Universidad de las Ciencias Informáticas

Facultad 4

Módulos para la gestión de la información de los procesos de Cotización e Ingreso a las filas de la UJC en la UCI

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores:

Yenisel Díaz Llanes

Jorge Andy González Sánchez

Tutores:

Ing. Leonardo Rodríguez González

Ing. Carlos Montenegro Amador

La Habana, Junio de 2016

‘‘Año 58 de la Revolución’’

Declaración de Autoría

Declaración de Autoría

Declaramos ser los autores del presente trabajo de diploma y otorgamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

_____	_____
Firma del Autor	Firma del Autor
Yenisel Díaz Llanes	Jorge Andy González Sánchez
_____	_____
Firma del Tutor	Firma del Tutor
Ing. Leonardo Rodríguez González	Ing. Carlos Montenegro Amador

Dedicatoria

De Yenisel

Dedico este trabajo, todo el esfuerzo que he realizado para verlo hecho realidad, mis años de estudio y dedicación:

A mi madre Zoraida: *por ser la mejor madre del mundo, inspirarme confianza y ser una mujer excepcional.*

A mi padre Noel: *por transmitirme la fuerza que se necesita día a día para mantenerse en pie.*

A mi hermana Yésica: *por ser la persona que más quiero en este mundo.*

A mis abuelas Yiya y Zora: *que siempre han querido lo mejor para mí y por más que abuelas hacer el papel de madres, por malcriarme y apoyarme en toda decisión.*

A mi novio Jorge Andy: *por todo su apoyo y amor incondicional.*

De Jorge Andy

Dedico este trabajo de diploma

A mis padres: *por ser mi gran apoyo día tras día, por brindarme su infinito amor y cariño, por sus consejos, por su firmeza, por confiar en mí y por estar presentes cada vez que los necesito, por haberme guiado siempre por el buen camino, por demostrarme seguridad y no dejarme caer nunca.*

A mi novia: *por ser mi compañera, amiga y por los duros momentos que pasamos juntos.*

A mis amigos: *a todos aquellos que en un momento u otro estuvieron a mi lado y compartimos aventuras inolvidables.*

A mis profesores*: por haber cultivado en mí buenos valores y adquirir nuevos conocimientos y habilidades. A todos ellos gracias.*

Agradecimientos

De Yenisel

Quisiera agradecer primeramente a las dos personas que les debo la vida y todo lo que soy, que desde el primer momento me educaron y enseñaron a ser una persona de bien, gracias por su comprensión, su confianza y apoyo incondicional mami y papi.

A mi abuela Zora por querer darme siempre lo mejor y por sobre todo a mi abuela Yiya por ser mi segunda madre y padre, por alimentarme, cuidarme, malcriarme y estar pendiente siempre de mí.

A mi novio y compañero de tesis Jorge Andy por haber depositado toda su confianza en mí para lograr este sueño, por apoyarme y estar conmigo en los buenos y malos momentos.

A mis tías Nory y Nordin por ayudarme en todo momento y de una forma u otra preocuparse siempre por mí.

A mis tutores Carlos y Leonardo porque sin su apoyo este sueño no se hubiera hecho realidad, gracias a ambos por su tiempo, paciencia, por brindarme sus conocimientos y ser ejemplos a seguir.

A mis suegros y a mi cuñada que desde que llegué a sus vidas me acogieron como una hija y hermana más.

A mis amigos, los viejos y los nuevos sin orden de preferencia, gracias por estar siempre a mi lado. Agradecer a Yinelys, Cleydis y Yaritza, mis amigas de siempre, por compartir conmigo momentos de tristezas y alegrías. A mis viejas compañeras de apartamento Dalianne y Lisandra y a las nuevas en especial a Yaiselín por estar siempre ahí para mí. En fin gracias a todos los que de una forma u otra aportaron su granito de arena para que todo este tiempo me sintiera como en casa.
Gracias a todos por confiar en mí, espero que estén orgullosos.

Agradecimientos

De Jorge Andy

A mis padres por haberme guiado en estos años, en especial a mi madre que estuvo presente cada vez que la necesité, ha sido madre, padre y amiga.

A mi hermana Arletys que me ha soportado todos estos años y ha sabido apoyarme en todas mis decisiones.

A mis suegros, mi cuñada, a Yiya, Danilo y Nordin que me aceptaron desde el principio como uno más de la familia.

A mi tía Barbarita y familia que me han apoyado a pesar de la distancia.

A Jose Alberto, que le voy a estar eternamente agradecido, ninguna palabra será suficiente para demostrarle mi gratitud.

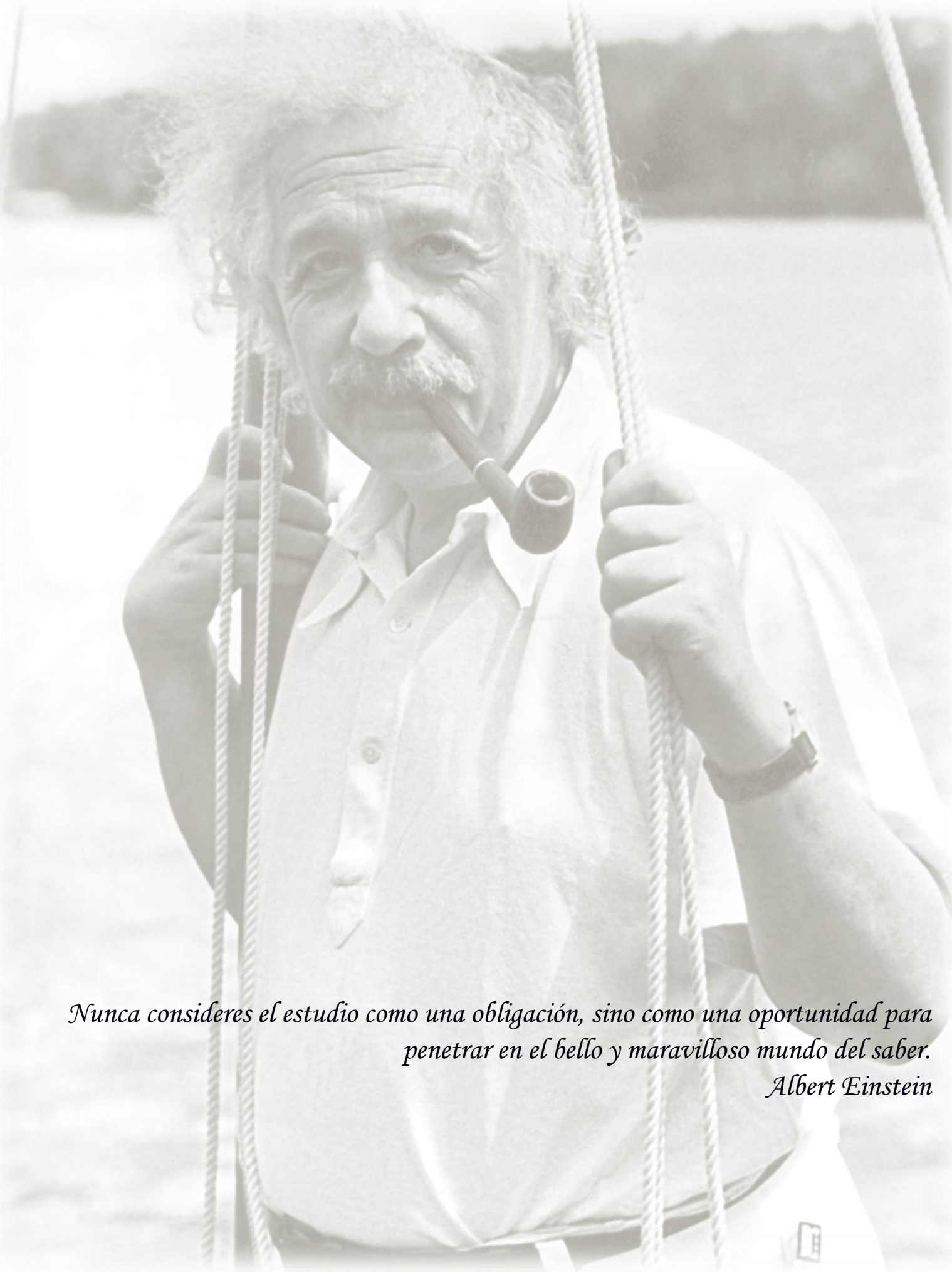
A yeni que no solo ha sido novia, sino también mi compañera de tesis,!!!puchinga te quiero!!!.

A mis tutores por haberme guiado a lo largo de esta ardua labor.

A Gino por estar cuando lo necesité.

Y a todos aquellos que de una forma u otra han sabido apoyarme durante estos cinco años.

A la gente de la zona, del edificio, a Yaiselín, Henrry, Rogelio, el flaco, Lester, a Mailín, Franly, Oriesniel, Reydel, el guille, a Yunierkis, Roylán, Pedro, Humberto y discúlpeme si se me queda alguien sin mencionar, pero en cinco minutos es difícil resumir a todas aquellas personas que de una forma u otra han aportado su ayuda.



Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber.

Albert Einstein

Resumen

El creciente desarrollo de la industria de software cubano y el avance logrado en la informatización del país, ha hecho crecer un interés continuo de las organizaciones políticas y de masas en la informatización de sus procesos orgánicos. En la Universidad de las Ciencias Informáticas existe una aplicación para la gestión de los procesos sustantivos de la Unión de Jóvenes Comunistas, la cual fue desarrollada para facilitar la ejecución de algunos de estos, entre los que se encuentra la gestión de los cierres de funcionamientos y del ID2, no teniendo presente la información que se genera de los procesos de ingreso a las filas y la cotización. Actualmente los reportes de estos son realizados de forma manual apoyados por el uso de herramientas ofimáticas, lo que genera un alto volumen de trabajo provocando búsquedas engorrosas y en algunos casos pérdida de la misma. El presente trabajo de diploma tiene como objetivo desarrollar los módulos: ‘‘Ingreso a las filas’’ y ‘‘Cotización’’ que permitan mejorar el control de la información de ambos procesos. Se transitó siguiendo la metodología de desarrollo AUP en la variante UCI, basándose en el marco de trabajo Symfony y el lenguaje PHP. Se realizaron los flujos de Modelado, Implementación y Prueba indicados por la metodología elegida en su escenario dos, resultando los módulos que responden a los requerimientos asociados al ingreso a las filas de la UJC y el control de la cotización.

Palabras clave:

Cotización, informatización, Ingreso a las filas, UJC

Contenido

Contenido

Introducción

En los Estatutos Unión de Jóvenes Comunistas (UJC) se plantea que la UJC es ‘‘la organización juvenil del Partido Comunista de Cuba (PCC) que agrupa, dirige y coordina a la vanguardia de la juventud cubana’’. Asume la responsabilidad de dar continuidad a la revolución y al modelo socialista, forjado por varias generaciones de cubanos (Comunistas, 2014).

La UJC como organización política tiene lugar en cada centro donde existan al menos cinco militantes y en la mayoría de los casos se encuentra estructurada como sucede en la Universidad de las Ciencias Informáticas (UCI), donde existe un Comité de la UJC al cual se subordinan varios Comités Primarios (CP) como estructuras intermedias entre el Comité UJC y los Comités de base (CB). Un CP es el encargado de orientar, planificar, organizar, dirigir y controlar el trabajo de la UJC en cada una de las facultades de la UCI.

En cada uno, por lo general, se desarrollan los mismos procesos sustantivos de la organización juvenil, entre ellos se encuentran:

- ✓ Gestión y reporte de actas
- ✓ Análisis de incumplimientos
- ✓ Recogida de cotización
- ✓ Aplicación de sanciones
- ✓ Cierre de funcionamiento
- ✓ Incorporaciones, traslados y bajas de militantes
- ✓ Constitución y desintegración de Comité de Bases
- ✓ Evaluación de los militantes
- ✓ Crecimiento y Construcción en la UJC
- ✓ Diagnóstico sociopolítico

Estos procesos sustantivos son dirigidos por diferentes frentes o esferas y tienen como objetivo garantizar el correcto funcionamiento de la organización.

Hoy día existe una aplicación nombrada: “Sistema integral para la gestión de los procesos sustantivos de la UJC en la UCI”, la cual fue desarrollada para facilitar la ejecución de algunos de los procesos de la

organización, entre los que se encuentra la gestión de los cierres de funcionamientos (resumen estadístico de las reuniones de CB) y la gestión del ID2 (incorporaciones, traslados y bajas de militantes).

En cuanto al cierre del funcionamiento, existe la necesidad de que el mismo sea exportado a Excel para que los datos a entregar al municipio, a la provincia o a la nación se carguen directamente de la Base de datos y no haya que realizarlo de forma manual. Además con relación a las actas de las asambleas ordinarias que se almacenan en la misma, no realiza el chequeo de los acuerdos de las reuniones anteriores, por lo que a la hora de verificar el cumplimiento de las tareas el proceso se torna un poco engorroso debido a que la información necesaria para el mismo puede estar duplicada o extraviada por el exceso y la no organización de la misma.

En estos momentos unos de los procesos que presentan dificultades en la organización a la hora de llevar a cabo una gestión organizada y verificar su comportamiento lo constituyen el ingreso a las filas y cotización debido al cúmulo de información que se maneja en ambos.

Las organizaciones de base y organismos de dirección de la UJC desarrollan el proceso de ingreso a la organización con los jóvenes cubanos, cumpliendo lo establecido en el artículo 2 de los Estatutos que señala: “Pueden ingresar a la Unión de Jóvenes Comunistas los jóvenes cubanos de 16 a 30 años que se hayan destacado por su fidelidad y consagración en el cumplimiento de las tareas de la Revolución y su activa participación en el trabajo, en el estudio y en la defensa de la Patria; que se distingan por su laboriosidad, responsabilidad, honestidad, honradez, sencillez, modestia, sensibilidad humana, voluntad de vencer y demás valores que caracterizan a un revolucionario’’ (Comunistas, 2014).

En la actualidad, no existe una vía digital por la que los jóvenes interesados en el proceso de ingreso hagan conocer sus razones, expectativas y criterios del por qué desean ingresar a la misma en el momento deseado. Además, no existe un sistema capaz de gestionar autobiografías con los datos correspondientes al joven que haya hecho la solicitud personal. Así como tampoco se cuenta con un canal de información que detalle el estado en que se encuentran los crecimientos por parte del Comité UJC UCI.

En el artículo 78 del Reglamento de la UJC se plantea que se establecen como vías de ingreso a la Organización el ser electo en la Asamblea de jóvenes ejemplares (vía fundamental de ingreso) o la asamblea de consulta con las masas, (donde se analizan las solicitudes personales). Ambas asambleas

requieren un acta donde queda plasmado la correcta realización del proceso, la misma no es como las que se redactan en las demás reuniones de la UJC ya que en estas reuniones no solo participan militantes sino también miembros del universo juvenil (jóvenes que no son militantes de la UJC), siendo este un problema debido a que la aplicación existente no permite incluir personas del universo juvenil en sus actas.

En cuanto al esclarecimiento de la trayectoria del joven, se debe profundizar y comprobar a través de la autobiografía, la comprobación y la entrevista individual. Aún no se cuenta con un modelo oficial que haga de esta última un proceso fluido para conocer los criterios y opiniones que él directamente pueda aportar. En el caso de los jóvenes trabajadores no se encuentra digitalizado el modelo de comprobación del DNI (Documento Nacional de Identidad), para la verificación de datos en archivo en busca de antecedentes penales y entregarlo al Comité Municipal de la UJC.

Para la realización del proceso a cada joven, el CB aprueba un dúo de crecimiento por joven y confecciona un cronograma para el desarrollo del mismo que no deberá exceder de 90 días desde su inicio hasta la entrega del carné. Este dúo es el encargado de llenar el expediente del militante teniendo en cuenta los aspectos antes mencionados relacionados con el esclarecimiento de la trayectoria del joven, lo que se torna un poco complicado debido a que el expediente cuenta con determinados campos a llenar donde se recogen los datos generales del joven y otras informaciones de interés relacionados con el proceso de ingreso, donde actualmente la mayoría no son actualizados.

Los jóvenes a los que se les apruebe el ingreso son incorporados oficialmente a la Organización de Base (OB). Aquellos, que por una razón u otra no se les apruebe el proceso, se le explicará con un adecuado tratamiento las causas por las que se adoptó tal decisión, no existiendo de forma digital por la que puedan establecer una reclamación cuando no estén de acuerdo con la decisión adoptada.

Una vez que los militantes de nuevo ingreso son incorporados al CB. En el mes posterior a la fecha de incorporación, comienzan a cotizar, que es un aporte económico individual para contribuir a sufragar los gastos de la organización en el cumplimiento de su Misión, lo cual incide en su formación, responsabilidad, disciplina, colectivismo y sentido de pertenencia (Comunistas, 2014).

Introducción

La Organización de Base designa un activista de cotización que tiene la responsabilidad de velar que el pago de la cuota establecida de todos los militantes se realice en el tiempo requerido y los montos sean los correspondientes. Actualmente los CB no tienen forma de obtener las estadísticas relacionadas con el proceso de cotización, por lo que desconocen los militantes con más de dos meses sin cotizar, la cantidad de cuotas recuperadas, los militantes que arriban a los 32 años de edad, así como el resumen gráfico del importe del CB ya sea mensual, trimestral o anual,. Además no se puede generar el listado de importe del CB, ni registrar el pago de la misma por cada militante.

Existen tres aspectos fundamentales que son motivo para que los militantes cumplan con sus deberes: las reuniones de la UJC, las relaciones entre militantes en una organización y la cotización debido a que la UJC establece en sus Estatutos que el militante que infringe los principios y normas de la misma o de la sociedad deberá ser sancionado y podrá cumplir las sanciones dentro o fuera de la organización. Por lo que mensualmente en las reuniones realizadas por el CB se analizan los incumplimientos de deberes asociados a lo antes descritos.

En el sistema se manejan además las sanciones aplicadas a cada militante. Este no es capaz de proponer una sanción teniendo en cuenta un deber incumplido por el militante, en este caso, al no pagar la cotización, lo que trae como consecuencia que no se actualice la plantilla ID2 en lo que a sanciones respecta.

Actualmente el control documental del proceso ingreso a las filas de la UJC y cotización son realizados de forma manual, lo que genera un alto volumen de trabajo y de información en formato duro muchas veces duplicados y con errores, una búsqueda engorrosa y en algunos de los casos pérdida de la misma.

A partir de la situación problemática descrita anteriormente se propone como **problema a resolver:** ¿Cómo agilizar la gestión de la información que se genera de los procesos de cotización e ingreso a las filas de la UJC en la UCI?
Con el fin de solucionar el problema, se define como **objeto de estudio** de la presente investigación: La gestión de la información de los procesos sustantivos de la UJC.
Enfocando el **campo de acción en:** La gestión de la información de los procesos de cotización y de ingreso a las filas de la UJC en la UCI.

Introducción

Teniendo en cuenta el problema a resolver se define como **objetivo general:** desarrollar módulos que garanticen la gestión de la información para contribuir a la agilización de los procesos de cotización e ingreso a las filas de la UJC en la UCI para el SIGPS_UCI.

Objetivos específicos:

1. Analizar referentes teóricos relacionado con los procesos de ingreso a las filas y cotización.
2. Definir las tecnologías y herramientas necesarias para el desarrollo de la propuesta de solución.
3. Desarrollar las funcionalidades de la propuesta de solución.
4. Realizar las pruebas de Caja negra, Integración y Aceptación a la solución informática obtenida.

Para guiar la investigación se define como **Hipótesis:** el desarrollo de los módulos ‘‘Ingreso a las filas’’ y ‘‘Cotización’’ que logren la gestión de la información de los Comité de base en la organización, contribuirá a favorecer la agilización de ambos procesos.

Para el desarrollo de la investigación se emplearon los siguientes métodos de la investigación científica:

Métodos Teóricos

- Histórico- Lógico: se empleó en el estudio del estado del arte, para conocer acerca de otras aplicaciones o soluciones similares a nuestra investigación para el desarrollo del problema planteado.
- Analítico-Sintético: se empleó para identificar los elementos más importantes que darán solución al problema después de haber realizado un estudio de las fuentes bibliográficas existentes referentes al tema.
- Modelación: se utilizó en la realización de los diagramas necesarios que modelan la solución para tener una mejor idea del flujo de trabajo de los procesos a informatizar.

Métodos Empíricos

- Observación: permitió estudiar más de cerca el objeto de la investigación, las acciones, causas y consecuencias. Se pudo observar cómo funcionan los procesos de ingreso a las filas y cotización y los principales problemas asociados a estos.

La presente investigación está conformada por tres capítulos, quedando de la siguiente manera:

Introducción

Capítulo 1: Fundamentación Teórica.

Abarca los conceptos asociados al tema, necesarios para la comprensión de la solución del problema planteado. Se exponen los elementos teóricos utilizados en la investigación, describiendo además las tecnologías, metodología, herramientas y lenguajes de programación utilizados en el desarrollo de la solución.

Capítulo 2: Propuesta de solución.

En este capítulo se exponen los elementos que permiten describir la propuesta de solución, tales como: modelo de dominio, requerimientos funcionales y no funcionales, diagramas de casos de uso y la descripción textual de cada uno de ellos incluyendo los prototipos de interfaz de usuario para lograr un entendimiento claro del sistema a desarrollar. Además, se realiza el modelado del diseño de las funcionalidades.

Capítulo 3: Implementación y prueba.

Este capítulo abarca todo lo relacionado con la implementación, es decir se implementan todas las funcionalidades, obteniendo un sistema con los requerimientos solicitados por el cliente. Se describen las pruebas realizadas al mismo una vez que concluye la implementación, para asegurar que cumple con las especificaciones requeridas.

Capítulo 1: Fundamentación teórica

1.1 Introducción

El desarrollo de software es una tarea que se torna complicada para los desarrolladores si no se cuenta con una guía para desarrollar el mismo. En este sentido ha habido un gran incremento tecnológico durante los últimos años y han surgido las metodologías de desarrollo de software, que son un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los mismos a realizar un nuevo software (Jacobson, et al., 2000), sin embargo es necesario definir primero la naturaleza del software antes de elegir un determinado ciclo de vida.

La base teórica del presente trabajo está sustentada por las tecnologías, metodología, lenguajes de programación y herramientas a utilizar en el análisis, diseño e implementación de la propuesta de solución.

1.2 Conceptos asociados a la investigación

Sistema de gestión de información

Según la bibliografía consultada, varios autores establecen que los sistemas de gestión de información han sido ampliamente implementados en entornos de negocios como un enfoque sistemático para la recopilación y presentación de la información necesaria para la toma de decisiones (HANSEN, et al.).

Unión de Jóvenes Comunistas

La UJC es la organización política de la vanguardia de la juventud cubana, es la organización juvenil del PCC y la principal cantera para el ingreso a sus filas, forjada en la concepción marxista-leninista, las ideas y la práctica del pensamiento de Maceo, Martí, Mella, Camilo, el Che y Fidel. Asume como misión: contribuir a la educación comunista de las nuevas generaciones, sustentada en el patriotismo, la fidelidad al PCC, la defensa de los más altos valores humanos, el aporte al desarrollo económico y social del país y en el espíritu profundamente antiimperialista e internacionalista que ha distinguido a la Revolución Cubana (Comunistas, 2014).

Capítulo1: Fundamentación teórica

Organizaciones de base

El término organizaciones de base sirve para identificar a las organizaciones de carácter social o político más cercanas a la comunidad a la que sirven (Comunistas, 2014).

Comité de Base

Forma de organización de base de la Juventud Comunista, se crea en las escuelas y centros de trabajo donde existen militantes de la juventud, con funciones muy similares a las de los núcleos del Partido, pues estos militantes de la juventud se definen como las canteras del Partido. En realidad, como mismo hacen los núcleos del Partido, los comités de base supervisan la actividad de los jóvenes, elaboran directivas y ejercen una función de control sobre el resto de los jóvenes de los centros de trabajo y las escuelas (Comunistas, 2014).

Comité Primario

Los CP dirigen el trabajo de la organización entre una y otra Asamblea de Balance, y tienen como funciones más importantes: orientar, planificar, organizar, dirigir y controlar el trabajo de la UJC en la facultad, realizar el diagnóstico sociopolítico partiendo del elaborado por los comités de base, controlar el cumplimiento de las normas establecidas para la cotización de las organizaciones estudiantiles bajo su dirección política, así como aprobar y evaluar el cumplimiento de los planes de trabajo (Comunistas, 2014).

Asamblea de Jóvenes Ejemplares

La Asamblea de Jóvenes Ejemplares es el espacio donde se eligen, por el voto favorable de la mayoría de los trabajadores presentes, los jóvenes que reúnen los requisitos para iniciarles el proceso de crecimiento a las filas de la UJC. Constituye la principal vía de ingreso a la UJC. Será convocada por el Comité de Base (Comunistas, 2014).

Asamblea de consulta con las masas

La Asamblea de consulta con las masas es el espacio donde se somete a consideración de los trabajadores, estudiantes o combatientes, a los jóvenes que entregaron por escrito a la dirección del

Comité de Base la solicitud personal. Una vez recogidos los criterios el Comité de Base decidirá a quién se le iniciará el proceso de ingreso (Comunistas, 2014).

Diagnóstico sociopolítico

El diagnóstico sociopolítico constituye la herramienta principal del trabajo cotidiano en la Organización de Base que le permite, entre otros aspectos significativos, conocer mejor a los jóvenes y ejecutar las acciones que lo preparen conscientemente para su futuro ingreso a la UJC (Comunistas, 2014).

Crecimiento y construcción en la UJC

El crecimiento se describe en 6 pasos fundamentales:
1. Asamblea de jóvenes ejemplares o de consulta con las masas (presencia del organismo superior).
2. Profundización, entrevista y comprobación al joven para el esclarecimiento de su trayectoria.
3. Reunión del Comité de Base para evaluar y decidir la aceptación de los jóvenes en proceso (presencia del organismo superior).
4. Aceptación o no por parte de organismo superior.
5. Reunión del Comité de base con cada joven en proceso para darle conclusiones.
6. Asamblea de presentación a las masas de los nuevos militantes y acto de entrega del carné (Comunistas, 2014).

Aplicación de sanciones

El militante que infrinja los principios y normas de la UJC o de la sociedad, deberá ser sancionado y podrá cumplir la sanción fuera o dentro de la organización. Las sanciones que la UJC aplica a sus militantes tienen como objetivos:

- Contribuir a su educación comunista, corregir sus defectos y errores e inculcarles la necesidad de la disciplina consciente.
- Mantener la unidad y la fuerza de las filas.

Estas pueden ser Internas (Amonestación, Separación del cargo, Limitación temporal de derechos).
Externas (Separación de las filas, Expulsión de las filas) (Comunistas, 2014).

Cotización

Aporte económico individual para contribuir a sufragar los gastos de la organización en el cumplimiento de su Misión, lo cual incide en su formación, responsabilidad, disciplina, colectivismo y sentido de pertenencia (Comunistas, 2014).

1.3 Análisis de Sistemas similares

Los sistemas existentes a nivel nacional que pudieran analizarse como antecedentes, son en su mayoría Portales Web y sistemas que solo informatizan una parte de un proceso sustantivo o un proceso en general. A continuación se realiza un análisis del estudio de algunos de estos sistemas. A nivel internacional no se tienen conocimientos de sistemas relacionados con la UJC debido a que esta es una organización creada en Cuba, con el objetivo de agrupar a la vanguardia de la juventud cubana, pero sí existen sistemas encargados de la gestión de información ya sea de un área, empresa u organización.

Sistemas similares a nivel internacional

Workmeter, herramienta de mejora de la productividad en el uso de aplicaciones: permite, mediante datos objetivos y de forma automática, medir y gestionar la actividad productiva de los empleados así como el uso de las aplicaciones proporcionando un exhaustivo listado de las mismas y del tiempo invertido en cada una de ellas. Facilita la toma de consciencia de cómo se reparte el tiempo de trabajo (en qué tareas se invierte ese tiempo a lo largo del día), permite valorar de forma objetiva la capacidad de producción y el esfuerzo generado y finalmente sustenta el análisis y la búsqueda de caminos de mejora. Las ventajas de aplicación son múltiples ya que sirve como herramienta base para la información con los propios trabajadores y mejora de los procesos y sistemas que se lleva a cabo en la empresa. Finalmente, dispone de un informe del tiempo productivo invertido en cada proyecto, pudiendo así comprobar si se están cumpliendo las planificaciones. El estudio de esta herramienta permitió establecer una relación con el proceso de cotización, ofreciendo un método por el cual brindar información de la cotización a los militantes del CB y el cumplimiento del plan de ingresos mensual. (workmeter, 2015).

GestPeople, sistema de Gestión de personas: es un programa ideal para llevar a cabo todo tipo de gestiones relacionadas con el personal de una empresa, independientemente del número total de

empleados presentes en ella. Los procesos realizables con esta herramienta están relacionados con los Recursos Humanos (GestPeople, 2014).

Sistemas similares a nivel nacional

Sistema de Control de la Militancia UJC (SICOM-UJC): fue desarrollo con el objetivo automatizar los procesos de gestión llevados a cabo dentro de la organización. Cuenta con los procesos de Gestión del ID2, la Gestión de estructuras y además gestiona parte del proceso de cotización o sea no soporta el ciclo completo del mismo. No realiza el proceso de ingreso sino que solo inserta militantes después de haberle realizado el mismo de forma manual. Su estudio sirvió para darle coherencia al flujo de trabajo de los procesos de ingreso a las filas y cotización (Mesa, et al., 2016).

Sistemas similares en la UCI

Sistema Integral para la Gestión de la Información de los Procesos Sustantivos de la UJC en la UCI. SIGPS_UCI: fue creado en la Facultad 4 para facilitar el desarrollo de algunos de los procesos de la organización, entre los que se encuentra la gestión de los cierres de funcionamientos (resumen estadístico de las reuniones de CB) y la gestión del ID2 (incorporaciones, traslados y bajas de militantes). La información que se gestiona en este sistema sirvió para conocer acerca de los principales aspectos relacionados con los militantes en la organización, cómo se realizan las reuniones y que salida se produce a partir de la realización de la misma (Martínez, et al., 2015).

Sistema Integral de Gestión de la UJC UCI Módulo Militante: fue creado en la Facultad 1 para formar parte de un sistema integral de gestión de la UJC en la UCI y permitir introducir mejoras en la gestión de la información de los militantes y el universo juvenil (Fajardo, y otros, 2009). El módulo automatiza los siguientes procesos:

- Crear estructuras organizativas
- Gestión de los datos de cada miembro del universo juvenil
- Realizar crecimiento del universo juvenil
- Dar Alta por Crecimiento
- Dar Alta por Traslado Externo
- Dar Alta por Ajuste

- Dar Alta por Traslado Interno
- Crear Expediente del militante
- Dar Baja por Ajuste
- Dar Baja Natural
- Dar Baja por Defunción

Hoy este sistema no se encuentra implementado en la UCI, del mismo solo está disponible su documentación. El estudio de este sistema apoyó a la investigación para conocer cómo se realizan internamente los movimientos dentro de la organización.

Sistema Informático para la Gestión de la Información de La Unión de Jóvenes Comunistas de la Facultad 2: este sistema permite gestionar los militantes y profesores de la facultad 2, genera reporte de ID2 para los comités de base compuestos por estos militantes, no gestionando la información de los militantes que no son de la facultad, ni trabajadores que pertenezcan al comité primario de la facultad 2 y además tampoco permite la entrada de datos para otros comités de base que no sean miembros de este comité primario (Olivares, et al., 2009).

1.4 Metodología de desarrollo

Una definición estándar de metodología de desarrollo puede ser el conjunto de métodos que se utilizan en una determinada actividad con el fin de formalizarla y optimizarla. Determina los pasos a seguir y como realizarlos para finalizar una tarea. Si esto se aplica a la ingeniería del software, se destaca que una metodología: (Jacobson, et al., 2000)

- Optimiza el proceso y el producto de software.
- Contiene métodos que guían en la planificación y en el desarrollo del software.
- Define qué hacer, cómo y cuándo durante todo el desarrollo y mantenimiento de un proyecto.

Existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Por una parte se tiene aquellas propuestas más tradicionales que se centran especialmente en el control del proceso, en llevar una documentación exhaustiva de todo el proyecto, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y

notaciones que se usarán. Sin embargo, durante el desarrollo se debe centrar también en otras dimensiones, como por ejemplo el factor humano o el producto software. Esta es la filosofía de las metodologías ágiles, las cuales ponen vital importancia en la capacidad de respuestas a los cambios, dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas (Letelier, y otros, 2003). El impacto de elegir la mejor metodología para un equipo de desarrollo, es trascendental para el éxito del producto en una empresa, pues los beneficios serían cuantiosos.

Teniendo en cuenta que el plazo para el desarrollo del producto es corto, el equipo de trabajo es pequeño y en el actual ambiente de negocio, las peticiones del cliente varían fácilmente, el enfoque que prevalece es el ágil debido a que se hace necesario minimizar el tiempo de entrega del producto. Para la elección de la metodología con enfoque ágil, fueron analizadas varias metodologías utilizadas en el proceso de desarrollo de software a nivel mundial.

Programación Extrema (Extreme Programming, XP): es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico (Letelier, y otros, 2003).

SCRUM: Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas *sprints*, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración (Letelier, y otros, 2003).

El Proceso Unificado Ágil (Agile Unified Process, AUP): Es una versión simplificada de *Rational Unified Process* RUP (Proceso Unificado Racional). Este define un flujo de trabajo con disciplinas diferentes a las de RUP, aunque conserva sus fases en cada una. La disciplina de modelación incluye la

modelación del negocio, requisitos, análisis y diseño. Por otra parte, se integran además la Gestión de Cambios y Gestión de Configuración en una sola disciplina (Ambler).

La metodología está compuesta por cuatro disciplinas que sirven de apoyo a la ingeniería (Modelación, Implementación, Prueba y Despliegue) y cuenta con tres destinadas al soporte (Gestión de configuración, Gestión de Proyecto y Ambiente). Estas son ejecutadas y guiadas de forma iterativa.

En la UCI, a pesar de la variedad de metodologías existentes, se ha comprobado que muy pocos proyectos la aplican en su totalidad, por lo que uno de los principales problemas detectados es que sin importar la metodología que se usa se está planificando con un único cronograma tipo. Para erradicar estos problemas la Universidad decidió hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI (Informáticas, 2015).

Una vez realizada la investigación de las metodologías, el equipo selecciona AUP en su variante UCI, debido a que posee familiarización con los artefactos manejados en las diferentes fases de la misma, ya que son los mismos que genera RUP, solo que establece un modelo más simple. Esta variante, de las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición), mantiene la fase de Inicio, unifica las tres restantes en una sola, llamada Ejecución y se agrega la fase de Cierre. En cuanto a las disciplinas, los flujos de trabajos: Modelado de negocio, Requisitos, Análisis y Diseño se consideran a cada uno de ellos métodos por separado. Se mantiene la disciplina Implementación, en el caso de Prueba se desagrega en 3 disciplinas: Pruebas Internas, de Liberación y Aceptación. Entre las ventajas que brinda está la colaboración con el cliente y el desarrollo incremental del software con iteraciones muy cortas, tiene bien definido los roles y fases, así como incrementa la productividad y facilita el trabajo en proyectos de pequeña envergadura. Se transitará a través del escenario dos que plantea: proyectos que modelen el negocio con Modelo Conceptual solo pueden modelar el sistema con Casos de uso del sistema.

- ✓ Debido a que los módulo a implementar son un sistema heredado, los lenguajes y herramientas a utilizar serán las mismas tratadas en el SIGPS_UCI, solo cambian versiones de los mismos debido a la compatibilidad con algunos de los servicios utilizados en la propuesta.

1.5 Lenguaje de Modelado

El lenguaje de modelado consiste en un conjunto de vistas, diagramas, símbolos y reglas que indican como emplear los elementos para la representación de la estructura de los sistemas informáticos (Larman, 2003).

Lenguaje Unificado de Modelado (UML) v2.0

Se decide utilizar el Lenguaje Unificado de Modelado (UML) debido a que es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Está compuesto por diferentes elementos gráficos que se combinan para conformar diagramas. UML es un lenguaje para construir modelos orientados a objetos; no guía al desarrollador en la forma de realizar el análisis y diseño orientados a objetos ni le indica cual proceso de desarrollo adoptar (Larman, 2003).
UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real (Jacobson, y otros, 2000).

1.6 Tecnologías y lenguajes del lado del cliente

HTML5

El lenguaje HTML (*hypertext markup language)* se utiliza para crear documentos que muestren una estructura de hipertexto. Un documento de hipertexto es aquel que contiene información cruzada con otros documentos, lo cual permite pasar de un documento al referenciado desde la misma aplicación con la que lo estamos visualizando. HTML permite, además, crear documentos de tipo multimedia, es decir, que contengan información más allá de la simplemente textual, como por ejemplo: imágenes, videos, sonidos entre otros. Los documentos HTML se conforman como documentos de texto plano, en los que todo el formato del texto se especifica mediante marcas de texto (llamados etiquetas), que delimitan los contenidos a los que afecta la etiqueta (Mateu, 2004).

CSS3

CSS (*Cascading Style Sheets*) es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Es utilizado para definir el

aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página (Eguíluz, 2008).
Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento, funciona a base de reglas, es decir, declaraciones sobre el estilo de uno o más elementos (BluePrintCSS, 2009).

JavaScript

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. 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 (Eguíluz, 2009).

1.7 Lenguajes del lado del servidor

PHP v5.4

PHP es un lenguaje sencillo, de sintaxis cómoda y similar a la de otros lenguajes. Es rápido, interpretado, orientado a objetos y multiplataforma. Para él se encuentra disponible una multitud de librerías. El intérprete de PHP, los diversos módulos y gran cantidad de librerías desarrolladas para PHP son de código libre, con lo que el programador de PHP dispone de una impresionante cantidad de herramientas libres para desarrollar aplicaciones. Posee una alta capacidad de conexión con la mayoría de los Sistemas Gestores de base de datos (SGBD) como MySQL, PostgreSQL entre otros (Mateu, 2004).

Twig

Twig es un motor y lenguaje de plantillas para PHP muy rápido y eficiente. Symfony2 recomienda utilizar *Twig* para crear todas las plantillas de la aplicación. La sintaxis de *Twig* se ha diseñado para que las plantillas sean concisas y muy fáciles de leer y de escribir (Eguiluz, 2011).

1.8 Framework del lado del cliente

Framework

Un *framework* o marco de trabajo es un producto que sirve como base para la programación avanzada de aplicaciones, que aporta una serie de funciones o códigos para realizar tareas habituales. Son librerías de código que contienen procesos o rutinas ya listos para usar. Los programadores utilizan los *frameworks* para no tener que desarrollar ellos mismos las tareas más básicas, puesto que en el propio *framework* hay implementaciones que están probadas, funcionan y no se necesitan volver a programar (Alvarez, 2009).

JQuery v1.9.1

JQuery es un *framework* Javascript, implementa una serie de clases (Programación Orientada a Objeto) que permite programar sin preocuparse por el navegador con que está visitando el usuario. Ofrece una infraestructura con la que se tiene mucha mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Es un producto serio, estable, bien documentado y con un gran equipo de desarrolladores a cargo de la mejora y actualización del *framework* (Alvarez, 2009).

Bootstrap 2

Bootstrap es un *framework* que simplifica el proceso de creación de diseños web combinando CSS y JavaScript. Permite un ahorro significativo de esfuerzo y tiempo debido a que facilita la creación de interfaces que se adapten a cualquier navegador, tanto de escritorio como tabletas y móviles a distintas escalas y resoluciones. Ofrece una serie de plantillas CSS y ficheros Javascript que permiten integrar el *framework* de forma sencilla y potente en nuestros proyectos webs. Además se integra perfectamente con las principales librerías Javascript, por ejemplo JQuery (Gen, 2015).

1.9 Framework del lado del servidor

Symfony v2.5

Symfony es un completo *framework* diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Symfony es compatible con SGBD, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. Fue diseñado para que se ajustara a los siguientes requisitos: fácil de instalar y configurar en la mayoría de plataformas,

independiente del SGBD, sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos, sigue la mayoría de mejores prácticas y patrones de diseño para la web, código fácil de leer que incluye comentarios de *phpDocumentor* y fácil de entender, lo que permite su integración con librerías desarrolladas por terceros (Potencier, y otros, 2007).

1.10 Capa de acceso a datos

Doctrine2

Doctrine es un Mapeador Objeto Relacional (ORM) para PHP que proporciona persistencia transparente de objetos PHP. Se sitúa en la parte superior de una poderosa capa de abstracción de base de datos. La principal tarea de los mapeadores objeto relacionales es la traducción transparente entre objetos PHP y las filas relacionales de la base de datos. Una de las características clave de Doctrine es la opción de escribir las consultas de base de datos en un dialecto SQL propio orientado a objetos llamado Lenguaje de Consulta Doctrine (DQL por *Doctrine Query Language*).

Además DQL difiere ligeramente de SQL en que abstrae considerablemente la asignación entre las filas de la base de datos y objetos, permitiendo a los desarrolladores escribir consultas de una manera sencilla y flexible (Team).

1.11 Sistema gestor de base de datos

PostgreSQL v9.4

PostgreSQL es uno de los SGBD más antiguos y conocidos del mundo del código libre. Las características más destacadas de PostgreSQL son (Mateu, 2004):

- Soporte para transacciones.
- Subconsultas.
- Soporte de vistas.
- Integridad referencial.
- Herencia de tablas.
- Tipos definidos por el usuario.

-

1.12 Servidor Apache

Apache v2.4.7

Apache es una plataforma de servidores web de código fuente. El grupo Apache creó originalmente una primera versión de un servidor Web altamente configurable, el cual se hizo popular rápidamente; en la versión 2, el grupo Apache se ha concentrado en la escalabilidad, en la seguridad y en el rendimiento. Entre las principales características que presenta Apache se puede mencionar que: funciona en plataformas virtuales muy utilizadas, es un servidor altamente configurable de diseño modular, es una tecnología gratuita de código fuente abierto, además de PHP y Perl trabaja con gran cantidad de otros lenguajes script. No solo funciona en la mayoría (prácticamente en todas) las versiones de Unix sino que, además, funciona en Windows y en muchos otros sistemas operativos de escritorio y de tipo servidor como son Amiga OS 3.x y OSl2 (Kabir, 2012).

1.13 Entorno de desarrollo

Netbeans v8.0

Netbeans es un programa compuesto por un conjunto de herramientas de programación que permite elaborar aplicaciones de escritorio Java, móvil, y aplicaciones web, así como las aplicaciones de HTML5 con HTML, JavaScript, y CSS. También provee un gran juego de herramientas para desarrolladores de PHP y C/C++. Ofrece funciones de entorno de desarrollo integrado avanzado como diseño de interfaces, asistente para la conexión con base de datos, creación automática de propiedades y clases. Es gratuito y de código abierto y tiene una gran comunidad de usuarios y desarrolladores de todo el mundo (Net, 2011).

1.14 Herramienta de modelado

Visual Paradigm v8.0

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Ayuda a una rápida construcción de aplicaciones con calidad y a un menor costo. Permite crear todos los tipos de diagramas de clases, ingeniería inversa, generar código desde diagramas y generar documentación (Vis, 2015).

1.15 Conclusiones Parciales

El análisis de los principales conceptos asociados al objeto y campo de la investigación permitió obtener un conjunto de elementos fundamentales para desarrollar una propuesta de solución dirigida a la informatización de los procesos ingreso a las filas y cotización en la UJC.

Se decide desarrollar los módulos utilizando como lenguaje de programación PHP v5.4 y JavaScript, integrando los *framework JQuery* v1.9.1, *Bootstrap* v2.0 y Symfony v2.5 sobre el entorno de desarrollo *Netbeans* v8.0. Además, se utilizará como servidor web Apache en su versión v2.4.7, como SGBD PostgreSQL v9.4 y *Doctrine* v2.0 como técnica para mapear objetos relacionales. Como herramienta de modelado Visual Paradigm v8.0 utilizando el lenguaje de modelado UML en su versión 2.0. Todo el proceso de desarrollo del software será guiado y controlado por la metodología de desarrollo AUP.

Capítulo 2: Propuesta de solución

2.1 Introducción

La disciplina de modelado de la metodología AUP comprende las fases de modelado de negocio, requisitos, análisis y diseño. En el presente capítulo los artefactos que se presentarán son los siguientes: el modelo del dominio, la especificación de los requisitos funcionales y no funcionales que debe cumplir el sistema, el modelo de caso de uso en el cual se realizan, los diagramas y descripciones de casos de uso. Además se elaborará el modelo de diseño de la solución propuesta, diagramas de secuencia y diagrama de despliegue. No se generarán los diagramas de clase debido a que el equipo de trabajo y el cliente poseen una total claridad de los requisitos.

2.2 Modelo de dominio

Un modelo de dominio o modelo conceptual es una representación de las clases conceptuales del mundo real, no de componentes de software. Utilizando la notación UML, un modelo del dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. Permite representar conceptos del dominio de un problema determinado (Larman, 2003).

Conceptos del dominio

Para un mayor entendimiento del problema se definen los términos más importantes asociados al dominio para la posterior confección del modelo.

- **Comité UJC UCI:** es el encargado de dirigir los procesos que realiza la UJC dentro de la UCI.
- **Comité Primario:** asegura metodológicamente el cumplimiento de los procedimientos establecidos para el desarrollo de los procesos de la UJC en las organizaciones de base que se le subordinan.
- **Comité de Base**: estructura más baja y principal de la organización, donde se desarrollan todos los procesos de la UJC.
- **Militante UJC:** persona que integra la organización, la cual participa en los procesos que desarrolla la UJC.
- **Miembro del universo juvenil:** son las jóvenes que no son militantes, y se encuentran en el radio de acción de la estructura de base.

- **Asamblea de Jóvenes Ejemplares:** es el espacio donde se eligen, por el voto favorable de la mayoría de los trabajadores presentes, los jóvenes que reúnen los requisitos para iniciarles el proceso de crecimiento a las filas de la UJC. Constituye la principal vía de ingreso a la UJC. Será convocada por el CB.
- **Asamblea de consulta con las masas:** es el espacio donde se somete a consideración de los trabajadores, estudiantes o combatientes, a los jóvenes que entregaron por escrito a la dirección del CB la solicitud personal. Una vez recogidos los criterios el CB decidirá a quién se le iniciará el proceso de ingreso.
- **Asamblea de inicio del proceso:** el CB deberá analizar los resultados de la Asamblea de jóvenes ejemplares o consulta con las masas y se aprobará el inicio o no del proceso de ingreso, el cronograma y el dúo de crecimiento.
- **Cronograma:** se confecciona para el desarrollo del proceso, que no deberá exceder de 90 días desde su inicio hasta a entrega del carné.
- **Dúo de crecimiento**: es el encargado de realizarle el proceso a cada joven previsto para el ingreso a las filas.
- **Entrevista individual**: instrumento que permite profundizar en la trayectoria del joven en el momento de iniciar el proceso.
- **Comprobación**: está dirigida a la búsqueda de elementos importantes sobre la actitud de procesado.
- **Autobiografía**: documento elaborada por el joven que recoge los elementos más importantes de su trayectoria como estudiante.
- **Solicitud del DNI**: datos a verificar en archivo en busca de antecedentes penales.
- **Estado de la militancia:** resumen estadístico de todos los movimientos realizados en las estructuras de base.
- **Incorporación:** proceso que se realiza en las estructuras de base cuando se integra un miembro a ella.
- **Traslado:** proceso que se realiza en las estructuras de base cuando se retira un miembro de ella hacia otra estructura.
- **Baja:** proceso que se realiza en las estructuras de base cuando se retira un miembro de ella.
- **Sanción:** medidas disciplinarias que se aplican a militantes que incumplen con sus deberes.

- **Asamblea de aceptación**: se acepta o no por parte del organismo superior el ingreso de los jóvenes.
- **Reporte de acta:** almacena toda la información de los puntos analizados en las reuniones desarrolladas en las estructuras.
- **Opiniones:** criterios emitidos por los participantes en una reunión.
- **Acuerdo:** tareas que se plantean en una reunión y se asignan a miembros de una estructura de base.
- **Cotización:** Aporte económico individual que realiza cada militante para sufragar los gastos de la organización.

Diagrama del modelo del dominio

Muestra las relaciones entre todas las entidades comprendidas en el ámbito del dominio del problema.

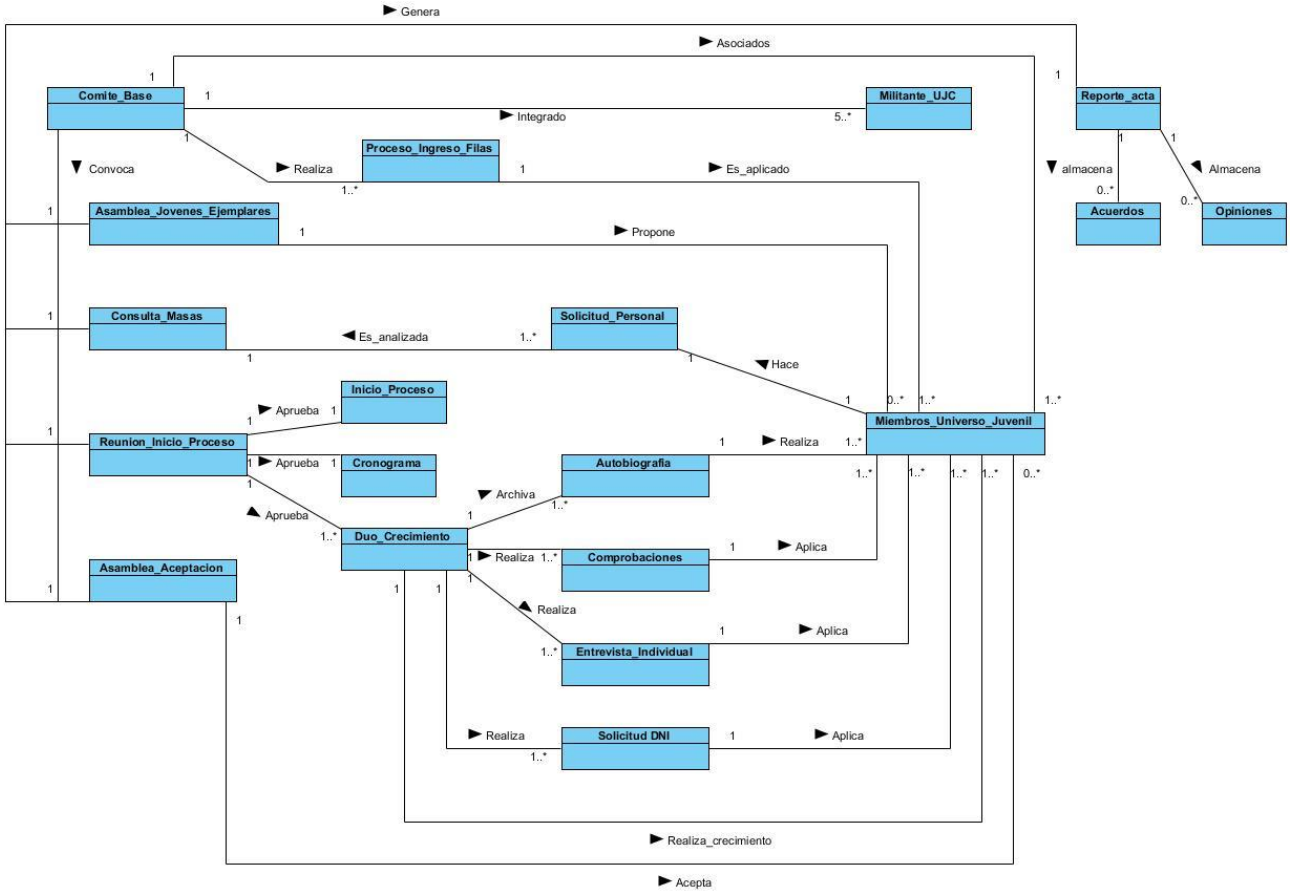


Figura1. Modelo del dominio. Proceso de Crecimiento (Elaboración propia).

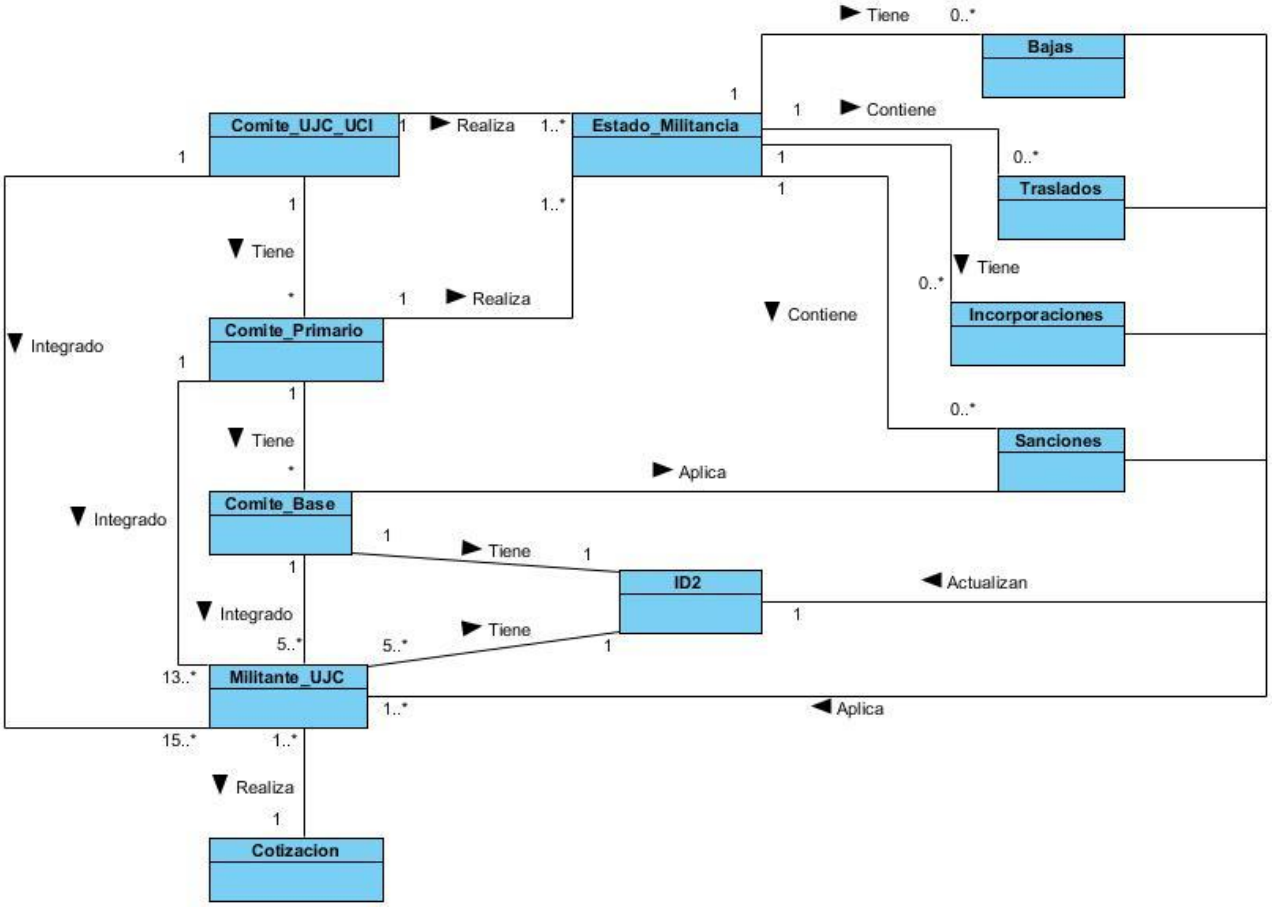


Figura2. Modelo del dominio. Proceso Cotización (Elaboración propia).

2.3 Requisitos del software

Los requisitos para un sistema son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema como el control de un dispositivo, hacer un pedido o encontrar información (Sommerville, 2005).

Requisitos funcionales

Los requerimientos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones

particulares (Sommerville, 2005). Los requisitos funcionales que debe cumplir la aplicación a desarrollar son los que a continuación se relacionan:

RF1 Autenticar usuario.
RF2 Asignar responsabilidades
RF3 Gestionar solicitud ingreso.

- **RF3.1** Crear solicitud de ingreso
- **RF3.2** Visualizar solicitud de ingreso.
- **RF3.3** Modificar solicitud de ingreso.
- **RF3.4** Mostrar requisitos a cumplir para el ingreso a las filas de la UJC.

RF4 Registrar solicitud de ingreso:

- **RF4.1** Visualizar solicitud de ingreso.
- **RF4.2** Aceptar o denegar solicitud de ingreso.
- **RF4.3** Eliminar solicitud de ingreso.

RF5 Modificar acta de asamblea ordinaria.

- **RF5.1** Mostrar invitados del universo juvenil.
- **RF5.2** Cargar acuerdos de asambleas anteriores.

RF6 Gestionar acta de Asamblea Jóvenes Ejemplares:

- **RF6.1** Crear acta.
- **RF6.2** Visualizar acta.

RF7 Gestionar acta de Consulta con las Masas:

- **RF7.1** Crear acta.
- **RF7.2** Visualizar acta.

RF8 Gestionar acta de Inicio del Proceso:

- **RF8.1** Crear acta.

- **RF8.2** Visualizar acta.

RF9 Gestionar acta de Asamblea de Aceptación:

- **RF9.1** Crear acta.
- **RF9.2** Visualizar acta.

RF10 Gestionar autobiografía:

- **RF10.1** Crear autobiografía.
- **RF10.2** Modificar autobiografía.
- **RF10.3** Visualizar autobiografía.

RF11 Gestionar entrevista individual:

- **RF11.1** Crear entrevista individual.
- **RF11.2** Modificar entrevista individual.
- **RF11.3** Visualizar entrevista individual.

RF12 Gestionar solicitud del DNI:

- **RF12.1** Crear solicitud del DNI.
- **RF12.2** Modificar solicitud del DNI.
- **RF12.3** Visualizar solicitud del DNI.

RF13 Gestionar comprobaciones:

- **RF13.1** Crear comprobaciones.
- **RF13.2** Modificar comprobaciones.
- **RF13.3** Visualizar comprobaciones.

RF14 Gestionar reclamación sobre el proceso

- **RF14.1** Crear reclamación.
- **RF14.2** Modificar reclamación.

- **RF14.3** Visualizar reclamación.

RF15 Registrar reclamación sobre el proceso de ingreso

- **RF15.1** Visualizar reclamación.
- **RF15.2** Aceptar o denegar reclamación.
- **RF15.3** Eliminar reclamación.

RF16 Actualizar planilla ID2

- **RF16.1** Actualizar incorporaciones de militantes
- **RF16.2** Actualizar sanciones por no pago de cotización

RF17 Gestionar cotización

- **RF17.1** Crear cotización.
- **RF17.2** Mostrar importe de CB.
- **RF17.3** Generar listado de importe de cotización por CB.
- **RF17.4** Sancionar militante por no pago de la cotización
- **RF17.5** Visualizar cotización.

RF18 Generar resumen estadístico del pago de la cotización.

- **RF18.1** Mostrar cantidad de cuotas recuperadas
- **RF18.2** Mostrar militantes que arriban a los 32 años de edad.
- **RF18.3** Mostrar militantes con más de 2 meses sin cotizar.
- **RF18.4** Mostrar militantes que no cotizan.
- **RF18.5** Graficar importe del pago de la cotización por CB mensual, trimestral y anual.

RF19 Exportar cierre de funcionamiento a Excel.

RF20 Exportar a .pdf.

- **RF20.1** Solicitudes de ingreso.

- **RF20.2** Autobiografías.
- **RF20.3** Actas de asambleas.
- **RF20.4** Modelo de entrevista individual.
- **RF20.5** Modelo de solicitud del DNI.
- **RF20.6** Modelo de comprobaciones.
- **RF20.7** Resumen de estado de crecimientos.
- **RF20.8** Reclamaciones sobre el proceso.
- **RF20.9** Resumen de estado de la cotización.

Requisitos no funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema (Sommerville, 2005). Los requisitos no funcionales identificados son:

- **Usabilidad**

RNF 1: el sistema debe poseer una interfaz fácil de utilizar para cualquier tipo de usuario con conocimientos básicos de computación en el manejo de ordenadores.

- **Portabilidad**

RNF 2: el usuario debe acceder a la aplicación desde cualquier sistema operativo.

- **Software**

RNF 5: el sistema puede compilar sobre los navegadores Mozilla Firefox 33.0 o superior y Google Chrome 28.0 o superior para poder acceder a las opciones que brinda el sistema.

- **Hardware**

RNF 6: para un buen funcionamiento, la computadora donde se instale el sistema debe contar con las siguientes características:

- ✓ Procesador Intel Celeron o superior.
- ✓ Memoria RAM de 512MG o superior.

RNF 7: Para un buen funcionamiento, el servidor donde se ejecute el sistema debe contar con las siguientes características:

- ✓ Procesador Intel Dual Core 2.8 GHz o superior.
- ✓ Memoria RAM de 1GB o superior.

- **Seguridad**

RNF 8: se podrá acceder a las funcionalidades del sistema solamente después de autenticarse.
RNF 9: se garantizará el acceso de los usuarios sólo a los niveles establecidos de acuerdo a la función que realizan.

2.4 Modelo de casos de uso del sistema

El modelo de casos de uso ayuda al cliente, a los usuarios y desarrolladores a llegar a un acuerdo sobre cómo utilizar el sistema. La mayoría de los sistemas tienen muchos tipos de usuarios. Cada tipo de usuario se representa mediante un actor. Los actores utilizan el sistema al interactuar con los casos de uso. Todos los actores y casos de uso del sistema forman un modelo de casos de uso (Jacobson, y otros, 2000).

Descripción de los actores del sistema

Tabla 1. Actores del sistema y su descripción

Actor	Descripción
Usuario	Actor que podrá acceder a las funcionalidades que brinda el sistema, tales como: autenticar usuario.
Administrador	Actor que tendrá acceso total a todas las funcionalidades del sistema, para realizar operaciones de gestión sobre la información que se maneja en el mismo.
Secretario CB	Es el encargado de visualizar y modificar todos los datos

	de los procesos realizados en su CB.
Activista	Es el encargado de Generar el Reporte de acta de las reuniones del CB.
Cotizador	Es el encargado de llevar el registro de control de las cuotas cotizadas por los militantes de un CB.
Miembro Universo Juvenil	Actor que podrá acceder a las funcionalidades que brinda el sistema tales como Gestionar solicitud de ingreso, Gestionar autobiografía y Gestionar Reclamación sobre el proceso.

Diagrama de casos de uso del sistema

Este diagrama permite mostrar las relaciones entre los actores y casos de uso del sistema.

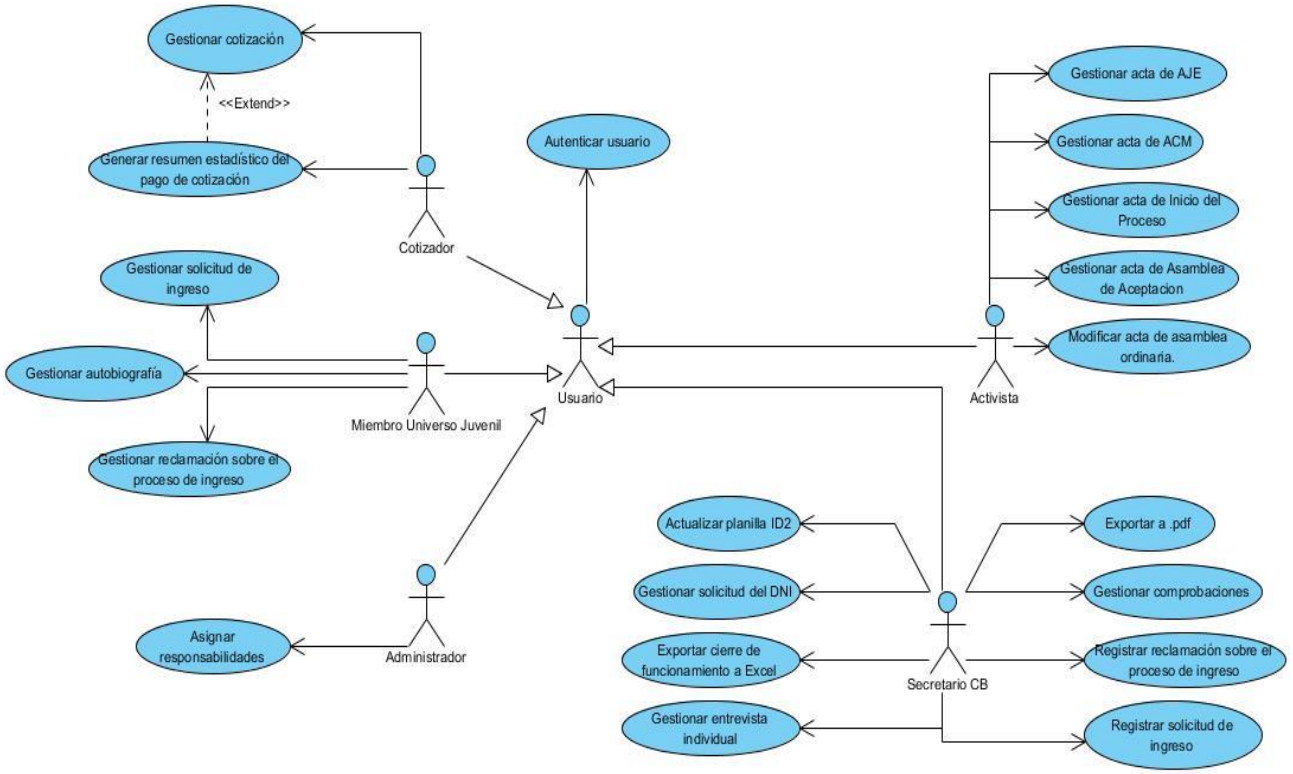


Figura3. Diagrama de casos de uso del sistema (Elaboración propia).

Patrones de casos de uso

Al modelado por casos de uso se le aplican un grupo de patrones que permiten simplificar el diagrama y mejorar su comprensión, los mismos describen diferentes soluciones para el modelado de problemas específicos, atendiéndolos por separado (Övergaard, et al., 2004).

Patrón CRUD: Es un patrón que permite agrupar las operaciones de crear, leer, actualizar y eliminar piezas de información de un tipo o entidad. El mismo puede ser completo o parcial, dependiendo si se agrupan o no todas estas operaciones en un mismo caso de uso (Övergaard, et al., 2004).

- **CRUD Parcial:** se puede observar su aplicación en el caso de uso ‘‘Gestionar solicitud de ingreso’’ donde este solo incluye las operaciones de crear, modificar y visualizar solicitud para el actor Miembro del Universo Juvenil, separando la opción eliminar solicitud para el actor Secretario CB.

Patrón múltiples actores

- **Rol común:** Permite que dos actores desempeñen el mismo rol, para que a través de este puedan acceder a un determinado caso de uso. (Övergaard, et al., 2004). Se puede apreciar este patrón al crearse el rol Usuario para acceder a un grupo de casos de uso, que pueden ser accedidos por el actor Administrador, mientras que este último puede iniciar otros escenarios que no pueden ser accedidos por el rol Usuario.

Relación de extensión (extend): la idea es crear un Caso de uso (CU) que extiende o añade, y con él, describe donde y bajo qué condiciones extiende el comportamiento de algún CU base (Larman, 2003).Esta relación se puede observar entre los CU ‘‘Generar resumen estadístico del pago de la cotización’’ y ‘‘Gestionar cotización’’ ya que para generar el resumen de la cotización tiene que existir un reporte que contenga los datos relevantes de la misma.

Descripción de los casos de uso del sistema

A continuación se muestran las descripciones de los CU Gestionar solicitud ingreso y Gestionar cotización. Para el estudio de los demás CU del sistema, remitirse a los Anexos.

Tabla 2. Descripción del caso de uso Gestionar solicitud de ingreso

Caso de uso	Gestionar solicitud de ingreso

Actor	Miembro del universo juvenil	
Resumen	El caso de uso inicia cuando un miembro del universo juvenil decide solicitar el ingreso a las filas. Para ello el actor selecciona la opción ‘‘Solicitar ingreso’’ del menú Ingreso a las filas. El sistema brinda la opción de crear una nueva solicitud. Si así lo desea el actor, el sistema muestra un formulario con los campos correspondientes a una solicitud y además permite ver los requisitos a cumplir para el ingreso haciendo clic en ‘‘Ver requisitos’’. Además permite visualizar los datos de la solicitud. En caso de que desee modificar los datos de la misma, el actor deberá seleccionar la opción ‘‘Editar’’ y realizar las modificaciones, terminando así el caso de uso.	
Precondiciones	El miembro del universo juvenil debe estar autenticado	
Postcondición	Se creó, visualizó o modificó la solicitud	
Referencias	RF3, RF3.1, RF3.2, RF3.3	
Flujo de eventos		
Flujo Básico		
	Actor	**Sistema**
1	El actor accede al Sistema y selecciona la opción “Solicitar ingreso” del menú Ingreso a las filas.	
2		El sistema brinda la opción de crear una nueva solicitud.
3	Selecciona la opción “Nueva”.	
4		Se muestra un formulario con el siguiente campo: • Motivo Y permite: Mostrar los requisitos a cumplir para el ingreso.

		Y la opción: • Enviar • Cancelar
5	Introduce los datos y selecciona la opción “Enviar”.	
6		Valida los datos.
7		Registra los datos de la solicitud.
8		Muestra un mensaje indicando: “Su solicitud ha sido creada”.
9		El caso de uso termina.
Flujos alternos		
***.a El usuario selecciona la opción cancelar**		
	Actor	**Sistema**
		*. a.1 Regresa al actor a la vista anterior.
		*.a.2 El caso de uso termina.
6.a Existen campos vacíos		
	Actor	**Sistema**
7.a.1		Muestra un mensaje de información “Rellene este campo’’.
7.a.2		Muestra un indicador sobre el campo vacío.
7.a.3		Regresa al paso 4 del flujo básico.
Sección 1: Editar datos de la solicitud **Flujo básico**		
	Actor	**Sistema**
1	Selecciona la opción “Solicitar ingreso” del menú Ingreso a las filas.	

2		Lista la solicitud y muestra las siguientes opciones: • Ver. Ver Sección 2: “Ver datos de la solicitud”. • Editar.
3	Selecciona la opción ‘‘Editar’’.	
4		Abre una ventana y muestra los datos de la solicitud permitiendo modificar el valor de la misma: • Motivo Y la opción: • Guardar • Cancelar
5	Modifica los datos y selecciona la opción “Guardar”.	
6		Valida los datos.
7		Actualiza los datos de la solicitud.
8		Muestra el mensaje: “Solicitud modificada correctamente”.
9		El caso de uso termina.
Flujos alternos		
***.a Selecciona la opción Cancelar**		
	Actor	**Sistema**
		*.a.1 Regresa al actor a la vista anterior
		*.a.2 El caso de uso termina.
6.a Existen campos vacíos		
	Actor	**Sistema**
7.a.1		Muestra un mensaje de información “Rellene este campo’’.

7.a.2		Muestra un indicador sobre el campo vacío.
7.a.3		Regresa al paso 4 del flujo básico.
Sección 2 : Ver datos de la solicitud **Flujo básico**		
	Actor	**Sistema**
1	Selecciona la opción “Solicitar ingreso” del menú Ingreso a las filas.	
2		Lista la solicitud y muestra las siguientes opciones: • Ver. • Editar. Ver Sección 1. “Editar datos de la solicitud”.
3	Selecciona la opción ‘‘Ver’’.	
4		Muestra los datos de la solicitud: • Nombre y Apellidos • Motivo Y la opción: • Cerrar
5	Selecciona la opción ‘‘Cerrar’’.	
6		Cierra la vista actual.
7		Retorna al usuario a la vista anterior.
8		Termina el caso de uso.
Prototipo de interfaz		

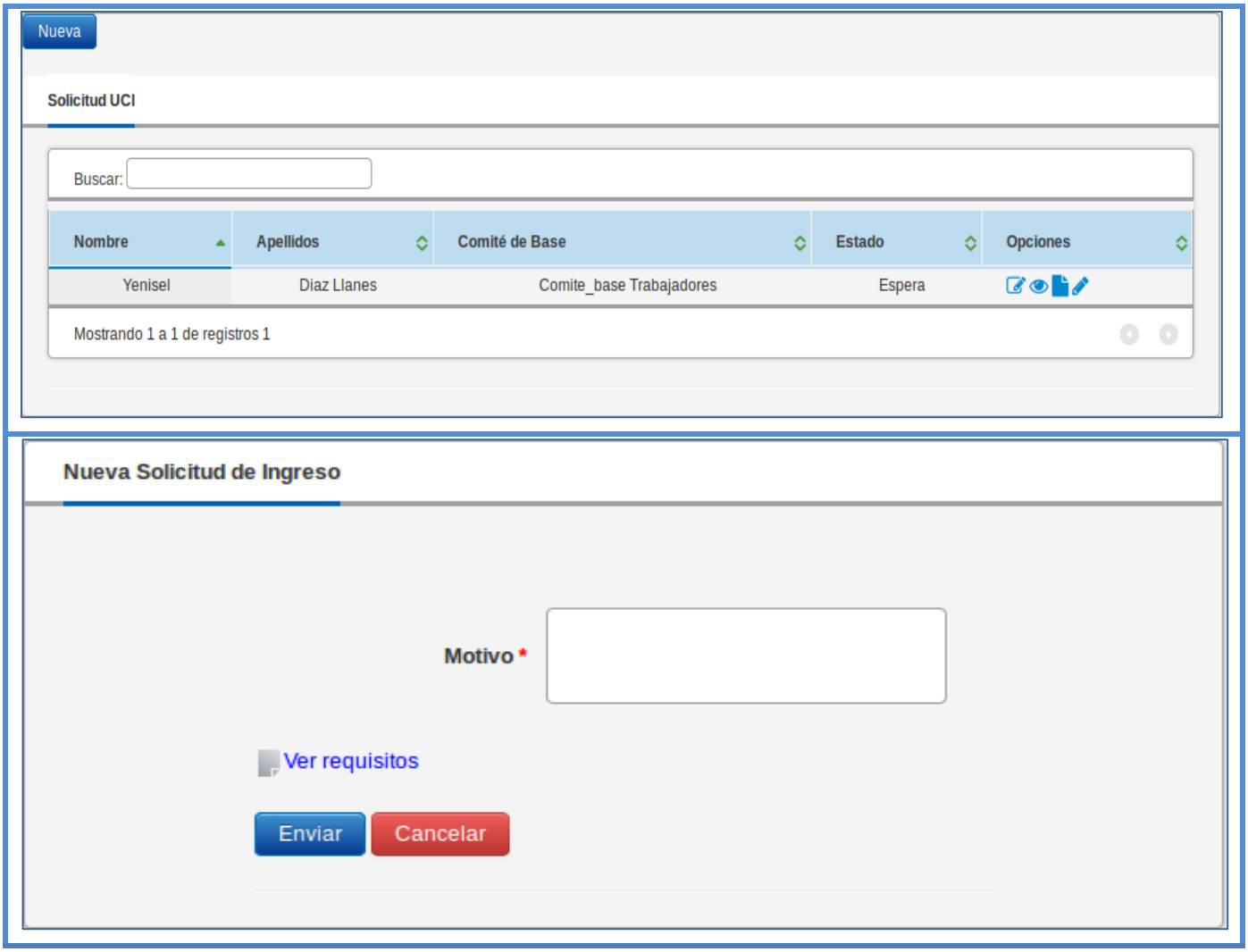


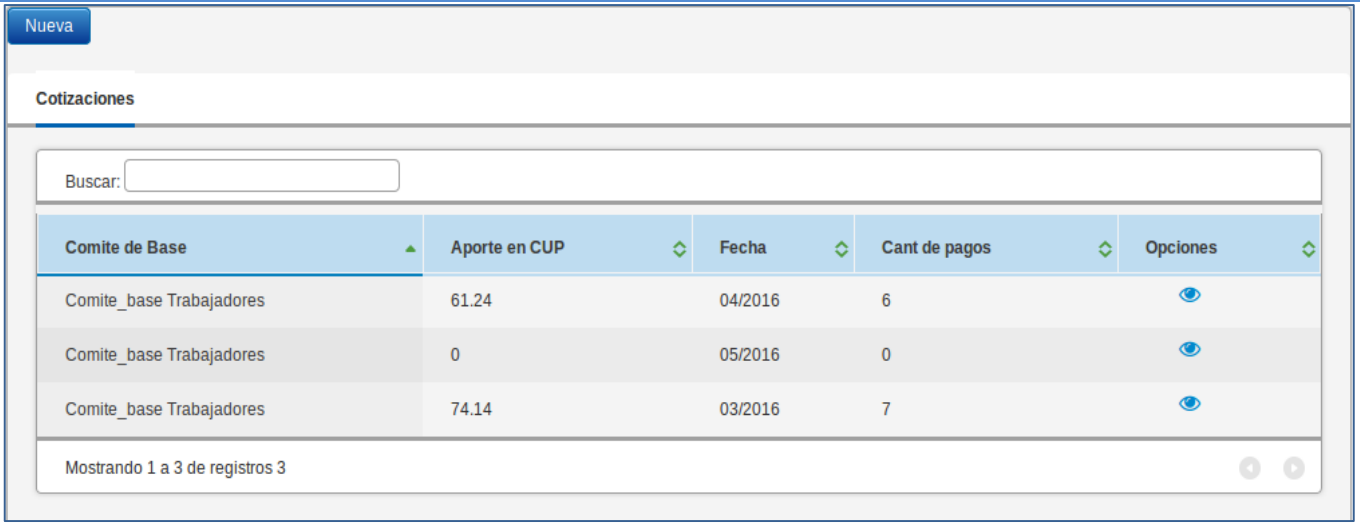
Tabla 3. Descripción textual del caso de uso Gestionar cotización

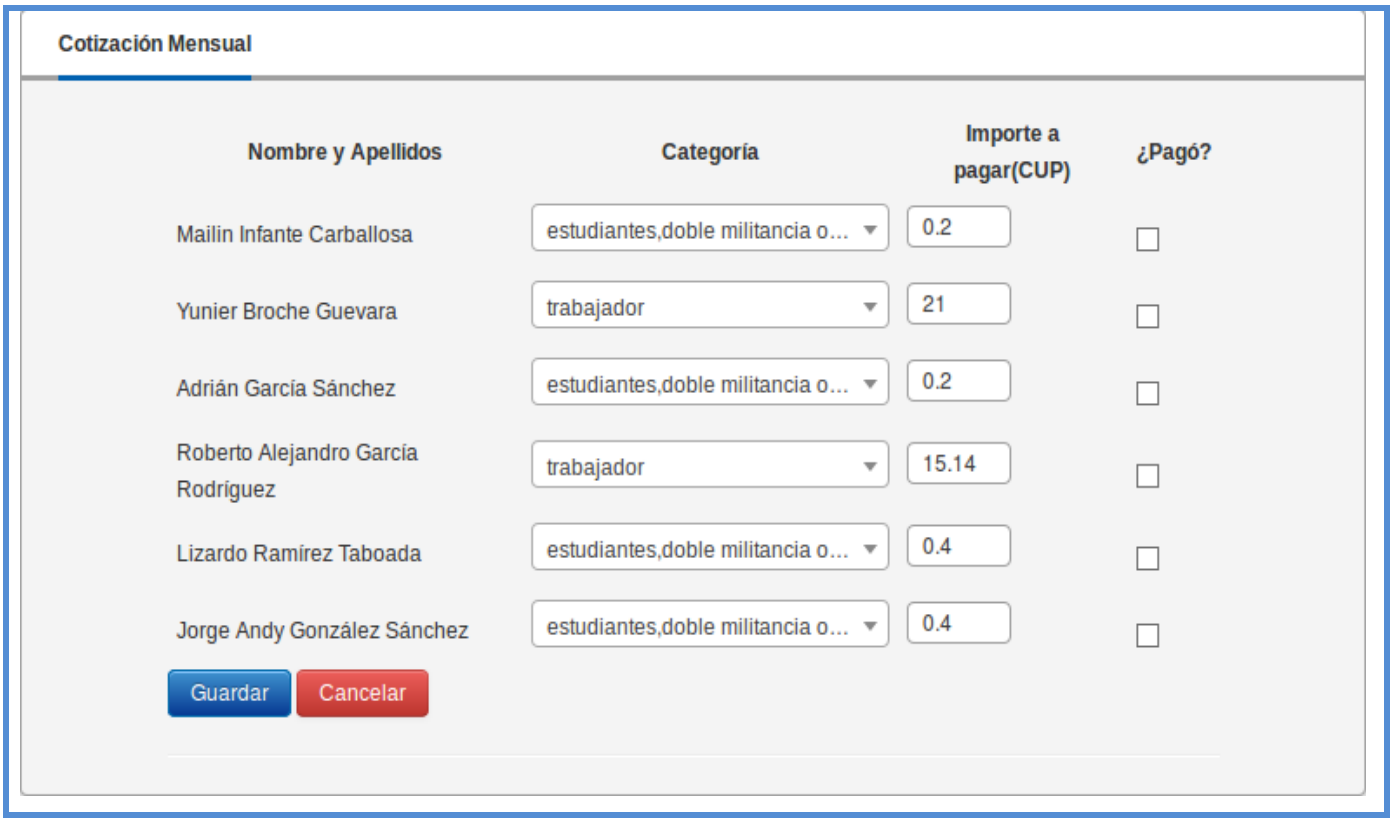
Caso de uso	**Gestionar cotización**
Actor	Cotizador
Resumen	El caso de uso inicia cuando el cotizador decide crear la cotización del mes. Para ello el actor selecciona la opción Cotización en el menú. El sistema permite crear una nueva cotización. Si así lo desea el actor, el sistema registra el pago del militante y si este pagó o no. Muestra los datos referentes a la misma y además propone una sanción a aquellos que hayan incumplido con el pago de la

	cotización permitiendo su modificación. Muestra el importe del comité de base y genera un listado con el importe del CB, terminando así el caso de uso.	
Precondiciones	El cotizador debe estar autenticado y no debe haber creado una cotización en el mismo mes.	
Postcondición	Se creó o visualizó la cotización.	
Referencias	RF17, RF17.1, RF17.2, RF17.3, RF17.4, RF17.5	
Flujo de eventos		
Flujo Básico		
	Actor	**Sistema**
1	El actor accede al Sistema y selecciona la opción ‘‘Cotización’’.	
2		Brinda la opción de crear una nueva cotización
3	Selecciona la opción “Nueva”.	
4		Permite: • Seleccionar la ocupación del militante • Registrar pago del militante • Marcar si pagó o no Y las opciones • Crear • Cancelar
5	Introduce los datos y selecciona la opción “Crear”.	
6		Valida los datos.
7		Registra los datos de la cotización.

8		El caso de uso termina.
Flujos alternos		
***.a El usuario selecciona la opción cancelar**		
	Actor	**Sistema**
		*.a.1 Regresa a la vista anterior
		*.a.2 El caso de uso termina.
Sección 1 : Ver datos de una cotización **Flujo básico**		
	Actor	**Sistema**
1	Selecciona la opción “Cotización”.	
2		Lista las cotizaciones disponibles en el sistema y por cada una brinda la posibilidad de: • Ver.
3	Selecciona la opción ‘‘Ver’’.	
4		Muestra los datos de la cotización: • La ocupación del militante • El pago del militante • Si pagó o no • Los meses sin pagar por cada militante del CB • Sanciones del militante si tiene Y permite: • Proponer sanción al militante por no pago de la cotización (Ver: Tabla 26. Descripción textual del caso de uso Proponer sanción) Y la opción: • Cerrar

5	Selecciona la opción ‘‘Cerrar’’.	
6		Cierra la vista actual.
7		Retorna al usuario a la vista anterior.
8		Termina el caso de uso.
Sección 2 : Mostrar importe de CB **Flujo básico**		
	Actor	**Sistema**
1	Selecciona la opción ‘‘Cotización’’.	
2		Lista las cotizaciones existentes en el sistema generando un listado mostrando el importe por CB.
Prototipo de interfaz		





2.5 Patrón arquitectónico Modelo Vista Controlador (MVC) en Symfony

Symfony está basado en un patrón clásico del diseño web conocido como arquitectura MVC, que está formado por 3 niveles:

- El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- La vista transforma el modelo en una página web que permite al usuario interactuar con ella.
- El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

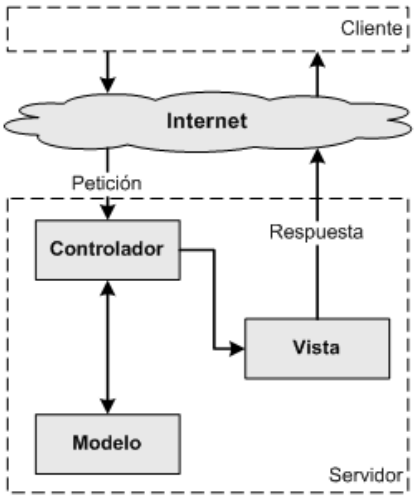


Figura4. Patrón Modelo, Vista, Controlador (MVC) (Potencier, y otros, 2007)**.**

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. Si por ejemplo una misma aplicación debe ejecutarse tanto en un navegador estándar como un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de Gestor de Bases de Datos (GBD) utilizado por la aplicación (Potencier, y otros, 2007).
El uso del patrón arquitectónico en la propuesta de solución permite separar las clases encargadas de la lógica de negocio como SolicitudIngreso, DNI, EntrevistaIndividual de las clases encargadas de controlar las peticiones del usuario como SolicitudIngresoController, DNIController, EntrevistaIndividualController y del código encargado de mostrar los datos como edit.html.twig, show.html.twig, new.html.twig. Además facilita la creación de una estructura de directorios dentro de cada *Bundle* para separar los objetos del modelo, la vista y el controlador.

2.6 Aplicaciones de los patrones de diseño en Symfony

Un patrón es una descripción de un problema y su solución que recibe un nombre y que puede emplearse en otros contextos; en teoría, indica la manera de utilizarlo en circunstancias diversas. Muchos patrones

ofrecen orientación sobre como asignar las responsabilidades a los objetos ante determinada categoría de problemas (Larman, 2003).

Patrones Generales de Software para Asignar Responsabilidades (GRASP):

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. A continuación se explicarán algunos de los patrones utilizados directamente en la solución.

Experto: se basa en asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad (Larman, 2003). La utilización de este patrón se evidencia en la clase SolicitudIngresoController, BiografiaController.

Creador: es el encargado de identificar quién debe ser el responsable de la creación o instanciación de nuevos objetos o clases (Larman, 2003). Un ejemplo de la utilización de este patrón se muestra en las clases ActaInicioProcesoController, BiografiaController, ComprobacionController.

Controlador: sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es el controlador quien recibe los datos del usuario y quien los envía a las distintas clases según el método llamado (Larman, 2003). Su uso se evidencia en las clases SolicitudIngresoController, ComprobacionController.

Alta Cohesión: Se basa en asignar una responsabilidad de modo que la cohesión siga siendo alta, resolviendo el problema de mantener la complejidad en las clase dentro de los límites manejables, permitiendo definir en las clases las funcionalidades relacionadas para que estas no realicen un trabajo enorme (Larman, 2003). La utilización de este patrón se evidencia en la clase ActaInicioProcesoCotroller.

Bajo acoplamiento: se evidencia en la capa modelo. Las clases de acceso a los datos tienen bastante independencia de las clases de abstracción de datos. La poca dependencia entre las clases permite una mayor reutilización (Larman, 2003). Este patrón se evidencia en la clase BiografiaController.

Patrones Banda de cuatro Gang of Four (GoF):

Patrón observador: define de una a muchas dependencias entre objetos de forma que cuando un objeto cambia de estado, todos sus dependientes son notificados y actualizadas de forma automática (Freeman, 2004).

Patrón decorador: responde a la necesidad de añadir dinámicamente funcionalidades a un objeto determinado (Freeman, 2004).

2.7 Modelo de diseño

En el diseño se modela el sistema y se encuentra la forma de que soporte todos los requisitos incluyendo los no funcionales y otras restricciones. Permite descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo (Jacobson, y otros, 2000).

Diagrama de clases del diseño

Los diagramas de clases del diseño se encuentran conectados a la realización de un CU. Son los encargados de mostrar todas las clases participantes, subsistemas y relaciones, así como los atributos y operaciones correspondientes a cada clase. Representan la vista estática del sistema (Jacobson, y otros, 2000).

A continuación se presentan los diagramas de clase del diseño correspondientes a los CU Gestionar solicitud de ingreso y Gestionar cotización. Para el estudio de los demás diagramas de clases del diseño remitirse a los Anexos.

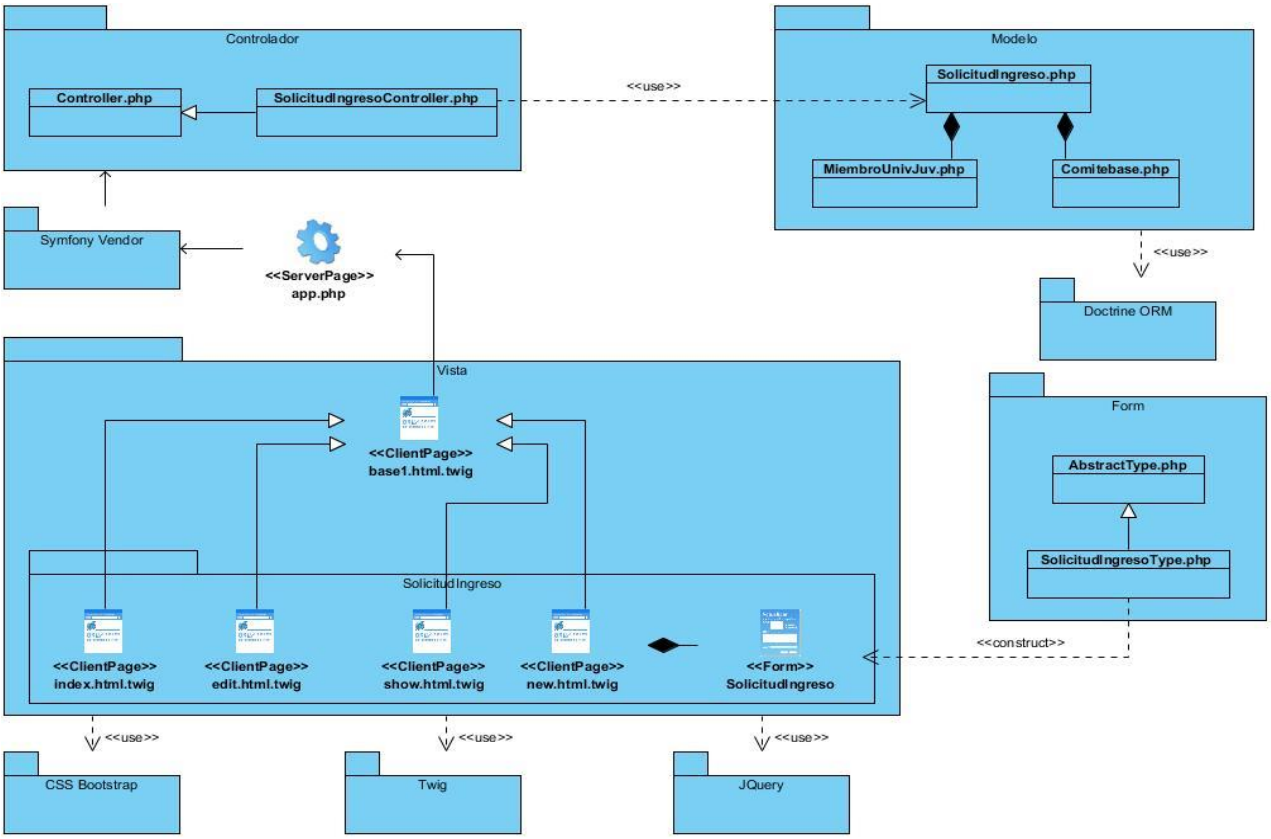


Figura5. Diagrama de clases del diseño del CU Gestionar solicitud ingreso (Elaboración propia).

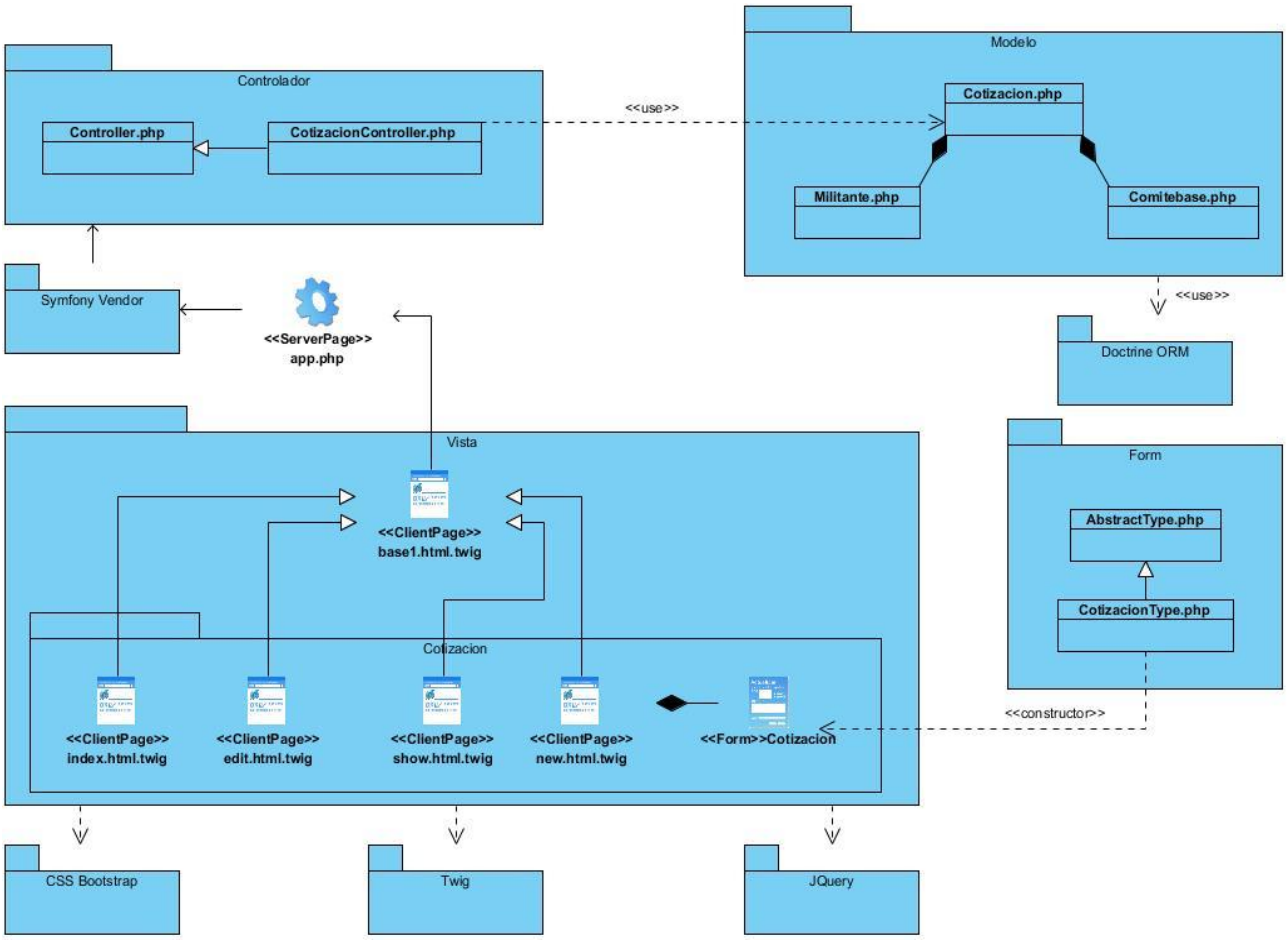


Figura 6. Diagrama de clases del diseño del CU Gestionar cotización (Elaboración propia).

Diagrama de secuencia del diseño

Los diagramas de secuencia muestran las interacciones entre objetos mediante transferencia de mensajes entre objetos o subsistemas. El nombre del mensaje debería indicar una operación del objeto que recibe la invocación o de una interfaz que el objeto proporciona (Jacobson, y otros, 2000).

A continuación se presentan los diagramas de secuencia del diseño referentes al CU Gestionar solicitud de ingreso, el resto de los diagramas se encuentran en los Anexos.

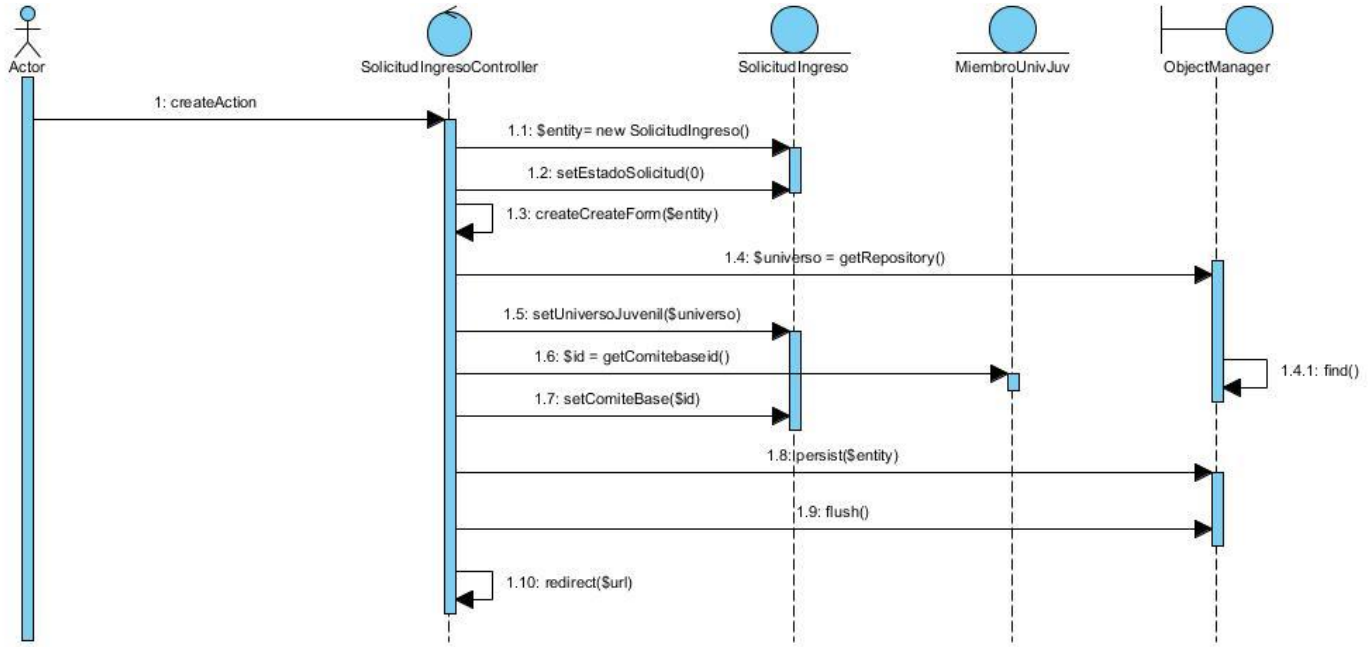


Figura7. Diagrama de secuencia de la sección Crear solicitud del caso de uso Gestionar solicitud de ingreso (Elaboración propia).

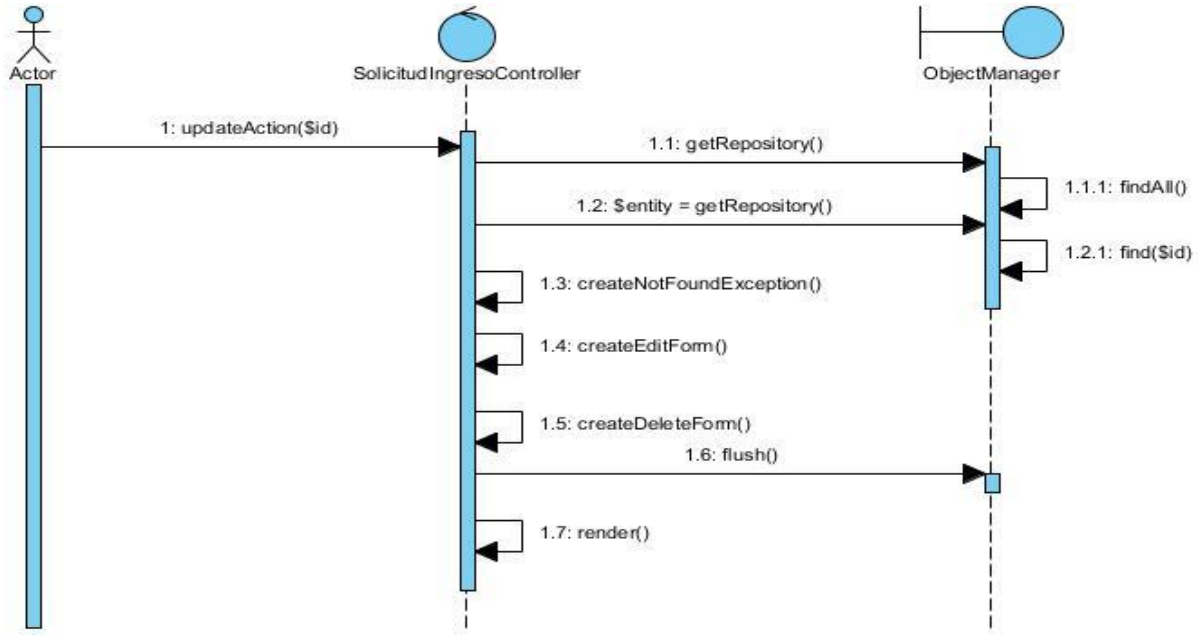


Figura8. Diagrama de secuencia de la sección Modificar solicitud del caso de uso Gestionar solicitud de ingreso (Elaboración propia).

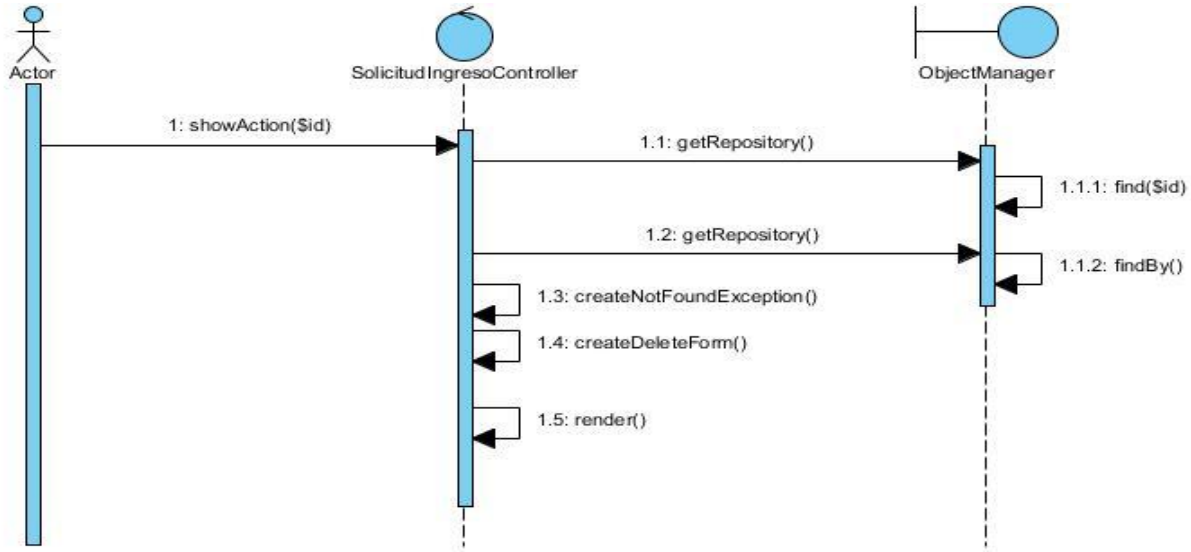


Figura9. Diagrama de secuencia de la sección Visualizar solicitud del caso de uso Gestionar solicitud de ingreso (Elaboración propia).

2.8 Diagrama de despliegue

El diagrama de despliegue es un modelo de objetos que describe la distribución física del sistema. Se utiliza como entrada principal en las actividades de diseño e implementación, debido a que la distribución del sistema tiene una influencia principal en su diseño (Jacobson, y otros, 2000).

A continuación se presenta el diagrama de despliegue del sistema:

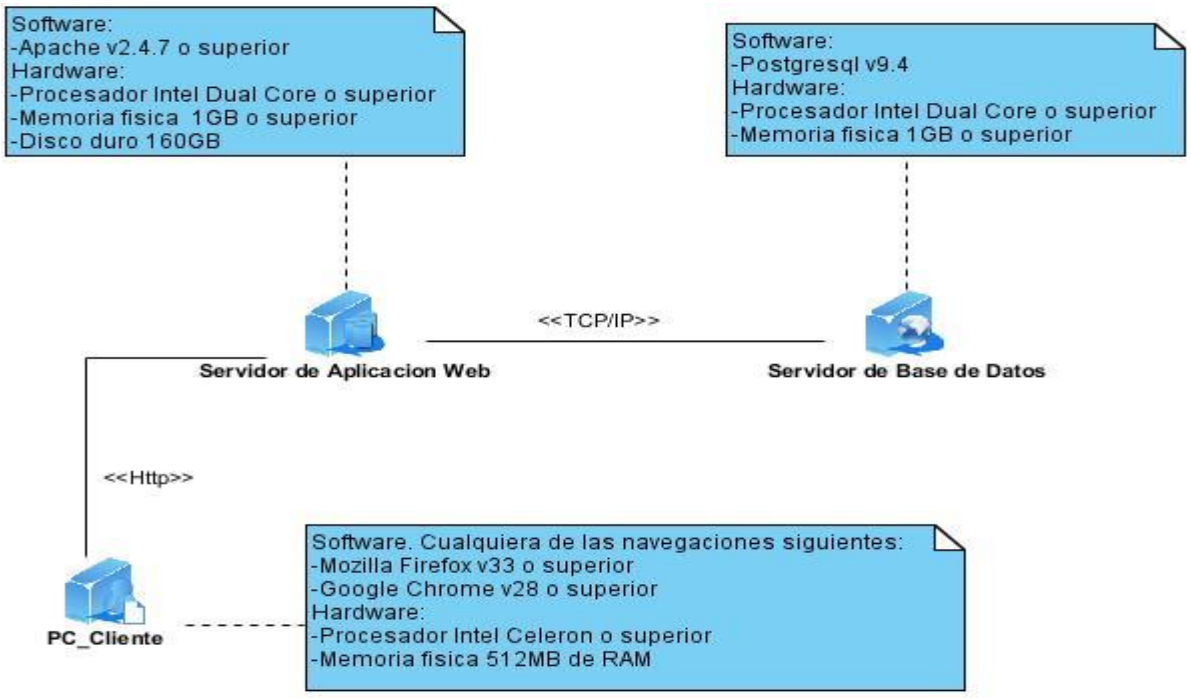


Figura10. Diagrama de despliegue (Elaboración propia).

2.9 Conclusiones parciales

Con el estudio de los principales conceptos asociados al dominio, se modeló su representación para facilitar su comprensión y la forma en que se vincularán las funcionalidades con el entorno. La definición de los requisitos funcionales y no funcionales del sistema establecieron las capacidades que debe cumplir la propuesta de solución.

La realización del modelado del sistema permitió obtener los diagramas de clases del diseño, secuencia y despliegue, los cuales le indicaron al equipo de desarrollo los objetos que se deben construir, los atributos y comportamientos de las clases, su relación y la forma en la que interactúan las mismas desde que el usuario realiza una acción.

Se definió la arquitectura MVC como arquitectura a aplicar por las facilidades que proporciona, así como se utilizaron varios patrones para el diseño que facilitan la reutilización y mejor organización de la solución propuesta.

Capítulo 3: Implementación y prueba

3.1 Introducción

Los artefactos generados en el diseño, constituyen la entrada esencial para el flujo de trabajo de Implementación, el cual propone mostrar cómo se implementan las funcionalidades en términos de componentes y cómo lograr la organización de los mismos. Después de realizar este flujo se elabora el de Pruebas, donde se efectúan las pruebas necesarias a las funcionalidades implementadas para verificar que estén correctamente desarrolladas, permitiendo comprobar su adecuado funcionamiento antes de ser usado por los usuarios finales.

3.2 Modelo de implementación

El modelo de implementación describe cómo los elementos del diseño se implementan en términos de componentes. Describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y los lenguajes de programación utilizados, y cómo dependen los componentes uno de otros (Jacobson, et al., 2000). Es la entrada principal de las etapas de prueba que siguen a la implementación.

Como parte del modelo de implementación se obtiene el diagrama de componentes que a continuación se presenta.

Diagrama de componentes

Los diagramas de componente permiten modelar la vista de implementación del sistema a partir del cual se construye la aplicación. Muestran la organización y las dependencias lógicas entre un conjunto de componentes de software y organiza los subsistemas de implementación en capas. Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño (Jacobson, et al., 2000).

A continuación se muestran los diagramas de componente correspondientes a los CU Gestionar solicitud de ingreso y Gestionar Cotización. Para el estudio de los demás diagramas de componentes remitirse a los anexos.

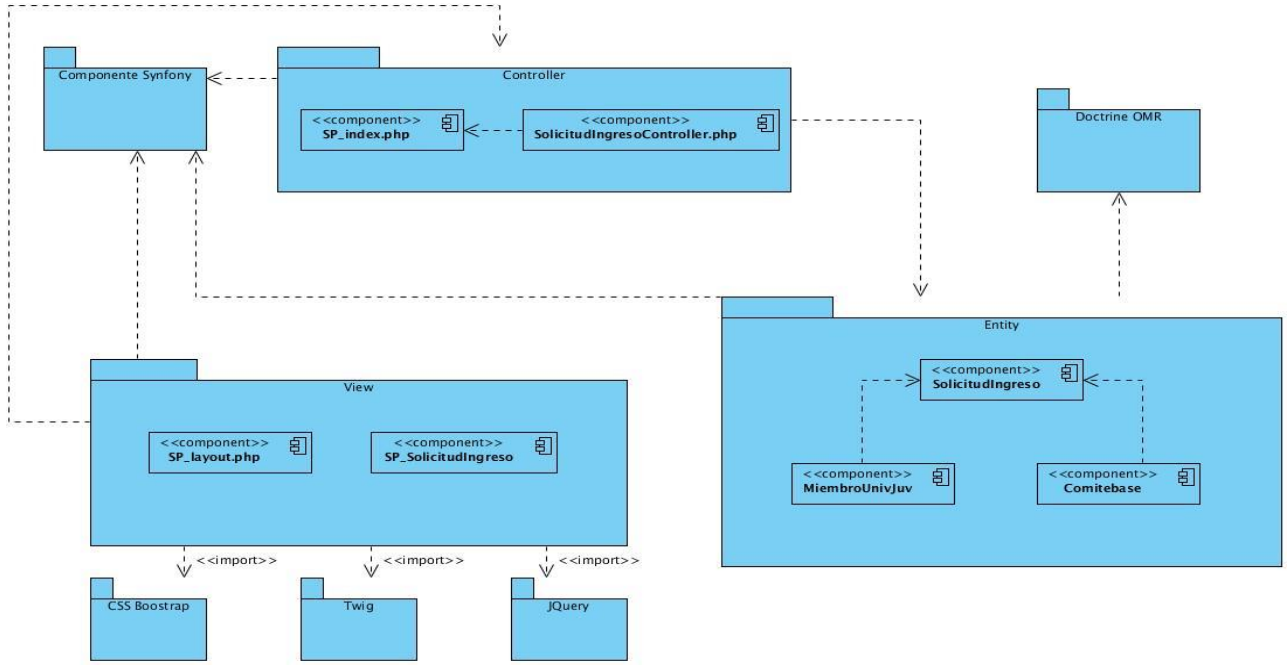


Figura11.Diagrama de componentes: CU Gestionar solicitud ingreso.

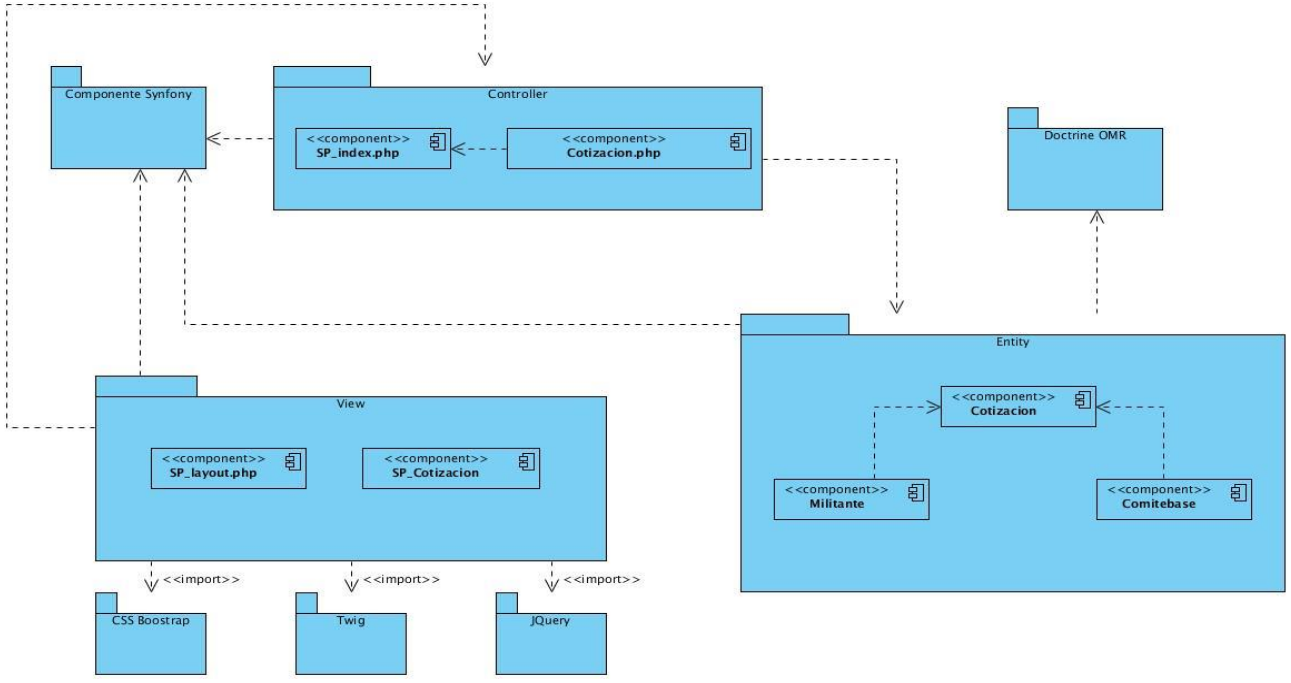


Figura12.Diagrama de componentes: CU Gestionar solicitud ingreso.

3.3 Pruebas de software

Las pruebas del software son una técnica dinámica de validación y verificación (V&V), las cuales implican ejecutar una implementación del software con datos de prueba, se examinan las salidas del software y su entorno operacional para comprobar que funciona tal y como se requiere (Sommerville, 2005).

Niveles de prueba

El proceso de prueba se realiza en un determinado momento del ciclo de desarrollo de las funcionalidades. A continuación se muestran algunos tipos de niveles de pruebas para valorar cuales serán utilizados:

Prueba de Unidad: centra el proceso de verificación en la menor unidad del diseño del software: el componente de software o módulo. Se prueba la interfaz del módulo para asegurar que la información fluye de forma adecuada hacia y desde la unidad de programa que está siendo probada (Ruiz, 2010).

Prueba de Integración: es una técnica sistemática para construir la estructura del programa mientras que, al mismo tiempo, se llevan a cabo pruebas para detectar errores asociados con la interacción. El objetivo es tomar los módulos probados mediante la prueba de unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño (Ruiz, 2010).

Pruebas del Sistema: está constituida por una serie de pruebas diferentes cuyo propósito primordial es ejercitar profundamente el sistema basado en computadora. Aunque cada prueba tiene un propósito diferente, todas trabajan para verificar que se han integrado adecuadamente todos los elementos del sistema y que realizan las funciones apropiadas (Ruiz, 2010).

Prueba de Aceptación: las realiza el usuario final en lugar del responsable del desarrollo del sistema, una prueba de aceptación puede ir desde un informal paso de prueba hasta la ejecución sistemática de una serie de pruebas bien planificadas. Se lleva a cabo con el objetivo de que el cliente valide los requisitos que debe poseer el sistema (Ruiz, 2010).

Capítulo3: Implementación y prueba

Métodos de prueba

Las técnicas de prueba proporcionan distintos criterios para generar casos de prueba que provoquen fallos en los programas. Estas técnicas se agrupan en (Juristo, et al., 2006):

- **Técnicas de caja blanca o estructural:** se basan en un minucioso examen de los detalles procedimentales del código a evaluar, por lo que es necesario conocer la lógica del programa.
- **Técnicas de caja negra o funcional:** realizan pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa. No es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar.

Para la validación de la propuesta de solución, se realizaron las pruebas unitarias no informatizadas ni documentadas, estas fueron realizadas por el equipo de desarrollo a medida que se fue implementando, por lo que las deficiencias detectadas se solucionaron durante la ejecución de las mismas. Además se realizaron las pruebas de caja negra teniendo en cuenta el método de particiones equivalentes.
Este método divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar. El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. (Pressman) También se realizaron las pruebas de Aceptación por parte del cliente.
Para la realización de las pruebas de integración se utilizó una estrategia de verificación paso a paso, teniendo como base de control de las mismas una matriz de trazabilidad, donde se asocian o definen dependencias entre las salidas de las funciones programadas en los módulos de cotización e ingreso a las filas y las funcionalidades ya existentes en la plataforma. Pudiendo verificar que no se generan errores o salidas no deseadas, acorde con lo solicitado por el cliente.

Diseño de casos de prueba

Un caso de prueba especifica una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las que ha de probarse. También puede derivarse de, y por

tanto puede seguir la traza de, un caso de uso en el modelo de caso de uso o de una realización de caso de uso en el modelo de diseño (Jacobson, et al., 2000).

A continuación se presentan los casos de prueba de los CU Gestionar solicitud ingreso y Gestionar Cotización. Los restantes casos de prueba se encuentran en los Anexos.

Tabla 4. Diseño de casos de prueba del caso de uso Gestionar solicitud de ingreso

CP Gestionar solicitud ingreso				
Descripción general El caso de uso inicia cuando un miembro del universo juvenil decide solicitar el ingreso a las filas. Para ello el actor selecciona la opción Solicitar ingreso del menú Ingreso a las filas. El sistema brinda la opción de crear una nueva solicitud. Si así lo desea el actor, el sistema muestra un formulario con los campos correspondientes a una solicitud. Permite mostrar todos los datos de la solicitud y además permite ver los requisitos a cumplir para el ingreso haciendo clic en ‘‘Ver requisitos’’. En caso de que desee modificar los datos de la solicitud, el actor deberá seleccionar la opción de editar y realizar las modificaciones, terminando así el caso de uso. **Condiciones de ejecución** El miembro del universo juvenil debe estar autenticado.				
SC1. Crear solicitud				
Escenario	**Descripción**	**Motivo**	**Respuesta del sistema**	**Flujo Central**
EC1.1	Selecciona la opción ‘‘Solicitar ingreso’’ del menú Ingreso a las filas.		Brinda la opción de crear una nueva solicitud.	Ingreso a las filas/Solicitar ingreso
EC1.2	Selecciona la opción ‘‘Nueva’’.		Se muestra un formulario con el siguiente campo: • Motivo	Ingreso a las filas/Solicitar ingreso/Nueva

			Y la opción: • Enviar • Cancelar Y permite: Mostrar los requisitos a cumplir para el ingreso	
EC1.3	Introduce los datos y selecciona ‘‘Enviar’’.	V	Valida los datos. Registra una nueva solicitud en el sistema. Muestra un mensaje de éxito.	Ingreso a las filas/Solicitar ingreso/Nueva/Enviar
EC1.4	Existen campos vacíos.	I	Muestra un mensaje informando: “Rellene este campo”. Muestra un indicador sobre el campo vacío. Regresa al 1.2	Ingreso a las filas/Solicitar ingreso/Nueva/Enviar
EC1.5	Selecciona la opción ‘‘Cancelar’’.		Regresa a la vista anterior.	Ingreso a las filas/Solicitar reclamaciones/Nueva/Cancelar

SC2 Editar datos de la solicitud				
Escenario	**Descripción**	**Motivo**	**Respuesta del sistema**	**Flujo Central**
EC2.1	Selecciona la opción ‘‘Solicitar ingreso’’ del menú Ingreso a las filas.		Lista la solicitud y muestra las siguientes opciones: • Editar • Ver. Ver SC3: ‘‘Ver datos de la solicitud’’.	Ingreso a las filas/Solicitud ingreso
EC2.2	Selecciona la opción ‘‘Editar’’.		Abre una ventana con los datos correspondientes a la solicitud permitiendo modificar el valor de la misma: • Motivo Y la opción • Guardar • Cancelar	Ingreso a las filas/Solicitud de ingreso/Editar
EC2.3	Modifica los datos y selecciona la opción ‘‘Guardar’’.	V	Valida los cambios realizados. Actualiza la solicitud en el sistema. Muestra un mensaje de éxito.	Ingreso a las filas/Solicitar ingreso/Editar/Guardar
EC2.4	Existen campos vacíos.	I	Muestra un mensaje informando: “Rellene este campo”. Muestra un indicador sobre el campo vacío.	Ingreso a las filas/Solicitar ingreso/Editar/Guardar

			Regresa al 2.2	
EC2.5	Selecciona la opción ‘‘Cancelar’’.		Regresa a la vista anterior.	Ingreso a las filas/Solicitar ingreso/Editar/Cancelar
SC3 Ver datos de la solicitud				
Escenario	**Descripción**	**Motivo**	**Respuesta del sistema**	**Flujo Central**
EC3.1	Selecciona la opción ‘‘Solicitar ingreso’’ del menú Ingreso a las filas.		Lista la solicitud y muestra las siguientes opciones: • Ver • Editar. Ver SC2: ‘‘Editar datos de la solicitud’’.	Ingreso a las filas/Solicitar ingreso
EC3.2	Selecciona la opción ‘‘Ver’’.		Abre una ventana con los datos correspondientes a la solicitud: • Nombre y apellidos • Motivo Y la opción: • Cerrar	Ingreso a las filas/Solicitar ingreso/ Ver
EC3.3	Selecciona la opción ‘‘Cerrar’’.		Cierra la vista actual.	Ingreso a las filas/Solicitar ingreso/Ver/Cerrar

Tabla 5. Diseño de caso de prueba del caso de uso Gestionar cotización

CP Gestionar cotización						
Descripción general El caso de uso inicia cuando el cotizador decide crear la cotización del mes. Para ello el actor selecciona la opción Cotización en el menú. El sistema permite crear una nueva cotización. Si así lo desea el actor, el sistema registra el pago del militante y si este pagó o no. Muestra los datos referentes a la misma y además propone una sanción a aquellos que hayan incumplido con el pago de la cotización permitiendo su modificación. Muestra el importe del comité de base y genera un listado con el importe del CB, terminando así el caso de uso. **Condiciones de ejecución** El cotizador debe estar autenticado y no debe haber creado una cotización en el mismo mes.						
SC1. Crear cotización						
Escenario	**Descripción**	**Ocupación**	**Aporte**	**Pagó**	**Respuesta del sistema**	**Flujo Central**
EC1.1	Selecciona la opción ‘‘Cotización’’.				Brinda la opción de crear una nueva cotización	Cotización
EC1.2	Selecciona la opción ‘‘Nueva’’.				Permite: • Seleccionar la ocupación del militante (inicialmente el sistema por defecto la ocupación que muestra es la de estudiante)	Cotización/Nueva

					• Marcar si pagó o no Registra el pago del militante Y las opciones: • Crear • Cancelar	
EC1.3	Introduce los datos y selecciona la opción “Crear”.	V	V	V	Valida los datos. Registra una nueva cotización.	Cotización/Nueva
EC1.4	Selecciona la opción ‘‘Cancelar’’.				Regresa a la vista anterior.	Cotización/Nueva
SC2 Ver datos de una cotización						
Escenario	**Descripción**	**Ocupación**	**Aporte**	**Pagó**	**Respuesta del sistema**	**Flujo Central**

EC2.1	Selecciona la opción “Cotización”.				Lista las cotizaciones disponibles en el sistema y por cada una brinda la posibilidad de: • Ver.	Cotización
EC2.2	Selecciona la opción ‘‘Ver’’.				Muestra los datos de la cotización: • La ocupación del militante • El pago del militante • Si pagó o no • Los meses sin pagar por cada militante del CB • Sanciones del militante si tiene Y permite: • Proponer una sanción al militante por no pago de la cotización (Ver: Tabla 43.Diseño de caso de prueba del caso de uso Proponer sanción) Y la opción: • Cerrar	Cotización/Ver

EC2.3	Selecciona la opción ‘‘Cerrar’’.				Cierra la vista actual. Retorna al usuario a la vista anterior. Termina el caso de uso.	Cotización/Ver/Cerrar
SC3 Mostrar importe de CB						
Escenario	**Descripción**	**Ocupación**	**Aporte**	**Pagó**	**Respuesta del sistema**	**Flujo Central**
EC3.1	Selecciona la opción ‘‘Cotización’’.				Lista las cotizaciones disponibles en el sistema generando un listado mostrando el importe por CB.	Cotización

Resultados obtenidos en las pruebas

En las pruebas realizadas al sistema mediante el método de caja negra, se pudo comprobar el cumplimiento de los requisitos funcionales y no funcionales del software obtenido.

A continuación se presenta el resultado de las pruebas realizadas:

Tabla6. Resultado de las pruebas de caja negra

Iteraciones	Cantidad de casos de prueba	No conformidades detectadas			
		Validación	Funcionalidad	Interfaz de usuario	Total
1	20	12	5	10	27
2	20	7	3	5	15
3	20	3	2	1	6

Prueba final	20	0	0	0	0

La tabla anterior muestra la ejecución de tres iteraciones y una cuarta iteración de prueba final, donde en cada una se le da solución a las no conformidades detectadas, mejorando la calidad del software y preparándolo progresivamente para su uso final.

Tabla7. Resultado de las pruebas de integración

Funcionalidades programadas	Funcionalidades existentes			
	Incorporar militante	Sancionar militante	Reporte de acta de asamblea ordinaria	Cierre de funcionamiento
Crear acta de asamblea de Aceptación	S			
Proponer sanción		S		
Cargar acuerdos de asambleas anteriores			S	
Mostrar invitados del universo juvenil			S	
Exportar cierre de funcionamiento a Excel				S

S: Resultados satisfactorios.

I: Resultados incorrectos.

Capítulo3: Implementación y prueba

Teniendo en cuenta la hipótesis planteada, se puede afirmar que la misma ha sido cumplida debido a que con los módulos implementados se logra la agilización de los procesos de ingreso a las filas y cotización. A continuación se muestran aspectos relacionados con los mismos, donde para la verificación de su cumplimiento anteriormente se contaba con varios pasos, reduciéndose los mismos en cuanto a tiempo y esfuerzo con el desarrollo de la propuesta planteada:

- Análisis de las solicitudes de ingreso
- En cuanto a la documentación que se genera de los procesos de ingreso a las filas y cotización, no necesariamente se debe esperar a que llegue en formato duro al nivel central para su análisis debido a que la misma puede ser consultada en el sistema
- Análisis de las autobiografías
- Tabulación de la cotización
- Verificación del pago de la cotización

3.4 Conclusiones Parciales

Los diagramas de componente obtenidos en el flujo de trabajo de implementación permitieron concebir le modelo de implementación, de forma tal que describe como los elementos del modelo de diseño se implementan en términos de componentes.

Las pruebas realizadas al sistema validaron que las funcionalidades desarrolladas satisfacen los requerimientos especificados.

Conclusiones generales

Con la realización del presente trabajo se arribaron a las siguientes conclusiones:

- Se realizó un estudio acerca de los procesos de Ingreso a las filas y Cotización evidenciando la necesidad de informatizar ambos procesos para mejorar el control de la información de la organización en la UCI.
- Se implementaron los módulos de gestión de información con los datos obtenidos en el estudio de los procesos Ingreso a las filas y Cotización de la UJC.
- Se hicieron pruebas de sistema y las no conformidades detectadas fueron resueltas.
- El cliente quedó satisfecho con los módulos desarrollados, permitiendo solucionar un grupo de problemas existentes en las filas de la UJC.

Recomendaciones

Concluida la investigación y con el objetivo de mejorar la calidad del desarrollo de los procesos de ingreso a las filas y cotización se recomienda:

- Los incumplimientos del pago de la cotización sean cargados automáticamente en el acta de las asambleas ordinarias de forma tal que por una mala planificación del orden del día no se dejen de analizar los incumplimientos con respecto al pago de la cotización.

- Una vez que se procede a sancionar a un militante por el incumplimiento de deberes en la organización debe exportarse al formato oficial todos los datos que lleva una sanción de forma tal que se pueda imprimir e incorporar al expediente físico del militante.

- Cuando se procede a la evaluación de la militancia en un período, deben ser incorporados en el aspecto señalamientos de forma automática, todos los incumplimientos del militante en el actual período.

Bibliografía

1. **Alvarez, Miguel A. 2009.** *Manual de jQuery.* 2009.
2. **Ambler, Scott.** The Agile Unified Process (AUP). [En línea] [Citado el: 4 de 2 de 2016.] http://www.ambysoft.com/unifiedprocess/agileUP.html.
3. **BluePrintCSS. 2009.** Sitio oficial de. Sitio oficial de BluePrintCSS. [En línea] 26 de 10 de 2009. [Citado el: 8 de 2 de 2016.] http://www.blueprintcss.org/..
4. **Cáceres, Abdiel González. 2004.** *Lenguajes de programación.* México : s.n., 2004.
5. **Cáceres, Jesús tello.** *Diagramas de Casos de Uso.* Arcalá : Universidad de Arcalá.
6. **Castellanos, Abel Olivares.** *Sistema Informático para la Gestión de la Información de La Unión de Jóvenes Comunistas de la Facultad 2. .*
7. **Comunistas, Union de Jovenes. 2014.** *Estatutos de la Union de Jovenes Comunistas.* 2014.
8. **Comunistas, Unión de Jóvenes. 2014.** *Reglamento de la Unión de Jóvenes Comunistas.* 2014.
9. **CORDERO, JORGE LUIS.** *METODOLOGIAS AGILES.PROCESO UNIFICADO AGIL (AUP).* LA PAZ, EL ALTO – BOLIVIA : s.n.
10. **Corporation, M. 2002.** *MSF Process Model v. 3.1.* 2002.
11. **Días, Marlen Fajardo.** *Sistema Integral de Gestión de la UJC UCI_Módulo Militante. .*
12. **Eguiluz, Javier. 2011.** *Desarrollo Web ágil con Symfony2.* 2011.
13. **Eguíluz, Javier Pérez. 2009.** Introducción a JavaScript. [En línea] 2009. [Citado el: 8 de 2 de 2016.] http://www.librosweb.es/javascript/.
14. **—. 2008.** *Introducción a CSS.* 2008.
15. **Eriksson, H.-E. and Penker, M. 2000.** *Business Modeling with UML: Business Patterns at Work.* 2000.
16. **Fajardo, Marlén Díaz y Feria Eraldo M., Sánchez. 2009.** *Sistema integral de gestión de la UJC UCI: Módulo Militante.* Habana : s.n., 2009.
17. **Ferré, Xavier Grau, Sánchez , María Isabel Segura.** *Desarrollo Orientado a Objetos con UML.* s.l. : Facultad de Informática- UPM.
18. **Freeman, E. and Freeman, E. 2004.** *Head First Design Patterns.* 2004.
19. **García, J. C. 2012.** *SOLID y GRASP. Buenas pr´acticas hacia el´exito en el desarrollo de software.* 2012.

20. **Gen. 2015.** GENBETA: desarrollo y software. [En línea] 2015. [Citado el: 7 de 2 de 2016.] http://www.genbetadev.com/frameworks/bootstrap.
21. **GestPeople. 2014.** GestPeople. [En línea] 16 de 4 de 2014. [Citado el: 27 de 1 de 2016.] http://gestpeople.waxoo.com/.
22. **Gutiérrez, J. J. 2009.** ¿Qué es un framework web? [En línea] 2009. [Citado el: 7 de 2 de 2016.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf..
23. **HANSEN, KENNETH E., JOHNSON, JAMES H. y and WILLIAMS, THOMASA.** *Development of an on-line management information system for community mental health centers.* s.l. : Behavior Research Methods &Instrumentation, Vol.IX.
24. **Hinostroza, R. R. 2005.** Características de PHP. [En línea] 2005. [Citado el: 9 de 2 de 2016.] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP..
25. **Informáticas, Universidad de las Ciencias. 2015.** *Metodología de desarrollo para la Actividad productiva de la UCI.* Habana : s.n., 2015.
26. **Ivar Jacobson, Grady Booch, James Rumbaugh. 1999.** *El Proceso Unificado de Desarrollo de Software. .* 1999.
27. **J.Schmuller. 2001.** Aprendiendo UML en 24 horas. [En línea] 2001. [Citado el: 8 de 2 de 2016.] http://www.intercambiosvirtuales.org/libros-manuales/aprendiendo-uml-en-24-horas-joseph..
28. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** *El lenguaje Unificado de Modelado.* Madrid : Pearson Educación S.A, 2000.
29. **—. 2000.** *El Proceso Unificado de Desarrollo de Software.* 2000.
30. **Julin, Mailen García.** *Módulo de reportes para el Sistema Integral de la UJC Nacional.*
31. **Juristo, Natalia, Moreno, Ana M. y Vegas, Sira. 2006.** *TÉCNICAS DE EVALUACIÓN DE SOFTWARE.* 2006.
32. **Kabir, Mohammed J. 2012.** *la Biblia.Servidor Apache2.* 2012.
33. **Larman, Craig. 2003.** *UML y Patrones. Segunda edición. Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* 2003.
34. **Larman, Craig. 2003.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México : s.n., 2003.
35. **Letelier, Patricio y Sánchez López, Emilio A. 2003.** Métodologías Ágiles en el Desarrollo de software. Valencia : Grupo ISSI(Ingeniería del Software y Sistemas de Información), 2003.

36. **Lorenzo, Fernando Castro. 2000.** *Modelos de datos, conceptos y clasificación.* 2000.
37. **Martínez, Yasmany Torres y Montenegro, Carlos Amador. 2015.** *Sistema Integral para la Gestión de la Información de los Procesos Sustantivos de la UJC en la UCI. SIGPS_UCI.* Habana : s.n., 2015.
38. **Mateu, Carles. 2004.** *Desarrollo de aplicaciones web.* 2004.
39. **Mesa, Yunior Reyes, y otros. 2016.** *MÓDULO DE GESTIÓN DE INGRESOS POR CONCEPTO DE COTIZACIÓN DE LA UNIÓN DE JÓVENES COMUNISTAS PARA EL SISTEMA DE CONTROL DE LA MILITANCIA (SICOM-UJC).* Habana : s.n., 2016.
40. **Montes, María. 1999.** *Herramientas CASE.* s.l. : INSTITUTO NACIONAL DE ESTADISTICA E INFORMATICA, 1999.
41. **Net. 2011.** Netbeans. [En línea] 2011. [Citado el: 10 de 1 de 2016.] http://netbeans.org/features/ide/index.htm..
42. **Olivares, Abel Castellanos y López, William Consuegra. 2009.** *Sistema Informático para la Gestión de la Información de La Unión de Jóvenes Comunistas de la Facultad 2.* Habana : s.n., 2009.
43. **Övergaard, Gunnar y Palmkvist, Karin. 2004.** *Use Cases Patterns and Blueprints.* s.l. : Addison Wesley Professional,, 2004.
44. **Palacio, J. 2006.** *El modelo Scrum.* 2006.
45. **PostgreSQL. 2010.** PostgreSQL- Sitio Oficial. [En línea] 2010. [Citado el: 12 de 1 de 2016.] http://www.postgresql.org..
46. **Potencier, Fabien y Zaninotto, Francois. 2007.** *Symfony la guía definitiva.* 2007.
47. **Pressman, Roger S.** *Ingeniería del software. Un enfoque práctico. Quinta edición.*
48. **Ramírez, Yaisel Avilés.** *Sistema Integral de Gestión de la UJC Nacional: Módulo Militante. .*
49. **Reyes, Yilen Yamilé Montejo. 2012.** *Sistema de Información para la UJC de la UCI. Subsistema para la gestión del ID2 de la UJC de la UCI.* Habana : s.n., 2012.
50. **Ridao, M. y Doorn, J. and Sampaio, J. 2000.** *Uso de Patrones en la Construcción.* 2000.
51. **Ruiz, Roberto Tenorio. 2010.** *Las Pruebas de Software y su Importancia en las Organizaciones.* 2010.
52. **Ruiz, Susel Durán y Piloto, Yenisleidy Lastra, Roselló, Reynaldo Núñez. 2015.** *El peligro de un Caso de Uso muy largo. Mitos y realidades.* 2015.

53. **Santiesteban, I. I. P., Medina, M. R., Piña, J. L. G., and Garrido, Y. V. 2011.** 2011.
54. **Sommerville, I. (2005.** *Ingeniería del Software.* (2005.
55. **Sommerville, Ian. 2005.** *Ingeniería del software. Séptima edición.* s.l. : Pearson Addison Wesley, 2005.
56. **Team, Doctrine.** Object Relational Mapper. [En línea] [Citado el: 8 de 2 de 2016.] http://www.doctrine-project.org/projects/orm.html.
57. **Urtate, Gerardo Camejo.** *Sistema de Gestión de la Unión de Jóvenes Comunistas de la Facultad 4.*
58. **Vis. 2015.** Visual Parading. [En línea] 2015. [Citado el: 10 de 1 de 2016.] http://www.visual-paradigm.com..
59. **workmeter. 2015.** workmeter_Buen trabajo. [En línea] 2015. [Citado el: 27 de 1 de 2016.] http://es.workmeter.com/workmeter-optimiza-productividad-empresarial.
60. **Young, R. 2004.** *The Requirements Engineering Handbook.* 2004.