
*FACULTAD DE INGENIERÍA*

*ESCUELA DE SISTEMAS*

*DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS Y COMPUTACIÓN*

*TEMA:*

# DISEÑO, IMPLEMENTACIÓN DEL SISTEMA DE GESTIÓN PARA EL CENTRO DE PSICOLOGÍA APLICADA DE LA P.U.C.E.

*AUTOR:*

**MARIO AGUSTÍN GRANJA ORAMAS**

**FRANCISCO JOSÉ GORDILLO CORONEL**

**QUITO – 2010**

# DEDICATORIA

El presente trabajo está dedicado a todas aquellas personas que de una u otra forma, estuvieron a nuestro lado prestándonos siempre su valiosa ayuda.

# AGRADECIMIENTOS

Agradezco a Dios por encaminar nuestro sendero creciendo profesionalmente, espiritualmente y por darnos la salud y fuerza para poder seguir adelante.

Agradezco a mis padres por su apoyo incondicional a lo largo de toda mi vida, instruyéndome para seguir por un camino de bien.

A mis profesores, por sus valiosas enseñanzas ya que sin ellas no habría podido alcanzar esta meta de mi vida y sus historias llenas de experiencia que me ayudaron a madurar profesionalmente.

# TABLA DE CONTENIDO

# TABLA DE FIGURAS

# ÍNDICE DE TABLAS

# RESUMEN

El presente trabajo muestra el desarrollo del sistema del Centro de Psicología Aplicada de la PUCE, para el desarrollo del mismo, se utilizó el paradigma de la programación orientada a objetos y la metodología del ciclo de vida en cascada.

La idea de desarrollar este sistema surge de la necesidad del Centro de Psicología Aplicada de automatizar sus procesos para incrementar su productividad y mantener la información de sus actores actualizada y consistente. Este trabajo está dividido en cinco capítulos, y una parte introductoria en la que se encuentran los objetivos que se esperan cumplir al final del trabajo así como la justificación y el alcance del mismo, en los cuatro primeros capítulos se desarrolla el sistema de acuerdo a las fases del ciclo de vida en cascada, el quinto capítulo está destinado para las conclusiones y recomendaciones que se obtendrán una vez concluido el desarrollo del sistema.

Las fases del ciclo de vida en cascada generan ciertos documentos o productos, dichos documentos se encuentran en los anexos al igual que la documentación de los procesos del Centro de Psicología Aplicada que están siendo automatizados.

A continuación la parte introductoria del trabajo:

# OBJETIVOS:

## GENERAL:

Diseñar e implementar un sistema de Gestión para el Centro de Psicología Aplicada de la Pontificia Universidad Católica del Ecuador.

## ESPECÍFICOS:

- Analizar la situación actual de los procesos y de los datos del Centro de Psicología Aplicada de la P.U.C.E.
- Diseñar un sistema de características dinámicas y parametrizable.
- Desarrollar la aplicación que incluya todos los procesos descritos en el Alcance.
- Realizar pruebas para la certificación del Sistema.
- Implementar el sistema.
- Realizar Conclusiones y Recomendaciones.

## ALCANCE:

- El trabajo culmina con la implementación del sistema de Gestión para el Centro de Psicología Aplicada de la PUCE. Este Sistema tendrá los siguientes módulos:
- Módulo de Terapeutas
- Módulo de Perfiles y Usuarios
- Módulo de Registro de Citas
- Módulo de Elaboración de Informes
- Módulo de Facturación
- Módulo de Reportes
- Módulo de Instituciones de referencia
- Módulo de Soporte del Sistema

# 1. CAPÍTULO I

En este primer capítulo, se define en que consiste el Centro de Psicología Aplicada, y también, se dará una explicación de cada módulo que interactúa en el sistema. Luego, dentro del marco teórico se dará las características de las herramientas del desarrollo de software, la metodología aplicada, tablas comparativas de otras herramientas y motivos de elección de las herramientas de desarrollo seleccionada.

## 1.1 Qué es el Centro de Psicología Aplicada de la PUCE

El Centro de Psicología Aplicada, CPsA., es un organismo de la Facultad de Psicología de la Pontificia Universidad Católica del Ecuador, el cual presta servicios a los usuarios de la comunidad universitaria y extra-universitaria, con proyectos de calidad en las áreas de:

### 1.1.1 Psicología Clínica

- Psicodiagnóstico individual.
- Psicoterapia: Individual, grupal, vincular.
- Procesos de Asesoramiento y consejería.
- Peritajes psicológicos.
- Supervisión clínica para terapeutas.

### 1.1.2 Psicología Organizacional

La principal área de influencia está dirigida a los sistemas y subsistemas de todas las organizaciones. Brindando asesoría para el correcto manejo del Recurso Humano.
Asesoramiento en procesos de capacitación, liderazgo, motivación, selección y entrevista personal.

- Procesos de Selección de personal.
    - Evaluación psicotécnica.
    - Entrevistas de preselección.
    - Presentación de terna final para la selección.

- Consultoría de Recursos Humanos
    - Diseño e implementación del proceso de selección de personal.

- Capacitación y Desarrollo Humano.
- Evaluación de Desempeño.
- Análisis y Descripción de cargos.
- Valoración y Clasificación de cargos.

- Consultoría en Desarrollo Organizacional
  - Diagnóstico de Ambiente y Clima Organizacional.
  - Asesoría en el mejoramiento de ambientes organizacionales.

**1.1.3 Psicología Educativa.**

- Orientación y Consejería Grupal.
- Orientación vocacional y profesional.
- Diagnóstico de problemas de aprendizaje.
- Entrenamiento en métodos de estudio.
- Evaluación Psicopedagógica a personas con necesidades especiales.
- Asesoramiento a Familias y profesores de personas con necesidades.
- Diagnóstico y Evaluación Psicopedagógica.
- Tratamiento de dificultades de aprendizaje.
- Servicios especiales de apoyo para estudiantes PUCE: Orientación y asesoría.
- Planificación, métodos y técnicas de estudio; Tratamiento en dificultades de aprendizaje.

## 1.2 Antecedentes

El Centro de Psicología Aplicada de la PUCE realizó un trabajo en el cual se ejecutó un levantamiento de procesos y se determinó la necesidad de implantar un sistema informático que permita optimizar recursos para mejorar la prestación y control de servicios de la organización.

Entre las recomendaciones que posee dicho documento se sugiere el contacto con la Facultad de Ingeniería de la PUCE con el propósito de encontrar estudiantes que se encarguen del desarrollo como tema de disertación de grado y genere el Sistema respondiendo al trabajo existente en el cual se detallan cada uno de los módulos que se requiere del sistema informático.

Por otro lado, la Facultad de Ingeniería y su Escuela de Sistemas permanentemente preocupadas del aporte que debe generar, posee la capacidad de llevar a cabo este proyecto con calidad pretendiendo obtener un sistema que satisfaga los requerimientos del Centro de Psicología Aplicada de la P.U.C.E.

## 1.3 Definición de los Módulos

**Estructura y Organización**

El Centro de Psicología Aplicada está conformado por los siguientes estamentos: Director (a), Psicólogos de Planta, Profesionales Ocasionales, Secretaria, Practicantes Pre-profesionales y Voluntarios, que bajo la coordinación de funciones constituyen el cuerpo institucional frente al que la comunidad universitaria y extrauniversitaria demanda sus servicios. La atención se brinda en 3 Áreas: Psicología Clínica, Educativa y Organizacional.

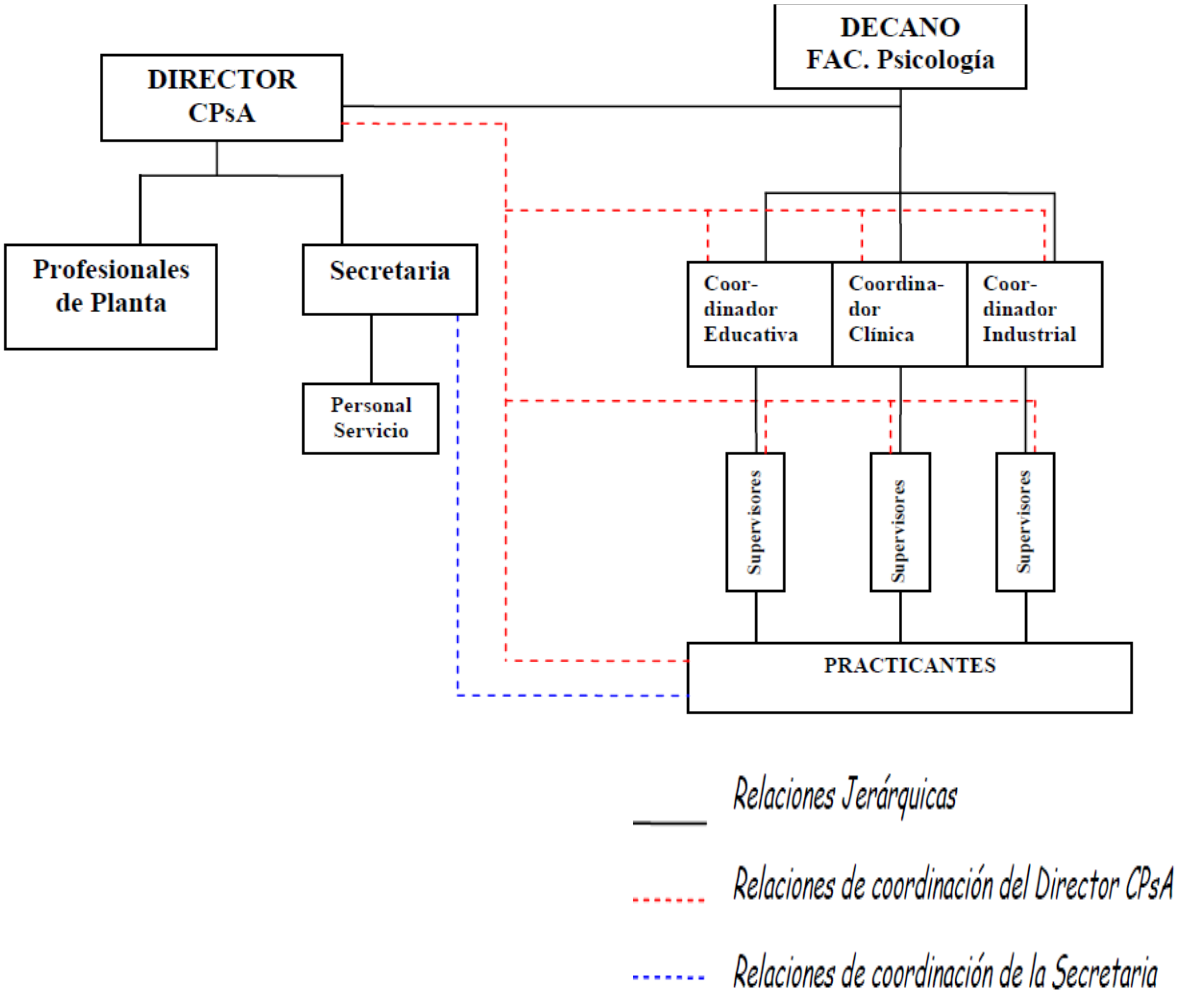
**Relación con la Facultad de Psicología**

El Centro de Psicología Aplicada facilita la formación de los estudiantes de la Facultad a través de las prácticas estudiantiles, las cuales conjuntamente con la actividad de los profesionales, articulan la oferta de servicios de atención, asesoría y terapia y la generación de proyectos que constituyen vías que posibilitan y contribuyen al autosostenimiento del Centro.
La relación con la Facultad tiene que dar cuenta de esta realidad en su gestión manteniendo una relación directa con el Centro de Psicología Aplicada en los temas que a continuación se exponen.

**Autonomía del Centro**

- Planificación y organización de los servicios y uso del espacio físico.
- Elaboración y Ejecución del Presupuesto.
- Elaboración, mercadeo y ejecución de Proyectos de asesoría o investigación.
- Celebración de acuerdos de cooperación con Instituciones Gubernamentales y No Gubernamentales para la venta de servicios.

### 1.3.0 Estructura Centro de Psicología Aplicada




| | |
|---|---|
| **Título** | Figura 1.0- Estructura Centro de Psicología Aplicada |
| **Descripción** | Estructura Organizacional entre El Decano de la Facultad de Psicología, el Director del Centro los Terapeutas Profesionales y los practicantes |
| **Fecha** | Marzo 2007 |
| **Copia o Fuente** | Normativa interna Centro de Psicología Aplicada.pdf |

### 1.3.1 Módulo de Terapeutas

Este módulo contiene la información general acerca de las personas que se encuentran prestando sus servicios en el CPsA, toma en cuenta a profesionales de planta y a practicantes.

Se debe considerar que los practicantes cambian cada semestre por lo que se propone que la información de quienes empiezan a realizar las prácticas la ingresen ellos mismos al sistema; esta renovación de información es semestral.

En el caso de los profesionales de planta, la información se ingresa una sola vez y se modifica en caso de que sea necesario, o se elimina si el profesional deja de prestar sus servicios al centro.

**Psicólogos de Planta**

Profesionales en las tres especializaciones, contratados bajo la modalidad de profesores por la universidad.

Funciones:

- Brindar atención a los usuarios del Centro de acuerdo a su especialidad y a los servicios ofertados por el CPsA.
- Colaborar con el Director en la planificación y diseño de modalidades de atención y otros proyectos del CPsA.

**Datos Generales del Terapeuta:**

- Apellidos.
- Nombres.
- Número de Cédula.
- Fecha de Nacimiento.
- Estado Civil.
- Dirección.
- Teléfono (Convencional y Celular).
- Correo electrónico.
- Especialización.
- Horario.

**Profesionales Ocasionales**

Profesionales en las tres especializaciones, contratados bajo la modalidad de honorarios.

Funciones:

- Encargarse del trabajo directo para el que fueron contratados de acuerdo a su especialidad.

**Practicantes**

**Definición**:

Se consideran practicantes a los estudiantes de Psicología que han llenado los requisitos necesarios estipulados por la Facultad y cuyos deberes y derechos están especificados en el Reglamento de Prácticas.

**Evaluación:**

La evaluación de los practicantes por parte del Centro se hará según parámetros relativos a su asistencia y cumplimiento de tarea tal y como se refleja en el título III de las sanciones; esto será puesto en conocimiento de los estudiantes al inicio del semestre.

**Sanciones:**

a. Las sanciones por inasistencias al Centro de Psicología Aplicada serán las siguientes:

- En caso de no tener pacientes asignados:
    - Por cada inasistencia injustificada: 1 punto menos a la nota final del semestre.
    - Se tolerará 1 inasistencia justificada. A partir de la 2da, por cada tres inasistencias justificadas, 1 punto menos.
    - Se tolerarán 5 atrasos de hasta 10 minutos. A partir del 6to atraso, pasado 10 minutos, 1 punto menos.

- En caso de tener pacientes asignados:
    - No se tolerarán inasistencias injustificadas. Por cada falta injustificada, 1 punto menos.
    - Por cada 3 faltas justificadas en el semestre 1 punto menos.
    - Por cada 3 atrasos pasado 10 minutos 1 punto menos
    - Si se trata de una primera cita, a partir de 10 minutos de atraso se le asignará el caso a otro practicante o terapeuta y se sancionará con 1 punto menos.

En general:

- Todo atraso que exceda los 30 minutos cuenta como falta injustificada y será sancionado según lo establecido anteriormente

- Los estudiantes deberán llenar los registros de pacientes del centro tal como se les indique; por cada 5 incumplimientos en los registros, se le sancionará con 1 punto menos a la nota final del semestre.

- Cada incumplimiento a las siguientes normas será sancionado con amonestación escrita:

    - Saludo a los pacientes y al personal del Centro
    - Manejo correcto de la documentación del Centro (registros, carpetas)
    - Empleo correcto de los materiales del Centro (papelería, material de juego, tests, libros, computadoras, teléfonos)
    - Empleo correcto de las instalaciones del Centro (cubículos, salas de espera, salas de juego, cámara de Gesel).

Por cada tres amonestaciones, se sancionará con un punto menos.

b. Las sanciones por incumplimiento de las normas del CPsA. o al código ético serán las que se establecen en el Reglamento General de Estudiantes.

**Datos Generales del Practicante:**

- Apellidos.
- Nombres.
- Número de Cédula.
- Fecha de Nacimiento.
- Estado Civil.
- Dirección.
- Teléfono (Convencional y Celular).
- Correo electrónico.
- Especialización.
- Horario.

### 1.3.2 Módulo de Perfiles y Usuarios

Este módulo abarca las Fichas Personales de cada usuario (registro personal) y la consulta de tarifas establecidas a cada usuario.

Este módulo controla los privilegios de los usuarios sobre el sistema, el Director del centro se encarga de decidir los alcances de uso de los usuarios (permisos, claves).

**Director**

Nombrado por el Rector de la PUCE para períodos de dos años, de los candidatos propuestos por el Consejo de Facultad de Psicología. En su perfil debe considerarse: ser psicólogo con capacidad para la gestión administrativa, tener respeto a diferentes enfoques teóricos, poseer conocimiento sobre proyectos, tener un buen manejo de las relaciones interpersonales.

**Funciones:**

- Organizar el trabajo de las prácticas que se realicen en el Centro.
- Fijar las políticas de atención a sus usuarios.
- Informar a los supervisores sobre el desempeño de los practicantes dentro del centro durante el período de prácticas.
- Realizar una evaluación del desempeño de los estudiantes al finalizar cada semestre de prácticas.
- Velar por el respeto de la normativa interna del Centro.
- Colaborar en la solución de problemas que eventualmente surjan en el trabajo de los practicantes dentro del Centro de Psicología.
- Fijar el costo de los servicios del Centro con el aval de las instancias pertinentes.
- Realizar convenios con instituciones públicas y privadas para la venta de los servicios del Centro.

**Secretaría**

Personal ad-hoc contratada por la Universidad para el efecto. En su perfil debe considerarse: manejo adecuado de programas informáticos, sensibilidad con las problemáticas sociales, formación extra en técnicas de manejo de crisis.

**Funciones:**

- Mantener el contacto con los usuarios.
- Dar información general del Centro cuando es requerida.
- Consignar datos iniciales sobre el usuario que demanda servicios.
- Asignar las citas al usuario según la disponibilidad del mismo y del terapeuta o practicante.
- Cobrar los honorarios a los usuarios.
- Responsabilizarse de la conservación del material y del mantenimiento y cuidado del archivo, carpetas, tarjetas, registros, etc.
- Transcribir documentos, memorándums, etc. que soliciten el Director y los profesionales de planta; documentos solicitados por los supervisores en asuntos relacionados con el Centro y certificados para los practicantes.

**Ficha Personal**

- Nº de Historia Clínica
- Fecha de ingreso
    - o TIPO DE USUARIO
        - Estudiante de la PUCE / Facultad.
        - Empleado de la PUCE / Ocupación actual.
        - Extrauniversitario.
    - o ÁREA PERSONAL
        - Nombre / Apellidos / Edad.
        - Sexo / Estado Civil.
        - Escolaridad / Institución.
        - Ocupación actual / Teléfono.
        - Domicilio / Teléfono.
        - Nombre de una persona a la que se pueda llamar en caso de ser necesario.
        - Nombre / Teléfono.
    - o ÁREA FAMILIAR
        - Nombre del Padre / Ocupación.
        - Lugar de trabajo / Teléfono.
        - Nombre de la Madre / Ocupación.
        - Lugar de trabajo / Teléfono.
        - Nombre del cónyuge / Edad.
        - Ocupación actual / Teléfono.
        - Referencia de hermanos o hijos que conforman la familia (Nombre – Edad).

- Remitido por.
- Paciente asignado a.
- Fecha propuesta.
- Horario.
- Tarifa establecida.

Los campos de la ficha personal se llenan de acuerdo al tipo de usuario y dependiendo del servicio que solicite o busque.

### 1.3.3 Módulo de Registro de Citas

Este módulo tiene como finalidad otorgar las citas a cada usuario, se basa en la disponibilidad de horarios y terapeutas.

### 1.3.4 Módulo de Informes

En este módulo es factible realizar el Informe de Cierre de caso, mismo que debe elaborarse para finalizar cada caso tratado por un terapeuta o un practicante.

Se propone que todo informe se realice en base a un solo formato, este formato debe estar acordado por el Director del CPsA, los profesionales de planta y los supervisores de cada área de especialización.

Luego de que se acuerde el formato, éste se ingresará al sistema.

### 1.3.5 Módulo de Facturación

En este Módulo trata sobre el manejo de ingresos y egresos de recursos financieros.

**Presupuesto**: El Centro de Psicología Aplicada cuenta con un presupuesto propio asignado por la Universidad que le permite solventar sus gastos de operación y algunos de inversión. Además, recibe de la Facultad de Psicología el rubro generado por el pago de aranceles de los estudiantes inscritos en Prácticas Pre-profesionales.

Todos los ingresos por pago de servicios van a la Tesorería de la Universidad.

**Pago:** Los servicios que ofrece el Centro de Psicología Aplicada, son pagados de acuerdo a tarifas existentes, fijadas de acuerdo a los niveles económicos de las personas, manteniéndose siempre de un 50 a un 70% más bajas que en la atención particular.
Se ha establecido en coordinación con la Dirección de Estudiantes, una tarifa diferenciada de acuerdo a la procedencia de la demanda.
Cuando se trate de Proyectos de Asesoría y Apoyo a la comunidad, se establecerán los costos siempre a niveles más bajos que los que se cobra en la práctica privada.
En la modalidad de convenios, los costos y la forma de pago se establecerán en consenso con cada una de las instituciones con las que se realicen convenios.

Los costos son revisados anualmente según la inflación y cambiados si las circunstancias lo ameritan.

### 1.3.6 Módulo de Reportes

Este módulo permite generar reportes útiles para la Dirección del CPsA y de quien esté interesado por conocer la gestión de la organización.

Los reportes que sean necesarios los propone el Director del CPsA, tomando en cuenta los aspectos que deben evaluarse constantemente como por ejemplo: satisfacción con el servicio de secretaría, satisfacción con el servicio de terapia brindado, número de casos atendidos por especialización, etc.

Para este fin, puede tomarse como base los indicadores de gestión asignados para el CPsA.

### 1.3.7 Módulo de Instituciones de referencia

Este módulo contiene una base de datos actualizada de instituciones que brinden servicios que estén fuera del alcance del centro y que sean útiles para remitir usuarios que no puedan ser atendidos por el CPsA.

Principalmente la utilidad de este módulo radica en que si un servicio no se presta en el centro, la secretaria o el becario estarán en capacidad de derivar el caso en mención a alguna otra institución en donde sí se preste el servicio requerido.

La construcción de esta base de datos está a cargo del Director del CPsA, de los profesionales de planta y de los docentes que puedan aportar con información útil para este fin. Debe contener la información que se considere necesaria, se propone lo siguiente:

- Nombre de la Institución.
- Especialización.
- Dirección.

- Teléfono.
- Persona de contacto.

### 1.3.8 Módulo de Soporte del Sistema

Este módulo permite al usuario conocer información acerca del Sistema.

Con la implantación de este sistema informático se automatizará el funcionamiento básico del CPsA, lo que se busca es mejorar la calidad de la prestación del servicio en general; además se logrará llegar a un ahorro de tiempo y de recursos materiales.

La intención es que la información se maneje tomando al sistema como la principal herramienta y para lograr este objetivo es importante ingresar todas las historias clínicas de los usuarios actuales y de los que han hecho uso del CPsA en los últimos 10 años (tiempo considerable para conservación de información).

El ingreso al sistema de las historias clínicas puede ser asignado a becarios de la facultad, es importante señalar que la información que se ingrese debe ser correcta, acorde a la historia clínica original.

Después de realizar esta actividad va a quedar una gran cantidad de papel sobrante y el Director del CPsA es el encargado de decidir qué hacer con eso; se propone reciclar todo ese papel y entregarlo a alguna empresa de reciclaje o a algún proyecto que demande este tipo de actividad. Así, se eliminaría el manejo manual de historias clínicas.

### 1.3.9 Módulo de Cubículos

El Centro reglamenta la relación del personal asistencial con quienes requieren atención a través de la distribución de salas, consignando y determinando el uso que se debe dar a los espacios, consultas, sala de juego, salas de supervisión, áreas de recepción etc. tal como aparecen consignados a continuación:

- La sala de recibo del Centro está reservada para los paciente, familiares de los pacientes que esperan ser atendidos; así como también para los practicantes, voluntarios que tienen que esperar a pacientes, hacer uso de los servicios de secretaría o esperar la iniciación de las reuniones de Supervisión de cada Área. Por lo tanto no es un espacio para actividades sociales, ni estudiantiles, ni un lugar en el que se vierta información de ninguna naturaleza. Pacientes y terapeutas necesitan ser respetados y respetar ese espacio.
- En el corredor interno, la circulación en el Centro y los accesos a los consultorios y salas de trabajo está reservado para el personal asistencial del Centro y los pacientes o familiares.

Sólo bajo autorización especifica el Director del Centro, las salas son usadas por estudiantes para la realización de otras tareas académicas.

- Los cubículos están reservados para tareas asistenciales (diagnósticas y terapéuticas exclusivamente) al igual que la Sala de juego, cuya reglamentación consta a continuación.
- El aula de supervisiones está reservada para tareas de supervisión y docente.
- Las salas de Juego y Cámara de Gesell para ser utilizadas necesitan de reservación previa.

**1.3.9.1 Ubicación:** El Centro de Psicología Aplicada, se encuentra ubicado en la Av. 12 de Octubre y Ladrón de Guevara, en la planta baja de la Casa Antigua de la Universidad Católica y cuenta con el siguiente espacio:

- Oficina de la Dirección.
- Secretaría y sala de espera.
- 11 consultorios para terapia individual o de pareja.
- 2 salas de reuniones.
- Una sala de clases.
- Dos salas de juegos y.
- Cámara de Gesel.

El mobiliario indispensable, 5 computadores, una impresora y 5 líneas telefónicas internas.

### 1.3.10 Módulo De Tarifas

Se propone este campo con la finalidad de que sea fácil encontrar la información acerca de tarifas establecidas para cada usuario. Se ingresa el número de cédula o el apellido del usuario y automáticamente el sistema arroja la mencionada información.

## 1.4 Características De La Herramienta De Desarrollo

### 1.4.0 Power Designer

Es una herramienta de modelamiento que permite visualizar, manipular y analizar metadatos de una manera más fácil, esto permite una arquitectura de información empresarial más efectiva.

Al tener una mejor visualización de la arquitectura de información, se pueden implementar más fácil los productos de software que se adapten a las necesidades de las empresas.

Fue diseñado por Sybase, funciona en Windows como una aplicación nativa y sobre Eclipse a través de un plugin. Entre las varias ventajas que tiene Power Designer podemos enumerar las siguientes:

- Generación automática de código para Java, C#, .NET, JSF, entre otros.
- Modelamiento para Data Warehouse.
- Modelamiento por diagramas UML.
- Modelamiento de datos para SGBD's Relacionales.

Power Designer se encuentra en la versión 15.3, liberada en Febrero de 2010, las novedades de esta versión son que incluye a todos los tipos de diagramas UML para modelamiento.

#### 1.4.0.1 Nuevas Características en Power Designer versión 15.3

- "Mejoras comunes
  - Finalizar las preferencias genéricas de pantalla.
  - ILM y XSM preferencias de visualización.
  - Creación fácil de diagramas.
    - Seleccione vecinos y crear diagrama de una vez.
    - Crear diagramas de Selección.
    - Crear modelo / diagrama en el menú contextual y agregarlo al diagrama correspondiente.
    - Crear diagramas relacionados más rápido.

- Mejorar Diálogos Grid
  - Exportar lista en Excel.
  - Imprimir listas.
  - Filtro parecido a Excel.
  - Células de selección múltiple en la matriz de dependencia.
- Plug-in de Visual Studio 2010.
- Integración Troux.
- Migración automática de scripts VBS / GTL.

- Simplificaciones de interfaz de usuario.
  - Renombrar definiciones de extensiones de modelo en extensiones.
  - Combinar contenido y formato en preferencias de visualización.Muestra una Vista individual en diálogos compuestos"[1]

Una vez que se ha dado una descripción del modelador, se procederá a hablar sobre el tipo de arquitectura y los diferentes diagramas UML que se utilizarán para el desarrollo del Sistema de Gestión.

## 1.4.1 Lenguaje De Modelado Unificado Uml

Definición de Modelo

- Un modelo es una simplificación de la realidad.
- Un modelo proporciona un anteproyecto del sistema.
- Es una abstracción de la realidad.
- Es una proyección a micro escala.

Debemos modelar para entender mejor el problema que estamos desarrollando

[1] ftp://ftp.sybase.cz/PD/153/readme.html, Características de Power Designer, Modificado: 24 de Febrero de 2010.

A través del modelado se:

- Ayuda a visualizar como es el sistema.
- Ayuda a especificar la conducta de la estructura del sistema.
- Da una guía base para la construcción del sistema.
- Documenta las decisiones que debemos tomar.

**El Lenguaje de Modelado Unificado (UML):** es un lenguaje estándar para la escritura de proyectos de software. El UML puede ser usado para visualizar, especificar, construir y documentar los componentes de un sistema de software.
El UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema.

**La estructura estática:** define los tipos de objetos importantes para un sistema y para su implementación, así como las relaciones entre los objetos.
**El comportamiento dinámico:** define la historia de los objetos en el tiempo y la comunicación entre objetos para cumplir sus objetivos.
El modelar un sistema desde varios puntos de vista, separados pero relacionados, permite entenderlo para diferentes propósitos.
Un lenguaje de modelado es un lenguaje cuyo vocabulario y reglas se enfocan en la representación conceptual y física de un sistema. El modelado permite entender un sistema.
El vocabulario y las reglas de un lenguaje tal como UML nos dicen como crear y leer modelos bien formados, pero no nos dicen que modelos crear y cuando crearlos. Un proceso bien definido guía en decidir que componentes producir, que actividades y que trabajadores usar para crearlos y manejarlos y como usar esos componentes para validar y controlar el proyecto como tal.

#### 1.4.1.0 Para Visualización:

UML no es un lenguaje de programación.
El UML tiene una semántica bien definida, detrás de cada símbolo de la notación. Un desarrollador puede escribir un modelo en el UML y otro desarrollador aun con otra herramienta puede interpretar ese modelo no ambiguamente.
El UML como tal, es un lenguaje gráfico. Algunas cosas son mejor modelados textualmente, otros son mejor modelados gráficamente.

#### 1.4.1.1 Para Especificación:

En este contexto, especificación significa modelos de construcción que son *precisos, no ambiguos y completos.*

El UML direcciona la especificación de todas las decisiones importantes de análisis, diseño e implementación que deben ser hechas para el desarrollo y estructuración de un sistema de software.

#### 1.4.1.2 Para Construcción:

El UML no es lenguaje de programación visual, pero su modelo puede ser directamente conectado a una variedad de lenguajes de programación, a tablas en una base de datos relacional o el almacenamiento persistente de una base de datos orientada a objetos. Este mapeo permite avances de ingeniería: La generación de un código dentro de un lenguaje de programación. La inversa también es posible. La ingeniería inversa requiere de herramientas de soporte con intervención humana.

#### 1.4.1.3 Para Documentación:

Una buena organización de software produce todos los tipos de componentes para el código ejecutable. Estos componentes incluyen:

- Requerimientos.
- Arquitectura.
- Diseño.
- Código fuente.
- Planes del proyecto.
- Pruebas.
- Prototipos.

Tales componentes no son solo los entregables de un proyecto, son también indispensables para controlar, validar y comunicar un sistema antes de su desarrollo y después de su estructuración.

Para entender UML es necesario formar un modelo conceptual del lenguaje y esto requiere aprender tres elementos:

- Los bloques de construcción básicos
- las reglas que dictan como esos bloques pueden ser combinados
- Algunos mecanismos que se aplican todo el tiempo en UML.

#### 1.4.1.4 Los bloques de construcción básicos

El vocabulario de UML que comprende tres tipos de bloques de construcción:

- Elementos.
- Relaciones.
- Diagramas.

Los elementos son las abstracciones que son ciudadanas de primera clase en un modelo, las relaciones enlazan estos elementos entre sí.

**Elementos:**

Existen 4 clases que constan de:

- Estructurales.
- Comportamiento.
- Agrupamiento.
- Anotacionales.

**1.4.1.5 Estructurales:** son los sustantivos de los modelos de UML. Estos son en la mayoría partes estáticas de un modelo, representando elementos que pueden ser conceptuales o físicos. Hay siete tipos de elementos estructurales.

**Una clase** es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semánticas. Una clase lleva a cabo una o más interfaces.

**Una interfaz** es una colección de operaciones que especifican un servicio de una clase o componente. Así, una interfaz describe el desempeño externamente visible de ese objeto. Una interfaz puede representar el funcionamiento completo de una clase o componente o solo una parte de ese desempeño.

**Una colaboración** es una descripción de una colección de objetos que interactúan para implementar un cierto comportamiento dentro de un contexto. Describe una sociedad objetos cooperantes unidos para realizar cierto propósito.

**Un caso de uso** es una descripción de un conjunto de secuencias de acciones que un sistema desempeña para permitir un resultado de valor observable para un actor particular.

**Una clase activa** es una clase cuyos objetos reconocen uno o más procesos o hilos y por lo tanto pueden iniciar una actividad de control. Es semejante a una clase excepto que sus objetos representan elementos cuya función es concurrente con otros elementos.

**Un componente** es un una parte física y remplazable de un sistema que conforma y proporciona la realización de un conjunto de interfaces. Además de componentes del procesos de desarrollo, como los archivos de código fuente.

**Un nodo** es un elemento físico que existe al tiempo de ejecución y representa un recurso computacional, generalmente tiene al menos una memoria y frecuentemente capacidad de procesamiento. Un conjunto de componentes puede residir en un nodo y puede también emigrar de un nodo a otro.

**1.4.1.6 Comportamiento**: Son las partes dinámicas de los modelos UML. Estos son los verbos de un modelo que representan la función sobre tiempo y espacio. De hecho, hay dos tipos principales de elementos de comportamiento.

**Una iteracción** es una función que comprende un conjunto de intercambios de mensajes entre un conjunto de objetos con un contexto particular para lograr un propósito específico. La función de una asociación de objetos o de una operación individual puede ser especificada con una interacción.

**Una máquina de estado** es una función que especifica la secuencia de estados de un objeto o una interacción dada durante su tiempo de vida en respuesta a eventos, junto con las respuestas de esos eventos.

**1.4.1.7 Agrupamiento:** Son las partes de organización de los modelos UML. Estas son cajas dentro de las cuales un modelo puede ser descompuesto. Hay un tipo principal de elementos de agrupamiento nombrados paquetes.

**Un paquete** es un mecanismo de propósito general para la organización de elementos en grupos. Los elementos estructurales, funcionales y aun los de agrupación pueden estar situados dentro de un paquete. A diferencia de los componentes un paquete es puramente conceptual.

**1.4.1.8 Anotacionales:** Son las partes explicativas de los modelos de UML. Son los comentarios que se pueden aplicar para describir, iluminar y remarcar algunos elementos de un modelo. Hay un tipo principal de Elementos anotacionales llamado nota.

**Una nota** es simplemente un símbolo para representar las limitaciones y comentarios asociados a un elemento o una colección de elementos.

Normalmente, se verán las partes estáticas del sistema usando uno de los cuatro diagramas siguientes:

- Diagramas de Clase.
- Diagrama de Objetos.
- Diagrama de Componentes.
- Diagrama de Implementación.

Para ver las partes dinámicas de un sistema, se usarán los siguientes diagramas:

- Diagrama de Casos de Uso.
- Diagrama de Secuencia.
- Diagrama de Colaboración.
- Diagrama de Estado.
- Diagrama de Actividad.

#### 1.4.1.9 Caso de Uso

Los casos de uso se basan en la descripción de escenarios en los cuales los usuarios interactúan con el sistema que se ha definido para alcanzar un objetivo o para cumplir una tarea particular
Los casos de uso ponen énfasis en los objetivos del usuario, mas no en las características del sistema

**Beneficios**

- Comunicación.
- Identificación.
- Verificación.

#### 1.4.1.9.1 Elementos

**Un actor** representa un conjunto coherente de roles que los usuarios de los casos de uso juegan al interactuar con tales casos. Los actores se representan como figuras delgadas. Se pueden definir

clases generales de actores (cliente) y especializarlas (cliente comercial) usando relaciones de generalización.

**Un caso de uso** es una descripción de un conjunto de secuencias de acciones, incluyendo variantes que un sistema ejecuta para producir un resultado importante para un actor.
Los nombres de casos de uso son frases con verbos activos pequeños, éstos nombran el comportamiento encontrado en el vocabulario del sistema que se está modelando.
Los casos de uso pueden asociarse entre si dependiendo el caso:
Una relación extends se usa cuando se tiene un caso de uso que es similar a otro, pero que hace un poco más.
Una relación include ocurre cuando se tiene una porción de comportamiento que es similar en más de un caso de uso y no se quiere copiar la descripción de tal conducta.


**Título** Figura 1.0.1- Representación de un caso de uso y de un actor

#### 1.4.1.10 Diagrama de Clase

Las clases son el bloque constructor más importante de cualquier sistema orientado a objetos; es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica.
El diagrama de clase describe los tipos de objetos que hay en el sistema y las diversas clases de relaciones estáticas que existen entre ellos.

##### 1.4.1.10.1 Elementos

**Nombres.** Cada clase debe tener un nombre que le distinga de otras clases. Normalmente el nombre de la clase se pone en mayúscula la primera letra.

**Atributos.** Es una propiedad de una clase identificada con un nombre. Una clase puede tener cualquier cantidad de atributos. Normalmente, se pone en mayúsculas la primera letra de cada palabra de un atributo, excepto la primera.

**Operaciones.** Es la implementación de un servicio que puede ser requerido a cualquier objeto de la clase para que muestre un comportamiento. Se pone en mayúsculas la primera letra de cada palabra en el nombre de una operación excepto la primera letra.

#### 1.4.1.10.2 Asociaciones

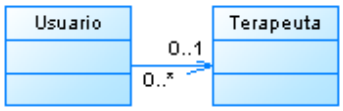
**Las asociaciones** representan relaciones entre instancias de clases.
Cuando se añade flechas a la asociación, se habla de Navegabilidad. Si existe una navegabilidad en una sola dirección, a la asociación se le llama asociación unidireccional.

**Subtipos**

Una generalización es una relación entre un elemento general y un caso más específico de ese elemento.

**Agregación**

Una agregación es un tipo especial de asociación y se especifica añadiendo a una asociación normal un rombo vacío en la parte del todo. Es útil cuando se desea representar una cosa grande que consta de elementos más pequeños.




**Título** Figura 1.0.2- Representación de un diagrama de clases.

**Modelar Aspectos Dinámicos Del Sistema**

- Diagramas de Secuencia.
- Diagramas de Colaboración.
- Diagramas de estado.

**Diagramas de Iteración**

- Diagramas de Secuencia.
- Diagramas de Colaboración.

Para modelar flujos de control por orden de tiempo se usan diagramas de secuencia. Se hace énfasis en el paso de mensajes, en cómo se desenvuelven sobre el tiempo, lo cual es una manera útil para visualizar el comportamiento dinámico en el contexto de un escenario de un caso de uso.

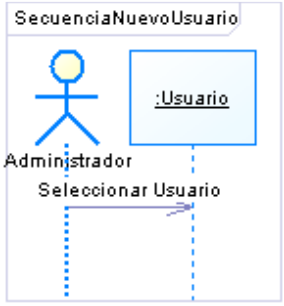
Para modelar flujos de control por organización se usan diagramas de colaboración. Se hace énfasis en las relaciones estructurales entre las instancias dentro de la interacción y junto con los mensajes que pueden ser pasados.

### 1.4.1.11 Diagramas de Secuencia

Los objetos que inicia la interacción se alinean a la izquierda. De manera incremental, los objetos subordinados se van colocando a la derecha. Los mensajes que esos objetos envían o reciben se ubican a lo largo del eje Y conforme arriban en el transcurso del tiempo.

**Tienen la línea de vida del objeto.** Esto es, la línea entrecortada vertical que representa la existencia de un objeto en el transcurso del tiempo. Los objetos que se alinean al tope del diagrama son aquellos que existen durante toda la interacción. Las líneas de vida de los objetos empiezan y terminan con la recepción de los mensajes estereotipados create y destroy.

**Contiene el foco de control.** Se representa por un largo y delgado rectángulo que muestra el período de tiempo durante el cual un objeto realiza una acción en forma directa o a través de un proceso subordinado. El tope del rectángulo se alinea con el inicio de la acción y el fondo con la finalización.




**Título** Figura 1.0.3- Representación de un diagrama de secuencia.

#### 1.4.1.12 Diagramas de Colaboración

Enfatiza la organización estructural de los objetos que envían y reciben mensajes. Gráficamente, es una colección de vértices y arcos.

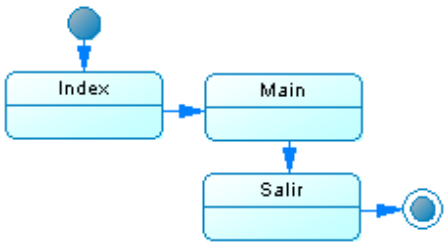
**Se presenta un camino.** Para indicar como un objeto se enlaza con otro, se puede marcar al final del enlace con ciertos indicadores como: local, parameter, global y self.

**Se tiene una secuencia numérica.** Para indicar el orden de los mensajes en el tiempo, se añade un prefijo numérico al mensaje.

#### 1.4.1.13 Diagramas de Estados

El comportamiento de una sociedad de objetos que colaboran puede ser modelado mediante interacción.

El comportamiento de un objeto individual, puede ser modelado mediante una máquina de estados. Una máquina de estados es un comportamiento que especifica las secuencias de estados por las que pasa un objeto durante su vida, en respuesta a eventos, junto con sus respuestas a esos eventos.




**Título** Figura 1.0.4- Representación de un diagrama de estado.

#### 1.4.1.13 Paquete

Los métodos estructurados se valieron de la descomposición funcional, en la cual el sistema era visto como una función que podía ser dividido en subfunciones. Las funciones eran como los casos de uso en un sistema orientado a objetos.

En la OO ha desaparecido esta separación entre el proceso y los datos, y la descomposición funcional.

La idea de un paquete se puede aplicar a cualquier elemento de un modelo, no solo a las clases. El término diagrama de paquetes sirve para indicar un diagrama que muestre los paquetes de clase y las dependencias entre ellos.

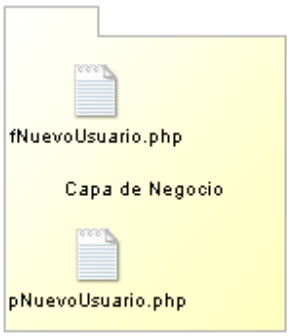
**Dependencias**

Existe una dependencia entre dos elementos si los cambios a la definición de un elemento pueden causar cambios al otro.

**Generalización**

Con los paquetes se puede aplicar generalización. Esto significa que el paquete específico debe conformarse a la interfaz del paquete general.

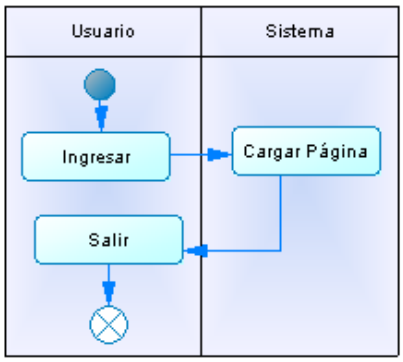
La generalización implica una dependencia del subtipo al supertipo. En un sistema ya existente, las dependencias se pueden deducir observando las clases. Un paso útil inicial es dividir las clases en paquetes y analizar las dependencias entre estos últimos.




**Título** Figura 1.0.5- Representación de un diagrama de paquetes.

**1.4.1.14 Diagrama de Actividad**

Un diagrama de actividades es un diagrama que muestra el flujo de control entre actividades a lo largo del tiempo, puede manejar procesos paralelos. Un diagrama de actividades podemos seleccionar el orden en el que se harán las cosas, tenemos una secuenciación.




**Título** Figura 1.0.6- Representación de un diagrama de actividad.

**División y Unión**

La división representa la separación de un flujo de control sencillo en dos o más flujos de control concurrentes.

La unión representa la sincronización de dos o más flujos de control concurrentes. En la unión los flujos concurrentes se sincronizan y continúan en un solo flujo de control.

**1.4.1.14.1 Elementos**

**Los carriles** permiten dividir los estados de un diagrama de actividad en grupos donde cada uno representa la parte de la organización responsable de esas actividades o subactividades y puede eventualmente ser implementada por uno o más objetos.

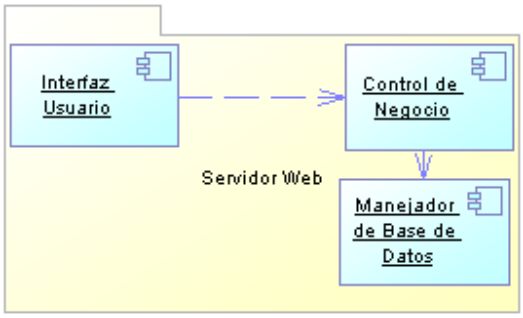
**1.4.1.15 Diagramas de Despliegue**

Los diagramas de despliegue se utilizan para modelar la vista de despliegue estática de un sistema.

Cada nodo de un diagrama de emplazamiento representa alguna clase de unidad cómputo; en la mayoría de los casos se trata de una pieza de hardware.

Las conexiones entre nodos muestran las rutas de comunicación a través de las cuales interactuará el sistema.

Las dependencias entre los componentes deben ser las mismas que las dependencias de paquetes. Estas dependencias muestran como se comunican los componentes con otros componentes.




**Título** Figura 1.0.7- Representación de un diagrama de despliegue.

## 1.4.2 Programación Orientada a Objetos

"La programación orientada utiliza objetos y sus interacciones en la cual de forma ordenada se pueda diseñar aplicaciones y programas informáticos. Está basada en técnicas como herencia, abstracción, polimorfismo y encapsulamiento. Se empezó a utilizar en la década de los años 1990. Hoy en día la mayoría de lenguajes soportan orientación a objetos.

La programación orientada a objetos es un paradigma que utiliza objetos como elementos fundamentales en la construcción de la solución. Surge en los años 70. Un objeto es una abstracción de algún hecho o ente del mundo real que tiene atributos que representan sus características o propiedades y métodos que representan su comportamiento o acciones que realizan. Todas las propiedades y métodos comunes a los objetos se encapsulan o se agrupan en clases. Una clase es una plantilla o un prototipo para crear objetos, por eso se dice que los objetos son instancias de clases.

**Introducción**

Los objetos son entidades que tienen un determinado estado, comportamiento e identidad:

- El estado está compuesto de datos, en lo cual se habrán asignado valores concretos a varios atributos.
- El comportamiento son los métodos a los que sabe responder el objeto en el cual interactúan con él.
- La identidad es la propiedad del objeto para poder diferenciar de las dos anteriores es su identificador.

Un objeto contiene información que permite definirlo frente a otros objetos que pertenecen a otra clase, o también contra objetos de una misma clase, ya que cada clase tienes sus propios atributos. Los objetos disponen mecanismos de interacción conocidos como métodos, que su función es la comunicación entre ellos. Esta comunicación es el cambio de estado entre objetos. Cado uno tratado como unidades indivisibles, en las que no se separe el estado y el comportamiento.

POO no es programación estructurada tradicional, en la que los datos y métodos están separados sin relación, en la cual se intenta lograr el procesamiento de datos de entradas para obtener datos de salida. La programación estructurada ayuda al programador sobre los términos de procedimiento, y en segundo lugar las estructuras de datos que los procedimientos manejan. En esta programación se escriben funciones que procesan datos. Los programadores que utilizan POO, por otro lado tienen que definir objetos para enviar mensajes entre ellos solicitándoles que realicen sus métodos por sí mismos.

Hoy por hoy en la industria y el ámbito académico es la POO. La orientación a objetos da mejoras de mayor alcance en la forma de diseño, desarrollo y mantenimiento del software ofreciendo soluciones a largo plazo hacia los problemas que existen desde el comienzo del desarrollo de software entre estos la falta de portabilidad del código y reusabilidad, código que es difícil de modificar, ciclos de desarrollo largos y técnicas de codificación no intuitivas.

Como elemento raíz de POO son los objetos. Podemos definir los objetos como conjunto de datos de diferentes tipos y estructura que posee y forma parte de una organización.

En su definición se especifica varias propiedades importantes. En primero lugar el objeto no es un dato simple, contiene en su interior cierto número de componentes bien estructurados. En segundo lugar, cada objeto pertenece a una clase en la cual no es un ente aislado, sino que parte de una organización jerárquica. ”[2]

[2] RUMBAUGH J., (2004), “Object-Oriented Modeling and Design”, Prentice- Hall

**Estructura de un objeto**

- Relaciones.
- Propiedades.
- Métodos.

**Las relaciones** permiten que el objeto se inserte en la organización y están formadas esencialmente por punteros a otros objetos.

**Las propiedades** hacen distinción de otros objetos que forman parte de la misma organización en el cual tiene valores que dependen de la propiedad que se trate. La propiedades del objeto son heredadas a sus descendientes en la organización dependiendo de como se las defina.

**Los métodos** son las operaciones que pueden realizarse sobre el objeto, que normalmente estarán incorporados en forma de programas que el objeto es capaz de ejecutar y que también pone a disposición de sus descendientes a través de la herencia.

**Origen**

"Los conceptos de la programación orientada a objetos tienen origen en Simula 67, un lenguaje diseñado para hacer simulaciones, creado por Ole-Johan Dahl y Kristen Nygaard del Centro de Cómputo Noruego en Oslo. En este centro, se trabajaba en simulaciones de naves, que fueron confundidas por la explosión combinatoria de cómo las diversas cualidades de diferentes naves podían afectar unas a las otras. La idea surgió al agrupar los diversos tipos de naves en diversas clases de objetos, siendo responsable cada clase de objetos de definir sus propios datos y comportamientos. Fueron refinados más tarde en Smalltalk, desarrollado en Simula en Xerox PARC (cuya primera versión fue escrita sobre Basic) pero diseñado para ser un sistema completamente dinámico en el cual los objetos se podrían crear y modificar "sobre la marcha" (en tiempo de ejecución) en lugar de tener un sistema basado en programas estáticos.

La programación orientada a objetos se fue convirtiendo en el estilo de programación dominante a mediados de los años ochenta, en gran parte debido a la influencia de C++, una extensión del lenguaje. Su dominación fue consolidada gracias al auge de las Interfaces gráficas de usuario, para las cuales la programación orientada a objetos está particularmente bien adaptada. En este caso, se habla también de programación dirigida por eventos.

Las características de orientación a objetos fueron agregadas a muchos lenguajes existentes durante ese tiempo, incluyendo Ada, BASIC, Lisp, Pascal, entre otros. La adición de estas características a los lenguajes que no fueron diseñados inicialmente para ellas condujo a menudo a problemas de compatibilidad y en la capacidad de mantenimiento del código. Los lenguajes orientados a objetos "puros", por su parte, carecían de las características de las cuales muchos programadores habían venido a depender. Para saltar este obstáculo, se hicieron muchas tentativas para crear nuevos lenguajes basados en métodos orientados a objetos, pero permitiendo algunas características imperativas de maneras "seguras". El Eiffel de Bertrand Meyer fue un temprano y moderadamente acertado lenguaje con esos objetivos pero ahora ha sido esencialmente remplazado por Java, en gran parte debido a la aparición de Internet, y a la implementación de la máquina virtual de Java en la mayoría de navegadores. PHP en su versión 5 se ha modificado, soporta una orientación completa a objetos, cumpliendo todas las características propias de la orientación a objetos. "[3]

**Conceptos Fundamentales**

En los últimos años la POO siendo una forma de programar trata de encontrar una solución a estos problemas por el cual introduce nuevos conceptos, que superan y amplían conceptos antiguos conocidos. Entre ellos destacan los siguientes:

- "**Clase:** Es la definición de las propiedades y comportamiento de un objeto. Cuando se instancia un objeto se realiza una lectura de estas definiciones y de la creación de un objeto a partir de ellas
- **Herencia:** En esta operación una clase hijo hereda de una clase padre con la cual hereda atributos y métodos, como si estos atributos y métodos fueran definidos en la clase hijo. Por lo tanto puede utilizar los mismos métodos y variables públicas declaradas en la clase padre. Lo componentes privados también se heredan, pero como no pertenecen a la clase, no se los puede utilizar y pueden ser accesados solo por métodos públicos.
- **Objeto:** Es la instancia de una clase, que contiene un conjunto de propiedades y métodos los mismos que reaccionan a eventos. Se corresponde con objetos reales del mundo que nos rodea.

[3] http://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos, Programación orientada a objetos, Última Modificación: 30 Diciembre del 2011.

- **Método:** Es código asociado a un objeto, por el cual necesita de un mensaje para su ejecución. Desde el punto de vista del comportamiento, es lo que el objeto puede realizar. Un método puede producir un cambio a las propiedades de un objeto, o genera un evento como nuevo mensaje para otra instancia de una clase del sistema.
- **Evento:** Es un suceso en el sistema. El sistema maneja el evento por medio del envío del mensaje del objeto pertinente. También se puede definir evento como la reacción de la acción de que un objeto genera.
- **Mensaje:** Es la comunicación dirigida al objeto, que le ordena que ejecute uno de los métodos definidos dentro de él, recibiendo parámetros asociados al evento que lo generó..
- **Atributo:** Es el contenedor de un tipo de datos asociados a un objeto, dependiendo de la forma de la definición hace los datos visibles desde fuera del objeto y cuyo valor puede ser alterado por un método.
- **Estado interno:** Es una variable que se declara privada, que puede ser únicamente accedida y alterada por un método del objeto, y que se utiliza para indicar distintas situaciones posibles para el objeto. No es visible al programador que maneja una instancia de la clase.
- **Componentes de un objeto:** Atributos, identidad, relaciones y métodos.
- **Identificación de un objeto:** Un objeto se representa por medio de una tabla o entidad que esté compuesta por sus atributos y funciones correspondientes.

**Características**

Existe un acuerdo acerca de qué características contempla la "orientación a objetos", las características siguientes son las más importantes:

- **Abstracción:** Denota las características esenciales de un objeto, donde se capturan sus comportamientos. Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y comunicarse con otros objetos en el sistema sin revelar cómo se implementan estas características. Los procesos, las funciones o los métodos pueden también ser abstraídos y cuando lo están, una variedad de técnicas son requeridas para ampliar una abstracción. El proceso de abstracción permite seleccionar las características relevantes dentro de un conjunto e identificar comportamientos comunes para definir nuevos tipos de entidades en el mundo real. La abstracción es clave en el proceso de análisis y diseño orientado a objetos, ya que mediante ella podemos llegar a armar un conjunto de clases que permitan modelar la realidad o el problema que se quiere atacar.

- **Encapsulamiento:** Se considera que todos los elementos pertenezcan a una misma clase. Este concepto no es lo mismo que el principio de ocultación, con su principal diferencia que se utiliza un conjunto.

- **Modularidad:** Es la propiedad que permite subdividir una aplicación en partes menores, cada módulo debe ser independiente como sea posible de la aplicación y del resto de partes. Estos módulos se compilan por separado, pero se interconectan con otros módulos. Al igual que la encapsulación, los lenguajes soportan modularidad en diferentes formas.

- **Principio de ocultación:** Cada objeto está aislado del exterior, es un módulo natural, y cada tipo de objeto expone una interfaz a otros objetos que específica cómo pueden interactuar con los objetos de la clase. El aislamiento protege a las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas, solamente los propios métodos internos del objeto pueden acceder a su estado. Esto asegura que otros objetos no pueden cambiar el estado interno de un objeto de maneras inesperadas, eliminando efectos secundarios e interacciones inesperadas. Algunos lenguajes relajan esto, permitiendo un acceso directo a los datos internos del objeto de una manera controlada y limitando el grado de abstracción. La aplicación entera se reduce a un agregado o rompecabezas de objetos.
- **Polimorfismo:** La manera en como un objeto se comporta, creando varios constructores pueden compartir el mismo nombre, en el momento de llamarlos dependiendo del número de argumentos se utilizará el comportamiento correspondiente al objeto que se use.
- **Herencia:** Las clases se relacionan entre sí, formando jerarquía de clasificación. Los objetos heredan propiedades y métodos de las clases que pertenecen. La herencia facilita el polimorfismo y encapsulamiento permitiendo a los objetos ser definidos y creados para ser especializados de objetos prexistentes. Estos comparten su comportamiento para que no exista una re implementación ahorrando espacio en el código y tiempo. Esto se hace habitualmente agrupando los objetos en clases que reflejan un comportamiento común. Si un objeto hereda de muchas clases es una herencia múltiple.
- **Recolección de basura:** Es una técnica por la cual el entorno de objetos se encarga de destruir automáticamente, para desvincular dicho objeto de memoria. Esto significa que el programador no debe preocuparse por la liberación de memoria, ya que el entrono asignará memoria al momento de crear uno o liberará cuando no lo esté usando. En la mayoría de los lenguajes híbridos que

soportan POO como C++, esta característica no existe y hay que quitar de memoria manualmente al objeto que se deje de utilizar. ”[4]

#### 1.4.2.0 Ciclo de Vida

Para el desarrollo de este sistema se ha seleccionado el ciclo de vida en cascada ya que este ciclo mantiene retroalimentación en cada fase. Con su estructura si se encuentra cualquier error en las pruebas se lo puede detectar y saber exactamente en cual etapa anterior se encuentra. Este ciclo de vida de software fue propuesto por Royce 1970, en la cual un número de 90% de desarrolladores optar por esta forma de desarrollo. Las fases del modelo son las siguientes.

- **Análisis:** Es la primera fase del modelo en cascada, aquí se analizan las necesidades del cliente y determina que objetivos el software debe cumplir. Es importante definir requerimientos en esta fase ya que esto dependerá el desarrollo del producto de software, produciendo el Documento de Requerimientos de Software.
- **Diseño:** contiene descripción de la estructura relacional globlar del sistema y la especificación que debe cumplir en cada una de sus partes. Este tiene dos diseños
    - Diseño General: se toma en cuenta requisitos generales de la arquitectura de la aplicación.
    - Diseño detallado: se realiza una definición precisa de cada subconjunto de los módulos del diseño general.
- **Implementación:** En base al diseño se realizan los diferentes módulos del sistema implementando el código fuente. Con el lenguaje web de programación se crean las bibliotecas y componentes o funciones reutilizables para hacer la programación un proceso más rápido.
- **Pruebas:** Finalizando la programación se ensamblan para componer el sistema y se comprueba que funcionen correctamente antes de ser puesto en la sección de implantación.
- **Mantenimiento:** Una vez el software implantado, se debe realizar un mantenimiento continuo del producto por medio de versionamiento.

**Ventajas de Usar este Ciclo de Vida:**

- Se tiene todo bien organizado y no se mezclan las fases.

[4] http://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos, Programación orientada a objetos, Última modificación 30 Diciembre del 2011 a las 04

- Es perfecto para proyectos rígidos, y además donde se especifiquen muy bien los requerimientos y se conozca la herramienta a utilizar.
- Al llevar las fases de la manera indicada, es más fácil comprender como está estructurado un producto de software sólo con revisar dicha documentación.
- La calidad del producto terminado es alta.

.

## 1.4.3 PHP

"**PHP** es un acrónimo recursivo que significa PHP Hypertext Pre-processor. Fue creado por Rasmus Lerdorf en 1994; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la Free Software Foundation considera esta licencia como software libre.

PHP es un lenguaje de programación diseñado originalmente para creación de páginas web dinámicas. Se lo utilizada del lado del servidor pero también puede ser usado desde una interfaz de línea de comandos o de otros tipos de aplicaciones con interfaz gráfica usando bibliotecas QT o GTK+.

Puede ser utilizado en la mayoría de los servidores web y en la mayoría de sistemas operativos. El lenguaje PHP se encuentra instalado en alrededor de 20 millones de sitios web y en un millón de servidores, el número de sitios de PHP ha compartido algo de su dominio con otro lenguajes no tan robustos desde agosto del 2005. El sitio de Wikipedia desarrolla en PHP. Es también Apache el módulo más popular como servidor web.

Cuando el cliente hace una petición al servidor web para desplegar una página, el servidor ejecuta el intérprete de PHP. Este procesa el script solicitado que generará el contenido de manera dinámica. Este resultado es enviado por el intérprete al servidor, quien a su vez envía al cliente. Usando extensiones también se puede generar archivos PDF, Flash e imágenes en diferentes formatos. "[5]

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite.

[5] http://es.wikipedia.org/wiki/PHP, Utilidades y Características de PHP, Última Modificación 13 Enero 2012 a las 21:51.

PHP es ejecutado en la mayoría de SO en lo cual puede interactuar con los servidores web más populares ya que existe la versión CGI, módulo para Apache e ISAPI.

El 13 de julio de 2004, fue lanzado PHP 5, utilizando el motor Zend Engine 2.0. Incluye todas las ventajas que provee el nuevo Zend Engine 2 como:

- Mejor soporte para la programación orientada a objetos, que en versiones anteriores era extremadamente rudimentario.
- Mejoras de rendimiento.
- Mejor soporte para MySQL con extensión mysqli.
- Mejor soporte a XML (XPath, DOM, etc.).
- Soporte nativo para SQLite.
- Soporte integrado para SOAP.
- Iteradores de datos.
- Manejo de excepciones.
- Mejoras con la implementación con Oracle.

**Características**

- "Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- El código fuente escrito en PHP es invisible al navegador web y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad conMySQL y PostgreSQL.
- Capacidad de expandir su potencial utilizando módulos.
- Posee una amplia documentación en su sitio web oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.

- Permite aplicar técnicas de programación orientada a objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Tiene manejo de excepciones.
- Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar, aun haciéndolo, el programador puede aplicar en su trabajo cualquier técnica de programación o de desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del patrón de diseño Modelo Vista Controlador ,que permiten separar el tratamiento y acceso a los datos, la lógica de control y la interfaz de usuario en tres componentes independientes. "[6]

**1.4.3.0 Comparación de Lenguajes de Programación WEB**

**1.4.3.0.0 PHP**

Al ser un lenguaje libre dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas:

- Soporte para una gran cantidad de base de datos: MysSQL, PostgreSQL, Oracle, MS SQL Server, Sybase Msql.
- Integración con varias bibliotecas externas, permite generar documentos en PDF hasta analizar código XML.
- Ofrece una solución y universal para las paginaciones dinámicas del web de fácil programación.
- Perceptible mas fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
- Con PHP se puede hacer cualquier cosa que podemos realizar con un script CGI, como el procesamiento de información en formularios, foros de discusión, manipulación de cookies y páginas dinámicas.

**Ventajas**

- Muy fácil de aprender.
- Se caracteriza por ser un lenguaje muy rápido.

[6] http://es.wikipedia.org/wiki/PHP, Utilidades y Características de PHP, Última Modificación 13 Enero 2012 a las 21:51.

- Soporta en cierta medida la orientación a objeto. Clases y herencia.
- Es un lenguaje multiplataforma: Linux, Windows, entre otros.
- Capacidad de conexión con la mayoría de los manejadores de base de datos: MysSQL,
- PostgreSQL, Oracle, MS SQL Server, entre otras.
- Capacidad de expandir su potencial utilizando módulos.
- Posee documentación en su página oficial www.php.net la cual incluye descripción y ejemplos de cada una de sus funciones.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Incluye gran cantidad de funciones.
- No requiere definición de tipos de variables ni manejo detallado del bajo nivel

#### 1.4.3.0.1 ASP .NET

- Combina la productividad del desarrollador sin precedentes con un rendimiento, fiabilidad y la implementación. ASP.NET rediseña toda el proceso.
- Fácil modelo de programación, hace que la creación de aplicaciones del mundo real web mucho más fácil. Controles de servidor ASP.NET permite una estilo de HTML, como de la programación declarativa que le permite crear grandes páginas con menos código que con ASP clásico.
- Las flexibles opciones de idioma, le permite aprovechar sus habilidades actuales lenguaje de programación.
- Gran herramienta de soporte, usando cualquier editor de texto, incluso el bloc de notas, sin embargo la extensión del framework de Visual Studio ofrece más de 4500 clases que encapsulan la funcionalidad como XML, acceso a datos, carga de archivos, expresiones de registro, generación de imágenes, control de rendimiento y operaciones de registro.
- Compilación de ejecución, es mucho mas rápido, detecta automáticamente los cambios, de forma dinámica compila los archivos si es necesario, y almacena los resultados compilados para la reutilización de las solicitudes posteriores.

**Ventajas**

- Se encarga de detectar el tipo de navegador utilizado por el cliente a la hora de realizar una petición al servidor y en consecuencia, determina la versión HTML que éste soporta.
- Es liviano.
- Se puede utilizar en cualquier computadora que esté conectada a la red que tenga instalado un navegador.

- Es muy fácil de programar y tiene muchas utilidades que con una breve línea de aprendizaje pueden ser modificadas a su gusto.
- Tiene la facilidad de conectarse con la base de datos, que hace que sea muy fácil.
- Permite a los proveedores de Web ofrecer aplicaciones de negocios interactivos y no simplemente meros contenidos publicables.
- Una de las limitaciones en el desarrollo con ASP es que con el tradicional utilizamos lenguajes de scripting no tipeados como VSBcrip o JScrip. Podemos instalar otros motores scripting que impongan verificación de tipos.

**1.4.3.0.2 JSP**

- Lenguaje simple, Java posee una curva de aprendizaje muy rápida.
- Orientado a objetos, Java fue diseñado como un lenguaje orientado a objetos desde el principio. Los objetos agrupan en estructuras encapsuladas tanto sus datos como los métodos que manipulan esos datos.
- Distribuido, Java proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets y establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.
- Interpretado y compilado a la vez, Java es compilado, en la medida en que su código fuente se transforma en una especie de código máquina, semejantes a las instrucciones de ensamblador.
- Dada la naturaleza distribuida de Java, donde los applets se bajan desde cualquier punto de la Red, la seguridad se impuso como una necesidad de vital importancia.
- El lenguaje Java y su sistema de ejecución en tiempo real son dinámicos en la fase de enlazado. Las clases sólo se enlazan a medida que son necesitadas. Se pueden enlazar nuevos módulos de código bajo demanda, procedente de fuentes muy variadas, incluso desde la Red.

**Ventajas**

- El JDK es una herramienta libre de licencias, creada por Sun. Está respaldado por un gran número de proveedores.
- Existe soporte dado por Sun.
- Debido a que existen diferentes productos de Java, hay más de un proveedor de servicios.
- Sun saca al mercado cada 6 meses una nueva versión del JDK.

- Es independiente de la plataforma de desarrollo.
- Existen dentro de su librería clases gráficas como awt y swing, las cuales permiten crear objetos gráficos comunes altamente configurables y con una arquitectura independiente de la plataforma.
- Java permite a los desarrolladores aprovechar la flexibilidad de la Programación Orientada a Objetos en el diseño de sus aplicaciones.
- Se puede acceder a bases de datos fácilmente con JDBC, independientemente de la plataforma utilizada. El manejo de las bases de datos es uniforme, es decir transparente y simple.

**Tabla Comparativa JSP, PHP y ASP .NET**

| **DESCRIPCIÓN** | **JSP** | **PHP** | **ASP** | **ASP.NET** |
|---|---|---|---|---|
| Soporta Clase | SI | NO | NO | SI |
| Multiplataforma | SI | SI | NO | NO |
| Conexión BDD | SI | NO | NO | SI |
| Velocidad de Acceso | SI | SI | NO | SI |
| Licencia libre | SI | SI | NO | NO |
| Constante Desarrollo | SI | NO | NO | SI |
| Distribuido | SI | NO | NO | SI |
| Fácil de Aprender | NO | SI | SI | SI |

**Título** Tabla 1.1.-Comparación JSP, PHP y ASP .NET
**Descripción** Tabla comparativa entre los lenguajes WEB más utilizados.
**Fecha** 02-04-2012
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

#### 1.4.3.1 Selección

Por pedido del Centro de Psicología Aplicado todo software utilizado debe ser Software libre en el cual observando las diferencias PHP es un lenguaje más flexible, es más sencillo y rápida conexión a la base de datos, es fiable y hay bastante tutoriales y guías gratuitas.

## 1.4.4 XAMPP

"**xampp** es un servidor independiente de plataforma, software libre, que contiene al momento de su instalación MYSQL, el servidor Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene de X que trata de especificar para cualquier SO. X Apache, MySQL, PHP, Perl.
El programa está liberado bajo licencia GNU y actúa como un servidor Web libre, fácil de usar e interpretar páginas dinámicas. Actualmente XAMPP esta disponible para Microsoft Windows, GNU/Linux, Solaris, y MacOS X."[7]

## 1.4.5 Apache Servidor Web

"Apache es un servidor Web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP-1.1. Cuando comenzó su desarrollo en 1995 se basó del código del popular NCSA HTTPd 1.3, después fue rescrito por completo. Su nombre se debe a Behelendorf que quería que tenga una expresión que sea firme y enérgico pero no agresivo. Apache consistía solamente en un conjunto de parches para aplicar al servidor de NCSA que era un servidor parcheado. El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

Apache siendo el servidor HTTP más usado, en el 2005 con su auge del 70% como servidor empleado en los sitios web del mundo, actualmente ha sufrido un descenso.

En cuanto a las vulnerabilidades de la seguridad que se descubrieron y se las resolvieron se puede utilizar solo por usuarios locales. Aunque algunas de éstas pueden se accionadas remotamente, o

[7] http://es.wikipedia.org/wiki/PHP, Utilidades y Características de PHP, Última Modificación 13 Enero 2012 a las 21:51.

explotar por los usuarios locales malévolos en las disposiciones de recibimiento compartidas que utilizan PHP como módulo de Apache. ”[8]

**Módulos**

El servidor es muy modular. El servidor consta de una sección core y diversos módulos que dan mucha funcionalidad que se la considera básica en comparación del resto de servidores web. Algunos de estos módulos son:

- “**mod_ssl** - Comunicaciones Seguras vía TLS.
- **mod_rewrite -** Reescritura de direcciones (generalmente utilizado para transformar páginas dinámicas como php en páginas estáticas html para así engañar a los navegantes o a los motores de búsqueda en cuanto a cómo fueron desarrolladas estas páginas).
- **mod_dav -** Soporte del protocolo WebDAV (RFC 2518).
- **mod_deflate -** Compresión transparente con el algoritmo deflate del contenido enviado al cliente.
- **mod_auth_ldap** - Permite autentificar usuarios contra un servidor LDAP.
- **mod_proxy_ajp** - Conector para enlazar con el servidor Jakarta Tomcat de páginas dinámicas en Java (servlets y JSP).

El servidor de base puede ser extendido con la inclusión de módulos externos entre los cuales se encuentran:

- **mod_cband** - Control de tráfico y limitador de ancho de banda.
- **mod_perl** - Páginas dinámicas en Perl.
- **mod_php** - Páginas dinámicas en PHP.
- **mod_python** - Páginas dinámicas en Python.
- **mod_rexx** - Páginas dinámicas en REXX y Object REXX.
- **mod_ruby** - Páginas dinámicas en Ruby.
- **mod_mono** - Páginas dinámicas en Mono

[8] http://es.wikipedia.org/wiki/Servidor_HTTP_Apache, Http servidor Web Apache, Última Modificación 28 Nov. 2011 a las 19:24.

- **mod_security -** Filtrado a nivel de aplicación, para seguridad. ” [9]

#### 1.4.5.0 Utilidad

Apache es usado para enviar páginas web estáticas y dinámicas en la WWW. Muchas aplicaciones actuales están diseñadas como ambiente de implantación a Apache, o usarán características propias este servidor.

Apache es un componente de servidor en la plataforma de aplicaciones LAMP, XAMPP, junto con MySQL y lenguaje de programación web PHP/Perl/Python/Ruby.

El servidor web es redistribuido como parte de varios paquetes de propiedad del software, incluyendo la base de datos Orache y WebSphere application server. El sistema operativo de Mac integra también apache como su servidor web y como soporte de su servidor aplicaciones WebObjects. Soporta Borland en las herramientas de desarrollo como Kylix y Delphi. Apache es incluido con Novell NetWare 6.5, donde es el servidor web por defecto, y en muchas distribuciones Linux.

La licencia en la cual el software esta dado esta bajo de la fundación Apache y distribuido como parte distinta de la historia del servidor Apache HTTP y la comunidad del código abierto. La licencia permite la distribución de sus derivados de código abierto y cerrado a partir de su código funte.

La Free Software Foundation no considera a la Licencia Apache como compatible con la versión 2 de la GNU General Public License (GPL), en la cual el software licenciado bajo la Apache License no puede ser integrado con software distribuido bajo la GPL

Es software libre pero no compatible con GLP. La razón del porque es incompatible es que tiene un requerimiento específico que no está incluido en la GPL, ya que la GPL tiene una terminación de paquetes que ésta no requiere. Dichos casos no son considerados terminación de patentes ya que son mala idea, pero a pesar de esto son incompatibles con la GNU GPL.

- [9] http://es.wikipedia.org/wiki/Servidor_HTTP_Apache, Http servidor Web Apache, Última Modificación: 28 Nov. 2011 a las 19:24

## 1.4.6 SMARTY

Smarty separa el código PHP de las plantillas, es decir como lógica de negocios, del código HTML, como lógica de presentación generando contenidos web mediante colocación de etiquetas Smarty en un documento para realizar la acción PHP deseada. Se encuentra baja la Licencia Pública General Reducida de GNU.

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

**Características**

- Expresiones regulares.
- Bucles foreach, while.
- Sentencias condicionales if, elseif, else.
- Modificadores de variables (por ejemplo: {$variable|nl2br}).
- Funciones creadas por el usuario.
- Evaluación de expresiones matemáticas en la plantilla.

**Filosofía de Smarty**

El diseño de Smarty fue provocada principalmente por los siguientes objetivos:

- Clara separación de la presentación del código de aplicación.
- Backend PHP, frontend del template.
- Complemento de PHP, no lo remplaza.
- Rápido desarrollo / implementación para los programadores y diseñadores.
- Rápido y fácil de mantener.
- Sintaxis fácil de entender, no requiere conocimientos de PHP.

- Flexibilidad para el desarrollo personalizado.
- Seguridad: aislamiento de PHP.
- Libre, de código abierto.

**Dos Campos del pensamiento**

Cuando se trata de plantillas en PHP, hay básicamente dos campos del pensamiento. El primer campo exclama que "PHP es un motor de plantillas". Este enfoque, simplemente es mezclar código PHP con código HTML. Aunque este método es más rápido de la secuencia de comandos de ejecución en el punto de vista, muchos podrían argumentar que la sintaxis de PHP es complicado y difícil de mantener cuando se mezcla con la presentación. PHP funciona bien para la programación, pero no tan bien para las plantillas.

El segundo campo exclama que la presentación debe ser desprovisto de todos los códigos de programación, y en lugar utilizar etiquetas simples para indicar el contenido de la aplicación revelada. Este enfoque es común con los motores de otra plantilla, y este es el enfoque de Smarty. La idea es mantener las plantillas centrado en la presentación, vacío de código de aplicación.

**Importancia de La Separación de PHP de Plantillas**

Hay muchos beneficios de separar el código PHP a partir de plantillas, algunas de las cuales son:

**Sintaxis:** Las plantillas normalmente consisten en marcas semánticas tales como HTML. La sintaxis de PHP que funciona bien para el código de aplicación, pero rápidamente se degrada cuando se mezcla con el lenguaje HTML. La sintaxis sencilla {tag} de smarty está diseñado específicamente para expresar su presentación. Smarty se centra en las plantillas de presentación y menos en el "código". Esto le da a la implementación de plantillas más rapidéz y fácil mantenimiento.

**Características:** El motor de la plantilla tiene muchas características para la presentación que de otro modo tendrían que ser desarrollados, probados y mantenidos en su propio código de aplicación.

**Caja de arena:** Cuando PHP se mezcla con las plantillas, no hay restricciones sobre el tipo de lógica se puede inyectar en una plantilla. Smarty aísla las plantillas de PHP, creando una separación controlada de presentación de la lógica de negocio. Smarty también tiene características de seguridad que pueden imponer restricciones sobre las plantillas.

**Porque no Usar XML**

Hay un par de buenas razones. En primer lugar, Smarty puede ser utilizado a más de solo plantillas basadas en XML / HTML, tales como generación de correos electrónicos, Javascript, CSV y documentos PDF. En segundo lugar, la sintaxis XML / XSLT es aún más detallada y frágil que el código PHP. Es perfecto para las computadoras, pero horribles para los seres humanos. Smarty se trata de ser fácil de leer, entender y mantener.

**Smarty no es un Framework**

Smarty no es MVC. Smarty no es una alternativa a Zend Framework , CodeIgniter , CakePHP , o cualquiera de los otros framework para PHP.

Smarty es un motor de plantillas, y funciona como el componente de visualización la aplicación. Smarty puede ser acoplado a cualquiera de los motores mencionados anteriormente como el componente de visualización. No es diferente de cualquier otro software, Smarty tiene una curva de aprendizaje. Smarty no garantiza un buen diseño de la aplicación o la separación adecuada de la presentación, esto todavía tiene que ser dirigida por un desarrollador y diseñador web avanzado.

**Como trabaja**

Smarty compila copias de las plantillas como scripts PHP. De esta manera usted obtiene los beneficios de ambas sintaxis de la etiqueta de la plantilla y la velocidad de PHP. La compilación sucede una vez que cada plantilla sea invocada por primera vez, y luego las versiones compiladas se utilizan desde ese punto hacia adelante. Este enfoque mantiene las plantillas fáciles de mantener, y sin embargo mantiene los tiempos de ejecución extremadamente rápido. Desde las versiones compiladas son PHP, aceleradores de código como APC o ZendCache continúan trabajando en los scripts compilados.

#### 1.4.6.0 Comparación de Frameworks

##### 1.4.6.0.0 CakePHP

Es un framework para aplicaciones de código abierto en PHP. Un framework para la construcción de sitios web que utilizan una BDD como fuente de recuersos inspirado en Rails, permite el desarrollo de aplicaciones web de manera ágil y estructurada, sin perder flexibilidad.

Entre las características más destacables de CakePHP se incluyen:

- Arquitectura basada en el patrón Modelo Vista Controlador y orientada a objetos: define clases modelo, vista y controlador con funcionalidades básicas y de las cuales heredan todas las clases que se ajustan a este patrón y que son usadas en la aplicación construida con el framework.

- Una comunidad activa de usuarios: creada tras la publicación del framework en 2005 y que ha contribuido a mejorar el framework y difundir su uso.

- Licencia flexible: es distribuido bajo la licencia X11, más conocida entre los desarrolladores de software como MIT License.

- Compatible con PHP4 y PHP5: aunque en PHP4 se requiere especificar algunos parámetros de configuración adicionales en las clases a implementar.

- Operaciones básicas en base de datos: estas operaciones están integradas para interacción con la base de datos y la simplificación de consultas.

- Estructura de aplicaciones: permite al programador hacer uso de un conjunto de convenciones aplicables a la estructura de la base de datos de la aplicación y el framework se encarga de generar el código para la interacción a lo largo de todas las capas de la aplicación.

- Despachador de peticiones: permite acceder a la aplicación a través de URLs amigables y configurables.

- Incorporación de validaciones a lo largo del framework.

- Generación de plantillas de manera rápida y flexible: usando la sintaxis de PHP y con asistentes o helpers.

- Incorporación de asistentes de construcción de vistas: para la automatización de la generación de código en AJAX, JavaScript, formularios HTML, entre otros.

- Componentes de seguridad, manejo de sesiones y de peticiones: que reúnen las mejoras prácticas estandarizadas por la industria del software.

- Listas de control de acceso flexibles: para gestionar el ingreso de usuarios a la aplicación construida con el framework.

- Verificación de ingreso de datos permitidos: permite determinar qué datos pueden ser ingresados y darle el formato adecuado a aquellos que no cumplen las reglas de validación.

- Almacenamiento en caché de las vistas: para acelerar la descarga de las páginas web.

- Trabaja en cualquier subdirectorio de un servidor web: requiere poca o nula configuración del servidor Apache donde se instalará.

**1.4.6.0.1 Symfony**

Es un framework para optimizar el desarrollo de las aplicaciones web a través a varias formas. Separa las reglas de negocio de la aplicación, la lógica del servidor y las plantillas. Tiene variedad de herramientas y clases para emplear menos tiempo de desarrollo en aplicaciones web complejas. Automatiza tareas comunes que el programador pueda enfocarse por completo en las especificaciones.

Entre las características generales del framework podemos citar:

- “Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso de Propel, permiten cambiar con facilidad de SGBD en cualquier fase del proyecto.
- Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- Sencillo de usar en la mayoría de casos, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos.
- Aunque utiliza MVC, tiene su propia forma de trabajo en este punto, con variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones.
- Basado en la premisa de “convenir en vez de configurar”, en la que el desarrollador sólo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.

- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
- Una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo. ”[10]

**1.4.6.0.2 Zend Framework**

El framework facilita no sólo facilitar la programación a través del patrón MVC, sino que automatiza tareas más específicas, como el acceso a base de datos, el filtrado de datos ingresados a la aplicación y la búsqueda en un sitio web ordenando resultados por relevancia.

Entre sus metas se encuentran:

- “Proveer un repositorio de componentes de alta calidad y que cuenten con soporte activo.
- Proveer un sistema completo para el desarrollo de aplicaciones web elaboradas en PHP5.
- Facilitar el aprendizaje en el uso del framework sin tener que aprender un nuevo lenguaje de programación.
- Organizar la colaboración de la comunidad para una programación avanzada en PHP5.

Los componentes con los que cuenta este framework pueden ser agrupados en las siguientes categorías:

- Infraestructura del núcleo del framework componentes requeridos por otros bloques del framework, como memoria caché, configuración del ambiente de trabajo, operación por línea de comandos, registro de actividades y gestión de memoria, entre otros.
- Autentificación y autorización de acceso: responsables de la configuración de listas de control de acceso, autentificación de usuarios y manejo de sesiones.
- Base de datos: clases de acceso, modificación de tablas, obtención de datos mediante consultas SQL y lectura en formato XML.

[10] http://es.wikipedia.org/wiki/Symfony, Detalles y Características de Synfony, Última Modificación: 7 de Feb. De 2012.

- Internacionalización y Localización: para configurar la fecha y hora, la ubicación geográfica que tomará como base la aplicación web, las unidades de medida a usar y la posibilidad de traducir la información a otros idiomas.
- Emails, formatos y búsquedas: generación de archivos PDF, mensajes de correo electrónico en formato de texto y MIME
- Modelo Vista Controlador: centra su atención en el desarrollo de controladores genéricos y específicos: de acciones, de atención de peticiones, de generación de URLs; además de una clase para las vistas.
- Servicios Web: permite que la aplicación web pueda hacer uso de servicios web ofrecidos por aplicaciones externas y exponga servicios web propios. "[11]

#### 1.4.6.1 Selección

Por pedido del Centro de Psicología Aplicado todo software utilizado debe ser Software libre en el cual Smarty se eligió ya que separa nítidamente el código PHP con las plantillas HTML y existe más experiencia por parte de nosotros en el manejo del framework.

## 1.4.7 MYSQL BASE DE DATOS

"MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario, en la cual por estadísticas hay más de 6 millones de instalaciones a nivel mundial. MySQL desde enero de 2008 es una subsidiaria de Sun Microsystems y ésta a su vez Oracle Corporation desde abril de 2009, desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Su licencia esta bajo GNU GPL para cualquier uso compatible, pero las empresas que lo incorporen como producto privado deben comprar a la empresa una licencia específica que le permita el uso del mismo. Está desarrollado en su mayor parte por ANSI C

[11] http://tuxpuc.pucp.edu.pe/articulo/comparativa-de-frameworks-en-php-cakephp-symfony-y-zend-framework,Comparación de framework entre Cake PHP con Synfony y Zend, Última Modificación: 2 de Sep. Del 2007.

Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet. MySQL AB fue fundado por David Axmark, Allan Larsson y Michael Widenius.

**Lenguajes de Programación**

Existen varias interfaces de programación de aplicaciones que permiten, a aplicaciones escritas en diversos lenguajes de programación, acceder a las bases de datos MySQL, incluyendo C, C++, C#,Pascal, Delphi (via dbExpress), Eiffel, Smalltalk, Java (con una implementación nativa del driver de Java), Lisp, Perl, PHP, Python, Ruby,Gambas, REALbasic (Mac y Linux), (x)Harbour (Eagle1),FreeBASIC, y Tcl; cada uno de estos utiliza una interfaz de programación de aplicaciones específica. También existe una interfaz ODBC, llamado MyODBC que permite a cualquier lenguaje de programación que soporte ODBC comunicarse con las bases de datos MySQL. También se puede acceder desde el sistema SAP, lenguaje ABAP. " [12]

MySQL utiliza el motor no transaccional MyISAM en la cual realiza los trabajos de lectura muy rápida en la base de datos, pero provoca problemas de integridad en entornos de alta concurrencia en la modificación. Es por ésto que en aplicaciones web es muy utilizado ya que hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace MySQl ideal para este tipo de aplicaciones. Es importante monitorizar de antemano el rendimiento para detectar y corregir errores tanto en SQL como de programación sea cual sea el entorno.

**Características**

MySQl al comienzo carecía de elementos esenciales en las bases de datos, tales como integridad referencial y transacciones. Pero por su simplicidad comenzó a ser muy utilizada por los desarrolladores web con páginas web con contenido dinámico.

[12] http://es.wikipedia.org/wiki/MySQL, Definición MYSQL, Última Modificación: 16 de Enero de 2012 a la 01:16

Al parar del tiempo esos elementos faltantes en MySQL están siendo incorporados tanto por desarrollo interno como por desarrolladores de software libre. La últimas versiones incorporan las siguientes características:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponibilidad en gran cantidad de plataformas y sistemas.
- Diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles.
- Transacciones y claves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda e indexación de campos de texto.

MySQl siendo un sistema de administración de BDD. Es una colección estructurada de tablas que contienen registros o datos. Siendo una simple lista de compras a una galería de vasto volumen de información en un red corporativa. Para insertar, seleccionar y procesar datos guardados en un computador, se necesita un administrador como MySQL Server. Ya que las computadores son excelentes en manejar grandes cantidades de información, existen también los administradores de BDD que juegan un papel central en computación, como aplicaciones independientes o como parte de otras aplicaciones.

**Características Adicionales**

- Usa GNU Automake, Autoconf, y Libtool para portabilidad.
- Uso de multihilos mediante hilos del kernel.
- Usa tablas en disco b-tree para búsquedas rápidas con compresión de índice.
- Tablas hash en memoria temporales.
- El código MySQL se prueba con Purify así como con Valgrind, una herramienta GPL.
- Completo soporte para operadores y funciones en cláusulas select y where.
- Completo soporte para cláusulas group by y order by, soporte de funciones de agrupación.
- Seguridad: ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor.
- Soporta gran cantidad de datos. MySQL Server tiene bases de datos de hasta 50 millones de registros.

- Se permiten hasta 64 índices por tabla (32 antes de MySQL 4.1.2). Cada índice puede consistir desde 1 hasta 16 columnas o partes de columnas. El máximo ancho de límite son 1000 bytes (500 antes de MySQL 4.1.2).
- Los clientes se conectan al servidor MySQL usando sockets TCP/IP en cualquier plataforma. En sistemas Windows se pueden conectar usando named pipes y en sistemas Unix usando socket Unix.
- En MySQL 5.0, los clientes y servidores Windows se pueden conectar usando memoria compartida.
- MySQL contiene su propio paquete de pruebas de rendimiento proporcionado con el código fuente de la distribución de MySQL.

**Características Distintivas de otras BDD**

- Múltiples motores de almacenamiento (MyISAM, Merge, InnoDB, BDB, Memory/heap, MySQL Cluster, Federated, Archive, CSV, Blackhole y Example en 5.x), permitiendo al usuario escoger la que sea más adecuada para cada tabla de la base de datos.
- Agrupación de transacciones, reuniendo múltiples transacciones de varias conexiones para incrementar el número de transacciones por segundo.

**Compilación del Servidor**

- Estándar: Los binarios estándares de MySQL son los recomendados para la mayoría de los usuarios, e incluyen el motor de almacenamiento InnoDB.
- Max (No se trata de MaxDB, que es una cooperación con SAP): Los binarios incluyen características adicionales que no han sido lo bastante probadas o que normalmente no son necesarias.
- MySQL-Debug: Son binarios que han sido compilados con información de depuración extra. No debe ser usada en sistemas en producción porque el código de depuración puede reducir el rendimiento.

**Licencia**

La licencia GNU GPL de MySQL obliga a que la distribución de cualquier producto derivado se haga bajo esa misma licencia. Si un desarrollador desea incorporar MySQL en su producto pero desea

distribuirlo bajo otra licencia que no sea la GNU GPL, puede adquirir una licencia comercial de MySQL que le permite hacer justamente eso.

### 1.4.7.0 Comparación de Bases de Datos

#### 1.4.7.0.0 Sql Server

"Es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son T-SQL y ANSI SQL

**Características de Microsoft SQL SERVER**

- Soporte de transacciones.
- Escalabilidad, estabilidad y seguridad.
- Soporta procedimientos almacenados.
- Incluye también un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos.

Esta BDD incluye una subversión llamada MSDE con el mismo motor de base de datos pero orientado a proyectos más pequeños, que sus versiones 2005 y 2008 con su nombre SQL Express Edition, que se distribuye de forma gratuita.

Es común desarrollar completos proyectos complementando Microsoft SQL Server y Microsoft Access a través de los llamados ADP(Access Data Project). De esta forma se completa la base de datos , con el entorno de desarrollo (VBA Access), a través de la implementación de aplicaciones de dos capas mediante el uso de formularios Windows.

En el manejo de SQL mediante líneas de comando se utiliza el SQLCMD

Para desarrollo de aplicaciones con mayor complejidad, la BDD incluye interfaces de acceso para varias plataformas de desarrollo, entre ellas .NET pero solo válido para Sistemas Operativo.

**Desventajas**

- MSSQL usa Address Windowing Extensión (AWE) para hacer el direccionamiento de 64-bit. Esto le impide usar la administración dinámica de memoria, y sólo le permite alojar un máximo de 64 GB de memoria compartida.
- MSSQL no maneja compresión de datos (excepto la versión 2008 Enterprise Edition, que sí lo hace), por lo que las bases de datos pueden llegar a ocupar mucho espacio en disco.
- MSSQL requiere de un sistema operativo Microsoft Windows, por lo que no puede instalarse, por ejemplo, en servidores Linux, por esta razon. " [13]

Microsoft desarrolló esta BDD que soporta y es compatible con casi todo, la idea principal es que es una sistema para ser vendido en diferencia de MySQL que se vende solo en casos especiales y no tienen muchas diferencias con SQL SERVER más que el precio.

**Tabla Comparativa**

| Característica | MySQl | SQL SERVER Express | SQL SERVER |
|---|---|---|---|
| Costo | Libre | Libre | Pagado |
| Open Source | SI | No | No |
| Plataformas | Linux, Windows | Windows | Windows |
| Limite tamaño BDD | Limitado por SO | 10GB | Limitado por SO |
| Transacciones | SI | SI | SI |
| Servicio de Reportes | No | SI | SI |

- [13] http://es.wikipedia.org/wiki/Microsoft_SQL_Server,Caracteristicas y detalles de SQL SERVER, Última Modificación: 24 de Marzo del 2012.

| | | | |
|---|---|---|---|
| Elegir diferentes formas de almacenar | SI | No | No |
| Claves Foráneas | Depende del Motor | SI | SI |
| Vistas | SI | SI | SI |
| Procedimientos Almacenados | Si | SI | SI |
| Triggers | Si | SI | Si |
| Cursores | SI | Si | SI |
| Subconsultas | SI | Si | SI |
| Replicación | SI | Limitado | SI |

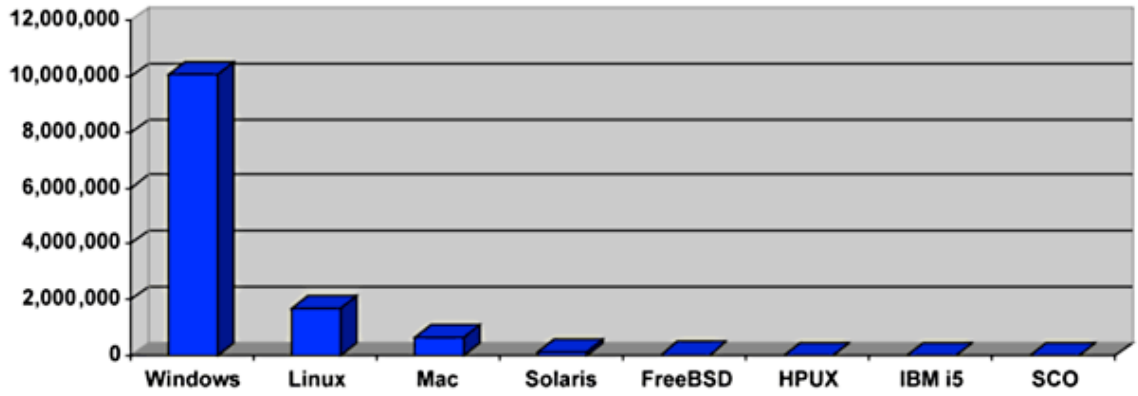
**Título** Tabla 1.2.-Tabla comparativa BDD.

**Descripción** Tabla comparativa de la versión libre de MySQL y la versión libre y pagada de SQL SERVER

**Fecha** 02-04-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**Descargas de MySQL en diferentes SO.**




*Descargas de MySQL desde Sistemas Operativos tomado de mysql.com*

**Título** Figura 1.1.0-Descargas de MySQL

**Descripción** Figura con detalle de descargas de diferentes SO de MySQl.

**Fecha** 08-09-2011

**Copia o Fuente** http://www.latindevelopers.com/articulos/sql-server/diferencias-entre-mysql-y-sql-server.php

#### 1.4.7.1 Selección

Por pedido del Centro de Psicología Aplicado todo software utilizado debe ser Software libre en el cual observando las diferencias no son muy grandes entre MySQL con otra BDD como SQL SERVER que tiene precios de licencias altas. En la cual la BDD seleccionada es MYSQL.

## 1.4.8 RPC

"Llamada a Procedimiento Remoto es un protocolo que permite a un programa ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambos. Este protocolo es definitivamente un gran avance en los sockets que se usan hasta el momento. De esta forma el desarrollador no tenía que estar pendiente de las comunicaciones, ya que RPC las manipula automáticamente.

RPC es muy utilizado dentro del paradigma cliente-servidor. Siendo el cliente el que inicia el proceso enviando una solicitud al servidor para que este ejecute cierta función y enviando de vuela la respuesta de la operación realizada por el cliente.
Esta aplicación web utiliza RPC para hacer validaciones al ejecutar llamadas remotas desde javascript a funciones PHP al realizar validaciones de códigos ingresados por el usuario en las administraciones. "[14]

[14] http://es.wikipedia.org/wiki/RPC, Definición RPC, Ultima Modificación 31 de Oct. Del 2011 a las 12:06

# 2. CAPÍTULO II

En el segundo capítulo, en primer lugar, se analiza la situación del Centro de Psicología Aplicada para conocer cómo se realizan la administración de sus pacientes, y con este análisis se procederá a realizar un levantamiento de los requerimientos necesarios para la realización del sistema de gestión del Centro de Psicología Aplicada, con ésto se logra generar los primeros diagramas para tener una idea de cómo está estructurado el sistema dentro de cada punto que trate sobre los diagramas UML. Se dará una breve explicación de qué hace cada diagrama.

## 2.1 Documento de Requerimientos

### 2.1.1 Introducción

Es importante iniciar con un acuerdo de sobre que contiene el producto y así tener un mecanismo que proteja los requerimientos originales de los desacuerdos encontrados; eso será posible con el documento de requerimientos claro y preciso, con el cual se podrán administrar los cambios y proteger tal documento contra un cliente que insista en interpretar de manera diferente una función que ya ha sido diseñada.

En el documento de requerimientos se describe la interpretación de la definición de necesidades; después de que los clientes hayan leído ese documento y estén de acuerdo en su contenido se podrá argumentar que cualquier cambio costará tiempo y/o dinero, en esto radica la importancia de dicho documento.

### 2.1.2 Objetivo

Con frecuencia se suele hacer cambios a los requerimientos, generalmente, los usuarios no pueden saber lo que necesitan en forma precisa hasta que tratan de usar el producto terminado. Sin embargo, cuando se introduce un sistema nuevo se cambia la manera en que ellos trabajan. Conforme el trabajo de desarrollo progresa empiezan a apreciar la manera en cómo afectará su ambiente, mientras más claro es eso para los usuarios, pensarán en nuevas funciones y características, por esto, los requerimientos constantemente cambian hasta que se congelan en un producto.

La parte difícil del proceso de requerimientos de software es entender que es lo que los usuarios creen que necesitan y ayudarlos a definir esas necesidades en términos de funciones que generen un producto útil. El siguiente paso es realizar la SRS que represente el acuerdo común entre los usuarios y el equipo de trabajo acerca de lo que necesitan.

### 2.1.3 Definición General del Sistema

**Antecedentes**

Tema: Sistema de Gestión Para el Centro de Psicología Aplicada

Se necesita del desarrollo del Sistema de Gestión para tener organizados los historiales de pacientes, los cobros pendientes, el calendario de los terapeutas y que puedan ofrecer un servicio mas rápido y eficiente tanto para todos los usuarios.

**Estrategia**

Consiste en aplicar un proceso cíclico para desarrollar el producto. En cada ciclo se deciden cuáles funcionalidades desarrollar. En el primero de ellos se diseña, implementa y evalúa una primera versión del producto; en el segundo ciclo, se incrementan las funcionalidades del producto para generar una segunda versión. Si hay tiempo, se produce una tercera versión.

**Metodología de Toma de Requerimientos WEB W2000**

W2000 es una metodología de toma de requerimientos WEB en la cual amplía la notación de UML de conceptos heredados de HDM que es el diseño del modelo hipermedia. Este proceso de desarrollo se divide en tres etapas:

- Análisis de Requisitos
    - Análisis de Requisitos Funcionales
    - Análisis Requisitos Navegacionales
- Diseño de Hipermedia
- Diseño Funcional

**Definición de Análisis de Requisitos**

Comienza haciendo un estudio de los diferentes roles de usuario que interactúan con el sistema, cada actor tendrá su modelo de requisitos de navegación y de requisitos funcionales. El modelo de requisitos funcionales es representado como modelo de casos de uso de UML en él se representa la

funcionalidad principal asociada a cada rol. Los requisitos navegacionales son posibilidades de navegación de cada actor.

**Definición de Diseño Hipermedia**

Una vez realizado el análisis de requisitos se pasa a la siguiente sección en la cual consta del modelo conceptual o modelo navegacional en el que los autores han extendido algunos modelos de UML como el diagrama de clases o de estados.

**Definición de Diseño Funcional**

Por último se pasa a la fase en la cual se adapta los diagramas de secuencias para expresar la funcionalidad del sistema que se encuentran en el Anexo A.

## 2.2 Análisis de Requisitos

### 2.2.1 Descripción General

Para la realización de este trabajo se decidió que se tuvieran 4 tipos de usuarios con privilegios y roles particulares:

- **Director:** Comúnmente llamado administrador que no necesariamente deba tener conocimientos de Sistemas ya que el sistema es parametrizable, en el cual este usuario se encargará de dar mantenimiento al sistema en cuestión de las diferentes administraciones de diferentes categorías que existen, siendo su responsabilidad que nuevos servicios del CPA se encuentre actualizado y mantener activo el Centro por medio de Reportes Globales para mantener una equidad de trabajo entre terapeutas así como ver los pacientes atendidos en el trascurso del tiempo especificado, mantener un control entre terapeutas y practicantes en el cual solo el director podrá rectificar un informe ya aprobado de parte de un terapeuta.
- **Secretaria:** este usuario es el que tiene la primera iteración con el paciente ya que el actualiza los datos del paciente, revisa horarios de los terapeutas para asignarle una cita al paciente de acuerdo al desarrollo de su caso, aparte de sus funciones importantes como los cobros al finalizar la cita.
- **Terapeuta/practicante:** tanto terapeuta como practicante tienen un perfil de profesional o estudiante de la carrera de Psicología su diferencia es el tipo de terapeuta que sea para poderlos diferenciar en el cual solo los terapeutas serán los supervisores de los practicantes. Los dos tendrán sus horarios como se les asignará casos con sus respectivas citas dependiendo su disponibilidad.

- **Becario:** este usuario se le otorgará más permisos que un practicante ya que es contratado semestralmente.

**Funcionalidades generales del sistema**

Con la planeación y desarrollo de este proyecto se creó un sistema que sea capaz de darle al usuario la capacidad de administrar el CPA.

**Funciones enfocadas al Director**

- Especialización
  - Actualizar Especialización
- Servicio
  - Nuevo Servicio
  - Actualizar Servicio
- Demanda
  - Nuevo Demanda
  - Actualizar Demanda
- Cubículos
  - Nuevo Cubículo
  - Actualizar Cubículo
- Tarifas
  - Nueva Tarifa
  - Actualizar Tarifa
- Perfiles
  - Nuevo Perfil
  - Actualizar Perfil
- Usuarios
  - Nuevo Usuario
  - Actualizar Usuario
- Terapeutas
  - Nuevo Terapeuta
  - Actualizar Terapeuta
- Distribución Cubículos
  - Asignar Cubículo
  - Actualizar Asignación / Buscar por Cubículo
  - Actualizar Asignación / Buscar por Terapeuta
- Pacientes
  - Nuevo Paciente
  - Actualizar Paciente
- Casos
  - Nuevo Caso
  - Actualizar Caso
- Informe Casos
  - Nuevo/Editar Informe
  - Aprobar Informes
- Citas
  - Registrar Cita
  - Actualizar Cita / Buscar por Paciente
  - Actualizar Cita / Buscar por Fecha
  - Actualizar Cita / Jornada por Terapeuta
  - Ver Calendario

- Cobros
    - Por Caso
    - Por Fechas
    - Por Fechas y Terapeuta
    - Citas Excentas Sin Facturar
    - Citas Con Cobro Incompleto
- Instituciones
    - Nueva Institución
    - Actualizar Institución
    - Referir Paciente
    - Actualizar Referencia
- Encuestas
    - Satisfacción Estudiantes
    - Area Secretaria
    - Satisfacción Servicio
- Registro Asistencia
- Reportes
    - Reporte Asistencia
    - Reporte Histórico Paciente
    - Tabulación Encuesta Estudiante
    - Reporte Informe
    - Reporte Tarifa
    - Reporte Detalle Terapeuta
    - Reporte InterUniversitario Específico
    - Reporte InterUniversitario Global
    - Reporte ExtraUniversitario Específico
    - Reporte ExtraUniversitario Global

**Funciones enfocadas al Terapeuta/practicante**

- Casos
    - Nuevo Caso
    - Actualizar Caso
- Informe Casos
    - Nuevo/Editar Informe
    - Aprobar Informes
- Citas
    - Ver Calendario
- Instituciones
    - Referir Paciente
    - Actualizar Referencia
- Encuestas
    - Satisfacción Estudiantes
- Registro Asistencia
- Ayuda

**Funciones enfocadas a la Secretaria**

- Especialización
    - Actualizar Especialización
- Servicio
    - Nuevo Servicio
    - Actualizar Servicio

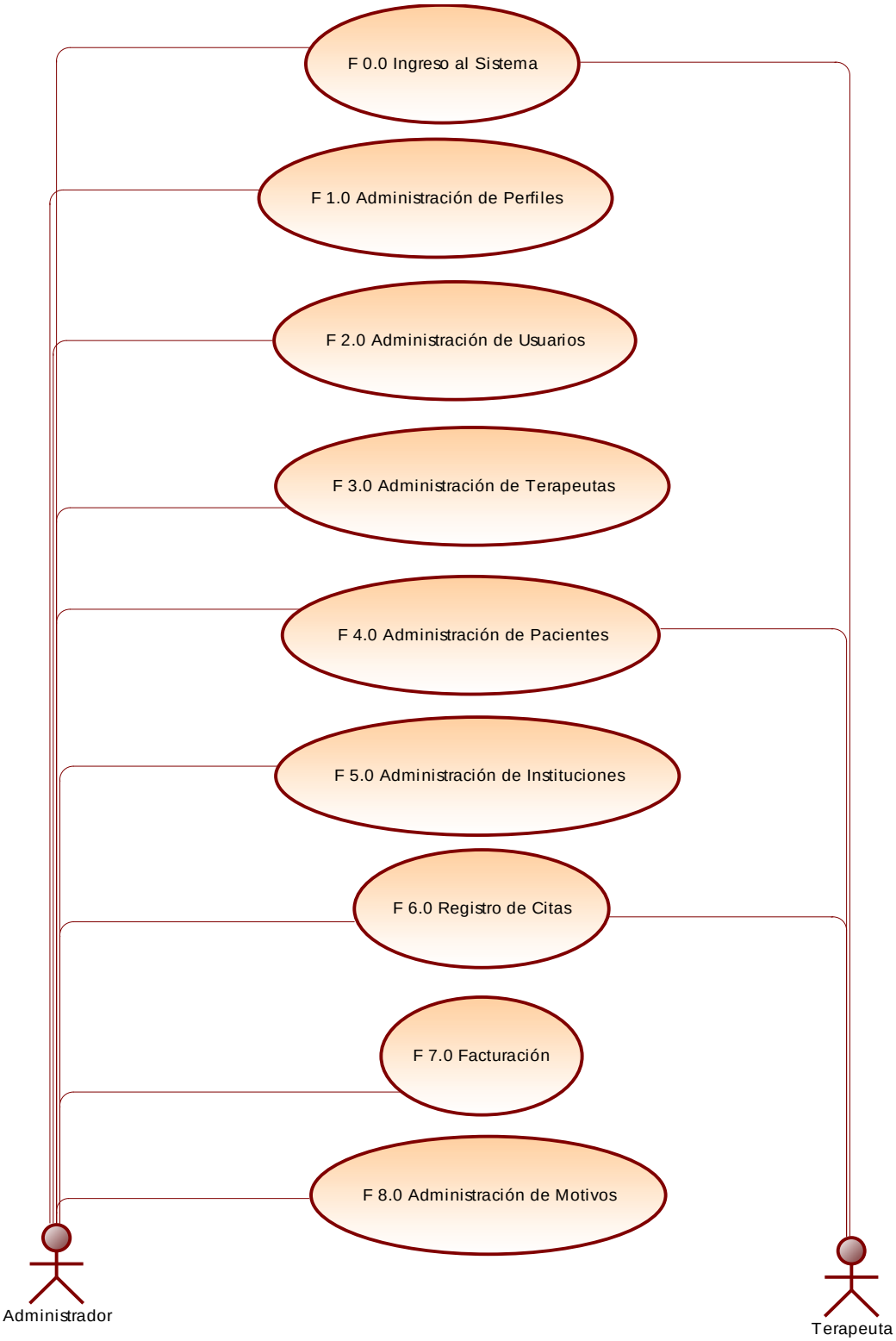
- Demanda
    - Nuevo Demanda
    - Actualizar Demanda
- Cubículos
    - Nuevo Cubículo
    - Actualizar Cubículo
- Tarifas
    - Nueva Tarifa
    - Actualizar Tarifa
- Terapeutas
    - Nuevo Terapeuta
    - Actualizar Terapeuta
- Distribución Cubículos
    - Asignar Cubículo
    - Actualizar Asignación / Buscar por Cubículo
    - Actualizar Asignación / Buscar por Terapeuta
- Pacientes
    - Nuevo Paciente
    - Actualizar Paciente
- Casos
    - Nuevo Caso
    - Actualizar Caso
- Citas
    - Registrar Cita
    - Actualizar Cita / Buscar por Paciente
    - Actualizar Cita / Buscar por Fecha
    - Actualizar Cita / Jornada por Terapeuta
    - Ver Calendario
- Cobros
    - Por Caso
    - Por Fechas
    - Por Fechas y Terapeuta
    - Citas Excentas Sin Facturar
    - Citas Con Cobro Incompleto
- Instituciones
    - Nueva Institución
    - Actualizar Institución
    - Referir Paciente
    - Actualizar Referencia
- Encuestas
    - Area Secretaria
- Registro Asistencia
- Reportes
    - Reporte Histórico Paciente
    - Reporte Detalle Terapeuta
    - Reporte InterUniversitario Específico
    - Reporte ExtraUniversitario Específico

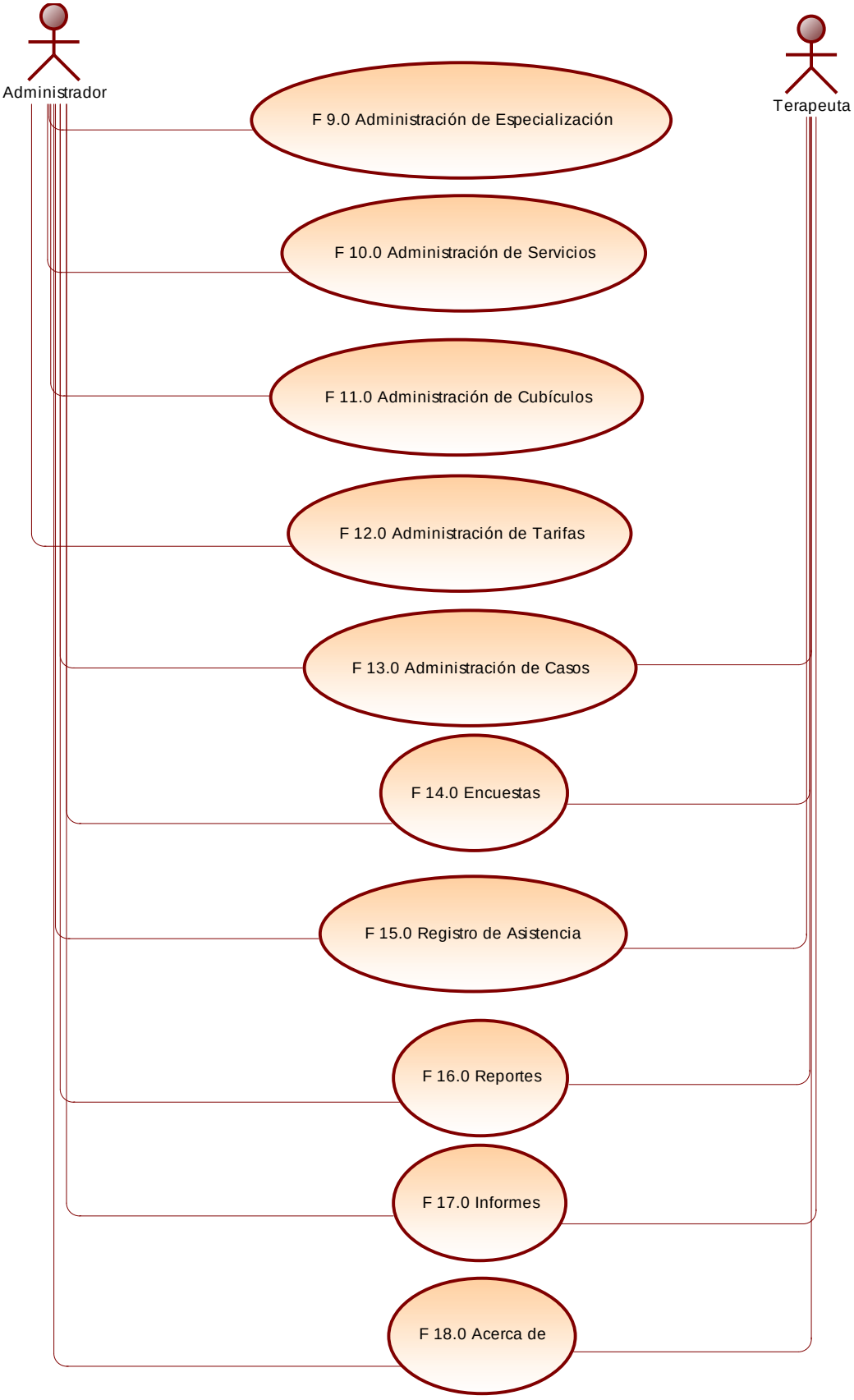
**Funciones enfocadas al Becario**

- Especialización
    - Actualizar Especialización

- Servicio
    - Nuevo Servicio
    - Actualizar Servicio
- Demanda
    - Nuevo Demanda
    - Actualizar Demanda
- Cubículos
    - Nuevo Cubículo
    - Actualizar Cubículo
- Pacientes
    - Nuevo Paciente
    - Actualizar Paciente
- Casos
    - Nuevo Caso
    - Actualizar Caso
- Citas
    - Registrar Cita
    - Actualizar Cita / Buscar por Paciente
    - Actualizar Cita / Buscar por Fecha
    - Actualizar Cita / Jornada por Terapeuta
    - Ver Calendario
- Cobros
    - Por Caso
    - Por Fechas
    - Por Fechas y Terapeuta
    - Citas Con Cobro Incompleto
- Encuestas
    - Satisfacción Estudiantes
- Registro Asistencia

## 2.3 Diagrama General (nivel 0) casos de uso General

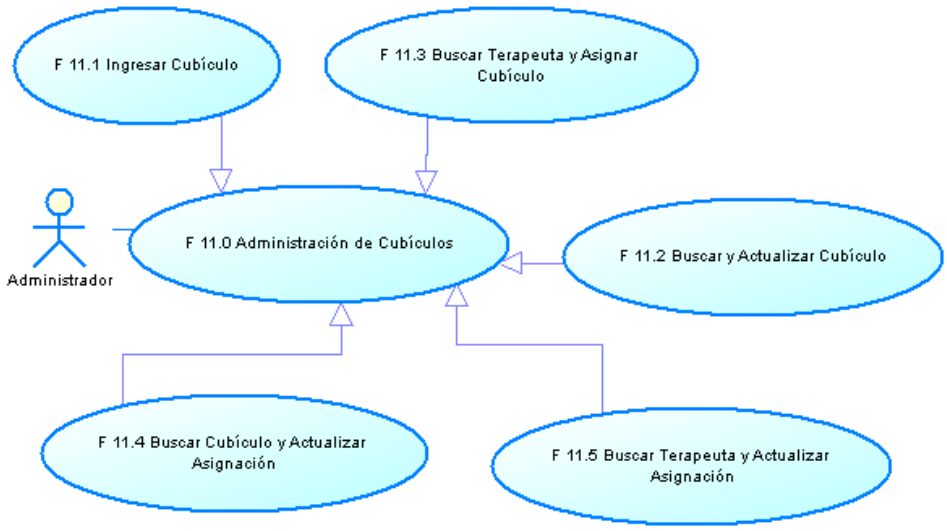



Administrador
Terapeuta
F 9.0 Administración de Especialización
F 10.0 Administración de Servicios
F 11.0 Administración de Cubículos
F 12.0 Administración de Tarifas
F 13.0 Administración de Casos
F 14.0 Encuestas
F 15.0 Registro de Asistencia
F 16.0 Reportes
F 17.0 Informes
F 18.0 Acerca de

| | |
|---|---|
| **Título** | Figura 1.1.-Diagrama General |
| **Descripción** | Diagrama a nivel 0 en el cual se observa todas las administraciones |
| **Fecha** | 08-09-2011 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

## 2.4 Diagrama de Casos de uso

### 2.4.12 Administración de Cubículos

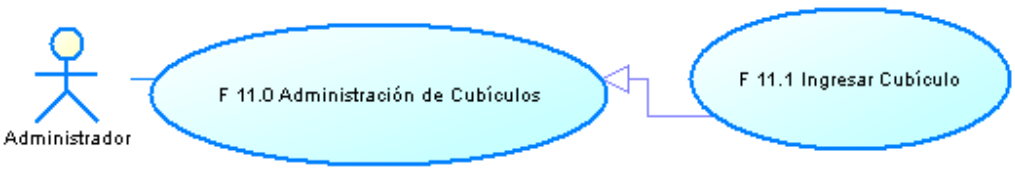



| | |
|---|---|
| **Título** | Figura 1.13.-Administración de Cubículos |
| **Descripción** | Mediante esta funcionalidad el usuario Administra Cubículos. |
| **Fecha** | 01-03-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

#### 2.4.12.1 Detalle:

- Casos de Uso F 11.1 Ingresar Cubículo
  Mediante esta funcionalidad el Administrador puede Ingresar un Cubículo.

- Autores: Administrador




**Título** Figura 1.13.1.-Ingresar Cubículo
**Descripción** Mediante esta funcionalidad el usuario puede Ingresar Cubículo.
**Fecha** 01-03-2012
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**Descripción:**

| Paso | Usuario | Paso | Sistema | Excepción |
|---|---|---|---|---|
| **1** | Selecciona la opción Nuevo Cubículo | **2** | Se carga la Página de Nuevo Cubículo | **E1** |
| **3** | Ingresa el Nombre de Cubículo | | | |
| **4** | Presiona botón Registrar | **5** | Llama al proceso Insertar Cubículo | **E2** |

**Excepciones:**

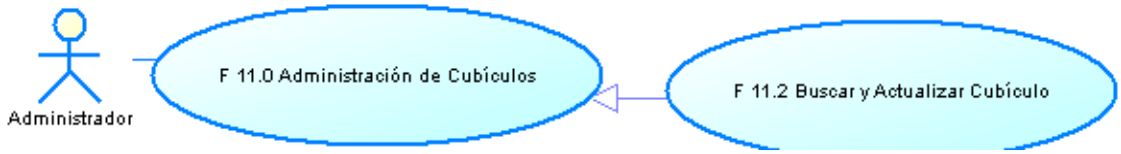
| Código | Descripción | Alternativa |
|---|---|---|
| **E1** | No hay conexión a Internet | Verificar conexión |
| **E2** | Ya existe Perfil | Ingresar Nuevamente |

**Título** Tabla 1.12.-Ingresar Cubículo
**Descripción** Tabla con pasos detallados en el cual el usuario Ingresa Cubículo.
**Fecha** 01-03-2012
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**2.4.12.2 Detalle:**

- Casos de Uso F 11.2 Buscar y Actualizar Cubículo
  Mediante esta funcionalidad el Administrador puede Buscar un Cubículo y Actualizarlo.

- Autores: Administrador




**Título** Figura 1.13.2.- Buscar y Actualizar Cubículo
**Descripción** Mediante esta funcionalidad el usuario puede Buscar y Actualizar Cubículo.
**Fecha** 01-03-2012
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**Descripción:**

| Paso | Usuario | Paso | Sistema | Excepción |
|---|---|---|---|---|
| **1** | Selecciona la opción Actualizar Cubículo | **2** | Se carga la Página de Consultar Cubículo | **E1** |
| **3** | Selecciona el nombre del Cubículo | **4** | | |
| **5** | Presiona botón Consultar Cubículo | **6** | Llama al proceso Consultar Cubículo | |
| | | **7** | Se carga la Página de Actualizar Cubículo | **E1** |
| **8** | Se modifica Nombre | | | |
| **9** | Presiona botón Registrar | **10** | Llama al proceso Actualizar Cubículo | **E1** |

**Excepciones:**

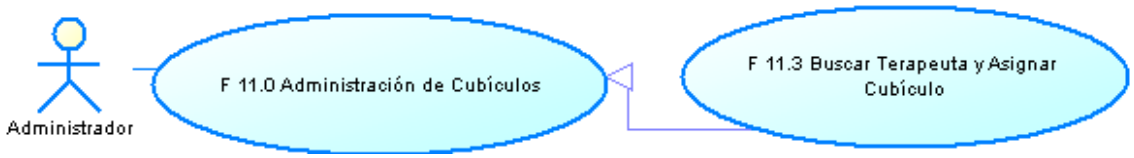
| Código | Descripción | Alternativa |
|---|---|---|
| **E1** | No hay conexión a Internet | Verificar conexión |

**Título** Tabla 1.12.1.- Buscar y Actualizar Cubículo
**Descripción** Tabla con pasos detallados en el cual puede Buscar y Actualizar Cubículo.
**Fecha** 01-03-2012
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**2.4.12.3 Detalle:**

- Casos de Uso F 11.3 Buscar Terapeuta y Asignar Cubículo
  Mediante esta funcionalidad el Administrador puede Ingresar un Cubículo.

- Autores: Administrador




| | |
|---|---|
| **Título** | Figura 1.13.3.- Buscar Terapeuta y Asignar Cubículo |
| **Descripción** | Mediante esta funcionalidad el usuario puede Buscar Terapeuta y Asignar Cubículo. |
| **Fecha** | 01-03-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

**Descripción:**

| Paso | Usuario | Paso | Sistema | Excepción |
|---|---|---|---|---|
| **1** | Selecciona la opción Asignar Cubículo | **2** | Se carga la Página de Nuevo Buscar Terapeuta | **E1** |
| **3** | Ingresa Nombre Terapeuta | **4** | | |
| **5** | Presiona botón Continuar | **6** | Llama al proceso Buscar Terapeuta | **E1** |
| | | **7** | Se carga la Página de Nueva Asignación | |
| **8** | Presiona botón Registrar | **9** | Llama al proceso Nueva Asignación Cubículo | **E2** |

**Excepciones:**

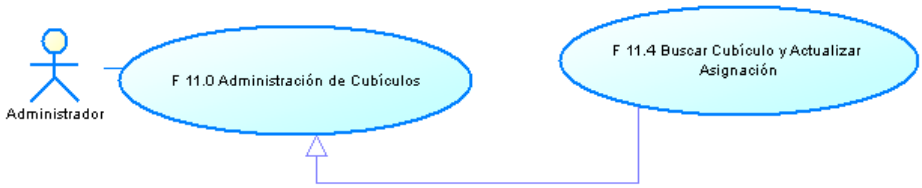
| Código | Descripción | Alternativa |
|---|---|---|
| **E1** | No hay conexión a Internet | Verificar conexión |
| **E2** | Ya existe Asignación | Ingresar Nuevamente |

| | |
|---|---|
| **Título** | Tabla 1.12.2.- Buscar Terapeuta y Asignar Cubículo |
| **Descripción** | Tabla con pasos detallados en el cual el usuario Busca Terapeuta y Asigna Cubículo. |
| **Fecha** | 01-03-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

**2.4.12.4 Detalle:**

- Casos de Uso F 11.4 Buscar Cubículo y Actualizar Asignación
  Mediante esta funcionalidad el Administrador puede mediante Búsqueda Cubículo Actualizar una Asignación.

- Autores: Administrador




| | |
|---|---|
| **Título** | Figura 1.13.4.- Buscar Cubículo y Actualizar Asignación |
| **Descripción** | Mediante esta funcionalidad el usuario puede Buscar Cubículo y Actualizar Asignación. |
| **Fecha** | 01-03-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

**Descripción:**

| Paso | Usuario | Paso | Sistema | Excepción |
|---|---|---|---|---|
| **1** | Selecciona la opción Actualizar Asignación por Cubículo | **2** | Se carga la Página de Consultar Cubículo | **E1** |
| **3** | Selecciona el nombre del Cubículo | **4** | | |
| **5** | Presiona botón Consultar | **6** | Llama al proceso | |

| | | | | |
|---|---|---|---|---|
| | Cubículo | | Consultar Cubículo | |
| | | **7** | Se carga la Página de Actualizar Asignación | **E1** |
| **8** | Presionar Link editar | **9** | Se carga la Página Actualizar Cubículo | **E1** |
| **10** | Se modifica datos de Asignación | | | |
| **11** | Presiona botón Registrar | **12** | Llama al proceso Actualizar Asignación | |

**Excepciones:**

| Código | Descripción | Alternativa |
|---|---|---|
| **E1** | No hay conexión a Internet | Verificar conexión |

**Título** Tabla 1.12.3.- Buscar Cubículo y Actualizar Asignación

**Descripción** Tabla con pasos detallados en el cual el usuario Busca Cubículo y Actualiza Asignación.

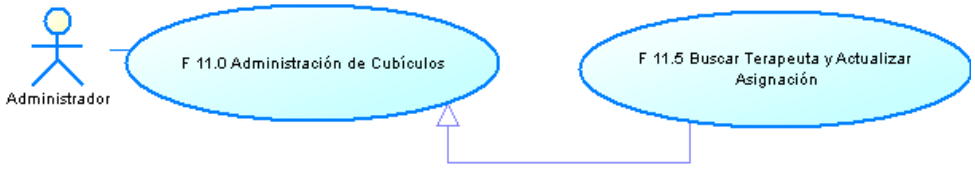
**Fecha** 01-03-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**2.4.12.5 Detalle:**

- Casos de Uso F 11.5 Buscar Terapeuta y Actualizar Asignación
  Mediante esta funcionalidad el Administrador puede mediante Búsqueda por Terapeuta Actualizar una Asignación.

- Autores: Administrador



**Título** Figura 1.13.5.- Buscar Terapeuta y Actualizar Asignación

**Descripción** Mediante esta funcionalidad el usuario puede Buscar Terapeuta y Actualizar Asignación.

**Fecha** 01-03-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**Descripción:**

| Paso | Usuario | Paso | Sistema | Excepción |
|---|---|---|---|---|
| **1** | Selecciona la opción Actualizar Asignación por Terapeuta | **2** | Se carga la Página de Consultar Cubículo | **E1** |
| **3** | Selecciona el nombre del Terapeuta | **4** | | |
| **5** | Presiona botón Continuar | **6** | Llama al proceso Consultar Terapeuta | |
| | | **9** | Se carga la Página de Asignación Cubículo | **E1** |
| **10** | Presionar Link editar | **11** | Se carga la Página Actualizar Cubículo | **E1** |
| **12** | Se modifica datos de Asignación | | | |
| **13** | Presiona botón Registrar | **14** | Llama al proceso Actualizar Asignación | |

**Excepciones:**

| Código | Descripción | Alternativa |
|---|---|---|
| **E1** | No hay conexión a Internet | Verificar conexión |

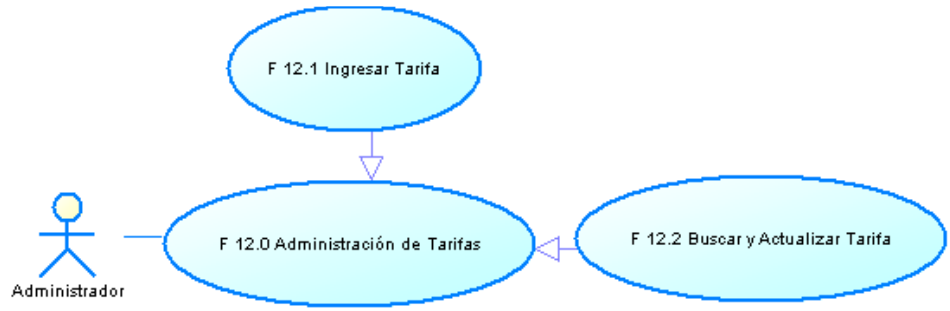
**Título** Tabla 1.12.4.- Buscar Terapeuta y Actualizar Asignación

**Descripción** Tabla con pasos detallados en el cual el usuario Busca Terapeuta y Actualiza Asignación.

**Fecha** 01-03-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

### 2.4.13 Administración de Tarifas




| | |
|---|---|
| **Título** | Figura 1.14.-Administración de Tarifas |
| **Descripción** | Mediante esta funcionalidad el usuario Administra Tarifas. |
| **Fecha** | 01-03-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

#### 2.4.13.1 Detalle:

- Casos de Uso F 12.1 Ingresar Tarifa
  Mediante esta funcionalidad el Administrador puede Ingresar una Tarifa.

- Autores: Administrador


| | |
|---|---|
| **Título** | Figura 1.14.1.-Ingresar Tarifa |
| **Descripción** | Mediante esta funcionalidad el usuario puede Ingresar Tarifa. |
| **Fecha** | 01-03-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

**Descripción:**

| Paso | Usuario | Paso | Sistema | Excepción |
|---|---|---|---|---|
| **1** | Selecciona la opción Nueva Tarifa | **2** | Se carga la Página de Nuevo Tarifa | **E1** |
| **3** | Ingresa el Tipo y Costo | | | |
| **4** | Presiona botón Registrar | **5** | Llama al proceso Insertar Tarifa | **E2** |

**Excepciones:**

| Código | Descripción | Alternativa |
|---|---|---|
| **E1** | No hay conexión a Internet | Verificar conexión |
| **E2** | Ya existe Tarifa | Ingresar Nuevamente |

**Título** Tabla 1.13.-Ingresar Tarifa
**Descripción** Tabla con pasos detallados en el cual el usuario Ingresa Tarifa.
**Fecha** 01-03-2012
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**2.4.13.2 Detalle:**

- Casos de Uso F 12.2 Buscar y Actualizar Tarifa
  Mediante esta funcionalidad el Administrador puede mediante Búsqueda Actualizar una Tarifa.

- Autores: Administrador


**Título** Figura 1.14.2.- Buscar y Actualizar Tarifa
**Descripción** Mediante esta funcionalidad el usuario puede Buscar y Actualizar Tarifa.
**Fecha** 01-03-2012
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

**Descripción:**

| Paso | Usuario | Paso | Sistema | Excepción |
|---|---|---|---|---|
| **1** | Selecciona la opción Actualizar Tarifa | **2** | Se carga la Página de Consultar Tarifa | **E1** |
| **3** | Selecciona el tipo de Tarifa | **4** | | |
| **5** | Presiona botón Consultar Tarifa | **6** | Llama al proceso Consultar Tarifa | |
| | | **7** | Se carga la Página de Actualizar Tarifa | **E1** |
| **8** | Se modifica el tipo y costo | | | |
| **9** | Presiona botón Registrar | **10** | Llama al proceso Actualizar Tarifa | **E1** |

**Excepciones:**

| Código | Descripción | Alternativa |
|---|---|---|
| **E1** | No hay conexión a Internet | Verificar conexión |

**Título** Tabla 1.13.1.- Buscar y Actualizar Tarifa

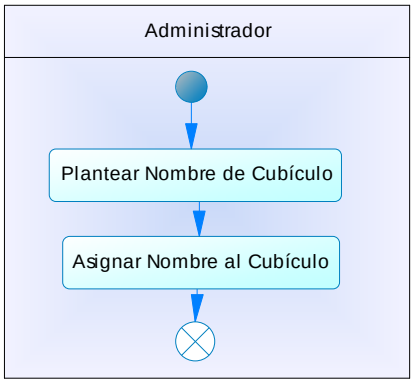
**Descripción** Tabla con pasos detallados en el cual el usuario Busca y Actualiza una Tarifa.

**Fecha** 01-03-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

## 2.5 Diagramas de Actividades

### F 11.1 Ingresar Cubículo



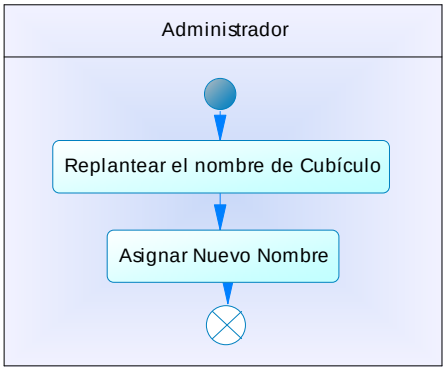

**Título** Figura 1.28.- Ingresar Cubículo

**Descripción** Mediante esta funcionalidad el usuario puede Ingresar Cubículo.

**Fecha** 01-04-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

### F 11.2 Buscar y Actualizar Cubículo



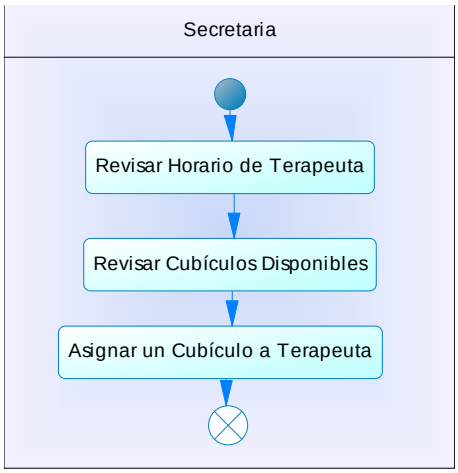

**Título** Figura 1.28.1.- Buscar y Actualizar Cubículo

**Descripción** Mediante esta funcionalidad el usuario puede Buscar y Actualizar Cubículo.

**Fecha** 01-04-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

### F 11.3 Buscar Terapeuta y Asignar Cubículo



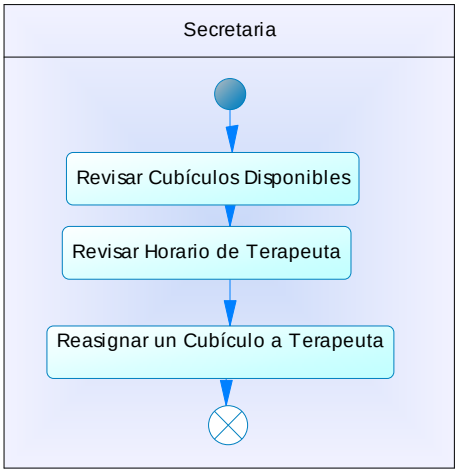

**Título** Figura 1.28.2.- Buscar Terapeuta y Asignar Cubículo

**Descripción** Mediante esta funcionalidad el usuario puede Buscar Terapeuta y Asignar Cubículo.

**Fecha** 01-04-2012

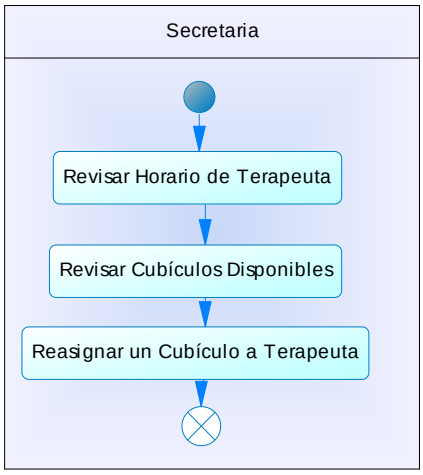
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

### F 11.4 Buscar Cubículo y Actualizar Asignación



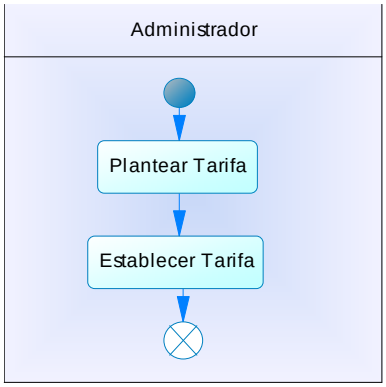
| | |
|---|---|
| **Título** | Figura 1.28.3.- Buscar Cubículo y Actualizar Asignación |
| **Descripción** | Mediante esta funcionalidad el usuario puede Buscar Cubículo y Actualizar Asignación. |
| **Fecha** | 01-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

**F 11.5 Buscar Terapeuta y Actualizar Asignación**




| | |
|---|---|
| **Título** | Figura 1.28.4.- Buscar Terapeuta y Actualizar Asignación |
| **Descripción** | Mediante esta funcionalidad el usuario puede Buscar Terapeuta y Actualizar Asignación. |
| **Fecha** | 01-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

**F 12.1 Ingresar Tarifa**



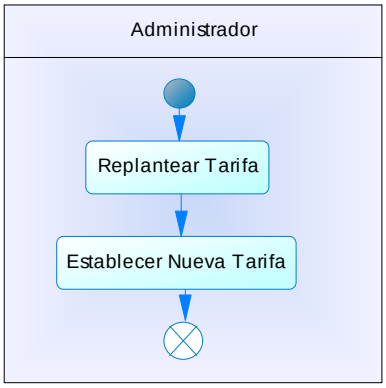
**Título** Figura 1.29.- Ingresar Tarifa

**Descripción** Mediante esta funcionalidad el usuario puede Ingresar Tarifa.

**Fecha** 01-04-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

### F 12.2 Buscar y Actualizar Tarifa




**Título** Figura 1.29.1.- Buscar y Actualizar Tarifa

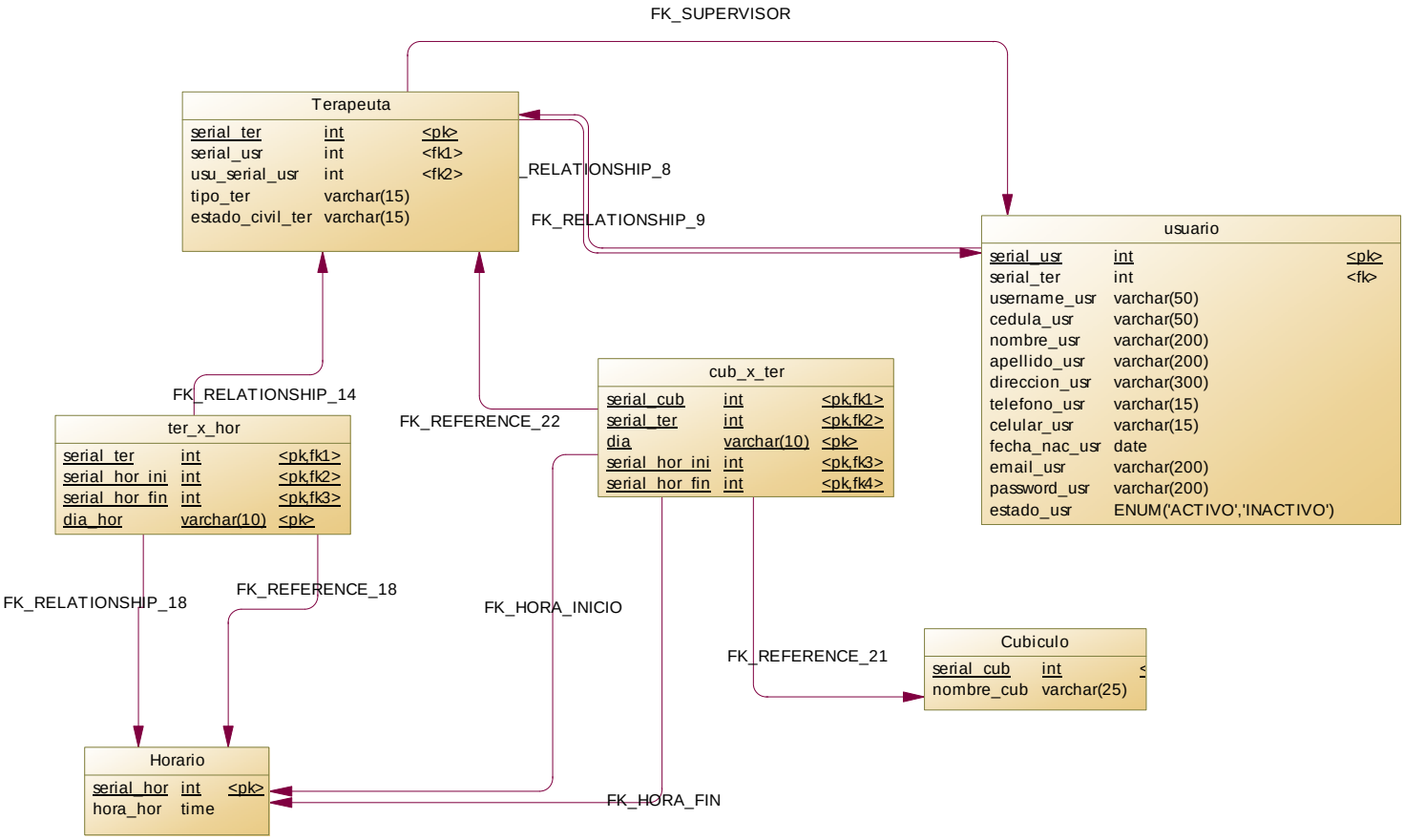
**Descripción** Mediante esta funcionalidad el usuario puede Buscar y Actualizar Tarifa.

**Fecha** 01-04-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

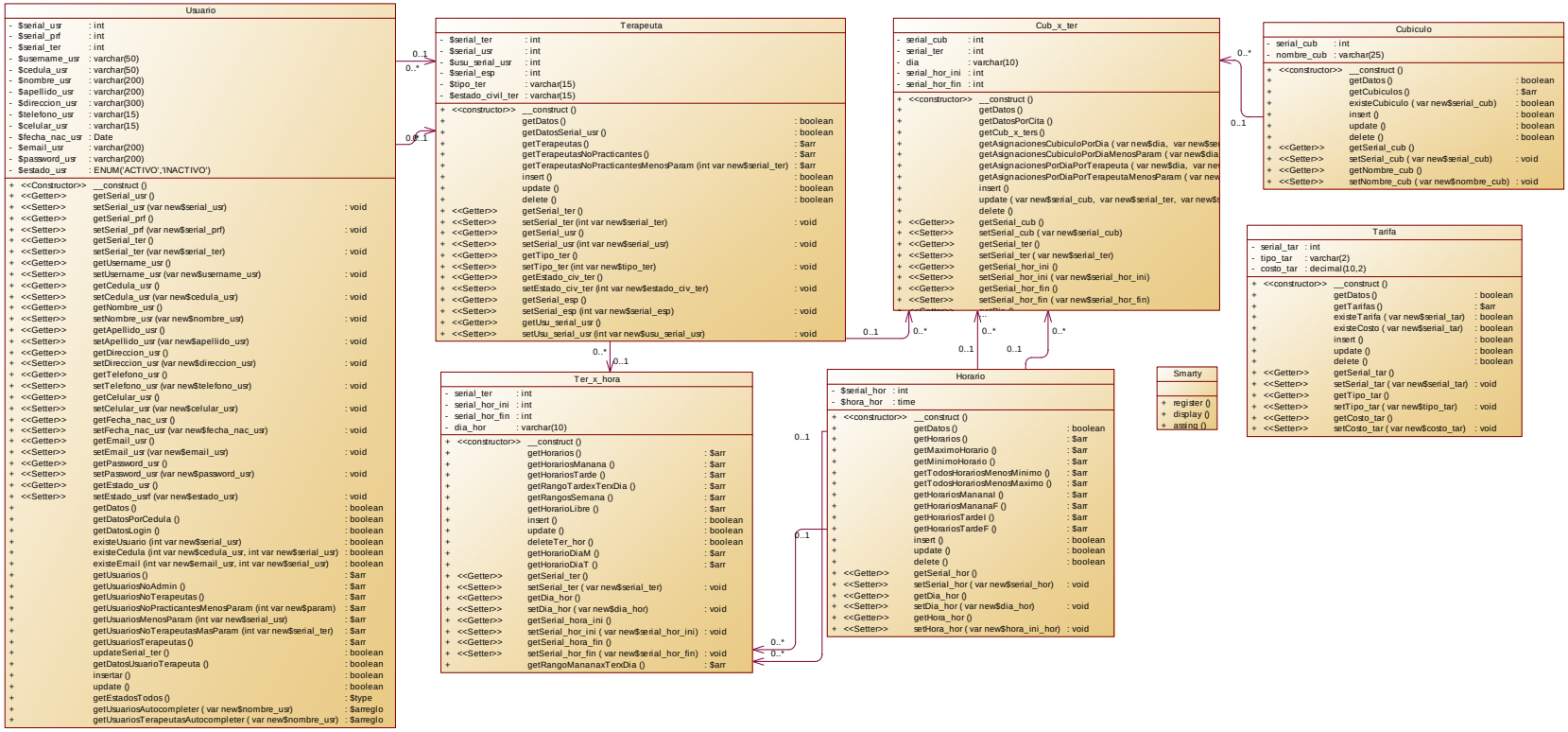
## 2.6 Diseño Hipermedia

### 2.6.1 Modelo Físico



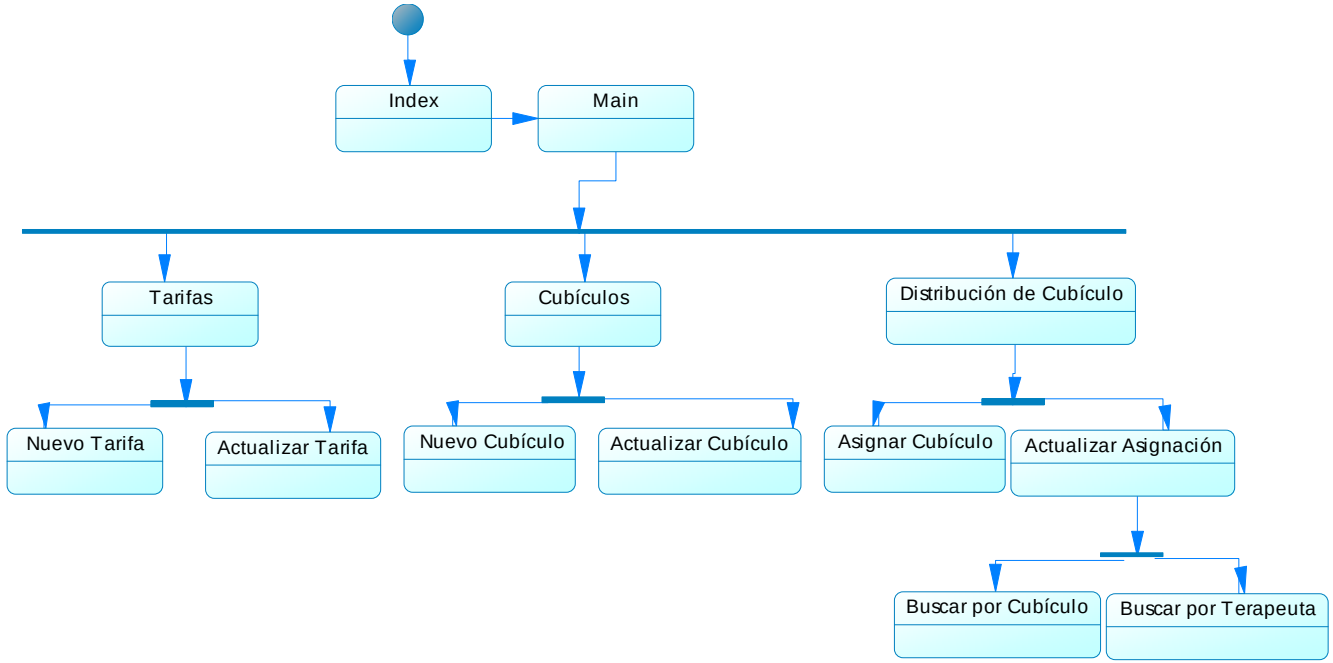

| | |
|---|---|
| **Título** | Figura 1.32.1.-Modelo Conceptual |
| **Descripción** | Se puede observar las relaciones entre cada entidad. |
| **Fecha** | 01-08-2011 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

## 2.6.2 Diagrama de Clases




| | |
|---|---|
| **Título** | Figura 1.32.2-Diagrama de Clases |
| **Descripción** | Se puede observar los atributos y métodos entre cada entidad. |
| **Fecha** | 01-08-2011 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

### 2.6.3 Diseño Navegacional o estados




| | |
|---|---|
| **Título** | Figura 1.32.3-Diagrama de Estados |
| **Descripción** | Se puede observar los estados entre cada entidad. |
| **Fecha** | 01-08-2011 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

## 2.7 Diseño funcional

Este apartado se encuentra en el Anexo A en el cual detalla todos los diagramas de secuencia.

# 3. CAPÍTULO III

En este tercer capítulo se implementará los módulos que interactúan en el sistema para el cual se separa por capas el código fuente, comenzamos por la capa de datos, la capa de negocio, y la interface gráfica de usuario, después comenzamos con las respectivas pruebas unitarias de los módulos utilizados en el sistema. Luego, se nombra los pasos a seguir para la implantación del sistema de gestión para el Centro de Psicología Aplicada de la PUCE.

Dentro de la metodología aplicada se realiza una estrategia de implementación, esta estrategia incluye revisión, desarrollo y pruebas unitarias.

La estrategia de revisión que se tiene para el sistema del Centro de Psicología Aplicada es, basándose en el plan de pruebas de integración, probar independientemente cada módulo, primero aquellos que sean considerados administración y después aquellos que sean considerados procesos.

La estrategia utilizada para el desarrollo del sistema del Centro de Psicología Aplicada es utilizar el mismo formato de comentarios para todas las funciones del sistema así como la creación de clases genéricas como la conexión a la base de datos.

Básicamente la estrategia de pruebas que se va a utilizar para el desarrollo del sistema es utilizar los planes de prueba realizados en las etapas de diseño y de requerimientos, esto nos da la garantía de que el sistema está desarrollado de acuerdo a los requerimientos de los clientes y que cumple con estándares de diseño y de calidad.

Por lo expuesto en anteriores capítulos, este capítulo cubrirá solo la administración de tarifas y el proceso de asignación de cubículos, el código fuente del resto de módulos se los puede encontrar en el CD.

Una vez que nos aseguramos que el diseño esté correctamente realizado, se procede a la codificación de cada módulo del sistema.

El primer módulo en ser implementado será la administración de tarifas:
A través del Anexo B se detalla las cartas de las reuniones con los usuarios, y la carta de aceptación del sistema.

### 3.1 Administración de Tarifas

Este módulo sirve básicamente para almacenar la información de la tarifa, que se verá reflejado en su tipo y su valor. El módulo consta de dos partes, la primera es el desarrollo del módulo nombrado, y la segunda parte es la construcción de interface de usuario.

#### 3.1.1 Ingresar una nueva tarifa

Es la primera funcionalidad para este módulo, para su implementación, se utilizó la arquitectura en tres capas, en el cual consta tres archivos diferentes, la primera pertenece a la capa de datos, la segunda es la parte de la capa del negocio, y la tercera es la capa de interface de usuario.

**Tarifa.class(Capa de datos)**

```
/**********************************************
   * funciongetDatos
   *Retorna todos los datos de la Tarifa seleccionada
**********************************************/
        functiongetDatos(){ //getData()
                if($this->serial_tar!=NULL){

                        $sql =   "SELECTt.serial_tar, t.tipo_tar,t.costo_tar
                                        FROM tarifa t
                                        WHERE serial_tar='".$this->serial_tar."'";
                        $result = $this->db->Execute($sql);

                        if ($result === false)
                                return false;

                        if($result->fields[0]){
                    $this ->serial_tar = $result->fields[0];
                                $this ->tipo_tar = $result->fields[1];
                                $this ->costo_tar = $result->fields[2];


                                return true;
                        }
                        else
```

```
                                        return false;

                        }else
                                return false;
                }

/***********************************************
   * functioninsert
   * Inserta un nuevo registro en la BDD
***********************************************/
function insert() {
        $sql= "
           INSERT INTO tarifa (
                                        tipo_tar,
                                        costo_tar

                                        )
VALUES(
                '".$this->tipo_tar."',
".$this->costo_tar.")";

        //echo $sql;
$result = $this->db->Execute($sql);


       // if ($result === false)return false;

if($result==true)
return true;// $this->db->Insert_ID();
else
return false;
   }
           /***********************************************
   * functionupdate
   * Actualiza un registro en la BDD
***********************************************/
           function update(){
                        $sql= "
```

```
                    UPDATE `tarifa` SET
                        `tipo_tar` = '".$this->tipo_tar."',
                        `costo_tar` = ".$this->costo_tar."";


            $sql.="  WHERE `tarifa`.`serial_tar` = '".$this->serial_tar."'";
            //echo $sql;
            $result = $this->db->Execute($sql);
            if($result==true)
                    return true;
            else
                    return false;
        }
//getters
functiongetSerial_tar(){
            return $this->serial_tar;
        }

functiongetTipo_tar(){
            return $this->tipo_tar;
        }

functiongetCosto_tar(){
            return $this->costo_tar;
        }

//setters
functionsetSerial_tar($serial_tar){
            $this->serial_tar = $serial_tar;
        }

        functionsetTipo_tar($tipo_tar){
             $this->tipo_tar = $tipo_tar;
        }
functionsetCosto_tar($costo_tar){
             $this->costo_tar = $costo_tar;
        }
```

**fNuevaTarifa.php (Capa de Negocio)**

Antes de desplegar la capa de interfaz de usuario smarty carga la capa de datos necesarios para ser enviados a la capa de negocio, en este caso como solo registra una tarifa, se llama a los métodos register para registrar variables utilizadas en el archivo y sean utilizados después y display para llamar a la capa de interfaz de usuario, estos son los métodos mas comunes y utilizados de smarty.

```
Request::setInteger('0:error');

$smarty->register('error');

$smarty->display();
```

**pNuevaTarifa.php**

Esta es la capa en donde se llaman a los métodos necesarios de la clase tarifa.

```
$tarifa=new Tarifa($db);

$tarifa->setTipo_tar($_POST['txtTipo']);

$tarifa->setCosto_tar($_POST['txtCosto']);

if($tarifa->insert()){

                        $error=1;

http_redirect('modulos/tarifa/fNuevaTarifa/1');

                        }


else{

                                $error=2;

http_redirect('modulos/tarifa/fNuevaTarifa/2');

}


$smarty->register('error');

$smarty ->register('usuario');

$smarty->display('modulos/institucion/fNuevaTarifa.tpl');
```

**fNuevaTarifa.es.tpl (Capa de Interfaz de Usuario)**

Como se pude observar, lo que hace esta capa es transmitir los datos que el usuario ingresa por pantalla a la capa de negocio la cual a su vez, envía estos parámetros a la capa de datos.

```
{assignvar="title" value="NUEVA TARIFA"}

<formname="frmNuevaTarifa" id="frmNuevaTarifa" method="post"
action="{$document_root}modulos/tarifa/pNuevaTarifa" >

        <div class="span-19  last title">

        Registro de Nueva Tarifa<br />

<label>(*) Campos Obligatorios</label>

</div>

        {if $error}

<div class="prepend-4 span-14 append-1 last">

        <div class="span-10  {if $error eq 1}success{else}error{/if}" align="center">

{if $error eq 1}

           Tarifa creada exit&oacute;samente.

         {elseif $error eq 2}

            No se pudo crear la  Tarifa. Por favor int&eacute;ntelo nuevamente.

{/if}

</div>

</div>

   {/if}

<div class="span-5 span-9 prepend-5 last">

        <ul id="alerts" class="alerts"></ul>

</div>

<div class="span-19 last   line">

<div class="prepend-1 span-7 label">* Tipo:</div>

<div class="span-8 append-2 last"><input type="text" name="txtTipo" id="txtTipo" title='El campo
"Tipo" esobligatorio.'/></div>
```

```
</div>

<div class="span-19 last   line">

<div class="prepend-1 span-7 label">* Costo:</div>

<div class="span-8 append-2 last"><input type="text" name="txtCosto" id="txtCosto" title='El campo "Costo" esobligatorio.'/></div>


</div>

        <div class="span-19 last buttons">

        <input type="submit" name="btnInsert" id="btnInsert" value="Registrar"/>

</div>

</form>
```

### 3.1.2 Actualizar datos de tarifa

La actualización de tarifas utiliza métodos similares a los métodos de la funcionalidad anterior, la diferencia está en que se tiene que buscar primero una tarifa, transmitir esos datos a la capa del negocio para poder modificarla.

**fBuscarTarifa.php**

```
Request::setInteger('0:error');

$tarifas = new Tarifa($db);

$tarifas = $tarifas ->getTarifas();

$smarty->register('tarifas,error');

$smarty->display();
```

**fActualizarTarifa.php**

```
$serial_tar=$_POST['selTarifa'];

$tarifa = new Tarifa($db);


$tarifa->setSerial_tar($serial_tar);

if($tarifa->getDatos()){
```

```
                        $data['serial_tar'] = $serial_tar;
$data['tipo_tar'] = $tarifa->getTipo_tar();
                        $data['costo_tar'] =$tarifa ->getCosto_tar();


                                $smarty ->register('user,data');
}
$smarty ->register('message,tarifa,data');
$smarty ->display();
```

**pActualizarTarifa.php**

```
$tarifa = new Tarifa($db);
if($_POST['serial_tar']){
        $tarifa ->setSerial_tar($_POST['serial_tar']);
        $tarifa ->getDatos();
                $data['tipo_tar'] = $_POST['txtTipo'];
                $data['costo_tar'] = $_POST['txtCosto'];
        $tarifa ->setTipo_tar($data['tipo_tar']);
        $tarifa ->setCosto_tar($data['costo_tar']);
        if($tarifa -> update()){
http_redirect('modulos/tarifa/fBuscarTarifa/1');
        }else{
                $message = 2;
                $smarty->register('message');
http_redirect('modulos/tarifa/fBuscarTarifa/2');
        }
}
```

**fBuscarTarifa.es.tpl**

```
{assignvar="title" value="BUSCARTARIFA"}
```

```
<form name="frmBuscarTarifa" id="frmBuscarTarifa"  method="post"
action="{$document_root}modulos/Tarifa/fActualizarTarifa" >

        <div class="span-19 last title">ConsultarTarifa<br />

        <label>(*) Campos Obligatorios</label>

</div>

        {if $error}

<div class="prepend-4 span-14 append-1 last">

        <div class="span-10  {if $error eq 1}success{else}error{/if}" align="center">

{if $error eq 1}

           Tarifa actualizada exit&oacute;samente.

            {elseif $error eq 2}

            No se pudo actualizar la Tarifa. Por favor int&eacute;ntelo nuevamente.

{/if}

</div>

</div>

   {/if}

        <div class="span-5 span-9 prepend-5 last">

        <ul id="alerts" class="alerts"></ul>

</div>

<div class="span-19 last  line">

<div class="prepend-1 span-7 label">* Tipo de Tarifa:</div>

<div class="span-8 append-2 last" id="profileContainer">

        <select id="selTarifa" name="selTarifa" title="Seleccione una Tarifa">

        <option value="">- Seleccione -</option>

          {foreach from="$tarifas" item="p"}

                <option value="{$p.serial_tar}">{$p.tipo_tar}</option>

        {/foreach}

</select>
```

```
</div>
        </div>
<div class="span-19 last buttons">
<input type="submit" name="btnSearch" value="ConsultarTarifa"/>
</div>
</form>
```

**fActualizarTarifa.es.tpl**

```
{assignvar="title" value="ACTUALIZARTARIFA"}
<form name="frmActualizarTarifa" id="frmActualizarTarifa" method="post"
action="{$document_root}modulos/tarifa/pActualizarTarifa" >
        <div class="span-19  last title">
        Actualizar Tarifa<br />
<label>(*) Campos Obligatorios</label>
</div>
        {if $error}
<div class="prepend-4 span-14 append-1 last">
        <div class="span-10  {if $error eq 1}success{else}error{/if}" align="center">
{if $error eq 1}
           Tarifa creada exit&oacute;samente.
         {elseif $error eq 2}
            No se pudo crear la  Tarifa. Por favor int&eacute;ntelo nuevamente.
{/if}
</div>
</div>
   {/if}
<div class="span-5 span-9 prepend-5 last">
        <ul id="alerts" class="alerts"></ul>
</div>
```

* Tipo:

<input type="text" name="txtTipo" id="txtTipo" title='El campo "Tipo" esobligatorio.' {if $data.tipo_tar}value="{$data.tipo_tar}"{/if}>

* Costo:

<input type="text" name="txtCosto" id="txtCosto" title='El campo "Costo" esobligatorio.'{if $data.costo_tar}value="{$data.costo_tar}"{/if}>

<input type="hidden" name="serial_tar" id="serial_tar" {if $data.serial_tar}value="{$data.serial_tar}"{/if} >

<input type="submit" name="btnInsert" id="btnInsert" value="Registrar"/>

</form>

**3.1.3 Pruebas unitarias**

## Ingresando una nueva tarifa

Una vez terminado la codificación del módulo tarifas, se procede a realizar las pruebas unitarias del mismo:

Para ingresar una nueva tarifa se debe llenar la información del tipo de tarifa y su respectivo el costo.


| | |
|---|---|
| **Título** | Figura 1.33.-Ingresando una Tarifa |
| **Descripción** | Ingreso de datos de tarifa |
| **Fecha** | 25-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

Una vez que la información ha sido ingresada por el usuario, se presiona el botón registrar y el sistema despliega un mensaje avisando que el registro ha sido creado exitosamente.


| | |
|---|---|
| **Título** | Figura 1.34.-Ingreso de Tarifa satisfactorio |
| **Descripción** | Ingreso de datos de tarifa correctamente |
| **Fecha** | 25-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

## Actualizando los datos de una tarifa

Esta es otra de las funcionalidades del módulo, para actualizar los datos de una tarifa, primero se tiene que seleccionar una tarifa del combo box.


| | |
|---|---|
| **Título** | Figura 1.35.-Seleccionando una tarifa |
| **Descripción** | Seleccionando una tarifa del combo |
| **Fecha** | 25-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

Una vez escogida la tarifa, se procede a cambiar el costo o tipo de tarifa.

| | |
|---|---|
| **Título** | Figura 1.36.-Cambiando costo de tarifa |
| **Descripción** | Cambiando costo de tarifa |
| **Fecha** | 25-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

En este caso, se cambió el costo original de la tarifa por 40, una vez realizado el cambio, se presiona el botón registrar y el sistema informa que se actualizo satisfactoriamente.


| | |
|---|---|
| **Título** | Figura 1.37.-Actualización tarifa con éxito |
| **Descripción** | Actualización tarifa con éxito |
| **Fecha** | 25-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

### 3.2 Proceso Asignación de Cubículos

Este proceso sirve para almacenar la información de la asignación de cubículos por terapeuta, que se verá reflejado en el cubículo que se escoja, el día de la semana y el horario de disponibilidad.

**Terapeuta.class**

```
/***********************************************
   * funciongetDatos
   *Retorna todos los datos del Terapeuta seleccionado
***********************************************/
        functiongetDatos(){ //getData()
```

```
                if($this->serial_ter!=NULL){

                        $sql =   "SELECT
t.serial_ter,t.serial_usr,t.usu_serial_usr,t.tipo_ter,t.serial_esp,t.estado_civil_ter
                                FROM terapeuta t
                                WHERE serial_ter='".$this->serial_ter."'";
                        //echo($sql);
                $result = $this->db->Execute($sql);

                        if ($result === false)
                                return false;

                        if($result->fields[0]){
                                $this ->serial_ter = $result->fields[0];
                                $this ->serial_usr = $result->fields[1];
                        $this ->usu_serial_usr = $result ->fields[2];
                        $this ->tipo_ter = $result->fields[3];
                        $this ->serial_esp = $result->fields[4];
                        $this ->estado_civil_ter = $result->fields[5];

                                return true;
                        }
                        else
                                return false;

                }else
                        return false;
        }
```

**Horario.class**

```
/***********************************************
   * funciongetTodosHorariosMenosMaximo
   * Retorna todos los Horarios menos el maximo
***********************************************/
        functiongetTodosHorariosMenosMaximo(){ //getProfiles()
        $sql =   "SELECT *
                                FROM Horario h
                WHERE hora_hor< (SELECT MAX(hora_hor) from Horario)
                ORDER BY hora_hor";
```

```
		$result = $this->db ->Execute($sql);
		$num = $result ->RecordCount();
		if($num> 0){
			$arr=array();
			$cont=0;

			do{
				$arr[$cont]=$result->GetRowAssoc(false);
				$result->MoveNext();
				$cont++;
			}while($cont<$num);
		}

		return $arr;
	}
/***********************************************
   * funciongetTodosHorariosMenosMinimo
   * Retorna todos los Horarios menos el mínimo
***********************************************/
	functiongetTodosHorariosMenosMinimo(){ //getProfiles()
	$sql =   "SELECT *
				FROM Horario h
		WHERE hora_hor> (SELECT MIN(hora_hor) from Horario)
		ORDER BY hora_hor";

		$result = $this->db ->Execute($sql);
		$num = $result ->RecordCount();
		if($num> 0){
			$arr=array();
			$cont=0;

			do{
				$arr[$cont]=$result->GetRowAssoc(false);
				$result->MoveNext();
				$cont++;
			}while($cont<$num);
		}
```

```
                return $arr;
        }
```

**Cubiculo.class**

```
/***********************************************
   * funciongetCubiculos
   * Retorna todos las Cubiculos
**********************************************/
        functiongetCubiculos(){ //getProfiles()
        $sql =   "SELECT *
                                FROM Cubiculo
                     ORDER BY nombre_cub";

                $result = $this->db ->Execute($sql);
                $num = $result ->RecordCount();
                if($num> 0){
                        $arr=array();
                        $cont=0;

                        do{
                                $arr[$cont]=$result->GetRowAssoc(false);
                                $result->MoveNext();
                                $cont++;
                        }while($cont<$num);
                }

                return $arr;
        }
```

**Hor_x_ter.class**

```
/***********************************************
   * functionRangoMananaxTerxDia
* Obtiene el rango de tiempo de trabajo de un terapeuta en determinado día
**********************************************/

        functiongetRangoMananaxTerxDia(){
        $sql =   "SELECT h.hora_hor
FROM Horario h, ter_x_hor t
WHERE (
```

```
t.serial_hor_ini = h.serial_hor
OR t.serial_hor_fin = h.serial_hor)
AND t.serial_ter =".$this->serial_ter."
AND t.dia_hor = "'.$this->dia_hor."'
AND h.hora_hor<= '13:00:00'
ORDER BY h.hora_hor
 ";

      // echo($sql);

                  $result = $this->db ->Execute($sql);
                  $num = $result ->RecordCount();
                  if($num> 0){
                        $arr=array();
                        $cont=0;

                        do{
                              $arr[$cont]=$result->GetRowAssoc(false);
                              $result->MoveNext();
                              $cont++;
                        }while($cont<$num);
                  }

                  return $arr;
            }
/**********************************************
* functionRangoTardexTerxDia
   * Obtiene el rango de tiempo de trabajo de un terapeuta en determinado día
**********************************************/

            functiongetRangoTardexTerxDia(){
            $sql =   "SELECT h.hora_hor
FROM Horario h, ter_x_hor t
WHERE (
t.serial_hor_ini = h.serial_hor
OR t.serial_hor_fin = h.serial_hor)
AND t.serial_ter =".$this->serial_ter."
AND t.dia_hor = "'.$this->dia_hor."'
```

```
AND h.hora_hor>= '13:00:00'
ORDER BY h.hora_hor
 ";

//echo($sql);
                $result = $this->db ->Execute($sql);
                $num = $result ->RecordCount();
                if($num> 0){
                        $arr=array();
                        $cont=0;

                        do{
                                $arr[$cont]=$result->GetRowAssoc(false);
                                $result->MoveNext();
                                $cont++;
                        }while($cont<$num);
                }

                return $arr;
        }
```

**fBuscarTerapeuta.php**

```
Request::setInteger('0:message');
      $user = new Usuario($db);


        $users = $user ->getUsuariosTerapeutas();
        if(is_array($users))
                foreach($users as $key=>$u){
                                $users_list [$key] = $u['nombre_usr'].' '.$u['apellido_usr'].' -
'.$u['username_usr'];
                }
        $smarty ->register('users_list,message');
        $smarty->display();
```

**fNuevaAsignacionCubiculo.php**

```
Request::setInteger('0:error');
$serial_ter = $_POST['serial_ter'];
$terapeuta = new Terapeuta($db);
$usuarios=new Usuario($db);

$terapeuta->setSerial_ter($serial_ter);
if($terapeuta->getDatos()){
                        $data['serial_ter'] = $serial_ter;
                        $data['serial_usr'] =$terapeuta->getSerial_usr();

            $usuarios->setSerial_usr($terapeuta->getSerial_usr());
$usuarios->getDatos();
             $data['username_usr'] = $usuarios->getUsername_usr();
            $data['nombre_usr'] = $usuarios->getNombre_usr();
            $data['apellido_usr'] = $usuarios->getApellido_usr();
}
$horario = new Horario($db);
$listahorariosI= $horario->getTodosHorariosMenosMaximo();
$listahorariosF= $horario->getTodosHorariosMenosMinimo();

$cubiculo = new Cubiculo($db);
$cubiculos = $cubiculo ->getCubiculos();
```

```
$smarty->register('data,listahorariosI,listahorariosF,error,cubiculos');
$smarty->display();
```

**pNuevaAsignacionCubiculo.php**

```
$asignacionCubiculo=new Cub_x_ter($db);
 /*****************************************************************************************
  Cargar Los Datos de La cita para no perderlos en caso de Error
*****************************************************************************************/
        $data['username_usr'] = $_POST['username_usr'];
$data['nombre_usr'] = $_POST['nombre_usr'];
        $data['apellido_usr'] =$_POST['apellido_usr'];
$data['serial_cub'] = $_POST['selCubiculo'];
            $data['serial_ter'] = $_POST['serial_ter'];
        $data['dia'] = $_POST['selDia'];
        $data['serial_hor_ini'] = $_POST['selHoraIni'];
        $data['serial_hor_fin'] =$_POST['selHoraFin'];
$horario = new Horario($db);
$listahorariosI= $horario->getTodosHorariosMenosMaximo();
$listahorariosF= $horario->getTodosHorariosMenosMinimo();

$cubiculo = new Cubiculo($db);
$cubiculos = $cubiculo ->getCubiculos();
 /*****************************************************************************************
  Comprobar que le terapeuta se encuentra en el Centro a la hora asignada
*****************************************************************************************/
$horario->setSerial_hor($_POST['selHoraIni']);
$horario->getDatos();
$hora_ini=$horario->getHora_hor();
$horario->setSerial_hor($_POST['selHoraFin']);
```

```
$horario->getDatos();
$hora_fin= $horario->getHora_hor();

$ter_x_hor = new Hor_x_Ter($db);
$ter_x_hor->setDia_hor($data['dia']);
$ter_x_hor->setSerial_ter($data['serial_ter']);
$vectorm=$ter_x_hor->getRangoMananaxTerxDia();
$vectort=$ter_x_hor->getRangoTardexTerxDia();
$disponible="no";
if($hora_ini>=$vectorm[0]['hora_hor'] && $hora_fin<=$vectorm[1]['hora_hor'])
{$disponible="si";}

if($hora_ini>=$vectort[0]['hora_hor'] && $hora_fin<=$vectort[1]['hora_hor'])
{$disponible="si";}


if($vectorm[1]['hora_hor']==$vectort[0]['hora_hor']){
if($hora_ini>=$vectorm[0]['hora_hor'] && $hora_fin<=$vectort[2]['hora_hor'])
   {$disponible="si";}}
if($disponible=="si")
{

 /*********************************************************************************************
  Comprobar que el terapeuta no tiene asignado otro cubiculo en el mismo horario
  *********************************************************************************************/
    $asignacionesDia=$asignacionCubiculo-
>getAsignacionesPorDiaPorTerapeuta($data['dia'],$data['serial_ter']);
```

```
for($i=0;$i<count($asignacionesDia);$i++)
{

if($hora_ini>=$asignacionesDia[$i]['hora_ini'] && $hora_ini<$asignacionesDia[$i]['hora_fin'])
{ $disponible="no";}
if($hora_fin>$asignacionesDia[$i]['hora_ini'] && $hora_fin<=$asignacionesDia[$i]['hora_fin'])
{ $disponible="no";}

if($hora_ini<=$asignacionesDia[$i]['hora_ini'] && $hora_fin>$asignacionesDia[$i]['hora_ini'])
{
 $disponible="no";
}

}

for($i=0;$i<count($asignacionesDia)-1;$i++)
for($j=1;$j<count($asignacionesDia);$j++)
{
if($hora_ini>$asignacionesDia[$i]['hora_fin'] && $hora_ini<$asignacionesDia[$i+1]['hora_ini'])
if($hora_fin>$asignacionesDia[$j]['hora_fin'] && $hora_fin<$asignacionesDia[$j+1]['hora_ini'])
            $disponible="no";

if($hora_ini>$asignacionesDia[$i]['hora_ini'] && $hora_ini<$asignacionesDia[$i]['hora_fin'])
if($hora_fin>$asignacionesDia[$j-1]['hora_fin'] && $hora_fin<$asignacionesDia[$j]['hora_ini'])
            $disponible="no";
   }
if($disponible=="si")
{
```

```
 /*****************************************************************************************
  Comprobar que no se cruze la distribución de cubículos
*****************************************************************************************/
$horario->setSerial_hor($_POST['selHoraIni']);
$horario->getDatos();
$hora_ini=$horario->getHora_hor();
$horario->setSerial_hor($_POST['selHoraFin']);
$horario->getDatos();
$hora_fin= $horario->getHora_hor();

 $asignacionesDia=$asignacionCubiculo-
>getAsignacionesCubiculoPorDia($data['dia'],$data['serial_cub']);

for($i=0;$i<count($asignacionesDia);$i++)
{ //echo($hora_ini.">=".$asignacionesDia[$i]['hora_ini']." &&
".$hora_ini."<".$asignacionesDia[$i]['hora_fin']."<br/>");
  //echo($hora_fin.">".$asignacionesDia[$i]['hora_ini']." &&
".$hora_fin."<=".$asignacionesDia[$i]['hora_fin']."<br/><br/>");

if($hora_ini>=$asignacionesDia[$i]['hora_ini'] && $hora_ini<$asignacionesDia[$i]['hora_fin'])
{ $disponible="no";}
if($hora_fin>$asignacionesDia[$i]['hora_ini'] && $hora_fin<=$asignacionesDia[$i]['hora_fin'])
{ $disponible="no";}

if($hora_ini<=$asignacionesDia[$i]['hora_ini'] && $hora_fin>$asignacionesDia[$i]['hora_ini'])
{
 $disponible="no";

}
```

```
}
for($i=0;$i<count($asignacionesDia)-1;$i++)
for($j=1;$j<count($asignacionesDia);$j++)
{
if($hora_ini>$asignacionesDia[$i]['hora_fin'] && $hora_ini<$asignacionesDia[$i+1]['hora_ini'])
if($hora_fin>$asignacionesDia[$j]['hora_fin'] && $hora_fin<$asignacionesDia[$j+1]['hora_ini'])
            $disponible="no";

if($hora_ini>$asignacionesDia[$i]['hora_ini'] && $hora_ini<$asignacionesDia[$i]['hora_fin'])
if($hora_fin>$asignacionesDia[$j-1]['hora_fin'] && $hora_fin<$asignacionesDia[$j]['hora_ini'])
            $disponible="no";
   }
if($disponible=="si")
{

$asignacionCubiculo->setSerial_cub($_POST['selCubiculo']);
$asignacionCubiculo->setSerial_ter($_POST['serial_ter']);
$asignacionCubiculo->setDia($_POST['selDia']);
$asignacionCubiculo->setSerial_hor_ini($_POST['selHoraIni']);
$asignacionCubiculo->setSerial_hor_fin($_POST['selHoraFin']);
if($asignacionCubiculo->insert()){
                        //Correcto
                         $error=1;
http_redirect('modulos/cubiculos/fBuscarTerapeuta/1'); break;
}

else{
```

```
                    //Incorrecto
                                $error=2;

                     $smarty->register('data,listahorariosI,listahorariosF,cubiculos,error');
                     $smarty->display('modulos/cubiculos/fNuevaAsignacionCubiculo.es.tpl');
                    }
}
else{
                    //Horario ya asignado
                                $error=3;
                     $smarty->register('data,listahorariosI,listahorariosF,cubiculos,error');
                     $smarty->display('modulos/cubiculos/fNuevaAsignacionCubiculo.es.tpl');
                    }

}else{
                    //Terapeuta ya tiene asignado un cubiculo al horario solicitado
                                $error=5;
                     $smarty->register('data,listahorariosI,listahorariosF,cubiculos,error');
                     $smarty->display('modulos/cubiculos/fNuevaAsignacionCubiculo.es.tpl');
                    }


}else{
                    //Terapeuta No  trabaja en el Centro al horario Ingresado
                                $error=4;
                     $smarty->register('data,listahorariosI,listahorariosF,cubiculos,error');
                     $smarty->display('modulos/cubiculos/fNuevaAsignacionCubiculo.es.tpl');
                    }
```

### 3.2.1 Pruebas unitarias

Una vez terminado con todo el proceso de asignación de cubículos, se procede a realizar las pruebas del mismo.

**Asignando un cubículo a un terapeuta**

Primero hay que ingresar las primeras letras sea del nombre o del apellido del terapeuta, y escoger el nombre deseado de la lista. Y se presiona el botón continuar.


| | |
|---|---|
| **Título** | Figura 1.38.-Seleccionando un terapeuta |
| **Descripción** | Seleccionando un terapeuta de la lista desplegable |
| **Fecha** | 26-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

El siguiente paso es escoger el cubículo, el día y el horario a ser utilizado por el terapeuta.


| | |
|---|---|
| **Título** | Figura 1.39.-Seleccionando parámetros de asignación |
| **Descripción** | Seleccionando parámetros de asignación. |
| **Fecha** | 26-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

Una vez que la información ha sido seleccionada por el usuario, se presiona el botón registrar y el sistema despliega un mensaje avisando al usuario que la asignación de cubículo se ha registrado correctamente.


| | |
|---|---|
| **Título** | Figura 1.40.-Mensaje al usuario de correcta asignación |
| **Descripción** | Mensaje al usuario de correcta asignación. |
| **Fecha** | 26-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

# 4. CAPÍTULO IV

En este cuarto capítulo, se separarán las clases dentro del sistema: dependientes e independientes. Después, se realizará las pruebas respectivas de cada clase con su breve descripción de dependencia, y las capturas de pantalla de los resultados esperados de dicha prueba.

## 4.1 Pruebas del Sistema

El plan de pruebas del sistema es un proceso muy necesario dentro del proceso de desarrollo de software que nos indica los resultados esperados de cada módulo del sistema, para el plan de pruebas de integración del sistema, se escogió el plan de pruebas de integración por dependencias ya que este modelo se comienza el ensamblaje del sistema probando aquellas clases, llamadas independientes, que no se relacionan con ninguna otra clase del sistema; también se probará las clases dependientes que necesitan para su funcionalidad la relación con otras clases del sistema.

En el caso del sistema de gestión del Centro de Psicología Aplicada, las clases independientes son las siguientes:

- Especialización
- Servicio
- Demanda
- Cubículos
- Tarifas
- Perfiles
- Pacientes
- Encuestas

Las clases dependientes son:

- Usuarios
- Terapeutas
- Distribución de Cubículos
- Casos
- Citas

- Cobros
- Instituciones

A continuación el plan de pruebas de integración para el Sistema de Gestión del Centro de Psicología Aplicada de la PUCE, para lo cual se seguirá el orden definido en el diagrama de casos de uso:

### 4.1.0 Caso de Prueba: F 12.0 Administración de Cubículos

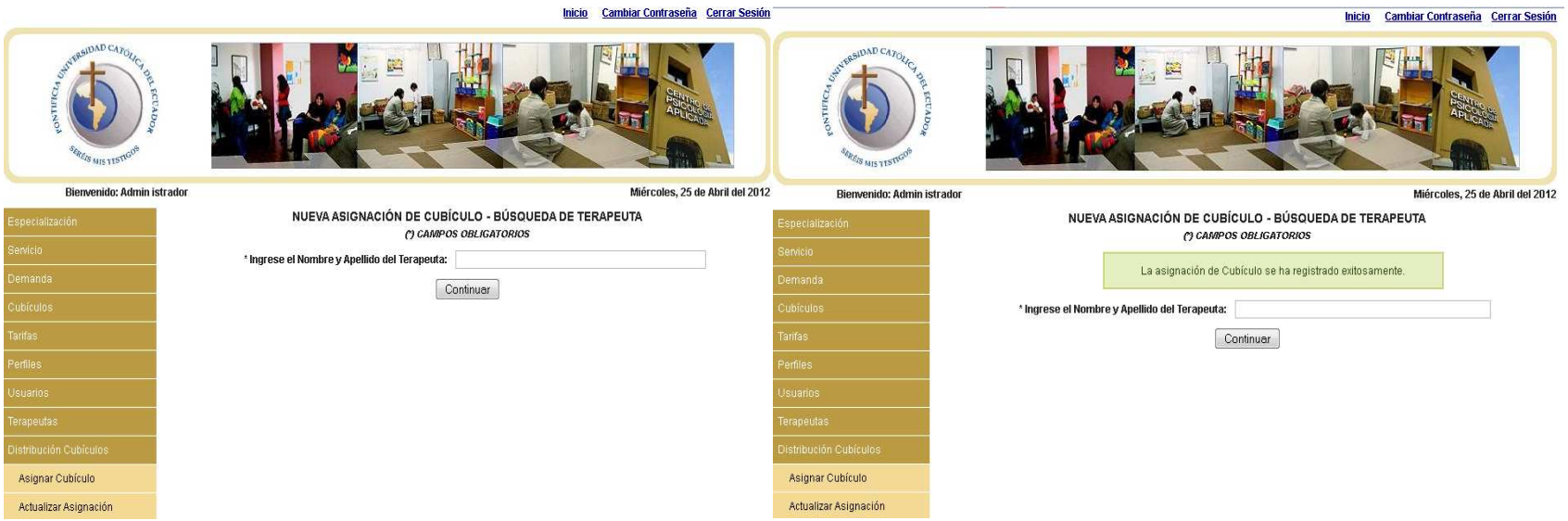
| Entradas | Resultados esperados | Casos de uso |
|---|---|---|
| Seleccionar Cubículos | Desplegar pagina para:<br>• Nuevo Cubículo<br>• Actualizar Cubículo | <br>F 11.1<br>F 11.2 |
| Seleccionar Distribución de Cubículos | Desplegar pagina para:<br>• Asignar Cubículo<br>• Actualizar Asignación<br>o Buscar por Cubículo<br>o Buscar por Terapeuta | <br>F 11.3<br><br>F 11.4<br><br>F 11.5 |

**Título** Tabla 1.27.- Administración de Cubículos
**Descripción** Tabla del Plan de Pruebas de Administración de Cubículos
**Fecha** 01-06-2012
**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo

Como se observa en la tabla contiene los resultados esperados cuando se realiza alguna actividad en el módulo Cubículo, a continuación las pruebas para este módulo:


| | |
|---|---|
| **Título** | Figura 1.53.- Resultados Esperados de Ingresar Cubículo |
| **Descripción** | Los resultados esperados al momento de realizar un ingreso de Cubículo |
| **Fecha** | 15-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |


| | |
|---|---|
| **Título** | Figura 1.53.1.- Resultados Esperados de Actualizar Cubículo |
| **Descripción** | Los resultados esperados al momento de realizar una actualización de Cubículo |
| **Fecha** | 15-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |




| | |
|---|---|
| **Título** | Figura 1.53.2.- Resultados Esperados de Asignar Cubículo |
| **Descripción** | Los resultados esperados al momento de realizar una asignación de Cubículo |
| **Fecha** | 15-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |


**Título** Figura 1.53.3.- Resultados Esperados de Actualizar Asignación Cubículo

**Descripción** Los resultados esperados al momento de realizar una actualización de Asignación por Cubículo

**Fecha** 15-04-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo


| | |
|---|---|
| **Título** | Figura 1.53.4.- Resultados Esperados de Actualizar Asignación Terapeuta |
| **Descripción** | Los resultados esperados al momento de realizar una actualización de Asignación por Terapeuta |
| **Fecha** | 15-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

### 4.1.1 Caso de Prueba: F 13.0 Administración de Tarifa

| Entradas | Resultados esperados | Casos de uso |
|---|---|---|
| Seleccionar Tarifas | Desplegar pagina para:<br>• Nueva Tarifa<br>• Actualizar Tarifa | <br>F 12.1<br>F 12.2 |

**Título** Tabla 1.28.- Administración de Tarifa

**Descripción** Tabla del Plan de Pruebas de Administración de Tarifa

**Fecha** 01-06-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo


**Título** Figura 1.54.- Resultados Esperados de Ingresar una Tarifa

**Descripción** Los resultados esperados al momento de realizar un ingreso de una Tarifa

**Fecha** 15-04-2012

**Copia o Fuente** Realizado Por Mario Granja, Francisco Gordillo


| | |
|---|---|
| **Título** | Figura 1.54.1.- Resultados Esperados de Actualizar Tarifa |
| **Descripción** | Los resultados esperados al momento de realizar una actualización de Tarifa |
| **Fecha** | 15-04-2012 |
| **Copia o Fuente** | Realizado Por Mario Granja, Francisco Gordillo |

# 5. CAPÍTULO V

Después de haber finalizado el presente trabajo y haber realizado pruebas sobre el mismo, se pudieron obtener las siguientes conclusiones y recomendaciones:

## 5.1 Conclusiones

- Una información actualiza de los pacientes en el Centro de Psicología Aplicada, facilita la tarea de localizar, o recordar pacientes tratados.
- Generando un sistema automatizado de administraciones se facilita el manejo de información de los pacientes del centro de Psicología.
- El director del Centro de Psicología Aplicada mediante reportes podrá: distribuir equitativamente el trabajo de los terapeutas con casos asignados, y analizar como es el desempeño de cada terapeuta al cerrar un caso.
- Con la herramienta de horario asignado a cada terapeuta, se controlará la puntualidad en ingreso.
- Con la metodología de toma de requerimientos W2000, se pudo levantar los requerimientos, y esto ayudó a comprender las diferentes funcionalidades del Sistema.
- Con un patrón en el texto, como en el desarrollo de software con estándares de calidad, se facilita la comprensión de cada función y variable dentro del Sistema.
- Una etapa muy significativa dentro del proceso de desarrollo es la documentación y elaboración de diagramas UML, ya que, sí en un futuro se necesita realizar cambios, solo con el acceso a visualizar el documento, se podrá comprender como esta estructurado cada modulo del Sistema.
- La sección de plan de pruebas del Sistema permite detectar errores, como comprobar que las funcionalidades funcionen correctamente.
- En el desarrollo del software, se debe tener una comunicación continua con el cliente para cumplir con todos los requerimientos y cambios impuestos por el cliente.
- Al utilizar una arquitectura de tres capas, se permite visualizar de forma clara, y aumenta la seguridad contra ataques maliciosos. También, se observa cómo se transmite la información desde la capa de manejo de datos, hasta la capa de GUI, ó interfaz de usuario.
- Usando la técnica de recursión: reusando código y clases, se facilita el trabajo de programación al convocar métodos de las clases en los demás módulos.
- Con la implementación del sistema, el tiempo de consulta de información se disminuye en 70%, Anexo C.

## 5.2 Recomendaciones

- Antes de implantar el sistema se debe verificar que los servidores tengan la capacidad de almacenamiento necesaria para que cumpla con lo requerimientos y que la aplicación funcione correctamente.
- Se recomienda tener un ambiente de producción así como un ambiente de pruebas en caso que se necesite hacer un cambio en el futuro.
- Se recomienda tener un servidor extra que aplique la técnica de replicación de datos en caso de contingencia de la base puesta en producción.
- Se recomiendo aplicar la virtualización de servidores para reducir costos de nivel de hardware en caso que aumente una cantidad considerable del personal del CPA.
- Es una buena práctica dar mantenimiento al sistema en un año por el mismo motivo de la futura expansión del CPA.
- Al programar en php una herramienta a considerar para ayudar al programador es el IDE Netbeans el cual ofrece características que se integran con php como edición de código fuente al reconocer la sintaxis de PHP 5.3, reconocimiento y carga de clases, métodos y objetos, para acelerar la programación, se integra con el motor de base de datos MySQL al permitir administrarlo en el mismo entorno, se puede integrar con sistemas de control de versiones siendo una buena opción para desarrollar en este lenguaje.

## 5.3 Bibliografía

### Libros


- BOOCH G., RUMBAUGH J., JACOBSON I., (1999), “El Lenguaje Unificado de Modelado”, Ed. Addison Wesley Iberoamericana.
- BOOCH G., BENJAMIN/CUMMINGS, (1994), “Object-Oriented Analysis and Design”.
- BOOCH G., JACOBSON I. and RUMBAUGH J., (1997), “The UML Specification DocumentRational”, Ed. Software Corp.,
- I. JACOBSON. (1992) , “Object-Oriented Software Engineering: A Use Case Driven Approach”, Addison-Wesley.
- C. LARMAN (1999), “UML y Patrones” , Prentice Hall.
- RUMBAUGH J., (2004) , “Object-Oriented Modeling and Design”, Prentice- Hall.
- UML Resource Center. Rational Software. http://www.rational.com/uml/

- SI ALHIRSinan, (2006), “Learning UML, O’Really”.
- RUMBAUGH J., JACOBSON I., BOOCH G.,(1999), “The Unified Modeling Language Reference Manual” , ADDISON-WESLEY.
- HUMPHREY, WATTS S, (2005), “PSP(sm): A Self-Improvement Process for Software Engineers (SEI Series in Software Engineering)” , Ed. Addison-Wesley Professional, Estados Unidos.
- HUMPHREY, WATTS S, (2004), “Introduction to the Team Software Process(sm)” , Ed. Addison-Wesley Professional, Primera Edición, Estados Unidos.
- SCHACH, STEPHENR., (2004), “Object-Oriented and Classical Software Engineering” , Ed. McGraw-Hill Science/Engineering/Math, 6ta. Edición, Estados Unidos.
- ERIKSSON HANS-ERIK, PENKER M., LYONS B., FADO D.,(2006), “UML 2 Toolkit”, Ed. Wiley Publishing INC., Estados Unido.
- HANNA L. SCHNEIDER, LILLI M. HUBER, (2008), “Social Networks: Development, Evaluation and Influence”, Nova Science Publishers.
- DE LA SERNA Manuel Cebrián, GÓNGORA ROJAS Andrea, (2003) , “Enseñanza virtual para la innovación universitaria”, Narcea Ediciones.


## Internet


- http://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos, Programación orientada a objetos, Última modificación 30 Diciembre del 2011 a las 04.
- http://www.monografias.com/trabajos/objetos/objetos.shtml, Definición y Características de Objetos en POO.
- http://es.wikipedia.org/wiki/PHP, Utilidades y Características de PHP, Última Modificación 13 Enero 2012 a las 21:51.
- http://es.wikipedia.org/wiki/Servidor_HTTP_Apache, Http servidor Web Apache, Ultima Modificación 28 Nov. 2011 a las 19:24.
- http://www.smarty.net/about_smarty, Definición de Smarty.
- http://es.wikipedia.org/wiki/Smarty, Definición de Smarty, Modificado: 8 de Nov. Del 2011 a las 22:09.
- http://es.wikipedia.org/wiki/Symfony, Detalles y Características de Synfony, Modificado: 7 de Feb. De 2012.
- http://tuxpuc.pucp.edu.pe/articulo/comparativa-de-frameworks-en-php-cakephp-symfony-y-zend-framework,Comparación de framework entre Cake PHP con Synfony y Zend, Modificado: 2 de Sep. Del 2007.
- http://es.wikipedia.org/wiki/MySQL, DefinicióMYSQL, Modificado: 16 de Enero de 2012 a la 01:16
- http://www.latindevelopers.com/articulos/sql-server/diferencias-entre-mysql-y-sql-server.php, Diferencias de SQLSERVER con MySQL, Modificado: 08Sep. Del 2011.

- http://dev.mysql.com/tech-resources/articles/move_from_microsoft_SQL_Server.html, Razones de cambiar de SQLSERVER a MySQL, Modificado: 2012.
- http://es.wikipedia.org/wiki/Microsoft_SQL_Server,Caracteristicas y detalles de SQL SERVER, Última Actualización 24 de Mar. Del 2012.
- http://es.wikipedia.org/wiki/RPC, Definición RPC, Ultima Actualización 31 de Oct. Del 2011 a las 12:06
- ftp://ftp.sybase.cz/PD/153/readme.html, Características de PowerDesigner, Modificado: 24 de Febrero de 2010.
- http://www.slideshare.net/Helmilpa/estudio-comparativo-de-php-aspnet-y-java,Comparación entre lenguajes de programación WEB, Modificado: 22 de Noviembre de 2010
- http://es.scribd.com/doc/59013250/21/Metodologias-para-el-Desarrollo-de-SIW, Características de metodologías de desarrollo.
- http://www.lsi.us.es/docencia/get.php?id=2086, Explicación y comparación entre metodologías de Toma de Requerimientos, Modificado: 2007.
- eqaula.org/eva/file.php/2022/_000028_.pdf,Estudio comparativo de metodologías WEB, Modificado: Sevilla Diciembre del 2002.