

1.5.2.1- Rational Rose.

Es una herramienta CASE (Computer–Arded Software Engineering), traducido al español como Ingeniería Asistida por Computadora, desarrollada por Rational Corporation basada en el Lenguaje Unificado de Modelación (UML), que permite crear los diagramas que se van generando durante el proceso de Ingeniería en el desarrollo del software. Las personas que

desarrollaron el Proceso Unificado del Rational (RUP) son miembros de Rational Corporation y brinda muchas facilidades en la generación de la documentación del software que se esté desarrollando, además de que posee un gran número de estereotipos predefinidos que facilitan el proceso de modelación del software. (17)

Características:

- Permite desarrollo multiusuario.
- Genera documentación del sistema.
- Disponible en múltiples plataformas. (18)

En la selección de la herramienta de modelado se estudiaron profundamente las dos antes mencionadas y se decidió escoger Visual Paradigm 6.4 ya que esta es multiplataforma al igual que el Rational Rose pero con la gran diferencia que es creada bajo la licencia gratuita y comercial, es de software libre, soporta varios idiomas y aplicaciones web, su diseño es centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad y fácil de instalar y actualizar.

1.5.3- Servidor de Aplicaciones Web.

1.5.3.1- Apache.

El sistema se despliega sobre el servidor de aplicaciones Apache. El paradigma más extendido para desarrollar servidores de información mediante aplicaciones de código abierto y gratuitas es el conocido como LAMP: Linux, Apache, MySql y PHP/PERL/Python. Los aspectos que hacen interesante el uso de estas herramientas para la publicación de información en Internet es fundamentalmente el ahorro de costes, ya que todos los programas y sistemas son gratuitos. La versatilidad de las aplicaciones permite crear configuraciones a la medida de cada sistema, ya que todos ellos cuentan con la posibilidad de ampliaciones y módulos que complementan o añaden beneficios al sistema.

La existencia de gran cantidad de aplicaciones y módulos adicionales gratuitos, que ayudan a mejorar la gestión y el acceso. La posibilidad de acceder de forma segura a la información, definiendo distintos niveles de control es otra de las ventajas fundamentales de este tipo de servidor de aplicaciones.

1.5.4- Sistemas Gestores de Bases de Datos.

Una Base de Datos (BD) es un conjunto de datos interrelacionados, almacenados con carácter

más o menos permanente en la computadora, puede ser considerada una colección de datos variables en el tiempo.

Un **Sistema Gestor de Base de Datos** (SGBD) 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.(Rosa María Matos, “Introducción al trabajo con Base de Datos. Asignatura de Sistemas de Gestión de Base de Datos,” 2004.)

El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado. En el mercado existen un sinnúmero de gestores de base de datos, todos con sus características que lo hacen una opción a escoger, pero la elección, la mayoría de las veces, depende del cliente y no del desarrollador.

A continuación mostramos una tabla (Tabla 1) comparativa de algunos de los SGBD en cuanto a si son Multiplataforma o no, Gratis o no y en cuanto a la disponibilidad del código fuente. Entre los sistemas de gestión de base de datos más utilizados en la capa de datos se encuentran SQL Server, MySQL, PostgreSQL, Oracle. (Riveros, 2008)

	Multiplataforma	Gratis	Código Fuente
MSSQL	No	No	No
MSDE	No	Si	No
MySQL	Sí	Si	Sí
Postgres	Sí	Si	Sí
Firebird	Sí	Si	Si
Interbase	Si	No	Sí
SyBase	Si	No	No
INFORMIX	Sí	No	No
Oracle	Sí	no	No

Tabla 1: Comparación entre SGBD

1.5.4.1- MySql-Server.

Se decide utilizar como Sistema Gestor de Base de Dato (SGBD) MySql-Server. Es un SGBD relacional desarrollado como software libre bajo un esquema de licencia dual. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas

empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. En la actualidad MySql constituye uno de los SGBD muy confiable, seguro y de gran rapidez, posee gran cantidad de librerías que facilitan su instalación y configuración, estas características lo hacen uno de los sistemas de mayor aceptación y altamente conveniente para la gestión de datos en grandes y pequeños volúmenes.

Además de hacer uso del SGBD MySql se necesita utilizar una Base de Dato Sql para acceder a parte de la información requerida, este contacto se realiza a base de consultas hechas a la misma, teniendo en cuenta que el mismo está estrechamente vinculado con el CMS Drupal. (Riveros, 2008)

1.5.4.2- PostgreSQL.

PostgreSQL es un SGBD relacional orientada a objetos y libre, publicado bajo la licencia BSD (Berkeley Software Distribution). (Riveros, 2008) PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas características que tradicionalmente sólo se podían ver en productos comerciales de alto calibre. (Riveros, 2008) PostgreSQL posee una serie de características:

Alta concurrencia.

Mediante un sistema denominado MVCC (Acceso concurrente multiversión, por sus siglas en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos.

Amplia variedad de tipos nativos.

PostgreSQL provee nativamente soporte para:

- Números de precisión arbitraria.
- Texto de largo ilimitado.
- Figuras geométricas (con una variedad de funciones asociadas)
- Direcciones IP (IPv4 e IPv6).
- Bloques de direcciones estilo CIDR.
- Direcciones MAC.

- Arrays.

Otras características.

- Claves ajenas también denominadas Llaves ajenas o Claves Foráneas (foreignkeys).
- Disparadores (triggers): Un disparador o trigger se define en una acción específica basada en algo ocurrente dentro de la base de datos. En PostgreSQL esto significa la ejecución de un procedimiento almacenado basado en una determinada acción sobre una tabla específica.

Las funciones son bloques de código que se ejecutan en el servidor. Pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos da, desde las operaciones básicas de programación, tales como bifurcaciones y bucles, hasta las complejidades de la programación orientada a objetos o la programación funcional.

(The PostgreSQL Global Development Group, 2003) PostgreSQL posee varios lenguajes para programar las funciones. Uno de estos lenguajes es el PL/pgSQL, muy parecido al PL/SQL de Oracle. (The PostgreSQL Global Development Group, 2003)

SQL es el lenguaje que PostgreSQL (y la mayoría del resto de bases de datos relacionales) usa como lenguaje de consultas. Es portable y fácil de aprender. Pero cada estamento SQL debe ser ejecutado individualmente por el servidor de bases de datos. Esto significa que su aplicación cliente debe enviar cada consulta al servidor de bases de datos, esperar a que se procese, recibir el resultado, realizar alguna computación, y luego enviar otras consultas al servidor. Todo esto incurre en una comunicación entre procesos y también puede sobrecargar la red si su cliente se encuentra en una máquina distinta al servidor de bases de datos. (The PostgreSQL Global Development Group, 2003) Con PL/pgSQL puede agrupar un grupo de computaciones y una serie de consultas dentro del servidor de bases de datos, teniendo así la potencia de un lenguaje procedural y la sencillez de uso del SQL, pero ahorrando una gran cantidad de tiempo porque no tiene la sobrecarga de una comunicación cliente/servidor. Esto puede redundar en un considerable aumento del rendimiento. (The PostgreSQL Global Development Group, 2003) PL/pgSQL añade a la potencia de un lenguaje procedural la flexibilidad y sencillez del SQL. Con PL/pgSQL puede usar todos los tipos de datos, columnas, operadores y funciones de SQL. (The PostgreSQL Global Development Group, 2003) Debido a que las funciones PL/pgSQL corren dentro de PostgreSQL, estas funciones funcionarán en

cualquier plataforma donde PostgreSQL corra. Así podrá reutilizar el código y reducir costes de desarrollo. (The PostgreSQL Global Development Group, 2003)

Entre las facilidades que brinda PostgreSQL podemos mencionar: la restauración continúa de la base de datos, es decir, puedes volver a un punto concreto. Es de suponer que esto representa una carga más para el sistema, pero es una opción interesante. Por otro lado ofrece mejoras de rendimiento y decisiones sobre el sistema de ficheros donde quieres guardar cosas, cambio de tipos de campo con alter table, entre otras. PostgreSQL, no tiene costo asociado a la licencia del software. Ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características, estabilidad y rendimiento. PostgreSQL tiene una legendaria confiabilidad y estabilidad y es fácil de administrar. (The PostgreSQL Global Development Group, 2003) (Riveros, 2008)

Por las características mostradas anteriormente se seleccionó MySQL como

SGBD en este proyecto.

1.5.4.3- MySQL.

MySQL surgió alrededor de la década del 90, creada por la empresa sueca MySQL AB. MySQL es un gestor de base de datos sencillo de usar e increíblemente rápido. También es uno de los motores de base de datos más usados en Internet. (Riveros, 2008)

Características importantes del Software de Base de Datos MySQL:

- Trabaja en múltiples plataformas.
- Posee numerosos tipos de datos: enteros con y sin signo, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, y ENUM.
- Records de longitud fija y variable.
- Soporte de operadores y funciones completo en las clausulas SELECT y WHERE de las consultas.
- Soporte para las cláusulas SQL: GROUP BY y ORDER BY. Además de funciones de agrupación (COUNT (), AVG (), STD (), SUM (), MAX (), MIN (), and GROUP_CONCAT ()). (5.0, 2011) (Riveros, 2008)

1.5.5- Sublime Text.

Sublime Text es un editor de código multiplataforma, ligero y con pocas concesiones a las florituras. Es una herramienta concebida para programar sin distracciones. Su interfaz de color oscuro y la riqueza de coloreado de la sintaxis, centra nuestra atención completamente.

Sublime Text permite tener varios documentos abiertos mediante pestañas, e incluso emplear varios paneles para aquellos que utilicen más de un monitor. Dispone de modo de pantalla completa, para aprovechar al máximo el espacio visual disponible de la pantalla.

1.6- Lenguajes.

Lado del servidor

1.6.1- PHP

PHP acrónimo recursivo de "PHP: Hypertext Preprocessor" (Preprocesador de Hipertexto), es un lenguaje de programación interpretado, con licencia OpenSource. Fue originalmente diseñado en Perl, seguido por la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador Danés-Canadiense Rasmus Lerdorf en el año 1994 para mostrar su currículum vital y guardar ciertos datos, como la cantidad de tráfico que su página Web recibía. (Munz, 2011)

Su interpretación y ejecución se da en el servidor en el cual se encuentra almacenada la página, el cliente solo recibe el resultado de la ejecución. Cuando el cliente hace una petición al servidor para que le envíe una página Web, enriquecida con código PHP, el servidor interpretará las instrucciones mezcladas en el cuerpo de la página y las sustituirá con el resultado de la ejecución antes de enviar el resultado a la computadora del cliente. Permite el uso de las técnicas de Programación Orientada a Objetos. El código PHP se incluye entre etiquetas especiales de comienzo y final que nos permitirán entrar y salir del modo PHP.

PHP tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos tales como UNIX, Linux, Windows y Mac OS X, y puede interactuar con los servidores de Web más populares. Además permite la conexión a numerosas bases de datos de forma nativa tales como: MySQL, Postgres, Oracle, ODBC, IBM DB2, Microsoft SQL Server y SQLite.

Lado del cliente

1.6.2- JavaScript

Es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario.

Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. (Barzanallana, 2012)

1.6.3- HTML

HTML, no es un lenguaje de programación, es un lenguaje de especificación de contenidos para un tipo específico de documentos. Es decir, mediante HTML se puede especificar, usando un conjunto de etiquetas o tags, cómo va a representarse la información en un navegador o browser. Se centra en la representación en la pantalla de la información.

HTML es un lenguaje muy sencillo que permite describir hipertexto, es decir, texto presentado de forma estructurada y agradable, con *enlaces* (*hyperlinks*) que conducen a otros documentos o fuentes de información relacionadas, y con *inserciones* multimedia como gráficos y sonidos. Contiene varias etiquetas (tags) las cuales son utilizadas por los desarrolladores para especificar la estructura lógica del contenido (títulos, párrafos de texto normal, enumeraciones, definiciones, citas, etc) así como los diferentes efectos que se quieren dar, tales como especificar los lugares del documento donde se debe poner cursiva, negrita, o un gráfico determinado.

Además el lenguaje HTML, permite a los desarrolladores crear documentos que pueden ser interpretados en ordenadores que tengan diferentes sistemas operativos. El HTML 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. Las marcas más utilizadas suelen describirse por textos descriptivos encerrados entre signos de "menor" (<) y "mayor" (>), siendo lo más usual que exista una marca de principio y otra de final. (Rodriguez J. , 2009)

1.6.4- CSS (Hojas de estilo en cascada).

Las Hojas de Estilo en Cascada o CSS constituyen un lenguaje sencillo que complementa el de HTML, suponiendo un apoyo fundamental a la hora de diseñar páginas Web, porque permiten una mayor precisión en el ajuste de los elementos de diseño. Esta técnica consiste en separar el diseño del contenido, de manera que las indicaciones para conformar el diseño se agrupan en una hoja de estilo o archivo fuera del contenido del documento de la página HTML. Lo que hace fundamentalmente el código de las hojas de estilos es transformar las etiquetas del lenguaje HTML y conformarlas a las características que se quiera darle; pero también, y esto es lo importante, con este código se pueden crear etiquetas nuevas, que se introducen dentro del documento. Una de las ventajas de las hojas de estilos es que se puede modificar algunas características de todos los documentos de un sitio Web desde un archivo, sin tener que modificarlas en cada uno de los documentos. (Valero, 2009)

1.7- Arquitectura de desarrollo de N Capas.

Subdividir una aplicación en partes lógicas es un detalle muy provechoso. Dividir software de gran tamaño en partes más pequeñas puede hacer más simples los procesos de generarlo, reutilizarlo y modificarlo. También puede ser útil para acomodar diferentes tecnologías o diferentes organizaciones de negocio. No obstante pueden considerarse otras alternativas. La modularidad y la reusabilidad son aspectos útiles, pero pueden dar lugar a aplicaciones que no sean tan seguras, manejables o rápidas como podrían ser de otro modo.

Un ejemplo de la factorización de una aplicación es el modelo de N Capas, el cual constituye un modelo mejorado desde la perspectiva de más de dos capas, llamadas también niveles. Aunque, algunas veces, los niveles residen físicamente en máquinas diferentes debe enfatizarse en la distribución lógica de los mismos. Los nombres de estos niveles difieren de acuerdo a la fuente, no obstante es bastante extendido el uso de las siguientes referencias en el modelo de 3 capas.

1. La capa de presentación. Generalmente es una interfaz gráfica que muestra los datos a los usuarios.

2. La capa de la lógica de negocios. Es responsable de procesar los datos recuperados y enviarlos a la capa de presentación.

3. La capa de datos. Almacena los datos de la aplicación en un almacén persistente, tal como una

base de datos relacional o archivos XML. (Cordero Carrasco, 2009)

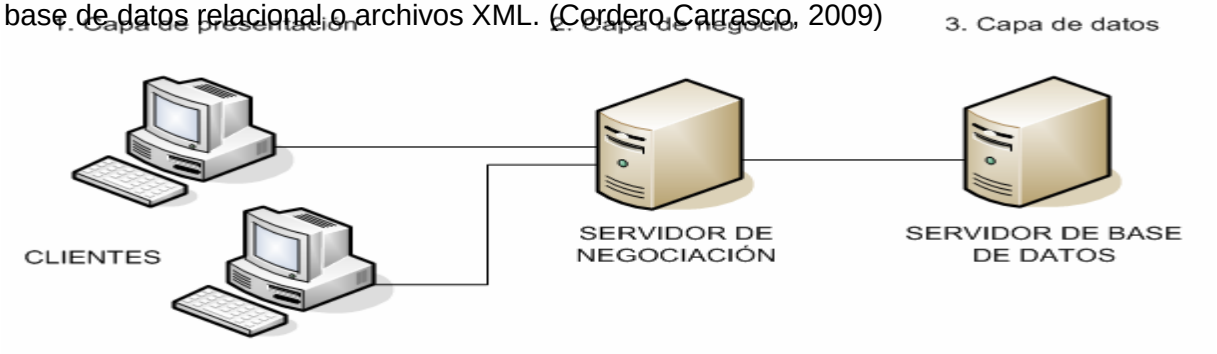


Figura 1: Arquitectura de tres capas.

1.8- Conclusiones.

En este capítulo se explica todo lo referente al proceso que tiene lugar dentro de la Residencia Estudiantil. Así como la necesidad de automatizar la gestión de la información dentro de esta. Por lo que analizando las tendencias, metodologías y tecnologías actuales, que se expusieron en este capítulo, se decidió utilizar: metodología RUP y el lenguaje UML, por las ventajas que proporcionan y el alto nivel de aceptación que han tenido, MySql Server se seleccionó como SGBD por su característica de ser Open Source y tener una buena seguridad en su información. Se escogieron Visual Paradigm 6.4 para el modelado y el IDE Sublime Text para la implementación de la aplicación Web dinámica con HTML, CSS y JavaScript del lado del cliente y PHP del lado del servidor. Como servidor se utilizó Apache.

Capítulo 2: Descripción de la Aplicación propuesta.

Introducción

En este capítulo se realiza un estudio del modelo del negocio, describiéndose la gestión de información en la Residencia Estudiantil Universitaria de la Universidad de Sancti Spíritus "José martí Pérez"; este proceso permite una mejor comprensión de la problemática. Se exponen las reglas del negocio a respetar para el diseño de la aplicación, se definen los actores del negocio, los trabajadores y los casos de uso que permitirán conformar el diagrama de casos de uso del negocio. Se realizan diagramas de actividades, modelo de objetos y expansiones de los casos de uso. Se plantean los requerimientos funcionales y los no funcionales, que no son más que las necesidades de los clientes y los usuarios finales. Además se muestran los casos de uso del sistema, actores, diagramas de casos de uso del sistema así como la descripción de cada uno según lo propuesto por la metodología RUP.

2.1- Descripción del modelo de negocio

Modelar e identificar el flujo de los procesos que serán objeto de automatización de un sistema informático, es un elemento clave para lograr un desarrollo exitoso del producto y una buena comunicación entre los desarrolladores, los clientes y el usuario final. A este flujo de trabajo se le denomina: Modelo del Negocio (Ferrá Grau, 2010) y permite comprender los problemas actuales de la organización y garantizar que los consumidores, usuarios finales y desarrolladores tengan un entendimiento común de la organización. El modelado del negocio está soportado por 2 tipos de modelos de UML: modelos de casos de uso y modelo de objetos (Ferrá Grau, 2010).

Seguidamente se describe el proceso de negocio que se lleva a cabo en la Residencia Estudiantil de la Universidad de Sancti Spíritus "José Martí Pérez", mediante los artefactos propuestos por la metodología RUP y modelados por el lenguaje UML.

2.2- Identificación de los procesos del negocio

Para iniciar el modelo del negocio debemos capturar y definir los procesos de negocio de la organización que se estudia. Los procesos de negocio son un grupo de tareas relacionadas lógicamente que se llevan a cabo en una determinada secuencia y forma, y que emplean los recursos de la organización para dar resultados que apoyen sus objetivos. (Ferrá Grau, 2010).

Basados en el concepto anterior se identificó el siguiente proceso de negocio:

- Solicitar beca: Proceso mediante el cual el estudiante solicita una beca para permanecer en la Residencia Estudiantil.
- Solicitar cambio de cuarto: Proceso mediante el cual el estudiante solicita que se le traslade de cuarto dentro de la Residencia Estudiantil.
- Solicitar baja: Proceso mediante el cual el estudiante solicita la baja de la Residencia Estudiantil.
- Solicitar información: Proceso mediante el cual se le solicita información a la Residencia Estudiantil, referente a sus trabajadores, mobiliarios y estudiantes.

2.3- Reglas del negocio a considerar.

Las reglas del negocio describen políticas que deben cumplirse o condiciones que deben satisfacerse, por lo que regulan algún aspecto del negocio. (Ferrá Grau, 2010)

Después de identificar los procesos de negocio se definen las siguientes reglas del negocio:

- El estudiante puede solicitar residencia una vez matriculado en alguna carrera de la universidad y solo así se le asigna un cuarto.
- La solicitud de cambio de cuarto es autorizada por la dirección de la Residencia Estudiantil.
- La solicitud de baja de la residencia es autorizada por la dirección de la Residencia Estudiantil.
- El estudiante solo se puede asignar a un solo local dentro de la Residencia Estudiantil.
- Cuando las instancias superiores solicitan alguna información, esta se considera de carácter urgente y debe ser elaborada en un margen de tiempo relativamente corto.
- Un mobiliario solo puede pertenecer a un local dentro de la Residencia Estudiantil.

2.4- Modelo de casos de uso del negocio.

El modelo de Casos de Uso del Negocio es un modelo que describe los procesos de negocio de una empresa en términos de casos de uso y actores del negocio en correspondencia con los procesos del negocio y los clientes, respectivamente. El modelo de casos de uso del negocio presenta un sistema (en este caso, el negocio) desde la perspectiva de su uso y esquematiza cómo proporciona valor a sus usuarios. (James, 2004)

El modelo de Casos de Uso del Negocio es definido a través de tres artefactos: el diagrama

de casos de uso del negocio, la descripción de los casos de uso del negocio y el diagrama de actividades de cada caso de uso del negocio.

2.4.1- Actores del negocio.

Un actor del negocio es cualquier individuo, grupo, entidad, organización, máquina o sistema de información externos; con los que el negocio interactúa. Lo que se modela como actor es el rol que se juega cuando se interactúa con el negocio para beneficiarse de sus resultados. (James, 2004)

La tabla muestra los actores y una descripción de su rol:

Actores del negocio	**Justificación**
Estudiantes	Es quien se beneficia e inicia la mayoría de los procesos del negocio, como la solicitud de beca, el registro de los datos personales, la solicitud de traslado de cuarto, la solicitud de la baja de la residencia estudiantil.
Directivo	Es el que inicia y se beneficia del proceso de solicitud de información a la residencia estudiantil.

Tabla 2: Actores del negocio

2.4.2- Diagramas de casos de uso del negocio.

Un proceso de negocio es un grupo de tareas relacionadas lógicamente que se llevan a cabo en una determinada secuencia y manera y que emplean los recursos de la organización para dar resultados en apopo a sus objetivos. Un caso de uso del negocio representa a un proceso de negocio, por lo que se corresponde con una secuencia de acciones que producen un resultado observable para ciertos actores del negocio. Desde la perspectiva de un actor individual, define un flujo de trabajo completo que produce resultados deseables. (Ferrá Grau, 2010)

Para comprender los procesos de negocio se construye el diagrama de casos de uso del negocio en el que aparece cada proceso del negocio relacionado con su actor.

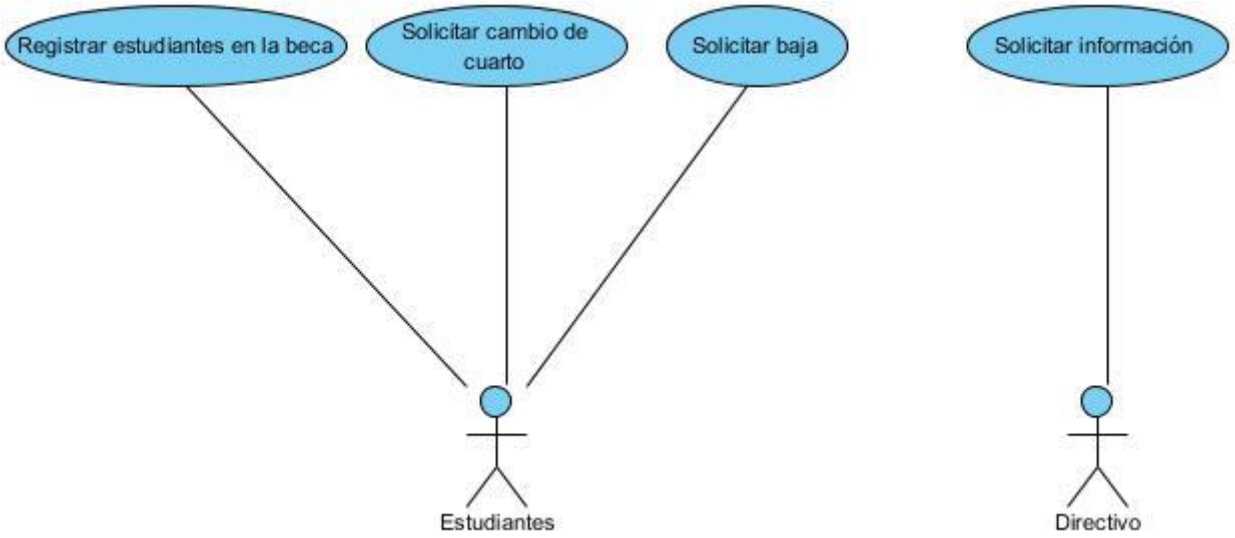


Figura 2: Modelo de casos de uso del negocio.

2.4.3 – Trabajadores del negocio.

Un trabajador del negocio es una abstracción de una persona (o grupo de personas), una máquina o un sistema automatizado; que actúa en el negocio realizando una o varias actividades, interactuando con otros trabajadores del negocio y manipulando entidades del negocio. Representa un rol. (James, 2004) Basado en el concepto se define el trabajador del negocio, como se muestra en la tabla siguiente:

Trabajadores del negocio	**Justificación**
Director Beca	Encargado de registrar a los estudiantes en la beca, cambio de cuarto de los estudiantes y baja de la residencia estudiantil.

Tabla 3: Trabajadores del Negocio.

2.4.4 – Descripción de los casos de uso del negocio.

Nombre del caso de uso del negocio:	Registrar estudiantes en la beca
Actores del negocio:	Estudiantes(inicia)
Propósito:	Asignar el estudiante a un local de la residencia estudiantil.
Resumen: El caso de uso se inicia cuando el estudiante arriba a la residencia estudiantil una vez matriculado en una de las carreras de la universidad con anterioridad. El **director** lo recibe y lo ubica en un cuarto de la residencia estudiantil. El caso de uso finaliza cuando el estudiante queda ubicado en el cuarto.	
Acción del actor	**Respuesta del negocio**
1. El estudiante se presenta en la residencia estudiantil para ubicarse. 3. El estudiante brinda sus datos. 6. El estudiante se ubica en el local.	2. El director le pide los datos al estudiante. 4. El director registra los datos del estudiante. 5. El director ubica al estudiante en un local de la residencia estudiantil.
Prioridad:	Alta: Es el principal proceso del negocio.
Mejoras:	La ubicación del estudiante se realizará por medio

	de una aplicación web donde quedará almacenada toda la información del estudiante becado.

Tabla 4: Descripción del caso de uso del negocio: Registrar estudiantes en la Beca.

Nombre del caso de uso del negocio:	Solicitar cambio de cuarto
Actores del negocio:	Estudiante(inicia)
Propósito:	Cambiar al estudiante de cuarto dentro de la residencia estudiantil.
Resumen: El caso de uso se inicia cuando el estudiante solicita que lo trasladen de cuarto dentro de la residencia. Esta solicitud es recibida y analizada por el director de la residencia estudiantil, que decide si la solicitud es aceptada o no.	
Acción del actor	**Respuesta del negocio**
1. El estudiante solicita cambio de cuarto. 5. El estudiante es notificado	2. El director recibe la solicitud. 3. El director analiza las causas de la solicitud. 4. El director acepta la solicitud y se lo comunica al estudiante.
Prioridad:	media
Mejoras:	La reubicación del estudiante dentro de la residencia se realizará de forma automatizada mediante una aplicación Web.

Tabla 5: Descripción del caso de uso del negocio: Solicitar cambio de cuarto.

Nombre del caso de uso del negocio:	Solicitar baja
Actores del negocio:	Estudiante(inicia)
Propósito:	Dar baja al estudiante de la residencia estudiantil universitaria.
Resumen: El caso de uso se inicia cuando el estudiante solicita la baja de la residencia estudiantil. La solicitud es recibida y analizada por la dirección de la residencia, culminando el caso de uso cuando la baja es otorgada.	
Acción del actor	**Respuesta del negocio**
1. El estudiante solicita la baja de la residencia estudiantil. 3. El estudiante es notificado.	2. El director confirma y actualiza los datos del estudiante.
Prioridad:	media
Mejoras:	El proceso se realizará de forma automatizada, lo que disminuirá el tiempo para realizar la acción.

Tabla 6: Descripción del caso de uso del negocio: Solicitar baja

Nombre del caso de uso del negocio:	Solicitar información.
Actores del negocio:	Directivo(inicia)
Propósito:	Solicitar información a la Residencia Estudiantil.
Resumen: El caso de uso inicia cuando un Directivo solicita información al Director de la residencia, esta información es recopilada y entregada. Culminando así el caso de uso.	
Acción del actor	**Respuesta del negocio**
1. El directivo solicita la información al director de la beca. 5. Recibe la información solicitada.	2. El director recibe la solicitud de la información. 3. El director verifica que el directivo tenga acceso a la información solicitada. 4. Se conforma un informe y se le entrega al directivo que la solicita.
Prioridad:	media
Mejoras:	El proceso se realizará de forma automatizada, utilizando la información que se encuentra almacenada en el por el sistema
Flujos Alternos:	
Línea 3	Si el directivo no tiene acceso a la información el director de la residencia notifica el rechazo de la

	solicitud culminando el caso de uso.

Tabla 7: Descripción del caso de uso del negocio: Solicitar información.

2.4.5- Diagramas de actividades del negocio.

El diagrama de actividad es un grafo que contiene los estados en que puede hallarse la actividad a analizar. Cada estado de la actividad representa la ejecución de una sentencia de un procedimiento, o el funcionamiento de una actividad en un flujo de trabajo. En resumen describe un proceso que explora el orden de las actividades que logran los objetivos del negocio. (James, 2004)

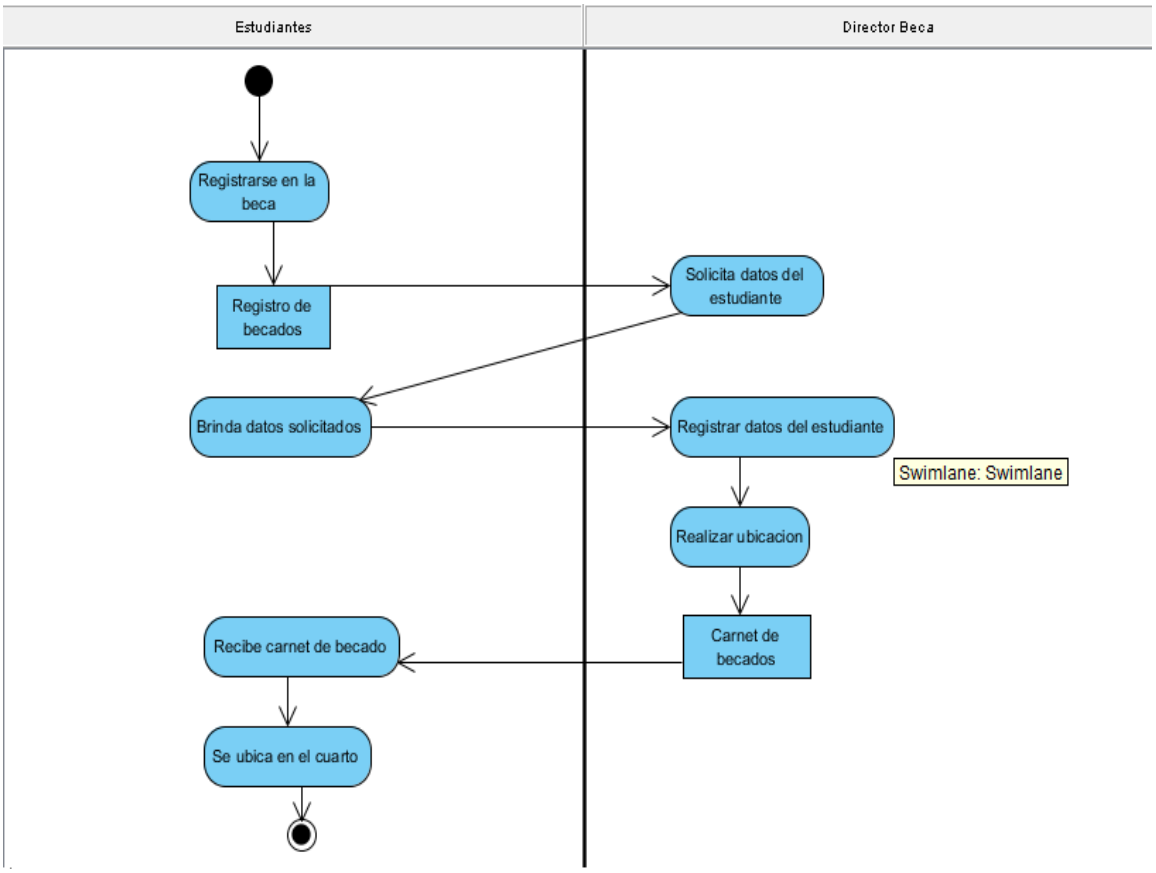


Figura 3: Diagrama de Actividad: Caso de uso Solicitar Beca

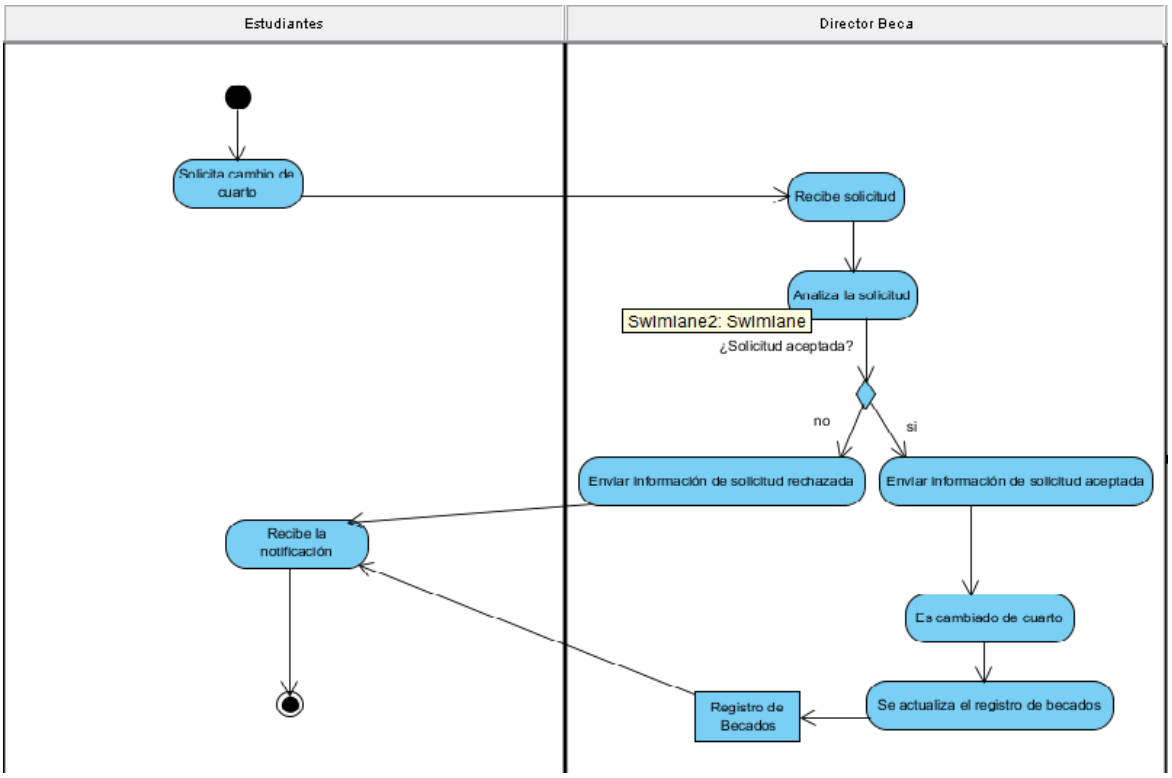


Figura 4: Diagrama de Actividad: Caso de Uso Solicitar Cambio de Cuarto.

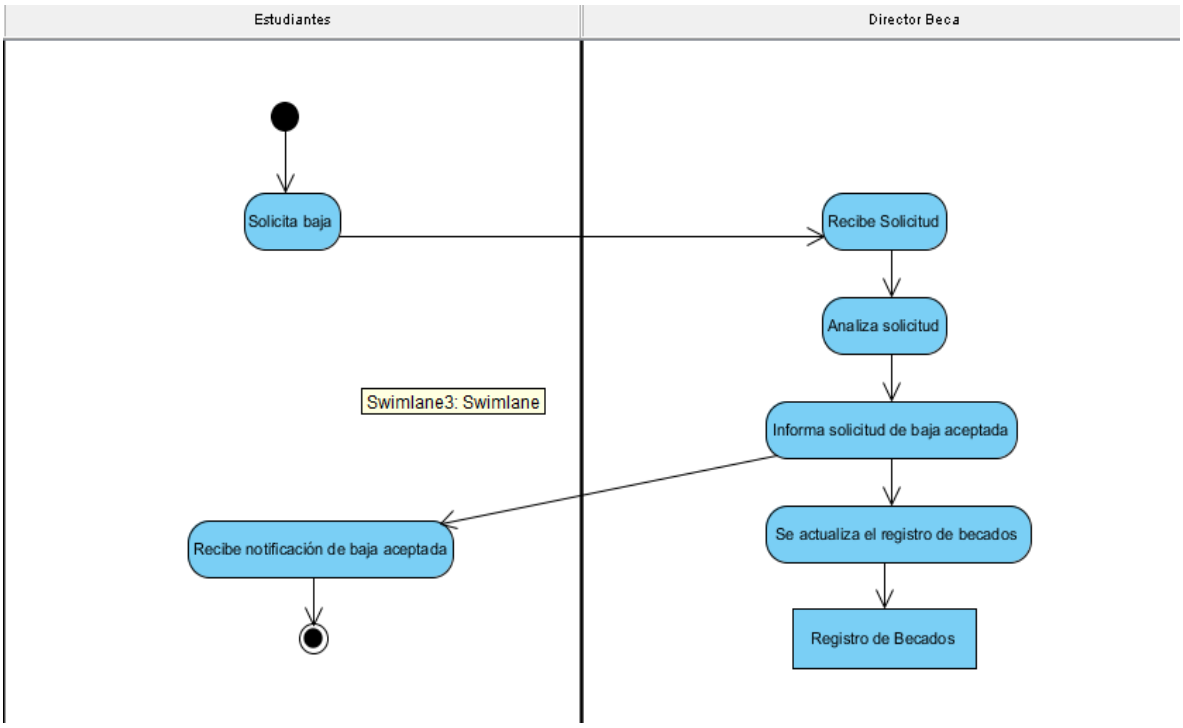


Figura 5: Diagrama de Actividad. Caso de Uso Solicitar Baja.

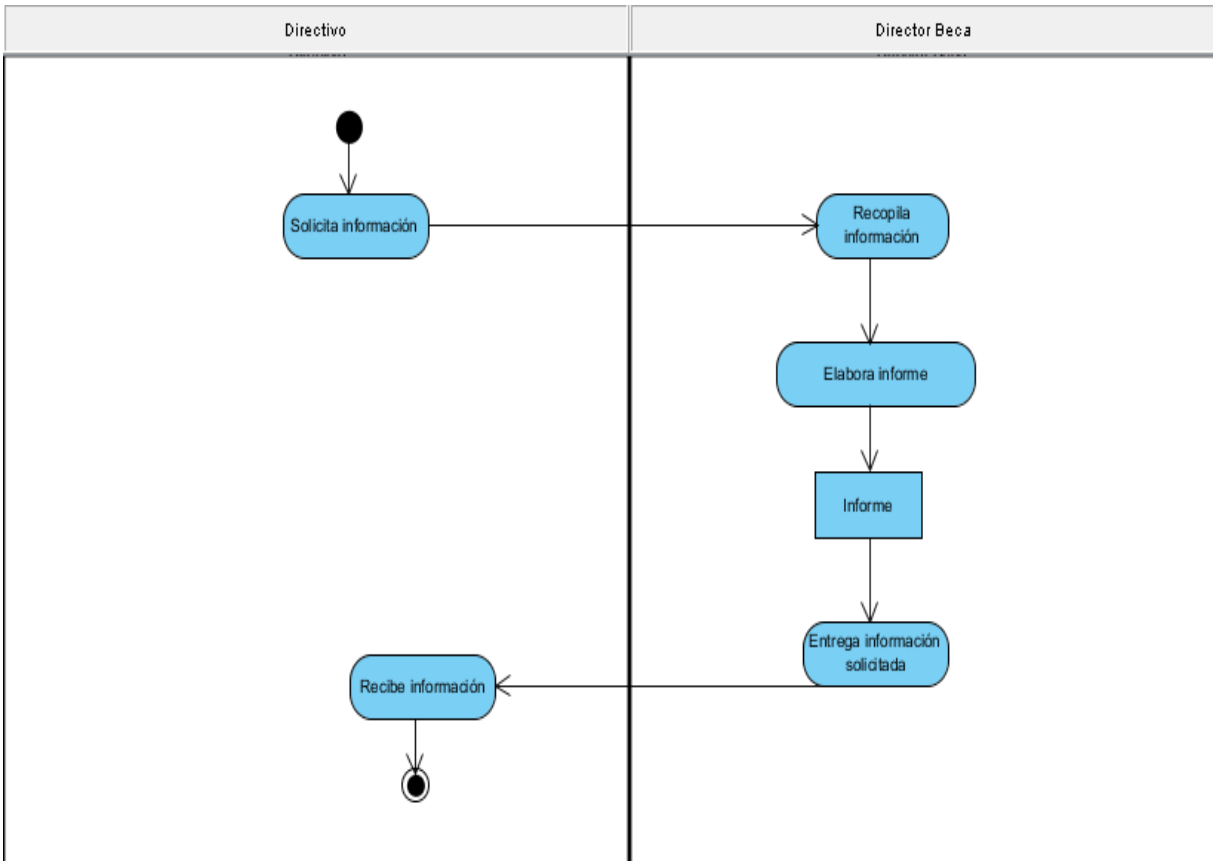


Figura 6: Diagrama de Actividad. Caso de Uso Solicitar Información.

2.5- Modelo de objetos del negocio.

El diagrama de clases, como artefacto que se construye para describir el modelo de objetos del negocio, muestra la participación de los trabajadores y entidades del negocio y la relación entre ellos. Las entidades de negocio representan a los objetos que los trabajadores del negocio toman, inspeccionan, manipulan, producen o utilizan durante la realización de los casos de uso de negocio (James, 2004).

En la siguiente figura se muestra el diagrama del modelo de objetos correspondiente al negocio que se modela.

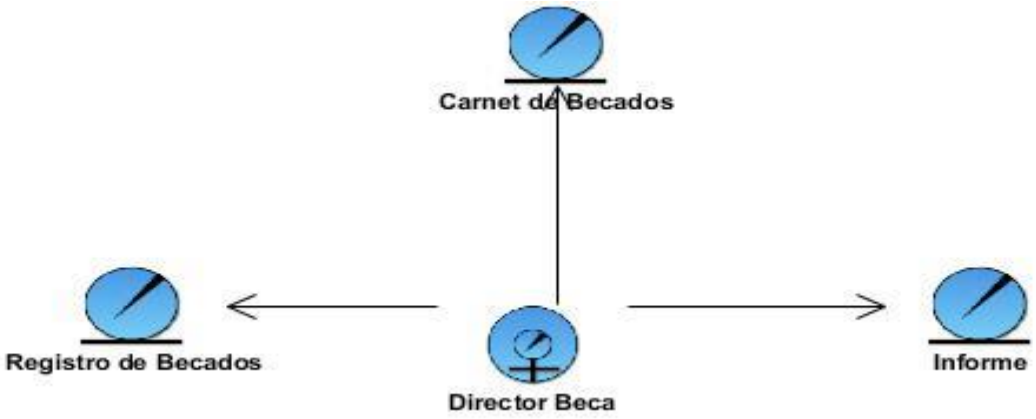


Figura 7: Diagrama de clases del modelo de objetos del negocio.

2.6- Requerimientos.

La IEEE Standard Glossary of Software Engineering Terminology define un requerimiento como Condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo. Todas las ideas que los clientes, usuarios y miembros del equipo de proyecto tengan acerca de lo que debe hacer el sistema, deben ser analizadas como candidatas a requisitos.

2.6.1- Requerimientos funcionales.

Los requerimientos funcionales permiten expresar una especificación más detallada de las responsabilidades del sistema que se propone. Ellos permiten determinar, de una manera clara, lo que debe hacer el mismo.

Todo esto basándose en las necesidades de los usuarios y clientes. Los requerimientos funcionales se mantienen invariables sin importar con que propiedades o cualidades se relacionen.

Teniendo en cuenta el criterio de los autores, el sistema propone los siguientes requerimientos funcionales:

1. Autenticar usuario
2. Insertar usuario
3. Modificar usuario
4. Eliminar usuario
5. Insertar estudiante

6. Modificar estudiante
7. Eliminar estudiante
8. Insertar trabajador
9. Modificar trabajador
10. Eliminar trabajador
11. Insertar medio básico
12. Modificar medio básico
13. Eliminar medio básico
14. Insertar local
15. Modificar local
16. Eliminar local
17. Insertar plaza de la beca
18. Modificar plaza de la beca
19. Eliminar plaza de la beca
20. Insertar sanción disciplinaria
21. Modificar sanción disciplinaria
22. Eliminar sanción disciplinaria
23. Insertar evaluación de los estudiantes becados
24. Modificar evaluación de los estudiantes becados
25. Eliminar evaluación de los estudiantes becados
26. Insertar carrera
27. Modificar carrera
28. Eliminar carrera
29. Insertar municipio
30. Modificar municipio
31. Eliminar municipio
32. Insertar organización política
33. Modificar organización política
34. Eliminar organización política
35. Insertar asignatura
36. Modificar asignatura
37. Eliminar asignatura
38. Insertar arrastre

39. Modificar arrastre
40. Eliminar arrastre
41. Insertar período de evaluación
42. Modificar período de evaluación
43. Eliminar período de evaluación
44. Mostrar listado de estudiantes de la residencia estudiantil
45. Mostrar estudiantes por carrera, año o local
46. Mostrar estudiantes por carrera, municipio o sexo
47. Mostrar evaluación de los estudiantes
48. Mostrar estudiantes sancionados
49. Mostrar datos de un local
50. Mostrar listado de medios básicos
51. Mostrar medios básicos por local
52. Mostrar trabajadores de la residencia estudiantil

2.6.2- Requerimientos no funcionales.

Los requisitos no funcionales especifican propiedades del sistema, como restricciones del entorno o de la implementación, rendimiento, dependencias de la plataforma, factibilidad de mantenimiento, extensibilidad y fiabilidad; con las cuales debe cumplir**.**

Apariencia o interfaz externa.

El sistema debe presentar una interfaz amigable, sencilla, agradable, legible y de fácil uso, de tal forma que en lugar de convertirse un problema para el usuario signifique una ventaja. El sistema además debe indicarle al usuario dónde está y qué puede hacer desde ese lugar.

Usabilidad.

El sistema propuesto es usado por todos los usuarios de la universidad personas que pueden o no, tener habilidades en el trabajo con la computadora, debido a que informa al usuario sobre los aspectos más importantes en la utilización del mismo. El sistema proporcionará un mejor desempeño del personal involucrado en la confección de los planes,

facilitando además el acceso a la información sin un costo elevado.

Rendimiento.

El sistema propuesto debe ser rápido en el procesamiento de la información así como a la hora de dar respuesta a la solicitud de los usuarios, además debe permitir el acceso simultáneo a los datos por diferentes usuarios. Todo esto depende en gran medida del uso que se le dé a los recursos que se disponen en el modelo Cliente/Servidor y de la velocidad de las consultas en la Base de Datos. El sistema deberá recuperarse en un corto período de tiempo ante cualquier falla.

Seguridad.

El sistema propuesto debe garantizar la protección de la información de acceso no autorizado, utilizando la autenticación para garantizar el cumplimiento de esto. En la implementación del sitio Web se harán validaciones de la información tanto en el cliente como en el servidor para contribuir a la seguridad del sistema. El sistema impondrá un estricto control de acceso que permitirá a cada usuario tener disponible solamente las opciones relacionadas con su actividad.

Requisitos de Rendimiento

Aunque no se requiere una velocidad de respuesta comparada con los sistemas de tiempo real, se debe garantizar la rapidez de respuesta del sistema ante las solicitudes de los usuarios. Como la aplicación está concebida para 3 capas, debe garantizarse que los tiempos de respuestas sean generalmente rápidos al igual que la velocidad de procesamiento de la información. Se seguirá una lógica de negocios en la comunicación entre el cliente y el servidor Web de modo tal que cada cual procesará lo que le corresponde, puesto que es posible mencionar que existen funciones que son más óptimas ejecutarlas en el cliente, otras por cuestiones de seguridad, o de acceso a los datos, deben realizarse en el servidor. Se realizará una parte del proceso en el cliente y en dependencia de esto se concluirá en el servidor, lo que facilitará un tiempo de respuesta más rápido, una mayor velocidad de procesamiento, y un mayor aprovechamiento de los recursos. El software estará disponible las 24 horas del día y debe recuperarse ante una falla lo más pronto posible aunque es válido destacar que una caída momentánea no alterará significativamente los procesos de gestión.

Requisitos de Soporte

- **Del lado del Servidor:**

Se requiere una computadora que cuente con un servidor Web Apache, con soporte para

PHP versión 4.0 o superior. Además, se requiere de un servidor de base de datos MySQL. Todo lo anterior para una eficiencia óptima, aunque todo el conjunto puede estar en una sola máquina.

- **Del lado del cliente:**

Por parte del cliente se requiere un navegador que interprete HTML y las funciones básicas de JavaScript, con cualquier sistema operativo.

Requerimientos políticos, culturales y legales.

La herramienta propuesta deberá responder a los intereses de la Constitución de la República de Cuba, asimismo no existirán prioridades en el servicio según el nivel social, cultural o étnico.

Requisitos de Ayuda y Documentación en Línea.

Debe disponerse de una ayuda sobre las principales opciones del sistema.

Requisitos de Software

La aplicación debe poderse ejecutar en entornos *Windows* y/o *Linux* *M*ultiplataforma).

Del lado del servidor se utilizará Apache como servidor Web, del lado del cliente cualquiera de los exploradores existentes en el mercado.

Requerimiento de Hardware

- **Servidor:**

La máquina servidora debe tener como mínimo las siguientes características de hardware: Procesador Pentium III 450 MHz o superior, 256 Mb de memoria RAM (incluye la utilizada por el Sistema Operativo) y 5Gb de capacidad en disco duro.

- **Cliente:**

Las computadoras situadas en los puestos de trabajo de los usuarios requerirán como mínimo un procesador Pentium II, 64 Mb de memoria RAM. Estas máquinas deberán estar conectadas en red con el servidor.

2.7- Modelo de casos de uso del sistema.

El modelo de casos de uso permite que los desarrolladores de software y los clientes lleguen a un acuerdo sobre los requisitos, es decir, sobre las condiciones y posibilidades que

debe cumplir el sistema. Describe lo que hace el sistema para cada tipo de usuario y proporciona la entrada fundamental para el análisis, el diseño y las pruebas.

2.7.1 – Actores del sistema.

Los actores representan terceros fuera del sistema que colaboran con este. Suelen corresponderse con trabajadores o actores en un negocio.

Actores	**Justificación**
Director Beca	Este actor tiene control total sobre el sistema, es el encargado de gestionar la información de los estudiantes becados, de insertar y actualizar las evaluaciones y sanciones, de ubicarlos en los locales y además gestiona la información relacionada con la ubicación y el estado del mobiliario en la residencia estudiantil.
Administrador del sistema	
Usuario del Sistema	Este actor es un usuario genérico que va a realizar las acciones comunes entre los otros actores.

Tabla 8: Actores del sistema.

2.7.2 – Paquetes y sus relaciones

Los paquetes son un mecanismo de organización de elementos que subdividen el modelo en otros más pequeños que colaboran entre sí. Este particionamiento debe hacerse sobre la base de los requerimientos funcionales y el dominio del problema; y debe ser reconocible por las personas con conocimiento del dominio.

A partir de lo anteriormente expuesto se propone el siguiente diagrama de paquetes:

Paquete Gestión de Información.

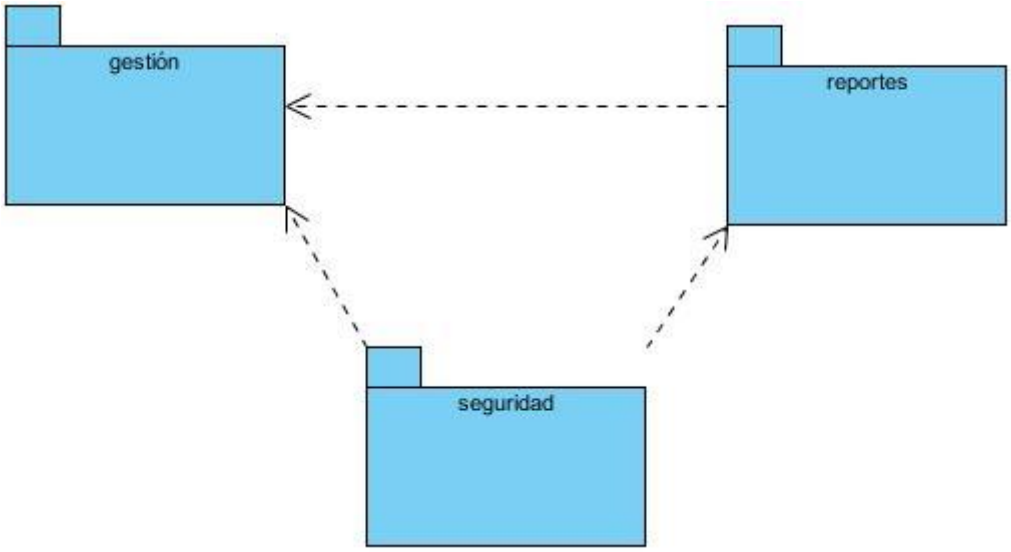


Figura 8: Diagrama de Casos de Usos por Paquetes.

Se agrupan todas las funcionalidades relacionadas con la actualización (inserción, modificación o eliminación) de los datos.

Paquete Reportes.

Se agrupan todas las funcionalidades relacionadas con, la visualización de los datos, filtrados según diferentes criterios de búsqueda y la obtención de reportes.

Paquete de Seguridad.

Se agrupan todas las funcionalidades relacionadas con la autenticación de los distintos tipos de usuarios del sistema.

2.7.3 – Diagramas de casos de uso del sistema.

Los casos de usos constituyen un instrumento para dividir la complejidad del sistema en varios pedazos más manejables que facilitarán su comprensión y análisis, estos deben estar basados en la forma en que el sistema es visto por un usuario.

A continuación se muestra el diagrama de casos de uso para cada uno de los paquetes del sistema.

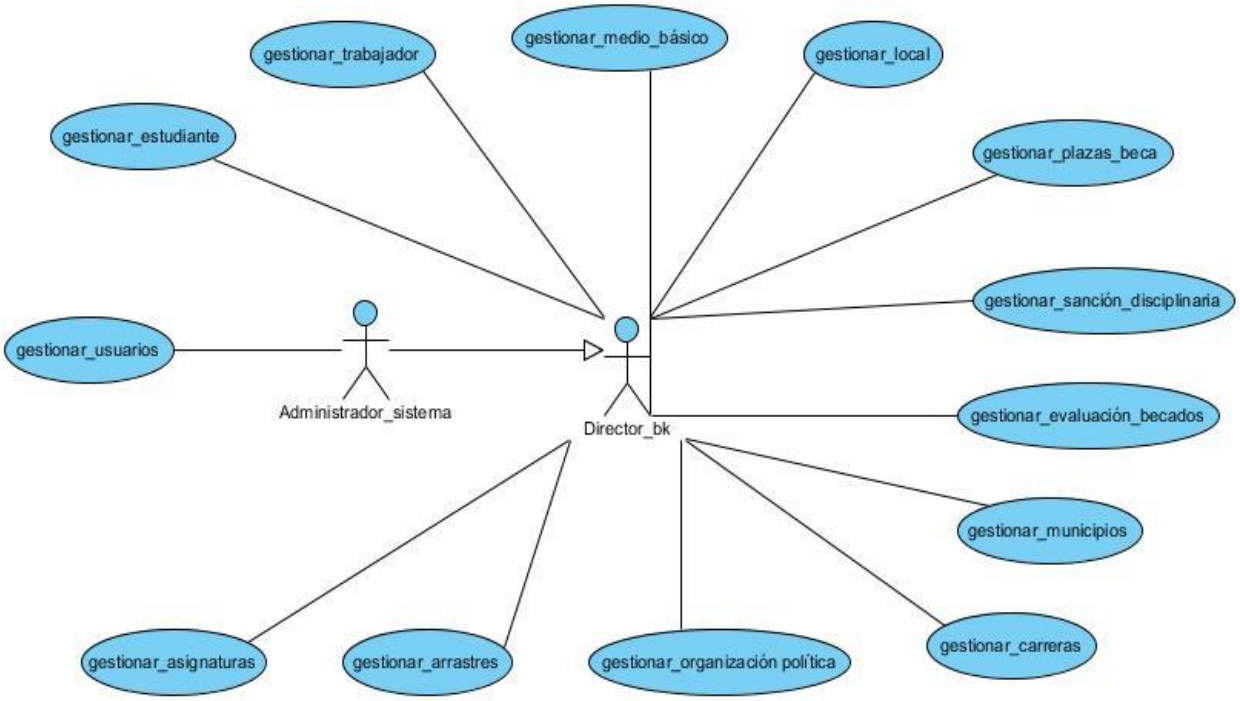


Figura 9: Diagrama de casos de uso del sistema Paquete Gestión de Información.

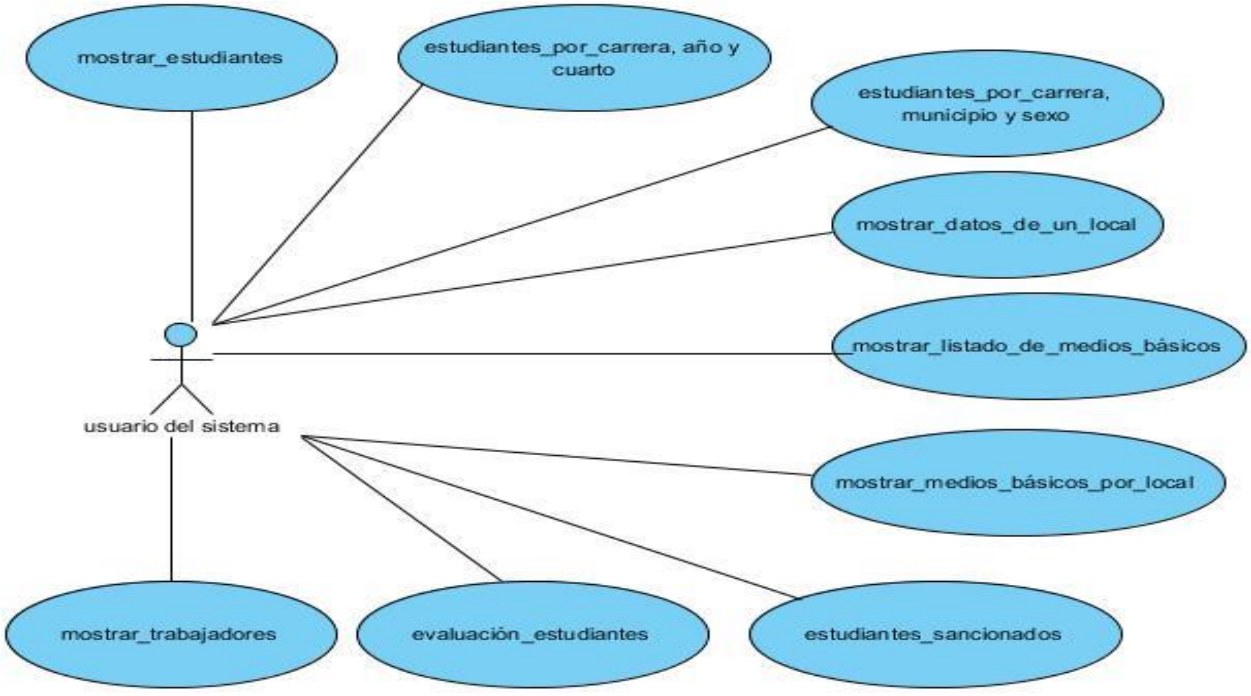


Figura 10: Diagrama de casos de uso del sistema Paquete Reportes.

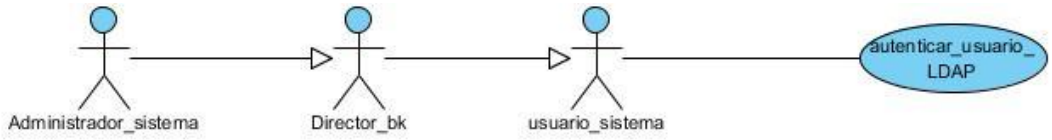


Figura 11: Diagrama de casos de uso del sistema Paquete de Seguridad.

2.7.4 – Descripción de los casos de uso del sistema.

Cada forma en que los actores usan el sistema se representa con un Caso de Uso. Los Casos de Uso son "fragmentos" de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. Un Caso de Uso especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores, incluyendo alternativas dentro de la secuencia.

CU	**Autenticar Usuario LDAP**
Actores	
Propósito	Permitir al usuario iniciar sesión en el Sistema.
Resumen El caso de uso comienza cuando el usuario avanzado decide iniciar sesión en el sistema y entra los datos. El sistema comprueba la información, la verifica contra el Directorio Activo y le da los privilegios correspondientes al usuario. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 9: Descripción del caso de uso: Autenticar Usuario LDAP

CU	**Gestionar Usuarios**
Actores	Administrador del sistema
Propósito	Tener control de todos los usuarios y sus permisos en la aplicación.
Resumen	

El caso de uso comienza cuando el Administrador del sistema decide registrar los datos de un nuevo usuario u otorgarle permisos a alguno existente, los que puede insertar, modificar o eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 10: Descripción del caso de uso: Gestionar Usuarios.

CU	**Gestionar Estudiante**
Actores	Director Beca
Propósito	Tener registrados todos los datos de los estudiantes becados.
Resumen El caso de uso comienza cuando el Director decide registrar los datos de los estudiantes, los que puede insertar, modificar o eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 11: Descripción del caso de uso: Gestionar Estudiante

CU	**Gestionar Trabajador**
Actores	Director Beca
Propósito	Tener registrados todos los datos de los trabajadores de la Residencia Estudiantil.
Resumen	

El caso de uso comienza cuando el Director decide actualizar los datos de los trabajadores de la Residencia Estudiantil, los que puede insertar, modificar o eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 12: Descripción del caso de uso: Gestionar Trabajador.

CU	**Gestionar Medios Básicos**
Actores	Director Beca
Propósito	Mantener actualizado la información referente al mobiliario de la residencia estudiantil.
Resumen El caso de uso se inicia cuando el Director se dispone a actualizar la información relacionada con el mobiliario de la residencia estudiantil, el cual se puede insertar, modificar o eliminar de un local de la residencia estudiantil. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 13: Descripción del caso de uso: Gestionar Medios Básicos.

CU	**Gestionar Locales**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a la estructura de la Residencia Estudiantil

Resumen El caso de uso comienza cuando el Director decide registrar la forma en que están estructurados los locales de la Residencia Estudiantil, los cuales se pueden insertar, modificar y eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 14: Descripción del caso de uso: Gestionar Locales.

CU	**Gestionar Carreras**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a las carreras que estudian los estudiantes becados.
Resumen El caso de uso comienza cuando el Director decide registrar la información de las carreras que estudian los estudiantes becados, esta información se puede insertar, modificar u eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 15: Descripción del caso de uso: Gestionar Carreras.

CU	**Gestionar Asignaturas**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a las asignaturas que cursan los estudiantes becados.

Resumen El caso de uso comienza cuando el Director decide registrar la información de las asignaturas que cursan los estudiantes becados, esta información se puede insertar, modificar u eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 16: Descripción del caso de uso: Gestionar Asignaturas

CU	**Gestionar Asignaturas de las Carreras**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a las asignaturas que cursan los estudiantes becados en su carrera.
Resumen El caso de uso comienza cuando el Director decide registrar la información de las asignaturas que cursan los estudiantes becados en su carrera, esta información se puede insertar, modificar u eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 17: Descripción del caso de uso: Gestionar Asignaturas de las Carreras.

CU	**Gestionar Municipios**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a los municipios a los que pertenecen los estudiantes becados y los trabajadores que allí laboran.

Resumen	
El caso de uso comienza cuando el Director decide registrar la información referente a los municipios a los que pertenecen los estudiantes becados y los trabajadores que allí laboran, esta información se puede insertar, modificar u eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 18: Descripción del caso de uso: Gestionar Municipios.

CU	**Gestionar Organización Política**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a las organizaciones políticas a las que pertenecen los estudiantes becados y los trabajadores que allí laboran.
Resumen	
El caso de uso comienza cuando el Director decide registrar la información referente a las organizaciones políticas a las que pertenecen los estudiantes becados y los trabajadores que allí laboran, esta información se puede insertar, modificar u eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 19: Descripción del caso de uso: Gestionar Organización Política.

CU	**Gestionar Sanción Disciplinaria**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a las sanciones disciplinarias que se les aplican a los estudiantes becados.

Resumen El caso de uso comienza cuando el Director decide registrar la información referente a las sanciones disciplinarias que se les aplican a los estudiantes becados, esta información se puede insertar, modificar u eliminar. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 20: Descripción del caso de uso: Gestionar Sanción Disciplinaria.

CU	**Gestionar Arrastres**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a las asignaturas pendientes de los estudiantes becados (arrastres).
Resumen El caso de uso comienza cuando el Director decide registrar la información referente a las asignaturas pendientes de los estudiantes becados.Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 21: Descripción del caso de uso: Gestionar Arrastres.

CU	**Gestionar Evaluación de los Becados**
Actores	Director Beca

Propósito	Mantener actualizada la información referente a las evaluaciones de los estudiantes becados.
Resumen El caso de uso comienza cuando el Director decide registrar la información referente a las evaluaciones de los estudiantes becados.Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 22: Descripción del caso de uso: Gestionar Evaluación de los Becados.

CU	**Gestionar Plazas de la Beca**
Actores	Director Beca
Propósito	Mantener actualizada la información referente a las plazas de trabajo que tiene la beca.
Resumen El caso de uso comienza cuando el Director decide registrar la información referente a las plazas de trabajo que tiene la beca, esta información se puede insertar, modificar u eliminar.Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 23: Descripción del caso de uso: Gestionar Plazas de la Beca.

CU	**Mostrar estudiantes de la Residencia Estudiantil**

Actores	Usuario del sistema
Propósito	Mostrar listado de los estudiantes becados.
Resumen El caso de uso comienza cuando un usuario del sistema decide obtener un listado de los estudiantes becados. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 24: Descripción del caso de uso: Mostrar estudiantes de la Residencia Estudiantil.

CU	**Mostrar trabajadores de la Residencia Estudiantil**
Actores	Usuario del sistema
Propósito	Mostrar listado de los trabajadores de la Residencia Estudiantil.
Resumen El caso de uso comienza cuando un usuario del sistema decide obtener un listado de los trabajadores de la Residencia Estudiantil. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 25: Descripción del caso de uso: Mostrar trabajadores de la Residencia Estudiantil.

CU	**Mostrar evaluación de los estudiantes.**
Actores	Usuario del sistema
Propósito	Mostrar evaluación de los estudiantes.

Resumen El caso de uso comienza cuando un usuario del sistema decide obtener la evaluación de los estudiantes becados. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 26: Descripción del caso de uso: Mostrar evaluación de los estudiantes

CU	**Mostrar estudiantes sancionados.**
Actores	Usuario del sistema
Propósito	Mostrar estudiantes sancionados.
Resumen El caso de uso comienza cuando un usuario del sistema decide obtener los estudiantes sancionados. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 27: Descripción del caso de uso: Mostrar estudiantes sancionados.

CU	**Mostrar estudiantes por carrera año y cuarto.**
Actores	Usuario del sistema
Propósito	Mostrar estudiantes por carrera año y cuarto.
Resumen	

El caso de uso comienza cuando un usuario del sistema decide obtener un listado de estudiantes por carrera año y cuarto. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 28: Descripción del caso de uso: Mostrar estudiantes por carrera año y cuarto.

CU	**Mostrar estudiantes por carrera municipio y sexo.**
Actores	Usuario del sistema
Propósito	Mostrar estudiantes por carrera municipio y sexo.
Resumen El caso de uso comienza cuando un usuario del sistema decide obtener un listado de estudiantes por carrera municipio y sexo. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 29: Descripción del caso de uso: Mostrar estudiantes por carrera municipio y sexo.

CU	**Mostrar listado de medios básicos de la Residencia Estudiantil.**
Actores	Usuario del sistema
Propósito	Mostrar listado de medios básicos que se encuentran en la Residencia Estudiantil.
Resumen	

El caso de uso comienza cuando un usuario del sistema decide obtener un listado de medios básicos que se encuentran en la Residencia Estudiantil. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 30: Descripción del caso de uso: Mostrar listado de medios básicos de la Residencia Estudiantil

CU	**Mostrar listado de medios básicos por local.**
Actores	Usuario del sistema
Propósito	Mostrar listado de medios básicos que se encuentran en un local determinado de la Residencia Estudiantil.
Resumen El caso de uso comienza cuando un usuario del sistema decide obtener un listado de medios básicos que se encuentran en un local determinado de la Residencia Estudiantil. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 31: Descripción del caso de uso: Mostrar listado de medios básicos por local.

CU	**Mostrar datos de un local de la Residencia Estudiantil.**
Actores	Usuario del sistema
Propósito	Mostrar datos de un local determinado de la Residencia Estudiantil.
Resumen	

El caso de uso comienza cuando un usuario del sistema decide obtener los datos de un local determinado de la Residencia Estudiantil. Culmina el caso de uso.	
Referencias	
Prototipo	

Tabla 32: Descripción del caso de uso: Mostrar datos de un local de la Residencia Estudiantil.

2.8- Conclusiones

En este capítulo fueron descritos los procesos que tiene lugar en la Residencia Estudiantil de la Universidad de Sancti-Spíritus, identificando a su vez los actores, trabajadores y objetos del negocio, así como su relación en esos procesos (casos de uso). Esta descripción fue realizada mediante el modelo del negocio, para lo cual se elaboraron los modelos de casos de uso y de actividad, lográndose una mejor comprensión de los procesos de la Residencia Estudiantil de la Uniss, dando paso al modelado del sistema.

Capítulo 3: Construcción de la solución propuesta.

Introducción.

El diseño es el primer paso en la fase de desarrollo de cualquier producto o sistema de ingeniería, es donde se fomenta la calidad en el desarrollo del software. Es la única manera de traducir con precisión los requerimientos funcionales del cliente en un sistema o producto de software.

En el presente capítulo se plasman los resultados de la etapa del diseño. Se presentan diagramas como por ejemplo: diagrama de clases persistentes y modelo de datos, diagrama de componente y de despliegue. Además se describen los principios de diseño aplicados.

3.1- Diagrama de clases.

El Diagrama de Clases es el diagrama principal de diseño y análisis para un sistema. Presenta las clases, junto con sus atributos, operaciones, interfaces y relaciones. Se convierte en el diagrama central del análisis del diseño orientado a objetos, y el que muestra la estructura estática del sistema. Durante el análisis del sistema, el diagrama se desarrolla buscando una solución ideal. Durante el diseño, se usa el mismo diagrama, y se modifica para satisfacer los detalles de las implementaciones. Puede ser dividido en capas: aplicación, y datos, las cuales muestran las clases que intervienen con la interfaz de usuario, la lógica del software de la aplicación, y el almacenamiento de datos respectivamente. 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 entre las páginas, donde cada página lógica puede ser representada como una clase (Ferrá Grau, 2010).

Los diagramas de clases web, fueron realizados a partir de los diferentes casos de uso del sistema y empleando las extensiones de UML para web.

3.2- Diseño de la base de datos

Las Bases de datos necesitan de una definición de su estructura que le permitan almacenar datos, reconocer el contenido y recuperar la información.

3.2.1- Diagrama de clases persistentes

En el diagrama de clases persistentes aparecen las clases que persisten, las cuales poseen la capacidad de mantener su valor en el espacio y en el tiempo. (Rumbaugh, Booch, & Jacobson,

2006). Está compuesto por clases, asociaciones y atributos; interfaces, con sus operaciones y constantes; métodos; información sobre los tipos de atributos, entre otros (Ferrá Grau, 2010).

A partir de este planteamiento se definieron las siguientes clases persistentes:

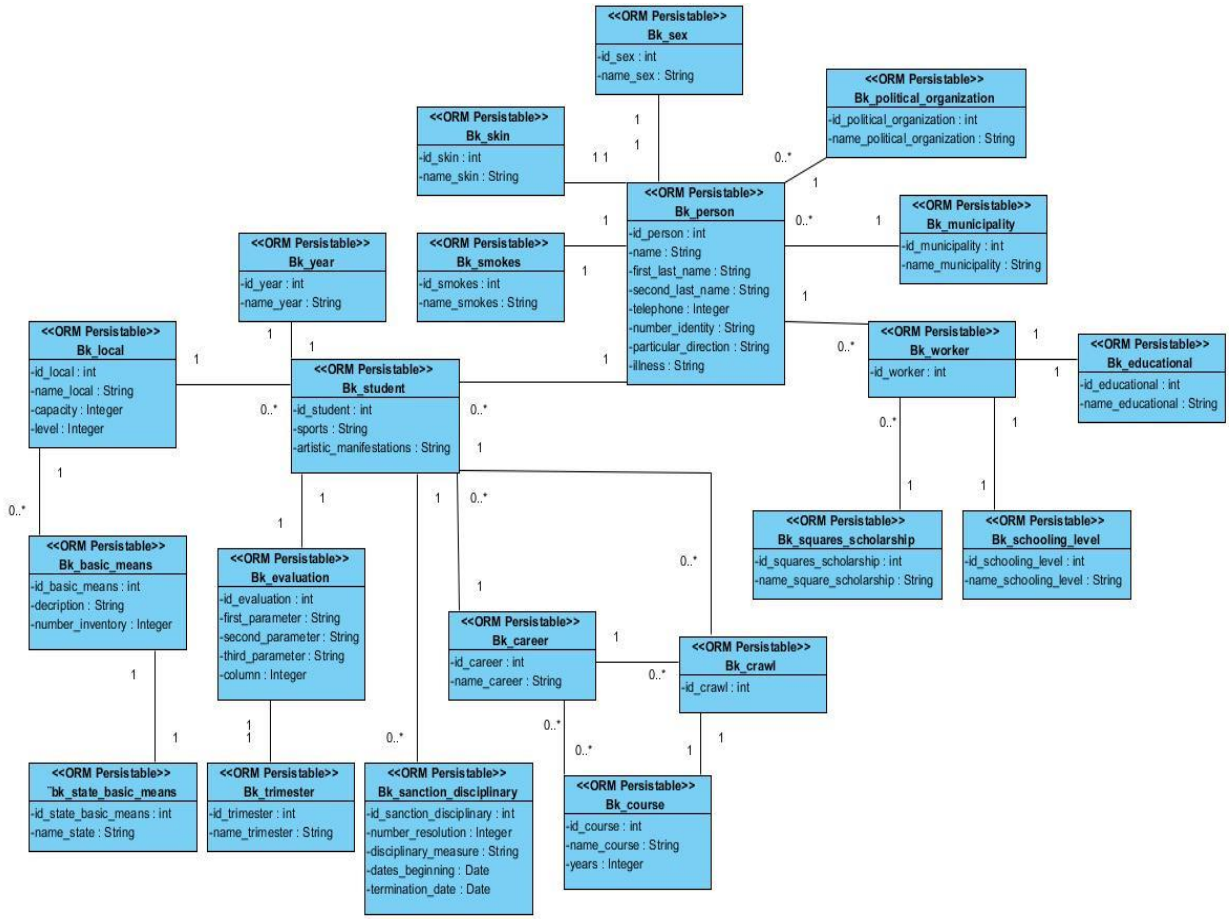


Figura 12: Diagrama de clases persistentes.

3.2.2- Modelo de datos

El modelo físico de datos, representa la estructura o descripción física de las tablas de la base de datos, obtenido a partir del modelo lógico de datos (Ferrá Grau, 2010).

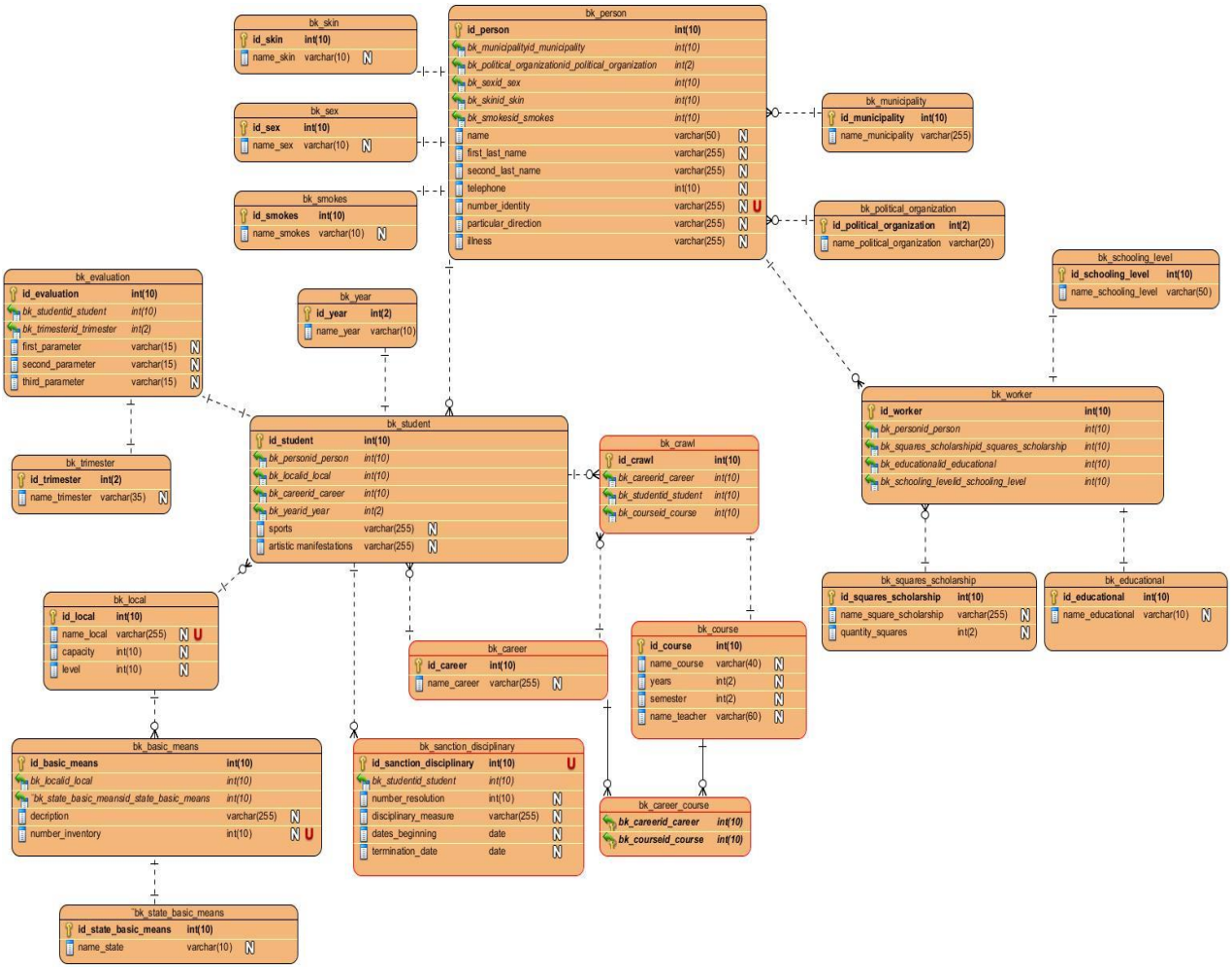


Figura 13: Modelo de datos.

3.3- Modelo de implementación

El modelo de implementación describe cómo los elementos del modelo del diseño, se implementan en términos de componentes, describe cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en los lenguajes de programación utilizados y cómo dependen unos componentes de otros. (Ferrá Grau, 2010)

3.3.1- Diagrama de despliegue

El diagrama de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuyen las funcionalidades entre los nodos de cómputo (Jacobson, Booch, & Rumbaugh, 2000). El modelo representa dos nodos: en el lado del servidor se

encuentra MySQL como servidor de base de datos y se comunica con el cliente a través del protocolo TCP/IP. Se visualiza la aplicación en el lado del cliente. (Ferrá Grau, 2010)

Figura 14: Diagrama de despliegue.

3.3.2- Diagrama de componentes

Un diagrama de componentes muestra un conjunto de elementos del modelo, se utiliza para modelar la vista estática de un sistema, muestra la organización y dependencias lógicas entre los componentes del software ya sean bibliotecas, ejecutables o componentes binarios. (Ferrá Grau, 2010)

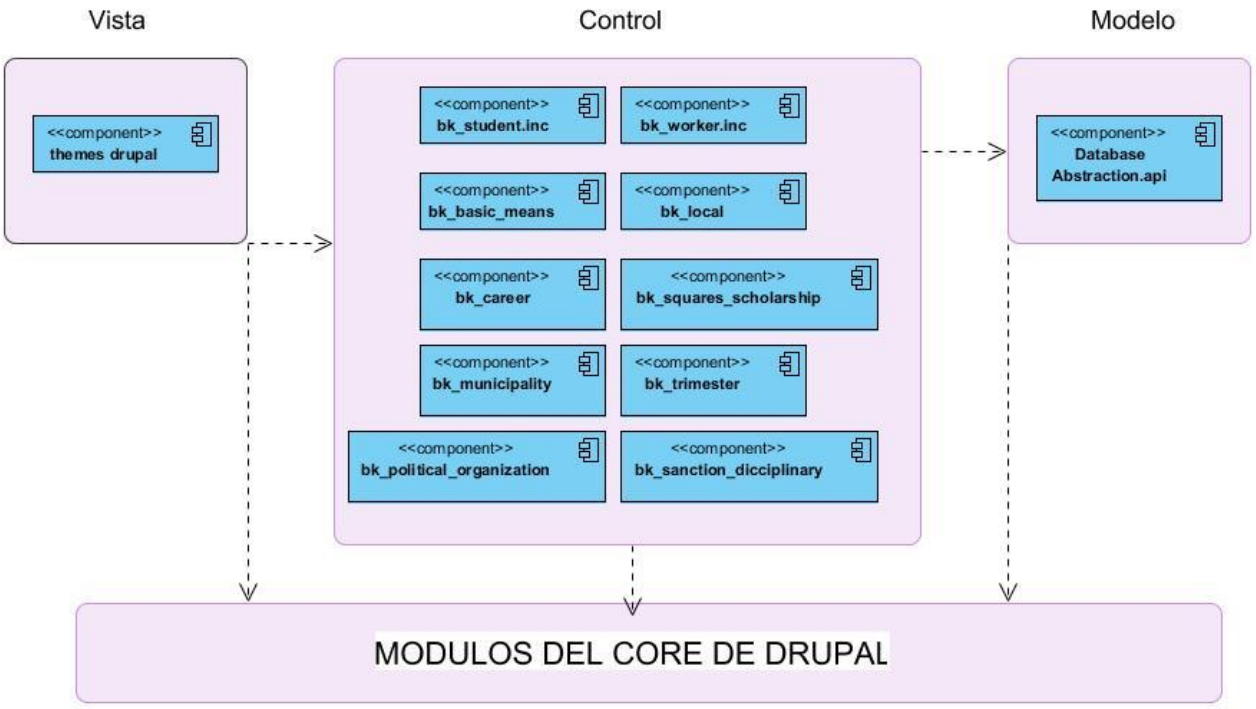


Figura 15: Diagrama de componentes.

3.4- Conclusiones

En este capítulo se describió el diagrama de clases de diseño, el diagrama de clases persistentes, el modelo de datos y la descripción de cada una de las tablas. Se enunciaron los principios de diseño determinando los estándares usados en la interfaz de la aplicación. Se describió además la implementación incluyendo el diagrama de despliegue y el diagrama de componentes.

Conclusiones

- Para el desarrollo de la aplicación web se utilizó como guía la metodología RUP y como lenguaje de modelación el Lenguaje Unificado de Modelado, lo cual posibilitó una adecuada documentación del análisis, el diseño y la implementación de la solución propuesta. Como herramienta de modelado se utilizó el Visual Paradigm.

- Se diseñó un sistema capaz de perfeccionar la gestión de usuarios la Universidad de Sancti Spíritus "José Martí Pérez". El mismo se sustenta en los preceptos del software libre, la arquitectura en tres capas y la programación orientada a objeto, todo esto le brinda flexibilidad y posibilidades para futuras modificaciones y mejoras.

- Se implementó un sitio web utilizando como lenguaje de programación PHP y para la interfaz de usuario HTML, además del uso de JavaScript para las validaciones de entradas del usuario y CSS para el estilo y apariencia. Como herramientas de desarrollo IDE Sublime Text para la programación Web y MySQL como SGBD.

Bibliografía

5.0, M. M. (2011). *Panorámica del sistema de gestión de bases de datos MySQL.*

Alvarez M., A., Monteiro L., J., & Méndez, N. (2009). *Desarrollo Web*. Recuperado el 6 de abril de 2014, de Desarrollo Web: http://www.desarrolloweb.com/manuales/27/

Bartle, P. (2009). *Informacion para la gestión y gestión de la información*. Recuperado el 16 de 4 de 2014, de http://www.scn.org/mpfc/modules/mon-miss

Barzanallana, R. (2012). *Desarrollo de Aplicaciones Web*. Recuperado el 13 de 4 de 2014, de http://www.apachefriends.org/es/xampp.html

Chappell, D. (2006). *De N-capas a .NET. Desarrollo de aplicaciones*. Recuperado el 11 de 4 de 2014, de http://www.microsoft.com/spanish/msdn/articulos/archivo/081102/voices/dncapas.asp

Curto, J. (2006). *Information Management*. Recuperado el 12 de 4 de 2014, de Reflexiones sobre las tecnologías de la información: http://www.informationmanagement.wordpress.com

Dante, G. (1998). *Principios, conceptos y aplicaciones.* Santiago de Chile: Universidad de Chile.

Estadística, O. (2009). *TICs. Usos y Acceso en Cuba.*

Ferrá Grau, X. (2010). *Desarrollo orientado a objetos con UML*. Recuperado el 12 de 4 de 2014, de http://www.clikear.com/manuales/uml/introduccion.asp

Leal, E. (2008). Las tecnologías de la información y las comunicaciones. *Revista de Universidad y Sociedad del Conocimiento*, 3-6.

Molina, M., & Castro, A. (2009). *Cliente- servidor*. Recuperado el 12 de 4 de 2014, de http://deymar-clienteservidor.blogspot.com/2009/04/si-bien-la-clasica-arquitecturas.html

Peláez, J. (2009). *Arquitectura basada en capas*. Recuperado el 14 de 4 de 2014, de http://geeks.ms/blogs/jpelaez/archive/2009/05/29/arquitectura-basada-en-capas.aspx

Riveros, F. (2008). *GEstor de Base de Datos: MySQL, PostgreSQL, SQLite*. Recuperado el 13 de 4 de 2014, de http://www.eaprende.com/base_de_datos_SQL_Server_con_PHP_y_ADODB.html

Rodriguez, L. (2012). *Del gesto a las tecnologías de la información y las comunicaciones.*

Rothberg, D. (2006). *Programming Languages You Should Learn Right Now*. Recuperado el 13 de 4 de 2014, de http://www.eweek.com/c/a/IT-Management/10-ProgrammingLanguages-You-Should-Learn-Right-Now/

Rumbaugh, J., Booch, G., & Jacobson, I. (2006). *El proceso unificado de desarrollo de software.* La Habana: Félix Varela.

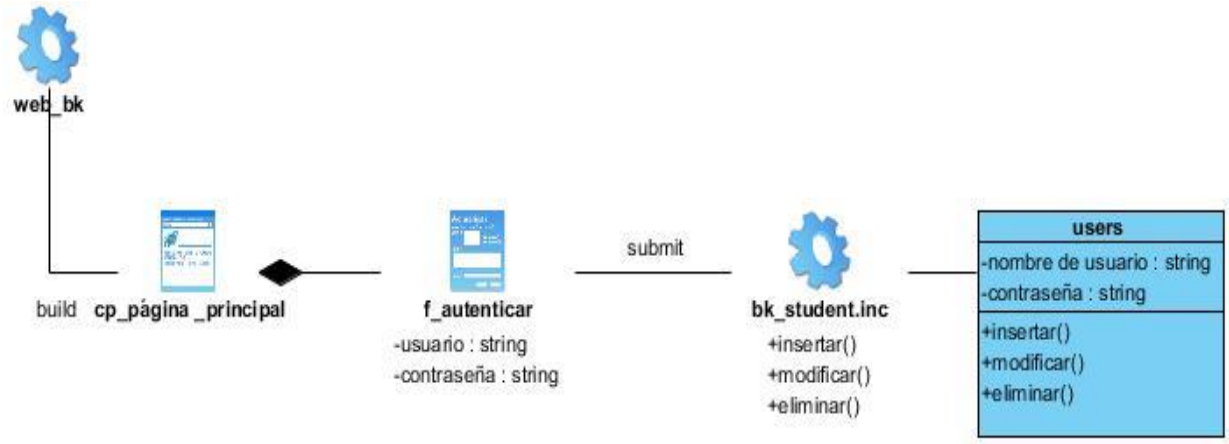
SoftwareLibre.html. (2011). *Software Libre.html*. Recuperado el 15 de 4 de 2014, de http://www.hispalinux.es

Sublime Text, un sofisticado editor de código multiplataforma. (2010). Recuperado el 17 de 4 de 2014, de http://Sublime Text, un sofisticado editor de código multiplataforma

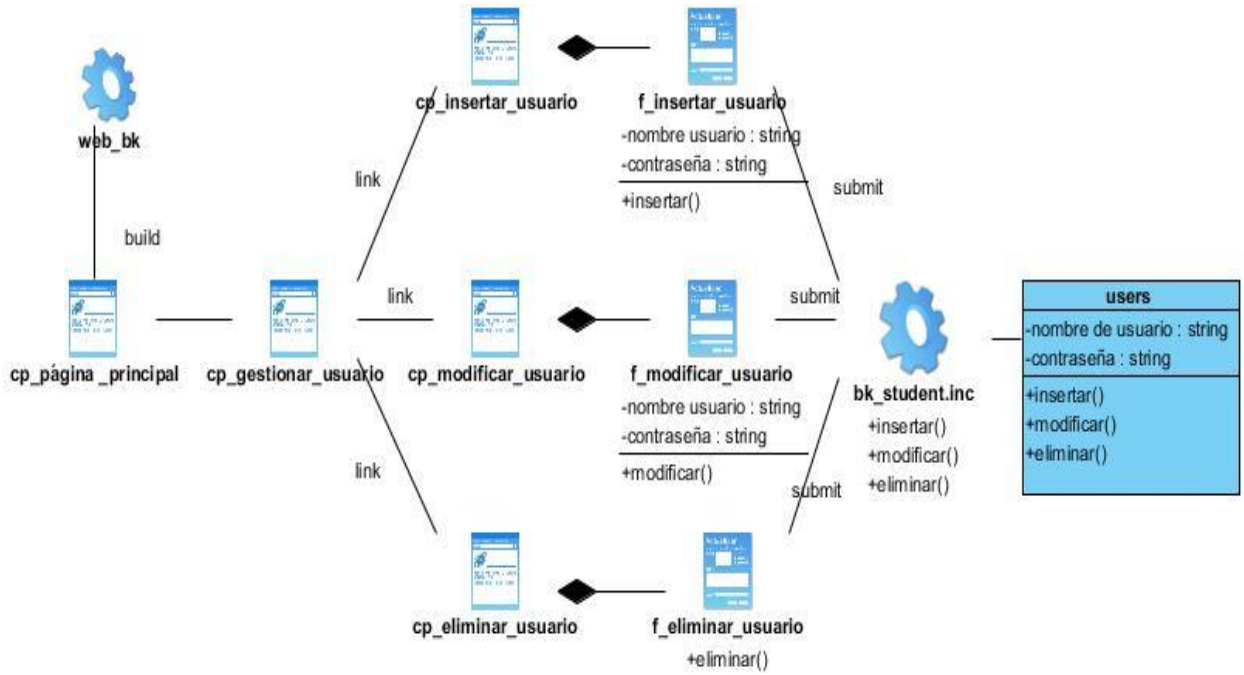
Triana Cabrera, E. (2013). *Aplicación web para la gestión de usuarios de dominio de los trabajadores de la Universidad de Sancti Spíritus "José Marti Pérez".* Sancti Spíritus.

Zaguero, P. (2008). *Administración de proyectos de Software*. Recuperado el 14 de 4 de 2014, de http://www.zohowriter.com/public/27201/38205

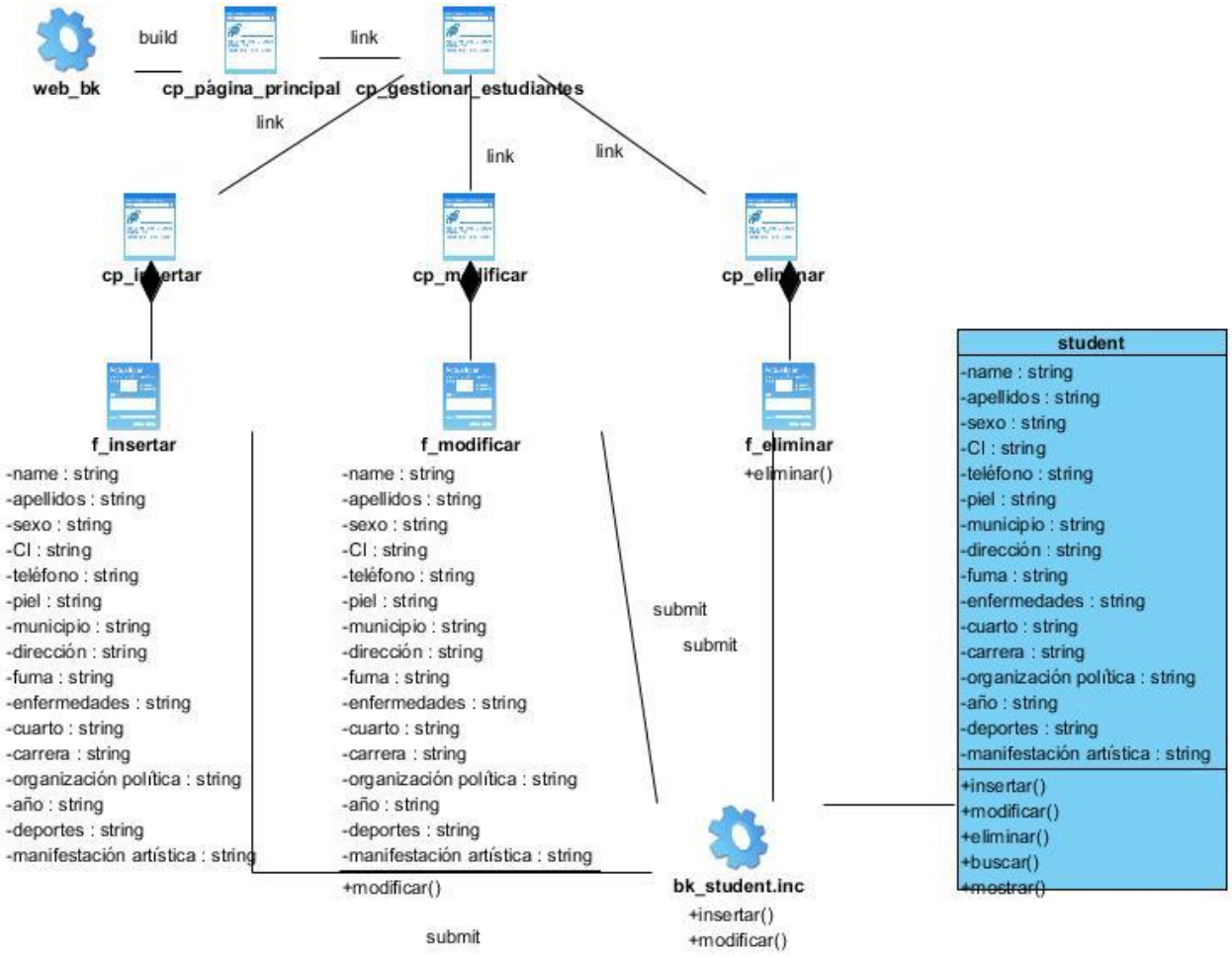
Anexos



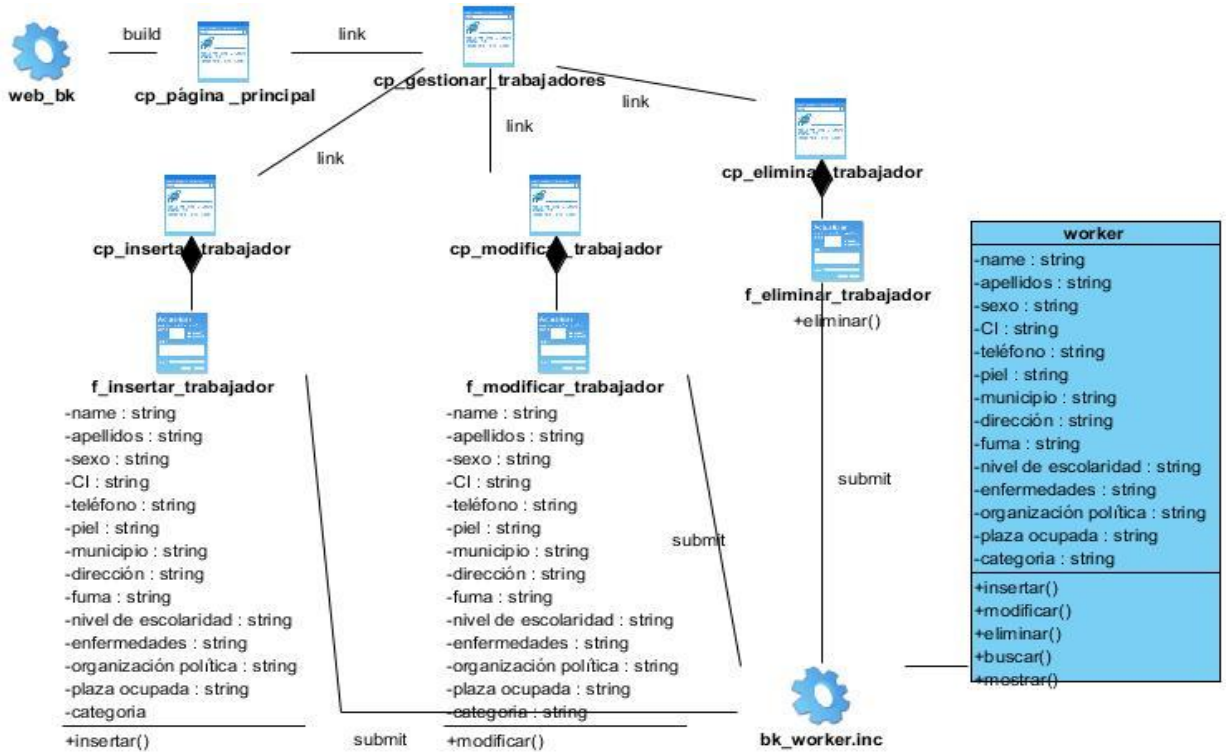
Anexo 1: Diagrama de Clase del diseño para CU Autenticarse.



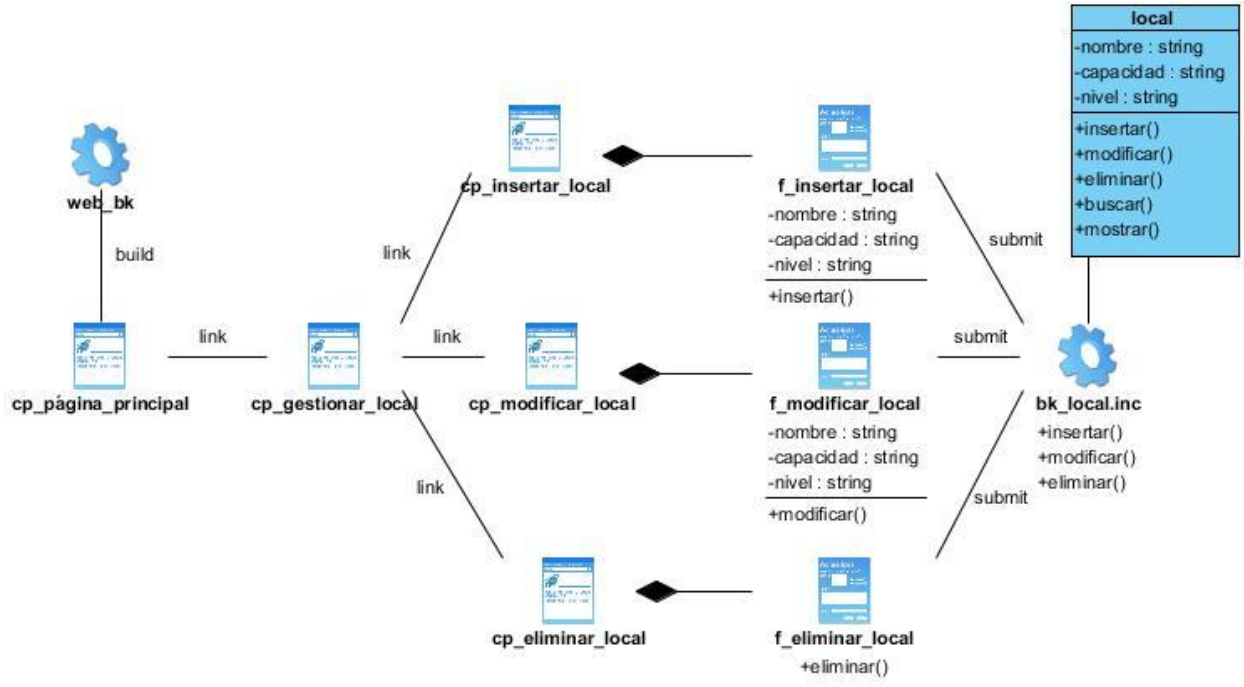
Anexo 2: Diagrama de Clase del diseño para CU Gestionar Usuario.



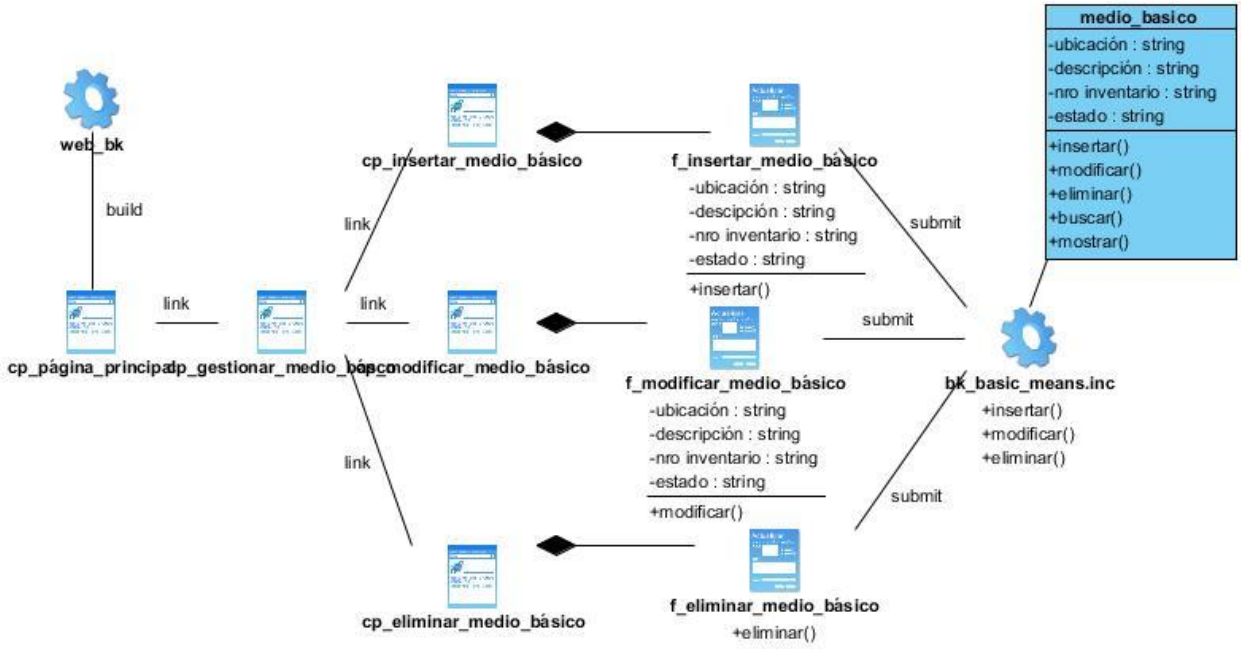
Anexo 3: Diagrama de Clase del diseño para CU Gestionar Estudiante.



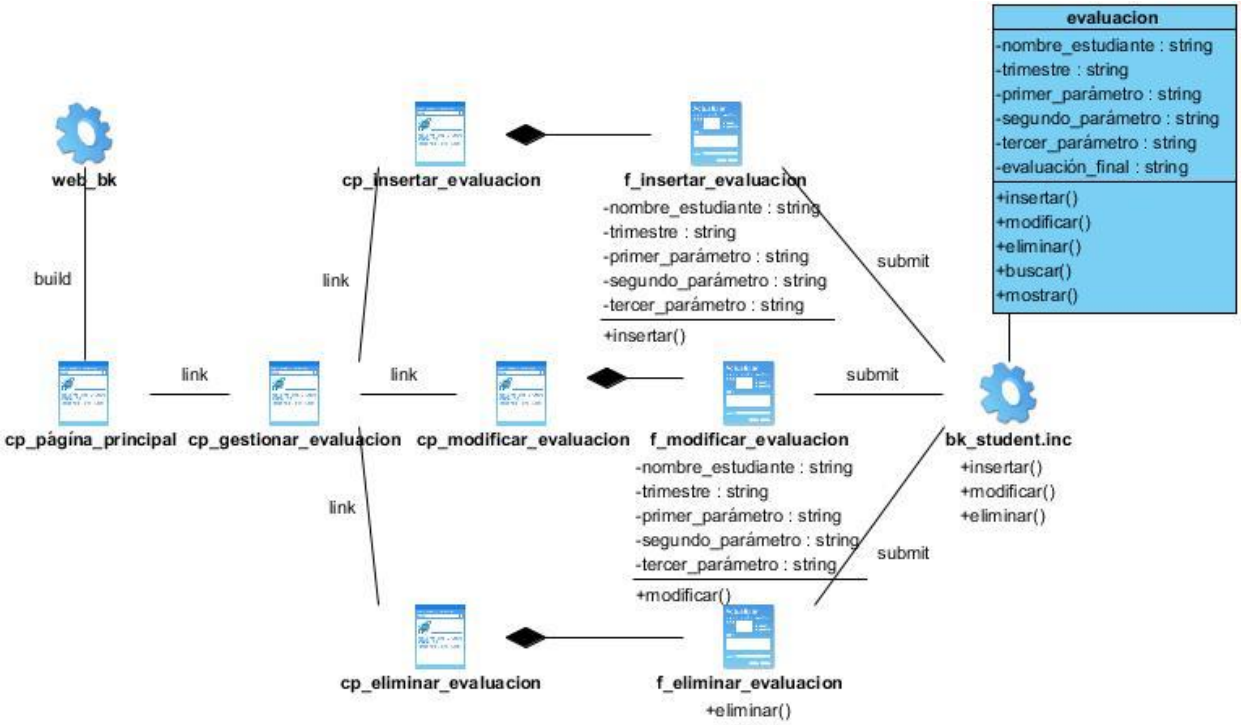
Anexo 4: Diagrama de Clase del diseño para CU Gestionar Trabajador.



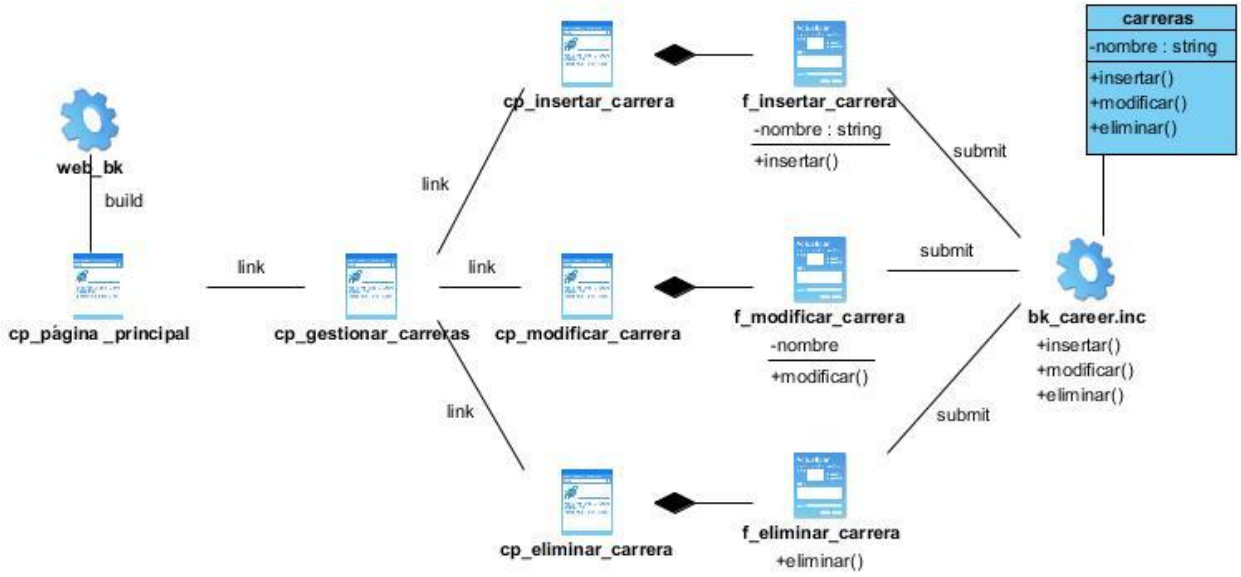
Anexo 5: Diagrama de Clase del diseño para CU Gestionar Local.



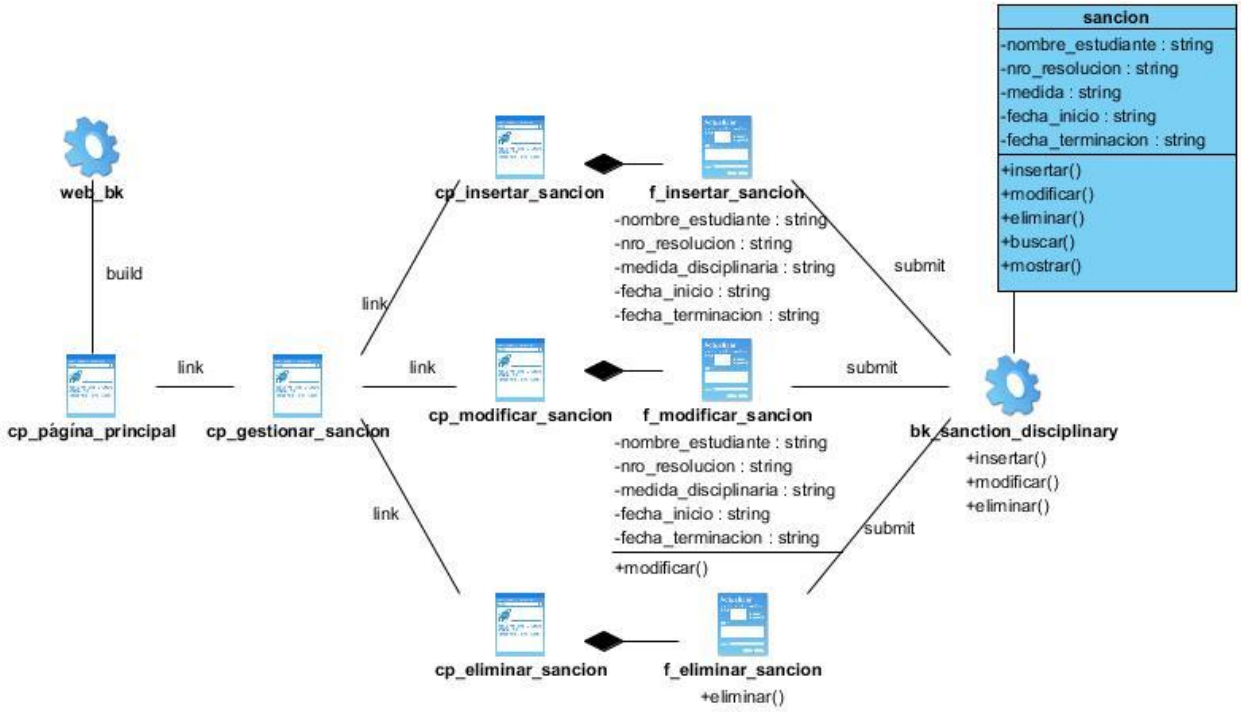
Anexo 6: Diagrama de Clase del diseño para CU Gestionar Medio Básico.



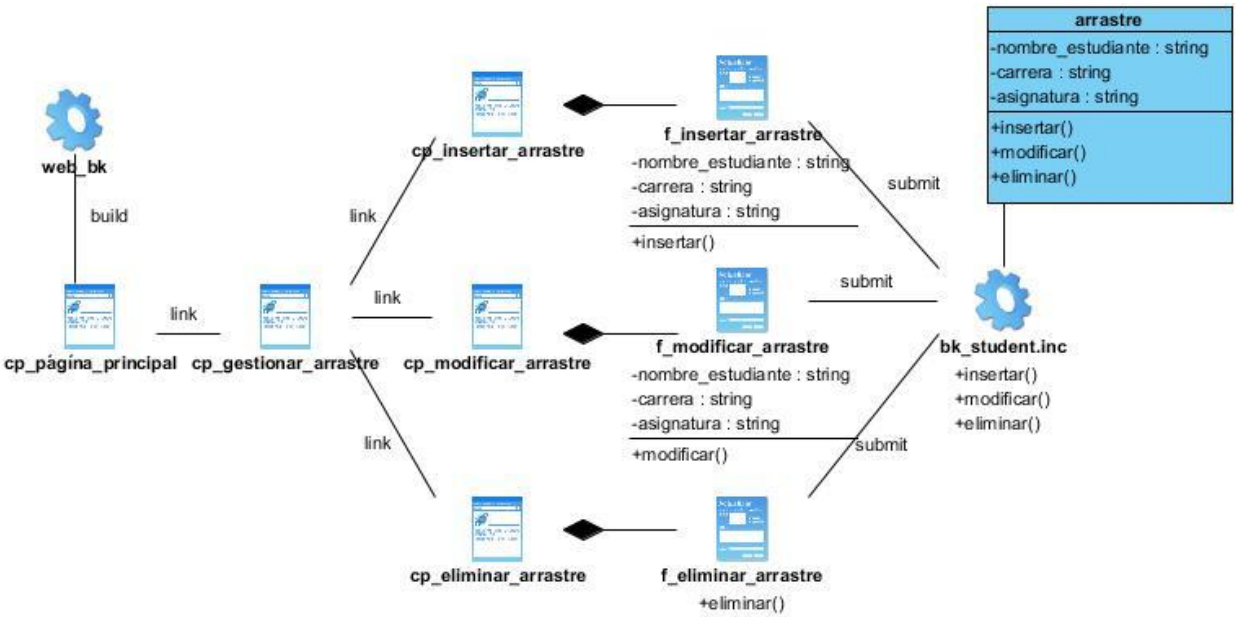
Anexo 7: Diagrama de Clase del diseño para CU Gestionar Evaluación.



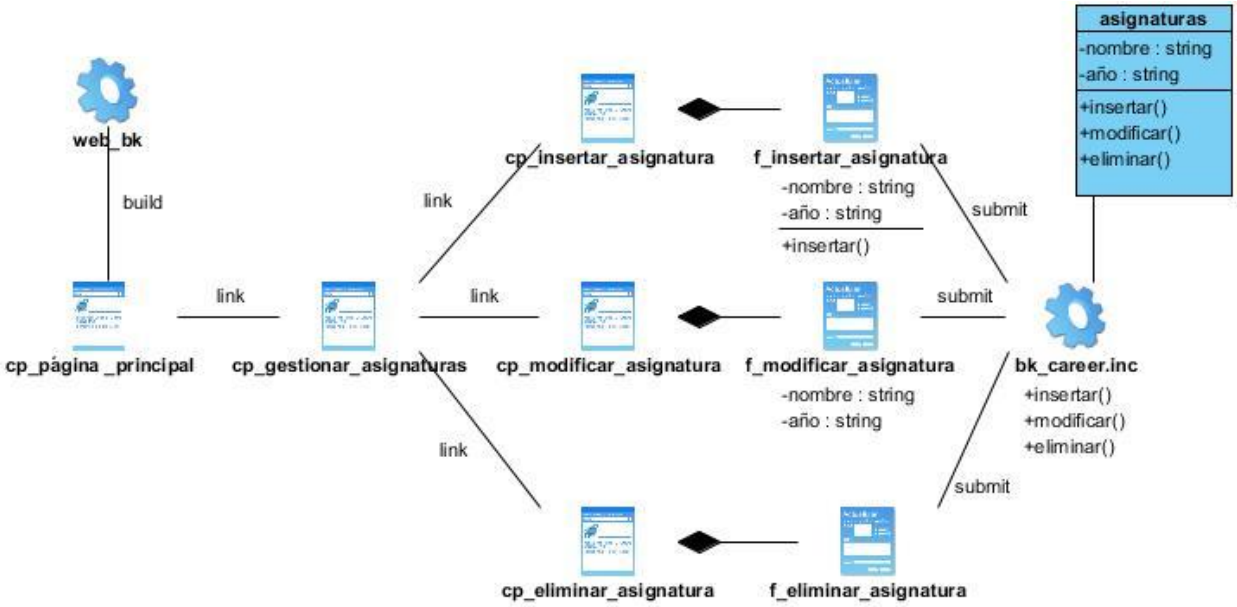
Anexo 8: Diagrama de Clase del diseño para CU Gestionar Carrera.



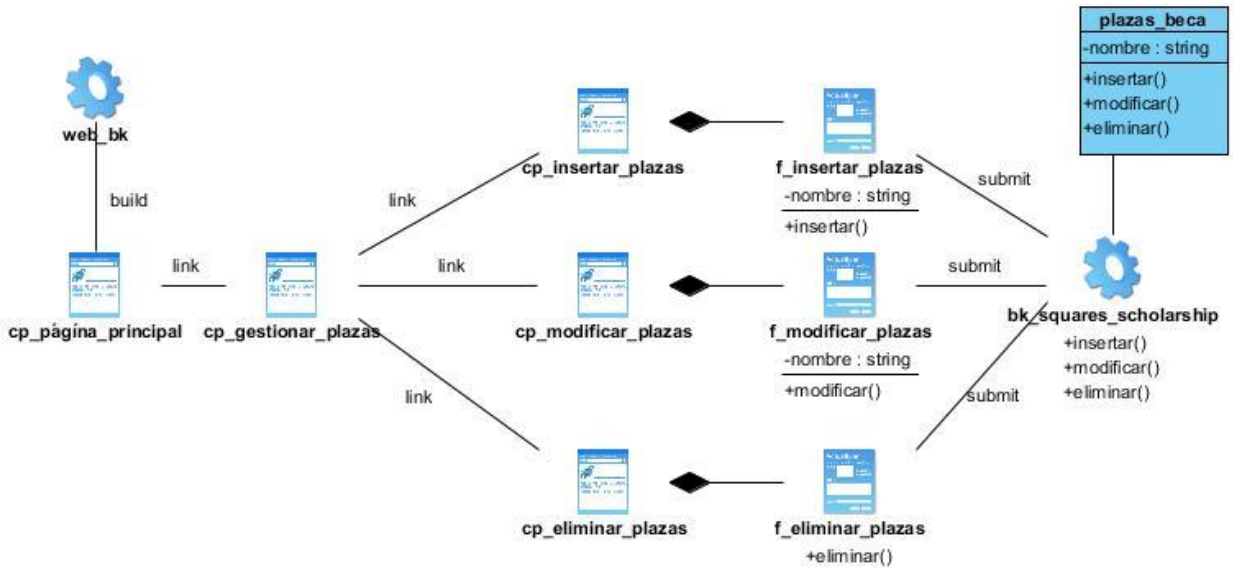
Anexo 9: Diagrama de Clase del diseño para CU Gestionar Sanción Disciplinaria.



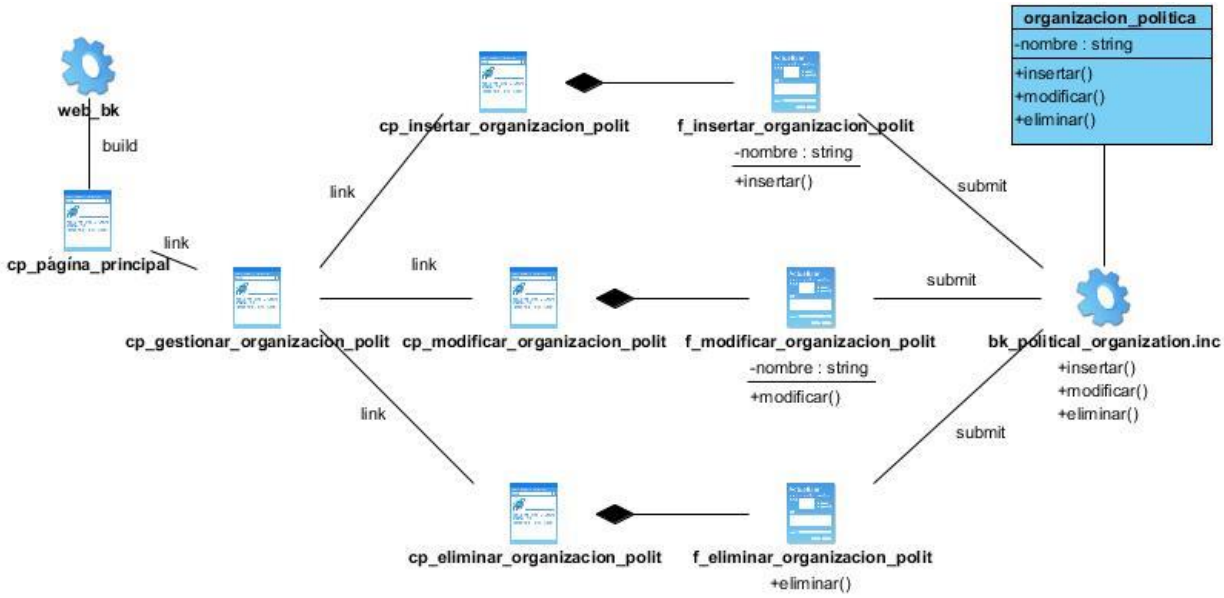
Anexo 10: Diagrama de Clase del diseño para CU Gestionar Arrastre.



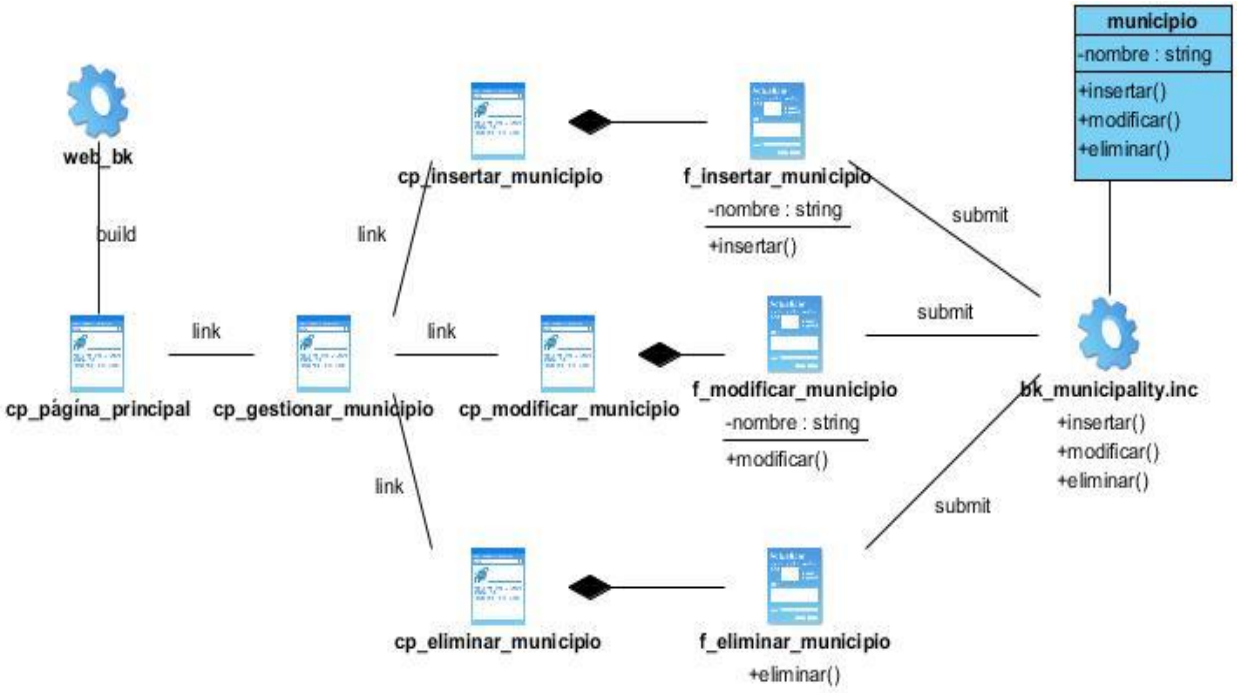
Anexo 11: Diagrama de Clase del diseño para CU Gestionar Asignaturas.



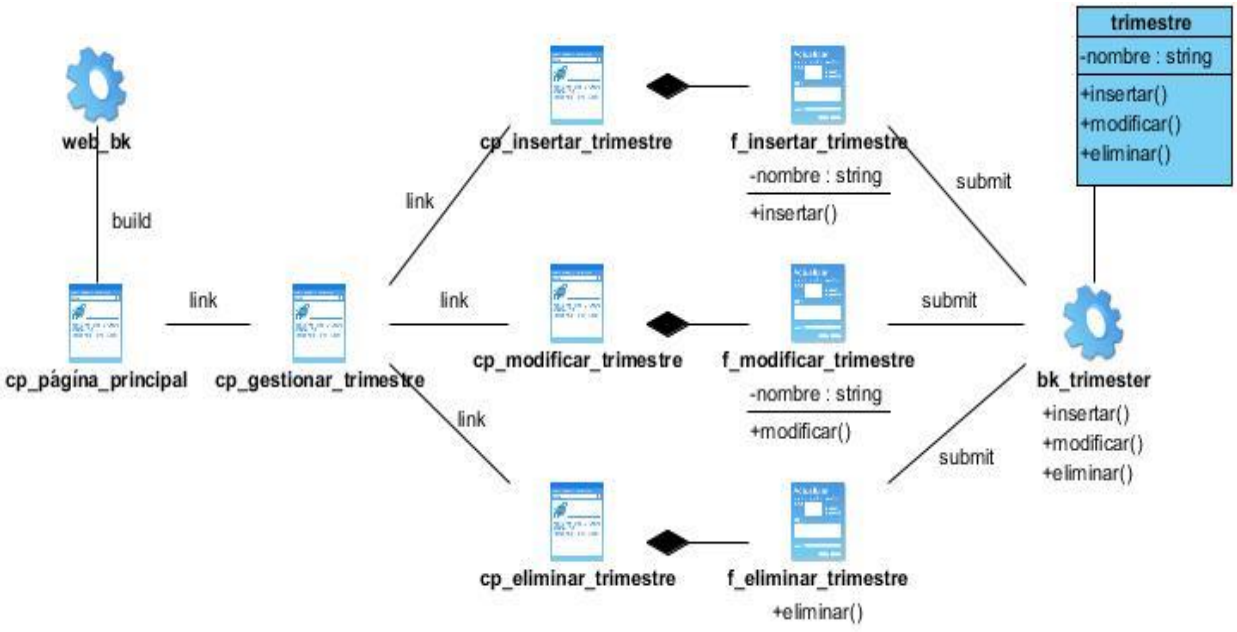
Anexo 12: Diagrama de Clase del diseño para CU Gestionar Plazas Trabajo de la Beca.



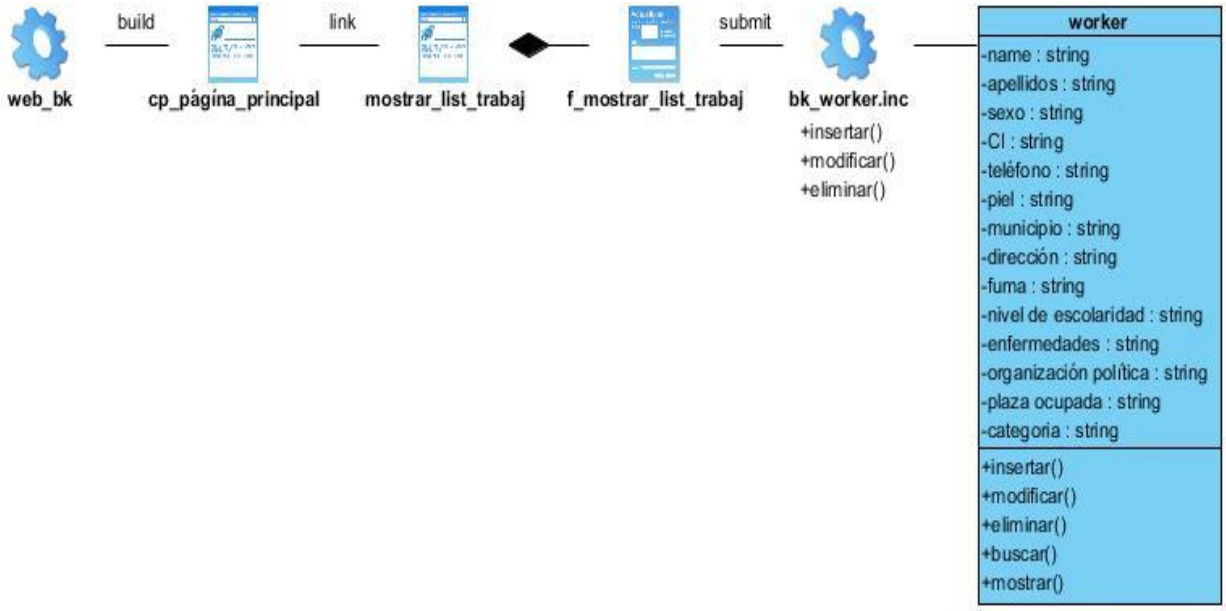
Anexo 13: Diagrama de Clase del diseño para CU Gestionar Organización Política.



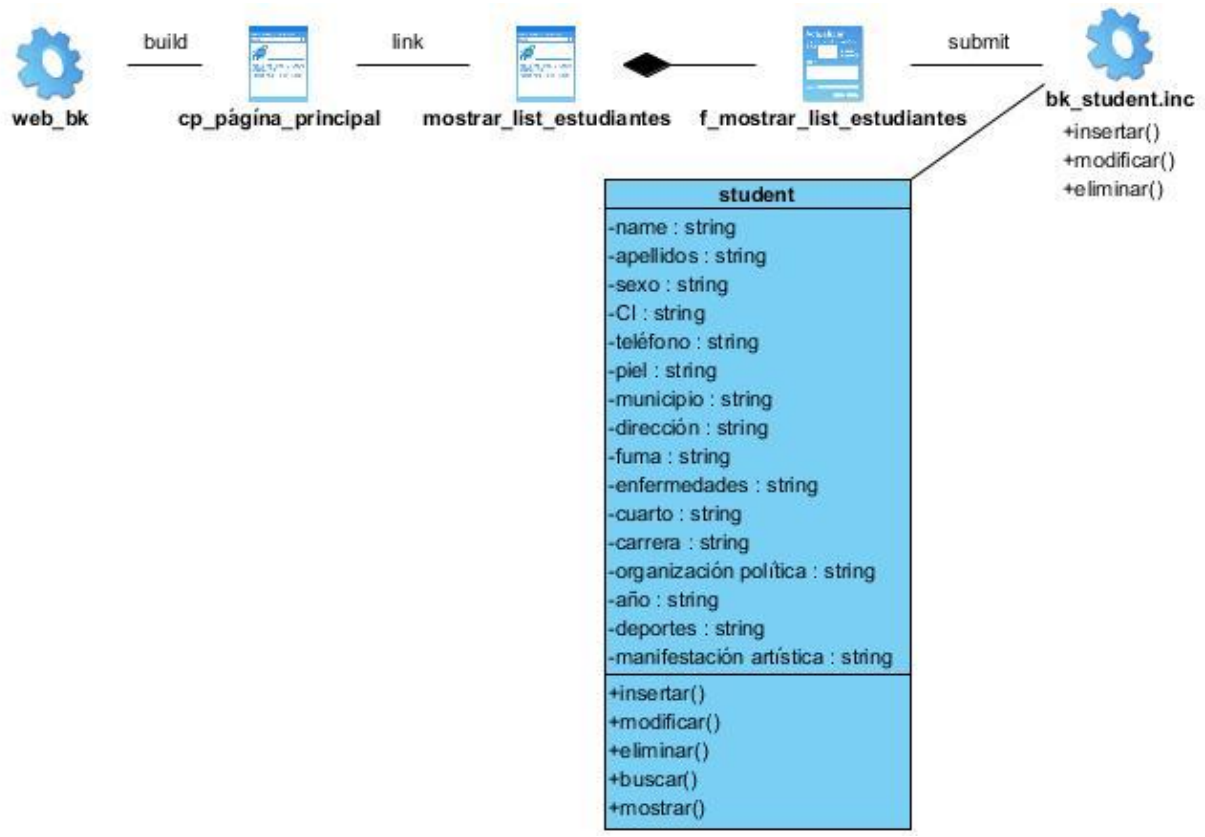
Anexo 14: Diagrama de Clase del diseño para CU Gestionar Municipio.



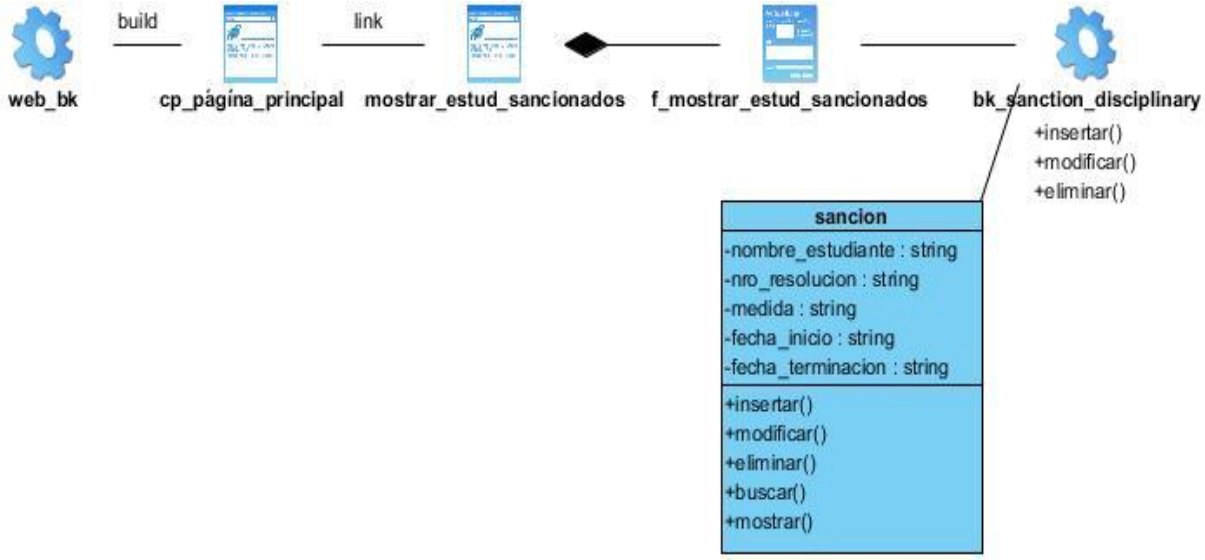
Anexo 15: Diagrama de Clase del diseño para CU Gestionar Trimestre.



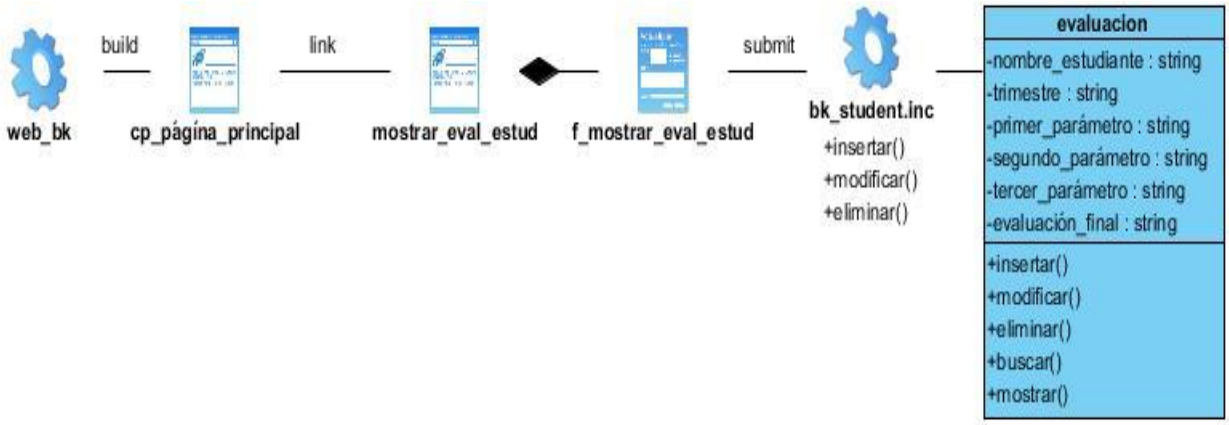
Anexo 16: Diagrama de Clase del diseño para CU Mostrar Listado de Trabajadores.



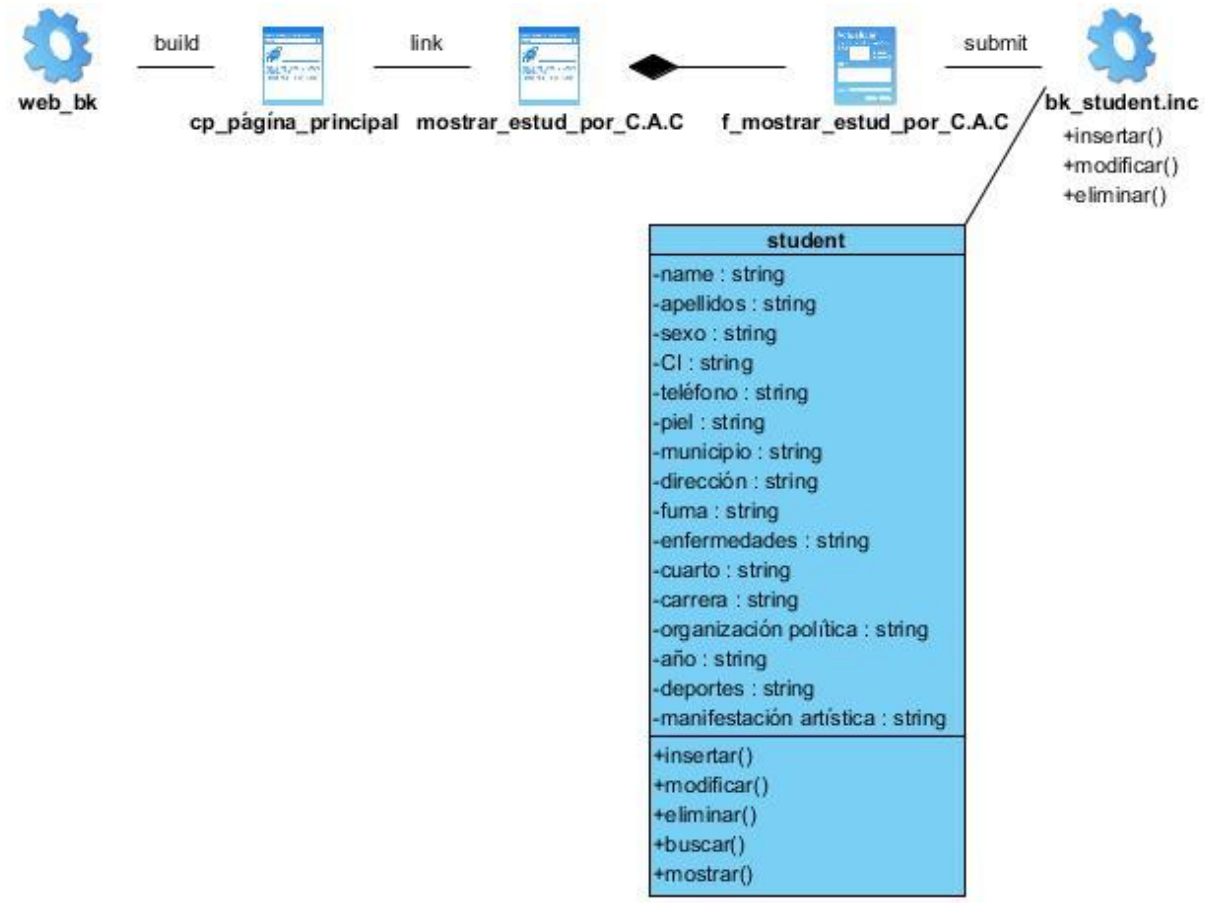
Anexo 17: Diagrama de Clase del diseño para CU Mostrar listado de estudiantes.



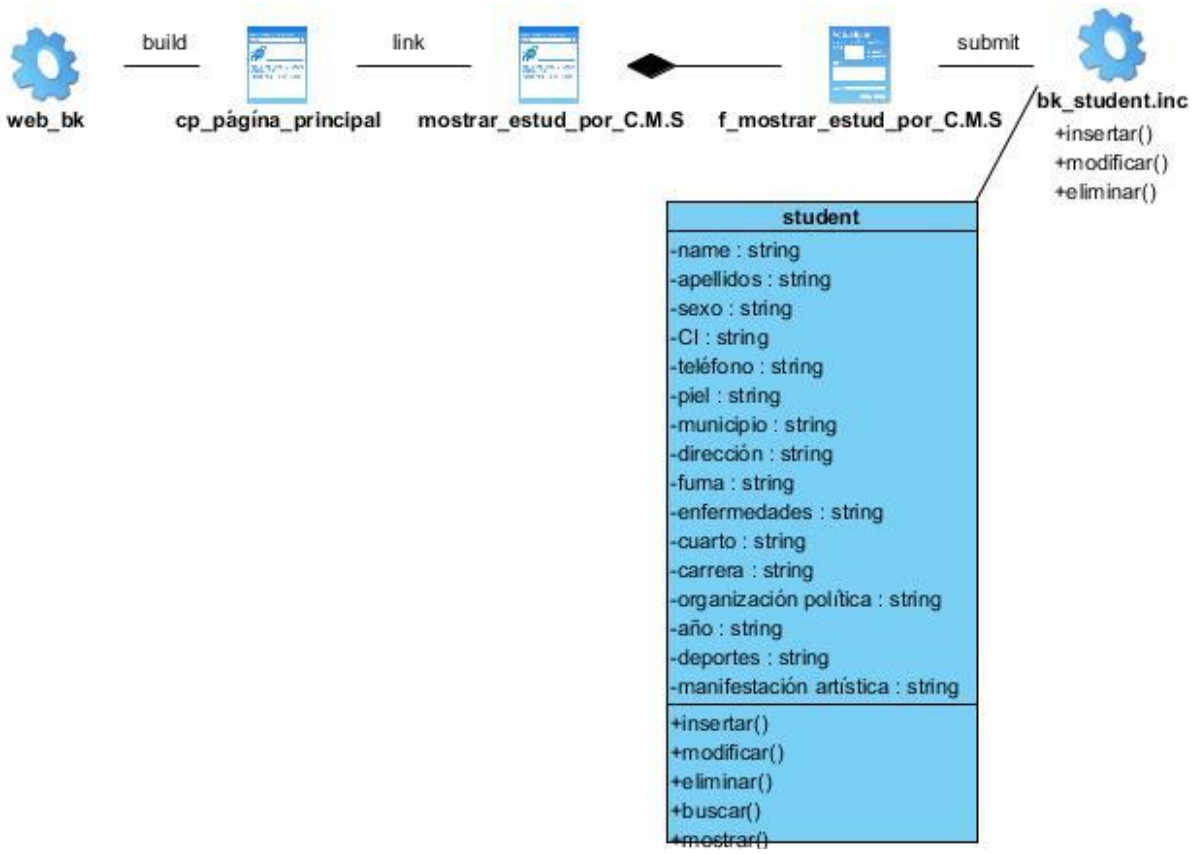
Anexo 18: Diagrama de Clase del diseño para CU Mostrar Estudiantes Sancionados.



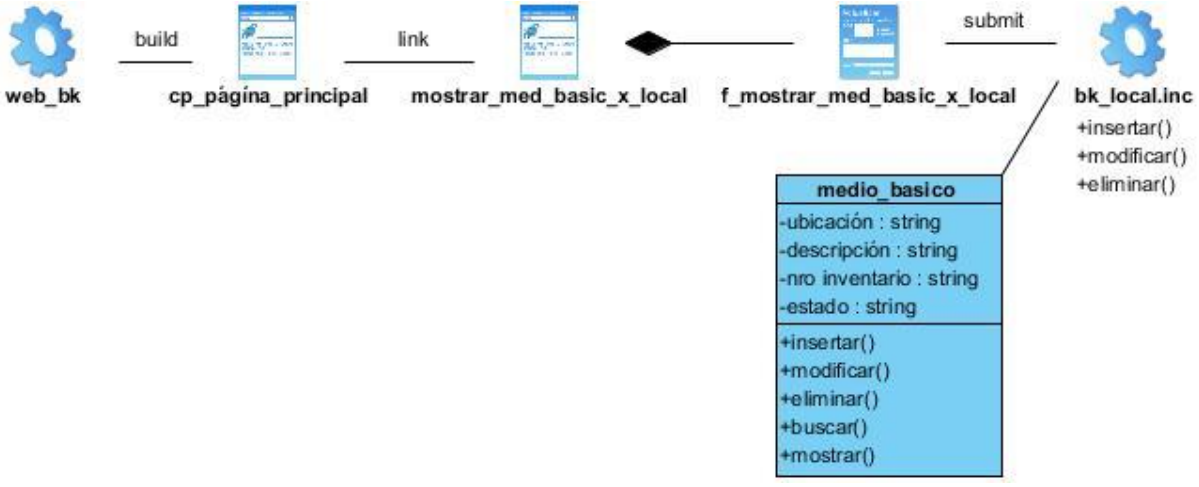
Anexo 19: Diagrama de Clase del diseño para CU Mostrar Evaluación de los Estudiantes.



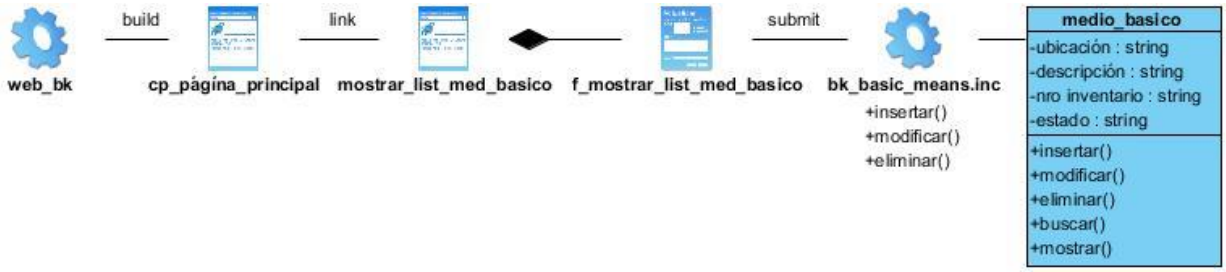
Anexo 20: Diagrama de Clase del diseño para CU Mostrar Estudiantes por Carrera, Año y Cuarto.



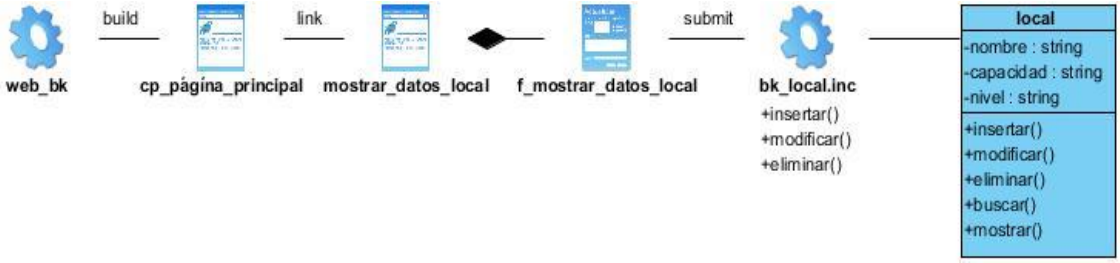
Anexo 21: Diagrama de Clase del diseño para CU Mostrar Estudiantes por Carrera, Municipio y Sexo.



Anexo 22: Diagrama de Clase del diseño para CU Mostrar Medios básicos por Local.



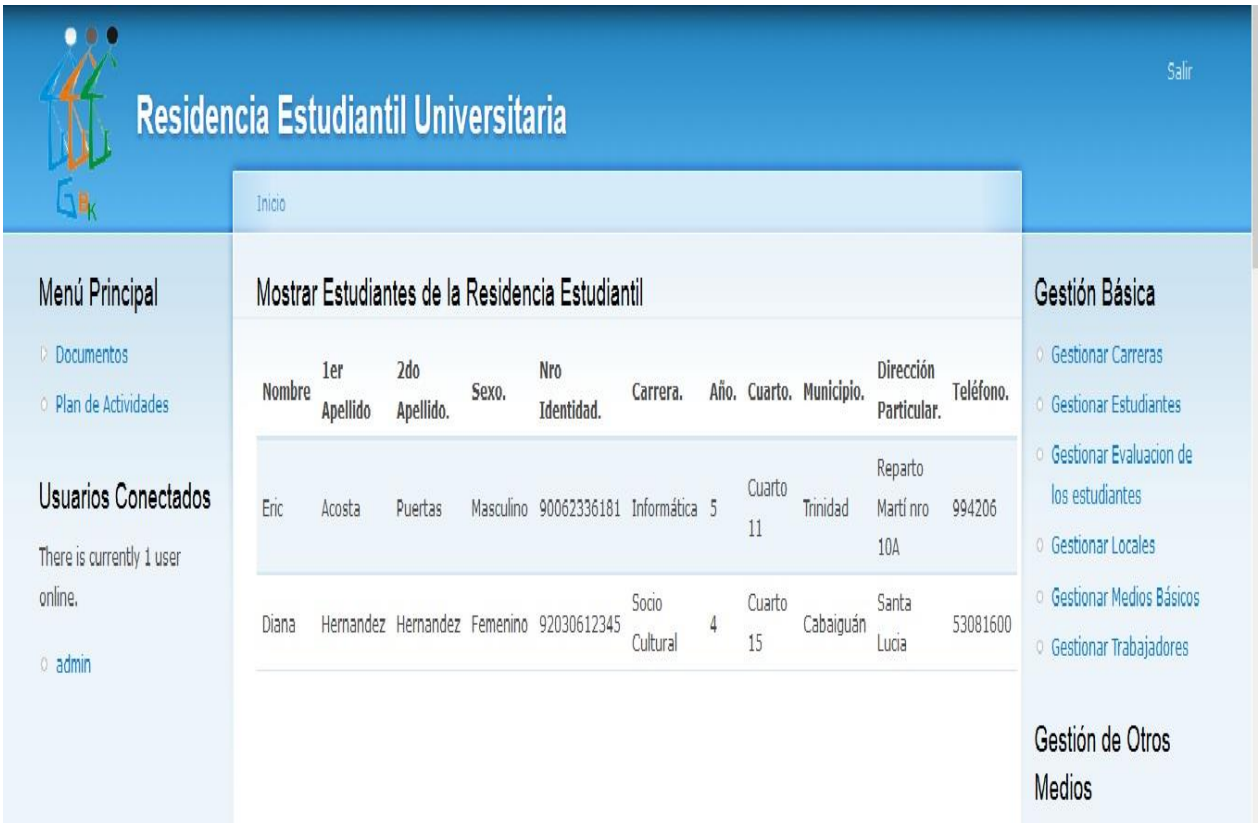
Anexo 23: Diagrama de Clase del diseño para CU Mostrar Listado de Medios Básicos



Anexo 24: Diagrama de Clase del diseño para CU Mostrar Datos de un Local.

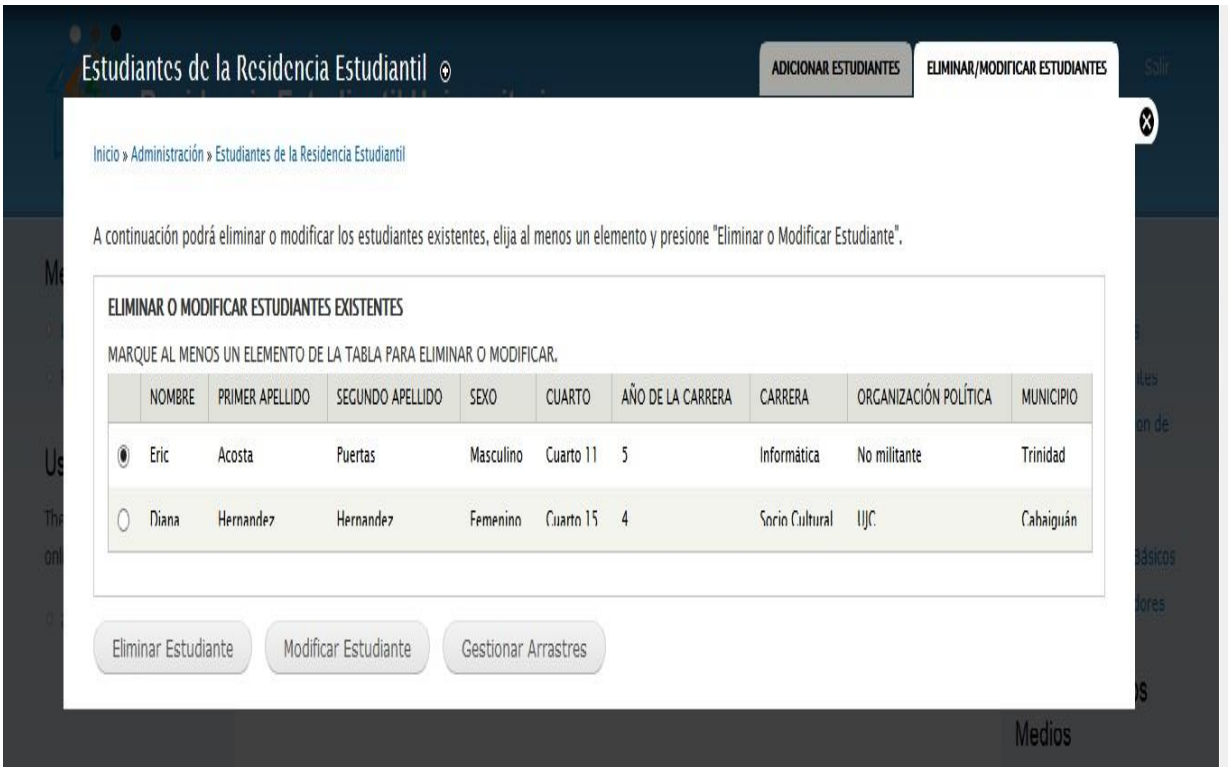
Anexo 25: Página inicio.

Anexo 26: página Principal.

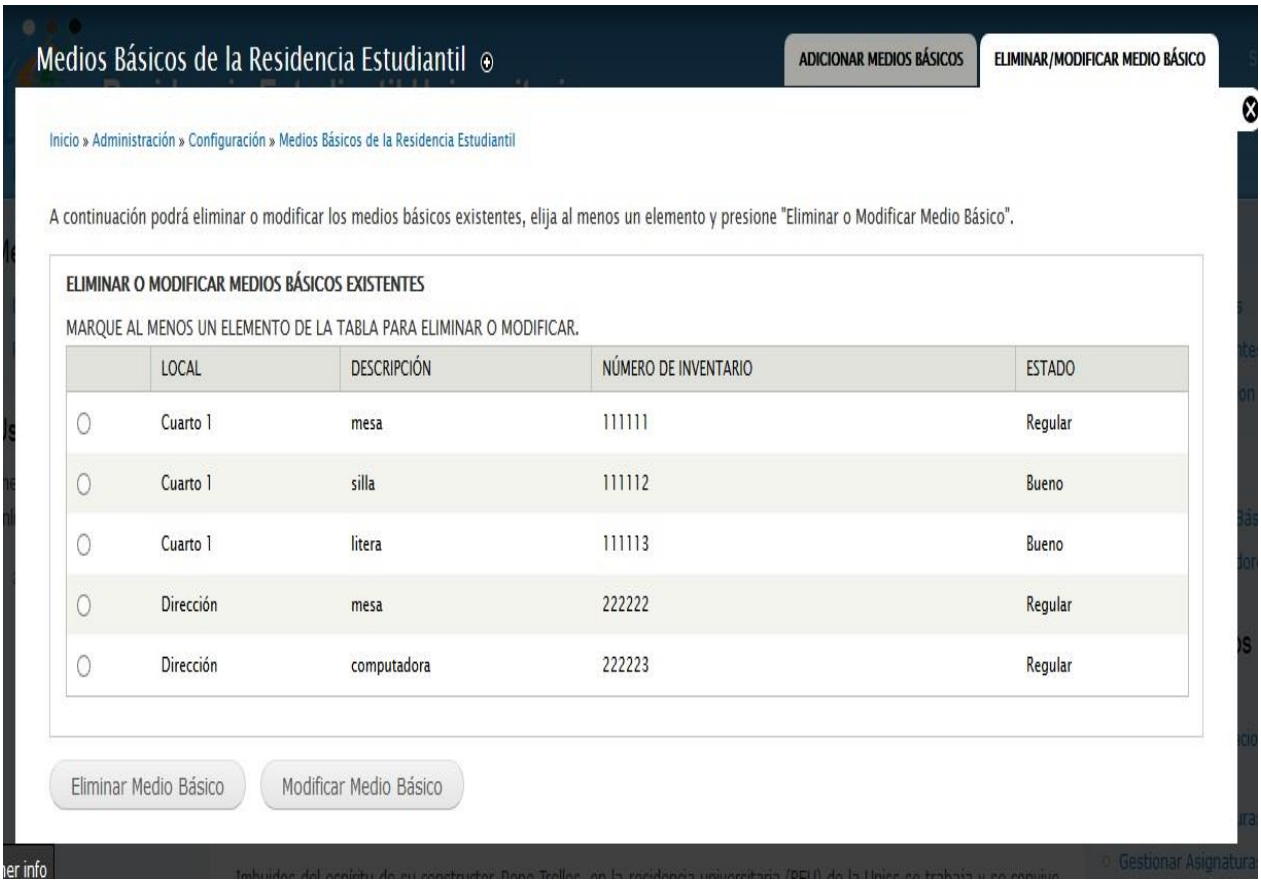


Anexo 27: Mostrar listado de estudiantes.

Anexo 28: Mostrar Medios Básicos por Local.



Anexo 29: Modificar o eliminar estudiante y gestionar arrastre



Anexo 30: Modificar o eliminar medio básico.