Universidad Autónoma del Estado de Hidalgo.

INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA.

LIC. EN SISTEMAS COMPUTACIONALES.

Desarrollo de sistema de información orientado a web mediante Rational Unified Process.

Caso de estudio: ***Svelta - Clínica de reducción de peso.***

Tesis que presenta

Rafael Augusto Miranda Aguilar

para obtener el Título de

Licenciado en Sistemas Computacionales

Asesor:

M.C. Ismael Domínguez Jiménez.

Co-asesor:

M.C. David Hernández Sánchez.

Mineral de la Reforma, Hidalgo Marzo, 2009.

Agradecimientos.

A **Dios y a la vida misma**, por darme la oportunidad de llegar hasta este punto de mi vida, satisfecho de los logros y del porvenir tan dichoso.

A mi papá, **Juan Rafael Miranda García**, a mi mamá, **Amalia Aguilar Torrentera**, y a mi hermano, **César Arturo Miranda Aguilar**, que siempre ha estado conmigo y sin su apoyo no hubiera sido posible mi realización como profesionista. Ustedes son mi orgullo, y el motivo principal de seguir mejorando como ser humano.

A mis profesores, por darme las armas necesarias para desempeñarme correctamente en el ámbito profesional. Un sincero agradecimiento al **M.C. Luis Heriberto García Islas**, quien es un gran amigo y mentor, no sólo académicamente hablando. Un honor haberlo tenido como profesor.

Al **Dr. Rafael Ordoñez Trejo** y la **Dra. Angélica Sánchez Pérez**, por abrirme las puertas de su negocio para darme la oportunidad de realizar este trabajo.

Al **Biol. Hugo Ramírez Rivera** y al **L.C. Martín Sergio Gómez Sánchez**, cuyo apoyo y flexibilidad me permitieron concluir esta tesis mientras continuaba con mi vida laboral. Mi más profundo reconocimiento.

A mis amigos y amigas, cuyos momentos juntos jamás serán olvidados en mi mente; especialmente, agradezco a **Marco Antonio Porras**

Álvarez y a **Jorge Alberto Gómez Mendoza**, por su apoyo y amistad durante el trasncurso de la carrera, y a **Antonio de Jesús Cruz Pérez**, cuya amistad me hizo darme cuenta de que valemos mucho más por lo que podemos enseñar a los demás sin importar nuestras circunstancias.

A todos ustedes, mil gracias.

Índice general

Índice de figuras

Antecedentes de la organización.

En este apartado se presenta una descripción del caso de estudio para el presente trabajo: el negocio denominado *Svelta - Clínica de Reducción de Peso*.

Historia.

Svelta - Clínica de Reducción de Peso es una clínica que se dedica a la reducción y moldeamiento corporal, la cual surge en septiembre del 2006, inicialmente bajo el nombre de *Cabina Reductiva*, y con dirección en Boulevard Lomas Residencial #113, Pachuca de Soto, Hidalgo, después de que Alejandra Maycotte Lamego concluyó la carrera de cosmeatra y decidió por medio de la electroescultura ayudar a la gente que tiene sobrepeso y desea moldear su figura. La empresa contaba con dos aparatos especializados en la reducción de peso y dos cabinas con dos camas.

Posteriormente, la empresa fue creciendo gracias al éxito que comenzó a tener y se tuvo la oportunidad de obtener equipo más sofisticado que los anteriores, el cual ayudaba también al moldeamiento del cuerpo de acuerdo a ciertas frecuencias de pulsos eléctricos dirigidos en ciertas partes a reducir. Con la obtención del nuevo equipamiento se logró atender a una mayor cantidad de personas. *Svelta* cambió de ubicación en febrero del 2007 a Everardo Márquez #209, para así ampliar las dimensiones de las cabinas y, de esta forma, el cliente pudiera tener más espacio durante la toma de su

sesión.

En julio del 2007, por razones de causa personal, la cosmeatra Alejandra Maycotte Lamego tuvo que dejar la empresa y quedó a cargo del doctor Rafael Ordoñez Trejo. Al estudiar y analizar el funcionamiento de la empresa, el doctor Rafael Ordoñez Trejo incorporó a una nueva terapeuta, la doctora Angélica Sánchez Pérez, y amplió todos sus sistemas de tratamiento, incorporando nuevos aparatos con más funciones y con mejores resultados que los anteriores; capacitó a las terapeutas para ofrecer otro tipo de tratamientos como mesoterapia, hieloterapia, entre otros.

Actualmente, la organización sigue brindando servicio de calidad a toda la población y cada vez con más formas de combatir sus problemas y poder contar con una mejor salud física y mental.

Misión.

Ayudar a la gente a solucionar sus problemas en materia de sobrepeso, identificando las posibles causas que dan origen a ello, y creándole hábitos, tales como alimentación balanceada y práctica de ejercicio, de acuerdo a cada constitución física, para que pueda mejorar su calidad de vida, a un precio accesible de acuerdo a sus posibilidades económicas.

Visión.

Ampliar y mejorar los métodos de moldeamiento y reducción corporal que ofrece como organización, para ser la empresa número uno de su clase, con ayuda de diversos medios electrónicos, tales como Internet, y crear sucursales para convertirse en la vanguardia estatal de reducción de peso, frente a sus numerosos competidores.

Modus operandi.

La organización no cuenta actualmente con página web ni con email institucional para dar a conocer sus servicios o controlar su manejo; mediante *spots*

radiofónicos promocionales en la estación 106.1[1] de frecuencia modulada, los clientes potenciales, ya sea que acudan al negocio o llamen al número telefónico en vigencia (actualmente, 21-21-616), piden la información necesaria a personal acerca del tratamiento. La labor principal del personal es convencer a los clientes para que decidan integrase al programa de reducción y moldeamiento corporal, el cual es conocido como *Plan de Nutrición de 1200 calorías*; éste consiste en:

- Sesión de terapia elegida durante 40 minutos con el aparato colocado en el área del cuerpo que se desea moldear y/o reducir.
- Sesión de ejercicio durante 10 minutos para eliminar grasa corporal.
- Seguir la dieta indicada por el especialista.

El cliente firma contrato, una vez que acepta cláusulas tanto de la mecánica del tratamiento como después de haber aprobado un examen médico previo, y elige su forma de pago; el monto dependerá si es temporada normal (10 sesiones por $3,500.00) o de oferta (se dan 2 paquetes de 10 sesiones al precio de una, que puede ser en dos modalidades: 10 sesiones para cada una de las 2 personas, o 20 sesiones la misma persona). El cliente creará su expediente una vez que ingresa al primer tratamiento. El personal registra el cierre de venta a su nombre, para futuras remuneraciones, y a partir de cada 10 ingresos nuevos a nombre de un empleado del personal, le otorga un determinado porcentaje de comisión al susodicho.

Los clientes calendarizan su(s) próxima(s) sesión(es) al término de una o desde el inicio del tratamiento, y el personal registra dicha calendarización. Únicamente se pueden calendarizar 3 sesiones por semana. Los datos que se deben considerar para la sesión son: la disponibilidad de cabinas, aparte de la fecha y hora en que se desee llevar a cabo. Al término de cada sesión, el personal registra las medidas obtenidas.

Se debe cancelar una sesión con 24 horas de anticipación a la fecha acordada para tener derecho a una recalendarización; caso contrario, se considera como sesión dada. Además, el cliente puede tomar más de una vez el tratamiento, u otro tratamiento al término del previo. Los tipos

[1] Anteriormente, *EXA FM*; actualmente, *Neurótica punto FM*

de tratamiento que ofrece *Svelta* se limitan a reducciones y moldeamiento corporales en las siguientes áreas:

- Brazos.
- Tórax.
- Piernas.
- Cadera.
- Cintura.

Objetivos estratégicos.

- Proporcionar servicios de calidad a los clientes para mejorar su calidad de vida, abarcando las diferentes causas que originan el padecimiento y el tratamiento óptimo para cada paciente.
- Aprovechar de la mejor manera la tecnología vigente en el mercado y los conocimientos médicos de quienes laboran en la empresa para convertirse en una organización de vanguardia en su campo.
- Crear paquetes de tratamientos que se ajusten a las posibilidades económicas de cada persona.
- Monopolizar los *spots* de radio en estaciones de alta audiencia para ganar mayor clientela potencial.

Problemática.

Durante las observaciones de campo realizadas dentro de las instalaciones del negocio *Svelta - Clínica de Reducción de Peso*, se ha detectado que, con el incremento de la demanda de los servicios que ofrece la organización, el número de expedientes ha crecido y, por ende, su grado de desorganización produce retrasos de tiempo e incomodidad para el personal en el momento de consultarlos o actualizarlos, y en algunos casos más críticos, puede generar pérdida considerable de información para la organización.

Aunado a esto, las operaciones diarias de la organización (ingreso de clientes, actualización de expedientes, calendarización y cancelación de sesiones del tratamiento solicitadas por el cliente) se realizan de forma *manual*, situación que conlleva, a mediano y largo plazo, a la inconsistencia de la información generada, ya que de los 469 expedientes registrados en la empresa, el 20.68 % de ellos cuentan con la información completa, y de este porcentaje, el 54 % se consideran completamente legibles. Además, las situaciones anteriores han conllevado a la provocación de pérdida de tiempo y *cuellos de botella*[2] dentro de los procesos que se llevan a cabo en la organización.

[2] Cualquier aspecto de un proceso o proyecto que retrasa el trabajo.

Propuesta de solución.

El mayor impacto asociado a la problemática descrita anteriormente consiste en almacenar toda la información referente a los clientes, la adquisición de tratamientos que ofrece *Svelta*, y la manipulación diaria de sesiones generadas, y que la información sea consistente y esté al instante accesible y actualizada en lugares físicamente muy distantes (en el caso de la aparición de sucursales), y así reducir los tiempos de operación en el negocio.

Analizando la problemática planteada en el apartado anterior, la propuesta de solución esta enfocada en la mejora del negocio satisfaciendo los siguientes requerimientos:

- Ingresar los datos correspondientes de los clientes que ingresan a la clínica una sola vez, de manera que se almacenen para futuras operaciones o consultas.

- Registrar los datos de sesión del cliente de un determinado día, la hora de inicio de su sesión (la hora de término no se ingresa debido a que la duración de la sesión es siempre de una hora), el pago realizado en la sesión (en caso de ocurrir esta situación, actualizar la caja con los pagos previos correspondientes hechos a dicho tratamiento), los centímetros que redujo durante la sesión en determinada área del cuerpo, y el número del empleado que atiende a dicho paciente, para así crear y actualizar los expedientes de los clientes sin generar inconsistencias de la información.

- Otorgar a cada una de los empleados un número de usuario y una contraseña, para optimizar el cierre de ventas, la manipulación de la calendarización de citas, y la consulta de expedientes de clientes. El dueño del negocio, mediante una contraseña de administrador, podrá realizar cambios en el manejo de paquetes, altas y bajas del personal, entre otras actividades, creando así mayor control del negocio.
- Hacer consultas de manera rápida y eficiente a los expedientes de los clientes.

El desarrollo del presente trabajo pretende resolver los puntos anteriores mediante el empleo de las tecnologías web y las metodologías o modelos de proceso que den como resultado la implementación y documentación del proyecto, el cual consiste en el desarrollo de un sistema de información orientado a la web que facilite la informatización del caso de estudio planteado, y permita un manejo más eficiente de la información dentro del negocio.

Justificación.

Cada vez resulta más frecuente la necesidad de los establecimientos de adquirir tecnología con el fin de asegurar su manejo eficiente y eficaz, y así asegurar que su información sea controlada, consistente e íntegra. México no es la excepción a la regla. Según la *Asociación Mexicana de Internet* [RJOR07] en el año 2007, de las 14.8 millones de PC's instaladas en México, el 59 % estaban conectadas a Internet, y la tasa de crecimiento de PC's con acceso a Internet representaba el 24 % con respecto al año anterior, lo cual denota la velocidad con la que está creciendo la utilización de Internet en México, lo cual beneficia al caso de uso presentado en este trabajo al captar un mayor rango de clientes potenciales que hacen uso de esta tecnología.

El empleo de RUP con UML como paradigma de desarrollo de software resulta útil ya que, al ser un modelo de proceso genérico, puede adaptarse a cualquier tipo de proyecto, (software genérico o para sistemas basados en programación orientada a objetos), y empresa (grandes y pequeñas); por otro lado, ofrece confiabilidad al ser la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos, además de contar con una mayor madurez que otras metodologías web, como es el caso de *Object Oriented Web Solution* (OOWS). En RUP existen diferentes elementos de planificación (plan de desarrollo, plan de iteración, plan de calidad, entre otros) con los que se controla el desarrollo del software, y, a través de un esquema de escalabilidad y gestión de riesgos, se pueden reconocer previamente problemas, y fallos de forma temprana y prevenirlos o corregirlos [UPTG04].

El proyecto se desarrolla utilizando PHP, considerado uno de los lenguajes web más populares de la última década, debido a que es relativamente más fácil de aprender; es una tecnología de código abierto, lo que permite su rápido adelanto, calidad de código superior, enorme biblioteca de código pre-escrito, y amplia gama de apoyo y documentación; es relativamente barato al desarrollar proyectos web, en comparación de otras tecnologías, como Java y Microsoft .NET, y posee capacidad robusta para interactuar con diferentes manejadores de bases de datos (MySQL entre otros) [JUMA08].

La otra parte del proyecto de desarrollará con Ajax ya que la experiencia del usuario en la navegación de una página bajo esta tecnología se vuelve más ágil al no volver a cargarse constantemente durante su interacción, el tiempo de espera para una petición se minimiza lo que trae como consecuencia la reducción de tráfico al servidor [TASE08].

SDTimes[3], en septiembre del 2006, publicó un estudio realizado por *BZ Research*[4] sobre el uso de esta tecnología en el mundo empresarial, en el cual se indica que el uso de Ajax con PHP dentro de proyectos web es de un 25.8 %, superado únicamente por Java con el 50.5 %, y C# con el 40.5 % [CIJA06]. Según una encuesta de Netcraft[5] publicado en abril de 2002, PHP estaba siendo utilizado por más del 24 % de los sitios en Internet de los 37.6 millones de sitios web en todo el mundo [RACR02]; actualmente, se ejecuta en más de 9 millones de sitios, y sigue creciendo a un ritmo explosivo. En los dos últimos años PHP ha tenido una tasa promedio de 6.5 % de crecimiento mensual. Con este indicador de aumento mensual en el uso de PHP, es inevitable pensar en el aumento del uso de esta tecnología con Ajax.

Para acceder a bases de datos es necesario emplear un sistema gestor de bases de datos, como MySQL, que realice las funciones de intérprete entre las aplicaciones y usuarios con las bases de datos. Esta utilidad se traduce en ventajas [POSM05], entre las que podemos mencionar el acceso a

[3] *Software Development Times*, revista *online*.

[4] Filial *online* de investigación de BZ Media LLC que conduce estudios regulares en los asuntos de interés de la industria de desarrollo de software. Los extractos de muchos de estos estudios son publicados en *SDTimes*.

[5] Compañía proveedora de servicios de Internet y análisis de *webserver* y *webhosting* dentro del mercado, ubicada en Bath, Gran Bretaña.

las bases de datos de forma simultánea por varios usuarios y/o aplicaciones, seguridad, en forma de permisos y privilegios, la manipulación de bases de datos de orden de seis mil tablas y alrededor de cincuenta millones de registros, disponibilidad de *Interfaces de Programación de Aplicaciones* (API's) para diferentes plataformas, conectividad entre diferentes máquinas con distintos sistemas operativos, entre otras.

Objetivo general.

Desarrollar un sistema de información web orientado al personal para resolver los inconvenientes de retrasos y falta de integridad o pérdida de información que se presenten en la dinámica del negocio, y así facilitar posteriormente el control de las franquicias potenciales.

Los objetivos específicos que se establecen en este proyecto son:

- Realizar el análisis de la dinámica de la organización, a través de la observación de campo, entrevistas, recolección de evidencias tales como expedientes de pacientes, documentos de ingreso, entre otros.
- Desarrollar una base de datos flexible que soporte al menos un ciclo de vida de aproximadamente 5 años, con los elementos extraídos del análisis previo, y empleando herramientas de tipo *Computer Aided Software Engineering* (CASE) para creación de bases de datos (en este caso, háblese de *CASEStudio ver. 2.25*).
- Aplicar una metodología o modelo de proceso de desarrollo de sistemas genérico, así como la construcción de modelos mediante la herramienta UML, utilizando herramientas de tipo CASE (en este caso, háblese de *Rational Rose Enterprise 2000*).
- Construir la documentación base necesaria para el sistema a desarrollar con fundamento en el esquema de trabajo (metodología o modelo de proceso) elegido.

- Desarrollar un sistema de información empleando las tecnologías web vigentes, como PHP y Ajax.

Introducción.

Hoy en día, las exigencias dentro de los negocios se vuelven cada vez mayores. El crecimiento de los pequeños, medianos y grandes empresarios ha generado que, además que su negocio marche en forma, busquen estar a la vanguardia en su campo mercantil.

Las pequeñas y medianas empresas (PyMEs) acuden cada vez más a los analistas, consultores y personal especializado para lograr la construcción de un sistema que automatice todo o partes estratégicas de su negocio, pues este aspecto les brinda una mayor calidad y eficiencia en el desempeño de sus actividades, aumentando la productividad y generando mejores ganancias.

Existen modelos de procesos y metodologías que permiten cumplir las metas planteadas por las PyMEs, respecto a las soluciones de tecnologías de información que pueden utilizar como herramientas en su proceso.

Uno de los modelos de proceso que permiten establecer un control fiable y bien estructurado de desarrollo de software es el *Rational Unified Process* (RUP), ya que posee la flexibilidad para adpatarse a las necesidades del proyecto presentado en este trabajo de tesis, por su esquema de artefactos, fases y flujos de trabajo.

En el presente trabajo se muestra el desarrollo de un sistema de información vía web, teniendo como caso de estudio el negocio denominado *Svelta - Clínica de Reducción de Peso*. El software desarrollado controla

los aspectos más críticos para la organización en cuestión. Hablando en términos de manejo de información, se emplea una base de datos montada sobre una plataforma basada en la *World Wide Web* (WWW), la cual está manejada por el sistema gestor de bases de datos MySQL. Para el desarrollo del presente sistema, se emplean lenguajes para construcción de páginas web dinámicas, como es el caso de *PHP Hypertext Preprocessor* (PHP) y *Asynchronous JavaScript And XML* (Ajax), éste último en el framework *Xajax*.

El trabajo está estructurado en cuatro capítulos, cada uno abarcando diferentes aspectos:

- En el primer capítulo se presenta el marco teórico que sustenta el desarrollo del proyecto, haciendo hincapié en las herramientas, lenguajes y tecnologías web que se emplean.
- En el segundo capítulo se abarca un panorama general de las tendencias actuales en materia de sistemas web que se relacionan con el proyecto.
- En el tercer capítulo se da la descripción de la metodología empleada, y sus limitantes en el desarrollo de este sistema, y se realiza el análisis de los requerimientos organizacionales, el modelado del negocio (apoyado con la construcción y especificación de diagramas de casos de uso) y la presentación de las vistas estática y dinámica del sistema, mediante diagramas tipo *Unified Modeling Language* (UML) y técnicas de diagramas Entidad-Relación (E-R).
- En el cuarto capítulo se procede a diseñar los componentes del sistema, y realizar las pruebas pertinentes para validar su funcionamiento.

Capítulo

1

Marco conceptual.

En el presente capítulo se abordan los conceptos necesarios para entender la complejidad de un sistema de información, las principales metodologías o modelos de proceso que soportan el desarrollo del mismo y las tecnologías vigentes que están involucradas en su construcción.

1.1. Sistemas de información.

Un sistema de información (*SI*) es un conjunto organizado de elementos interrelacionados entre sí [MAD77], que generalmente consta de:

- Personas.
- Datos.
- Actividades o técnicas de trabajo.
- Recursos materiales en general (típicamente recursos informáticos y de comunicación, aunque no tienen por qué ser de este tipo obligatoriamente).

Todos estos elementos se aúnan para procesar los datos y la información (incluyendo procesos manuales y automáticos) y distribuirla de la manera más adecuada posible en una determinada organización en función de sus objetivos (ver fig. 1.1).

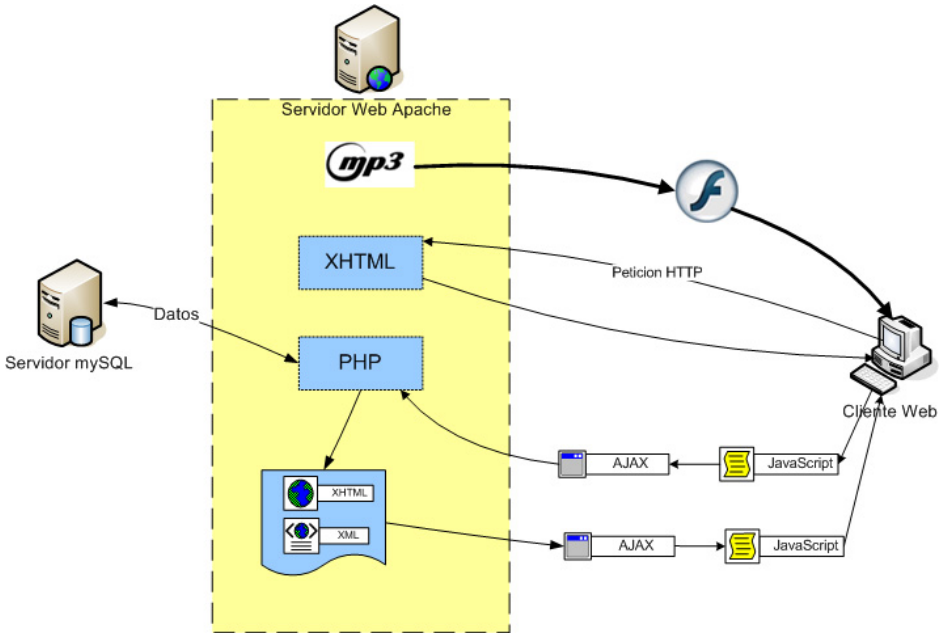


Figura 1.1: Arquitectura de un sistema de información actual.

1.1.1. Características de los sistemas de información.

El nuevo enfoque de los sistemas integrados se manifiesta particularmente en dos grandes perspectivas [MAD77]:

- **Sistema total**.

  Hay una tendencia ininterrumpida hacia el desarrollo de sistemas que interrelacionan muchas funciones dentro de la organización. Lo que interesa es el desempeño global de toda la organización.

- **Proceso de decisión**.

  El sistema total - esto es, el sistema formado por hombres y máquinas - interviene cada vez más en el proceso de decisión. El sistema se hace una pieza vital para el funcionamiento y la administración de la organización.

1.1.2. Ciclo de vida de los sistemas de información.

Cualquier sistema de información va pasando por una serie de fases a lo largo de su vida [BEFR06]. Su ciclo de vida comprende una serie de etapas (ver fig. 1.2), entre las que se encuentran las siguientes:

1. **Planificación**.

   En esta etapa se realizan una serie de tareas previas que influyen decisivamente en la finalización con éxito del proyecto. Estas tareas iniciales se conocen popularmente como el *fuzzy front-end* del proyecto, al no estar sujetas a plazos [BEFR06], las cuales incluyen la determinación del ámbito del proyecto, la realización de un estudio de viabilidad, el análisis de los riesgos asociados al proyecto, una estimación del coste del proyecto, su planificación temporal y la asignación de recursos a las distintas etapas del proyecto.

2. **Análisis**.

   Lo primero que se debe hacer para construir un sistema de información es averiguar *qué es exactamente lo que tiene que hacer*

el sistema. La etapa de análisis en el ciclo de vida del software corresponde al proceso mediante el cual se intenta descubrir qué es lo que realmente se necesita y se llega a una comprensión adecuada de los requerimientos del sistema (las características que el sistema debe poseer) [BEFR06].

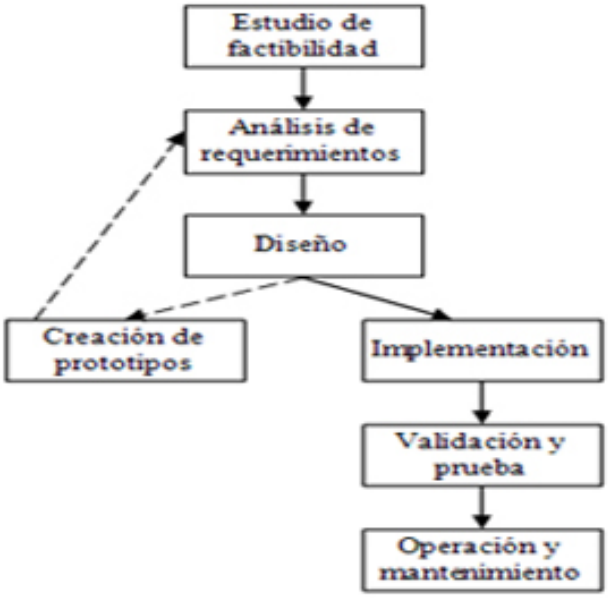


Figura 1.2: Ciclo típico de los sistemas de información.

Esta etapa resulta esencial porque, de no saber con precisión qué es lo que se necesita, ningún proceso de desarrollo permitirá obtenerlo. El problema radica en que ni el cliente sepa qué es exactamente lo que necesita.

3. **Diseño**.

Mientras que los modelos utilizados en la etapa de análisis representan los requisitos del usuario desde distintos puntos de vista (el *qué*), los modelos que se utilizan en la fase de diseño representan las características del sistema que nos permitirán implementarlo de forma efectiva (el *cómo*) [BEFR06]. En esta etapa se estudian posibles

alternativas de implementación para el sistema de información a construir y se decide la estructura general que tendrá el sistema (*diseño arquitectónico*).

4. **Implementación**.

   En esta fase se seleccionan las herramientas adecuadas, un entorno de desarrollo que facilite nuestro trabajo y un lenguaje de programación apropiado para el tipo de sistema que se va a construir. La elección de estas herramientas dependerá en gran parte de las decisiones de diseño que se tomen hasta el momento y del entorno en el que nuestro sistema deberá funcionar. Además de la programación, en esta fase se da la adquisición de recursos para que el sistema funcione [BEFR06].

5. **Pruebas**.

   Tiene como objetivo detectar los errores que se hayan podido cometer en las etapas anteriores del proyecto (y, eventualmente, corregirlos), antes de que el usuario final del sistema los detecte. Las pruebas resultan particularmente delicadas en este sentido, ya que su objetivo es, al fin y al cabo, encontrar defectos [BEFR06].

6. **Instalación (Despliegue)**.

   En esta fase se planifica el entorno en el que el sistema debe funcionar, tanto hardware como software [BEFR06]: equipos necesarios y su configuración física, redes de interconexión entre los equipos y de acceso a sistemas externos, sistemas operativos (actualizados para evitar problemas de seguridad), bibliotecas y componentes suministrados por terceras partes, entre otras situaciones. Si el sistema reemplaza a un sistema anterior o se despliega paulatinamente en distintas fases, también hemos de planificar cuidadosamente la transición del sistema antiguo al nuevo de forma que sus usuarios no sufran una disrupción en el funcionamiento del sistema.

7. **Uso y mantenimiento**.

   La etapa de mantenimiento consume típicamente del 40 al 80

por ciento de los recursos de una empresa de desarrollo de software. De hecho es probablemente la etapa más importante del ciclo de vida del software [BEFR06]. Dada la naturaleza del software, el mantenimiento incluye tres facetas diferentes:

- Eliminar los defectos que se detecten durante su vida útil (**mantenimiento correctivo**).
- Adaptarlo a nuevas necesidades (**mantenimiento adaptativo**), cuando el sistema ha de funcionar sobre una nueva versión del sistema operativo o en un entorno hardware diferente, por ejemplo.
- Añadirle nueva funcionalidad (**mantenimiento perfectivo**), cuando se proponen características deseables que supondrían una mejora del sistema ya existente.

Muchos sistemas de administración basados en la computadora se han diseñado durante los últimos años aplicando estos principios. No todos han logrado el buen éxito que se esperaba. La clave del buen éxito no es simplemente el desarrollo de un sistema *en línea* ni la integración de varios sistemas en uno mayor [MAD77]; está en el desarrollo de sistemas capaces de atender una mayor parte de las necesidades reales de la administración y de hacerlo más eficientemente que un conjunto de sistemas individuales. Al pretender satisfacer estos objetivos, se han encontrado muchos problemas, y los principales parecen corresponder a dos categorías:

- **Obsolencia**.

  Una vez instalado el sistema administrativo, resulta difícil y costoso modificarlo o mejorarlo. Los negocios son dinámicos y varían continuamente. Los sistemas administrativos han demostrado a menudo ser incapaces de seguir el ritmo de cambio de la organización. por consecuencia, el sistema pronto se hace obsoleto y deja de responder a las necesidades de la organización.

- **Programas de desarrollo**.

  Es difícil administrar el desarrollo de un sistema, porque éste por lo general ha requerido más tiempo, esfuerzo y gasto de lo previsto. No ha sido posible prever ni definir problemas potenciales de desarrollo con la anticipación suficiente para evitar demoras en los programas.

1.1.3. Modelos de proceso.

Hasta hace poco, la producción de software era realizada con un enfoque *artístico*, a diferencia de un enfoque industrial. Ante la constante presencia de proyectos fallidos, y con el objeto de mejorar la calidad de los productos, en los últimos años las organizaciones introdujeron los métodos de ingeniería de software [RUM05].

A partir de estos, se formalizó el enfoque de ingeniería de producto para desarrollar software. Factores como la gloablización han obligado a las organizaciones a contar con marcos de trabajo que las ayuden a hacer las cosas de manera más eficiente. Fue entonces que se incorporó la ingeniería de procesos de desarrollo de software.

Según la definición brindada en [RUM05], un proceso de desarrollo de software es un conjunto de personas, estructuras de organización, reglas, políticas, actividades y sus procedimientos, componentes de software, metodologías y herramientas utilizadas o creadas específicamente para definir, desarrollar, ofrecer un servicio, innovar y extender un producto de software. El proceso de software posee las siguientes ventajas:

1. Permite estandarizar esfuerzos, promover reuso, repetición y consistencia entre proyectos.

2. Provee la oportunidad de introducir mejores prácticas de la industria.

3. Permite entender que las herramientas deben ser utilizadas para soportar un proceso.

4. Establece la base para una mayor consistencia y mejoras futuras.

5. Define cómo manejar los cambios y liberaciones a sistemas de software existentes.

6. Define cómo lograr la transición del software a la operación, y cómo ejecutar los esfuerzos de operación y soporte.

En los siguientes apartados se describen los modelos de proceso más destacados por su eficiencia y madurez.

1.1.3.1. CMM.

De las siglas *Capability Maturity Model*, es un marco que describe elementos clave de procesos efectivos de software [RUM05]. Describe un *camino evolutivo* en cinco niveles de mejora de procesos para lograr su madurez. Cubre prácticas de planeación, ingeniería y administración del desarrollo y mantenimiento de software. Los niveles son:

1. **Inicial**.

   El proceso es caótico e impredicible. El éxito depende de los esfuerzos de individuos.

2. **Repetible**.

   El reto es institucionalizar procesos efectivos de administración de proyectos de software, que permitan a las organizaciones repetir prácticas exitosas, desarrolladas en proyectos previos.

3. **Definido**.

   El proceso estándar para desarrollar y mantener el software en la organización está documentado, incluyendo procesos de administración e ingeniería de software, y estos procesos están integrados.

4. **Administrado**.

   Se establece un conjunto de metas cuantitativas para medir el nivel de calidad y desempeño de los proyectos y del proceso organizacional.

5. **Optimizado**.

   Previene, evita, detecta y resuelve problemas al implementar prácticas proactivas. Permite la mejora continua de procesos.

1.1.3.2. CMMI.

De las siglas *Capability Maturity Model Integration*, proporciona una guía para desarrollar procesos y que ayuda en la evaluación de la madurez de la organización o capacidad de un área de procesos [RUM05]. El modelo esta

representado de forma continua y escalonada; contiene 22 áreas de procesos, y cada una está formada por objetivos específicos, prácticas específicas, objetivos genéricos y prácticas genéricas. CMMI se estructura en cinco niveles de madurez de manera general; dependiendo la forma, es el nombre que adquiere cada nivel:

1. **Inicial**.

   La madurez del proceso es caracterizada por resultados impredecibles.

2. **Administrado**.

   La madurez del proceso se caracteriza por el desempeño repetible del proyecto. El foco clave del proceso está en actividades y prácticas a nivel proyecto.

3. **Definido**.

   La madurez del proceso es caracterizada por mejorar el desempeño del proceso dentro de una organización.

4. **Administrado cuantitativamente**.

   Caracterizado por mejorar el desempeño organizacional.

5. **Optimizado**.

   La madurez del proceso se caracteriza por un desempeño organizacional rápido y configurable, la mejora continua y cuantitativa de procesos.

1.1.3.3. ISO 9001-2000.

Constituye un conjunto de estándares que establecen los *requerimientos* para la gestión de sistemas de calidad [RUM05]. Está estructurado en ocho secciones:

1. Alcance.

2. Normas para consulta.

3. Términos y definiciones.

4. Sistema de gestión de calidad.

5. Responsabilidad de la dirección.

6. Gestión de recursos.

7. Realización del producto.

8. Medida, análisis y mejoras.

Muchas organizaciones han optado por gestionar su sistema de calidad en base a este estándar y obtener una certificación de manera internacional, a pesar de no otorgar un estándar específico para sistemas de desarrollo de software.

1.1.3.4. ISO/IEC 15504.

Estándar internacional que ofrece un marco para la *evaluación* de procesos [RUM05]. Puede ser aplicado a múltiples disciplinas y permite a diferentes comunidades definir sus propios procesos específicos, modelos de referencia o buenas prácticas. Se compone de cinco documentos diferentes:

1. Conceptos y vocabulario.

2. Realización de una evaluación.

3. Guía para la realización de la evaluación.

4. Guía de uso para la mejora del proceso y determinación de la capacidad del proceso.

5. Ejemplar del Modelo de Evaluación del Proceso.

Las organizaciones de software no tienen que seleccionar entre 15504 y su modelo actual, debido a que el primero provee un marco de trabajo en el que los modelos y métodos de evaluación puedan existir de forma armónica; es decir, no está diseñado para utilizarse solo.

1.1.3.5. PSP.

De las siglas *Personal Software Process*, es un proceso diseñado para ayudar a los ingenieros de software a controlar, manejar y mejorar su trabajo. Su filosofía: *la calidad de software depende del trabajo de cada uno de los ingenieros de software* [RUM05]. Está basado en prácticas centradas en CMM, y es usado para estructurar y disciplinar el desarrollo de software. El ingeniero de software puede planear mejor el trabajo, conocer con precisión el desempeño, medir la calidad de productos y mejorar las técnicas.

1.1.3.6. TSP.

De las siglas *Team Software Process*, es un marco para el desarrolo de software que pone énfasis en el proceso, producto y trabajo en equipo. Se basa en PSP y se fundamenta en que el software, en su mayoría, es desarrollado por equipos, por lo que los ingenieros deben saber controlar su trabajo, y después saber trabajar en equipo. Enseña a los ingenieros a desenvolverse como un miembro efetivo del equipo, y a los administradores a guiar y soportar estos equipos [RUM05]. La estrategia de TSP consiste en:

1. Proveer un proceso sencillo basado en PSP.

2. Desarrollar productos en varios ciclos: Lanzamiento, Estrategia, Plan, Requerimientos, Diseño, Implementación, Pruebas y Postmortem.

3. Establecer medidas estándares para calidad y desempeño.

4. Proveer definiciones de roles, y evaluaciones de rol y de equipo.

5. Requerir disciplina de proceso.

6. Proveer guía para manejo de problemas de trabajo en equipo.

1.1.3.7. RUP.

El modelo de proceso *Rational Unified Process* (RUP) se describe con detalle en la sección 1.4, debido a que es el modelo adoptado para el desarrollo del trabajo de tesis presente.

1.1.3.8. MoProSoft.

Modelo de procesos para la industria de software mexicano, pensado en facilitar a las organizaciones dedicadas al desarrollo y mantenimiento de software la adopción de mejores prácticas reconocidas internacionalmente a través de modelos como SWW-CMM, CMMI, TSP, PSP, ISO/IEC 15504, PMBOK y SWEBOK [RUM05]. Sus principales características son:

1. Estructura adaptable: los procesos están organizados en tres categorías que corresponden a la estructura de cualquier organización.
2. Número reducido de procesos: Sólo seis procesos principales y tres subprocesos.
3. Procesos integrados a través de roles y productos.
4. Gestión del negocio como proceso explícito.
5. Integración de la gestión de recursos humanos, de infraestructura y conocimiento.
6. Verificaciones y validaciones explícitas.
7. Manejo de situaciones excepcionales.
8. Productos de trabajo con características mínimas explícitas.
9. Indicadores y propuestas de mediciones para evaluar el cumplimiento de los procesos.
10. Guías de ajuste para adecuar los procesos.

1.1.4. Metodologías web.

Cuando se presenta el desarrollo de un artefacto web, se tiene la posibilidad de elegir entre una serie de metodologías que son exclusivamente orientadas a la web; sin embargo, para el desarrollo de este trabajo se optó por emplear una metodología más madura y popular, debido a que las metodologías orientadas a web aún no se llevan a cabo de una manera asidua, tanto en los entornos académicos como en sectores profesionales.

Como cita Pérez Sánchez [PEJ05], tal vez esto ocurra ya que la mayoría de las personas que toman en serio el proceso de software, están esperando la evolución necesaria para la maduración y suceda lo que pasó con las metodologías que actualmente son las más populares (*Rational Unified Process, eXtreme Programming* y *Agile Programming*), donde fue necesario la unificación (*Unified Modeling Language*, UML) y la toma de mejores prácticas, para que los principales actores de la industria dieran respaldo y ocasionaran su propagación y utilización.

En los siguientes apartados se describen brevemente las metodologías web con vigencia en el mercado.

1.1.5. RMM.

De las siglas *Relational Management Methodology*, esta metodología se emplea para el diseño estructurado y la construccion de sistemas hieprmedia encaminado a facilitar el diseño de sitios web y la integración del mismo con bases de datos, estructura e interfaz de usuario [PEJ05]. Se centra en el diseño, no aborda las fases más tempranas y tardías del ciclo de vida, como la dirección del proyecto, estudios de viabilidad, análisis de requisitos, planificación, evaluación y mantenimiento.

1.1.6. OOHDM.

De las siglas *Object Oriented Hypermedia Design Method*, esta metodología se basa en técnicas de modelado de objetos. Establece que el desarrollo de un hiperdocumento es un proceso de cuatro fases en el que se combinan diferentes estilos de desarrollo, como el incremental, iterativo y prototipado [PEJ05]. Las tres primeras fases son referentes a diseño, en las que se obtiene un conjunto de modelos orientados a objetos que describen el documento que será construido en la última fase.

1.1.7. RMM.

De las siglas *Relational Management Methodology*, esta metodología se emplea para el diseño estructurado y la construccion de sistemas hieprmedia encaminado a facilitar el diseño de sitios web y la integración del mismo con bases de datos, estructura e interfaz de usuario [PEJ05]. Se centra en

el diseño, no aborda las fases más tempranas y tardías del ciclo de vida, como la dirección del proyecto, estudios de viabilidad, análisis de requisitos, planificación, evaluación y mantenimiento.

1.1.8. EORM.

De las siglas *Enhaced Object Relationship Model*, es una metodología que involucra el desarrollo de un hiperdocumento a través de una primera fase de análisis orientado a objetos del sistema, obteniendo un modelo de objetos que refleja la estructura de la información y el comportamiento del sistema [PEJ05]. Considera una segunda fase de diseño durante la cual se procede a modificar el modelo de objetos durante el análisis añadiendo la semántica apropiada a las relaciones entre clases para convertirlas en enlaces hipermedia. Esta metodología garantiza flexibilidad y reutilización.

1.1.9. SOHDM.

De las siglas *Scenario-Based Object Oriented Hypermedia Design Methodology*, es una metodología centrada en el diseño de la navegación e implementación utilizando requisitos obtenidos en una fase previa al diseño [PEJ05]. La utilización de escenarios aumenta la usabilidad de la aplicación. En la primera fase se emplean diagramas *System Scope Diagrams* (SSD) para representar límites del sistema y representar la aplicación objetivo, así como escenarios para identificar los requisitos tanto de información de usuarios como de navegación. Es útil para sitios web de comercio electrónico y empresas virtuales.

1.1.10. VHDM.

De las siglas *View-Based Hypermedia Design Methodology*, es una metodología orientada al diseño de aplicaciones hipermedia basada en vistas de usuario. Las vistas se generan a partir del diagrama E-R y son utilizadas en las dos últimas fases. Resulta efectiva en la integración de bases de datos de empresas con sistemas hipermedia distribuidos en la web [PEJ05].

1.1.11. WSDM.

De las siglas *Web Site Design Method*, es una metodología basada en un conjunto de potenciales visitantes del sitio web a desarrollar. Es más enfocado al usuario del sitio que a la orientación a datos. Los usuarios son clasificados en clases de usuarios y los datos que se acceden se modelan desde el punto de vista de las anteriores clases. Proporciona sitios web más adaptados a cada usuario elevando el nivel de usabilidad y resultando por lo tanto más satisfactorio [PEJ05].

1.1.12. OOWS.

De las siglas *Object Oriented Web Solution*, es una metodología que destaca por la utilización de la notación UML adaptada a OO-Method para definir de manera más precisa un modelo navegacional; es decir, para ampliar los conceptos del modelado OO-Method e introducir semántica en la navegación. Permite introducir una nueva etapa en el modelo conceptual para definir los mapas de navegación de cada agente, y generar automáticamente software para aplicaciones *e-business* en entornos web [PAOS02].

1.1.13. Comparación entre metodologías.

Se presenta en esta sección una tabla comparativa entre las metodologías anteriormente expuestas, acorde con el trabajo de [SIDA01].

Siglas	Técnica de modelado	Proceso	Representación gráfica	Notación	Herramienta de soporte
RMM	E-R[1]	1. Diseño E-R. 2. Diseño *Slice*[2]. 3. Diseño de navegación. 4. Diseño de protocolo de conversión. 5. Diseño de UI[3] 6. Diseño de comportamiento en tiempo de ejecución. 7. Prueba y construcción.	1. Diagrama E-R. 2. Diagrama *Slice*. 3. Diagrama RMDM[4].	1. E-R. 2. Propia.	RMCase
OOHDM	Orientado a Objetos (OO)	1. Diseño conceptual. 2. Diseño navegacional. 3. Diseño abstracto de la UI. 4. Implementación.	1. Diagrama de clases. 2. Diagrama navegacional, ADV y contexto. 3. Diagrama de configuración de ADV.	1. ADVs. 2. OMT / UML. 3. Propia.	OOHDM-web
EORM	OO	1. Clases del entorno de desarrollo. 2. Composición del entorno de desarrollo. 3. Entorno de desarrollo de la UI. 4. Implementación.	1. Diagrama E-R. 2. Diagrama *Slice*. 3. Diagrama RMDM.	OMT18	ONTOS Studio
SOHDM	Escenarios y Vistas OO	1. Análisis del dominio. 2. Modelo en OO. 3. Diseño de la vista. 4. Diseño navegacional. 5. Diseño de implementación. 6. Construcción.	1. Diagrama de escenarios de actividad. 2. Diagrama de estructura de clases. 3. Vista OO 4. Esquema de enlace navegacional. 5. Esquema de páginas.	Propio	

[1] *Entity – Relationship*, Entidad - Relación

[2] Unidades de presentación que aparecen como páginas de una aplicación hipermedia.

[3] *User Interface*, Interfaz de Usuario.

[4] *Relationship Management Data Model*, Modelo de Datos de Administración de Relaciones.

VHDM	OO	1. Análisis de requisitos. 2. Diseño E-R. 3. Vistas de diseño. 4. Diseño de navegación. 5. Mapas. 6. Implementación.	1. Diagrama E-R. 2. Esquema de vista. 3. Esquema navegacional.	1. E-R. 2. Propia.	
WSDM	OO	1. Modelado del usuario. 2. Diseño conceptual. a) Modelo de objetos. b) Diseño navegacional. 3. Diseño de implementación.	1. Diagrama de clases o E-R. 2. Capas de navegación.	1. E-R/OMT. 2. Propia.	
OOWS	OO	1. Especificación del sistema. a) Análisis de requisitos utilizando una aproximación basada en casos de uso. b) Modelado conceptual del sistema. 2. Desarrollo de la solución. a) Capas. b) Mapeo.	UML	1. UML. 2. Propia.	OOWS Suite

1.2. Bases de datos.

Una base de datos es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso [BD08]. En la actualidad, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos tienen formato electrónico, que ofrece un amplio rango de soluciones al problema de almacenamiento de datos.

Las aplicaciones más usuales son para la gestión de empresas e instituciones públicas; también son ampliamente utilizadas en entornos científicos con el objeto de almacenar la información experimental.

Aunque las bases de datos pueden contener muchos tipos de datos, algunos de ellos se encuentran protegidos por las leyes de varios países. Por ejemplo, en España, los datos personales se encuentran protegidos por la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD) [BD08].

1.2.1. Propiedades.

Las bases de datos poseen algunas propiedades implícitas [SIA02] como son:

- Representa algunos aspectos del mundo real, en ocasiones denominado minimundo o Universo del Discurso.
- Es una colección coherente de datos con significados inherentes.
- Se diseña, construye y llena con datos para un propósito específico, destinada a un grupo de usuarios concreto y tiene algunas aplicaciones preconcebidas en las cuales están interesados dichos usuarios.
- Puede tener cualquier tamaño y complejidad.
- Puede mantenerse y crearse manualmente o puede estar informatizada, mediante un conjunto de programas de aplicación o un sistema de gestión de base de datos.

1.2.2. Tipos.

Las bases de datos pueden clasificarse de varias maneras [BD08], de acuerdo al criterio elegido para su clasificación:

- **Según la variabilidad de los datos almacenados.**
  - **Estáticas**.

    También conocidas como *data warehouses*, son bases de datos de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el

comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones.

- **Dinámicas**.

  Bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta. Un ejemplo de esto puede ser la base de datos utilizada en un sistema de información de una tienda de abarrotes, una farmacia, un videoclub, entre otras.

- **Según el contenido.**
  - **Bibliográficas**. Sólo contienen un surrogante (representante) de la fuente primaria, que permite localizarla. El contenido son cifras o números. Por ejemplo, una colección de resultados de análisis de laboratorio, entre otras.
  - **De texto completo**. Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.
  - **Directorios**. Un ejemplo son las guías telefónicas en formato electrónico.
  - ***Bibliotecas* de información Biológica**. Son bases de datos que almacenan diferentes tipos de información proveniente de las ciencias de la vida o médicas. Se pueden considerar en varios subtipos:
    - Aquellas que almacenan secuencias de nucleótidos o proteínas.
    - Las bases de datos de rutas metabólicas.
    - Bases de datos de estructura, comprende los registros de datos experimentales sobre estructuras 3D de biomoléculas.
    - Bases de datos clínicas.
    - Bases de datos bibliográficas (biológicas).

1.2.3. Diseño.

1.2.3.1. Modelos de los datos.

Bajo la estructura de la base de datos se encuentra el modelo de datos: una colección de herramientas conceptuales para describir los datos, las rela-

ciones, la semántica y las restricciones de consistencia [SIA02]. Los diferentes modelos propuestos se clasifican en tres grupos diferentes:

- **Modelo entidad - relación (E-R).**

  Este modelo esta basado en una percepción del mundo real que consta de una serie de objetos llamados *entidades* y de *relaciones* entre dichos objetos [SIA02]. Las entidades se describen en una base de datos mediante un conjunto de atributos. Además de entidades y relaciones, el modelo E-R representa ciertas restricciones que los contenidos de la base de datos debe cumplir; la más importante, la *correspondencia de cardinalidad*, que expresa el número de entidades con las que otra entidad se puede asociar a través de un conjunto de relaciones. Es un modelo *basado en objetos*.

- **Modelo relacional.**

  Utiliza un grupo de tablas para representar los datos y las relaciones entre ellos. Cada tabla está compuesta por varias columnas con un nombre único. Éste es un ejemplo de modelo *basado en registros*, debido a que cada tabla contiene registros de un tipo particular, y cada tipo de registro define un número fijo de campos o atributos. Es el modelo de datos más ampliamente usado, y una gran mayoría de los sistemas de bases de datos actuales se basan en él; sin embargo, puede presentar información duplicada innecesariamente [SIA02].

- **Otros modelos de datos.**

  El **modelo de datos orientado a objetos** se observa como una extensión del modelo E-R con nociones de encapsulación, métodos e identidad de objetos. El **modelo de datos relacional orientado a objetos** combina las características del modelo de datos orientado a objetos y el modelo de datos relacional [SIA02]. Los **modelos de datos semiestructurados** permiten la especificación de datos donde los elementos de datos individuales del mismo tipo pueden tener diferentes conjuntos de atributos. Los modelos de datos **de red y jerárquicos** complicaban la tarea del modelado de datos, por lo que actualmente se usan en el código de bases de datos antiguo que aún está en servicio en algunos lugares.

1.2.3.2. Abstracción de datos.

Uno de los propósitos principales de un sistema de bases de datos es proporcionar a los usuarios una visión *abstracta* de los datos [SIA02]; es decir, que el sistema esconde ciertos detalles de cómo se almacenan y mantienen los datos. Este ocultamiento de complejidad se logra a través de niveles de abstracción para simplificar la interacción de los usuarios con el sistema, los cuales son:

- **Nivel físico.**

  El nivel más bajo de abstracción que describe *cómo* se almacenan realmente los datos, y se describe en detalle las estructuras de datos complejas de bajo nivel.

- **Nivel lógico.**

  Describe *qué* datos se almacenan y qué relaciones existen entre éstos; así, la base de datos completa se describe en términos de un número pequeño de estructuras relativamente simples. Este nivel es empleado por los administradores de bases de datos.

- **Nivel de vistas.**

  Describe sólo parte de la base de datos completa, debido a que los usuarios no necesitan de toda la información contenida en ella.

1.2.4. Sistema Gestor de Bases de Datos (*SGBD*).

En Informática, existen los Sistemas Gestores de Bases de Datos (*SGBD*), que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada [SIA02]. Un *SGBD* es una colección de programas de propósito general que permiten a los usuarios crear y mantener una base de datos, y a su vez, facilitar los procesos de *definición, construcción* y *manipulación* de bases de datos para distintas aplicaciones.

Se entiende por *definición* el especificar los datos, estructuras y restricciones para los datos que se van a almacenar en dicha base; *construcción*

es el proceso de almacenar datos concretos sobre algún medio de almacenamiento controlado por el *SGBD*, y *manipulación* incluye funciones como consultar a la base de datos para recuperar datos específicos, actualizar la base de datos para reflejar cambios ocurridos en el minimundo, y generar informes a partir de los datos [SIA02].

Entre las ventajas de utilizar un *SGBD* se encuentran las siguientes:

- Control de la redundancia.
- Restricción a accesos no autorizados.
- Suministro de almacenamiento persistente de objetos y estructuras de datos de programas.
- Capacidad de realizar inferencias y acciones usando reglas.
- Suministro de múltiples interfaces de usuario.
- Representación de vínculos complejos entre datos.
- Garantizar el cumplimiento de restricciones de integridad.
- Suministro de copias de seguridad y recuperación.

1.2.4.1. Estructura.

Un SGBD se divide en módulos que se encargan de cada una de las responsabilidades del sistema completo; los módulos más importantes son el el **gestor de almacenamiento** y el **procesador de consultas** [SIA02].

El **gestor de almacenamiento** es un módulo de programa que proporciona la interfaz entre los datos de bajo nivel en la base de datos y los programas de aplicación y consultas emitidas al sistema; es decir, *traduce* las diferentes instrucciones LMD a órdenes de un sistema de archivos de bajo nivel. Por lo tanto, es responsable del almacenamiento, recuperación y actualización de los datos en la base de datos.

Los componentes de este módulo incluyen:

- **Gestor de autorización e integridad**, que comprueba que se satisfagan las restricciones de integridad y autorización de acceso a datos.
- **Gestor de transacciones**, que asegura la consistencia de la base de datos.
- **Gestor de archivos**, que reserva el espacio en disco y estructuras de datos para representar la información almacenada en disco.
- **Gestor de memoria intermedia**, que trae los datos del disco a la memoria principal y decide qué datos trata en la memoria caché.

Además, implementa varias estructuras de datos como parte de la implementación física del sistema, como son el **archivo de datos**, el **diccionario de datos** y los **índices** [SIA02].

El **procesador de consultas** ayuda al SGBD a simplificar y facilitar el acceso a los datos. Los componentes de éste son:

- **Intérprete del LDD**, que interpreta las instrucciones del LDD y registra las definiciones en el diccionario de datos.
- **Compilador de LMD**, que traduce las instrucciones del LMD a un plan de evaluación de instrucciones de bajo nivel; además, realiza la optimización de consultas.
- **Motor de evaluación de consultas**, que ejecuta las instrucciones de bajo nivel generadas por el compilador del LMD.

1.2.5. MySQL.

Es un sistema de gestión de bases de datos relacional, *multihilo* y multiusuario con más de seis millones de instalaciones [MYS08]. Apegándose a los criterios de Ullman [ULL04], es la base de datos *open source* más popular y, posiblemente, la mejor del mundo. Fue creado y sigue siendo desarrollado por MySQL AB, una compañía que radica en Suecia, como software libre en un esquema de licenciamiento dual; MySQL AB pertenece a Sun Microsystems desde enero del 2008 [MYS08].

MySQL creció a partir de una base de datos *open source* existente:

mSQL, la cual continúa en fase de desarrollo. Se puede interactuar con MySQL utilizando los lenguajes de programación más populares, como PHP, Perl y Java. Fue escrito en C y C++ y funciona exactamente igual en distintos sistemas operativos. MySQL es capaz de manejar bases de datos de hasta 60,000 tablas de hasta 8 millones de terabytes en algunos sistemas operativos y hasta 1 gigabyte en otros, y más de cinco mil millones de filas [MYS08]. El software de MySQL consta de varios elementos, entre los que destacan el servidor de MySQL (*mysqld*, que se encarga de ejecutar y administrar las bases de datos), el cliente MySQL (*mysql*, que proporciona la interfaz para el servidor) y numerosas utilidades de mantenimiento de otra índole (ver fig. 1.3).

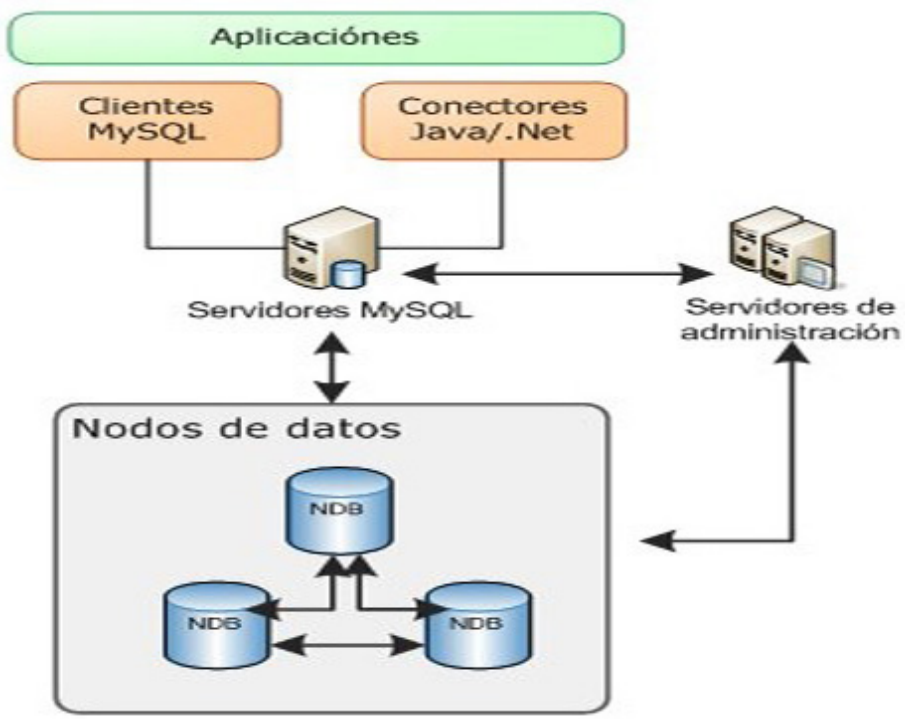


Figura 1.3: Arquitectura del servidor MySQL.

1.2.5.1. Características de la versión 5.0.22

Específicamente hablando de ésta, la última versión de MySQL hasta el momento, se enlistan las características más importantes del mismo [MYS08].

- Un amplio subconjunto de ANSI SQL 99, y varias extensiones.
- Soporte a multiplataforma.
- Procedimientos almacenados.
- *Triggers*.
- Cursores.
- Vistas actualizables.
- Soporte a *VARCHAR*.
- *INFORMATION_SCHEMA*.
- Modo *Strict*.
- Soporte X/Open XA de transacciones distribuidas; transacción en dos fases como parte de esto, utilizando el motor InnoDB de Oracle.
- Motores de almacenamiento independientes (MyISAM para lecturas rápidas, InnoDB para transacciones e integridad referencial).
- Transacciones con los motores de almacenamiento InnoDB, BDB Y Cluster; puntos de recuperación(savepoints) con InnoDB.
- Soporte para SSL.
- *Query caching*.
- SELECT anidados.
- Réplica con un maestro por esclavo, varios esclavos por maestro, sin soporte automático para múltiples maestros por esclavo.
- *Indexing* y buscando campos de texto completos usando el motor de almacenamiento MyISAM.
- *Embedded database library*.
- Soporte completo para Unicode.

1.2.5.2. Características únicas de MySQL.

Las siguientes características son implementadas únicamente por MySQL [MYS08].

- 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 varias de ellas de diferentes conexiones para incrementar el número de transacciones por segundo.

1.2.5.3. Sentencias.

MySQL ofrece una extensa gama de sentencias para generar y manipular bases de datos; algunas de ellas pertencen al conjunto de cláusulas del lenguaje relacional SQL. Para conocer las sentencias empleadas en MySQL, consultar el Apéndice A.

1.3. UML.

El UML es una definición oficial de un lenguaje pictórico con símbolos y relaciones comunes que tienen un significado común [KIP07]. Proveniente de las siglas *Unified Modeling Language*, es uno de los lenguajes mas recientemente inventados por la humanidad, alrededor de 1997. Es llamado un *lenguaje de modelado*, no un método. Los métodos consisten de un lenguaje de modelado y de un proceso [GOCJ04].

Se considera un estándar y que lo que este lenguaje es o no es lo define un consorcio de empresas que constituyen el *Object Management Group* (OMG, Grupo de Administración de Objetos) [BOG99].

UML comprende símbolos y una gramática que define la manera en que se pueden usar éstos símbolos. Una de la metas principales de UML es avanzar en el estado de la integración institucional proporcionando herramientas de interoperabilidad para el modelado visual de objetos. Sin embargo para lograr un intercambio exitoso de modelos de información

entre herramientas, se requirió definir a UML una semántica y una notación [GOCJ04]. Entre las herramientas UML más populares se encuentran *Enterprise Architect*, *Rational Rose*, *Microsoft Visio*, entre otras.

1.3.1. Diferencias entre UML y el Modelo E-R.

Para fines del actual estudio, se reconoce que el diagrama de clases de UML se basa en diagramas E-R; sin embargo, hay algunas diferencias entre ambos que se deben tener presentes. Por citar algunas [SIA02], encontramos las siguientes:

1. UML muestra los conjuntos de entidades como cuadros, a diferencia de E-R, que muestra los atributos dentro del cuadro en lugar de elipses separadas.

2. UML modela *realmente objetos*, mientras que E-R modela *entidades*.

3. Los diagramas de clases pueden describir métodos además de atributos, y E-R no.

4. La ubicación de las restricciones de cardinalidad en UML es exactamente inverso a la ubicación de éstas en E-R.

1.3.2. Diagramas de UML.

Los diagramas en UML se clasifican en dos tipos: los diagramas estructurales y los diagramas de comportamiento.

1.3.2.1. Diagramas estructurales.

Los cuatro diagramas estructurales de UML existen para visualizar, especificar, construir y documentar los aspectos estáticos de un sistema, los cuales son aquellos que representan su esqueleto y andamiaje [BOG99]; éstos se organizan en líneas generales alrededor de los principales grupos de elementos que aparecen al modelar un sistema.

- **Clases**.

  Presenta un conjunto de clases, interfaces, colaboraciones y relaciones entre ellas (ver fig. 1.4). Son los más comunes en el modelado de sistemas orientados a objetos, y los que incluyen clases activas se utilizan para cubrir la vista estática de procesos de un sistema [BOG99].

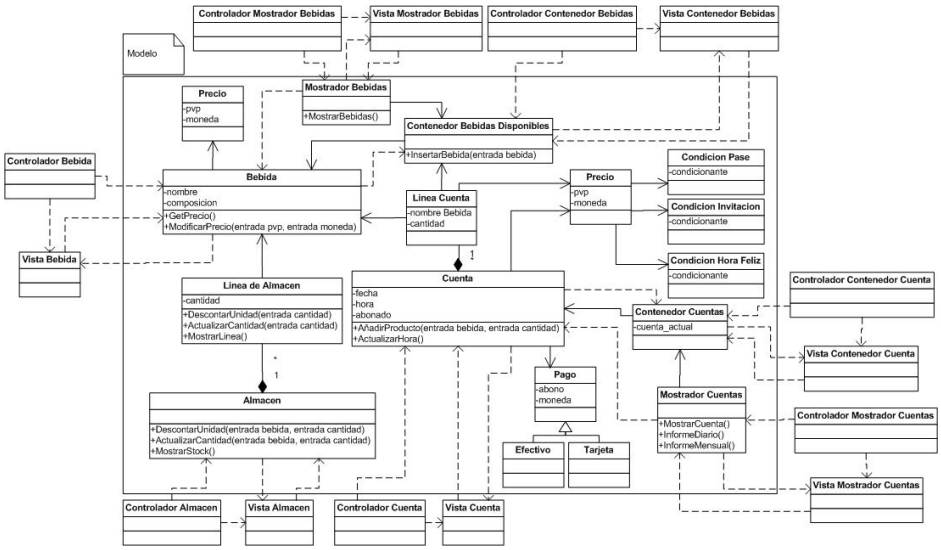


Figura 1.4: Diagrama de clases de bebidas de un almacén.

- **Objetos**.

  Presenta un conjunto de objetos y sus relaciones. Se utilizan para describir estructuras de datos, instantáneas de las instancias de los elementos encontrados en los diagramas de clases (ver fig. 1.5). Describen la vista estática del sistema desde la perspectiva de casos reales o prototípicos [BOG99].

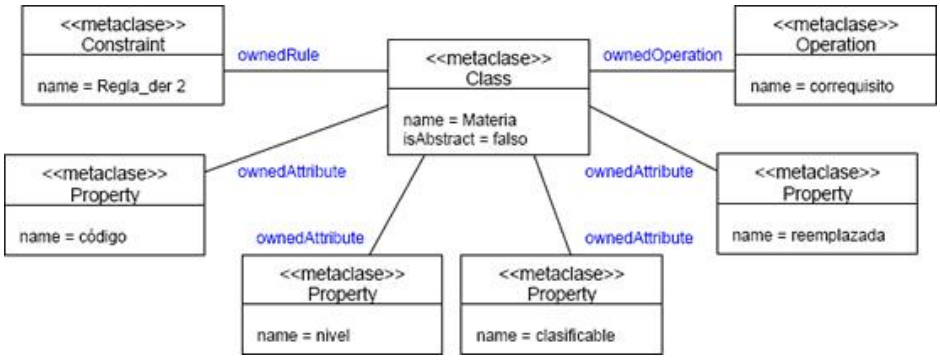


Figura 1.5: Diagrama de objetos de materias escolares.

- **Componentes**.

  Presenta un conjunto de componentes y sus relaciones. Describe la vista de implementación estática de un sistema (ver fig. 1.6); se relaciona con el diagrama de clases en que un componente normalmente se corresponde con una o más clases, interfaces o colaboraciones [BOG99].

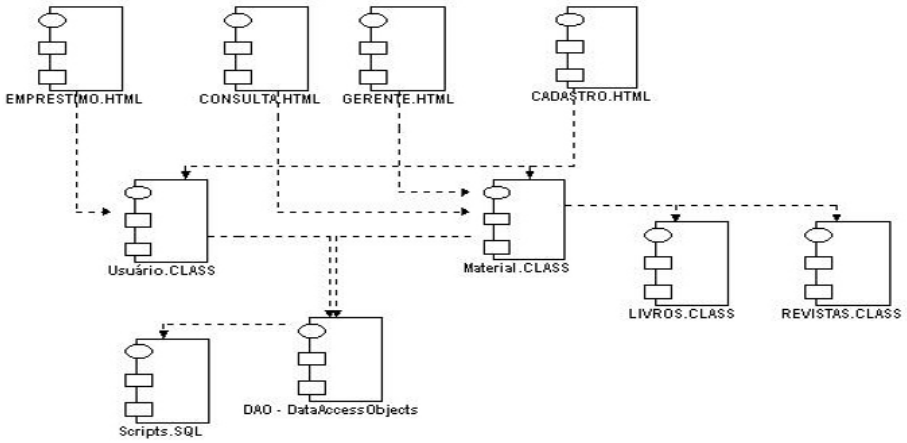


Figura 1.6: Diagrama de componentes de un sistema de biblioteca.

- **Despliegue**.

  Presenta un conjunto de nodos y sus relaciones; describen la vista de despliegue estática de una arquitectura [BOG99]. Se relacionan con los diagramas de componentes en que un nodo normalmente incluye uno o más componentes (ver fig. 1.7).

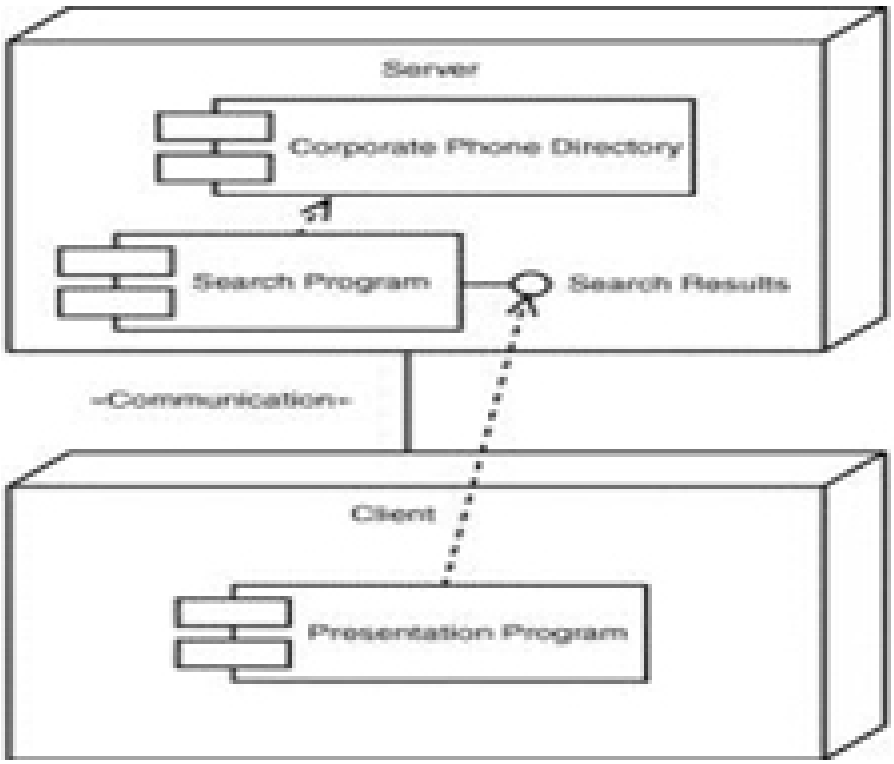


Figura 1.7: Diagrama de despliegue de un sistema de agenda corporativa.

1.3.2.2. Diagramas de comportamiento.

Los cinco diagramas de comportamiento se utilizan para visualizar, especificar, construir y documentar los aspectos dinámicos del sistema, los cuales son aquellos que representan sus partes mutables [BOG99]; éstos se organizan en líneas generales alrededor de las formas principales en que se puede modelar la dinámica de un sistema:

- **Caso de uso**.

  Presenta un conjunto de casos de uso y actores (un tipo especial de clases) y sus relaciones; se emplean para describir la vista de casos de uso estática de un sistema [BOG99]. Son especialmente importantes para organizar y modelar el comportamiento del sistema (ver fig. 1.8).

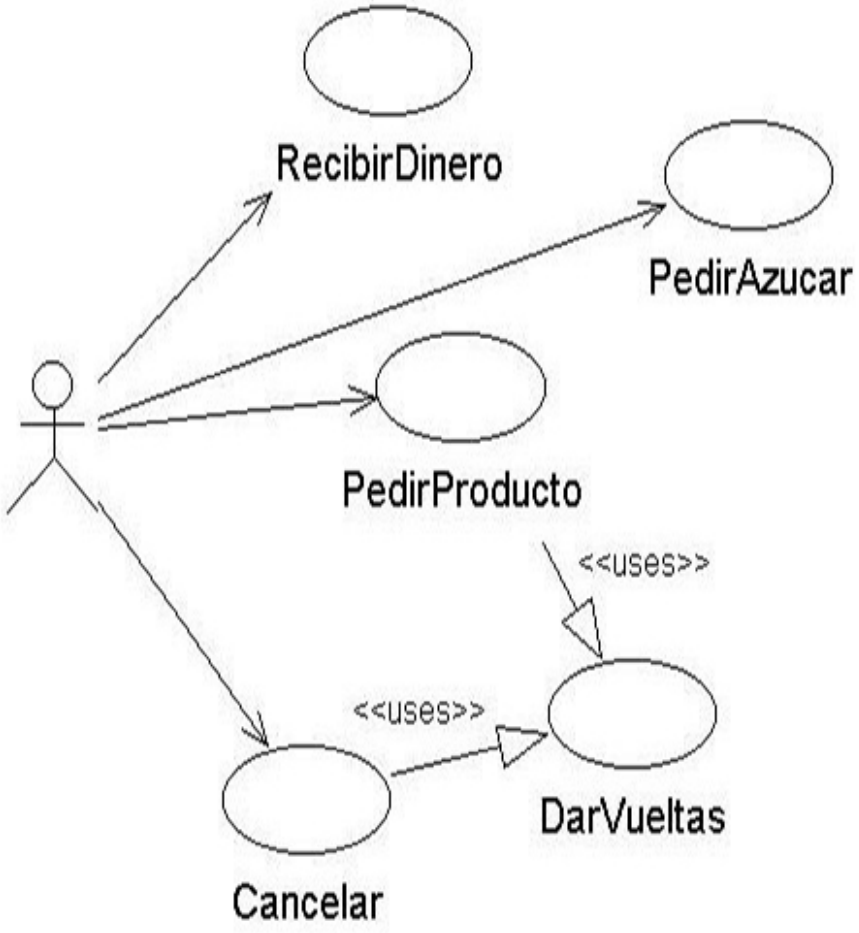


Figura 1.8: Diagrama de casos de uso para comprar un café.

- **Interacción**.

  Existen dos tipos: de *secuencia* y de *colaboración*. Ambos transmiten la misma información, empleando una perspectiva poco diferente; por lo tanto son isomorfos [BOG99]. No es necesario crear los dos tipos de diagramas; se percibe que un diagrama de secuencia es más fácil de leer y más común.

  - **Secuencia**: Presenta un diagrama de interacción que resalta la ordenación temporal de los mensajes; presenta un conjunto de objetos (instancias a nombres o anónimas de clases) y los mensajes enviados y recibidos por ellos (ver fig. 1.9).

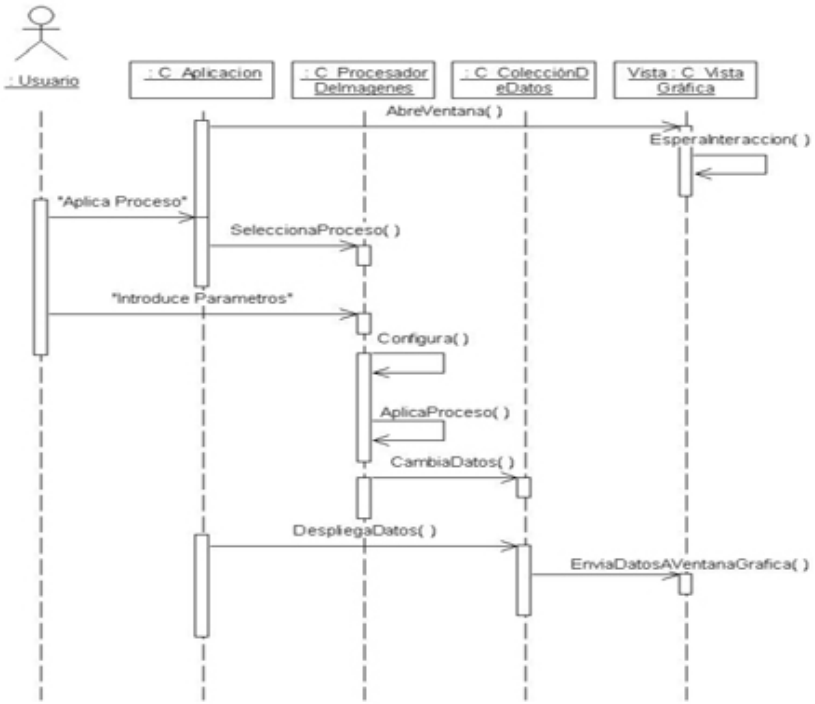


Figura 1.9: Diagrama de secuencia de un visor de un visor de imágenes.

- **Colaboración**: Presenta un diagrama de interacción que resalta la organización estructural de los objetos que envían y reciben mensajes; muestra un conjunto de objetos, enlaces entre esos objetos y mensajes enviados y recibidos por esos objetos (ver fig. 1.10).

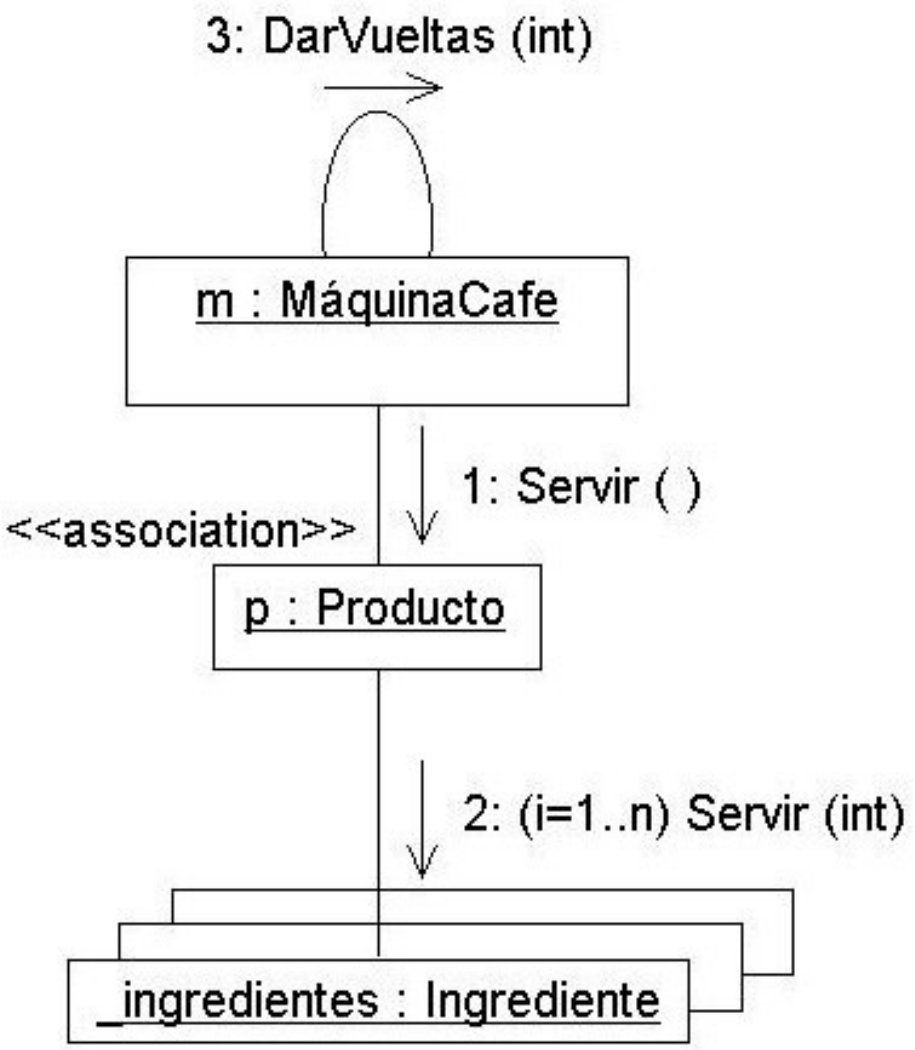


Figura 1.10: Diagrama de colaboración de una máquina de café.

- **Estado**.

  Representa una máquina de estados, constituida por estados, transiciones, eventos y actividades; se utilizan para modelar el comportamiento de una interfaz, una clase o colaboración [BOG99]. Representan el comportamiento dirigido por eventos de un objeto, lo que es útil al modelar sistemas reactivos (ver fig. 1.11).

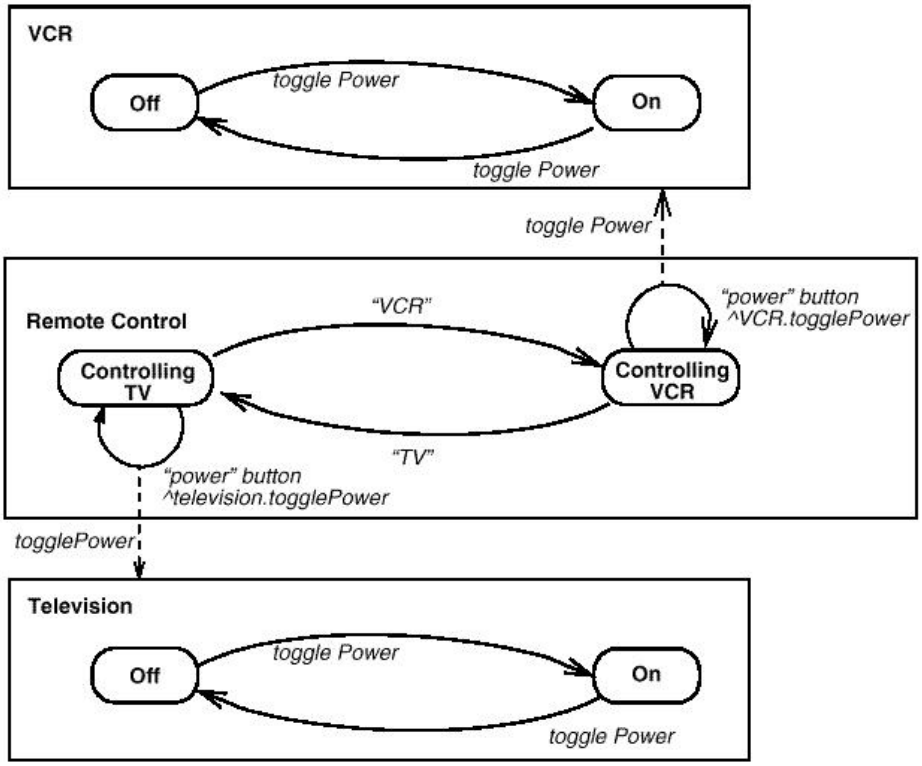


Figura 1.11: Diagrama de estados para el control de TV.

- **Actividades**.

  Es la versión UML de diagramas de flujo. Muestra flujo de actividades en un sistema. Una actividad muestra un conjunto de actividades, el flujo secuencial o ramificado de actividades, y los objetos que actúan y sobre los que actúa [BOG99]. Son importantes

al modelar la función de un sistema, así como para resaltar el flujo de control entre objetos (ver fig. 1.12).

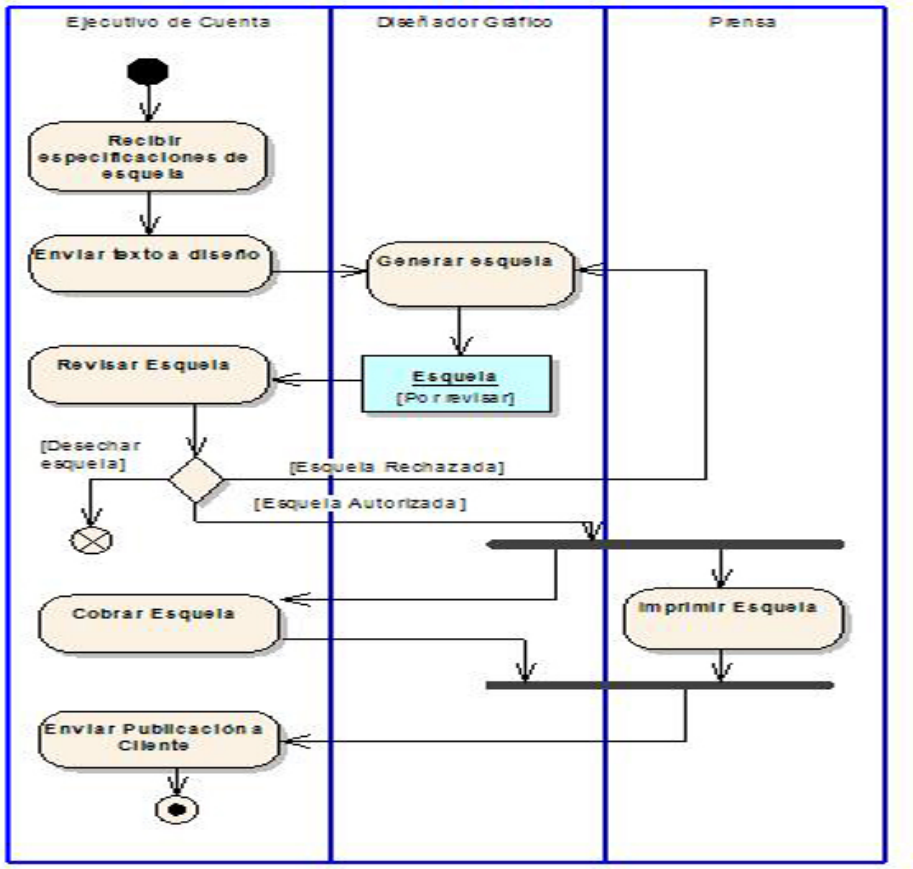


Figura 1.12: Diagrama de actividades de publicación de nota de periódico.

1.4. Rational Unified Process

El modelo de *Desarrollo Iterativo y Creciente (o Incremental)* es un proceso de desarrollo de software, creado en respuesta a las debilidades del modelo tradicional de cascada. Para apoyar el desarrollo de proyectos por medio de este modelo se han creado *frameworks* (entornos de trabajo), de los cuales uno de los más famosos es Rational Unified Process. El *Proceso*

Unificado Racional (RUP) es un proceso de desarrollo de software diseñado y comercializado por Rational Software (ahora parte de IBM) que, junto con UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos [RUP08].

El RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización, que incluye información entrelazada de diversos artefactos y descripciones de las diversas actividades [BOG99]; es decir, es un *framework* que pretende ser personalizado o configurado para organizaciones y proyectos específicos y una guía que define roles, actividades, flujos de trabajo y lineamientos para ejecutar proyectos de software de acuerdo a los principios en los que está basado.

1.4.1. Principales características.

Entre las características más notorias de *RUP*, se encuentran las siguientes, obtenidas de [BOG99]:

1. Es un proceso *iterativo*, el cual propone una comprensión incremental del problema a través de refinamientos sucesivos y un crecimiento incremental de una solución efectiva a través de varios ciclos.

2. Las actividades de RUP destacan en la creación y mantenimiento de *modelos* más que documentos sobre papel.

3. El desarrollo bajo RUP está *centrado en la arquitectura* para guiar el desarrollo del sistema.

4. Las actividades de desarrollo bajo RUP están *dirigidas por los casos de uso* para guiar el flujo de procesos desde la captura de requisitos hasta las pruebas, y para proporcionar caminos que se pueden reproducir durante el desarrollo del sistema.

5. RUP soporta las *técnicas orientadas a objetos.*

6. Es un proceso *configurable*; es decir, adaptable para cubrir las necesidades de proyectos que van desde pequeños equipos hasta grandes empresas de desarrollo.

7. Impulsa un *control de calidad* y una *gestión del riesgo* objetivos y continuos.

1.4.2. Principios.

El RUP está basado en 5 principios clave [RUP08], que son:

- **Adaptar el proceso**.

  El proceso deberá adaptarse a las características propias del proyecto u organización. El tamaño del mismo, así como su tipo o las regulaciones que lo condicionen, influirán en su diseño específico. También se deberá tener en cuenta el alcance del proyecto.

- **Balancear prioridades**.

  Los requerimientos de los diversos inversores pueden ser diferentes, contradictorios o disputarse recursos limitados. Debe encontrarse un balance que satisfaga los deseos de todos.

- **Demostrar valor iterativamente**.

  Los proyectos se entregan, aunque sea de un modo interno, en etapas iteradas. En cada iteración se analiza la opinión de los inversores, la estabilidad y calidad del producto, y se refina la dirección del proyecto así como también los riesgos involucrados

- **Elevar el nivel de abstracción**.

  Este principio dominante motiva el uso de conceptos reutilizables tales como patrón del software, lenguajes 4GL o esquemas (*frameworks*) por nombrar algunos. Esto previene a los ingenieros de software ir directamente de los requisitos a la codificación de software a la medida del cliente. Un nivel alto de abstracción también permite discusiones sobre diversos niveles arquitectónicos. Éstos se pueden acompañar por las representaciones visuales de la arquitectura, por ejemplo con UML.

- **Enfocarse en la calidad**.

  El control de calidad no debe realizarse al final de cada iteración, sino en todos los aspectos de la producción. El aseguramiento de la calidad forma parte del proceso de desarrollo y no de un grupo independiente.

1.4.3. Fases.

El ciclo de vida RUP es una implementación del *Desarrollo en espiral*, creado ensamblando los elementos en secuencias semiordenadas. El ciclo de vida organiza las tareas en fases e iteraciones. Una *fase* es el intervalo de tiempo entre 2 hitos importantes del proceso durante el cual se cumple con un conjunto bien definido de objetivos, se completan artefactos y se toman decisiones sobre si pasar a la siguiente fase [BOG99]. Una *iteración* representa un ciclo de desarrollo completo, que va desde la captura de requisitos en el análisis hasta la implementación y pruebas, que producen como resultado la entrega al cliente o salida al mercado de un proyecto ejecutable [BOG99].

El RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al culminar cada una de ellos (ver fig. 1.13); éstos a la vez se dividen en fases que finalizan con un hito donde se debe tomar una decisión importante:

- **Concepción**.

  Se establece la planificación del proyecto y se delimita su alcance; incluye los criterios de éxito, la evaluación de riesgo, estimaciones de recursos que se necesitarán y un plan de fases que muestre la planificación de los hitos principales. Es frecuente crear un prototipo ejecutable que sirva para probar los conceptos. Al final de esta fase se examinan los objetivos del ciclo de vida del proyecto y se decide si proceder con el desarrollo del sistema.

- **Elaboración**.

  Se analiza el dominio del problema, se establece una base arquitectónica sólida, se desarrolla el plan del proyecto y se eliminan los elementos de más alto riesgo del proyecto. Al final de la fase se examina

el alcance y los objetivos del sistema, la elección de la arquitectura y la resolución de los riesgos más grandes, y se decide si se pasa a la construcción.

- **Construcción**.

  Se desarrolla de forma iterativa e incremental un producto completo que está preparado para la transición hacia la comunidad de usuarios; describe los requisitos restantes y los criterios de aceptación, refinando el diseño y completando la implementación y las pruebas de software. Al final se decide si el software, los lugares donde se instalará y los usuarios están preparados para empezar a funcionar.

- **Transición**.

  El software se despliega en la comunidad de usuarios. Una vez que el sistema ha sido puesto en manos del usuario final, a menudo aparecen cuestiones que requieren un desarrollo adicional para ajustar el sistema, corregir algunos problemas no detectados o finalizar algunas características que habían sido pospuestas. Comienza con una versión beta, que después será reemplazada con el sistema de producción. Al final se decide si se han satisfecho los objetivos del ciclo de vida del proyecto y se determina si se debería empezar otro ciclo de desarrollo.

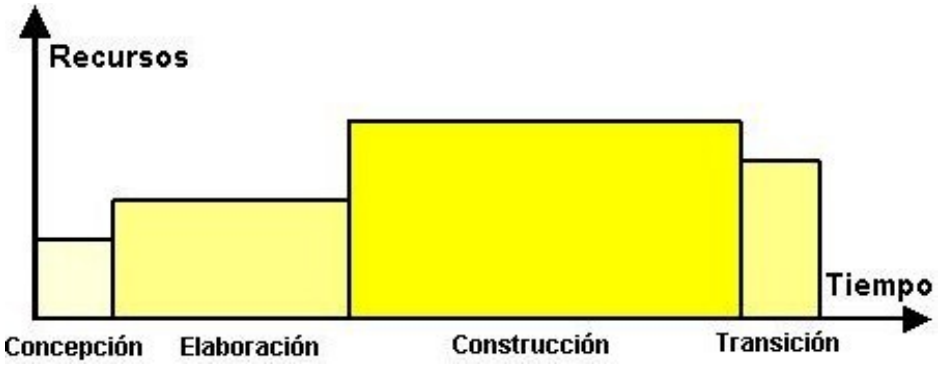


Figura 1.13: Ciclo de vida de *RUP*.

1.4.4. Flujos de trabajo.

RUP consta de nueve flujos de trabajo, que son llevados a cabo a la par con las fases del desarrollo del proyecto [BOG99].

1. **Modelado del negocio**: describe la estructura y dinámica de la organización.
2. **Requisitos**: Describe el método basado en los casos de uso para extraer los requisitos
3. **Análisis y diseño**: describe las diferentes vistas arquitectónicas.
4. **Implementación**: tiene en cuenta el desarrollo de software, prueba de unidades e integración
5. **Pruebas**: describe los casos de pruebas, procedimientos y métricas para evaluación de defectos
6. **Despliegue**: cubre la configuración del sistema entregable
7. **Gestión de configuraciones**: controla los cambios y mantiene la integridad de los artefactos de un proyecto
8. **Gestión del proyecto**: describe estrategias de trabajo en un proceso iterativo
9. **Entorno**: cubre la infraestructura necesaria para desarrollar un sistema

1.4.5. Artefactos.

Una organización de software que trabaja bien produce toda clase de artefactos, además de código ejecutable [BOG99]. Estos artefactos, de corte de gestión, incluyen:

- Requisitos.
- Arquitectura.
- Diseño.
- Código fuente.

- Planificación del proyecto.
- Pruebas.
- Prototipos.
- Versiones.

Algunos se tratan más o menos formalmente que otros. Los artefactos no sólo son los entregables de un proyecto, también son críticos en el control, medición y comnicación que requiere un sistema durante su desarrollo y después de su despliegue.

Otros artefactos, de corte técnico, son:

- Conjunto de requisitos, que describen *qué* debe hacer el sistema.
- Conjunto de diseño, que describe *cómo* se va a construir el sistema.
- Conjunto de implementación, que describe el *ensamblado* de los componentes de software.
- Conjunto de despliegue, que brinda todos los *datos para la configuración* entregable.

1.4.6. Modelos.

Los modelos son el tipo de artefacto más importante de RUP. Un modelo es una simplificación de la realidad, creada para comprender mejor el sistema que se está creando [BOG99]. En RUP, hay nueve modelos que en conjunto cubren todas las decisiones importantes de un sistema con gran cantidad de información.

1. **Modelo de negocio**: establece una abstracción de la organización
2. **Modelo de dominio**: establece el contexto del sistema
3. **Modelo de casos de uso**: establece los requisitos funcionales del sistema
4. **Modelo de análisis (opcional)**: establece un diseño de ideas

5. **Modelo de diseño**: establece el vocabulario del problema y su solución
6. **Modelo del proceso (opcional)**: establece los mecanismos de concurrencia y sincronización del sistema.
7. **Modelo de despliegue**: establece la topología hardware sobre la cual se ejecutará el sistema
8. **Modelo de implementación**: establece las partes que se utilizarán para ensamblar y hacer disponible el sistema físico
9. **Modelo de pruebas**: establece las formas de validar y verificar el sistema.

1.5. Lenguajes orientados a la *web*.

En esta sección se describen los lenguajes más populares para el desarrollo web (PHP, Ajax, HTML y Javascript).

1.5.1. HTML.

Del acrónimo inglés *HyperText Markup Language*, desarrollado por el Centro Europeo de Investigación Nuclear en Ginebra (GEING), es un lenguaje de marcado diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas web. Gracias a Internet y a los navegadores como IE, Opera, Firefox, Netscape o Safari, HTML se ha convertido en uno de los formatos más populares y fáciles de aprender que existen para la elaboración de documentos para web [HTM08] (ver fig. 1.14).

De acuerdo con Leduc y St. Pierre [LED99], HTML no es un lenguaje de programación, aunque sí permite incluirle código en lenguajes de dicha naturaleza bajo ciertos criterios, extendiendo así su capacidad y funcionalidad, aunque eso se logre excediendo los alcances de HTML en sí.

Además, es considerado como un protocolo utilizado para codificar las páginas web en el WWW, y transferir archivos *http* (*HiperText Transfer Protocol*).

1.5.2. Javascript.

Es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web con una sintaxis muy semejante a la del lenguaje Java y el lenguaje C [ORC02], pero Javascript es más sencillo que Java. Fue creado por Netscape con el objeto de integrarse en HTML y facilitar la creación de páginas interactivas sin la necesidad de utilizar *scripts* de GCI o Java; su código se introduce en el documento HTML y no necesita ser compilado: es el propio navegador el que se encarga de traducir dicho código.

Javascript es un lenguaje orientado a objetos, diseñado para el diseño de aplicaciones tipo cliente-servidor a través de Internet. Algo a considerar es que no todos los navegadores soportan Javascript; para solucionar este problema, se engloba el código con la etiqueta de HTML encargada de introducir comentarios [ORC02].

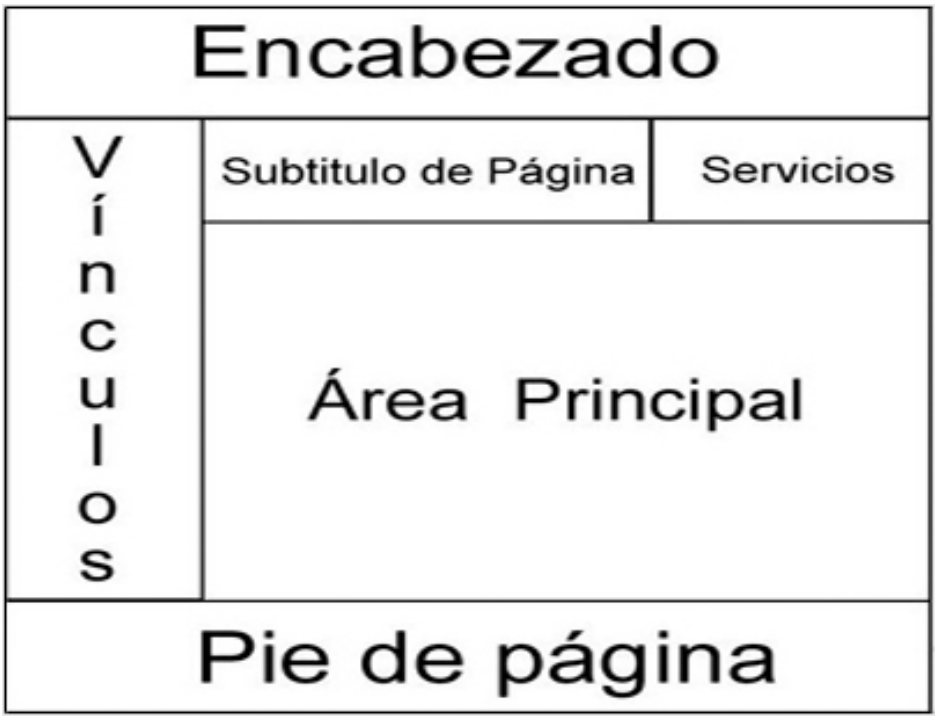


Figura 1.14: Estructura de HTML.

1.5.3. PHP.

De su acrónimo en inglés, *PHP Hypertext Preprocessor*, es un lenguaje *open source* de alto nivel y de *scripting* del lado del servidor, lo cual significa que los *scripts* se ejecutan en el servidor (la PC donde se ubica la página web), en comparación de Javascript, que se ejecuta del lado del cliente [VAJ04]. Como los *scripts* se ejecutan en el servidor, PHP puede crear dinámicamente el código HTML que genera la página web, por lo que permite a los usuarios individuales ver páginas personalizadas. Los visitantes de la página web ven el *output* de los *scripts*, pero no los *scripts* mismos.

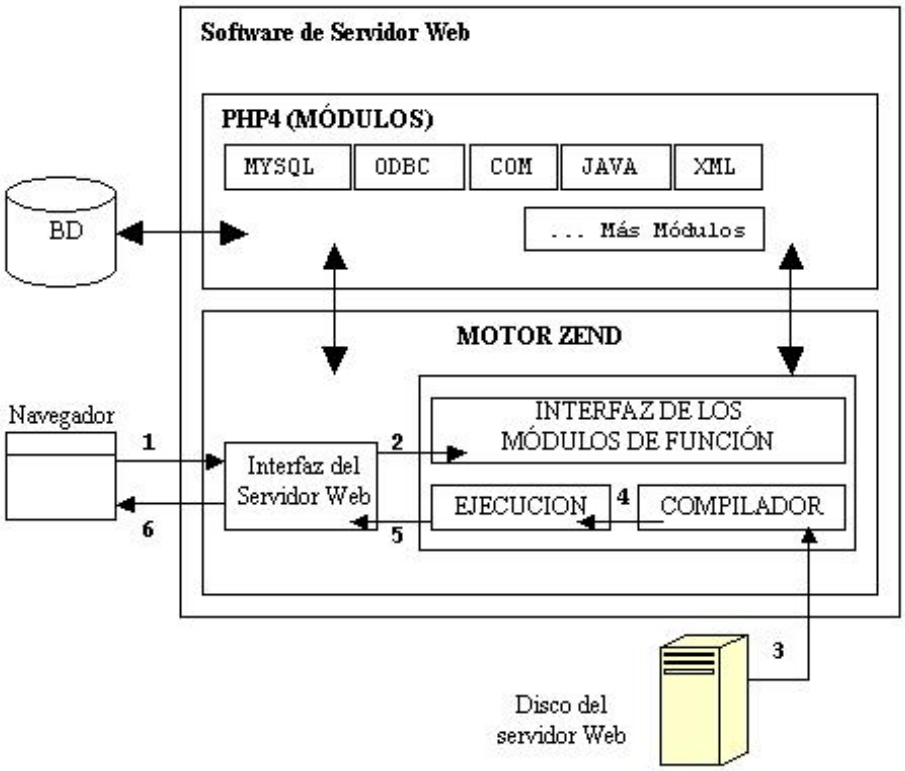


Figura 1.15: Arquitectura de PHP con motor compilador Zend.

Originalmente, se diseñó para usarse en el desarrollo de sitios web bajo el nombre de *Personal Home Page*, desarrollado por Rasmus Lerdorf para ayudar a los usuarios con tareas de páginas web. Creció tan rápidamente que pasó a convertirse en un lenguaje en forma con las características que posee

hoy en día; la más importante: *procesar páginas web antes de mostrarlas* [VAJ04]. Actualmente, también se puede utilizar para la creación de otros tipos de programas, incluyendo aplicaciones con interfaz gráfica usando las bibliotecas *QL* o *GTK++*, todo mediante su intérprete (*Zend*). En la fig. 1.15 se puede apreciar la arquitectura de PHP.

Una cuestión a considerar es que PHP no está integrado con todos los servidores web, pero funciona con los más populares: se encuentra como proyecto de Apache Software Foundation y, por lo tanto, funciona mejor con Apache; también funciona con Microsoft IIS/PWS, iPlanet (anteriormente Netscape Enterprise Center) y otros.

1.5.3.1. Características.

PHP tiene muchas características diseñadas específicamente para su uso en sitios web [VAJ04], incluyendo las siguientes [VAJ04]:

- **Interacción con formularios HTML**.

  PHP puede mostrar un formulario HTML y procesar la información que el usuario digita.

- **Se comunica con las bases de datos**.

  PHP puede interactuar con bases de datos para almacenar la información que se le enseña al usuario.

- **Genera páginas web seguras**.

  PHP permite al desarrollador crear páginas web seguras que obligan a los usuarios a digitar un nombre de usuario y una contraseña válida antes de ver el contenido de una página web.

1.5.4. Xajax.

Ajax es una tecnología que permite una nueva gama de aplicaciones interactivas en la web, mucho más ricas y rápidas, utilizando a su vez otra combinación de tecnologías, tales como XML y Javascript, para realizar peticiones de contenido o computación de servidor sin tener que recargar la

página en la que está el usuario [OSBA08]. Para Ajax, al igual que PHP, se han desarrollado varios *frameworks* para trabajar con él; para PHP, existe una *biblioteca* especializada en esta tecnología, denominada *Xajax*.

Xajax es una clase *open source* compatible con navegadores como Firefox, u otros navegadores basados en Mozilla, Internet Explorer u Opera, realizada con PHP que permite generar aplicaciones interactivas Web con tecnología Ajax designando qué funciones de código PHP se convierten en funciones Ajax [XAJX08]. Una de las ventajas más significativas de esta clase es que, para instalarla, simplemente se descomprime la carpeta de archivos en cualquier servidor *Apache* o *IIS* que tenga compatibilidad con PHP 4.3.x o superiores. Actualmente, la versión *estable* de este framework es Xajax 0.2.5 [XAJX08].

1.5.4.1. Funciones de Xajax.

Las funciones Xajax [XAJX07] tienen dos objetivos: interactuar con el servidor (para crear un usuario nuevo, para examinar los archivos que hay en un directorio, etc.) y devolver una respuesta XML que será enviada al navegador y interpretada por Javascript. Esta respuesta será hecha por el objeto **XajaxResponse**, que se configurará con métodos y creará el XML que será enviado posteriormente al navegador interpretado por Javascript. El constructor de esdta función consta de dos argumentos: la codificación (por defecto *UTF-8*) y un valor *booleano* que, si es verdadero, muestra los caracteres especiales en el navegador, y si es falso, por defecto, los oculta. Para mayor referencia, consultar el Apéndice B.

Capítulo

2

Trabajos relacionados.

Como parte del desarrollo de un proyecto, se debe hacer hincapié en algunos trabajos que guarden cierta similitud con el desarrollo del presente trabajo, para así llevar a cabo un análisis de las tendencias en materia de sistemas web. El presente capítulo aborda el análisis de artículos que reflejan dichas tendecias.

2.1. SCIA.

Martínez Luma G. L. y Guzmán Arenas A., del Laboratorio de Sistemas de Información y Bases de Datos del Centro de Investigación en Computación del Instituto Politécnico Nacional (IPN); Vázquez Gallo M., Camacho González G. M. y Hernández García O., de la Escuela Superior de Cómputo del IPN, y Cedeño Nicolás J., de la Escuela Nacional de Antropología e Historia, realizan el proyecto conocido como Sistema de Información para Consultas a Investigaciones Arqueológicas (SCIA), cuyo fin es el proporcionar información de referencia y material de trabajo a alumnos y profesores para la capacitación en técnicas de análisis e interpretación, así como de un gran número de materias optativas relacionadas (ver fig. 2.1).

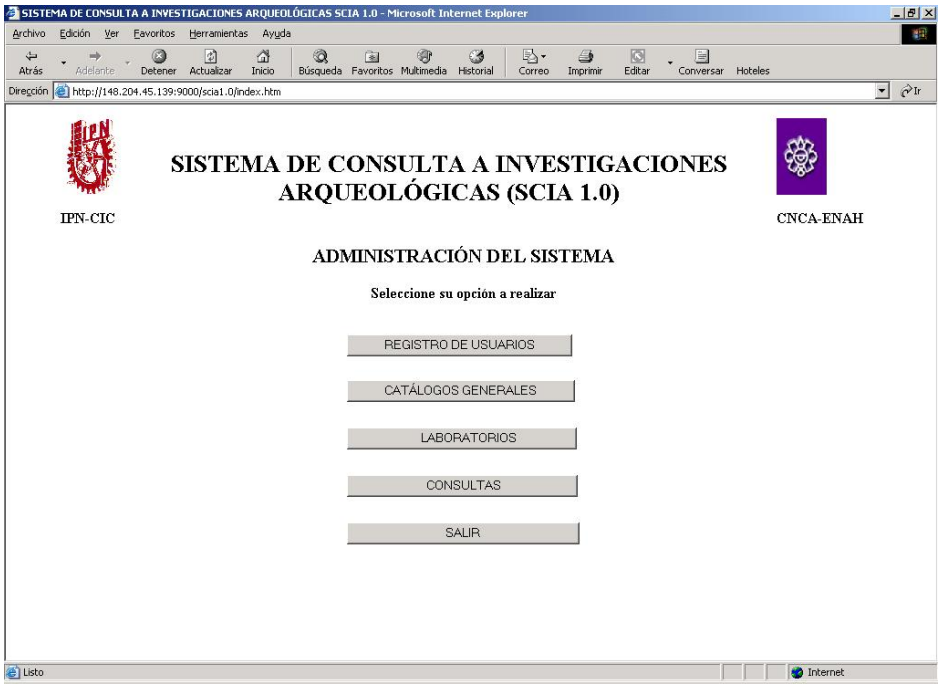


Figura 2.1: Página principal del SCIA.

Para la construcción de dicho sistema, se emplean las tecnologías de HTML, programas en JSP, el driver JDBC, el sistema administrador de bases de datos MySQL en su versión software libre y un servidor web Tomcat, montado bajo redes TCP/IP (ver fig. 2.2). Para cumplir con los objetivos planteados, el

diseño inicial del sistema contempla cuatro módulos:

- Administración de Usuarios.
- Catálogos Generales de Apoyo.
- Laboratorios.
- Consultas.

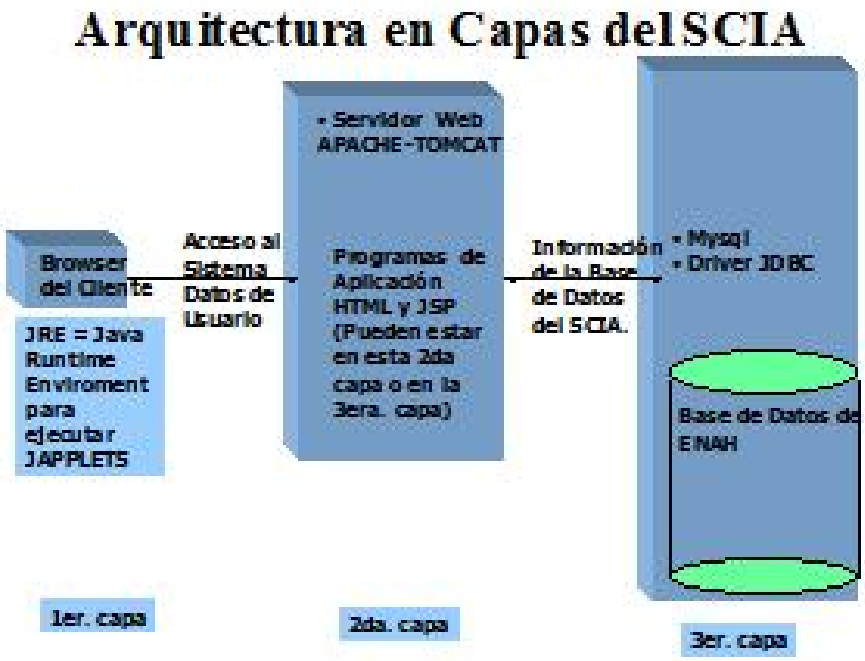


Figura 2.2: Arquitectura del SCIA.

Es claro que este sistema una vez en operación ayudará a facilitar la administración de la información de los laboratorios al poder acceder a datos puntuales que se deseen, así como al compartir los datos que se tienen en las colecciones. El desarrollo en la presente fase no es de un alto grado de complejidad, pero sí de bastante labor de construcción de programas y de organizar la información a manejar. Se cumple que la complejidad, como en

todo sistema especializado, es el tipo de conocimiento tan fino que hay que poseer del área por parte de los analistas y diseñadores del sistema, pero que se resuelve con las continuas consultas a los especialistas del área.

También donde está el grado de complejidad es en el lograr ingresar en la forma de pensamiento del arqueólogo o antropólogo, y así construir sistemas que ayuden realmente a la labor de análisis, investigación, estudio e interpretación de la utilidad y uso de los objetos arqueológicos, con lo cual se logra el explicar las culturas ya desaparecidas o casi por desaparecer en México. Tarea muy reconocida en nuestro país, pero poco apoyada para obtener herramientas tecnológicamente recientes y de gran utilidad.

Una vez diseñada y establecida la Base de Datos y el software del Sistema de Información de la Escuela Nacional de Antropología e Historia, se puede proceder a construir programas prototipos de herramientas para realizar análisis de objetos arqueológicos que podrán ayudar a generar nuevas metodologías de interpretación de los objetos que se estudian en los laboratorios, y así asegurar o corroborar teorías en base a las fechas de los objetos, cuánto tiempo permaneció la cultura en cierto lugar, rutas de desplazamiento, clasificación y agrupación de objetos, etc. Las herramientas deberán utilizar técnicas de Inteligencia Artificial, Minería de Datos y Reconocimiento de Patrones [MALG06].

2.2. Sistema de Información para Área Académica (UAEH).

Magín Villegas Lara, alumno egresado de la Lic. en Sistemas Computacionales en la Universidad Autónoma del Estado de Hidalgo, expone en su tesis el desarrollo de un sistema orientado a la web, diseñado mediante la metodología IDEF0 e IDEF1x, y construido con código SQL - PHP - HTML, que resuelva el problema del manejo inadecuado de la información generada por los Cuerpos Académicos (CA) relacionada con las actividades que llevan a cabo. La carencia de un sistema de información provoca problemas en la obtención y consulta de la misma, tanto para los CA como para el Área Académica correspondiente [VIM06]; en este caso, el Área Académica de Sistemas Computacionales, del Instituto de Ciencias Básicas e Ingeniería

(fig. 2.3).

La idea fundamental del diseño del mismo radica en que los profesores realicen actividades académicas, como investigación, docencia, difusión y vinculación, y toda la información que estas actividades generan se almacenen y clasifiquen adecuadamente, de modo que se pueda actualizar conforme se generen nuevos productos [VIM06]; además, es deseable para toda Institución de Educación Superior contar con un sistema de información que permita el uso eficiente de los datos referentes a los CA, y que cuente con funciones que faciliten el manejo de los datos. El sistema permite la interacción con los usuarios en diferentes niveles:

Figura 2.3: Pantalla principal del sistema de información vía web.

- El Profesor de Tiempo Completo (PTC) puede actualizar su información a través del sistema.
- El coordinador de los programas académicos accede y valida la información que vacían los PTC.
- La coordinación de investigación está habilitada para dar seguimiento a los proyectos de investigación llevados a cabo dentro del Centro de In-

vestigación de Tecnologías de Información y Sistemas de la Universidad Autónoma del Estado de Hidalgo (CITIS).

El sistema que presenta es un prototipo, implantado para validar modelos desarrollados en dicho trabajo, orientados a IDEF1x. Tiene la posibilidad de adicionar nuevas funcionalidades, de acuerdo a las necesidades de información que planteen los usuarios del mismo. Sin embargo, el método de integración que plantea, así como los modelos desarrollados, constituyen un avance significativo en la construcción continua del sistema, pues contempla el tipo de información que se genera y consulta a lo largo de la dinámica real de los CA [VIM06].

2.3. Sistema para Deportes LSI 03.

César López Rodríguez, José Luis Martínez Herrero, Germán Mira Rico, Miguel Antonio Mascilla Guzmán, José Antonio Mocholí Agües y Eduardo Bueno Medina, de la Universidad Politécnica de Valencia, en Laboratorio de Sistemas de Información, exponen el desarrollo de un sistema de información orientado a la web para una tienda de artículos deportivos denominada *Deportes LSI 03*, llevando a la práctica un acercamiento de la metodología RUP [LORC03] (ver fig. 2.4).

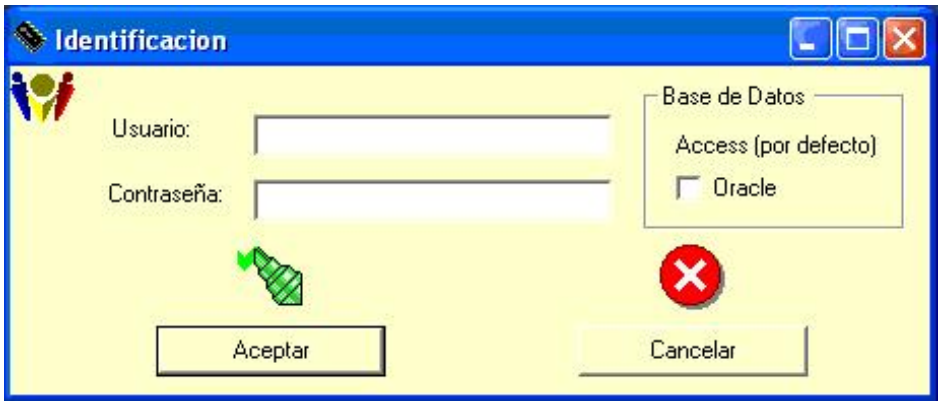


Figura 2.4: Interfaz inicial de *LSI 03*.

El sistema tiene como propósito proporcionar una propuesta para el desarrollo de todos los subsistemas implicados en la gestión de artículos deportivos y bases de datos departamentales. Estos subsistemas se pueden diferenciar en siete grandes bloques:

1. **Gestión de Almacenes** (ver fig. 2.5), incluyendo:
   - Gestión de nuevos pedidos.
   - Reserva de *stock* para la preparación de pedidos.
   - Gestión de incidencias de *stock*.
   - Gestión de pedidos para envío.
   - Gestión de consultas de estado de pedidos
   - Cancelación de pedidos solicitado por el cliente.

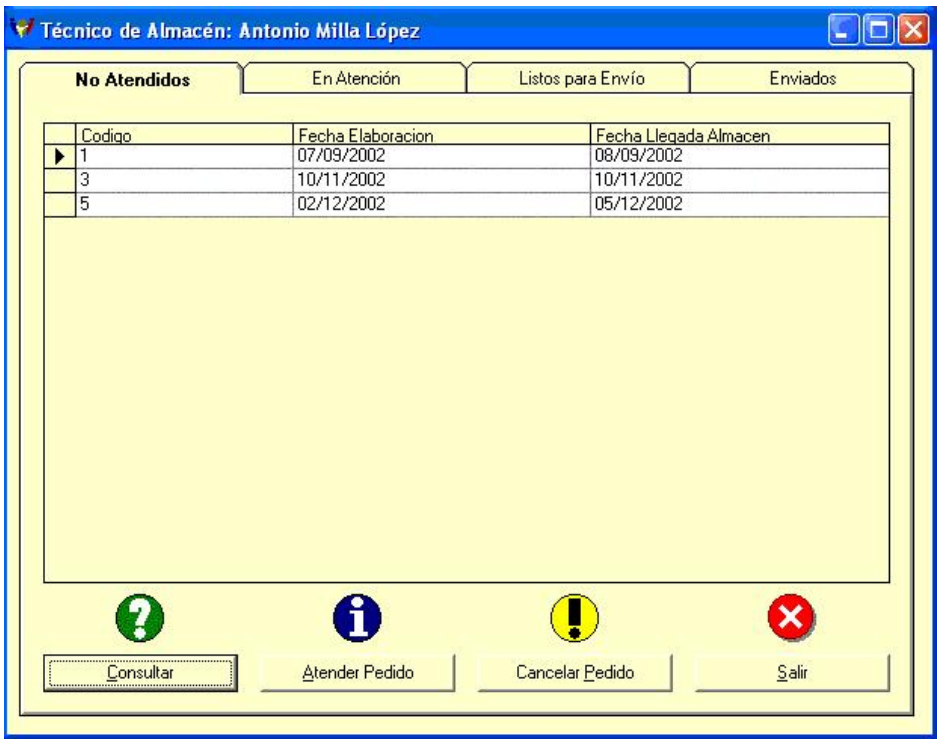


Figura 2.5: Interfaz de *Almacén*.

2. **Gestión de Ventas**, incluyendo:
   - Procedimiento de venta de productos vía operadoras de teléfono.
   - Procedimiento de venta mediante la atención de comerciales a domicilio del cliente.
   - Procedimiento de venta mediante el sistema *online*, vía web.
3. **Gestión de Envíos** (ver fig. 2.6), incluyendo:
   - Gestión de pedidos para envío.
   - Gestión de recibos.

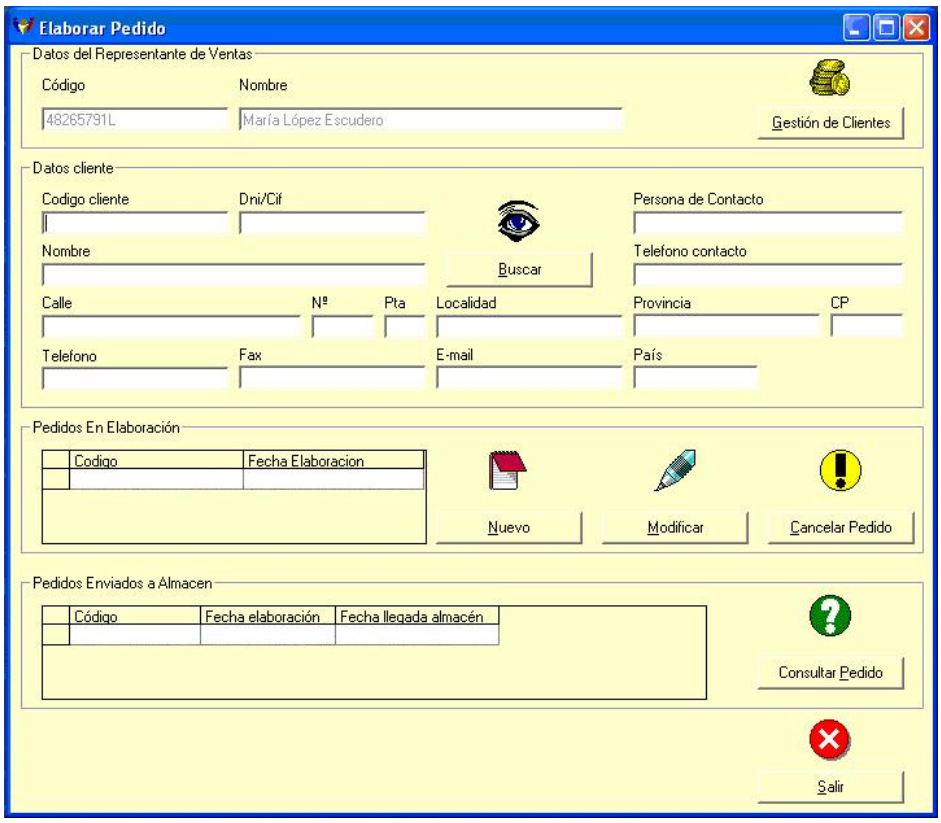


Figura 2.6: Interfaz de *Pedidos*.

4. **Departamento de Recursos Humanos**.

5. **Departamento de Marketing**.
6. **Departamento de Logística**.
7. **Contabilidad y Facturación**.

Todos los artefactos RUP que son generados son objeto de modificaciones a lo largo del proceso de desarrollo, con lo cual, sólo al término del proceso se puede obtener una versión definitiva y completa de cada uno de ellos. Sin embargo, el resultado de cada iteración y los hitos del proyecto están enfocados a conseguir un cierto grado de completitud y estabilidad de los artefactos; entre los artefactos generados y utilizados para el desarrollo de este proyecto se encuentran: Plan de Desarrollo del Software, Modelo de Casos de Uso del Negocio, Modelo de Objetos del Negocio, Glosario, Modelo de Análisis y Diseño, Casos de Prueba, entre otros.

Para facilitar el desarrollo de este proyecto se utilizaron como soporte plantillas basadas en la metodología RUP y también los documentos de información adicional de la empresa de deportes. El proyecto se basa en una serie de etapas, que contemplan diferentes aspectos.

- **Gestión del Proyecto.**

  Presenta las planificaciones temporales de desarrollo del proyecto en su fase de inicio y de elaboración, así como el diario de ejecución del proyecto, junto con el diario de construcción de la aplicación y cumplimiento de los plazos estimados.

- **Modelado del Negocio.**

  Engloba los artefactos utilizados de la metodología RUP para definir un modelo del negocio, modelos de objetos del negocio y el modelo del dominio.

- **Requisitos.**

  Consta de plan de desarrollo software, el documento visión, el documento glosario y las especificaciones, tanto de los casos de uso como de los casos de pruebas relacionados con éstos. También se recoge la gestión del proyecto con la herramienta *Rational Requisite Pro*,

con la que además de llevar el control de toda la documentación, se puede seguir la trazabilidad entre requerimientos de todo el proyecto. En este apartado se muestran las matrices de atributos de todos los requerimientos así como la navegabilidad entre ellos. Por añadidura también se muestran los casos de uso de cada subsistema generados con la herramienta *Rational Rose*, y desde los cuales también se puede consultar la especificación del caso de uso.

- **Análisis/Diseño.**

  Muestra tanto el modelo de análisis/diseño (diagrama de clases) como el modelo de datos (modelo Entidad-Relación), desde los cuales se puede consultar la especificación de los métodos de clase más relevantes o las especificaciones de atributos.

- **Implementación.**

  Presenta los prototipos de interfaces de usuario de la aplicación, tanto para el sistema de gestión de ventas como para el sistema de gestión de almacén, además de los diagramas de componentes y diagrama de despliegue que modela las aplicaciones incorporadas en el proyecto hasta la segunda iteración de la fase de construcción (según la definición de fases e iteraciones de la metodología RUP) y desde los cuales, a través de los componentes se puede consultar el código fuente de cada uno.

- **Pruebas.**

  Engloba la documentación de especificación de casos de pruebas funcionales. Presenta únicamente los casos de pruebas generados para los casos de uso incorporados hasta la segunda iteración de la fase de construcción.

La aplicación se desarrolló bajo el lenguaje de programación Visual Basic 6.0, teniendo que soportar tanto acceso a una base de datos Oracle como a una base de datos Access, dependiendo de la selección del usuario en el arranque del programa. Cabe citar que el equipo de desarrollo estaba limitado a unos conocimientos medios del lenguaje de programación, por lo que las soluciones adoptadas no son completamente eficientes [LORC03].

Capítulo

3

Concepción y Elaboración.

Las fases iniciales de RUP corresponden a las tareas de modelar el negocio y definir los casos de uso de mayor relevancia y que serán automatizados, para así posteriormente darles un enfoque sistémico desde una perspectiva estática y dinámica. En el presente capítulo se da la descripción de la metodología a seguir durante la realización del proyecto, así como la descripción del negocio y los modelados mencionados anteriormente, la creación del diagrama de clases y del modelo relacional, detallando los métodos que emplea cada clase para su mayor entendimiento, en la parte estática del sistema, y los diagramas de actividades, secuencia y estado para proveer un acercamiento a los comportamientos del mismo.

3.1. Vista general del proyecto.

El producto a desarrollar es un sistema global vía web para la empresa *Svelta - Clínica de Reducción de Peso*, con la intención de agilizar su funcionamiento. Los usuarios del mismo entrarán al sistema identificándose sobre un ordenador con conexión a Internet y tras este paso se dirijirán a la dirección web de la página inicial del negocio, en la cual, en un apartado especial, se accederá al sistema. Este software cuenta con una interfaz de ventana de Internet; por lo tanto, los usuarios estarán familiarizados con el entorno. Las áreas (o subsistemas) a tratar por el sistema son: *clientes, personal* y *tratamientos*. En la tabla 3.1 se muestra un listado con los beneficios que obtendrá el cliente a partir del producto.

Beneficio del cliente	**Características que lo apoyan**
Mayor agilidad en el registro de clientes y paquetes adquiridos, dando la posibilidad de hacerlo vía servicios web.	Aplicación web desde la cual poder realizar el alta y actualización de clientes, y la incorporación de éste a un tratamiento ofrecido por la organización.
Gestión más controlada de la calendarización de sesiones dando la posibilidad de hacerlo vía servicios web.	Sistema de optimización de calendarización de sesiones.
Mayor facilidad para el control de usuarios que se manejan dentro del negocio, así como los expedientes derivados de los tratamientos de los clientes.	Base de datos con acceso remoto con la información de todo el personal y de los clientes.
Posibilidad de cancelación eficiente de sesiones dando la posibilidad de hacerlo vía servicios web.	Aplicación web con privilegio de cancelación de sesiones.
Seguridad e integridad de la información que se manipula.	Sistema con *scripts* de validaciones para campos de texto y seguridad en caso de acceso ilegal al mismo.

Cuadro 3.1: Beneficios del sistema a desarrollar.

3.2. Entregables del proyecto.

A continuación se indican y describen cada uno de los artefactos que serán generados y utilizados por el proyecto y que constituyen los entregables (ver fig. 3.1).

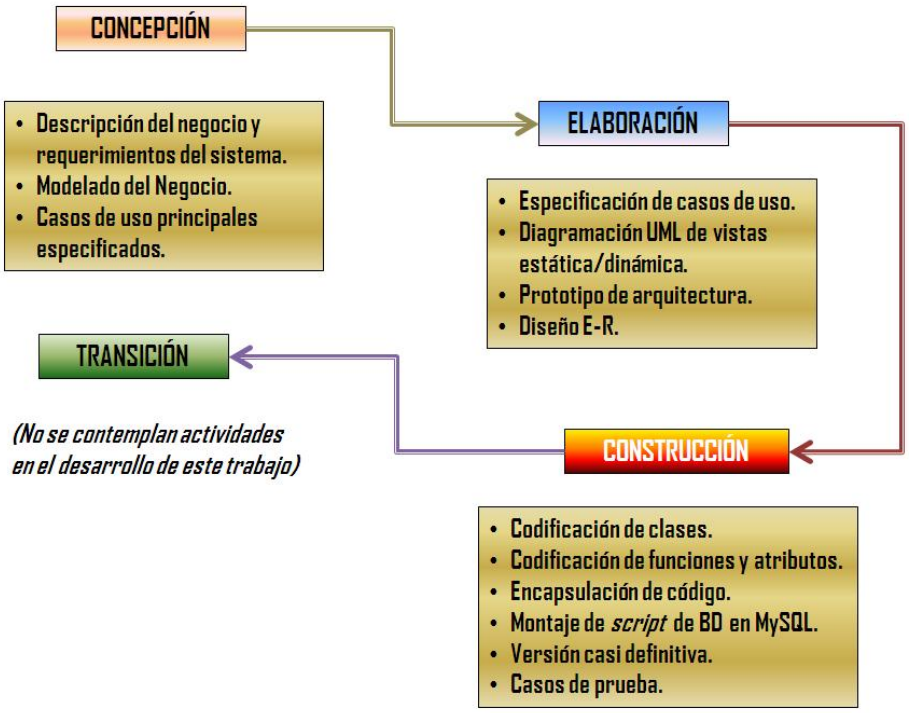


Figura 3.1: Estructura de la metodología del proyecto.

Es preciso destacar que de acuerdo a la filosofía de RUP (y de todo proceso iterativo e incremental), todos los artefactos son objeto de modificaciones a lo largo del proceso de desarrollo, con lo cual, sólo al término del proceso se podría tener una versión definitiva y completa de cada uno de ellos.

Sin embargo, el resultado de cada iteración y los hitos del proyecto están enfocados a conseguir un cierto grado de completitud y estabilidad de los artefactos. La tabla 3.2 detalla los puntos a contemplar en el seguimiento de la metdología escogida para el presente trabajo.

Descripción	**Hito**
Fase de Concepción	En esta fase desarrollará los requisitos del producto desde la perspectiva del usuario. Los principales casos de uso serán identificados y se hará un refinamiento de los mismos. La aceptación del cliente/usuario de dichos casos de uso marcan el final de esta fase.
Fase de Elaboración	En esta fase se analizan los requisitos y se desarrolla un prototipo de arquitectura (incluyendo las partes más relevantes y/o críticas del sistema). Al final de esta fase, todos los casos de uso correspondientes a requisitos que serán implementados en la primera versión de la fase *Construcción* deben estar analizados y diseñados (en las perspectivas estáticas y dinámicas del sistema). Se procede a la identificación y especificación de los principales casos de uso, y se dará una revisión general del estado de los diagramas generados hasta este punto y ajustarlos de ser necesarios para asegurar el cumplimiento de los objetivos planteados al inicio de este trabajo. La revisión y aceptación del prototipo de la arquitectura del sistema, así como el inicio de su construcción, marca el final de esta fase.
Fase de Construcción	En esta fase se termina de analizar y refinar todos los diagramas, y se procede a codificar el prototipo de arquitectura. Dentro de las iteraciones que se generan en esta fase, se va produciendo una versión a la cual se le aplican las pruebas y se valida con el cliente/usuario. El hito que marca el fin de esta fase es la entrega de la versión con la capacidad operacional parcial del producto que se haya considerado como crítica, instalada en un servidor de tipo *local*, lista para ser entregada a los usuarios para pruebas preliminares.
Fase de Transición	Esta fase no se contempla dentro del desarrollo del proyecto, debido a que la versión final no se ha desplegado en la comunidad de los usuarios del mismo, y por ende, no se dan muestras de realizar un desarrollo adicional para ajustar el sistema, corregir algunos problemas no detectados o finalizar algunas características que habían sido pospuestas por el cliente, lo cual se contempla en trabajo a futuro (ver *Trabajo a Futuro*).

Cuadro 3.2: Hitos de las Fases de la Metodología RUP.

3.3. Descripción de la dinámica del negocio.

En el presente negocio, por poseer una infraestructura no compleja debido a que no se ha dado hasta la fecha la alta de alguna franquicia , no esta dividida en departamentos o secciones; por lo tanto, no se procede a desarrollar un diagrama de subsistemas por departamentos. La dinámica del negocio queda plasmada en la fig. 3.2.

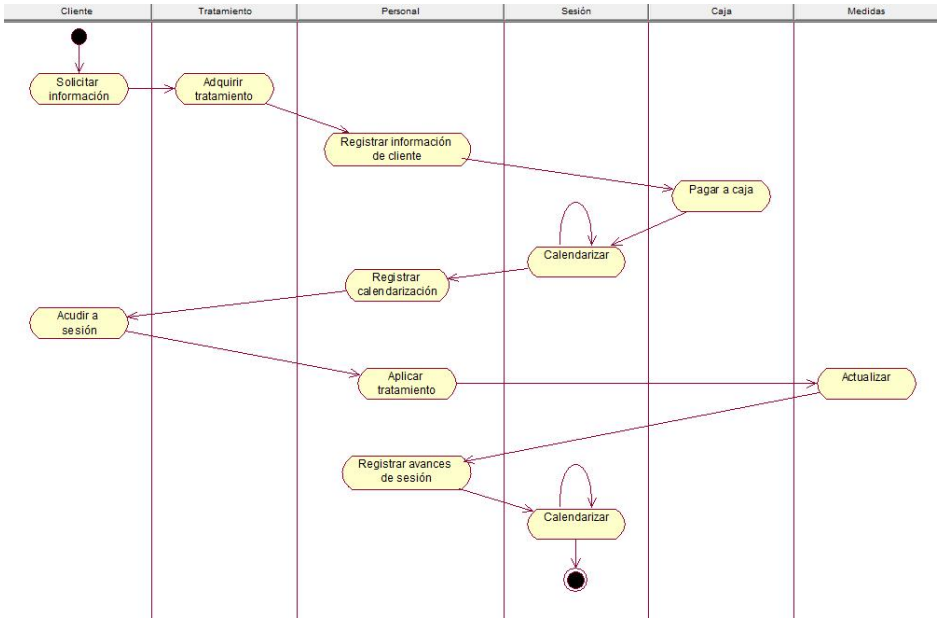


Figura 3.2: Dinámica general de *Svelta*.

3.4. Modelado del Negocio.

El modelado de negocio se basa en dos diagramas principales (*modelo de casos de uso del negocio* y *modelo del dominio*) y en los *modelos de objetos del dominio*.

La organización interactúa con dos elementos: el *cliente* (persona o entidad que solicita alguno de los tratamientos que el negocio ofrece), y el *personal* que labora dentro del negocio, el cual es el elemento que interactúa

más intensamente con el sistema. Como este negocio no vende un producto, sino ofrece un servicio, no contempla proveedores.

Después de revisar el anterior punto, se pueden discernir una serie de actores y actividades base que son importantes para el desarrollo del sistema. Para ilustrar esto, se realiza el diagrama de casos de uso general del negocio, como se muestra en la figura 3.3. Así mismo, analizando la descripción de la dinámica del negocio, se puede derivar el modelo de dominio, el cual esta representado en la siguiente figura 3.4, y del que se pueden abstraer una serie de elementos que se verán complementados con los modelos de objetos del negocio.

3.4.1. Modelo de objetos del dominio.

Los modelos de objetos del dominio están asociados a cada uno de los casos de uso del negocio. En este apartado, se debe considerar una priorización de los casos de uso obtenidos; en ese sentido, el listado quedaría de la siguiente forma: *Adquirir tratamiento*[1] (fig. 3.5); *Acudir a sesiones* (fig. 3.6); *Ingresar cliente* (fig. 3.7), y *Registrar avance de tratamiento* (fig. 3.8).

3.5. Especificación de Casos de Uso.

En el presente apartado, se presentan las especificaciones de los casos de uso seleccionados en el capítulo 3. Cabe resaltar que se incluye el caso de uso *Validar usuario*, por ser una de las operaciones claves dentro del sistema, y no contemplada dentro de los casos de uso iniciales, debido a la inexistencia de algún sistema.

[1] Los casos de uso *Calendarizar sesiones, Consultar expediente* y *Actualizar expediente* involucran una similitud con este modelo de objetos, por lo que no es necesario su presentación.

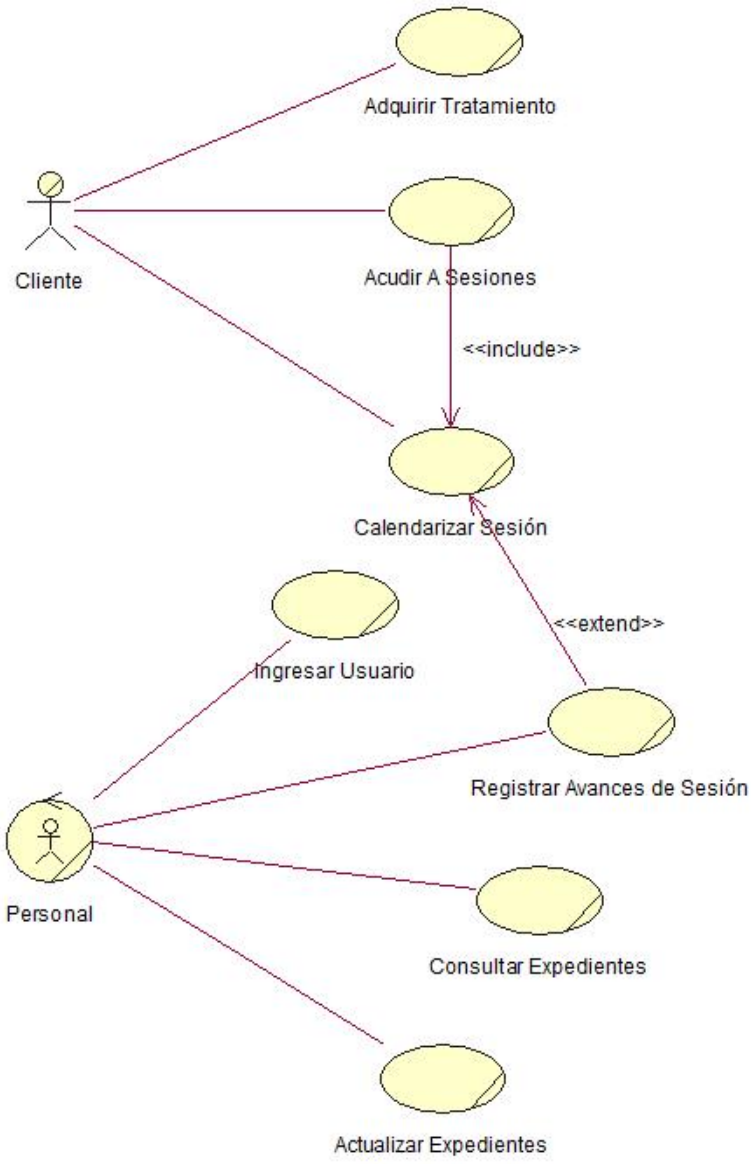


Figura 3.3: Caso de uso general de *Svelta*.

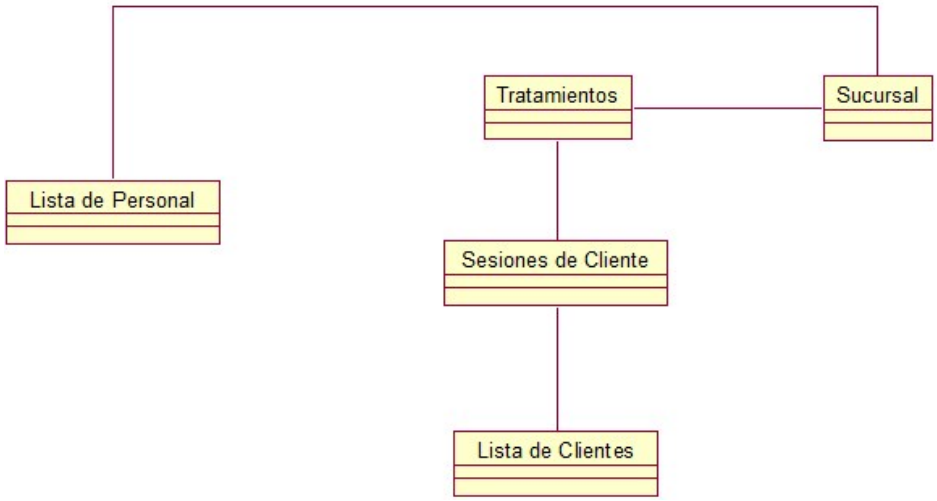


Figura 3.4: Modelo de dominio de *Svelta*.

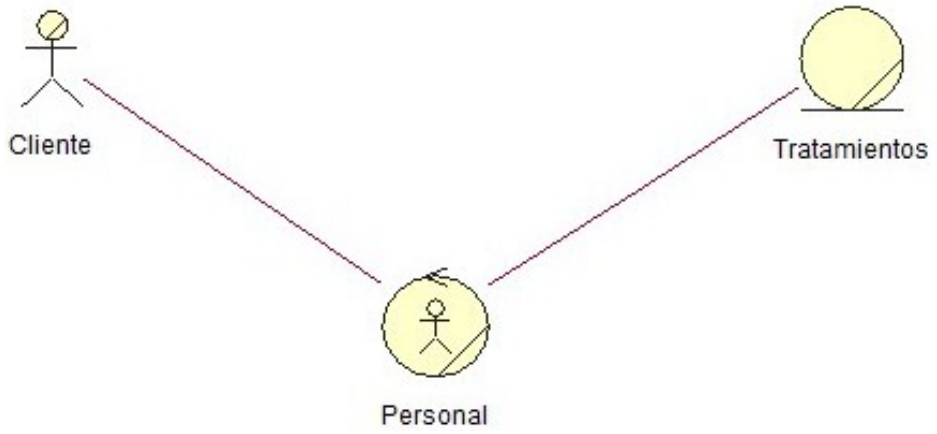


Figura 3.5: Modelado de objetos de negocio de *Adquirir tratamiento*.

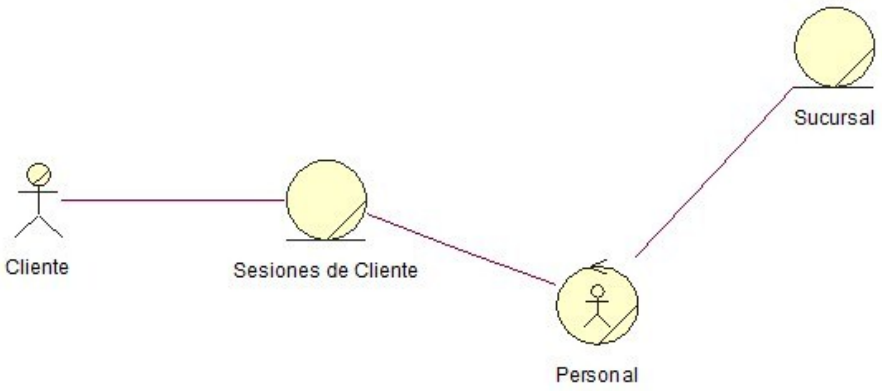


Figura 3.6: Modelado de objetos de negocio de *Acudir a sesiones*.

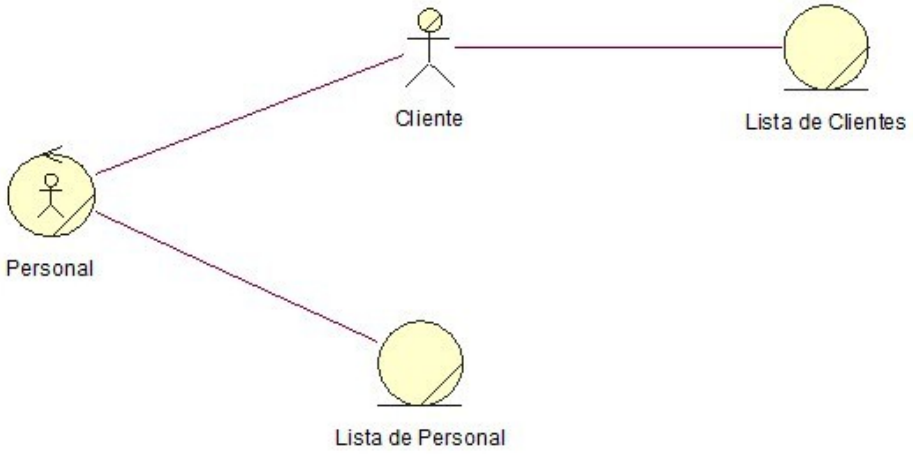


Figura 3.7: Modelado de objetos de negocio de *Ingresar cliente*.

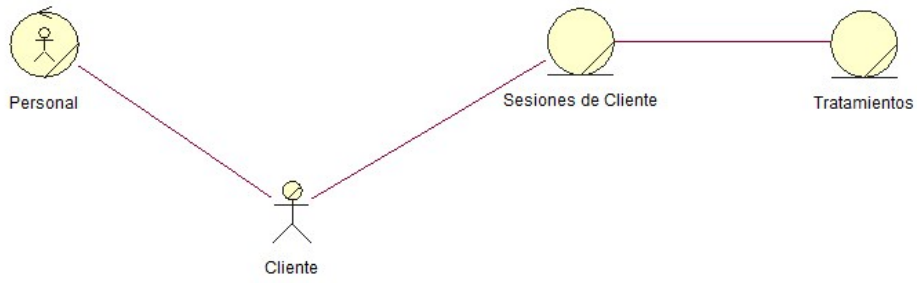


Figura 3.8: Modelado de objetos de negocio de *Registrar avances de sesión.*

3.6. Caso de uso *Validar usuario.*

3.6.1. Descripción.

El personal entra al inicio de sesión para validar su número de cuenta y contraseña.

3.6.2. Flujo de eventos.

Flujo básico.

1. El personal entra a la página principal de *Svelta.*
2. Posteriormente, el personal elige la opción *Entrar.*
3. El personal ingresa su número de usuario y su contraseña.
   a) El personal accede a la página principal del sistema y tendrá la libertad de acceder a las diversas opciones; abandona el sistema dando clic en *Salir.*
   b) El personal regresa a la página principal de *Svelta.*

Flujos alternativos.

- **En paso 3**: De no ingresarse un número de usuario y/o una contraseña válida, o de dejar alguno de estos dos campos vacíos, regresa a la pantalla principal de *Svelta*, desplegando un mensaje de error.

3.6.3. Precondiciones.

El personal está dado de alta en el sistema.

3.6.4. Poscondiciones.

Habiendo ingresado los campos requeridos del inicio de sesión, el personal ingresa al sistema para acceder a las opciones desplegadas en pantalla, dependiendo de su nivel de acceso a dichas características.

3.7. Caso de uso *Adquirir tratamiento.*

3.7.1. Descripción.

El cliente adquiere el servicio de reducción de peso, el cual consta de la adquisición de un paquete formado por tratamientos y que conforma en su totalidad el tratamiento de reducción de peso.

3.7.2. Flujo de eventos.

Flujo básico.

1. El cliente elige uno de los paquetes de reducción de peso ofrecidos por el negocio, y paga su monto.
2. El personal da de alta en el sistema el paquete adquirido por el cliente.
3. El cliente estará en la posibilidad de iniciar la calendarización de sesiones.

Flujos alternativos.

- **En paso 3**: consultar la especificación **Calendarizar sesiones**.

3.7.3. Precondiciones.

1. El personal y el cliente están dados de alta en el sistema.
2. El personal debe haber validado su número de usuario y su contraseña.
3. El paquete debe existir en los registros del sistema.

3.7.4. Poscondiciones.

Una vez que se adquiere el tratamiento, quedará registrado dentro del expediente del cliente en cuestión, y de haber calendarizado una o más sesiones, se registrarán dichas fechas para que en dichos días se desplieguen los datos correspondientes en la pantalla principal del sistema.

3.8. Caso de uso *Calendarizar sesiones.*

3.8.1. Descripción.

El cliente elige día, hora y cabina correspondiente a la sesión en turno, dando la posibilidad de calendarizar más de una sesión a la vez.

3.8.2. Flujo de eventos.

Flujo básico.

1. El cliente solicita reservar una sesión en un determinado día, a una determinada hora y con la opción de elegir en qué cabina desea recibir su sesión.

2. El personal verifica en el sistema la disponibilidad de hora y cabina para el día solicitado por el cliente.

    a) El personal registra los datos correspondientes en el sistema para la reservación de sesión.

    b) El sistema devuelve al personal a la pantalla del subsistema *Clientes.*

Flujos alternativos.

- **En paso 2**: De no existir disponibilidad de fecha, se da la opción de elegir otro día de calendarización; si no hay disponibilidad de horario, se da la posibilidad de elegir otra hora, y si no hay disponibilidad de cabina, se da la posibilidad de elegir otra cabina.

3.8.3. Precondiciones.

1. El personal y el cliente están dados de alta en el sistema.
2. El personal debe haber validado su número de usuario y su contraseña.
3. El cliente debe haber elegido y pagado un paquete.
4. El cliente aún no debe haber terminado el tratamiento en curso.

3.8.4. Poscondiciones.

El sistema registra la sesión para determinado día, hora y cabina, y otorga el privilegio de calendarizar más sesiones hasta que se cumplan las 20 sesiones que conforman el paquete.

3.9. Caso de uso *Acudir a sesiones.*

3.9.1. Descripción.

El cliente acude a la sesión correspondiente al paquete adquirido incialmente.

3.9.2. Flujo de eventos.

Flujo básico.

1. El cliente acude a la sesión que calendarizó con anterioridad.
2. El personal registra los avances obtenidos en la sesión.
3. El cliente se encuentra en la posibilidad de calendarizar una nueva sesión.

Flujos alternativos.

- **En paso 1**: Si el cliente canceló dicha sesión con 24 horas de antelación, se encuentra en la posibilidad de recalendarizar esa sesión; caso contrario, se toma como sesión dada, sin posibilidad de recalendarización.
- **En paso 2**: consultar la especificación **Registrar avances de sesión**.
- **En paso 3**: consultar la especificación **Calendarizar sesiones**.

3.9.3. Precondiciones.

1. El personal y el cliente están dados de alta en el sistema.
2. El personal debe haber validado su número de usuario y su contraseña.
3. El cliente debe haber elegido y pagado un paquete.
4. El cliente aún no debe haber terminado el tratamiento en curso.
5. El cliente debe haber asistido a dicha sesión.

3.9.4. Poscondiciones.

Una vez concluida la sesión, y habiendo registrado los avances de ésta, el expediente del cliente se actualiza.

3.10. Caso de uso *Registrar avances de sesión.*

3.10.1. Descripción.

El personal registra las medidas correspondientes al tratamiento del cliente por sesión, para así conocer su evolución.

3.10.2. Flujo de eventos.

Flujo básico.

1. El personal ingresa al sistema.
2. El personal accede al apartado correspondiente al registro de avances de sesión.
3. Procede a ingresar las medidas del cliente obtenidas en la sesión, y de existir algún comentario relevante, lo anota en el campo correspondiente.
   a) El cliente puede calendarizar otra sesión.
   b) El personal regresa a la pantalla principal del subsistema *Clientes.*

Flujos alternativos.

- **En paso 3.1**: consultar la especificación **Calendarizar sesiones**.

3.10.3. Precondiciones.

1. El personal y el cliente están dados de alta en el sistema.
2. El personal debe haber validado su número de usuario y su contraseña.
3. El cliente debe haber elegido y pagado un paquete.
4. El cliente aún no debe haber terminado el tratamiento en curso.

3.10.4. Poscondiciones.

Cuando son registradas las medidas de la sesión, el expediente del cliente se actualiza; éste puede o no solicitar la calendarización de otra sesión.

3.11. Caso de uso *Ingresar usuario.*

3.11.1. Descripción.

El personal da de alta a un usuario, dependiendo de su nivel de acceso al sistema.

3.11.2. Flujo de eventos.

Flujo básico.

1. El personal ingresa al subsistema correspondiente (*Clientes* o *Personal*).
2. El usuario proporciona los datos solicitados por el personal.
3. El personal registra los datos en el sistema.
4. El sistema valida y carga valores extras por defecto, dependiendo del tipo de usuario.

a) Si el usuario ingresado es un nuevo cliente, se carga el número de usuario del personal que lo ingresó y la fecha de ingreso.

b) Si el usuario ingresado es un nuevo miembro del personal, se inicializa su número de clientes ingresados a *cero*.

Flujos alternativos.

- **En paso 4**: Si los datos ingresados no son válidos, el sistema envía al usuario un mensaje de error para su corrección inmediata.

3.11.3. Precondiciones.

1. El personal que ingresa al usuario está dado de alta en el sistema.
2. El personal que ingresa al usuario valida su número de usuario y su contraseña.

3.11.4. Poscondiciones.

El sistema envía al personal a la pantalla principal del subistema correspondiente, mostrando el registro recién ingresado.

3.12. Caso de uso *Consultar expedientes.*

3.12.1. Descripción.

El personal realiza la búsqueda de datos de un usuario, dependiendo su nivel de acceso.

3.12.2. Flujo de eventos.

Flujo básico.

1. El personal ingresa la totalidad o parte del número de usuario del usuario a localizar (búsqueda exacta o aproximada, respectivamente).
2. El sistema despliega las coincidencias de la búsqueda realizada.

3. El personal da clic en el ícono correspondiente a *Más detalles* para conocer los datos completos del usuario.

   a) El personal realiza la consulta deseada, y puede realizar otras operaciones con dicho registro.

   b) El personal regresa a la pantalla principal del subsistema relacionado con el usuario de búsqueda.

Flujos alternativos.

- **En paso 2**: De no existir coincidencia alguna, el sistema despliega un mensaje en pantalla sobre la inexistencia del registro.

3.12.3. Precondiciones.

El personal está dado de alta en el sistema, y ha validado su número de usuario y su contraseña.

3.12.4. Poscondiciones.

Una vez localizado el usuario en cuestión, el personal se encuentra en posibilidad de realizar otras operaciones con dicho registro.

3.13. Caso de uso *Actualizar expedientes.*

3.13.1. Descripción.

El personal modifica uno o varios datos de un usuario, dependiendo su nivel de acceso.

3.13.2. Flujo de eventos.

Flujo básico.

1. El personal ingresa la totalidad o parte del número de usuario del usuario a localizar (búsqueda exacta o aproximada, respectivamente).

2. El sistema despliega las coincidencias de la búsqueda realizada.

3. El personal da clic en el ícono correspondiente a *Actualizar* para ingresar a la pantalla de modificación de datos del usuario.

   a) El personal captura los datos del usuario que se requieran actualizar.

   b) Los datos son actualizados en el sistema.

   c) El personal regresa a la pantalla principal del subsistema relacionado con el usuario de búsqueda.

Flujos alternativos.

- **En paso 2**: De no existir coincidencia alguna, el sistema despliega un mensaje en pantalla sobre la inexistencia del registro.

3.13.3. Precondiciones.

El personal está dado de alta en el sistema, y ha validado su número de usuario y su contraseña.

3.13.4. Poscondiciones.

Una vez actualizados los datos correspondientes del usuario en cuestión, el sistema envía al personal a la pantalla del subsistema correspondiente, con el registro modificado.

3.14. Casos de uso con *Rational Rose.*

Se exponen en esta sección los casos de uso enfocados al diseño del sistema, tomando en consideración los casos de uso del negocio presentados anteriormente.

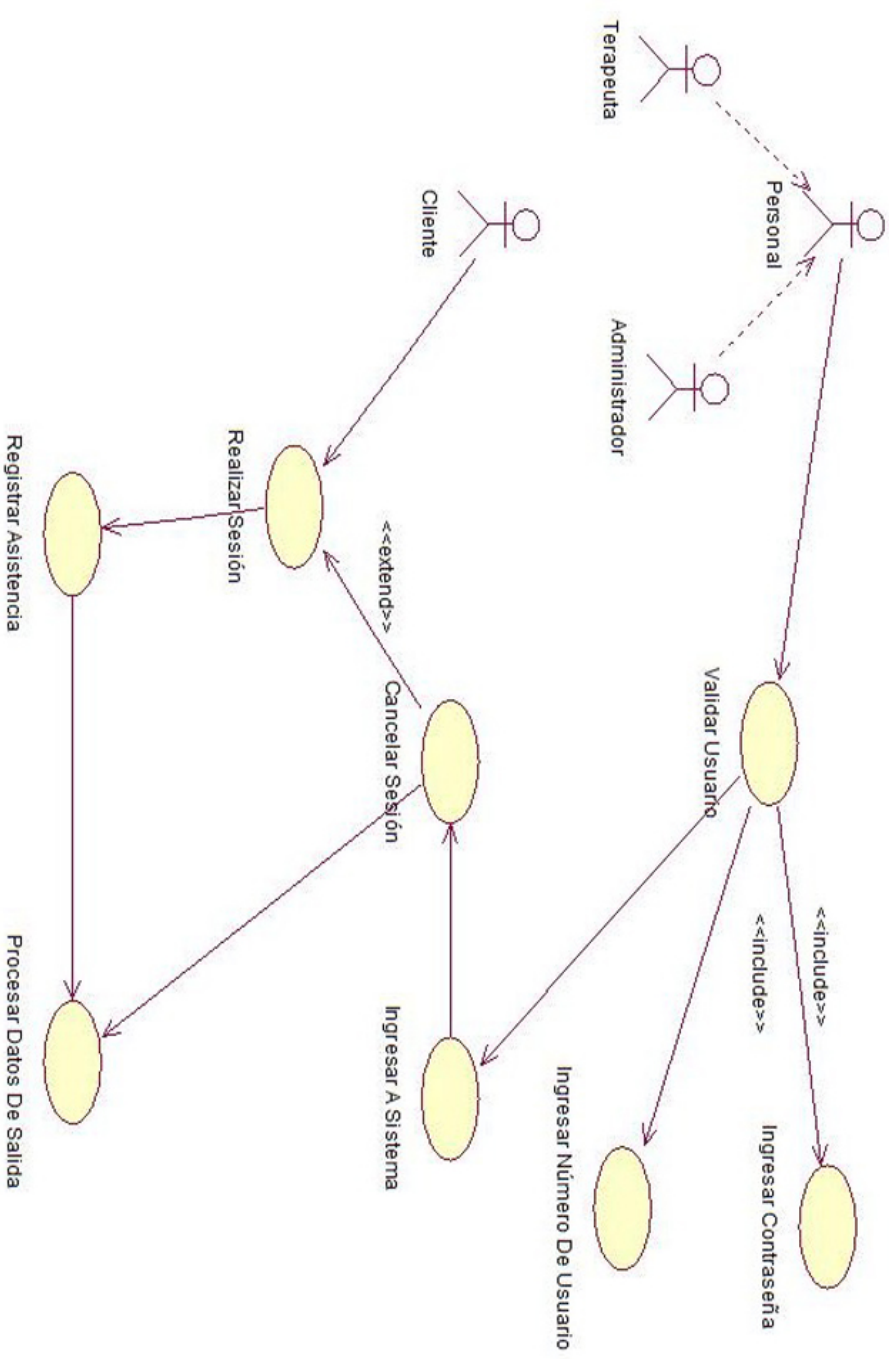


Figura 3.9: Caso de uso *Acudir a sesiones.*

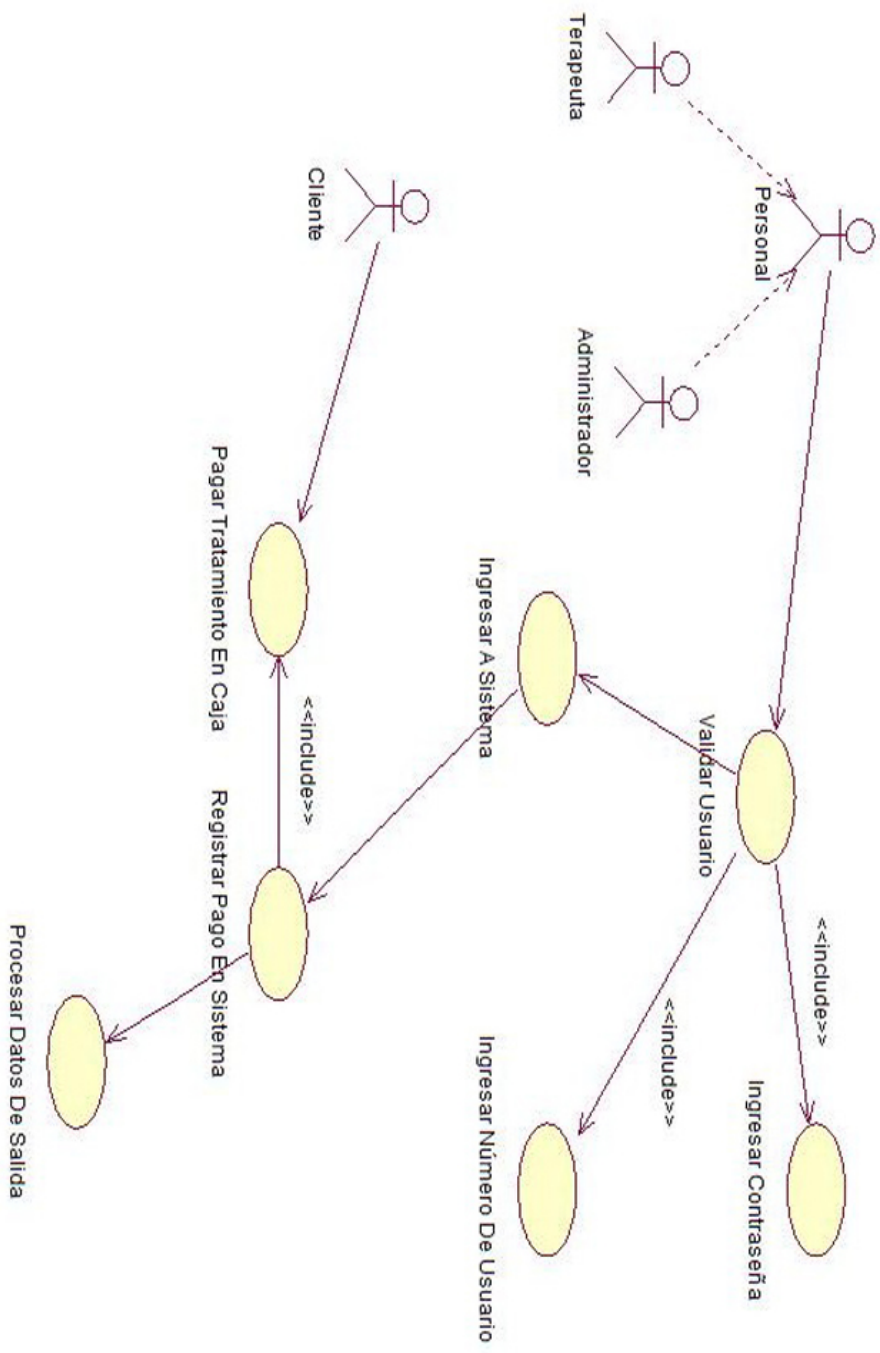


Figura 3.10: Caso de uso *Adquirir tratamiento*.

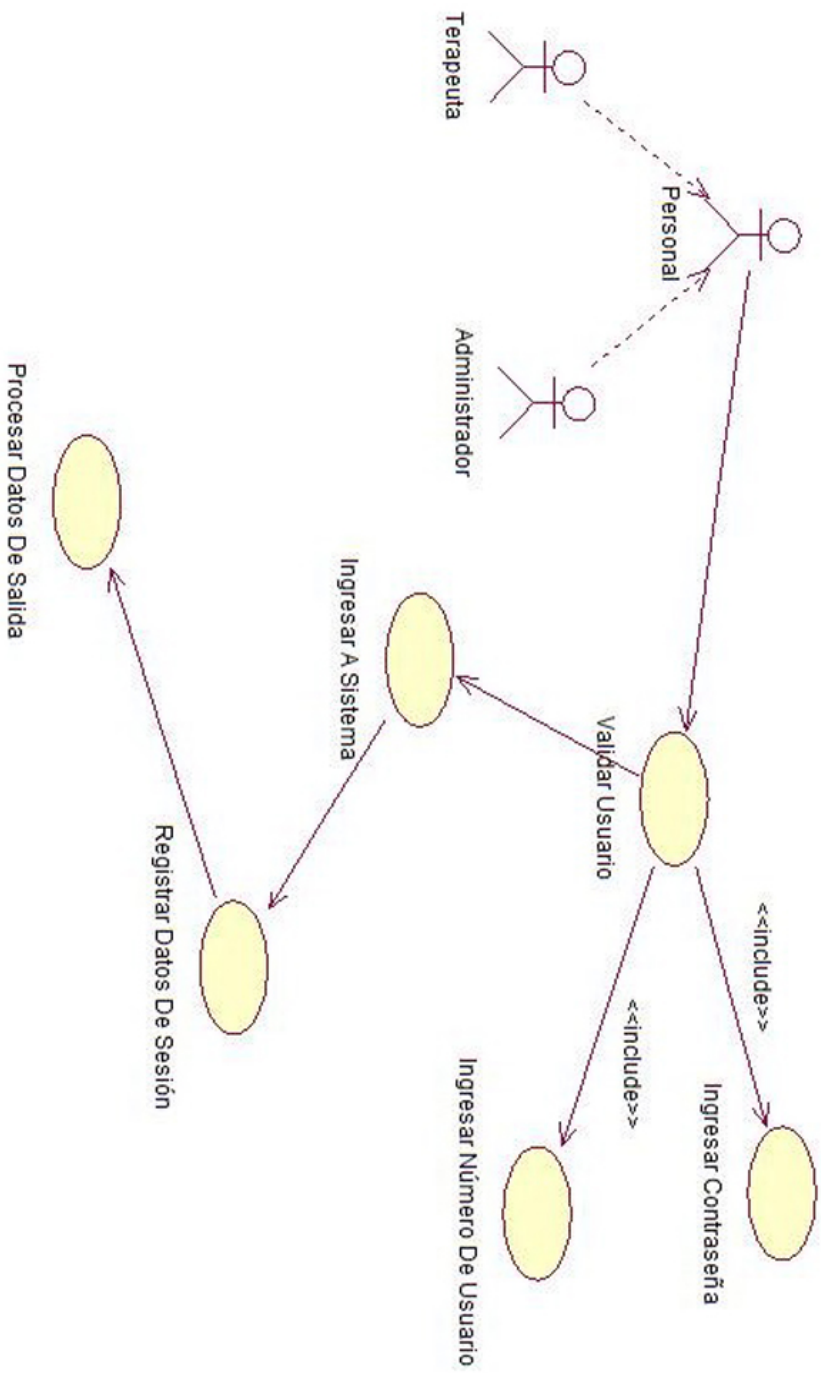


Figura 3.11: Caso de uso *Calendarizar sesiones.*

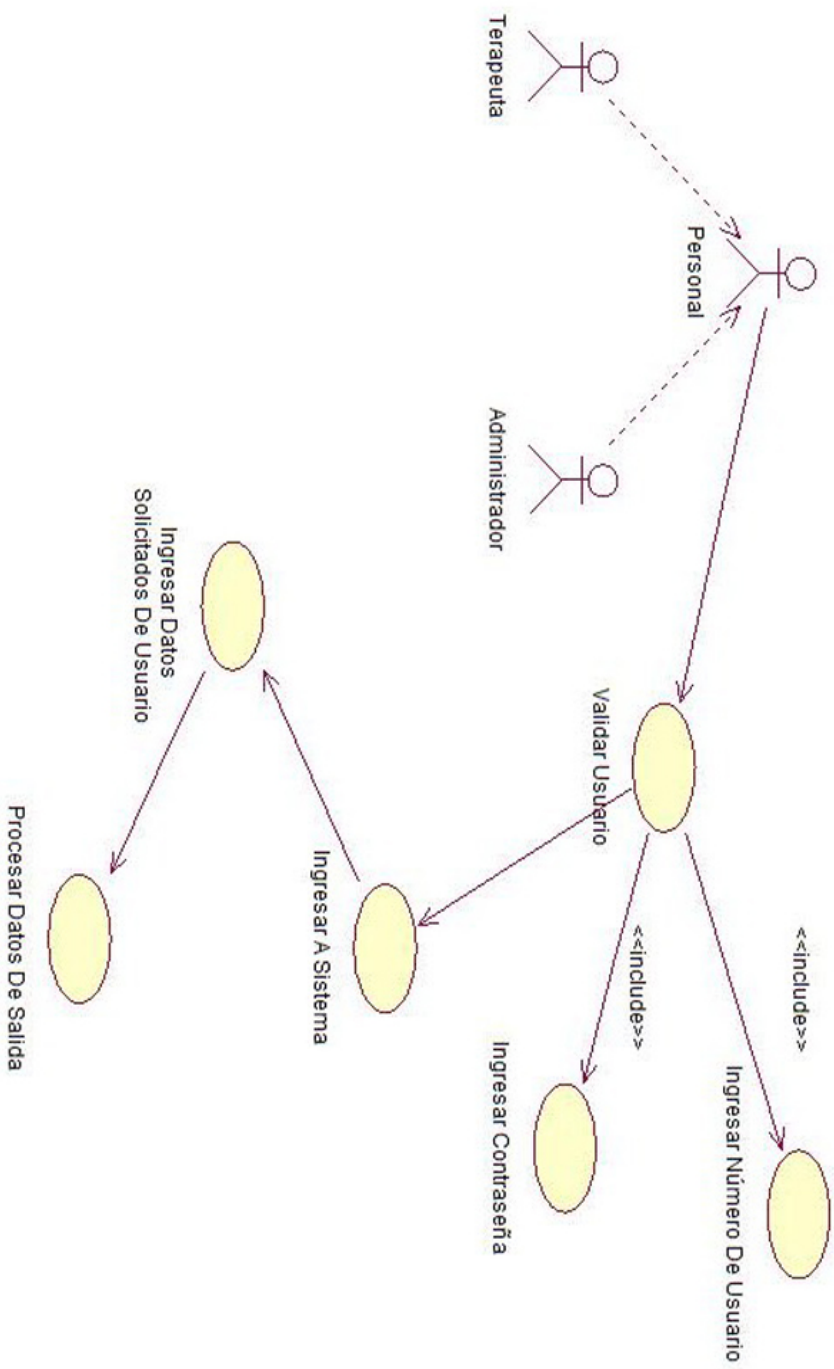


Figura 3.12: Caso de uso *Ingresar usuario*.

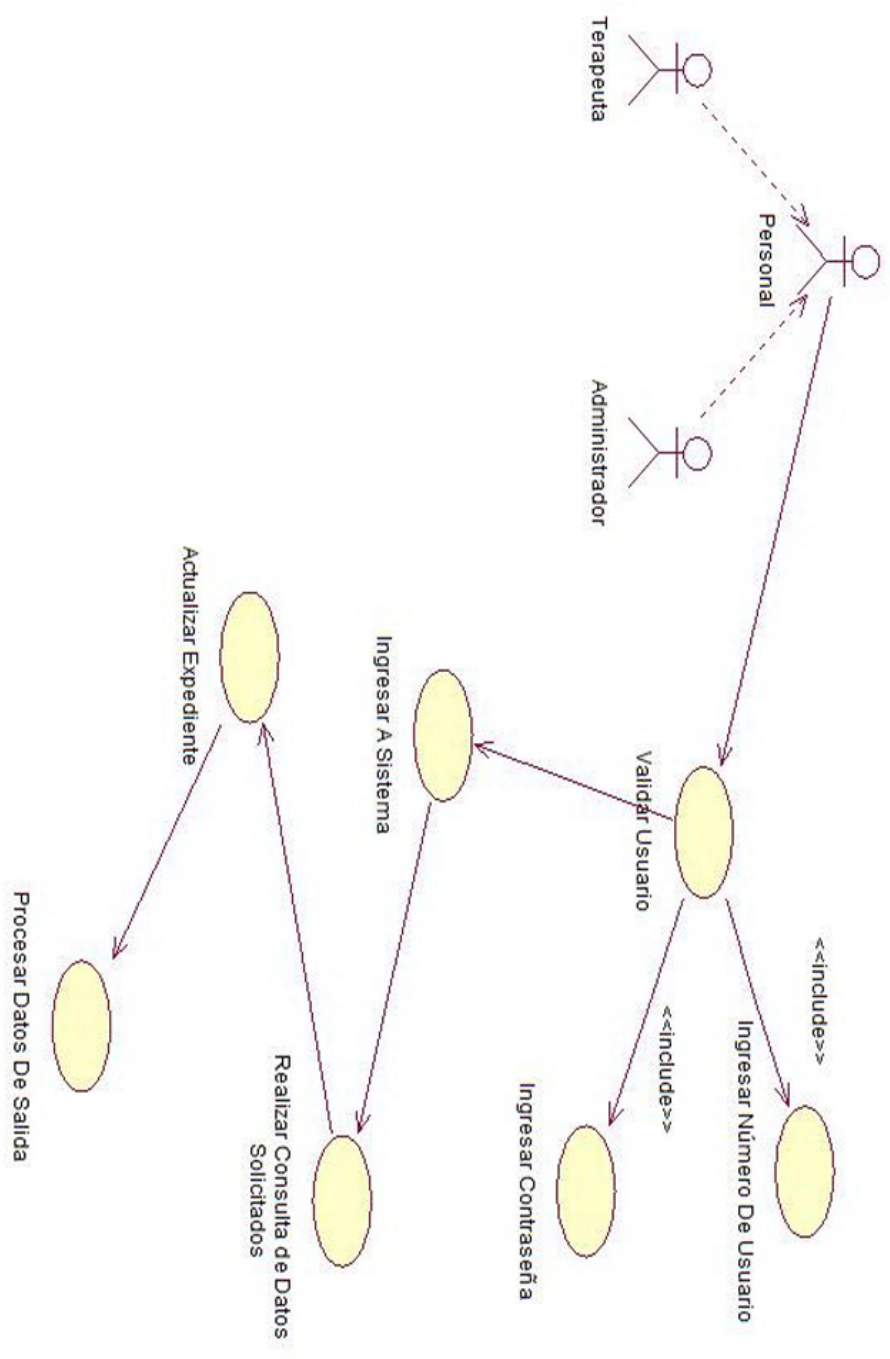


Figura 3.13: Caso de uso *Registrar avances de sesión.*

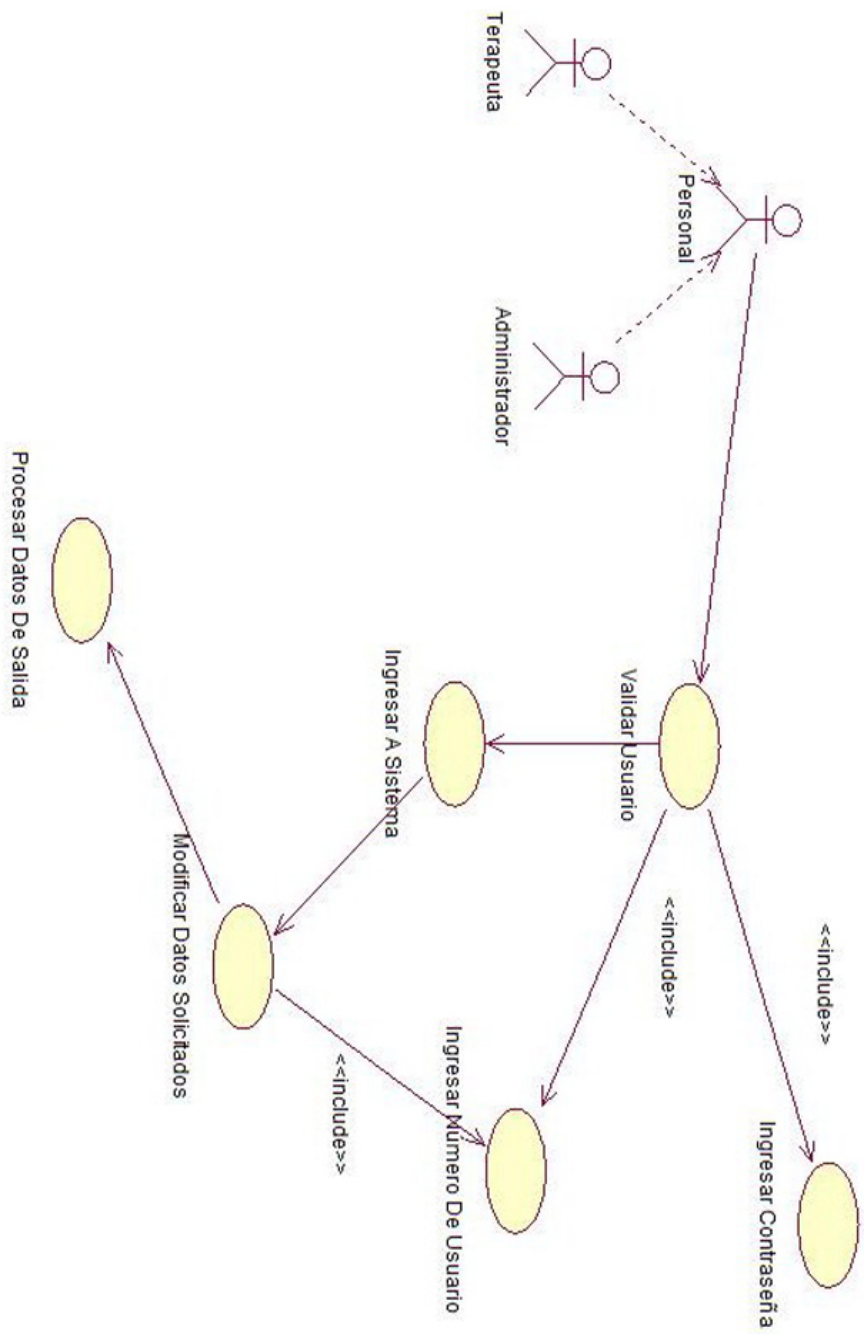


Figura 3.14: Caso de uso *Actualizar expediente*.

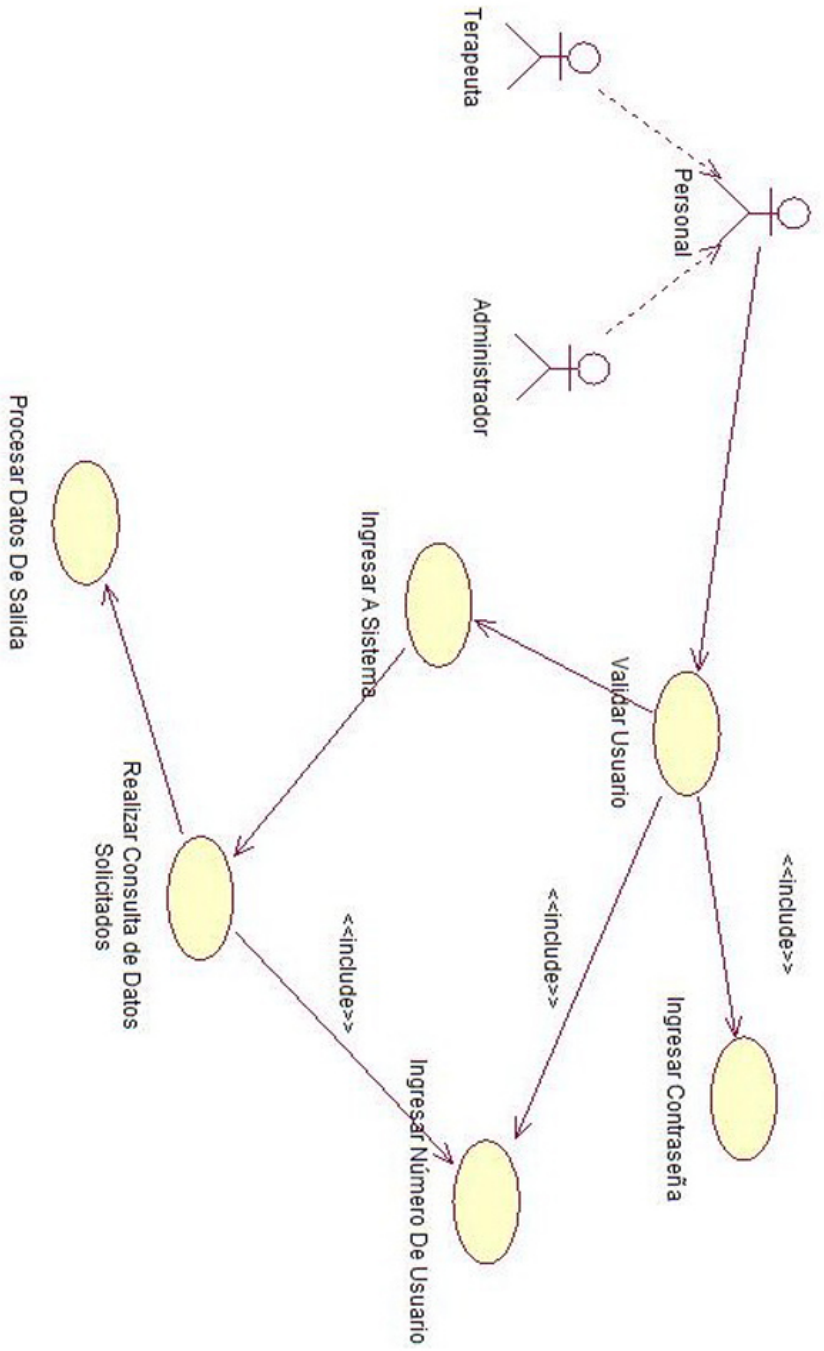


Figura 3.15: Caso de uso *Consultar expediente*.

3.15. Representación en términos de análisis.

Basado en la información recopilada durante el Modelado del Negocio y la captura de Requisitos de Casos de Uso, se procede a derivar ciertos elementos claves para lograr la construcción de la primera representación del sistema. El diagrama de clases resultante es el presentado en la figura 3.16.

Cabe resaltar que en este diagrama ya se contempla la aparición de sucursales, debido a que, hasta el momento, no se ha logrado la consolidación de alguna sucursal; sin embargo, el diagrama queda abierto para el trabajo a futuro (ver *Trabajo a futuro*).

Se ha empleado únicamente la herramienta *Rational Rose*; sin embargo, para el modelo relacional se utiliza *CASEStudio ver. 2.25*, con la finalidad de obtener el *script* de MySQL (ver apéndice C). La figura 3.17 presenta dicho modelo, y el diccionario de datos correspondiente a cada entidad se muestra a continuación de él.

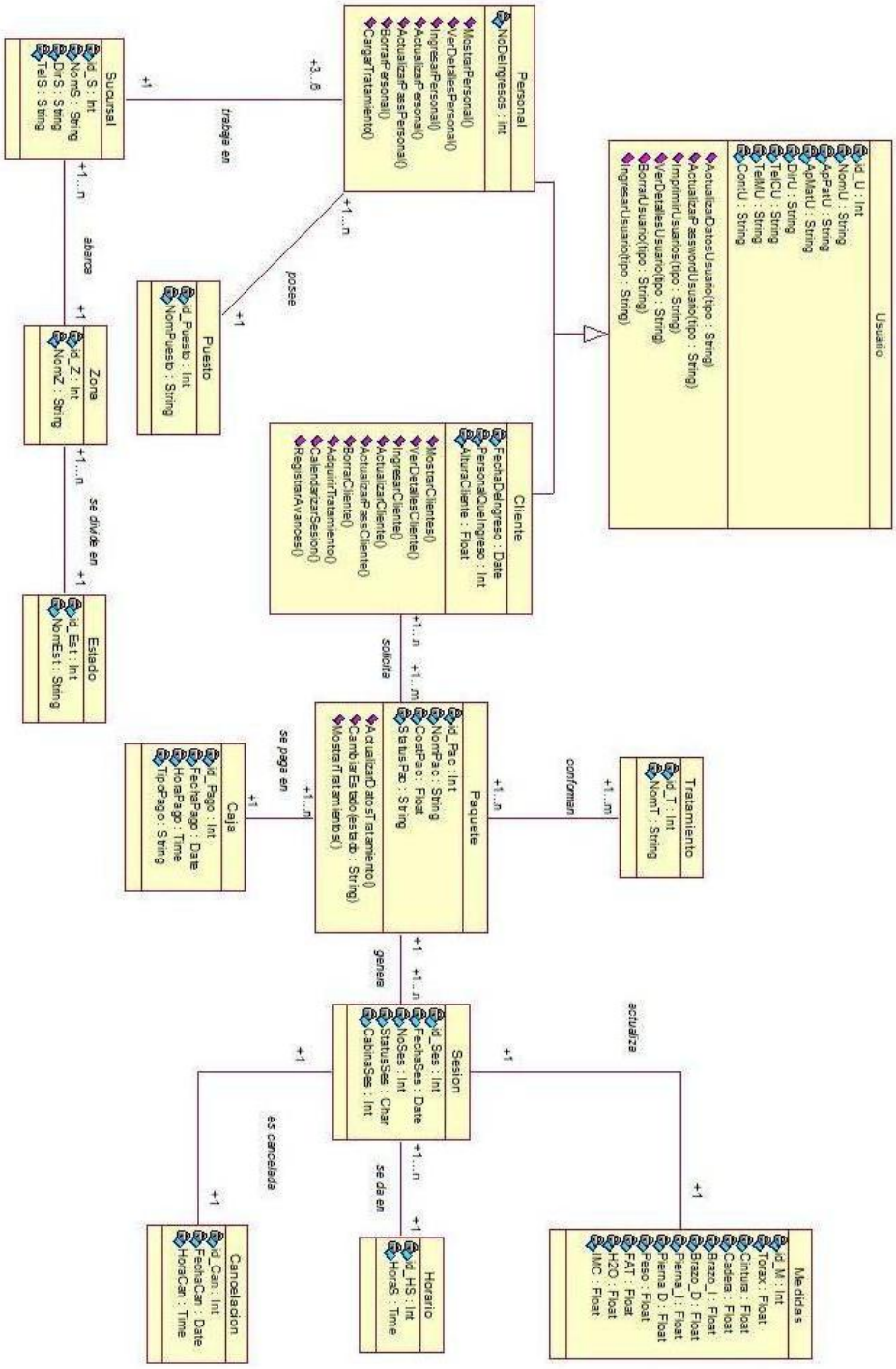


Figura 3.16: Diagrama de clases del sistema de información *Svelta*.

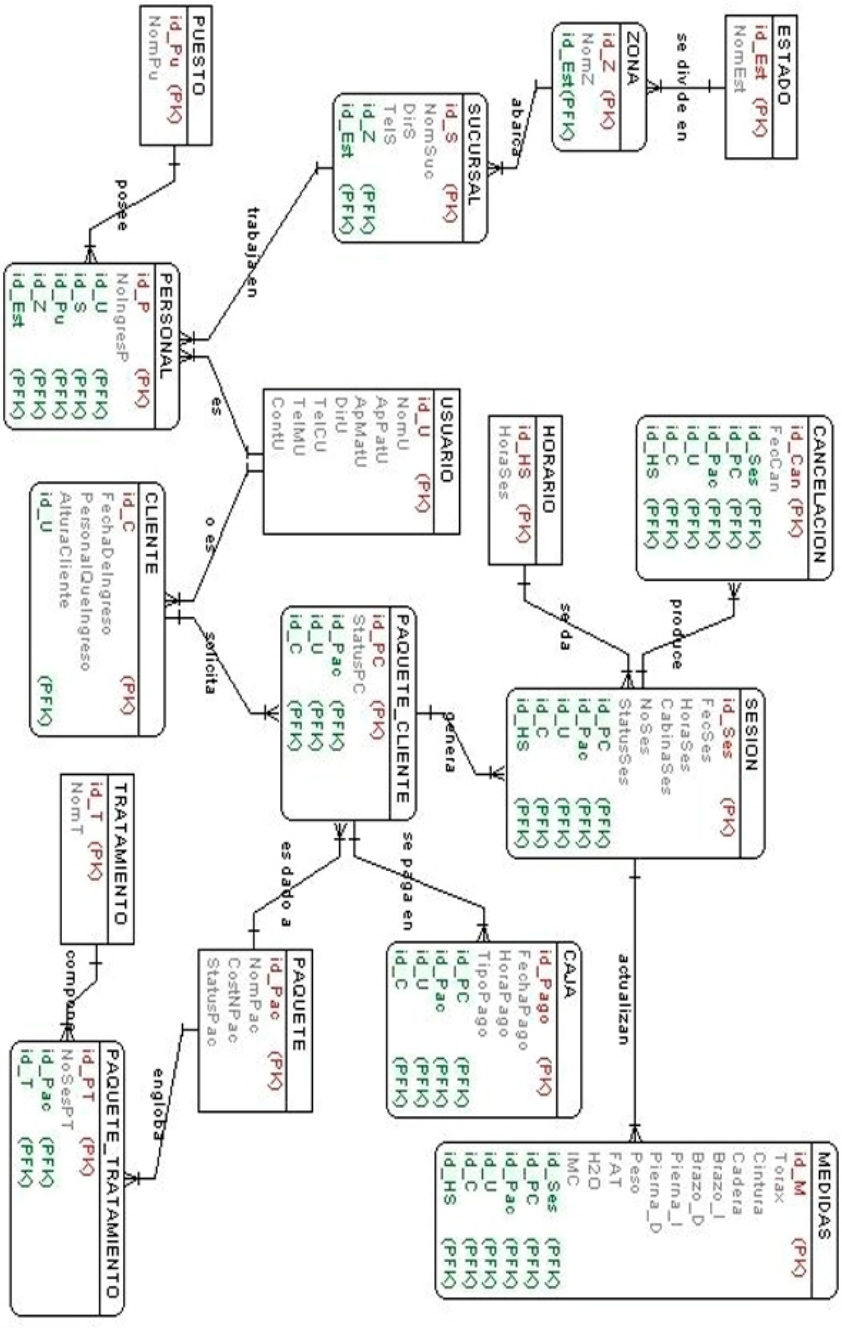
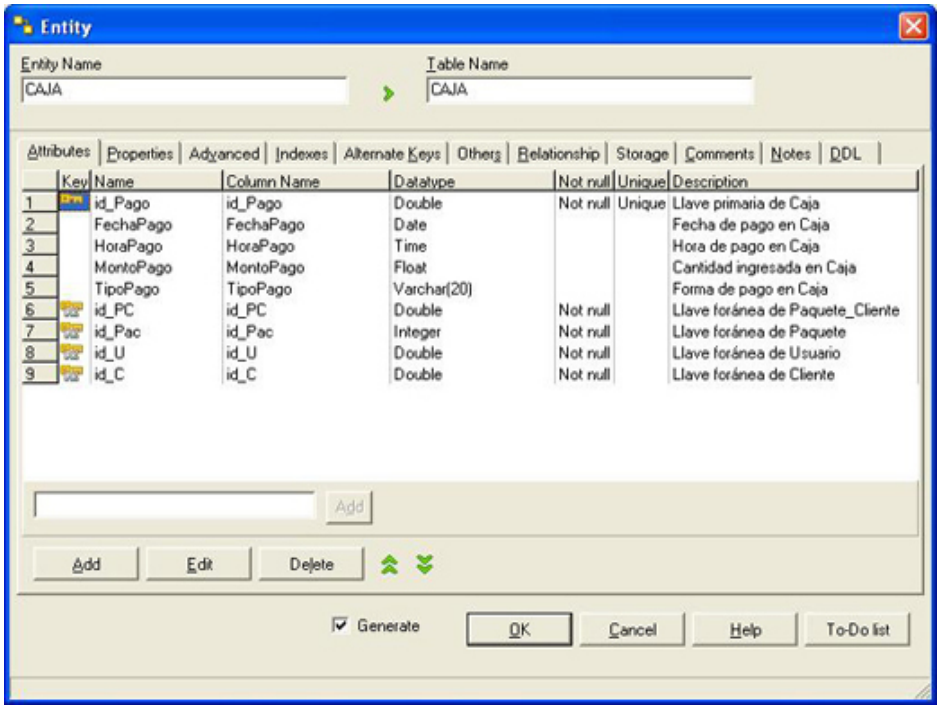


Figura 3.17: Modelo relacional de *Svelta*.



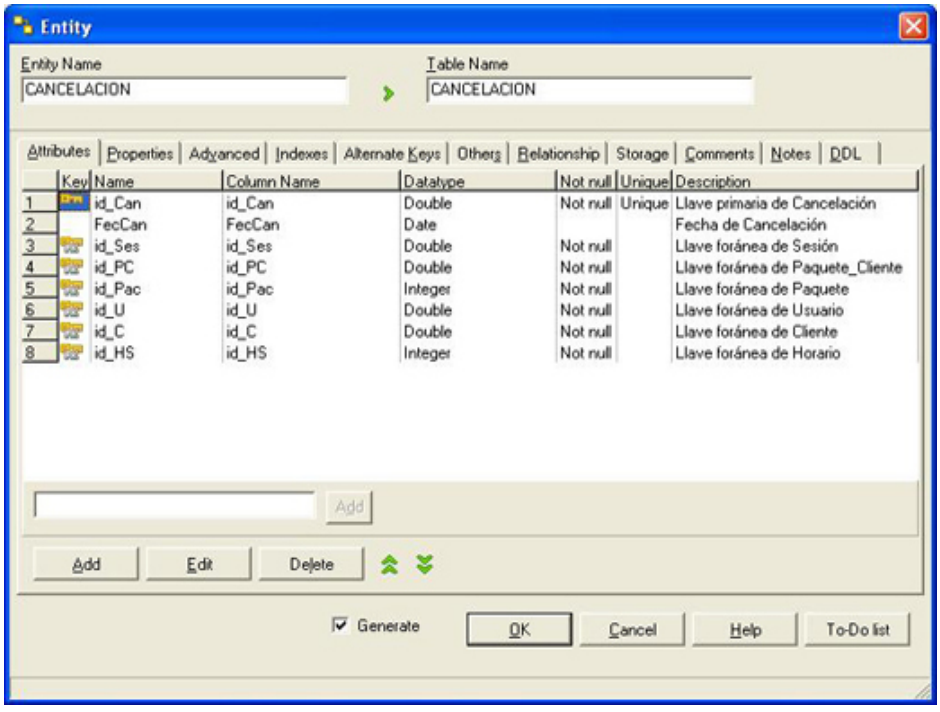
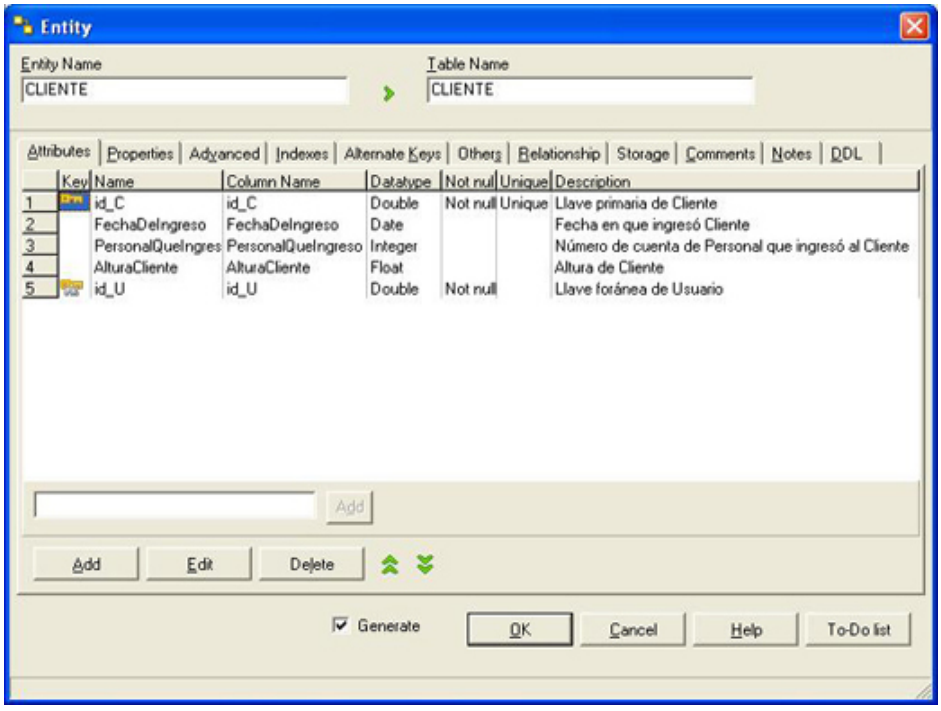


Figura 3.18: Diccionario de Datos: *Caja/Cancelación.*



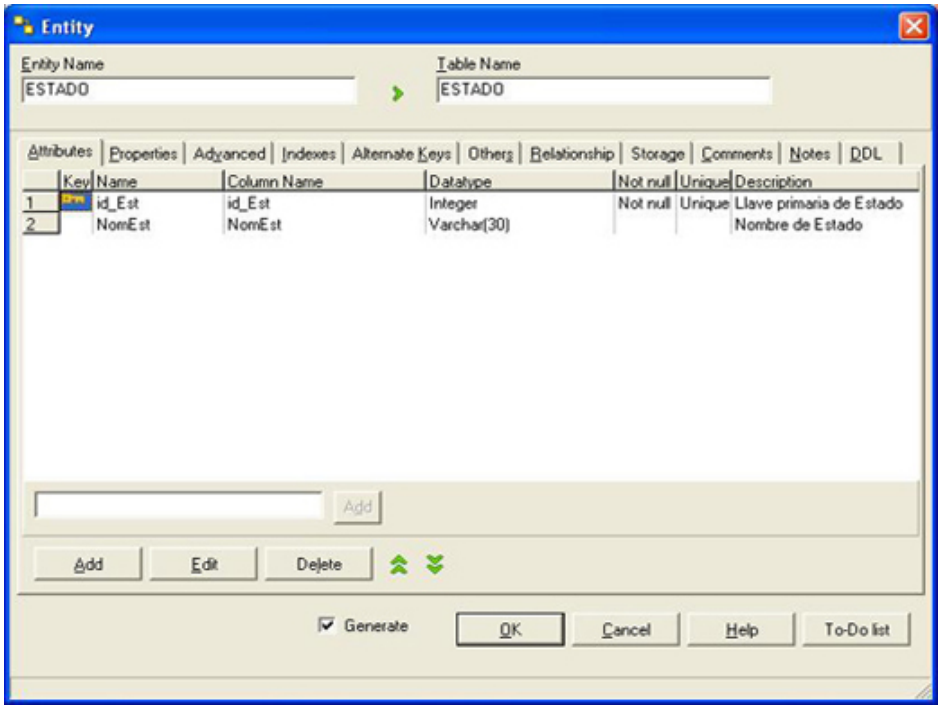
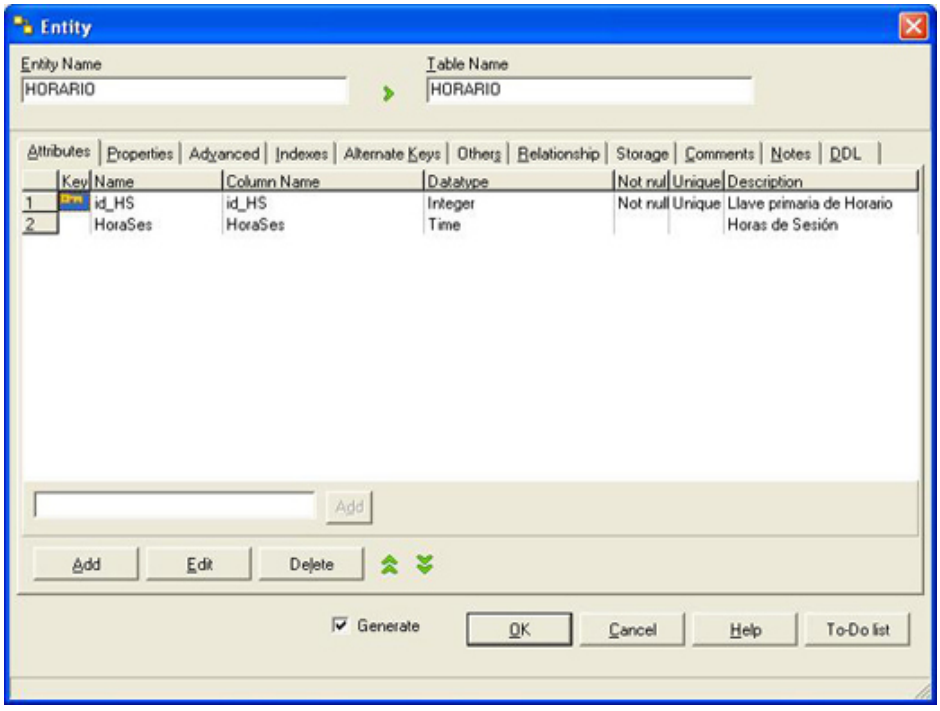


Figura 3.19: Diccionario de Datos: *Cliente/Estado*.



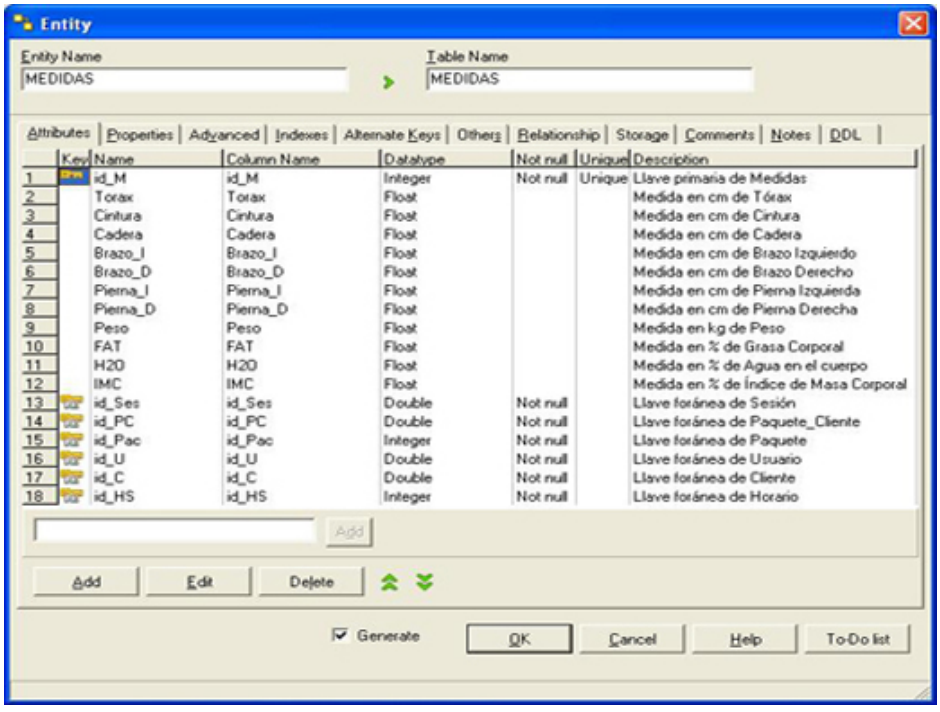
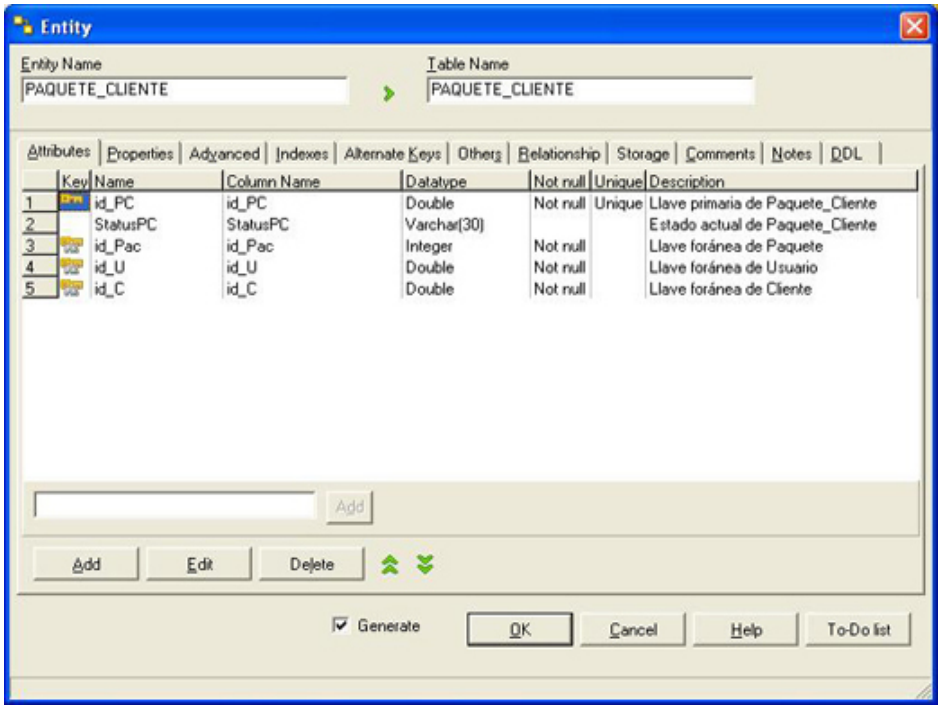


Figura 3.20: Diccionario de Datos: *Horario/Medidas*.



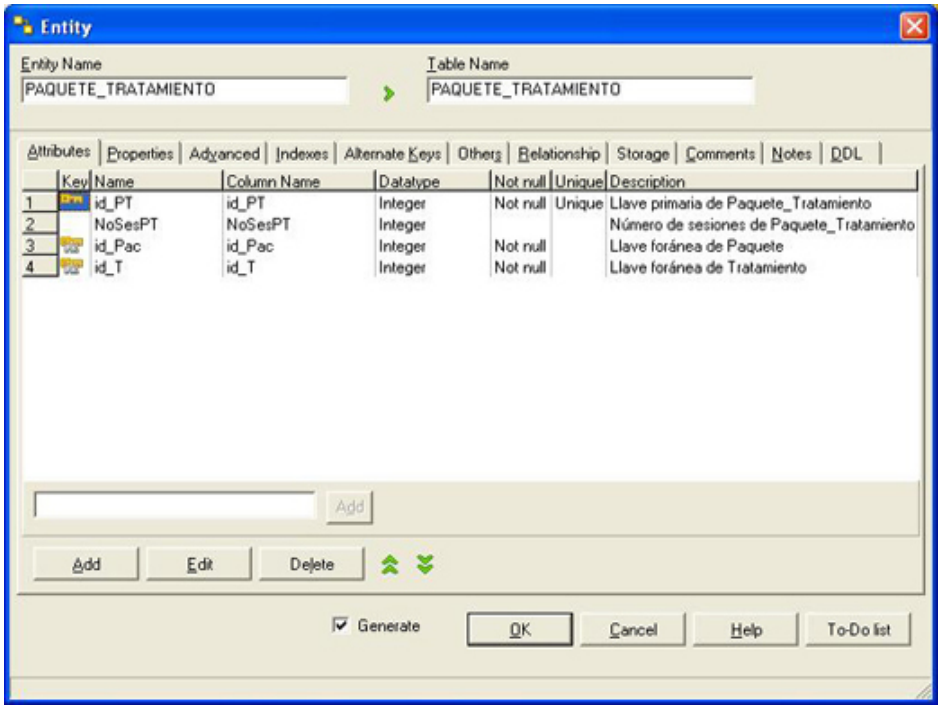
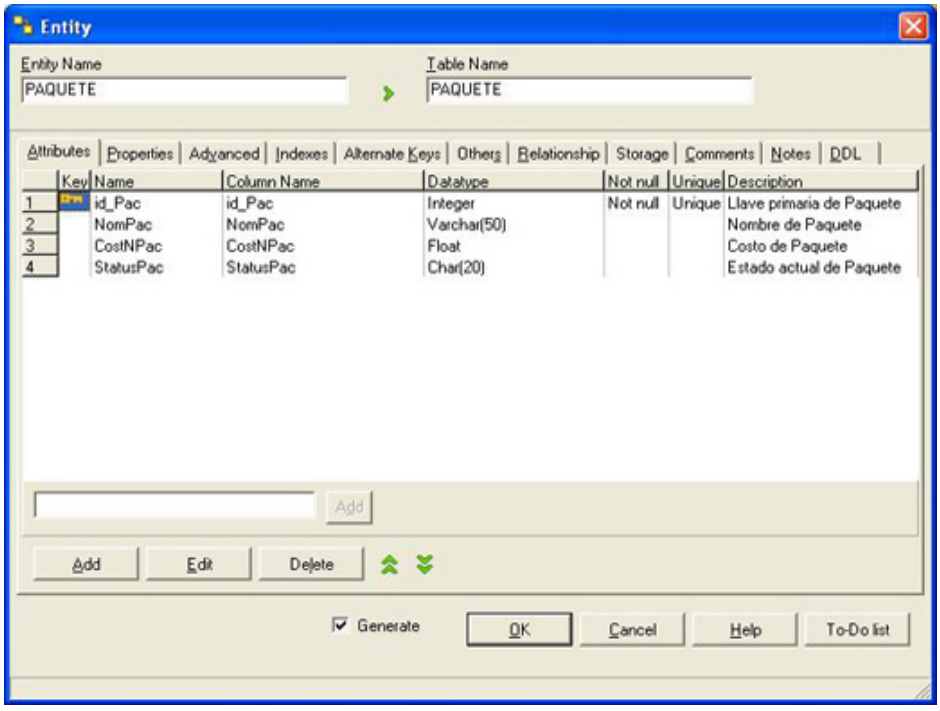


Figura 3.21: Diccionario de Datos: *Paquete_ Cliente*/*Paquete_ Tratamiento*.



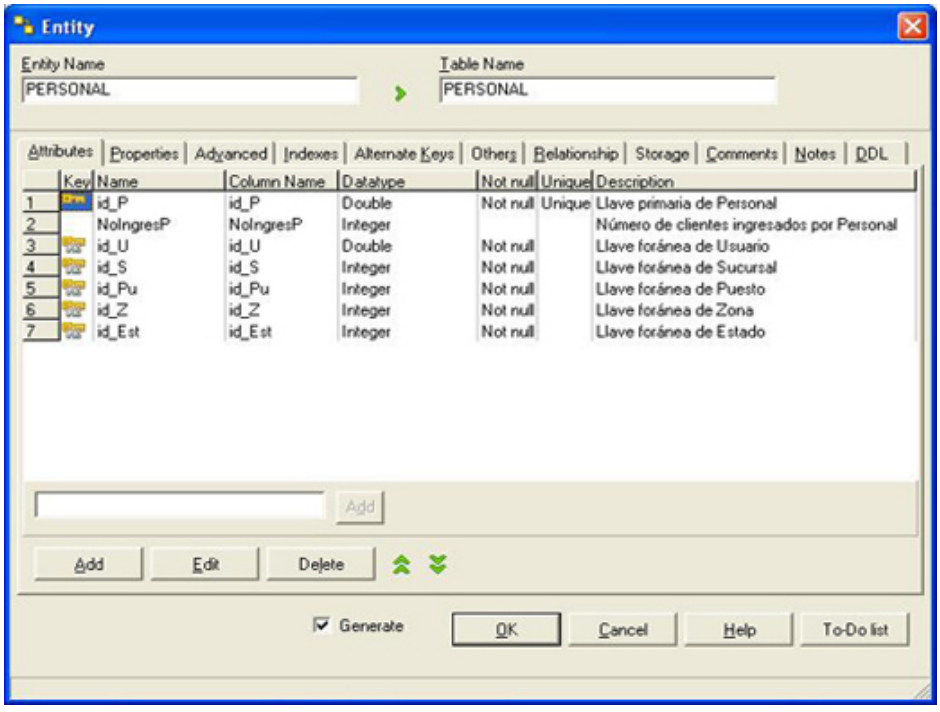
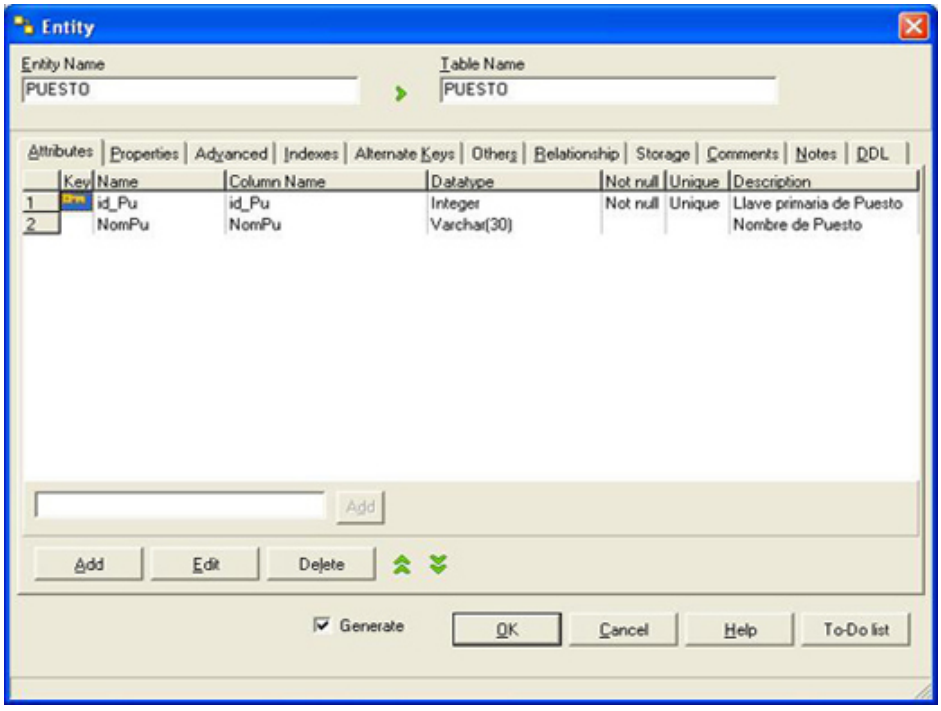


Figura 3.22: Diccionario de Datos: *Paquete/Personal.*



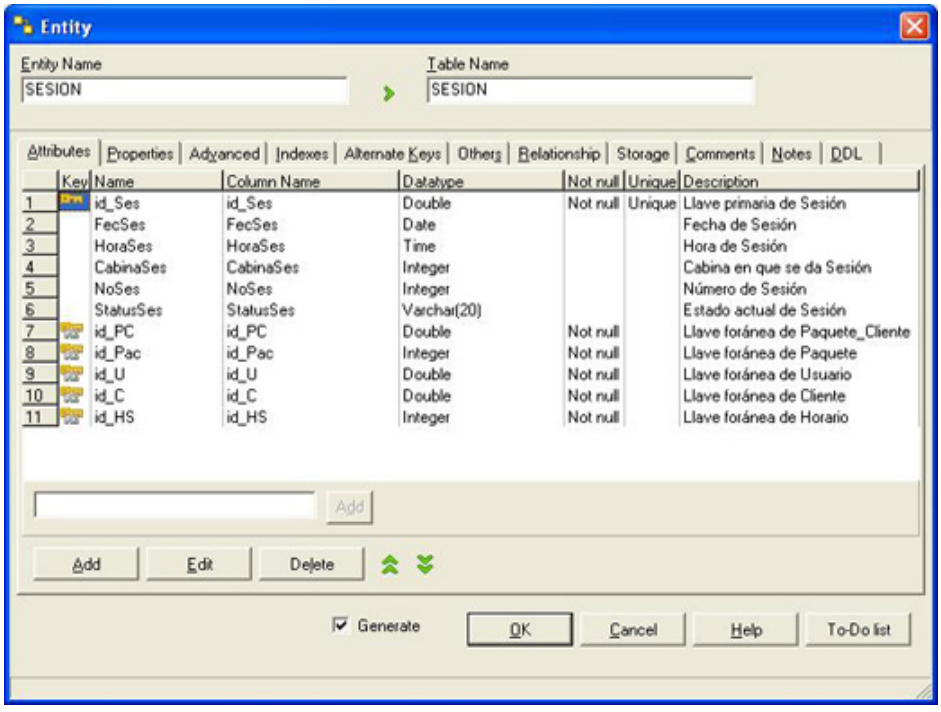
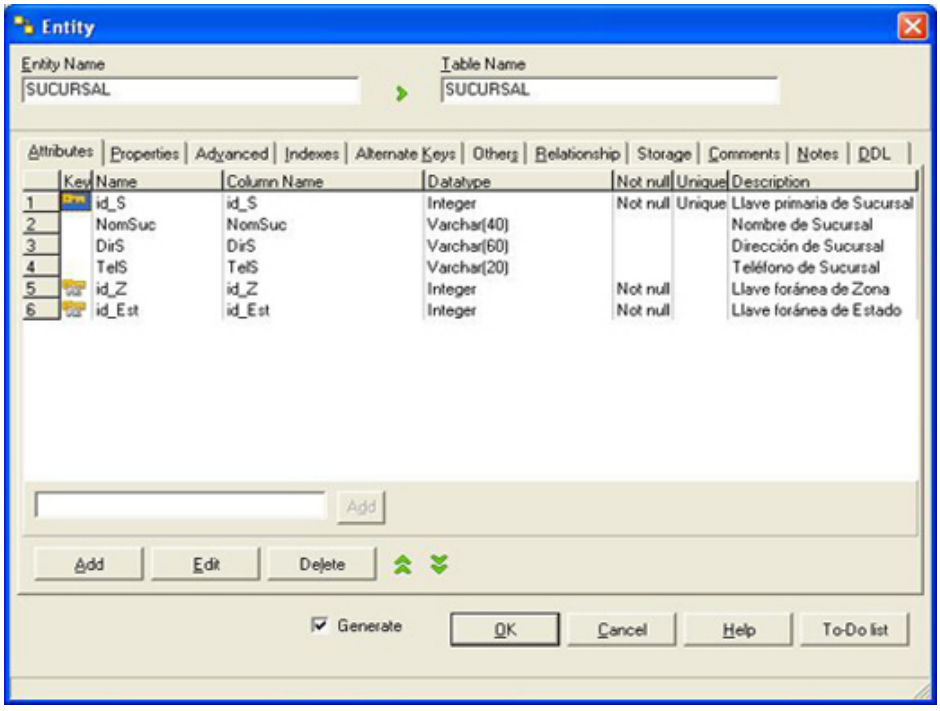


Figura 3.23: Diccionario de Datos: *Puesto/Sesión.*



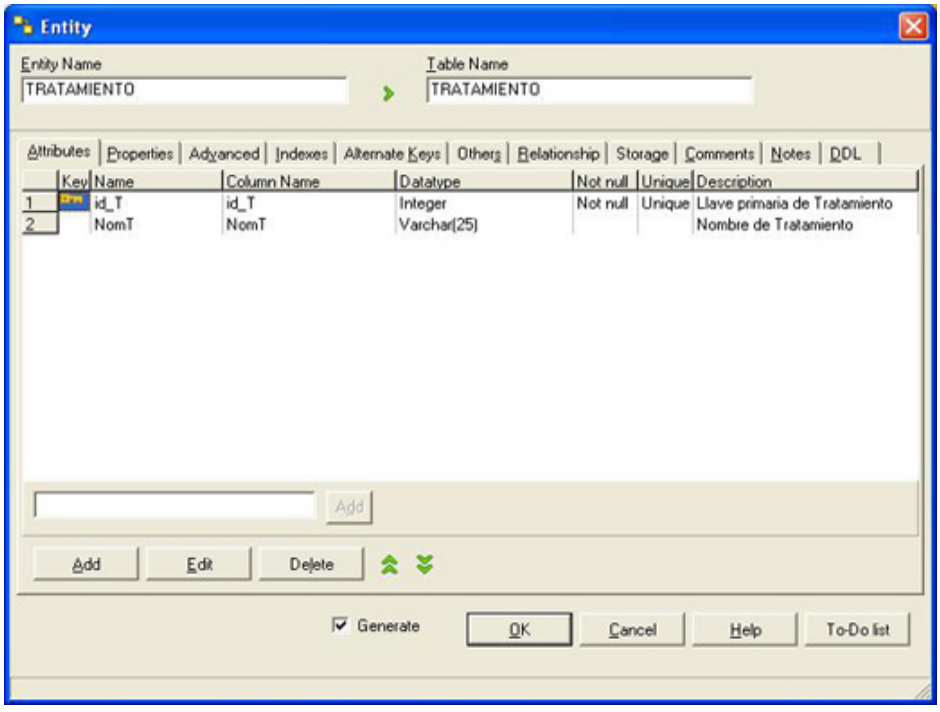
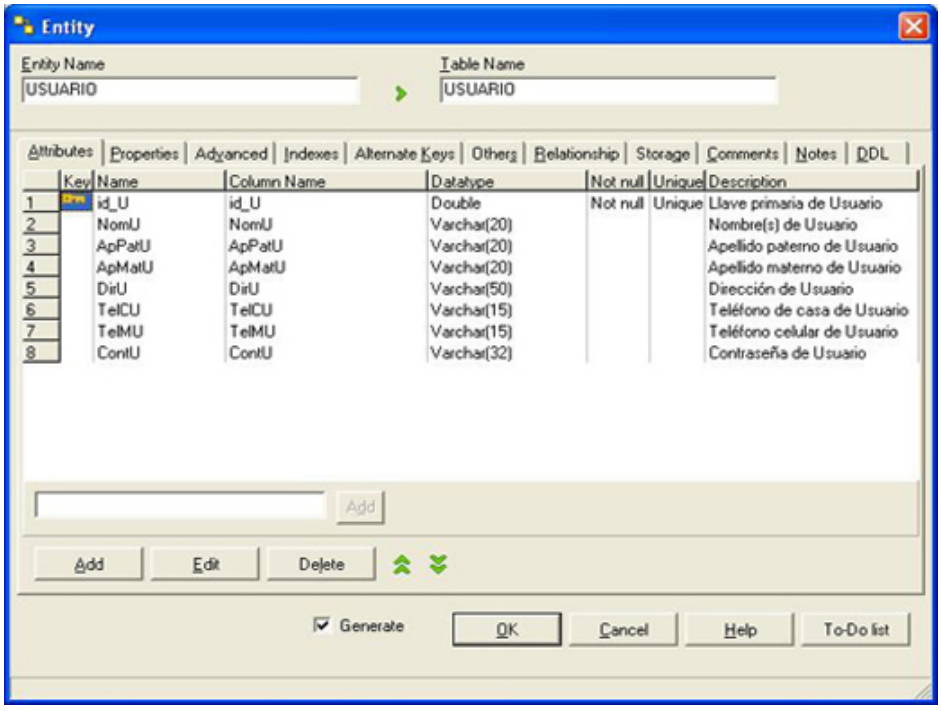


Figura 3.24: Diccionario de Datos: *Sucursal* / *Tratamiento*.



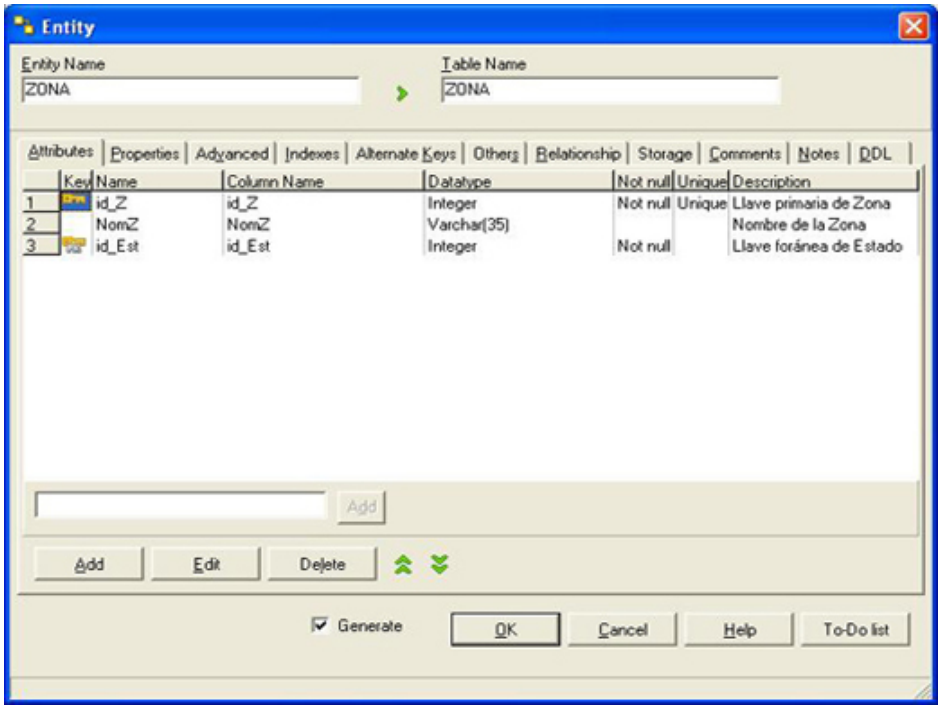


Figura 3.25: Diccionario de Datos: *Usuario/Zona*.

3.16. Representación en términos de diseño.

Para lograr esta representación, se toma en consideración la descripción de la organización descrita anteriormente. Después del análisis al diagrama de actividades sobre la dinámica del negocio, se construye un nuevo diagrama de actividades, el cual queda representado en la fig. 3.26, con el fin de apreciar las actividades que llevará a cabo el sistema.

Con el diagrama anterior, se procede a la elaboración del diagrama de estados resultante, tanto del modelo estático como del modelo dinámico. La fig. 3.27 refleja el diagrama resultante.

Posteriormente, se plantean dos escenarios claves para el mejor comprensión de la dinámica del negocio apoyada por el sistema a desarrollar, así como para facilitar el desarrollo del sistema; éstos serán la inscripción de clientes a algún tratamiento, y el cumplimiento de sesiones, por ser los de mayor prioridad. Se presentan los diagramas de secuencia para dichos escenarios[2] (ver figs. 3.28 - 3.30).

3.16.1. Escenario de *Inscripción.*

1. El cliente pide información al personal acerca del tratamiento, y logra convencerlo de tomar el tratamiento.

2. El cliente firma contrato, una vez que acepta clausulas tanto de la mecánica del tratamiento como de posibles enfermedades que puedan impedir la aplicación del mismo.

3. El cliente es dado de alta, proporcionando datos personales.

4. El cliente paga a caja el consecuente monto requerido.

5. El cliente, una vez hecho esto, solicita al personal que comience a calendarizar sus citas.

6. El personal verifica si existe disponibilidad del día en que el cliente quiere realizar sus sesiones.

[2]El segundo escenario (*Sesiones*) se divide en dos escenarios para mejor entendimiento del mismo.

7. Si existe disponibilidad en el día, el personal calendariza las citas del cliente.

8. El cliente está satisfecho con el trato y dispuesto a iniciar el tratamiento.

3.16.2. Escenario de *Sesiones*.

1. El cliente acude a la sesión de tratamiento en la fecha correspondiente.

2. El personal se encarga de aplicar el tratamiento.

3. Una vez que concluye el tiempo correspondiente de la sesión, se van actualizando las medidas del cliente, para registrar el avance de su tratamiento.

4. El cliente, viendo que su tratamiento es exitoso, programa la(s) siguiente(s) sesión(es).

5. El personal checa la disponibilidad de día para la próxima cita.

6. De existir disponibilidad, el personal calendariza la cita.

7. El cliente al final de un tratamiento o al estar satisfecho antes del mismo, puede decidir ingresar a otro nuevo, repitiendo los pasos 4 a 8 del escenario anterior, o bien traspasar las sesiones restantes a otro cliente.

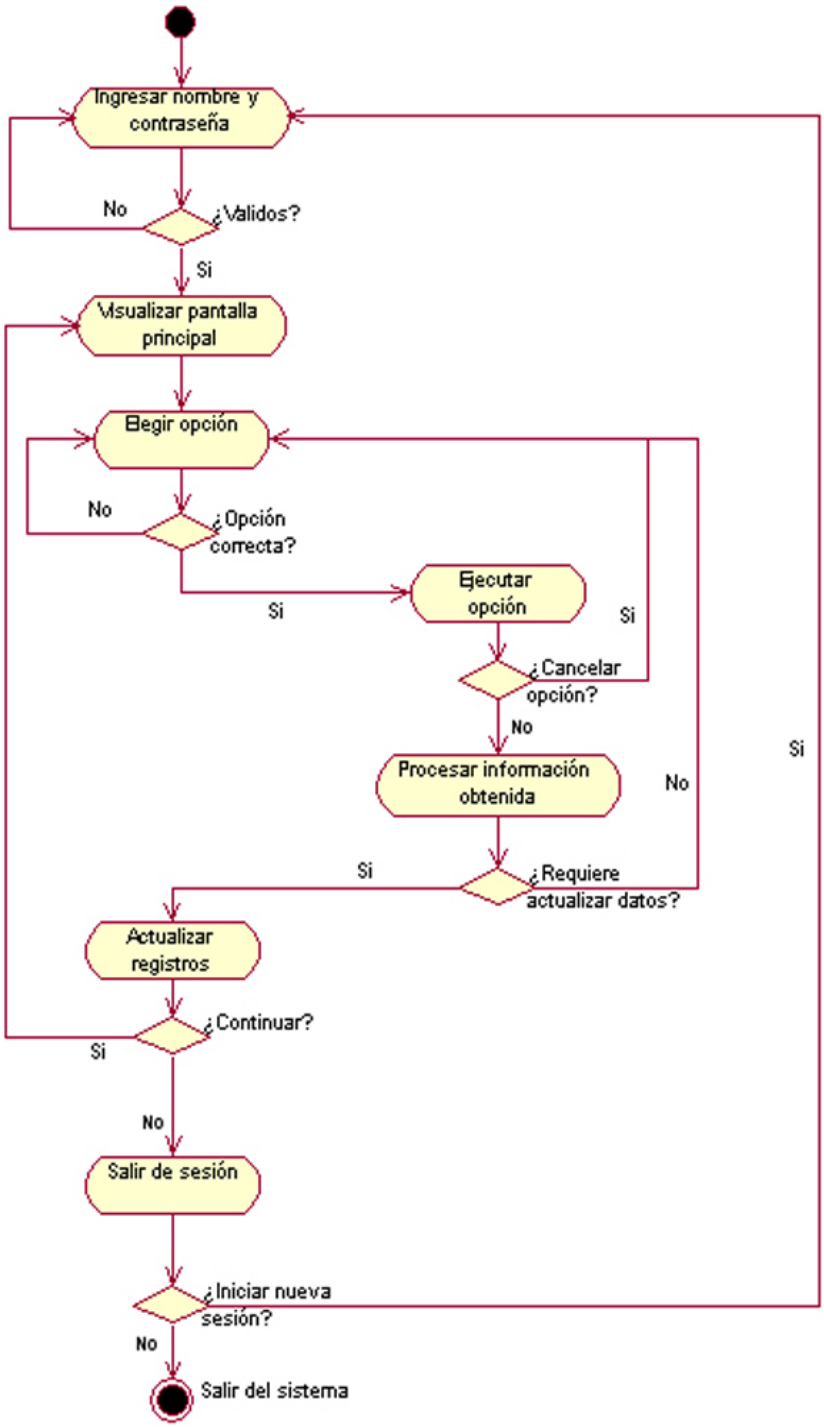


Figura 3.26: Diagrama de actividades del sistema de información *Svelta*.

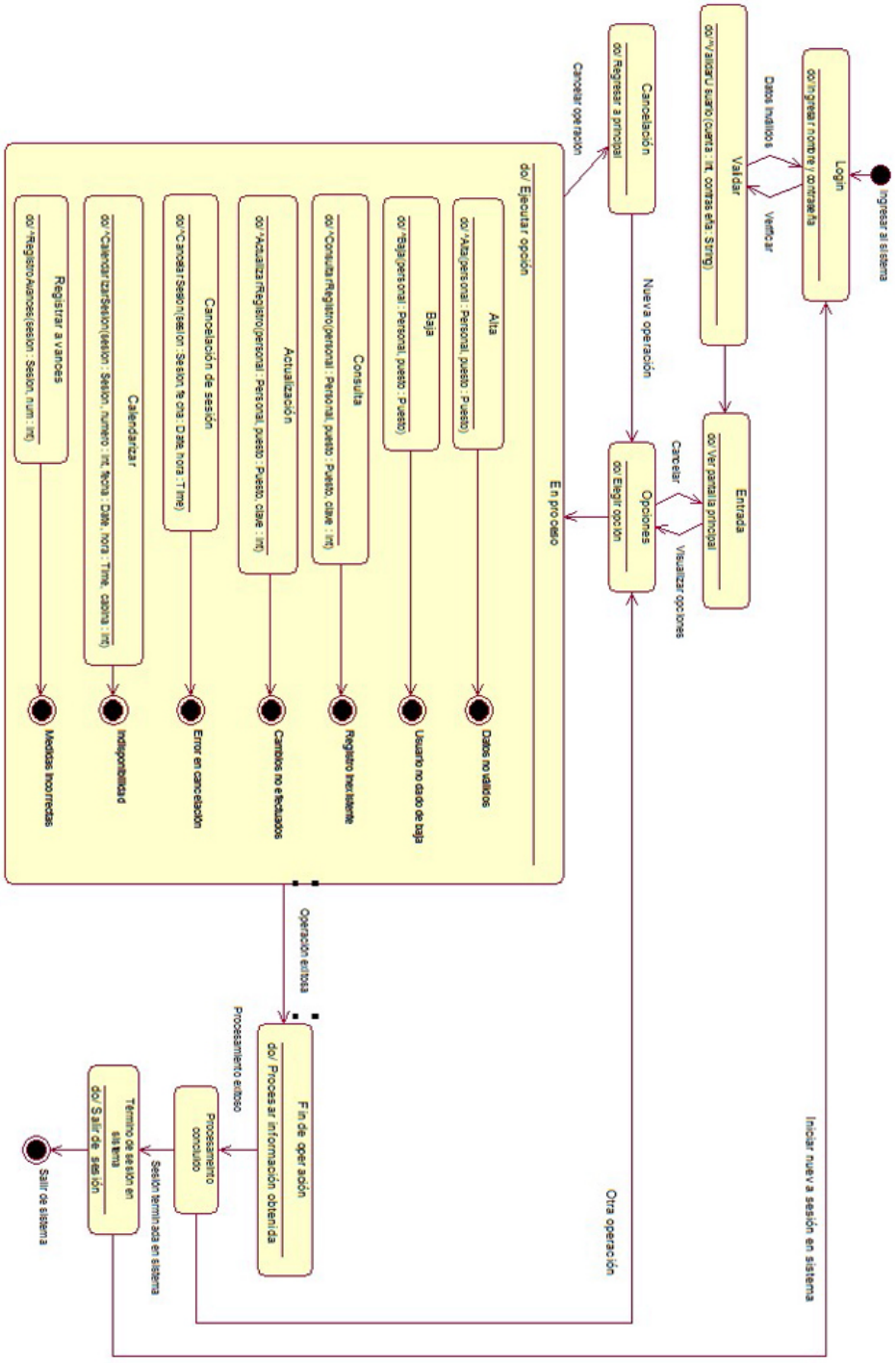


Figura 3.27: Diagrama de estados del sistema de información de *Svelta*.

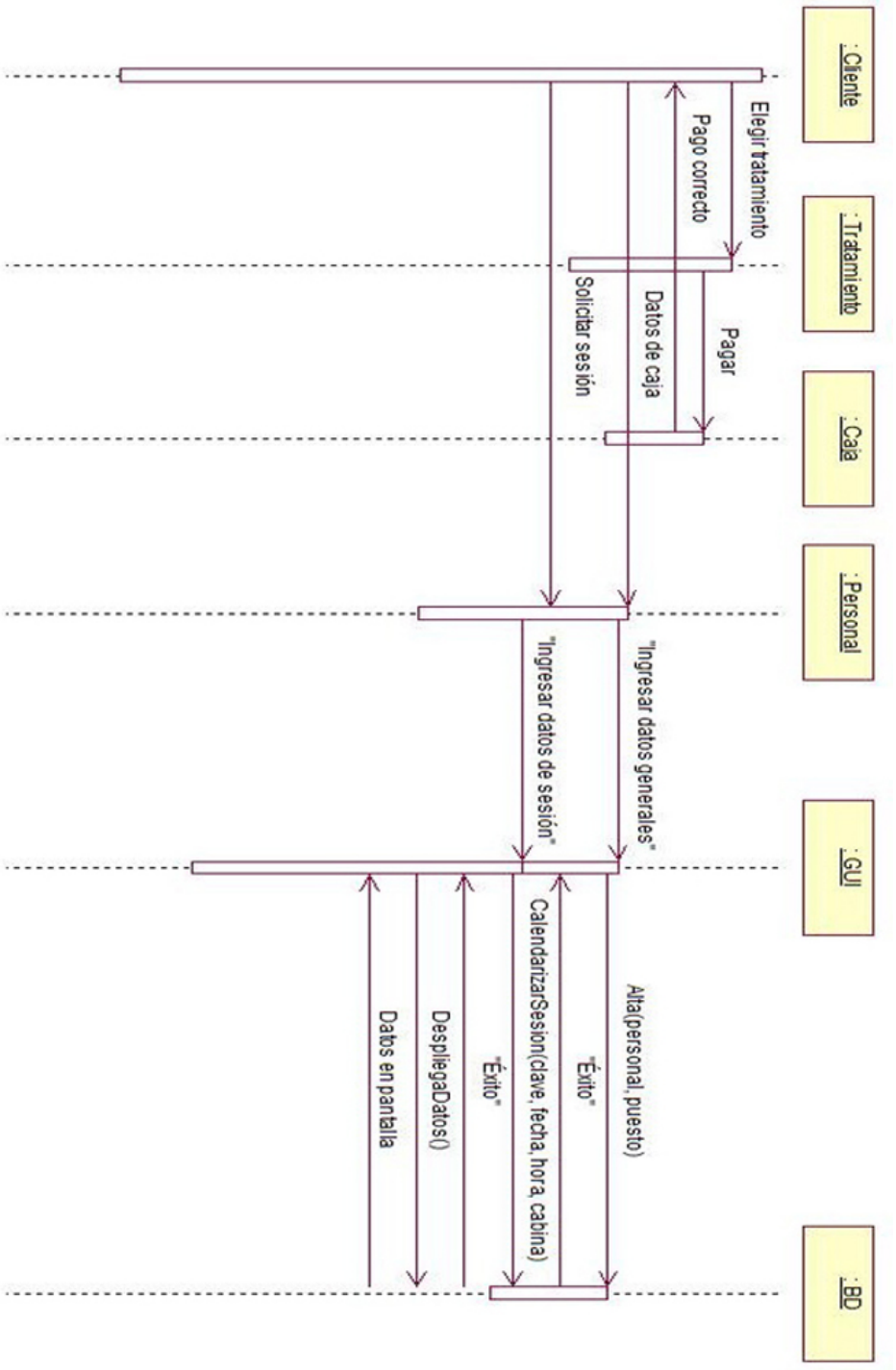


Figura 3.28: Diagrama de secuencia del escenario *Inscripción*.

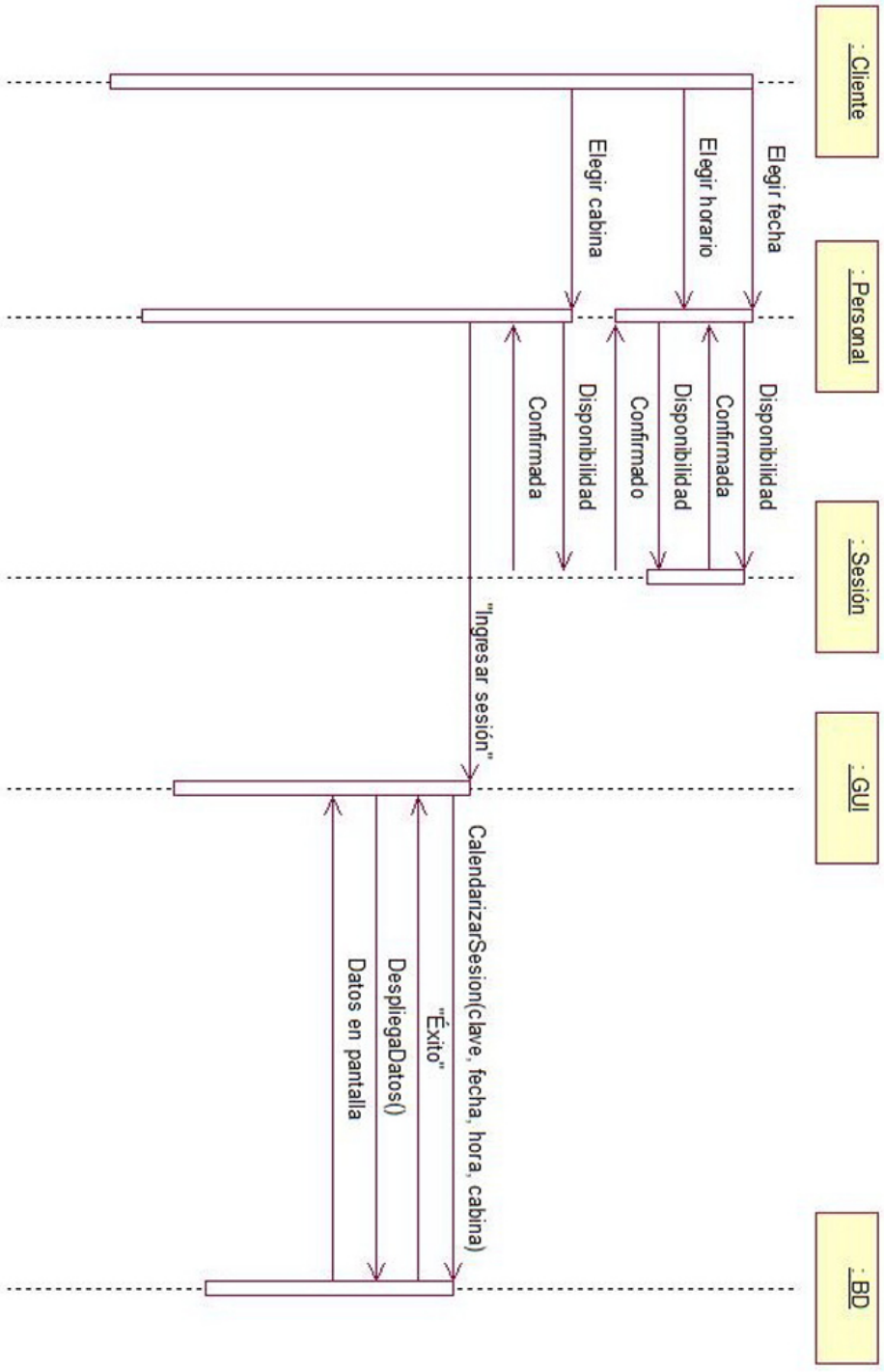


Figura 3.29: Diagrama de secuencia del escenario *Sesiones: Calendarizar sesiones.*

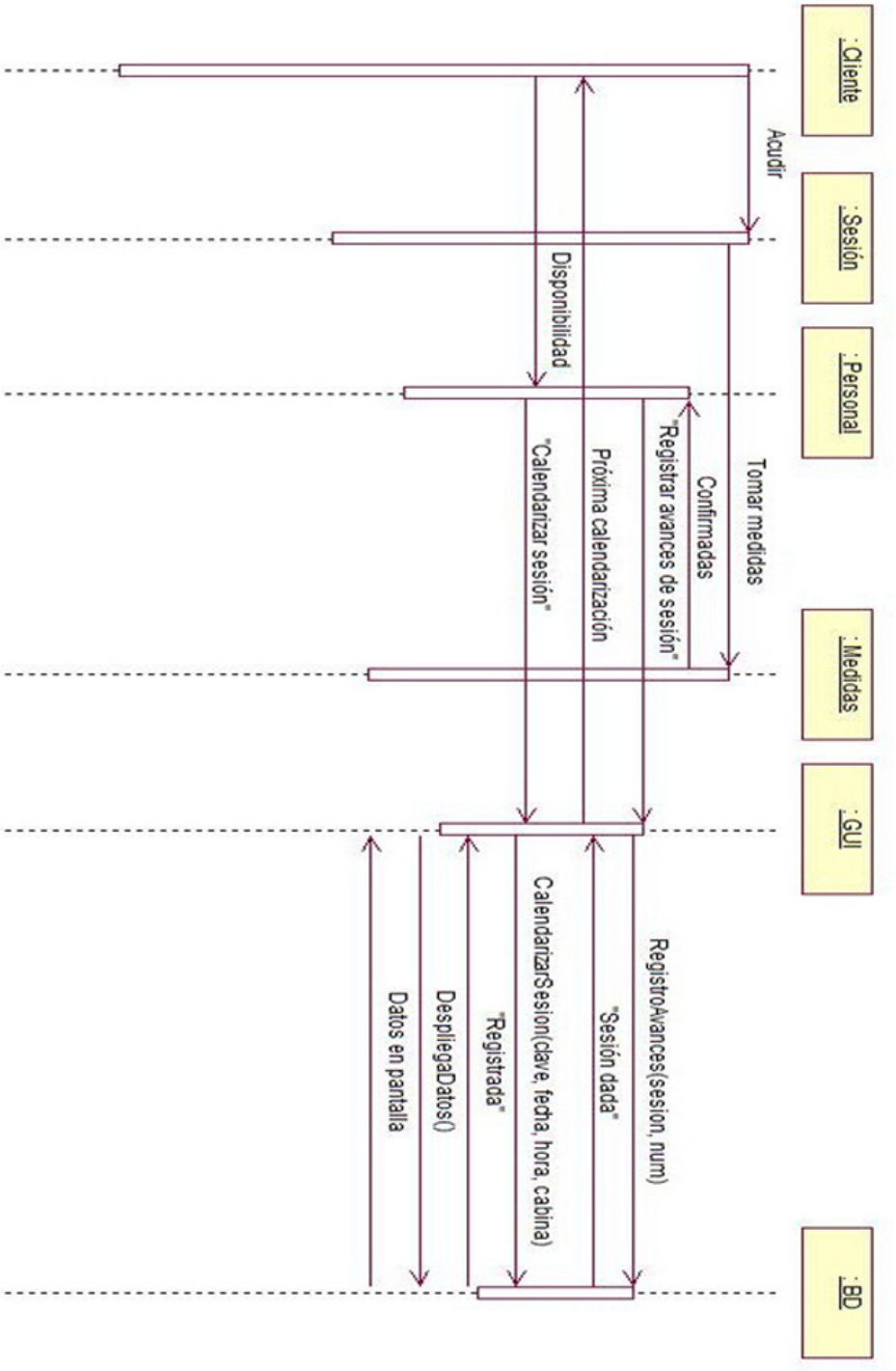


Figura 3.30: Diagrama de secuencia del escenario *Sesiones: Acudir a sesiones.*

Capítulo

4

Construcción.

Después de realizar el análisis y diseño correspondientes, se procede a construir las interfaces correspondientes, incluyendo los diagramas de componentes y de despliegue de *Rational Rose*, los cuales se presentan en este capítulo.

Cabe mencionar que RUP establece correspondencias desde un modelo UML a un lenguaje de programación, como por ejemplo Java, Visual Basic o C++ [BOG99]; al poseer PHP gran similitud con C++, se puede asegurar una afinidad de correspondencia en su construcción mediante este enfoque.

4.1. Diagrama de componentes.

La estructura del sistema de información a desarrollar queda plasmada en la fig. 4.2.

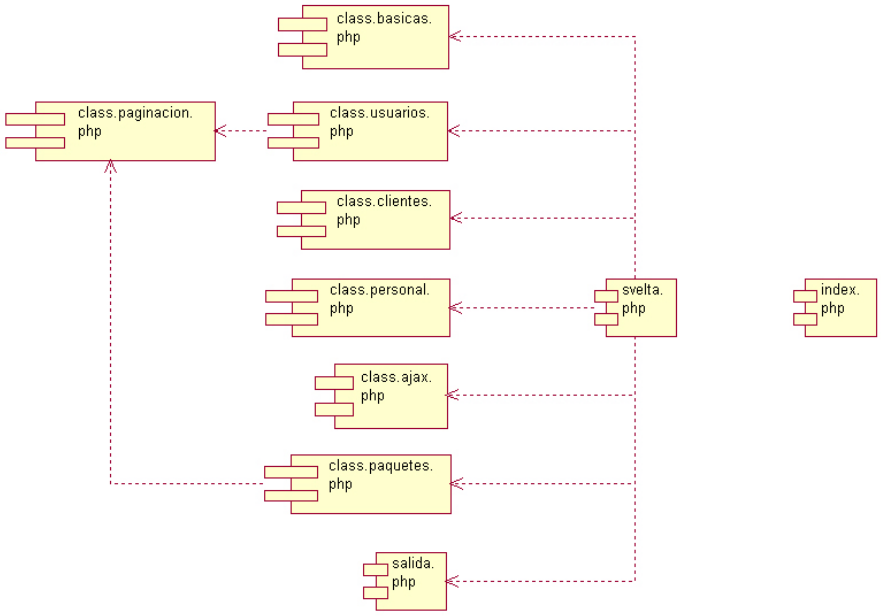


Figura 4.1: Diagrama de despliegue de *Svelta*.

Como se aprecia, los dos componentes base son **index.php** y **svelta.php**; éste último es el *contenedor* de las *clases* para lograr mayor encapsulación del código y hacer el sistema más seguro y fácil de manejar. **index.php** corresponde a las secciones del sitio web en general, abarcando las áreas de *¿Quiénes somos?*, *Sucursales* y *Login*, mientras que **svelta.php** es el sistema propiamente.

Cada uno de los componentes contenidos en *svelta.php* llevan el nombre del subsistema al que pertenecen (*Clientes*, *Personal* y *Paquetes*), además de incluir un *script* de salida de sistema de manera extrena a alguna de las clases, debido a que, por condiciones de destrucción de sesión, en un momento dado pueden causar conflicto e invalidar funciones del código generado en PHP.

Las clases **class.usuarios.php** y **class.paquetes.php** contienen a la clase **class.paginacion.php**, cuya única función es realizar, como su nombre lo indica, la paginación de los registros dentro de algunas funciones de las clases mencionadas.

4.2. Diagrama de despliegue.

El esquema general de la implementación del sistema, es decir, cómo sería la distribución de los elementos necesarios para lograr el funcionamiento del sistema, se contempla en la figura 4.2. Cabe mencionar que este diagrama contempla ya la posibilidad de aparición de sucursales.

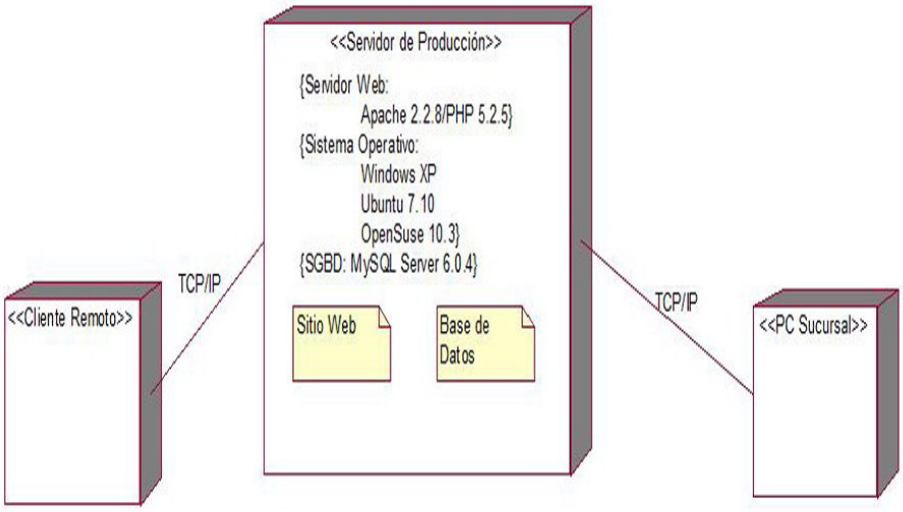


Figura 4.2: Diagrama de despliegue de *Svelta*.

4.3. Interfaces gráficas.

A continuación, se presentan, de manera general, las interfaces de los subsistemas, siguiendo el diagrama de componentes planteado anteriormente. Para conocer el código fuente construido, ver *Apéndice D*.

4.3.1. Login.

Esta interfaz representa la entrada al sistema mediante el uso de un número de cuenta o de usuario y contraseña de usuario que, al ser introducidos por el personal, se comparan con los registros de la base de datos, y así aceptan o niegan la entrada al sistema (ver fig. 4.3). Los campos de texto están validados mediante código de Javascript, mediante el cual impide, en el caso del campo de número de cuenta, ingresar letras o caracteres especiales, mientras que, en el campo de NIP, acepta todo menos acentos.

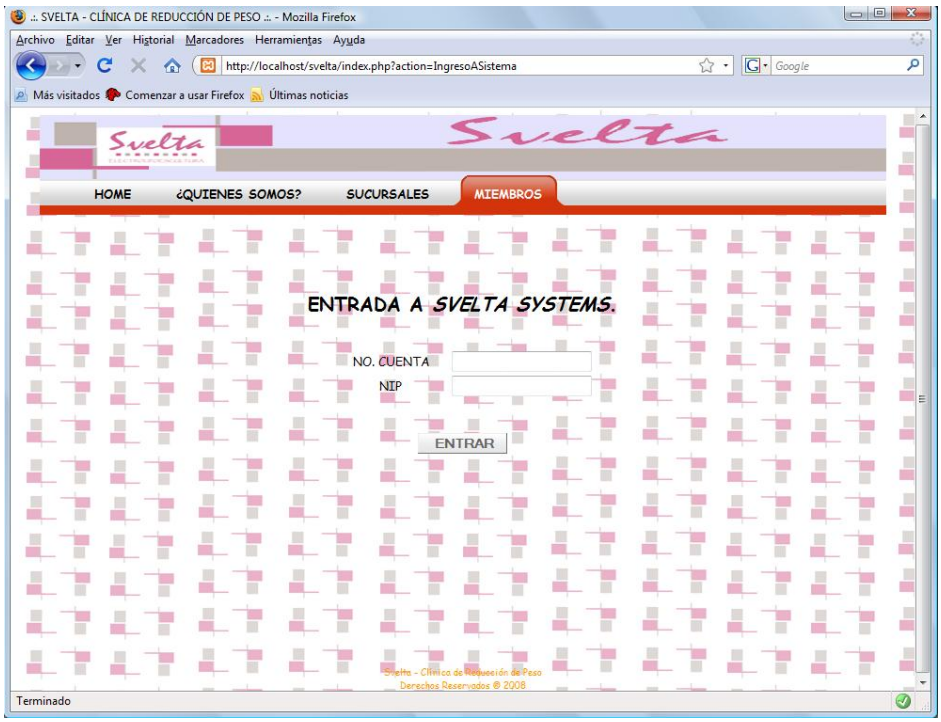


Figura 4.3: Interfaz de ingreso al sistema.

4.3.2. Principal.

En esta interfaz se muestran algunos datos del usuario que ingresa al sistema, tales como nombre completo, la fecha del día presente, la sucursal a la que pertenece el miembro del personal, qué tipo de usuario es (privilegios dados por los puestos de la organización), un reloj digital, basado en un archivo Javascript, y la orden de sesiones del día en curso, las cuales se pueden cancelar desde dicha interfaz (ver fig. 4.4).

Dependiendo del nivel de privilegios que posee el usuario, será el menú que se desplegará; es decir, si el usuario posee privilegios de *Administrador* o *Gerente*, el sistema le permite entrar a las opciones de Paquetes, debido a que, en esta interfaz se permite modificar precios o dar de alta o baja nuevos paquetes de tratamientos, situación que únicamente le corresponde a los niveles directivos de la empresa, no a los operativos (*Terapeuta*).

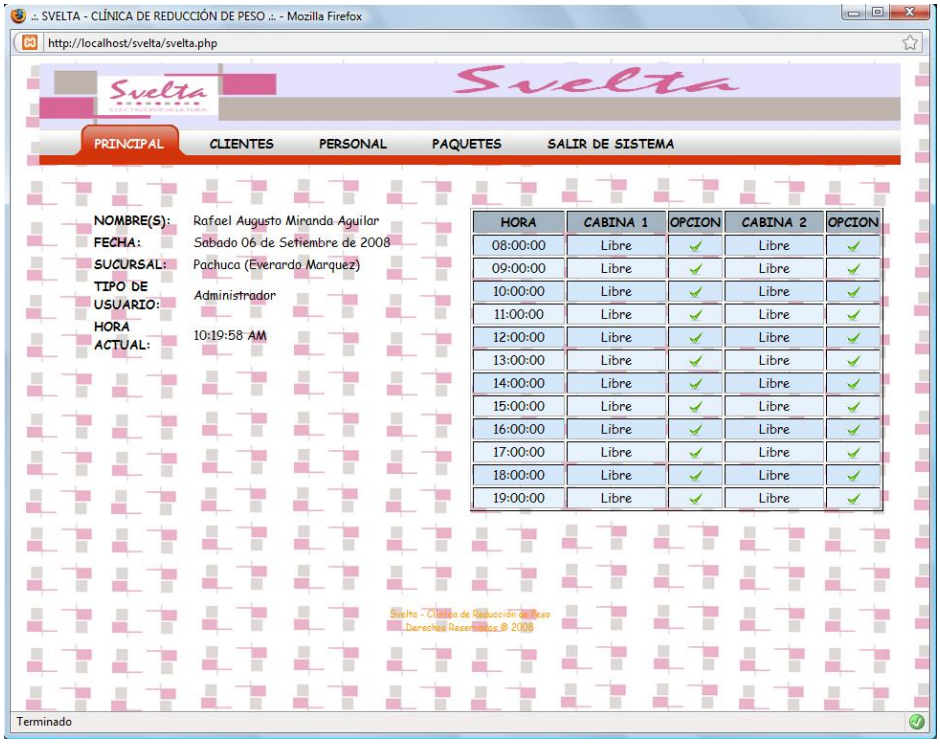


Figura 4.4: Interfaz principal del sistema.

4.3.3. Clientes.

La interfaz de este subsistema despliega en tabla los datos generales de los clientes de *Svelta*, tales como número de cuenta, nombre completo, así como un vínculo para acceder al expediente completo de cada uno de ellos; permite altas, bajas, actualización de datos y/o de contraseña, adquisición de tratamientos, calendarización de sesiones y registro de avances por sesión de clientes (ver fig. 4.5).

Los usuarios *Administrador* y *Gerente* poseen privilegio para ver los clientes de todas las sucursales y para dar de baja a clientes, mientras que el resto de los miembros del personal, sólo podrán tener acceso a los clientes de la sucursal para la que laboran. El subsistema cuenta con un motor de búsqueda orientado a clientes que permite consultas precisas o con multiplicidad de resultados.

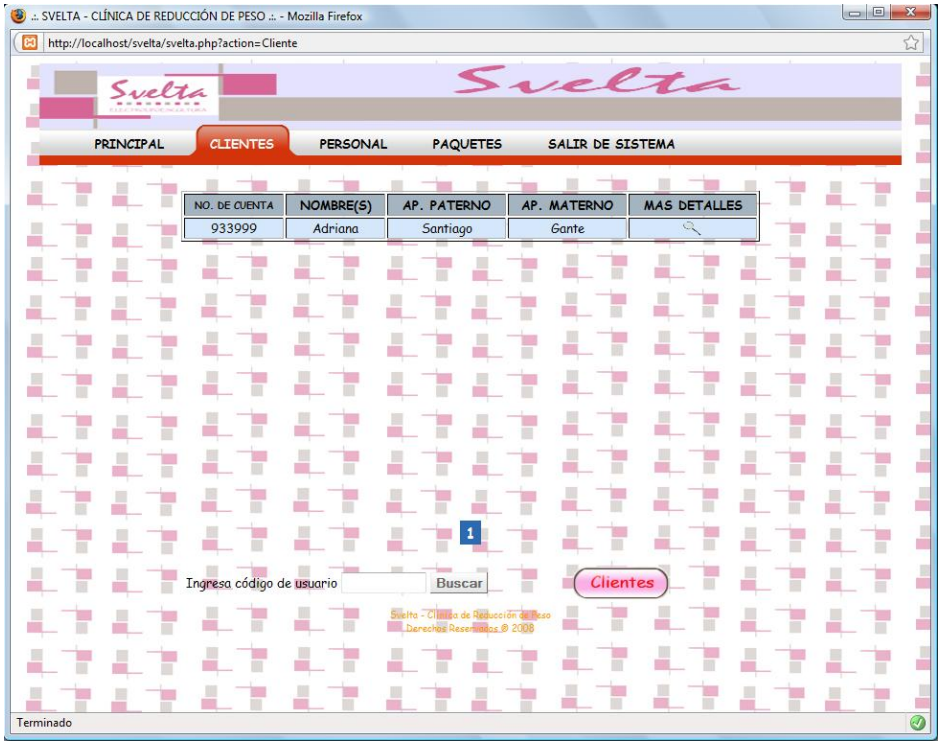


Figura 4.5: Interfaz de subsistema *Clientes*.

4.3.4. Personal.

Este subsistema despliega mediante una tabulación los datos generales del personal de *Svelta*, tales como número de cuenta, nombre completo y el vínculo para datos completos y operaciones exclusivas del personal; permite altas, bajas, actualización de datos y/o de contraseña de empleados (ver fig. 4.6).

El *Administrador* y *Gerente* tienen el privilegio de alterar el puesto de cada miembro del personal, ver a todos los empleados de cualquier sucursal y dar de baja a un miembro del personal. El resto del personal podrá tener acceso a los datos del personal que labore en la misma sucursal, mas no podrá alterarlos, exceptuando los suyos. El subsistema, al igual que el anterior, posee un motor de búsqueda orientado al personal que permite consultas precisas o con multiplicidad de resultados.

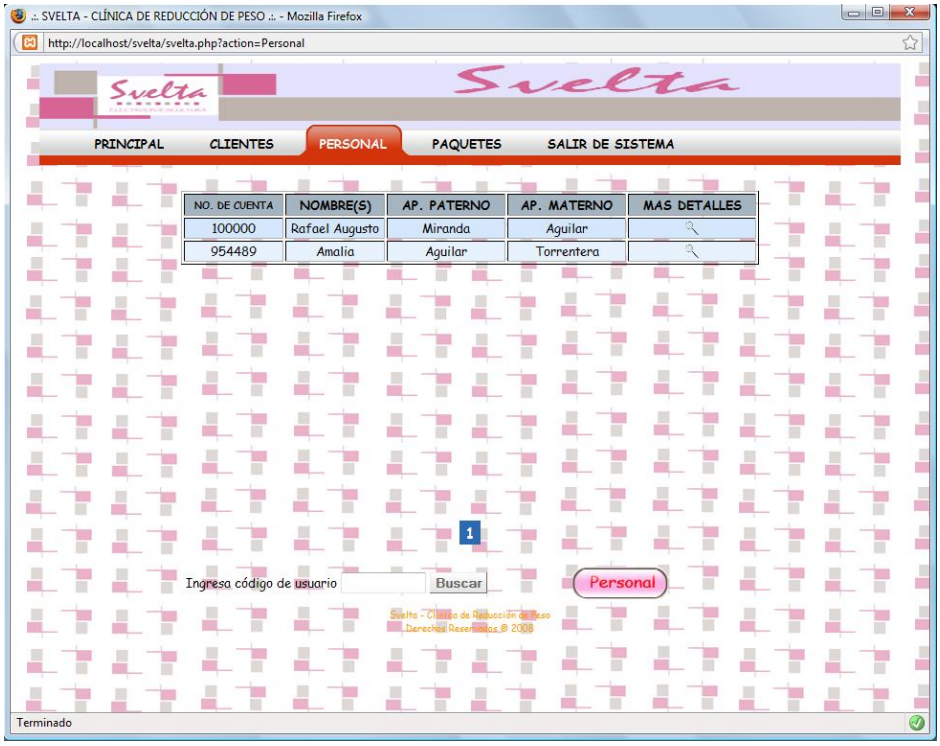


Figura 4.6: Interfaz de subsistema *Personal*.

4.3.5. Paquetes.

Esta interfaz, que corresponde al tercer y último subsistema de *Svelta*, despliega los datos generales de los paquetes que ofrece *Svelta*, tales como el nombre del paquete, el costo del mismo, un vínculo para cambiar el estado del paquete (*Activo* o *No activo*) y otro vínculo para poder actualizar los datos; permite altas, cambios de estado y actualización de datos de paquetes (ver fig. 4.7).

Los usuarios *Administrador* y *Gerente* son los únicos que tienen acceso a esta área del sistema. Debido a que no es un número grande de paquetes los que maneja la organización, este subsistema no posee un motor de búsqueda de tratamientos; sin embargo, sí esta paginado, lo que permite ver la totalidad de paquetes disponibles.

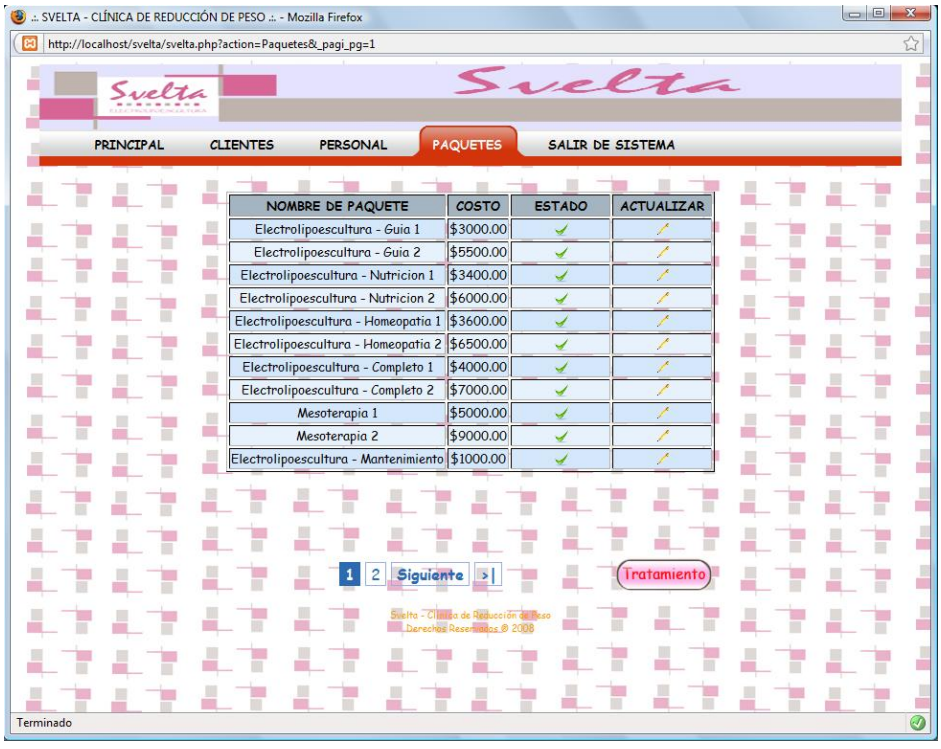


Figura 4.7: Interfaz de subsistema *Paquetes*.

4.4. Pruebas al sistema.

En esta sección se presentan los resultados obtenidos para las pruebas hechas al sistema, con el fin de conocer su evolución y las fallas detectadas, así como la versión final del sistema. Para este capítulo, se han elegido como casos de prueba los casos de uso de negocios planteados en el Capítulo 3, exceptuando el caso de uso *Acudir a sesión*, ya que no involucra una participación del sistema en él.

4.4.1. Caso 1: Validar usuario.

4.4.1.1. Descripción.

Esta documentación cubre un conjunto de pruebas realizadas sobre el Caso de Uso del mismo nombre.

Se realizaron dos pruebas que consisten en el ingreso al sistema por parte de un administrador y un terapeuta, debido a que es en estos casos donde las pantallas de inicio del sistema se alteran de forma considerable.

El entorno del cual se parte para realizar la prueba es el *formulario de entrada* al sistema.

4.4.1.2. Validar usuario (Administrador).

4.4.1.3. Descripción.

Se ingresa en el sistema como administrador y se despliegan las opciones que corresponden a su nivel de acceso. El sistema muestra en pantalla los datos del usuario que ingresó, confirmando su puesto, y la orden de sesiones del día, así como el menú de control completo de las *áreas* del negocio.

4.4.1.4. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario administrador *100000* y su contraseña correspondiente están dados de alta en la base de datos.

4.4.1.5. Entrada.

1. Se introduce **100000** en el campo *Número de cuenta*.
2. Se introduce **adone** en el campo *NIP* (ver fig. 4.8).
3. Se pulsa el botón *Entrar* de la aplicación.
4. Aparece la interfaz propia de un administrador (ver fig. 4.9).
5. El administrador pulsa el botón *Salir* para regresar a la página principal de *Svelta*.

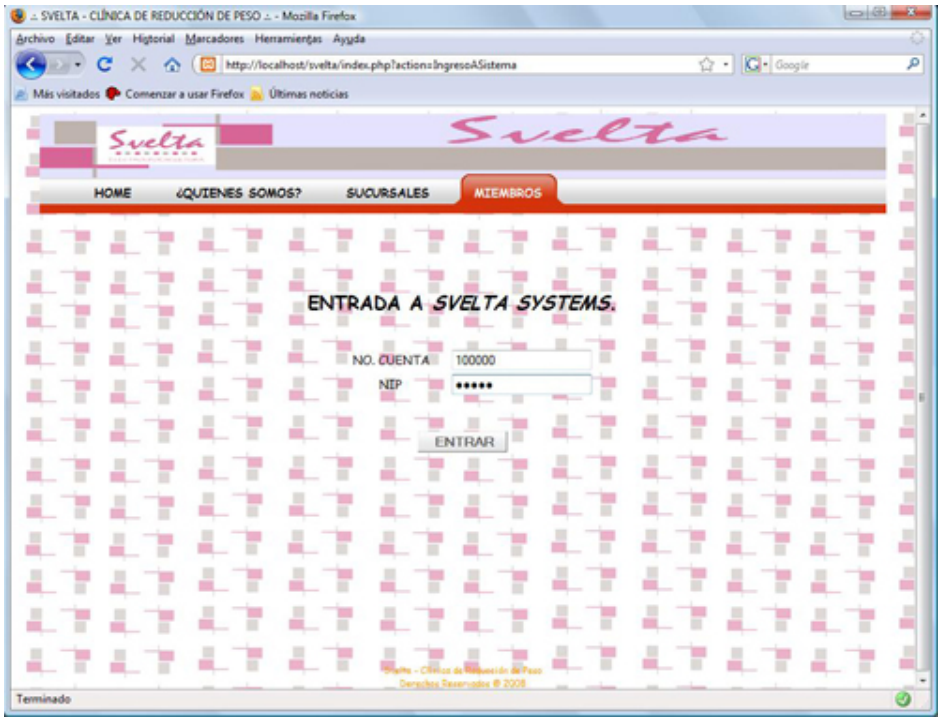


Figura 4.8: Entrada de Administrador.

4.4.1.6. Resultado esperado.

La interfaz muestra los datos del usuario administrador, acceso a los menús de todas las áreas del sistema y la orden de sesiones del día.

4.4.1.7. Evaluación de la prueba.

Prueba superada con éxito.

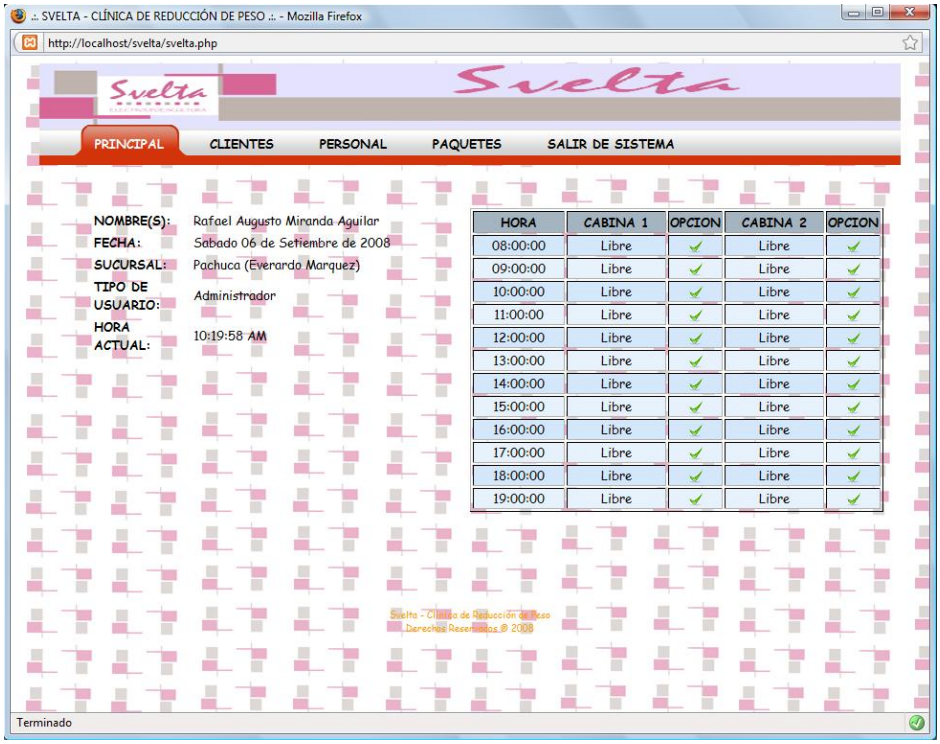


Figura 4.9: Interfaz principal de Administrador.

4.4.1.8. Validar usuario (Terapeuta).

4.4.1.9. Descripción.

Se ingresa en el sistema como terapeuta y se despliegan las opciones que corresponden a su nivel de acceso. El sistema muestra en pantalla los datos del usuario que ingresó, confirmando su puesto, y la orden de sesiones del día, así como el menú de control completo de las *áreas* del negocio.

4.4.1.10. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario y su contraseña correspondiente están dados de alta en la base de datos.

4.4.1.11. Entrada.

1. Se introduce **954489** en el campo *Número de cuenta*.
2. Se introduce **amalia** en el campo *NIP* (ver fig. 4.10).
3. Se pulsa el botón *Entrar* de la aplicación.
4. Aparece la interfaz propia de un terapeuta (ver fig. 4.11).
5. El terapeuta pulsa el botón *Salir* para regresar a la página principal de *Svelta*.

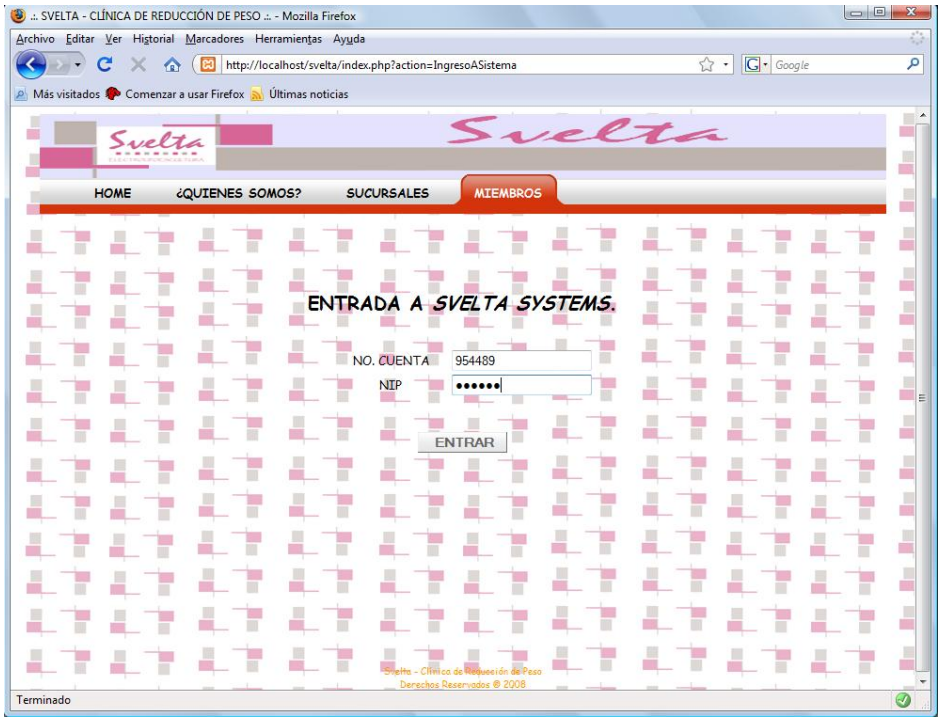


Figura 4.10: Entrada de Terapeuta.

4.4.1.12. Resultado esperado.

La interfaz muestra los datos del usuario terapeuta, a los menús de todas las áreas del sistema, con excepción del subsistema *Paquetes*, y actualización de datos personales, y la orden de sesiones del día.

4.4.1.13. Evaluación de la prueba.

Prueba superada con éxito.

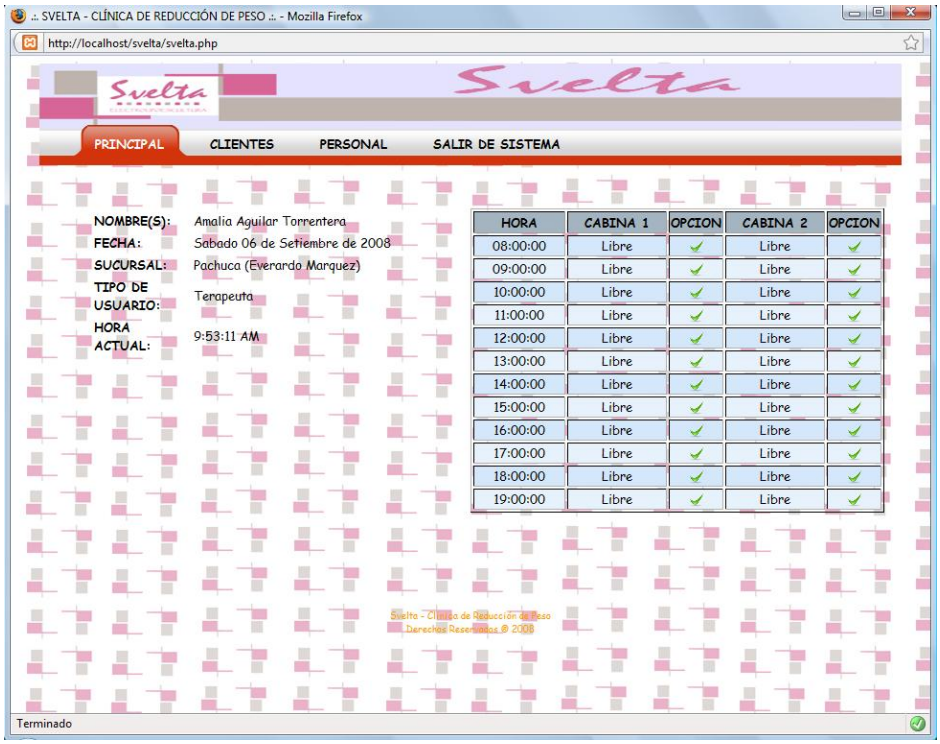


Figura 4.11: Interfaz principal de Terapeuta.

4.4.2. Caso 2: Ingresar usuario.

4.4.2.1. Descripción.

Esta documentación cubre un conjunto de pruebas realizadas sobre el Caso de Uso del mismo nombre.

Se realizaron dos pruebas que consisten en el ingreso de un cliente y de un nuevo miembro del personal.

Los entornos de los cuales se parte para realizar las pruebas son el subsistema *Cliente*, en la opción *Ingresar cliente*, y el subsistema *Personal*, en la opción *Ingresar personal*.

4.4.2.2. Ingresar usuario (Cliente).

4.4.2.3. Descripción.

Se da de alta en el sistema un nuevo cliente, y se confirma su alta en el momento en que la pantalla principal del subsistema *Clientes* se actualice con los datos principales del cliente ingresado.

4.4.2.4. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario administrador 100000 y su contraseña correspondiente estén dados de alta en la base de datos, y que el administrador haya ingresado previamente al sistema.

4.4.2.5. Entrada.

1. El administrador pulsa el botón *Clientes* para ingresar al subsistema del mismo nombre.

2. El administrador pulsa el botón *Ingresar cliente* para entrar en la interfaz correspondiente.

3. El administrador introduce la información solicitada en el formulario de ingreso del cliente (ver fig. 4.12). Cabe destacar que el número de cuenta es creado de forma automática, para evitar duplicidad de éstos. La información ingresada es:

   - **Nombre(s)**: Adriana.
   - **Apellido Paterno**: Santiago.
   - **Apellido Materno**: Gante.
   - **Dirección**: Blvd. de San Angel #203. San Javier.

- **Teléfono (Casa)**: 7152389.
- **Teléfono (Móvil)**: 7711428609.
- **Contraseña**: ad341988 (como campo *password*).
- **Confirmar contraseña**: ad341988 (ídem).
- **Talla**: 1.65.

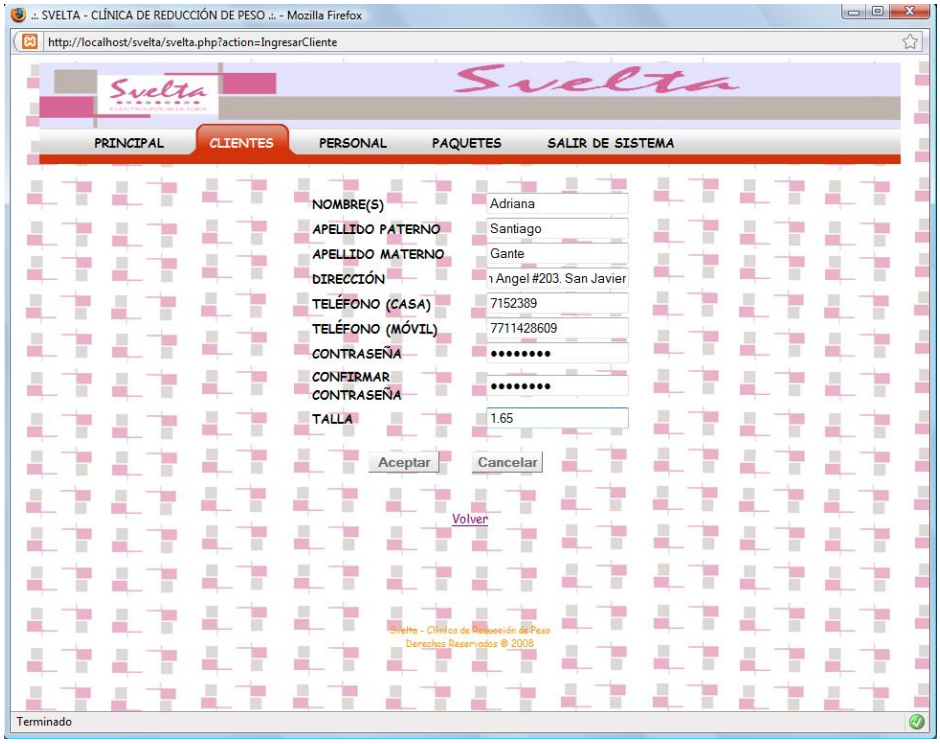


Figura 4.12: Ingreso de datos de *Adriana Santiago Gante*.

4. Se pulsa el botón *Aceptar* de la aplicación para cargar los datos en la base de datos.

5. Aparece el listado de clientes con los datos básicos (ID, Nombre(s), Apellido Paterno, Apellido Materno y Ver Completo) de Adriana Santos Gante (ver fig. 4.13).

4.4.2.6. Resultado esperado.

La pantalla del subsistema *Clientes* queda actualizada, mostrando el listado de clientes con el registro recién ingresado.

4.4.2.7. Evaluación de la prueba.

Prueba superada con éxito.

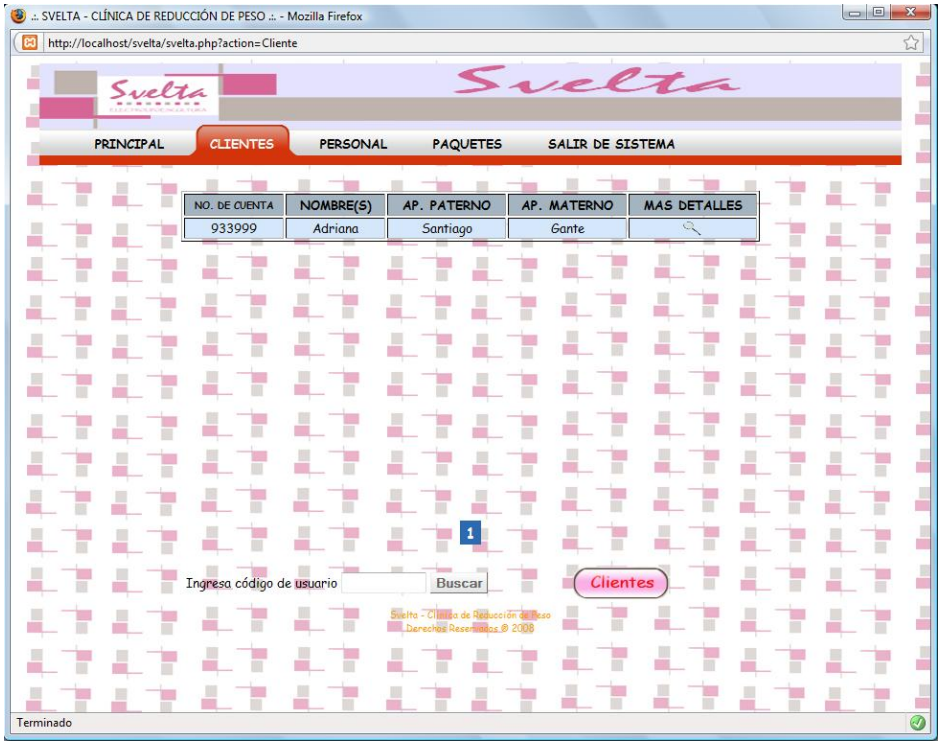


Figura 4.13: Resultado de ingreso de cliente.

4.4.2.8. Ingresar usuario (Personal).

4.4.2.9. Descripción.

Se da de alta en el sistema un nuevo miembro del personal, y se confirma su alta en el momento en que la pantalla principal del subsistema *Personal* se actualice con los datos principales del miembro del personal ingresado.

4.4.2.10. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario administrador 100000 y su contraseña correspondiente estén dados de alta en la base de datos, y que el administrador haya ingresado previamente al sistema.

4.4.2.11. Entrada.

1. El administrador pulsa el botón *Personal* para ingresar al subsistema del mismo nombre.

2. El administrador pulsa el botón *Ingresar personal* para entrar en la interfaz correspondiente.

3. El administrador introduce la información solicitada en el formulario de ingreso del miembro del personal (ver fig. 4.14). Cabe destacar que el número de cuenta es creado de forma automática, para evitar duplicidad de éstos. La información ingresada es:

   - **Nombre(s)**: Rafael.
   - **Apellido Paterno**: Ordoñez.
   - **Apellido Materno**: Trejo.
   - **Dirección**: Clzda. Villa Nueva #12. Periodistas.
   - **Teléfono (Casa)**: 2128938.
   - **Teléfono (Móvil)**: 7711220988.
   - **Contraseña**: rafael (como campo *password*).
   - **Confirmar contraseña**: rafael (ídem).
   - **Puesto**: Administrador.

4. Se pulsa el botón *Aceptar* de la aplicación para cargar los datos en la base de datos.

5. Aparece el listado de clientes con los datos básicos (ID, Nombre(s), Apellido Paterno, Apellido Materno y Ver Completo) de Rafael Ordoñez Trejo (ver fig. 4.15).

4.4.2.12. Resultado esperado.

La pantalla del subsistema *Personal* queda actualizada, mostrando el listado de clientes con el registro recién ingresado.

4.4.2.13. Evaluación de la prueba.

Prueba superada con éxito.

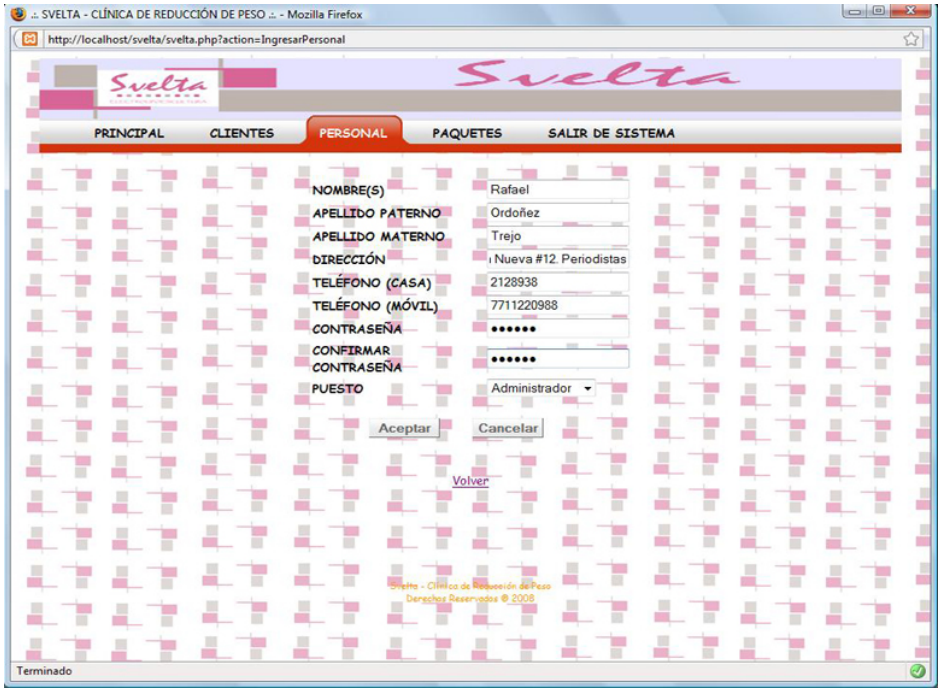


Figura 4.14: Ingreso de datos de *Rafael Ordoñez Trejo*.

4.4.3. Caso 3: Consultar expedientes.

4.4.3.1. Descripción.

Esta documentación cubre un conjunto de pruebas realizadas sobre el Caso de Uso del mismo nombre.

Se realizó una prueba que consistió en la consulta del historial de

Adriana Santiago Gante, empleando el motor de búsqueda del subsistema *Clientes*. El funcionamiento es el mismo para el subsistema *Personal*.

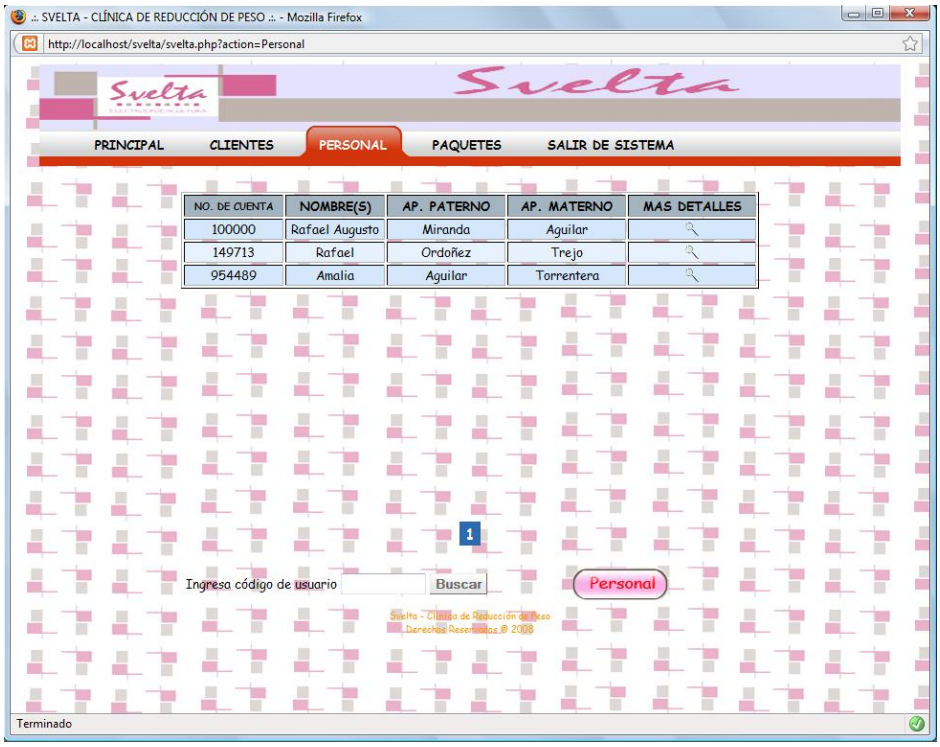


Figura 4.15: Resultado de ingreso de personal.

El entorno del cual se parte para realizar la prueba es la interfaz principal del subsistema *Clientes*.

4.4.3.2. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario administrador 100000 y su contraseña correspondiente estén dados de alta en la base de datos, y que esté actualmente dentro del subsistema *Clientes*.

4.4.3.3. Entrada.

1. El administrador visualiza el listado de clientes en el subsistema del mismo nombre.

2. El administrador escribe en el campo de búsqueda el número (indicar parte del número de ID de Adriana Santiago Gante) y pulsa el botón *Buscar*.

3. El subsistema se actualiza de manera casi inmediata, permitiendo reducir el número de registros; entre ellos, se localiza a Adriana Santiago Gante. Se pulsa el botón de la columna *Ver Completo* de ella para ingresar a su historial y al menú de operaciones del cliente.

4. Aparece el historial de Adriana Santos Gante con el conjunto de opciones que puede realizar.

4.4.3.4. Resultado esperado.

La pantalla, al emplear el motor de búsqueda, reduce su número de registros (en caso de no saber el ID completo del cliente en cuestión), facilitando su búsqueda. La pantalla del historial del cliente muestra sus datos, validando que existe dentro del sistema.

4.4.3.5. Evaluación de la prueba.

Prueba superada con éxito.

4.4.4. Caso 4: Actualizar expedientes.

4.4.4.1. Descripción.

Esta documentación cubre un conjunto de pruebas realizadas sobre el Caso de Uso del mismo nombre.

Se realizaron dos pruebas que consistieron en la actualización de datos de Adriana Santiago Gante, y la actualización de la contraseña de Amalia Aguilar Torrentera.

El entorno del cual se parte para realizar la prueba es el historial de cada uno de los usuarios en cuestión, para acceder a las operaciones de cada uno.

4.4.4.2. Actualizar datos de usuario (Cliente).

4.4.4.3. Descripción.

Se actualizan los datos deseados de Adriana Santiago Gante y se confirma su actualización con la visualización de la interfaz que muestra su expediente.

4.4.4.4. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario administrador 100000 y su contraseña correspondiente estén dados de alta en la base de datos, y que esté actualmente dentro del sistema.

4.4.4.5. Entrada.

1. El administrador pulsa el botón *Actualizar datos* en el historial de Adriana Santiago Gante.

2. El administrador introduce la información a actualizar en el formulario de actualización de datos del usuario (ver fig. 4.16.a). Cabe destacar que los valores en los campos de texto por defecto son llamados desde la base de datos para evitar el tener que escribir todos los datos nuevamente. La información a actualizar es:

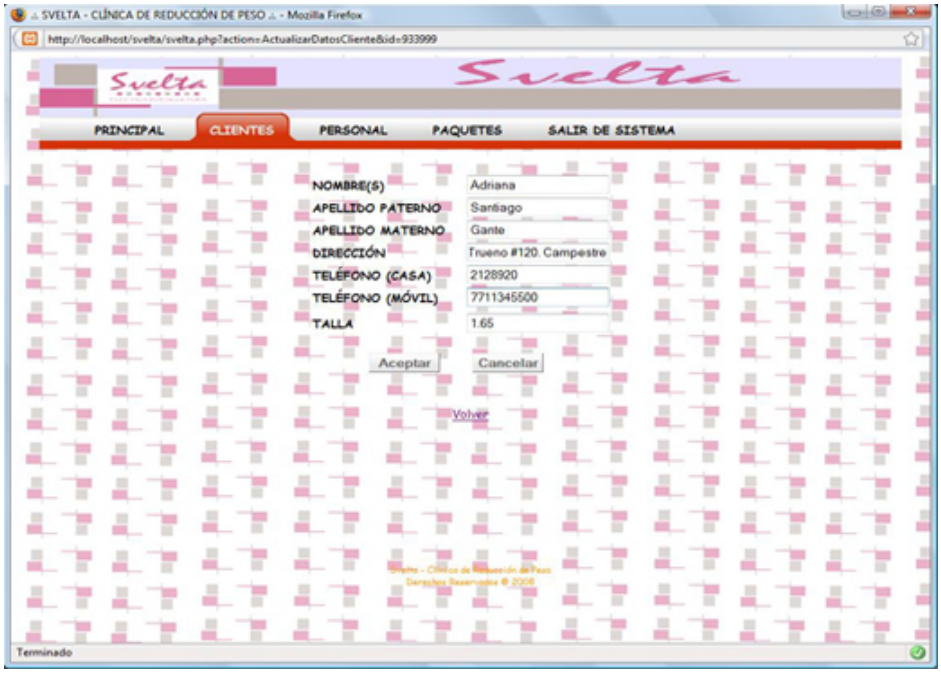
   - **Dirección**: Calle del Trueno #120. Campestre (reemplaza a Blvd. de San Angel #203. San Javier).
   - **Teléfono (Casa)**: 2128920 (reemplaza a 7152389).
   - **Teléfono (Móvil)**: 7711345500 (reemplaza a 7711428609).

3. Se pulsa el botón *Aceptar* de la aplicación para cargar los datos actualizados en la base de datos.

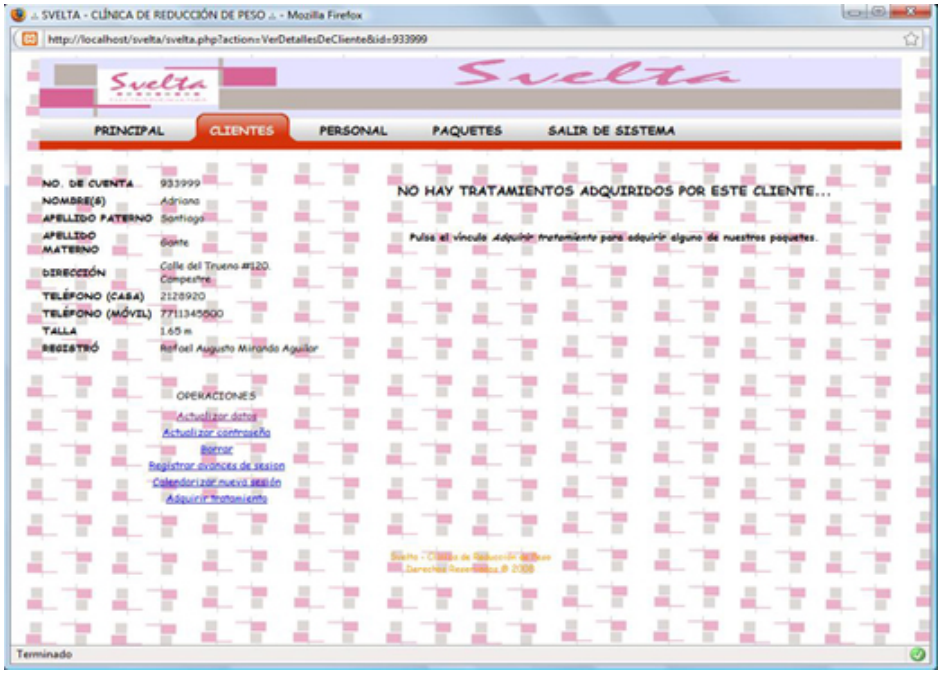
4. La pantalla del historial de Adriana Santiago Gante muestra los cambios previamente hechos a sus datos personales (ver fig. 4.16.b).

4.4.4.6. Resultado esperado.

La pantalla del historial del cliente muestra sus datos actualizados.



a) Actualización de datos de *Adriana Santiago Gante*.



b) Resultado de actualización de datos.

Figura 4.16: Actualizar datos de cliente.

4.4.4.7. Evaluación de la prueba.

Prueba superada con éxito.

4.4.4.8. Actualizar contraseña de usuario (Personal).

4.4.4.9. Descripción.

Se actualiza la contraseña de Amalia Aguilar Torrentera y se confirma su actualización con la visualización de la interfaz que muestra su expediente, así como un mensaje que confirma la actualización de dicha contraseña.

4.4.4.10. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario terapeuta 954489 y su contraseña correspondiente estén dados de alta en la base de datos, y que esté actualmente dentro del sistema.

4.4.4.11. Entrada.

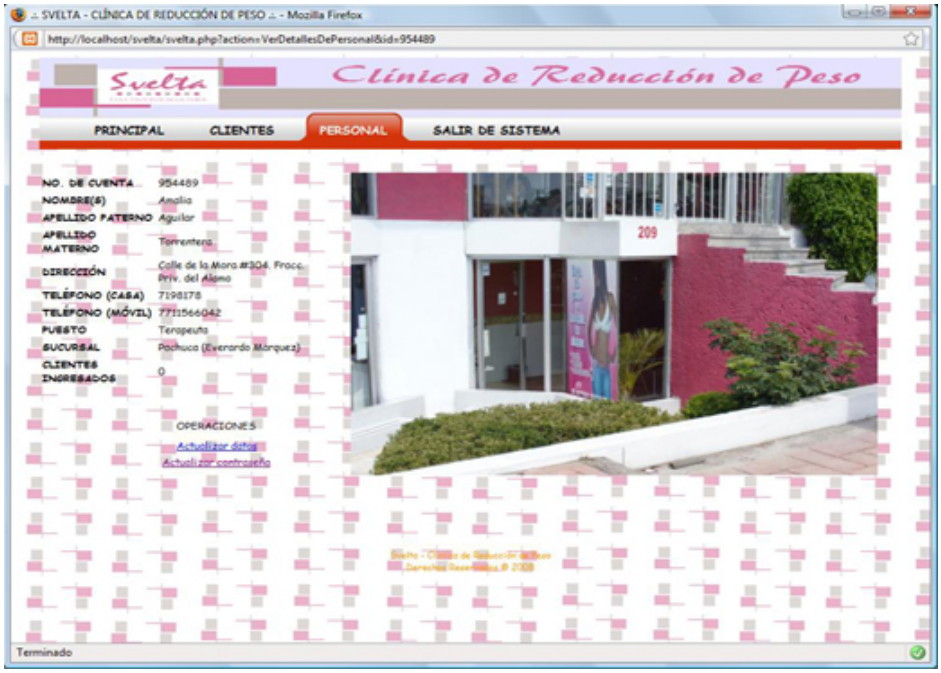
1. El terapeuta pulsa el botón *Actualizar pasword* en su expediente (Amalia Aguilar Torrentera) (ver fig. 4.17.a).

2. El terapeuta introduce la información requerida en el formulario de actualización de contraseña del usuario. Cabe destacar que los valores ingresados en los campos de texto serán de tipo *password*, por lo que la información no podrá verse. La información a ingresar es:

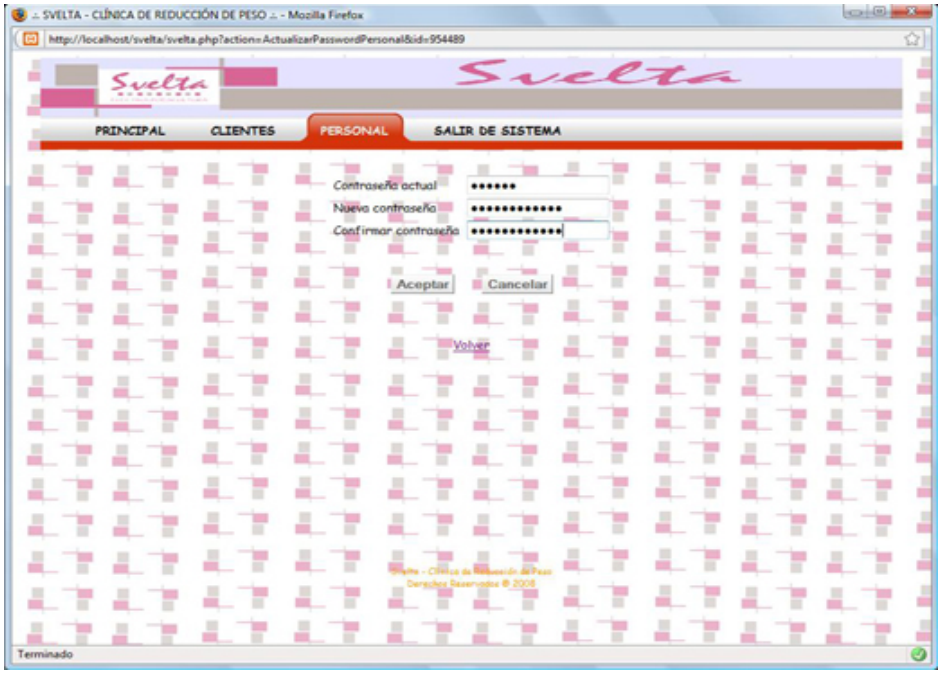
   - **Ingresar pasword**: amalia.
   - **Password nuevo**: amalitamayos.
   - **Confirmar password**: amalitamayos.

3. Se pulsa el botón *Aceptar* de la aplicación para cargar la contraseña actualizada en la base de datos.

4. Se muestra un mensaje confirmando la actualización de la contraseña, y aparece la interfaz del expediente de Amalia Aguilar Torrentera (ver fig. 4.17.b).



a) Expediente de *Amalia Aguilar Torrentera.*



b) Actualización de password de *Amalia Aguilar Torrentera.*

Figura 4.17: Actualizar datos de personal.

4.4.4.12. Resultado esperado.

Al actualizar la contraseña, se despliega un mensaje de confirmación, y se muestra la pantalla del expediente del usuario en cuestión.

4.4.4.13. Evaluación de la prueba.

Prueba superada con éxito.

4.4.5. Caso 5: Adquirir tratamiento.

4.4.5.1. Descripción.

Esta documentación cubre un conjunto de pruebas realizadas sobre el Caso de Uso del mismo nombre.

Se realizó una prueba que consistió en la adquisición de un tratamiento disponible en el negocio para Adriana Santiago Gante.

El entorno del cual se parte para realizar la prueba es la opción *Ver Completo*, en el subsistema *Clientes*, del cliente Adriana Santiago Gante.

4.4.5.2. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario administrador 100000 y su contraseña correspondiente estén dados de alta en la base de datos, y que esté actualmente dentro del sistema.

4.4.5.3. Entrada.

1. En la interfaz del expediente de Adriana Santiago Gante, en el lado derecho, se muestra un mensaje de ausencia de tratamiento adquirido.

2. El administrador pulsa el botón *Adquirir tratamiento* en el historial de Adriana Santiago Gante.

3. El administrador selecciona el tratamiento solicitado por Adriana Santiago Gante (*Electrolipoescultura - Completo 1*), y rellena el resto del formulario del pago (*$2000.00*) y la forma en que lo realizará (*Efectivo*) (ver fig. 4.18).

4. Se pulsa el botón *Aceptar* de la aplicación para cargar la contraseña actualizada en la base de datos.

5. Aparece la interfaz del expediente de Adriana Santiago Gante, con el historial de 20 sesiones por defecto que incluye cada tratamiento del lado derecho (ver fig. 4.19).

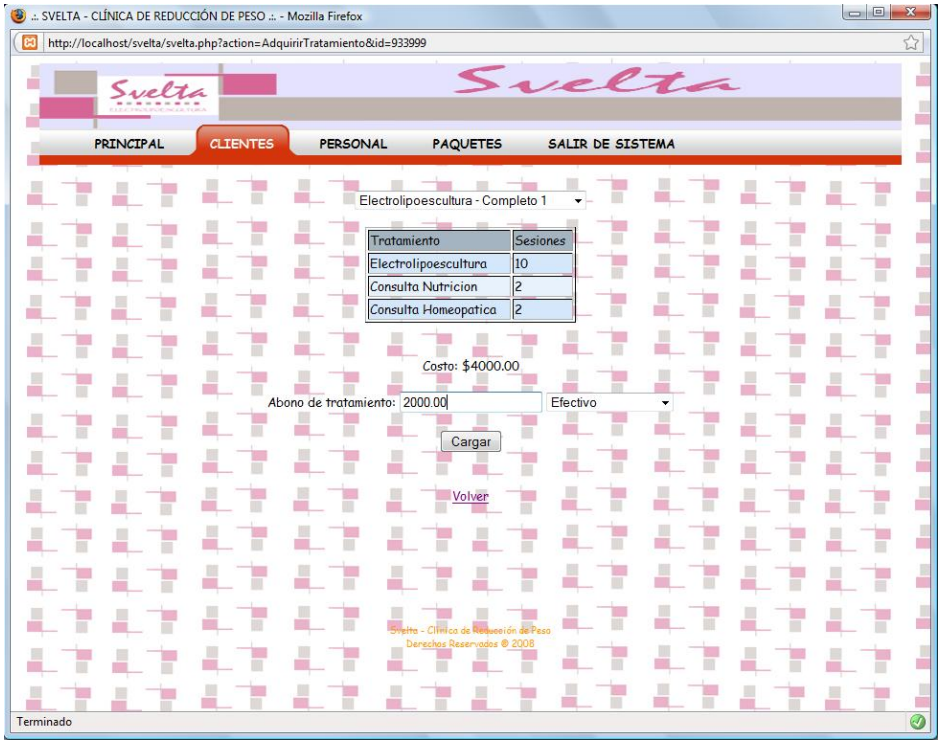


Figura 4.18: Adquisición de tratamiento de *Adriana Santiago Gante*.

4.4.5.4. Resultado esperado.

Aparece en el expediente del cliente una tabla con los datos de sesiones, una vez adquirido el tratamiento.

4.4.5.5. Evaluación de la prueba.

Prueba superada con éxito.

4.4.6. Caso 6: Calendarizar sesión.

4.4.6.1. Descripción.

Esta documentación cubre un conjunto de pruebas realizadas sobre el Caso de Uso del mismo nombre.

Se realizó una prueba que consistió en la calendarización de la sesión primera al tratamiento elegido por Adriana Santiago Gante. El entorno

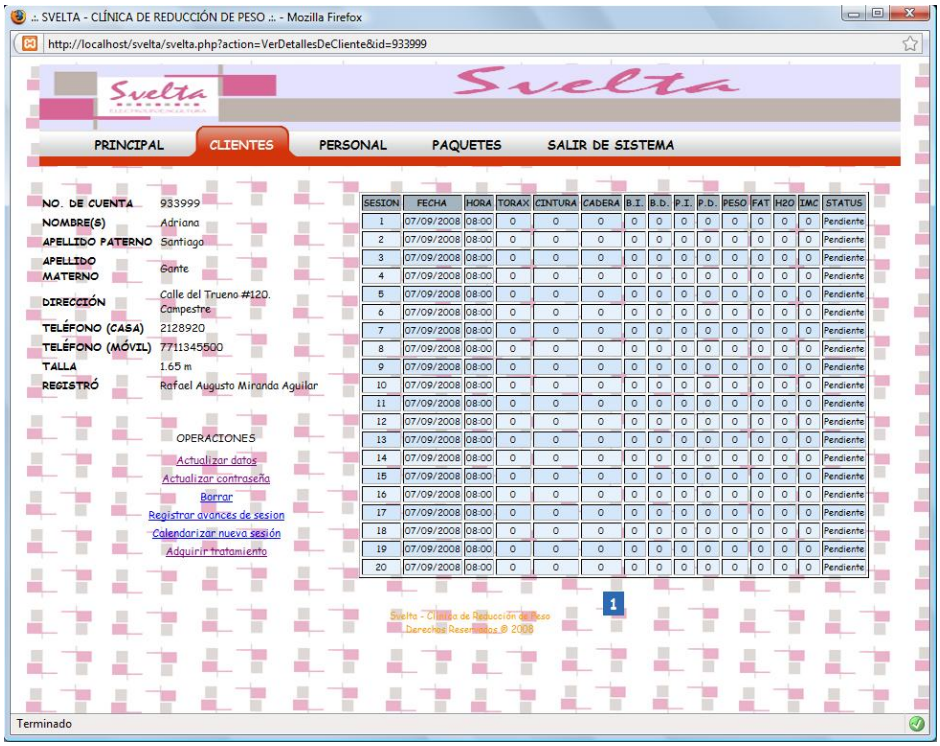


Figura 4.19: Resultado de adquisición de tratamiento.

del cual se parte para realizar la prueba es la opción *Ver Completo*, en el subsistema *Clientes*, del cliente Adriana Santiago Gante.

4.4.6.2. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario administrador 100000 y su contraseña correspondiente estén dados de alta en la

base de datos, y que esté actualmente dentro del subsistema Clientes.

4.4.6.3. Entrada.

1. El administrador pulsa el botón de la columna *Ver Completo* de Adriana Santiago Gante, en el subsistema *Clientes*, para ingresar a su historial y al menú de operaciones (ver fig. 4.20).

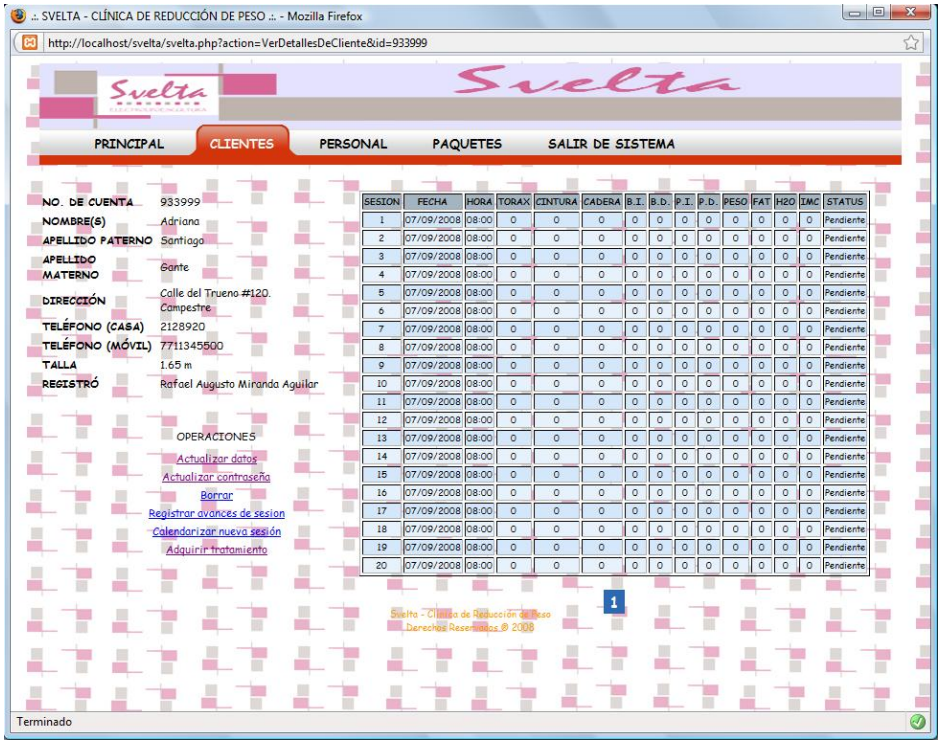


Figura 4.20: Interfaz de historial de *Adriana Santiago Gante*.

2. El administrador pulsa el botón *Calendarizar sesión*, para entrar al formulario de reservación de sesiones (ver fig 4.21.a).

3. El administrador elige una fecha correspondiente del calendario flotante, la cual quedará inscrita en el campo de texto, y el número de la sesión que desea calendarizar. El campo de fecha tiene la fecha del día actual como valor por defecto; se elige la fecha y el número 1 de

la lista desplegable, para dar paso a la aparición a la tabla de disponibilidad de horario de dicha fecha. Posteriormente, se elige una de las cabinas en una hora disponible (denotada por poseer una flecha verde; caso contrario, se muestra una flecha roja).

4. Se pulsa en el enlace de la columna *Cabina 2*, en la hora *17:00:00*.

5. Aparece el historial de Adriana Santos Gante, en cuyo registro de avances se han actualizado los campos de fecha y hora de la sesión, y actualizando el estado de la sesión *Pendiente* a *Calendarizada* (ver fig. 4.21.b).

4.4.6.4. Resultado esperado.

La pantalla del historial del cliente muestra los datos actualizados de la sesión elegida a calendarizar, y actualiza el estado de ésta a *Calendarizada*, dando a entender que se ha reservado satisfactoriamente dicha sesión.

4.4.6.5. Evaluación de la prueba.

Prueba superada con éxito.

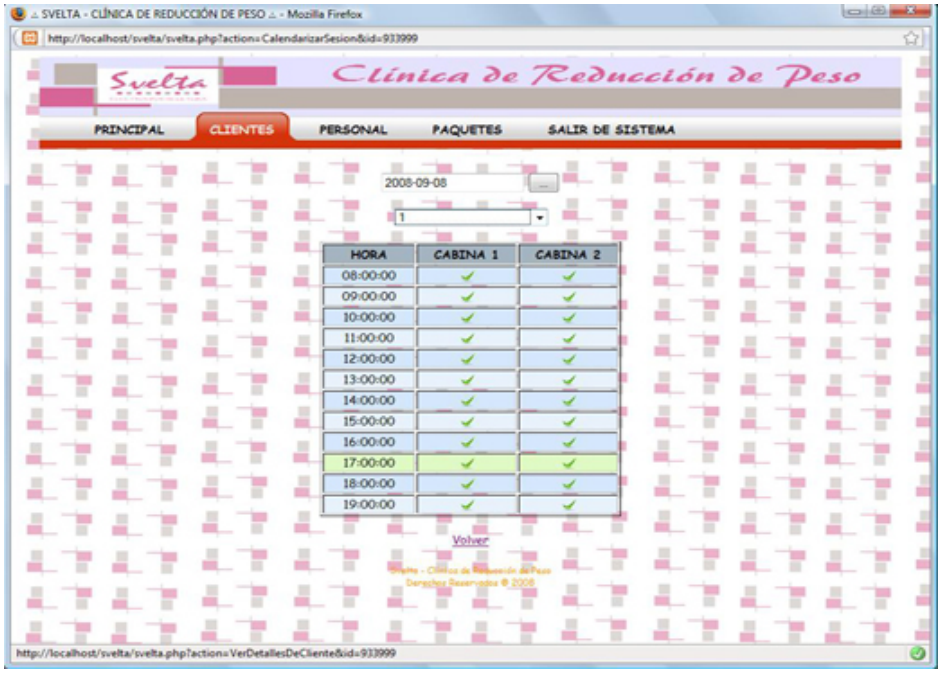
4.4.7. Caso 7: Registrar avances de sesión.

4.4.7.1. Descripción.

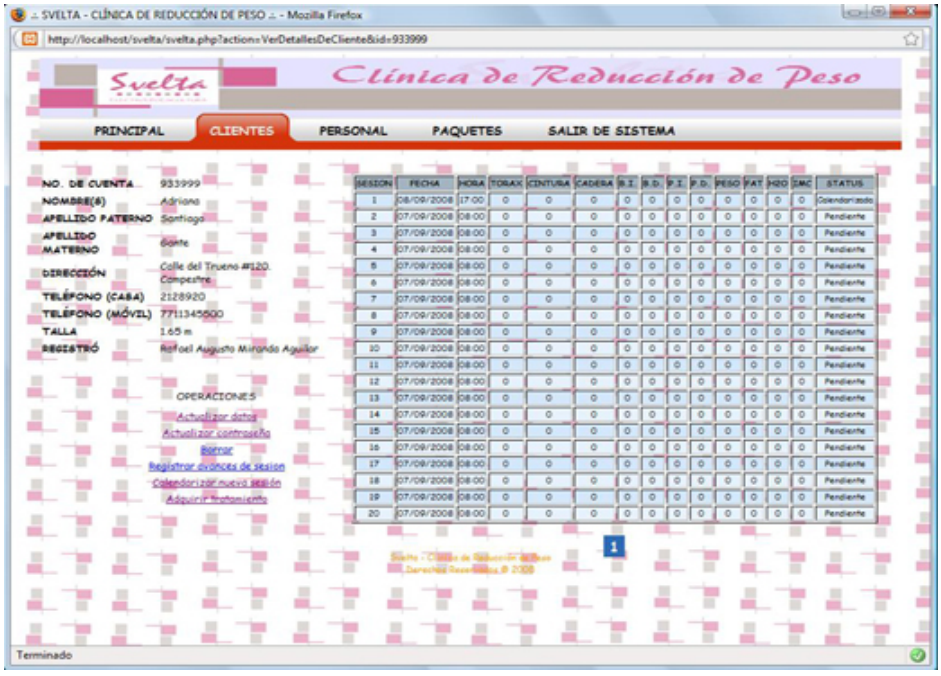
Esta documentación cubre un conjunto de pruebas realizadas sobre el Caso de Uso del mismo nombre.

Se realizó una prueba que consistió en el ingreso de medidas (y en su caso, abono al tratamiento) de la sesión con estado Actual de Adriana Santiago Gante.

El entorno del cual se parte para realizar la prueba es la opción *Ver Completo*, en el subsistema *Clientes*, de Adriana Santiago Gante.



a) Calendarización de la sesión 1 de *Adriana Santiago Gante*.



b) Resultado de adquisición de tratamiento.

Figura 4.21: Calendarización de sesiones.

4.4.7.2. Condiciones de ejecución.

Las condiciones de ejecución del caso de prueba son que el usuario administrador 100000 y su contraseña correspondiente estén dados de alta en la base de datos, y que esté actualmente dentro del subsistema Clientes.

4.4.7.3. Entrada.

1. El administrador pulsa el botón de la columna *Ver Completo* de Adriana Santiago Gante, en el subsistema *Clientes*, para ingresar a su historial y al menú de operaciones (ver fig. 4.22).

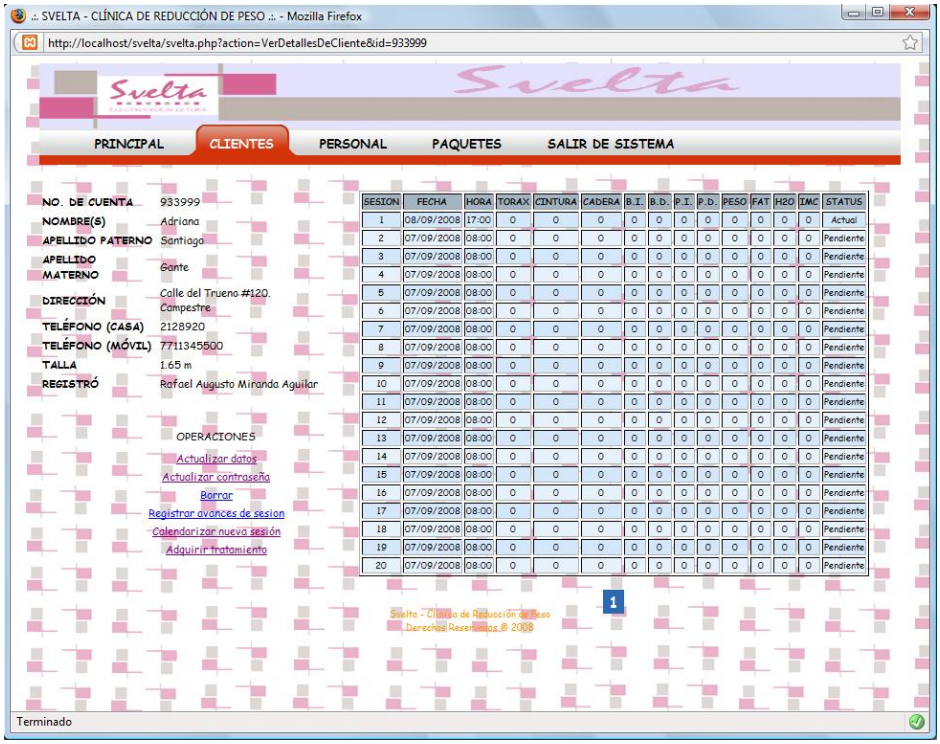


Figura 4.22: Sesión en curso de *Adriana Santiago Gante*.

2. El administrador pulsa el botón *Registrar avance de sesión*, para entrar al formulario de registro de medidas correspondiente. La sesión en curso está denoatada por estar en estatus de *Actual*.

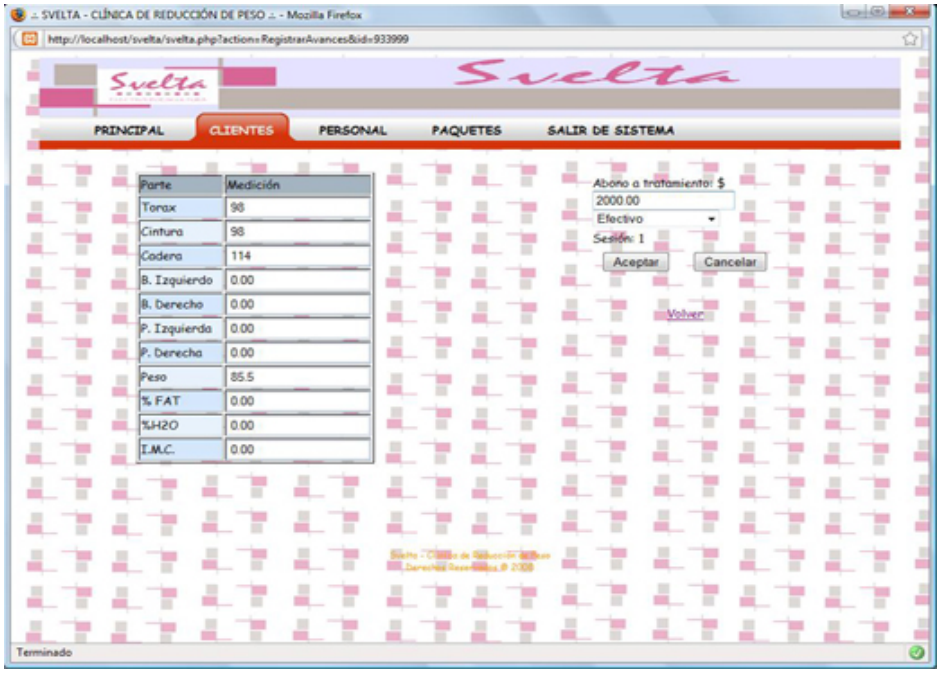
3. El administrador introduce la información siguiente solicitada en el formulario de registro de medidas (ver fig. 4.23.a). Cabe destacar que los campos de dicho formulario poseen un valor por defecto de *0.0*, por cuestiones de practicidad al rellenar el formulario; así que únicamente se mencionan los valores de cuyos campos se alteró su valor:
   - **Tórax**: 98.
   - **Cintura**: 98.
   - **Cadera**: 114.
   - **Peso**: 85.5.
   - **Abono**: $2000.00
   - **Forma de pago**: Efectivo.
4. Se pulsa el botón *Aceptar* de la aplicación para cargar los datos en la base de datos.
5. Aparece el historial de Adriana Santos Gante, en cuyo registro de avances se han actualizado algunos valores con los ingresados en el formulario (ver fig. 4.23.b), y actualizando el estado de la sesión *Calendarizada* a *Dada*.

4.4.7.4. Resultado esperado.

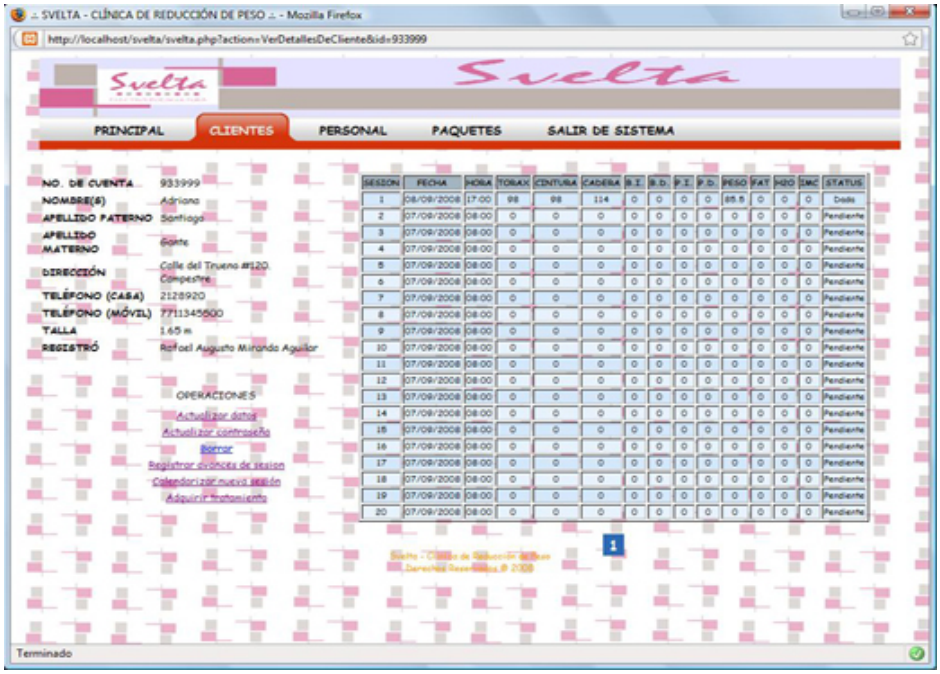
La pantalla del historial del cliente muestra los datos actualizados de la sesión en curso, y actualiza el estado de ésta a *Dada*, dando a entender que se ha cumplido satisfactoriamente dicha sesión.

4.4.7.5. Evaluación de la prueba.

Prueba superada con éxito.



a) Registro de avances de sesión 1 de *Adriana Santiago Gante*.



b) Resultado de registro de sesión.

Figura 4.23: Registro de sesiones.

Conclusiones.

Como se mencionó anteriormente, las actividades principales de *Svelta - Clínica de Reducción de Peso* consisten en la caledarización de sesiones, la alta de clientes a la organización y el registro de avances por sesión de un cliente.

Al realizar las pruebas pertinentes del sistema de información web de dicho negocio, se observó que los tiempos de operaciones se reducen de manera considerable, especialmente en la captura de ingreso de clientes, así como un mejor control de la calendarización de sesiones, lo cual representa uno de los mayores conflictos dentro de la operación diaria de la organización.

Otro punto a considerar es que, al mantener la información de manera organizada, los procesos de búsqueda son más eficientes que si se realizaran de manera manual; aunado a ello, la implementación de mecanismos de seguridad en los campos de llenado de información permitieron evitar inconsistencias de la información y llevar a cabo un mejor manejo de la misma.

Es de suponerse que las búsquedas que se produzcan dentro del sistema conlleven a una exploración más profunda de registros, especialmente en la cuestión de expedientes de clientes, o a realizar otras operaciones; por ello, se integraron una serie de enlaces dentro de los resultados de búsquedas que conducen a los detalles de cada uno de los resultados de búsqueda, asi como a posibles operaciones.

La seguridad del sistema en cuestión del control de tiempo de duración de sesión en un usuario inactivo dentro del sistema y el uso de pasaporte y contraseña para el ingreso, permitió asegurar la integridad de la información al permitir su acceso únicamente a usuarios autorizados, y el establecimiento de privilegios llevó a mejorar el funcionamiento del sistema, asignando elementos para realizar actividades propias de cada nivel jerárquico dentro del negocio.

Por otra parte, el empleo de un modelo de proceso, como lo fue RUP, permitió trabajar de manera organizada y rápida al facilitar el trabajo con todas las fases que involucra a la vez, por ser de naturaleza iterativo incremental; esta situación conllevó a avanzar en cada una de las etapas e irlas perfeccionando hasta llegar a la creación de artefactos casi definitivos. Este modelo de proceso en particular resultó muy cómodo y útil por su flexibilidad al adaptarse a cualquier proyecto y permitir refinaciones constantes que facilitan las transiciones o evoluciones posteriores del sistema en un menor tiempo y con la posibilidad de un decremento de posibles fallos en las nuevas versiones. Durante el desarrollo del sistema, las fases más difíciles de llevar a cabo fueron la *Elaboración*, debido a la complejidad de diagramación llevada a cabo, y la de *Construcción* en la cuestión del poco conocimiento previo sobre los lenguajes con los cuales se codificó el sistema y que, retrasaron los tiempos de entrega de módulos del mismo.

A modo de conclusión, se puede reafirmar que el sistema de información para *Svelta - Clínica de Reducción de Peso* permite desempeñar las actividades principales del negocio de modo eficaz y con menor coste de tiempo; las funcionalidades que presenta el sistema permiten la recolección, manipulación y procesamiento de la información de manera rápida, evitando así duplicidad de datos y optimizando los espacios del negocio al eliminar en gran medida la papelería que se genera dentro de la empresa, y cumplir los objetivos establecidos.

Como trabajo a futuro, se contemplan los siguientes aspectos:

1. Proceder a la etapa *Transición* de RUP para así realizar la entrega de una versión *definitiva* del proyecto, implementando el sistema dentro de las instalaciones de *Svelta - Clínica de Reducción de Peso.*

2. Renovar la *Hoja de Estilo en Cascada* (CSS) para dar mayor presentación al sistema y al sitio mismo, prestando atención en los diferentes aspectos que adquiere dependiendo el navegador empleado (Internet Explorer, Mozilla, Opera, etc.); especialmente, en la maquetación de página, vista de formularios, creación de tablas, botones y vínculos del sistema.

3. Aplicar Xajax en los formularios para reemplazar los mensajes tipo *alert* manejados mediante Javascript, y en operaciones de validación de campos y manipulación de base de datos con información ingresada en ellos.

4. Implementar en las interfaces construidas elementos que permitan adaptarse al surgimiento de próximas sucursales.

5. Como el sistema está totalmente orientado al personal del negocio, se pretende implementar dicho sistema con orientación a clientes, con el propósito que el cliente también interactúe con el sistema en cuanto a actualizar sus datos, calendarizar o cancelar sus sesiones sin la necesidad de acudir personalmente al negocio.

6. Extender la clase *Tratamiento* incluyendo los precios claves de las temporadas de ofertas, dependiendo el porcentaje establecido por la organización.

7. Incluir las clases *Pedido* y *Proveedor* para ampliar el campo de acción de *Svelta* al darse la aparición de franquicias, y extender el diseño del sistema para este fin.

8. Desarrollar manual de usuario del sistema.

9. Proponer la expansión del sistema a *Customer Relationship Management* (CRM) como estrategia corporativa, incorporando módulos de *marketing* especializado (publicidad *online*, *profile* corporativo), herramientas de análisis de datos (generación de reportes, estadísticas para toma de decisiones gerenciales) y servicios de atención al cliente, y reestructurando los módulo ya existentes del sistema.

Glosario.

- **Actualizar expediente.**

  Caso de uso que consiste en la modificación de datos de algún usuario del sistema por parte del personal, dependiendo del nivel de acceso de éste.

- **Adquirir tratamiento.**

  Actividad realizada por el cliente, el cual, una vez informado de los diversos paquetes que ofrece *Svelta*, queda convencido y decide comprar uno de los paquetes de tratamiento.

- **Ajax.**

  Acrónimo de *Asynchronous JavaScript And XML*.

- **C/C++.**

  C es un lenguaje de programación creado en 1972 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL. Al igual que B, es un lenguaje orientado a la implementación de sistemas operativos, concretamente Unix. Por otro lado, C++ es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C. Se puede decir que C++ es un lenguaje que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos.

- **Cabina.**

  División del establecimiento acondicionado para poder brindar el tratamiento de reducción de peso a cada uno de los clientes. Cuenta con sonido ambiental, televisión, camilla y un aparato reductor de peso. En cada sucursal, se contempla un número máximo de 2 cabinas de esta naturaleza.

- **Caja.**

Entidad abstracta que define el manejo de los ingresos económicos de la organización. Sirve como referencia para analizar los ingresos monetarios en un tiempo determinado, así como control de dihos recursos.

- **Calendarizar sesiones.**

  Caso de uso que consiste en que el cliente elija la fecha correspondiente para asistir a su próxima sesión del tratamiento adquirido. La función del personal es verificar la disponibilidad de día para dicha reservación.

- **Cancelar sesión.**

  Cuando un cliente ha calendarizado una sesión, y por situaciones de diversa naturaleza, no puede cumplir con ella, se encuentra en la posibilidad de cancelarla. Dentro de las políticas englobadas en el contrato del cliente se establece que, de cancelar su sesión antes de un lapso de 24 horas, tendrá derecho a recalendarizar esa sesión sin que afecte el progreso de su tratamiento; caso contrario, se tomará como sesión dada.

- **Cliente.**

  Cada uno de las personas que acuden a una determinada sucursal para solicitar informes sobre los servicios que ofrece *Svelta* y, posteriormente, adquieren un tratamiento. Son el motor primordial del buen funcionamiento del negocio.

- **Consultar expediente.**

  Caso de uso que consiste en la búsqueda de algún usuario dado de alta en el sistema para realizar alguna operación con dicho registro, como la actualización del registro.

- **Cuerpos Académicos.**

  Término aplicado a un grupo de profesores de tiempo completo que comparten objetivos académicos y una o varias Líneas de Generación o Aplicación de Conocimiento (LGAC).

- **Data warehouse.**

  Recopilación, unificación limpieza y filtro de datos, herramienta que facilita la implantación de un sistema *Data Mining*.

- **Dinámica del negocio.**

  Se refiere al conjunto de operaciones diarias que se llevan a cabo dentro de la organización, con o sin sistema; las actividades de mayor peso dentro de la organización son: *ingreso de clientes, calendarización de sesiones* y *registro de avances por sesión*, hablando en términos de utilidad para el sistema.

- **Disponibilidad de día.**

  Término empleado para denotar la certeza de que, al calendarizar una sesión, exista disponibilidad, tanto de fecha, hora y cabina, para reservarla, y así brindarla.

- **Estado.**

  Cada una de las entidades federativas que conforman la República Mexicana, debido a que las sucursales pueden expandirse hasta nievl nacional.

- **Expediente.**

  El expediente de un cliente es el conjunto de los tratamientos que ha adquirido a lo largo del tiempo, la evolución de cada uno de ellos y sus datos personales; el expediente de un miembro del personal consiste en sus datos personales, incluyendo su número de clientes ingresados hasta el momento.

- **Framework.**

  Estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un *framework* puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

- **IBM.**

  Abreviatura de *International Business Machines*. Una de las mayores compañías de fabricación de ordenares en el mundo desde los años cincuenta. Fue la creadora del PC, también se la conoce como *El Gigante Azul*, haciendo referencia al color de su logotipo.

- **Ingreso de usuarios.**

  Caso de uso en el cual un miembro del personal gestiona la alta en el sistema de un cliente u otro miembro del personal, dependiendo de su nivel de acceso. Entre los datos que se dan de alta para un usuario son: *nombre completo, dirección, teléfono particular, teléfono celular*, entre otros.

- **Iniciar sesión.**

  Actividad que consiste en entrar al *login* de la página principal de *Svelta*, para ingresar el número de cuenta del usuario y su contraseña, y así ingresar al sistema del negocio.

- **Input.**

  Tag de HTML empleado para mostrar controles que permiten a los usuario ingresar datos en un formulario. La conducta del control depende mayormente del atributo *type*. Este atributo define el tipo de control que se mostrará.

- **JDBC.**

  Driver de Java para el manejo de bases de datos.

- **JSP.**

  Archivo de tipo Javascript.

- **LDD.**

  Abreviatura de *Lenguaje de Definición de Datos.*

- **LMD.**

  Abreviatura de *Lenguaje de Manipulación de Datos.*

- **Mecánica del negocio.**

  Ver *Dinámica del negocio.*

- **Medidas.**

  Cantidad medida, en centímetros o en porcentaje, al término de la sesión de un cliente, con lo cual se actualiza el historial de progrso del mismo.

- **Nivel de acceso.**

  Terminología relacionada con los privilegios de manipulación que un usuario tiene sobre el sistema, y que va determinado por el puesto que ocupe en la organización. Los niveles de acceso del sistema de infromación de *Svelta* son:

  - **Administrador del sistema**: Todas las áreas, incluyendo mantenimiento y actualización del sitio y del sistema de la organización.
  - **Administrador del negocio**: Todas las áreas.
  - **Gerente**: Todas las áreas, exceptuando al personal que tenga el mismo puesto que él.

- **Terapeuta**: Área de Clientes y los subsistemas que involucra (*Calendarzación de Sesiones* y *Registro de Avances de Sesión*), y actualización de datos personales.

- **Output.**

  Salida de datos, se llama a los procesos de una computadora que entregan datos a otro dispositivo o directamente al usuario.

- **Paquete.**

  Conjunto de métodos de reducción de peso que conforman un tratamiento ofrecido por *Svelta*, y el cual selecciona el cliente, dependiendo de las necesidades de éste y sus posibilidades económicas.

- **Personal.**

  Cada uno de los miembros que laboran dentro de una sucursal del negocio en cuestión. Sinónimo de *empleado*.

- **Prompt.**

  Caracter o conjunto de caracteres que se muestran en una línea de comandos para indicar que está a la espera de órdenes. Éste puede variar dependiendo del intérprete de comandos y suele ser configurable.

- **Puesto.**

  Es la variedad de funciones que definidas concretamente en la organización, son realizadas por una persona como actividad laboral. Los puestos asignados dentro de *Svelta* son: **administrador de sistema, administrador de negocio, gerente** y *terapeuta*.

- **Registrar avance de sesión.**

  Caso de uso que consiste en actualizar en el sistema las medidas que se van obteniendo en una determinada sesión de un cliente, con el fin de analizar la evolución del mismo. Este caso de uso permite la actualización automática del expediente de dicho cliente.

- **Scripting.**

  En la programación de computadoras, un *script* es un programa o una secuencia de instrucciones que es interpretado y llevado a cabo por otro programa en lugar de ser procesado por el procesador de la computadora.

- **Sesión.**

  Es el marco de espacio temporal en el que se brinda el tratamiento adquirido por un cliente; ésta comprende una fecha determinada, un horario específico y una cabina donde se llevará a cabo la sesión de reducción de peso.

- **SQL.**

  Abreviatura de *Structured Query Language*; lenguaje desarrollado por IBM como parte del proyecto de System R, a principios de 1970. Actualmente, numerosos productos son compatibles con este lenguaje, y se ha convertido en el estándar de bases de datos relacionales.

- **SQLite.**

  Sistema de gestión de bases de datos relacional compatible con ACID, y que está contenida en una relativamente pequeña ($\tilde{5}$00KB) librería en C. SQLite es un proyecto de dominio público creado por D. Richard Hipp.

- **Sucursal.**

  Establecimiento que ofrece los servicios de la empresa a los clientes dentro de una determinada zona. Hasta el momento, la única sucursal existente es la matriz, localizada en Everardo Márquez #209, en Pachuca de Soto, Hidalgo.

- **Tag.**

  Marca con tipo que delimita una región en los lenguajes basados en XML.

- **Tratamiento.**

  Consiste en las distintas metodologías terapéuticas que permiten la reducción de peso y el moldeamiento corporal, y que, en conjunto, constituyen los paquetes que ofrece la empresa a sus clientes, y a su vez el paquete adquiere el nombre del tratamiento. Entre estos métodos se encuentran mesoterapia, masaje corporal, balón gástrico, entre otros. El tratamiento consta de 20 sesiones por máximo.

- **Usuario.**

  Generalización de todos los agentes externos que pueden acceder al sistema; ésto es, el personal que labora en *Svelta* y los clientes que contratan sus servicios. Dentro del Diagrama de Clases, se considera como la única *metaclase* dentro del modelo.

- **UTF-8.**

  8-bit Unicode Transformation Format es una norma de transmisión de longitud variable para caracteres codificados utilizando Unicode, creada por Rob Pike y Ken Thompson; usa grupos de bytes para representar el estándar de Unicode para los alfabetos de muchos de los lenguajes del mundo. Es especialmente útil para la transmisión sobre sistemas de correo de 8 bits. Usa de 1 a 4 bytes por carácter, dependiendo del símbolo de Unicode.

- **Validar usuario.**

  Caso de uso que permite determinar, mediante una comparación de campos con respecto a la base de datos de la organización, si un usuario es apto para ingresar al sistema o no.

- **WWW.**

  Acrónimo de *World Wide Web* (telaraña o malla mundial), es el sistema de información distribuido con mecanismos de hipertexto. Es el universo de servidores *http* que permiten mezclar texto, gráficos y archivos de sonido juntos.

- **XML.**

  Acrónimo de *Extensible Markup Language*, es un metalenguaje extensible de etiquetas desarrollado por el *World Wide Web Consortium* (W3C). Es una simplificación y adaptación del SGML (*Standard Generalized Markup Language*) y permite definir la gramática de lenguajes específicos de la misma manera que HTML es a su vez un lenguaje definido por SGML. Por lo tanto, XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.

- **Zona.**

  Se considera sinónimo de *región, municipio* o *delegación*; es la división de una entidad federativa para la cuestión de organización de las sucursales que vaya gestionando la organización. Hasta el momento, la única zona a considerar es *Pachuca de Soto*.

Apéndices.

Apéndice

A

Sentencias de *MySQL*.

A continuación se presentan las sentencias que emplea el manejador de bases de datos conocido como MySQL. Los presentes comandos aplican hasta las versiones más recientes, las cuales incluyen sólo algunos nuevos. Para mayor información sobre las sentencias, consultar la referencia [POSS05].

Sentencia	Descripción
ALTER TABLE	Permite modificar la estructura de una tabla existente.
ANALYZE TABLE	Analiza y almacena la distribución de claves de una tabla. Durante el análisis, la tabla se bloquea para lectura. Funciona con tablas MyISAM y BDB.
BACKUP TABLE	Copia al directorio de copia de seguridad el número mínimo de ficheros de tablas necesarios para restaurar la tabla, después de escribir cualquier cambio al disco. La sentencia funciona sólo para tablas MyISAM.
BEGIN	Se puede usar en lugar de **START TRANSACTION** para iniciar una transacción.
BEGIN WORK	Ídem.
CHECK TABLE	Sólo funciona con tablas MyISAM e InnoDB. Verifica la tabla o tablas buscando errores.
CHECKSUM TABLE	Devuelve la suma de comprobación (*checksum*) de una tabla.
COMMIT	Almacena los cambios en disco y asegura que las transacciones que se han rebobinado no se almacenen.
CREATE DATABASE	Crea una base de datos con el nombre dado.
CREATE TABLE	Crea una tabla con el nombre dado. Se debe poseer el privilegio CREATE para la tabla.
CREATE USER	Crea nuevas cuentas MySQL. Para usarla se debe tener el privilegio GRANT OPTION para la base de datos.
DELETE	Elimina columnas desde *table_name* que satisfagan la condición dada por la *where_definition*, y devuelve el número de registros borrados.
DESCRIBE	Proporciona información sobre las columnas de una tabla.
DO	Versión taquigráfica de expresión **SELECT**, pero tiene la ventaja de que es un poco más rápida cuando el resultado no importa.
DROP DATABASE	Elimina en su totalidad la base de datos.
DROP INDEX	Elimina el índice con el nombre *index_name* de la tabla *tbl_name*.
DROP TABLE	Elimina una o más tablas.
DROP USER	Elimina una o más cuentas MySQL. Para usarla se debe poseer el privilegio **GRANT OPTION** para la base de datos.

FLUSH	Limpia algo del caché interno usado por MySQL. Para poder ejecutarlo, se debe poseer el privilegio **RELOAD**.
GRANT	Permite a los administradores del sistema crear cuentas de usuario MySQL y conceder derechos de esas cuentas.
HANDLER	Proporciona un acceso directo al interfaz del motor de almacenamiento de una tabla MyISAM.
INSERT	Inserta nuevas filas en una tabla existente.
INSERT...SELECT	Inserta filas seleccionadas de otra tabla o tablas.
INSERT DELAYED	Obtiene una confirmación de inserción por parte del servidor de forma inmediata, y la fila se almacena en una cola para ser insertada cuando la tabla no esté en uso por ningún otro proceso.
JOIN	Soporta sintaxis para ser usadas como la parte de referencia de tabla en sentencias **SELECT** y sentecias **DELETE** y **UPDATE** multitabla.
KILL	Termina un hilo de conexión a *mysqld*.
LOAD DATA	Lee filas desde un fichero de texto a una tabla a gran velocidad.
LOCK TABLES	Bloquea tablas para el hilo actual.
OPTIMIZE TABLE	Permite recuperar el espacio no usado y para desfragmentar el fichero de datos.
RENAME TABLE	Renombra una o más tablas.
REPAIR TABLE	Repara una tabla posiblemente corrupta. Sólo funciona con tablas MyISAM.
REPLACE	Trabaja exactamente igual que **INSERT**, excepto que si existe algún registro viejo en la tabla que tenga el mismo valor que uno nuevo para un índice **PRIMARY KEY** o **UNIQUE**, el viejo se borra antes de que el nuevo sea insertado.
RESET	Se usa para limpiar el estado de varias operaciones del servidor. También actúa como una versión más fuerte del comando **FLUSH**.
REVOKE	Permite a los administradores del sistema crear cuentas de usuario MySQL y revocar derechos de esas cuentas.

ROLLBACK	Permite ignorar los cambios hechos desde el principio de la transacción.
SELECT	Se usa para recuperar filas seleccionadas de una o más tablas.
SET	Activa varias opciones que afectan al funcionamiento del servidor y el cliente.
SET TRANSACTION	Asigna el nivel de aislamiento de transacción para transacciones globales, la sesión completa o para la siguiente transacción.
SHOW	Proporciona información sobre bases de datos, tablas, columnas o información de estado sobre el servidor. Si se usa la parte LIKE, la cadena de patrón puede usar los caracteres % y _ como comodines. El patrón es útil para restringir la salida de la sentencia a los valores que coincidan con él.
SHOW CHARACTER SET	Muestra los conjuntos de caracteres disponibles.
SHOW COLLATION	Incluye todos los conjuntos de caracteres disponibles.
SHOW COLUMNS	Lista las columnas de una tabla dada.
SHOW CREATE DATABASE	Muestra una sentencia **CREATE DATABASE** que creará la base de datos dada.
SHOW CREATE TABLE	Muestra la sentencia **CREATE TABLE** que creará la tabla dada.
SHOW CREATE VIEW	Muestra una sentencia **CREATE VIEW** que creará la vista dada.
SHOW DATABASES	Lista las bases de datos en el ordenador del servidor MySQL.
SHOW ENGINES	Muestra la información de estado sobre los motores de almacenamiento.
SHOW ERRORS	Similar a **SHOW WARNINGS**, excepto que en lugar de mostrar errores, avisos y notas, muestra sólo errores.
SHOW GRANTS	Lista las sentencias **GRANT** que deben ser lanzadas para duplicar los privilegios para una cuenta de usuario MySQL.
SHOW INDEX	Devuelve la información de índices de una tabla en un formato parecido a la llamada SQLStatistics de ODBC.

SHOW INNODB STATUS	Muestra información detallada sobre el estado del motor de almacenamiento InnoDB.
SHOW KEYS	Devuelve la información de índices de una tabla en un formato parecido a la llamada SQLStatistics de ODBC.
SHOW LOGS	Muestra información de estado sobre los ficheros de diario existentes.
SHOW PRIVILEGES	Muestra la lista de privilegios del sistema que el servidor MySQL actual soporta.
SHOW PROCESSLIST	Muestra qué procesos están corriendo.
SHOW STATUS	Proporciona información sobre el estado del servidor.
SHOW TABLE STATUS	Funciona como **SHOW TABLE**, pero proporciona mucha más información sobre cada tabla.
SHOW TABLES	Lista las tablas no temporales en una base de datos dada.
SHOW VARIABLES	Muestra el valor de algunas variables de sistema de MySQL.
SHOW WARNINGS	Muestra los mensajes de error, aviso y notas que resulten de la última sentencia que haya generado mensajes, o nada si la última sentencia que ha usado la tabla no ha generado mensajes.
START TRANSACTION	Modo recomendado para empezar una transacción.
TRUNCATE	Vacía una tabla por completo.
UNION	Se usa para combinar los resultados de varias sentencias **SELECT** en un único conjunto de resultados.
UNLOCK TABLES	Libera cualquier bloqueo para el hilo actual.
UPDATE	Actualiza columnas de filas existentes de una tabla con nuevos valores.
USE	Indica a MySQL que use la base de datos *db_name* como la base de datos por defecto (actual) en sentencias subsiguientes.

Apéndice

B

Funciones de *Xajax*.

Para comprender la calidad del *framework* conocido como Xajax, empleado dentro del desarrollo del presente proyecto, se presentan la clasificación de funciones que pueden ser empleadas por el programador para generar aplicaciones de mayor calidad.

B.1. Métodos de creación.

Permiten crear una etiqueta dentro del árbol de etiquetas HTML.

Función	Descripción
addCreate($id_etiqueta_superior, $etiqueta, $id)	Crea una etiqueta del tipo *$etiqueta* dentro de la etiqueta especificada por *$id_etiqueta_superior* y con el id *$id*.
addInsert($id_etiqueta_anterior, $etiqueta, $id)	Funciona exactamente igual que *addCreate*, pero éste, en lugar de crear una etiqueta *$etiqueta* **dentro** de *$id_etiqueta_superior*, la crea **antes** de la etiqueta con el id *$id_etiqueta_anterior*.
addInsertAfter($id_etiqueta_posterior, $etiqueta, $id)	Parecida a la anterior, pero en lugar de crear la etiqueta **antes** de la que indica su primer argumento, la crea **después**.
addCreateInput($id_etiqueta_superior, $type, $nombre, $id)	Crean etiquetas del tipo *input*. A éstas se les aplican los atributos *type=$type* y *name=$nombre*.
addInsertInput($id_etiqueta_anterior, $type, $nombre, $id)	Ídem.
addInsertInputAfter($id_etiqueta_posterior, $type, $nombre, $id)	Ídem.

B.2. Métodos de interacción con Javascript.

Ésta es una colección de métodos que ejecutan Javascript.

Función	Descripción
addScript($código_javascript)	Ejecuta el código Javascript que tiene como parámetro.
addScriptCall($función_javascript, $arg1, $arg2...)	Llama a una función Javascript con el nombre *$function_javascript* y como argumentos los argumentos del segundo al último de éste método.
addIncludeScipt($fichero_js)	Tiene como argumento la ruta de un fichero *.js* y lo carga dentro del documento.

addAlert($advertencia) / addRedirect($página_nueva, $tiempo)	Son exactamente lo mismo que *addScript('alert("$advertencia");')*, que alerta la *$advertencia*, y que *addScript("setTimeOut(window.location='".$página_nueva."', ".$tiempo*1000.");")* que redirecciona a la página *$página_nueva* al cabo de *$tiempo segundos*.
addConfirmComands($num, $mensaje)	Lanza un *prompt* con el *$mensaje*. Si el usuario presiona *Cancelar*, los siguientes *$num* comandos serán ignorados.

B.3. Métodos de modificación.

Permiten cambiar el contenido o un atributo de una etiqueta.

Función	Descripción
addAssign($id, $atributo, $datos)	Reemplaza todo el contenido de un atributo (*$atributo*) de una etiqueta (con el id *$id*) por el texto *$data*. Así que si se desea modificar el contenido de una capa o un parágrafo <p>, se ha de pasar por $id el del parágrafo o la capa, por *innerHTML* el *$atributo* y como el texto nuevo el *$datos*.
addAppend($id, $atributo, $datos)	Añade texto al fin del texto que ya hay en la etiqueta con id *$id*.
addPrepend($id, $atributo, $datos)	Añade texto antes de la etiqueta con id *$id*.
addReplace($id, $atributo, $texto_a_buscar, $texto_a_reemplazar)	Permite, en el atributo *$atributo* de la etiqueta con el id *$id[i]*, reemplazar el texto[i] *$texto_a_buscar* por el texto *$texto_a_reemplazar*.
addClear($id, $atributo)	Borra el contenido de un atributo (*$atributo*) o del contenido (poniendo por *$atributo innerHTML*) de una etiqueta (con id *$id*); es lo mismo que hacer un *addAssign($id, $atributo, "")*.

B.4. Métodos de adición de eventos.

Como su categoría lo indica, permiten añadir eventos de diferente naturaleza.

Función	Descripción
addEvent($id, $evento, $código_javascript)	Añade un evento a una etiqueta con el id *$id*, y cuando se dispara ese evento, se ejecuta el *$código_javascript*.
addHandler($id_etiqueta, $evento, $función_javascript)	Funciona de forma parecida al anterior, pero en lugar de poner un código como tercer argumento se coloca una función Javascript.
addRemoveHandler($id_etiqueta, $evento, $función_javascript)	Elimina el evento de los dos métodos anteriores.
setCharEncoding($codificación)	Cambia la codificación configurada por defecto de la respuesta XML de Xajax (*UTF-8*).
outputEntitiesOn()	Cuando se usa un método de modificación donde se añade texto donde hay etiquetas HTML, estas etiquetas se mostrarán como texto en el navegador.
outputEntitiesOff()	Cuando se usa un método de modificación donde se añade texto donde hay etiquetas HTML, estas etiquetas serán leídas por el navegador como tales.
loadXML($xml)	Permite asignar a un objeto *xajaxResponse* el XML de otro objeto *xajaxResponse*.

B.5. Métodos de eliminación.

De métodos para eliminar una *tag* sólo hay uno: **addRemove($id)**, y tan sólo se le ha de pasar como argumento el *id* de la etiqueta a eliminar.

Apéndice

C *Script* de Base de Datos de *Svelta.*

El resultado del Diagrama Relacional hecho en el Capítulo 3 es el *script* para generar la base de datos de *Svelta*, el cual se presenta en este Apéndice.

```
Drop database if exists SVELTA;
Create database SVELTA;
Use SVELTA;

Create table PUESTO (
    id_Pu Int NOT NULL auto_increment,
    NomPu Varchar(30),
    UNIQUE (id_Pu),
 Primary Key (id_Pu)) ENGINE = MyISAM;

Create table ESTADO (
    id_Est Int NOT NULL auto_increment,
    NomEst Varchar(30),
    UNIQUE (id_Est),
 Primary Key (id_Est)) ENGINE = MyISAM;

Create table SUCURSAL (
    id_S Int NOT NULL auto_increment,
    NomS Varchar(40),
    DirS Varchar(60),
    TelS Varchar(20),
    id_Z Int NOT NULL,
    UNIQUE (id_S),
 Primary Key (id_S,id_Z)) ENGINE = MyISAM;

Create table PERSONAL (
    id_P Double NOT NULL,
    NoIngresP Int,
    id_S Int NOT NULL,
    id_Pu Int NOT NULL,
    UNIQUE (id_P),
 Primary Key (id_P,id_S,id_Pu)) ENGINE = MyISAM;

Create table CLIENTE (
    id_C Double NOT NULL,
    AlturaCliente Float,
    FechaDeIngreso Date,
    PersonalQueIngreso Int,
    UNIQUE (id_C),
 Primary Key (id_C)) ENGINE = MyISAM;

Create table PAQUETE (
    id_Pac Int NOT NULL auto_increment,
    NomPac Varchar(50),
    CostNPac Float,
    StatusPac Varchar(20),
    UNIQUE (id_Pac),
 Primary Key (id_Pac)) ENGINE = MyISAM;

Create table SESION (
    id_Ses Double NOT NULL auto_increment,
    FecSes Date,
    CabinaSes Int,
    NoSes Int,
    StatusSes Varchar(20),
    id_PC Double NOT NULL,
    id_HS Int NOT NULL,
```

```
    UNIQUE (id_Ses),
 Primary Key (id_Ses,id_PC,id_M,id_Pago,id_HS)) ENGINE = MyISAM;

Create table CANCELACION (
    id_Can Double NOT NULL auto_increment,
    FecCan Date,
    id_Ses Double NOT NULL,
    UNIQUE (id_Can),
 Primary Key (id_Can,id_Ses)) ENGINE = MyISAM;

Create table PAQUETE_CLIENTE (
    id_PC Double NOT NULL auto_increment,
    StatusPC Varchar(30),
    id_Pac Int NOT NULL,
    id_C Double NOT NULL,
    UNIQUE (id_PC),
 Primary Key (id_PC,id_Pac,id_C)) ENGINE = MyISAM;

Create table ZONA (
    id_Z Int NOT NULL auto_increment,
    NomZ Varchar(35),
    id_Est Int NOT NULL,
    UNIQUE (id_Z),
 Primary Key (id_Z,id_Est)) ENGINE = MyISAM;

Create table USUARIO (
    id_U Double NOT NULL,
    NomU Varchar(20),
    ApPatU Varchar(20),
    ApMatU Varchar(20),
    DirU Varchar(50),
    TelCU Varchar(15),
    TelMU Varchar(15),
    ContU Varchar(32),
    Conectado Char(1),
    Actividad Char(15),
    UNIQUE (id_U),
 Primary Key (id_U)) ENGINE = MyISAM;

Create table CAJA (
    id_Pago Double NOT NULL auto_increment,
    FechaPago Date,
    HoraPago Time,
    MontoPago Float,
    TipoPago Varchar(20),
    id_PC Double NOT NULL,
    UNIQUE (id_Pago),
 Primary Key (id_Pago)) ENGINE = MyISAM;

Create table MEDIDAS (
    id_M Double NOT NULL auto_increment,
    Torax Float,
    Cintura Float,
    Cadera Float,
    Brazo_I Float,
    Brazo_D Float,
    Pierna_I Float,
```

```
    Pierna_D Float,
    Peso Float,
    FAT Float,
    H2O Float,
    IMC Float,
    id_Ses Double NOT NULL,
    UNIQUE (id_M),
 Primary Key (id_M)) ENGINE = MyISAM;

Create table TRATAMIENTO (
    id_T Int NOT NULL auto_increment,
    NomT Varchar(25),
    UNIQUE (id_T),
 Primary Key (id_T)) ENGINE = MyISAM;

Create table PAQUETE_TRATAMIENTO (
    id_PT Int NOT NULL auto_increment,
    NoSesPT Int,
    id_Pac Int NOT NULL,
    id_T Int NOT NULL,
    UNIQUE (id_PT),
 Primary Key (id_PT,id_Pac,id_T)) ENGINE = MyISAM;

Create table HORARIO (
    id_HS Int NOT NULL auto_increment,
    HoraSes Time,
    UNIQUE (id_HS),
 Primary Key (id_HS)) ENGINE = MyISAM;


Alter table PERSONAL add Foreign Key (id_Pu) references PUESTO (id_Pu)
  on delete  restrict on update  restrict;
Alter table ZONA add Foreign Key (id_Est) references ESTADO (id_Est)
  on delete  restrict on update  restrict;
Alter table PERSONAL add Foreign Key (id_S) references SUCURSAL (id_S)
  on delete  restrict on update  restrict;
Alter table PAQUETE_CLIENTE add Foreign Key (id_C) references CLIENTE
  (id_C) on delete  restrict on update  restrict;
Alter table PAQUETE_CLIENTE add Foreign Key (id_Pac) references
  PAQUETE (id_Pac) on delete  restrict on update  restrict;
Alter table PAQUETE_TRATAMIENTO add Foreign Key (id_Pac) references
  PAQUETE (id_Pac) on delete  restrict on update  restrict;
Alter table CANCELACION add Foreign Key (id_Ses) references SESION
  (id_Ses) on delete  restrict on update  restrict;
Alter table SESION add Foreign Key (id_PC) references PAQUETE_CLIENTE
  (id_PC) on delete  restrict on update  restrict;
Alter table SUCURSAL add Foreign Key (id_Z) references ZONA (id_Z)
  on delete  restrict on update  restrict;
Alter table PAQUETE_TRATAMIENTO add Foreign Key (id_T) references
  TRATAMIENTO (id_T) on delete  restrict on update  restrict;
Alter table SESION add Foreign Key (id_HS) references HORARIO (id_HS)
  on delete  restrict on update  restrict;
Alter table CAJA add Foreign Key (id_PC) references PAQUETE_CLIENTE
  (id_PC) on delete  restrict on update  restrict;
Alter table MEDIDAS add Foreign Key (id_Ses) references SESION (id_Ses)
  on delete  restrict on update  restrict;
```

```
Insert into ESTADO values
(1, 'Chihuahua'),
(2, 'Hidalgo'),
(3, 'Coahuila'),
(4, 'Sonora'),
(5, 'Tamaulipas'),
(6, 'Nuevo Leon'),
(7, 'Baja California Norte'),
(8, 'Baja California Sur'),
(9, 'Sinaloa'),
(10, 'Zacatecas'),
(11, 'San Luis Potosi;'),
(12, 'Durango'),
(13, 'Nayarit'),
(14, 'Aguascalientes'),
(15, 'Jalisco'),
(16, 'Guanajuato'),
(17, 'Queretaro'),
(18, 'Colima'),
(19, 'Michoacan'),
(20, 'Mexico'),
(21, 'Distrito Federal'),
(22, 'Tlaxcala'),
(23, 'Morelos'),
(24, 'Guerrero'),
(25, 'Puebla'),
(26, 'Veracruz'),
(27, 'Oaxaca'),
(28, 'Tabasco'),
(29, 'Chiapas'),
(30, 'Campeche'),
(31, 'Yucatan'),
(32, 'Quintana Roo');


Insert into HORARIO values
(1, '08:00:00'),
(2, '09:00:00'),
(3, '10:00:00'),
(4, '11:00:00'),
(5, '12:00:00'),
(6, '13:00:00'),
(7, '14:00:00'),
(8, '15:00:00'),
(9, '16:00:00'),
(10, '17:00:00'),
(11, '18:00:00'),
(12, '19:00:00');

Insert into PUESTO values
(1, 'Administrador'),
(2, 'Supervisor'),
(3, 'Gerente'),
(4, 'Recepcionista'),
(5, 'Terapeuta');

Insert into SUCURSAL values
```

```
(1, 'Pachuca (Everardo Marquez)', 'Blvd. Everardo Marquez #209 Col.
 Lomas Residencial', '7712121616', 1);

Insert into USUARIO values
('100000', 'Rafael Augusto', 'Miranda', 'Aguilar', 'Calle de la Mora
 #304 Fracc. Priv. del Alamo', '7198178', '7711431162',
 '903abc663a963dae3c56397ce08cf2b4', '0', '0');

Insert into PERSONAL values
('100000', 0, 1, 1);

Insert into ZONA values
(1, 'Pachuca de Soto',2);

Insert into PAQUETE values
(1, 'Electrolipoescultura - Guia 1', 3000, 'Activo'),
(2, 'Electrolipoescultura - Guia 2', 5500, 'Activo'),
(3, 'Electrolipoescultura - Nutricion 1', 3400, 'Activo'),
(4, 'Electrolipoescultura - Nutricion 2', 6000, 'Activo'),
(5, 'Electrolipoescultura - Homeopatia 1', 3600, 'Activo'),
(6, 'Electrolipoescultura - Homeopatia 2', 6500, 'Activo'),
(7, 'Electrolipoescultura - Completo 1', 4000, 'Activo'),
(8, 'Electrolipoescultura - Completo 2', 7000, 'Activo'),
(9, 'Mesoterapia 1', 5000, 'Activo'),
(10, 'Mesoterapia 2', 9000, 'Activo'),
(11, 'Electrolipoescultura - Mantenimiento', 1000, 'Activo'),
(12, 'Balon Intragastrico', 30000, 'Activo');


Insert into PAQUETE_TRATAMIENTO values
(1, 10, 1, 1),
(2, 1, 1, 2),
(3, 20, 2, 1),
(4, 1, 2, 2),
(5, 10, 3, 1),
(6, 2, 3, 3),
(7, 20, 4, 1),
(8, 3, 4, 3),
(9, 10, 5, 1),
(10, 1, 5, 2),
(11, 2, 5, 4),
(12, 20, 6, 1),
(13, 1, 6, 2),
(14, 3, 6, 4),
(15, 10, 7, 1),
(16, 2, 7, 3),
(17, 2, 7, 4),
(18, 20, 8, 1),
(19, 3, 8, 3),
(20, 3, 8, 4),
(21, 10, 9, 1),
(22, 2, 9, 3),
(23, 2, 9, 4),
(24, 10, 9, 5),
(25, 20, 10, 1),
(26, 3, 10, 3),
(27, 3, 10, 4),
```

```
(28, 20, 10, 5),
(29, 4, 11, 1),
(30, 1, 11, 3),
(31, 1, 11, 4),
(32, 6, 12, 6);


Insert into TRATAMIENTO values
(1, 'Electrolipoescultura'),
(2, 'Guia Nutricional'),
(3, 'Consulta Nutricion'),
(4, 'Consulta Homeopatica'),
(5, 'Mesoterapia'),
(6, 'Balon Intragastrico'),
(7, 'Auriculoterapia'),
(8, 'Acupuntura'),
(9, 'Fibra Svelta');
```

Apéndice

D Código fuente de componentes.

Según el Diagrama de Componentes presentado en el Capítulo 4, en este Apéndice se muestra el código fuente generado para cada una de las partes que componen el sistema.

Cabe destacar que el componente *class.paginacion.php* fue extraído de la fuente [PINJ05], y modificado para mejorar su diseño visual.

D.1. class.ajax.php

```
<?php

function formulario_tratamiento($form_entrada)
{
    $respuesta = new xajaxResponse('ISO-8859-1');

    $pac=$form_entrada["pc"];
    $idc=$form_entrada["id"];
    $cont_form='<form id="carga"><div align="center">';

    $sql="SELECT * FROM paquete_tratamiento, tratamiento WHERE
        paquete_tratamiento.id_Pac=".$pac." AND paquete_tratamiento.
        id_T= tratamiento.id_T";
    $res=mysql_query($sql);

    if($res)
    {
        $color0="#A4B4C1";
        $color1="#D5E7FB";
        $color2="#E8F2FC";
        $color3="#E0FAC5";
        $color=$color1;

        $cont_form.= '</div><td width="207"><div align="center">
            <table width="220" border="1">
            <tr bgcolor="'.$color0.'"><td>Tratamiento</td><td>
            Sesiones</td></tr>';

        while($row=mysql_fetch_array($res))
        {
            $tratamiento=$row['NomT'];
            $noses=$row['NoSesPT'];

            $cont_form.= '<tr bgcolor="'.$color.'" onMouseOver=
                "this.style.backgroundColor=\''.$color3.'\';"
                onMouseOut="this.style.backgroundColor=\''.$color.
                '\';"><td>'.$tratamiento.'</td><td>'.$noses.'
                </td></tr>';

                if($color==$color1)
                {
                    $color=$color2;
                }
                else
                {
                    $color=$color1;
                }
        }

        mysql_free_result($res);
        $cont_form.='</table>';

        $sql="SELECT CostNPac FROM paquete WHERE id_Pac=".$pac."";
        $res=mysql_query($sql);
```

```
        if($row=mysql_fetch_array($res))
        {
            $cont_form.='<br><br>Costo: $'.$row['CostNPac'].'.00';
        }

        mysql_free_result($res);

        $cont_form.='<br /><br />Abono de tratamiento:
        <input type="text" name="pago" value="0.00" onKeyPress="return
            validarSoloNumeros(event)">
        <select name="forma_pago">
        <option value="Efectivo">Efectivo</option><option value="Tarjeta
            de credito">Tarjeta de credito</option></select>
        <input type="hidden" name="paquete" value="'.$pac.'">
        <input type="hidden" name="id" value="'.$idc.'">
        <br><br>
        <input type="button" class="formbutton" name="registrar"
        value="Cargar" onclick="xajax_registrar_paquete(xajax.
        getFormValues(\'carga\'))"></div></form>';
    }

    $respuesta->addAssign("datos","innerHTML",$cont_form);
    return $respuesta;
}

function registrar_paquete($form_entrada)
{
    $respuesta = new xajaxResponse('ISO-8859-1');

    $numero=$form_entrada["paquete"];
    $monto=$form_entrada["pago"];
    $tipo=$form_entrada["forma_pago"];
    $id=$form_entrada["id"];

    $sql="INSERT INTO paquete_cliente (StatusPC,id_Pac,id_C) VALUES
        ('En Curso','".$numero."','".$id."')";
    mysql_query($sql);

    $sql="SELECT id_PC FROM paquete_cliente WHERE StatusPC='En Curso'
        AND id_C=".$id."";
    $res=mysql_query($sql);

    if($row=mysql_fetch_array($res))
    {
        $hoy=date("Y-m-d");
        $hora=date("H:i:s");

        if($monto!=0.0)
        {
            $sql1="INSERT INTO caja (FechaPago, HoraPago, MontoPago, TipoPago, id_PC)
            VALUES ('".$hoy."','".$hora."',".$monto.",'".$tipo."',".$row['id_PC'].")";
            mysql_query($sql1);
        }

        $sql2="INSERT INTO sesion (FecSes, CabinaSes, NoSes, StatusSes,id_PC,id_HS) VALUES ";
        for ($i=1; $i<=20; $i++)
        {
```

```
            $sql2.="('".$hoy."',0,'".$i."','Pendiente',".$row['id_PC'].", 1)";

            if($i<20)
            {
                $sql2.=",";
            }
        }

        mysql_query($sql2);

        mysql_free_result($res);

        $sql3="SELECT id_Ses FROM sesion, paquete_cliente WHERE sesion.id_PC=paquete_cliente.id_PC
        AND sesion.id_PC=".$row['id_PC']."";
        $res3=mysql_query($sql3);

        while($row=mysql_fetch_array($res3))
        {
            $sql4="INSERT INTO medidas (Torax, Cintura, Cadera, Brazo_I, Brazo_D, Pierna_I,
            Pierna_D, Peso, FAT, H2O, IMC, id_Ses) VALUES (0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0,0.0, 0.0, ".$row['id_Ses'].")";
            mysql_query($sql4);
        }

        mysql_free_result($res3);
    }

    $respuesta->addScript("window.location='".$_SERVER['PHP_SELF']."?action=VerDetallesDeCliente&
    id=".$id."'");
    return $respuesta;
}

function mostrar_sesiones($form_entrada)
{
    $respuesta = new xajaxResponse('ISO-8859-1');

    $fecha=$form_entrada['fecha_sesion'];
    $num=$form_entrada['numero'];
    $id=$form_entrada['id'];

    $sql="SELECT * FROM horario";
    $res=mysql_query($sql);

    if($res)
    {
        $color0="#A4B4C1";
        $color1="#D5E7FB";
        $color2="#E8F2FC";
        $color3="#E0FAC5";
        $color=$color1;

        $cont_form='<table border="1"><tr bgcolor="'.$color0.'">
            <td width="92"><div align="center"><strong>HORA</strong></div></td>
            <td width="100"><div align="center"><strong>CABINA 1</strong></div></td>
            <td width="100"><div align="center"><strong>CABINA 2</strong></div></td>';
```

```
while($row=mysql_fetch_array($res))
{
    $cont_form.='<tr bgcolor="'.$color.'" onMouseOver="this.style.backgroundColor=
    \''.$color3.'\';" onMouseOut="this.style.backgroundColor=\''.$color.'\';">
    <td width="92"><div align="center">'.$row['HoraSes'].'</div></td>';

    $sql="SELECT * FROM sesion,horario WHERE FecSes='".$fecha."' AND sesion.id_HS=
    ".$row['id_HS']." AND HoraSes='".$row['HoraSes']."' AND
    (StatusSes='Actual' OR StatusSes='Calendarizada')";
    $iBandera=0;
    $res2=mysql_query($sql);

    while($row2=mysql_fetch_array($res2))
    {
        $iBandera=1;
        $sesion=$row2['id_Ses'];

        if($row2['StatusSes']=='Actual'||$row2['StatusSes']=='Calendarizada')
        {
            if($row2['CabinaSes']==1)
            {
                $sql="SELECT CabinaSes,StatusSes FROM sesion,horario WHERE FecSes=
                '".$fecha."' AND sesion.id_HS=".$row['id_HS']."
                AND HoraSes='".$row['HoraSes']."' AND
                (StatusSes='Actual' OR StatusSes='Calendarizada') AND CabinaSes=2";
                $res3=mysql_query($sql);

                if(mysql_num_rows($res3)==0)
                {
                    $cont_form.='<td width="100"><div align="center">
                    <img src="imagenes/paloma.gif"
                    width="15" height="15" border="0"></div></td>
                    <td width="100"><div align="center"><a href="'.$_SERVER['PHP_SELF'].'?
                    action=VerDetallesDeCliente&id='.$id.'" onclick="xajax_cargar_sesion
                    ('.$row['id_HS'].','.$num.',\''.$fecha.'\',2,'.$id.')">
                    <img src="imagenes/palomita.png" alt="Libre" width="15" height="15"
                    border="0"></a></div></td>';
                }
                else
                {
                    $cont_form.='<td width="100"><div align="center">
                    <img src="imagenes/paloma.gif" width="15" height="15" border="0">
                    </div></td>';
                }

                mysql_free_result($res3);
            }
            elseif($row2['CabinaSes']==2)
            {
                $sql="SELECT CabinaSes,StatusSes FROM sesion,horario WHERE FecSes=
                '".$fecha."' AND sesion.id_HS=".$row['id_HS']."
                AND HoraSes='".$row['HoraSes']."' AND
                (StatusSes='Actual' OR StatusSes='Calendarizada') AND CabinaSes=1";
                $res3=mysql_query($sql);

                if(mysql_num_rows($res3)==0)
                {
```

```
                        $cont_form.='<td width="100"><div align="center">
                        <a href="'.$_SERVER['PHP_SELF'].'?action=VerDetallesDeCliente&id='.$id.'"
                        onclick="xajax_cargar_sesion('.$row['id_HS'].',
                        '.$num.',\''.$fecha.'\',1,'.$id.')">
                        <img src="imagenes/palomita.png" alt="Libre" width="15"
                        height="15" border="0"></a></div></td>
                        <td width="100"><div align="center"><img src="imagenes/paloma.gif"
                        width="15" height="15" border="0"></div></td>';
                    }
                    else
                    {
                        $cont_form.='<td width="100"><div align="center">
                        <img src="imagenes/paloma.gif" width="15" height="15" border="0">
                        </div></td>';
                    }

                    mysql_free_result($res3);
                }
            }
            else
            {
                $iBandera=0;
            }
        }

        if ($iBandera==0)
        {
            $cont_form.='<td width="100"><div align="center"><a href="'.$_SERVER['PHP_SELF'].'?
            action=VerDetallesDeCliente&id='.$id.'" onclick="xajax_cargar_sesion('.$row['id_HS'].',
            '.$num.',\''.$fecha.'\',1,'.$id.')"><img src="imagenes/palomita.png" alt="Libre"
            width="15" height="15" border="0"></a></div></td>
            <td width="100"><div align="center"><a href="'.$_SERVER['PHP_SELF'].'?action=
            VerDetallesDeCliente&id='.$id.'" onclick="xajax_cargar_sesion('.$row['id_HS'].',
            '.$num.',\''.$fecha.'\',2,'.$id.')"><img src="imagenes/palomita.png"
            alt="Libre" width="15" height="15" border="0"></a></div></td>';
        }

        if($color==$color1)
        {
            $color=$color2;
        }
        else
        {
            $color=$color1;
        }

        $cont_form.="</tr>";
    }

    mysql_free_result($res);
    $cont_form.="</table>";
}

$respuesta->addAssign("datos","innerHTML",$cont_form);
return $respuesta;
}
```

```
function cargar_sesion($hora, $sesiones, $fecha, $cabina, $id)
{
    $respuesta = new xajaxResponse('ISO-8859-1');

    $sql="UPDATE sesion SET FecSes='".$fecha."', CabinaSes=".$cabina.", StatusSes='Calendarizada',
    id_HS=".$hora." WHERE id_Ses=".$sesiones."";
    mysql_query($sql);

    $respuesta->addScript("window.location='".$_SERVER['PHP_SELF']."?action=VerDetallesDeCliente&
    id=".$id."'");
    return $respuesta;
}

function nuevo_tratamiento($form_entrada)
{
    $respuesta = new xajaxResponse('ISO-8859-1');

    $numero=$form_entrada['sesiones'];

    if($numero!=0)
    {
        $cont_form='<form id="numero_tratamientos" action="">';
        for($i=0; $i<$numero; $i++)
        {
            $cont_form.='<select name="tratamiento_'.$i.'">';

            $sql="SELECT * FROM tratamiento";
            $res=mysql_query($sql);

            while($row=mysql_fetch_array($res))
            {
                $cont_form.='<option value="'.$row['id_T'].'">'.$row['NomT'].'</option>';
            }

            mysql_free_result($res);

            $cont_form.='</select>
                <input type="text" name="sesiones_'.$i.'">
                <input type="hidden" name="noses" value="'.$numero.'"><br />';
        }

        $cont_form.='<br /><br />Nombre <input type="text" name="nombre"><br /><br />
            Costo <input type="text" name="costo" value="0.00"><br /><br />
            <input type="button" class="formbutton" name="cargar_paquete" value="CARGAR"
            onclick="xajax_cargar_tratamientos(xajax.getFormValues(\'numero_tratamientos\'))">
            </form>';
    }

    $respuesta->addAssign("contenido","innerHTML",$cont_form);
    return $respuesta;
}

function cargar_tratamientos($form_entrada)
{
    $respuesta = new xajaxResponse('ISO-8859-1');
```

```
    $val=$form_entrada['noses'];
    $nom=$form_entrada['nombre'];
    $cost=$form_entrada['costo'];

    for($i=0; $i<$val; $i++)
    {
        $var1[$i]=$form_entrada["tratamiento_".$i];
        $var2[$i]=$form_entrada["sesiones_".$i];
    }

    $sql="INSERT INTO paquete (NomPac, CostNPac, StatusPac) VALUES ('".$nom."', '".$cost."', 'Cargando')";
    mysql_query($sql);

    $sql="SELECT * FROM paquete WHERE StatusPac='Cargando'";
    $res=mysql_query($sql);

    if($row=mysql_fetch_array($res))
    {
        $paquete=$row['id_Pac'];
        $sql="INSERT INTO paquete_tratamiento (NoSesPT, id_Pac, id_T) VALUES ";

        for($i=0; $i<$val; $i++)
        {
            $sql.="(".$var2[$i].",".$paquete.",".$var1[$i].")";

            if($i<($val-1))
            {
                $sql.=", ";
            }
        }

        mysql_query($sql);

        mysql_free_result($res);

        $sql="UPDATE paquete SET StatusPac='Activo' WHERE StatusPac='Cargando'";
        mysql_query($sql);
    }

    $respuesta->addScript("window.location='".$_SERVER['PHP_SELF']."?action=Paquetes'");
    return $respuesta;
}

$xajax->registerFunction("nuevo_tratamiento");
$xajax->registerFunction("cargar_tratamientos");
$xajax->registerFunction("mostrar_sesiones");
$xajax->registerFunction("cargar_sesion");
$xajax->registerFunction("formulario_tratamiento");
$xajax->registerFunction("registrar_paquete");
$xajax->processRequests();
?>
```

D.2. class.basicas.php

```
<?php
```

```
class DB
{
var $bd;
    var $serv;
    var $user;
    var $clave;
    var $Conexion_ID = 0;
    var $Errno = 0;
var $Error = "";

function conectar($bd, $host, $user, $pass)
{
if ($bd != "") $this->bd = $bd;
if ($host != "") $this->serv = $host;
if ($user != "") $this->user = $user;
if ($pass != "") $this->clave = $pass;

$this->Conexion_ID = mysql_connect($this->serv, $this->user, $this->clave);
        if (!$this->Conexion_ID)
        {
            $this->Error = "Ha fallado la conexión.";
            return 0;
        }

        if (!mysql_select_db($this->bd, $this->Conexion_ID))
        {
            $this->Error = "Imposible abrir ".$this->bd;
            return 0;
        }

        return $this->Conexion_ID;
    }
}

class basicas
{
    public function idal()
    {
        $num = rand(100000,999999);

        $sql="SELECT id_U FROM usuario";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
            if($row['id_U']!=$this->num)
            {
                return $num;
            }
            else
            {
                $idal();
            }
        }
        else
        {
```

```
            return $num;
        }
    }

    public function fecha()
    {
        $dia=date("l");
        if ($dia=="Monday") $dia="Lunes";
        if ($dia=="Tuesday") $dia="Martes";
        if ($dia=="Wednesday") $dia="Miércoles";
        if ($dia=="Thursday") $dia="Jueves";
        if ($dia=="Friday") $dia="Viernes";
        if ($dia=="Saturday") $dia="Sabado";
        if ($dia=="Sunday") $dia="Domingo";

        $dia2=date("d");

        $mes=date("F");
        if ($mes=="January") $mes="Enero";
        if ($mes=="February") $mes="Febrero";
        if ($mes=="March") $mes="Marzo";
        if ($mes=="April") $mes="Abril";
        if ($mes=="May") $mes="Mayo";
        if ($mes=="June") $mes="Junio";
        if ($mes=="July") $mes="Julio";
        if ($mes=="August") $mes="Agosto";
        if ($mes=="September") $mes="Setiembre";
        if ($mes=="October") $mes="Octubre";
        if ($mes=="November") $mes="Noviembre";
        if ($mes=="December") $mes="Diciembre";

        $anio=date("Y");

        $fecha_hoy = "$dia $dia2 de $mes de $anio";
        return $fecha_hoy;
    }
}

?>
```

D.3. class.clientes.php

```
<?php

class cliente extends usuario
{
    public function mostrarClientes()
    {
        parent::imprimirUsuarios("Cliente");
    }

    public function verDetallesCliente()
    {
        parent::verDetallesUsuario("Cliente");
    }
```

```
    public function ingresarCliente()
    {
        parent::ingresarUsuario("Cliente");
    }

    public function actualizarCliente()
    {
        parent::actualizarDatosUsuario("Cliente");
    }

    public function actualizarPassCliente()
    {
        parent::actualizarPasswordUsuario("Cliente");
    }

    public function borrarCliente()
    {
        parent::borrarUsuario("Cliente");
    }

    public function adquirirTratamiento()
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        $id=$_GET['id'];

        $sql="SELECT StatusPC FROM paquete_cliente WHERE StatusPC='En curso' AND id_C=".$id."";
        $res=mysql_query($sql);

        if(mysql_num_rows($res)==0)
        {

            $sql="SELECT id_Pac, NomPac FROM paquete WHERE StatusPac='Activo'";
            $res=mysql_query($sql);

            if($res)
            {
?>
                <form id="tratamientos" action="">
                <input type="hidden" name="id" value="<?= $id?>">
                <select name="pc" onchange="xajax_formulario_tratamiento(xajax.getFormValues
                ('tratamientos'))">
                <option>Selecciona un paquete...</option>
<?php
                while ($row=mysql_fetch_array($res))
                {
?>
                    <option value="<?= $row['id_Pac']?>"><?= $row['NomPac']?></option>
<?php
                }
```

```
                mysql_free_result($res);
            }
?>
            </select>
            </form>
            <br />
            <div id="datos" align="center"></div>
<?php
        }
        else
        {
?>
            <strong>POSEE TRATAMIENTO EN CURSO... AL CONCLUIRLO, PODR&Aacute;
            ADQUIRIR UNO NUEVO</strong>
<?php
        }
?>
        <br /><br /><a href="<?= $_SERVER['PHP_SELF']?>?action=VerDetallesDeCliente&amp;
        id=<?= $id?>">Volver</a>
        </div>
<?php

    }

    public function calendarizarSesion()
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        $id=$_GET['id'];

        $sql="SELECT * FROM sesion, paquete_cliente, cliente WHERE paquete_cliente.id_PC=sesion.id_PC
        AND cliente.id_C=paquete_cliente.id_C AND cliente.id_C=".$id." AND (StatusSes='Pendiente'
        OR StatusSes='Recalendarizada') AND StatusPC='En Curso'
        ORDER BY NoSes";
        $res=mysql_query($sql);

        if(mysql_num_rows($res)!=0)
        {
?>
            <form id="calendarizacion" action="">
            <input type="text" name="fecha_sesion" id="campo_fecha" value="<?= date("Y-m-d")?>">
            <input type="button" id="lanzador" value="...">
            <input type="hidden" name="id" value="<?= $id?>">
            <!-- script que define y configura el calendario-->
            <script type="text/javascript">
            Calendar.setup({
                inputField : "campo_fecha",
                ifFormat : "%Y-%m-%d",
                button : "lanzador"
            });
            </script>
```

```
            <br /><br />
            <select name= "numero" onchange="xajax_mostrar_sesiones(xajax.getFormValues('calendarizacion'))">
            <option>Selecciona la sesi&oacute;n...</option>
<?php
            while($row=mysql_fetch_array($res))
            {
?>
                <option value="<?=$row['id_Ses']?>"><?= $row['NoSes']?></option>
<?php
            }

            mysql_free_result($res);
?>
            </select>
            </form><br/>
            <div id="datos" align="center"></div>
<?php
        }
        else
        {
?>
            <strong>NO HAY SESIÓN POR CALENDARIZAR...</strong><br><br>
<?php
        }
?>
        <br /><a href="<? $_SERVER['PHP_SELF']?>?action=VerDetallesDeCliente&amp;id=<?= $id?>">Volver</a>
        </div>
<?php
    }

    public function registrarAvances()
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        $id=$_GET['id'];

        $sql="SELECT paquete_cliente.id_PC, NoSes, sesion.id_Ses FROM sesion, cliente, paquete_cliente,
        medidas WHERE cliente.id_C=paquete_cliente.id_C AND sesion.id_PC=paquete_cliente.id_PC AND
        sesion.id_Ses=medidas.id_Ses AND StatusSes='Actual' AND cliente.id_C=".$id."";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
?>
            <form id="medidas_sesion" action="<? $_SERVER['PHP_SELF']?>?action=VerDetallesDeCliente
            &amp;id=<?= $id?>" method="post">
            <div id="izquierdo" align="center">
            <table width="249" border="1"><tr bgcolor="#A4B4C1">
            <td width="89">Parte</td><td width="144">Medici&oacute;n </td>
            </tr><tr bgcolor="#D5E7FB" onMouseOver="this.style.backgroundColor='#E0FAC5';"
            onMouseOut="this.style.backgroundColor='#D5E7FB';"><td>Torax</td>
            <td><input type="text" name="torax" value="0.00" onKeyPress=" return
```

```
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#E8F2FC" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#E8F2FC';">
<td>Cintura</td>
<td><input type="text" name="cintura" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#D5E7FB" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#D5E7FB';">
<td>Cadera</td>
<td><input type="text" name="cadera" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#E8F2FC" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#E8F2FC';">
<td>B. Izquierdo </td>
<td><input type="text" name="brazo_iz" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#D5E7FB" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#D5E7FB';">
<td>B. Derecho </td>
<td><input type="text" name="brazo_der" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#E8F2FC" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#E8F2FC';">
<td>P. Izquierda </td>
<td><input type="text" name="pierna_iz" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#D5E7FB" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#D5E7FB';">
<td>P. Derecha </td>
<td><input type="text" name="pierna_der" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#E8F2FC" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#E8F2FC';">
<td>Peso</td>
<td><input type="text" name="peso" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#D5E7FB" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#D5E7FB';">
<td>% FAT </td>
<td><input type="text" name="fat" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#E8F2FC" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#E8F2FC';">
<td>%H2O</td>
<td><input type="text" name="h2o" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr><tr bgcolor="#D5E7FB" onMouseOver="this.style.backgroundColor='#E0FAC5';"
onMouseOut="this.style.backgroundColor='#D5E7FB';">
<td>I.M.C.</td>
<td><input type="text" name="imc" value="0.00" onKeyPress=" return
validarSoloNumeros(event)"></td>
</tr></table>
<input type="hidden" name="pc" value="<?= $row['id_PC'] ?>">
</div>

<div id="derecho" align="center">
<table width="200" border="0">
```

```
            <tr><td>Abono a tratamiento: $
            <input name="caja" type="text" value="0.00" onKeyPress="return validarSoloNumeros(event)">
            <select name="forma_pago">
            <option value="Efectivo">Efectivo</option>
            <option value="Tarjeta de credito">Tarjeta de credito</option>
            </select></td>
            <tr><td>Sesión: <?= $row['NoSes']?></td></tr>
            </table>
            <input type="hidden" name="sesion" value="<?= $row['id_Ses']?>">
            <input type="hidden" name="numero" value="<?= $row['NoSes']?>">
            <input type="hidden" name="id" value="<?= $id?>">
            <table width="200" border="0"><tr><td align="center">
            <input type="submit" class="formbutton" name="registrar_sesion" value="Aceptar">
            </td><td align="center">
            <input type="reset" class="formbutton" name="Reset" value="Cancelar"></td></tr></table>
            <br /><br />
            <a href="<? $_SERVER['PHP_SELF']?>?action=VerDetallesDeCliente&amp;id=<?= $id?>">Volver</a>
            </div>
            </form>
<?php
        }
        else
        {
?>
            <strong>NO HAY SESIÓN POR REGISTRAR...</strong><br><br>
            <a href="<? $_SERVER['PHP_SELF']?>?action=VerDetallesDeCliente&amp;id=<?= $id?>">Volver</a>
<?php
        }
?>
        </div>
<?php
    }

    public function cargarAvances()
    {
        $hoy=date("Y-m-d");
        $hora=date("H:i:s");

        $med1=$_POST['torax'];
        $med2=$_POST['cintura'];
        $med3=$_POST['cadera'];
        $med4=$_POST['brazo_iz'];
        $med5=$_POST['brazo_der'];
        $med6=$_POST['pierna_iz'];
        $med7=$_POST['pierna_der'];
        $med8=$_POST['peso'];
        $med9=$_POST['fat'];
        $med10=$_POST['h2o'];
        $med11=$_POST['imc'];

        $forma=$_POST['forma_pago'];
        $ses=$_POST['sesion'];
        $pago=$_POST['caja'];
        $paquete=$_POST['pc'];
        $num=$_POST['numero'];
        $id=$_POST['id'];
```

```
        if($pago!="0.00")
        {
            $sql="INSERT INTO caja (FechaPago, HoraPago, MontoPago, TipoPago, id_PC) VALUES
            ('".$hoy."','".$hora."','".$pago."','".$forma."','".$paquete."')";
            mysql_query($sql);
        }

        $sql="UPDATE medidas SET Torax='".$med1."', Cintura='".$med2."', Cadera='".$med3."',
        Brazo_I='".$med4."', Brazo_D='".$med5."', Pierna_I='".$med6."', Pierna_D='".$med7."',
        Peso='".$med8."', FAT='".$med9."', H2O='".$med10."', IMC='".$med11."' WHERE
        id_Ses=".$ses."";
        mysql_query($sql);

        $sql="UPDATE sesion SET StatusSes='Dada' WHERE id_Ses=".$ses."";
        mysql_query($sql);

        $sql="SELECT max(NoSesPT) AS sesion FROM paquete_tratamiento, paquete, paquete_cliente
        WHERE paquete_tratamiento.id_Pac =paquete.id_Pac AND paquete.id_Pac = paquete_cliente.id_Pac
        AND id_PC =".$paquete."";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
            if($row['sesion']==$num)
            {
                $sql="UPDATE paquete_cliente SET StatusPC='Concluido' WHERE id_PC=".$paquete."";
                mysql_query($sql);
            }
        }

        mysql_free_result($res);
    }
}
?>
```

D.4. class.paginacion.php

```
<?php

if(empty($_pagi_sql))
{
   die("<b>Error Paginator : </b>No se ha definido la variable \$_pagi_sql");
}

if(empty($_pagi_cuantos))
{
   $_pagi_cuantos = 20;
}

if(!isset($_pagi_mostrar_errores))
{
   $_pagi_mostrar_errores = true;
}

if(!isset($_pagi_conteo_alternativo))
```

```
{
   $_pagi_conteo_alternativo = false;
}

if(!isset($_pagi_separador))
{
   $_pagi_separador = " ";
}

if(isset($_pagi_nav_estilo))
{
   $_pagi_nav_estilo_mod = "class=\"$_pagi_nav_estilo\"";
}
else
{
   $_pagi_nav_estilo_mod = "";
}

if(!isset($_pagi_nav_anterior))
{
   $_pagi_nav_anterior = "Anterior";
}

if(!isset($_pagi_nav_siguiente))
{
   $_pagi_nav_siguiente = "Siguiente";
}

if(!isset($_pagi_nav_primera))
{
   $_pagi_nav_primera = "|&lt;";
}

if(!isset($_pagi_nav_ultima))
{
   $_pagi_nav_ultima = "&gt;|";
}

if (empty($_GET['_pagi_pg']))
{
   $_pagi_actual = 1;
}
else
{
   $_pagi_actual = $_GET['_pagi_pg'];
}

if($_pagi_conteo_alternativo == false)
{
   $_pagi_sqlConta = eregi_replace("select[[:space:]](.*)[[:space:]]from", "SELECT COUNT(*) FROM",
   $_pagi_sql);
   $_pagi_result2 = mysql_query($_pagi_sqlConta);
   if($_pagi_result2 == false && $_pagi_mostrar_errores == true)
   {
       die (" Error en la consulta de conteo de registros: $_pagi_sqlConta.
       Mysql dijo: <b>".mysql_error()."</b>");
   }
```

```
   $_pagi_totalReg = mysql_result($_pagi_result2,0,0);
}
else
{
   $_pagi_result3 = mysql_query($_pagi_sql);
   if($_pagi_result3 == false && $_pagi_mostrar_errores == true)
   {
       die (" Error en la consulta de conteo alternativo de registros: $_pagi_sql. Mysql dijo:
       <b>".mysql_error()."</b>");
   }
   $_pagi_totalReg = mysql_num_rows($_pagi_result3);
}

$_pagi_totalPags = ceil($_pagi_totalReg / $_pagi_cuantos);

$_pagi_enlace = $_SERVER['PHP_SELF'];
$_pagi_query_string = "?";

if(!isset($_pagi_propagar))
{
  if (isset($_GET['_pagi_pg']))
  {
    unset($_GET['_pagi_pg']);
  }

  $_pagi_propagar = array_keys($_GET);
}
elseif(!is_array($_pagi_propagar))
{
   die("<b>Error Paginator : </b>La variable \$_pagi_propagar debe ser un array");
}

foreach($_pagi_propagar as $var)
{
    if(isset($GLOBALS[$var]))
    {
        $_pagi_query_string.= $var."=".$GLOBALS[$var]."&amp;";
    }
    elseif(isset($_REQUEST[$var]))
    {
        $_pagi_query_string.= $var."=".$_REQUEST[$var]."&amp;";
    }
}

$_pagi_enlace .= $_pagi_query_string;

$_pagi_navegacion_temporal = array();

if ($_pagi_actual != 1)
{
    $_pagi_url = 1;
    $_pagi_navegacion_temporal[] = "<li><a class='prevnext' href='".$_pagi_enlace."_pagi_pg=
    ".$_pagi_url."'>$_pagi_nav_primera</a></li>";
    $_pagi_url = $_pagi_actual - 1;
    $_pagi_navegacion_temporal[] = "<li><a class='prevnext' href='".$_pagi_enlace."_pagi_pg=
    ".$_pagi_url."'>$_pagi_nav_anterior</a></li>";
}
```

```
if(!isset($_pagi_nav_num_enlaces))
{
    $_pagi_nav_desde = 1;
    $_pagi_nav_hasta = $_pagi_totalPags;
}
else
{
    $_pagi_nav_intervalo = ceil($_pagi_nav_num_enlaces/2) - 1;
    $_pagi_nav_desde = $_pagi_actual - $_pagi_nav_intervalo;
    $_pagi_nav_hasta = $_pagi_actual + $_pagi_nav_intervalo;

    if($_pagi_nav_desde < 1)
    {
        $_pagi_nav_hasta -= ($_pagi_nav_desde - 1);
        $_pagi_nav_desde = 1;
    }

    if($_pagi_nav_hasta > $_pagi_totalPags)
    {
        $_pagi_nav_desde -= ($_pagi_nav_hasta - $_pagi_totalPags);
        $_pagi_nav_hasta = $_pagi_totalPags;
        if($_pagi_nav_desde < 1)
        {
            $_pagi_nav_desde = 1;
        }
    }
}

for ($_pagi_i = $_pagi_nav_desde; $_pagi_i<=$_pagi_nav_hasta; $_pagi_i++)
{
    if ($_pagi_i == $_pagi_actual)
    {
        $_pagi_navegacion_temporal[] = "<li><a class='currentpage'>$_pagi_i</a></li>";
    }
    else
    {
        $_pagi_navegacion_temporal[] = "<li><a href='".$_pagi_enlace."_pagi_pg=".$_pagi_i."'>
        ".$_pagi_i."</a></li>";
    }
}

if ($_pagi_actual < $_pagi_totalPags)
{
    $_pagi_url = $_pagi_actual + 1;
    $_pagi_navegacion_temporal[] = "<li><a class='prevnext' href='".$_pagi_enlace."_pagi_pg=
    ".$_pagi_url."'>$_pagi_nav_siguiente</a></li>";
    $_pagi_url = $_pagi_totalPags;
    $_pagi_navegacion_temporal[] = "<li><a class='prevnext' href='".$_pagi_enlace."_pagi_pg=
    ".$_pagi_url."'>$_pagi_nav_ultima</a></li>";
}

$_pagi_navegacion = implode($_pagi_separador, $_pagi_navegacion_temporal);
$_pagi_inicial = ($_pagi_actual-1) * $_pagi_cuantos;
$_pagi_sqlLim = $_pagi_sql." LIMIT $_pagi_inicial,$_pagi_cuantos";
$_pagi_result = mysql_query($_pagi_sqlLim);
```

```
if($_pagi_result == false && $_pagi_mostrar_errores == true)
{
   die ("Error en la consulta limitada: $_pagi_sqlLim. Mysql dijo: <b>".mysql_error()."</b>");
}

$_pagi_desde = $_pagi_inicial + 1;
$_pagi_hasta = $_pagi_inicial + $_pagi_cuantos;

if($_pagi_hasta > $_pagi_totalReg)
{
   $_pagi_hasta = $_pagi_totalReg;
}

$_pagi_info = "desde el $_pagi_desde hasta el $_pagi_hasta de un total de $_pagi_totalReg";
?>
```

D.5. class.paquetes.php

```
<?php

class paquete
{
    var $id;
    var $nombre;
    var $costo;
    var $status;

    public function __construct()
    {
        $this->id=0;
        $this->nombre="";
        $this->costo=0;
    }

    public function actualizarDatosPaquete()
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        $this->id=$_GET['id'];

        $sql="SELECT * FROM paquete WHERE id_Pac=".$this->id."";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
?>
            <form action="<?= $_SERVER['PHP_SELF']?>?action=Tratamientos" method="post">
            <table width="331" border="0">
            <tr>
            <td width="148"><strong>Nombre de paquete:</strong></td>
```

```
            <td width="167">
            <input type="text" name="nom_tratamiento" value="<?= $row['NomPac']?>" size="25" /></td>
            </tr><tr>
            <td><strong>Costo de paquete: </strong></td>
            <td><input type="text" name="cost_tratamiento" value="<?= $row['CostNPac']?>" size="15"/></td>
            </tr></table>
            <br />
            <strong>Tratamientos del paquete</strong><br /><br /><table width="200" border="0">
<?php
            mysql_free_result($res);

            $sql="SELECT * FROM paquete, paquete_tratamiento, tratamiento WHERE
            paquete.id_Pac=paquete_tratamiento.id_Pac AND paquete_tratamiento.id_T=tratamiento.id_T
            AND paquete.id_Pac=".$this->id."";
            $res=mysql_query($sql);
            $i=0;

            while($row=mysql_fetch_array($res))
            {
?>
                <tr><td><strong><?= $row['NomT']?></strong></td>
                <td><input type="text" name="ses_tratamiento_<?= $i?>"
                value="<?= $row['NoSesPT']?>" size="5"></td></tr>
                <tr><td><input type="hidden" name="ses_original_<?= $i?>"
                value="<?= $row['NoSesPT']?>"></td></tr>
<?php
                $i++;
            }

            mysql_free_result($res);
?>
            </table><br />
            <input type="hidden" name="no" value="<?= $i?>">
            <input type="hidden" name="id" value="<?= $this->id?>">
            <table width="240" border="0"><tr>
            <td width="112">
            <input type="submit" class="boton" name="actualizar_paquete" value="Aceptar">
            </td><td width="112">
            <input type="reset" class="boton" name="cancelar" value="Cancelar"></td></tr>
            </table></form>
            </div>
<?php
        }
    }

    public function cargarActualizacionPaquete()
    {
        $this->id=$_POST['id'];
        $this->nombre=$_POST['nom_tratamiento'];
        $this->costo=$_POST['cost_tratamiento'];
        $limit=$_POST['no'];

        for($i=0; $i<$limit; $i++)
        {
            $sesion[$i]=$_POST["ses_tratamiento_".$i];
            $original[$i]=$_POST["ses_original_".$i];
        }
```

```
        $sql="UPDATE paquete SET NomPac='".$this->nombre."', CostNPac='".$this->costo."' WHERE
        id_Pac=".$this->id."";
        mysql_query($sql);

        for($i=0; $i<$limit; $i++)
        {
            $sql="UPDATE paquete_tratamiento SET NoSesPT=".$sesion[$i]." WHERE id_Pac=".$this->id."
            AND NoSesPT=".$original[$i]."";
            mysql_query($sql);
        }
?>
        <script type="text/javascript">
            window.alert('Datos de paquete actualizados exitosamente');
        </script>
<?php
    }

    public function cambiarEstadoPaquete($estado)
    {
        $this->id=$_GET['id'];

        $sql="UPDATE paquete SET ";

        if($estado=="Activo")
        {
            $sql.="StatusPac='No activo' ";
        }
        elseif($estado=="No activo")
        {
            $sql.="StatusPac='Activo' ";
        }

        $sql.="WHERE id_Pac=".$this->id."";
        mysql_query($sql);
    }

    public function cargarPaquete()
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);
?>
        <form id="paquetes" action="">
        Numero de tratamientos
        <br /><select name="sesiones" onchange="xajax_nuevo_tratamiento(xajax.getFormValues('paquetes'))">
        <option>Elige numero de sesiones...</option>
        <option value="1">1</option>
        <option value="2">2</option>
        <option value="3">3</option>
        <option value="4">4</option>
        </select>
        </form>
        <br />
```

```
        <div id="contenido" align="center"></div>
        <br /><a href="<? $_SERVER['PHP_SELF']?>?action=Paquetes">Volver</a>
        </div>
<?php
    }

    public function mostrarPaquetes()
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        $_pagi_sql="SELECT * FROM paquete ORDER BY id_Pac";
        $_pagi_cuantos = 11;
        $_pagi_nav_num_enlaces=8;

        include("class.paginacion.php");

        if(mysql_num_rows($_pagi_result)!=0)
        {
            $color0="#A4B4C1";
            $color1="#D5E7FB";
            $color2="#E8F2FC";
            $color3="#E0FAC5";
            $color=$color1;

?>
            <table border="1"><tr bgcolor="<?= $color0 ?>" align="center">
            <td><strong>NOMBRE DE PAQUETE</strong></td>
            <td><strong>COSTO</strong></td>
            <td><strong>ESTADO</strong></td>
            <td><strong>ACTUALIZAR</strong></td></tr>
<?php
            while($row = mysql_fetch_array($_pagi_result))
            {
                $this->id=$row['id_Pac'];
?>
                <tr align="center" bgcolor="<?= $color ?>" onMouseOver="this.style.backgroundColor=
                '<?= $color3 ?>';" onMouseOut="this.style.backgroundColor='<?= $color ?>';">
                <td><?= $row['NomPac'] ?></td>
                <td>$<?=$row['CostNPac'] ?>.00</td>
<?php
                if($row['StatusPac']=="Activo")
                {
?>
                    <td width="100"><a href="<?= $_SERVER['PHP_SELF']?>?action=DesactivarPaquete&amp;
                    id=<?= $this->id?>"><img src="imagenes/palomita.png" alt="Da click AQUI para
                    desactivarlo" width="15" height="15" border="0"></a></td>
<?php
                }
                else
                {
?>
                    <td width="100"><a href="<?= $_SERVER['PHP_SELF']?>?action=ActivarPaquete&amp;
```

```
                    id=<?= $this->id?>"><img src="imagenes/paloma.gif" alt="Da click AQUI para
                    activarlo" width="15" height="15" border="0"></a></td>
<?php
                }
?>
                <td width="100"><a href="<?= $_SERVER['PHP_SELF']?>?action=ActualizarPaquete&amp;
                id=<?= $this->id?>"><img src="imagenes/lapiz.png" alt="Da click AQUI para actualizar
                datos del paquete" width="15" height="15" border="0"></a></td>
<?php
                if($color==$color1)
                {
                    $color=$color2;
                }
                else
                {
                    $color=$color1;
                }
?>
                </tr>
<?php
            }
?>
            </table>
<?php
        }
        else
        {
            echo "No hay resultados";
        }
?>
        </div>
        <div id="paginador">
        <br/><br/><br/>
        <table border="0" align="center"><tr><td width="400">
        <div class="paginacion" align="center"><ul><?= $_pagi_navegacion ?></ul></div></td>
        <td><a href="<?= $_SERVER['PHP_SELF']?>?action=CargarPaquete">
        <b>INGRESAR PAQUETE</b></a></td>
        </tr></table>
        </div>
<?php
    }
}
?>
```

D.6. class.personal.php

```
<?php

class personal extends usuario
{
    public function actualizarPersonal()
    {
        parent::actualizarDatosUsuario("Personal");
    }
```

```
    public function actualizarPassPersonal()
    {
        parent::actualizarPasswordUsuario("Personal");
    }

    public function borrarPersonal()
    {
        parent::borrarUsuario("Personal");
    }

    public function mostrarPersonal()
    {
        parent::imprimirUsuarios("Personal");
    }

    public function verDetallesPersonal()
    {
        parent::verDetallesUsuario("Personal");
    }

    public function ingresarPersonal()
    {
        parent::ingresarUsuario("Personal");
    }
}
?>
```

D.7. class.usuarios.php

```
<?php

class usuario
{
    var $id;
    var $nom;
    var $pat;
    var $mat;
    var $dirr;
    var $telc;
    var $telm;
    var $pass1;
    var $pass2;
    var $dato;

    public function __construct()
    {
        $this->id=0;
        $this->nom="";
        $this->pat="";
        $this->mat="";
        $this->dirr="";
        $this->telc=0;
        $this->telm=0;
        $this->pass1="";
        $this->pass2="";
```

```
        $this->dato=0;
    }

    public function actualizarDatosUsuario($tipo)
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        $this->id=$_GET['id'];

        if(($tipo=="Personal")&&($_SESSION['num']!=$this->id))
        {
?>
            <strong>NO PUEDES CAMBIAR LOS DATOS DE OTRO MIEMBRO DEL PERSONAL</strong>
<?php
        }
        else
        {
            if($tipo=="Cliente")
            {
                $usuario=", cliente";
                $condicion="id_C=id_U";
            }
            elseif($tipo=="Personal")
            {
                $usuario=", personal";
                $condicion="id_P=id_U";
            }

            $sql="SELECT * FROM usuario".$usuario." WHERE id_U=".$this->id." AND ".$condicion."";
            $res=mysql_query($sql);

            if($row=mysql_fetch_array($res))
            {
?>
                <form action="<? $_SERVER['PHP_SELF']?>?action=VerDetallesDe<?= $tipo?>&amp;id=
                <?= $this->id?>" method="post">
                <table width="337" border="0" align="center"><tr>
                <td><strong>NOMBRE(S)</strong></td>
                <td><input type="text" name="nom_usuario" value="<?= $row['NomU']?>"
                maxlength="20" onKeyPress="return validarSoloTexto(event)"></td>
                </tr><tr>
                <td><strong>APELLIDO PATERNO </strong></td>
                <td><input type="text" name="ap_pat_usuario" value="<?= $row['ApPatU']?>"
                maxlength="20" onKeyPress="return validarSoloTexto(event)"></td>
                </tr><tr>
                <td><strong>APELLIDO MATERNO </strong></td>
                <td><input type="text" name="ap_mat_usuario" value="<?= $row['ApMatU']?>"
                maxlength="20" onKeyPress="return validarSoloTexto(event)"></td>
                </tr><tr>
                <td><strong>DIRECCI&Oacute;N</strong></td>
                <td><input type="text" name="dir_usuario" value="<?= $row['DirU']?>"
                maxlength="50" onKeyPress="return invalidarAcento(event)"></td>
```

```
                </tr><tr>
                <td><strong>TEL&Eacute;FONO (CASA) </strong></td>
                <td><input type="text" name="tel_c_usuario" value="<?= $row['TelCU']?>"
                maxlength="8" onKeyPress="return validarSoloNumeros(event)"></td>
                </tr><tr>
                <td><strong>TEL&Eacute;FONO (M&Oacute;VIL) </strong></td>
                <td><input type="text" name="tel_m_usuario" value="<?= $row['TelMU']?>"
                maxlength="10" onKeyPress="return validarSoloNumeros(event)"></td>
                </tr><tr>
                <td><input type="hidden" name="id" value="<?=$this->id?>"></td>
                </tr>
<?php
                if($tipo=="Cliente")
                {
?>                  <tr><td><strong>TALLA </strong></td>
                    <td><input type="text" name="talla_usuario" value="<?= $row["AlturaCliente"]?>"
                    maxlength="4" onKeyPress="return validarSoloNumeros(event)">
                    </td></tr>
<?php
                    mysql_free_result($res);
                }
                elseif($tipo=="Personal"&&($_SESSION['puesto']=="Administrador")||
                ($_SESSION['puesto']=="Gerente"))
                {
?>
                    <tr><td><strong>PUESTO</strong></td>
                    <td><select name="puesto_personal">
<?php
                    $sql="SELECT * FROM Puesto";
                    $res=mysql_query($sql);

                    while($row=mysql_fetch_array($res))
                    {
?>
                        <option value="<?= $row['id_Pu']?>"><?= $row['NomPu']?></option>
<?php
                    }

                    mysql_free_result($res);
?>
                    </select></td></tr>
<?php
                }
?>
                </table>
                <br />
                <table width="277" border="0" align="center"><tr>
                <td align="center">
                <input type="submit" class="boton" name="actualizar_<?= $tipo?>" value="  Aceptar  "
                onClick="return validarCliente(this.form)">
                </td>
                <td>
                <input type="reset" class="boton" name="cancelar" value=" Cancelar ">
                </td></tr>
                </table></form>
<?php
            }
```

```
        }
?>
        <br/><br /><a href="<?= $_SERVER['PHP_SELF']?>?action=VerDetallesDe<?= $tipo?>&amp;
        id=<?= $this->id?>">Volver</a>
        </div>
<?php

    }

    public function actualizarPasswordUsuario($tipo)
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        $this->id=$_GET['id'];

        if($_SESSION['num']!=$this->id)
        {
?>
            <strong>NO PUEDES CAMBIAR EL PASSWORD DE OTRO USUARIO</strong>
<?php
        }
        else
        {
?>
            <form action="<? $_SERVER['PHP_SELF']?>?action=VerDetallesDe<?= $tipo?>&amp;id=
            <?= $this->id?>" method="post">
            <div align="center"><table width="294" border="0"><tr>
            <td width="135">Contrase&ntilde;a actual </td>
            <td width="149"><input type="password" name="contraseña_usuario" maxlength="15"
            onClick="return invalidarAcentos(event)"></td>
            </tr><tr>
            <td>Nueva contrase&ntilde;a </td>
            <td><input type="password" name="confirmar_usuario1" maxlength="15"
            onClick="return invalidarAcentos(event)"></td>
            </tr><tr>
            <td>Confirmar contrase&ntilde;a </td>
            <td><input type="password" name="confirmar_usuario2" maxlength="15"
            onClick="return invalidarAcentos(event)"></td>
            </tr><tr>
            <td><input type="hidden" name="id" value="<?= $this->id?>"></td>
            </tr></table>
            <br />
            <table width="200" border="0">
            <tr><td width="89" height="47"><div align="center">
            <input type="submit" class="boton" name="cambiar_pass_<?= $tipo?>" value="Aceptar"
            onClick="return validarContraseña(this.form)">
            </div></td>
            <td width="95"><div align="center">
            <input type="reset" class="boton" name="cancelar" value="Cancelar">
            </div></td></tr></table></div></form>
<?php
        }
```

```
?>
        <br /><br /><a href="<?= $_SERVER['PHP_SELF']?>?action=VerDetallesDe<?= $tipo?>&amp;id=
        <?= $this->id?>">Volver</a>
        </div>
<?php
    }

    public function borrarUsuario($tipo)
    {
        $this->id=$_GET['id'];

        if($_SESSION['puesto']=="Administrador"||$_SESSION['puesto']=="Gerente")
        {
            $sql="DELETE FROM usuario WHERE id_U = ".$this->id."";
            mysql_query($sql);

            if($tipo=="Cliente")
            {
                $sql="DELETE FROM cliente WHERE id_C = ".$this->id."";
                mysql_query($sql);
            }
            elseif($tipo=="Personal")
            {
                $sql="DELETE FROM personal WHERE id_P = ".$this->id."";
                mysql_query($sql);
            }
?>
            <script type="text/javascript">
                window.alert('Usuario dado de baja exitosamente...')
            </script>
            <meta http-equiv="refresh" content="0; url=<?= $_SERVER['PHP_SELF']?>?
            action=<?= $tipo?>">
<?php
        }
        else
        {
?>
            <strong>.:. NO TIENE PRIVILEGIOS PARA EJECUTAR ESTA ACCION .:.</strong>
            <br/><br/><br/>
            <a href="<?= $_SERVER['PHP_SELF']?>?action=VerDetallesDe<?= $tipo?>&amp;
            id=<?= $this->id?>">Volver</a>
            </div>
<?php
        }
    }

    public function ingresarUsuario($tipo)
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

         if($tipo=="Cliente"||($tipo=="Personal"&&($_SESSION['puesto']=="Administrador"||
         $_SESSION['puesto']=="Gerente")))
```

```
        {
?>
            <form id="ingreso_usuario" action="<?= $_SERVER['PHP_SELF']?>?action=<?= $tipo?>"
            method="post"><table width="337" border="0" align="center"><tr>
            <td><strong>NOMBRE(S)</strong></td>
            <td><input type="text" name="nom_usuario" maxlength="20" onKeyPress="return
            validarSoloTexto(event)"></td>
            </tr><tr>
            <td><strong>APELLIDO PATERNO </strong></td>
            <td><input type="text" name="ap_pat_usuario" maxlength="20" onKeyPress="return
            validarSoloTexto(event)"></td>
            </tr><tr>
            <td><strong>APELLIDO MATERNO </strong></td>
            <td><input type="text" name="ap_mat_usuario" maxlength="20" onKeyPress="return
            validarSoloTexto(event)"></td>
            </tr><tr>
            <td><strong>DIRECCI&Oacute;N</strong></td>
            <td><input type="text" name="dir_usuario" maxlength="50" onKeyPress="return
            invalidarAcento(event)"></td>
            </tr><tr>
            <td><strong>TEL&Eacute;FONO (CASA) </strong></td>
            <td><input type="text" name="tel_c_usuario" maxlength="8" onKeyPress="return
            validarSoloNumeros(event)"></td>
            </tr><tr>
            <td><strong>TEL&Eacute;FONO (M&Oacute;VIL) </strong></td>
            <td><input type="text" name="tel_m_usuario" maxlength="10" onKeyPress="return
            validarSoloNumeros(event)"></td>
            </tr><tr>
            <td><strong>CONTRASEÑA</strong></td>
            <td><input type="password" name="pass1_usuario" maxlength="15" onKeyPress="return
            invalidarAcento(event)"></td>
            </tr><tr>
            <td><strong>CONFIRMAR CONTRASEÑA</strong></td>
            <td><input type="password" name="pass2_usuario" maxlength="15" onKeyPress="return
            invalidarAcento(event)"></td>
            </tr>
<?php
            if($tipo=="Cliente")
            {
?>
                <tr><td><strong>TALLA </strong></td>
                <td><input type="text" name="talla_usuario" maxlength="4" onKeyPress="return
                validarSoloNumeros(event)">
                </td></tr>
<?php
            }
            elseif($tipo=="Personal")
            {
?>
                <tr><td><strong>PUESTO</strong></td>
                <td><select name="puesto_personal">
<?php
                $sql="SELECT * FROM Puesto";
                $res=mysql_query($sql);

                while($row=mysql_fetch_array($res))
                {
```

```
?>
                    <option value="<?= $row['id_Pu']?>"><?= $row['NomPu']?></option>
<?php
                }

                mysql_free_result($res);
?>
                </select></td></tr>
<?php
            }
?>
            </table>
            <br />
            <table width="277" border="0" align="center"><tr>
            <td align="center">
            <input type="submit" class="boton" name="ingresar_<?= $tipo?>" value="  Aceptar  "
            onClick="return validarCliente(this.form)">
            </td>
            <td>
            <input type="reset" class="boton" name="cancelar" value=" Cancelar ">
            </td></tr>
            </table></form>
<?php
        }
        else
        {
?>
            <strong>NO PUEDE DAR DE ALTA...</strong>
<?php
        }
?>
        <br /><br><a href="<?= $_SERVER['PHP_SELF']?>?action=<?= $tipo?>" >Volver</a>
        </div>
<?php
    }

    public function cargarUsuario($tipo)
    {
        $operacion=new basicas;

        $this->id = $operacion->idal();
        $this->nom=trim($_POST['nom_usuario']);
        $this->pat=trim($_POST['ap_pat_usuario']);
        $this->mat=trim($_POST['ap_mat_usuario']);
        $this->dirr=trim($_POST['dir_usuario']);
        $this->telc=trim($_POST['tel_c_usuario']);
        $this->telm=trim($_POST['tel_m_usuario']);
        $this->pass1=trim($_POST['pass1_usuario']);
        $this->pass2=trim($_POST['pass2_usuario']);

        $password=md5($this->pass1);

        $sql="INSERT INTO usuario VALUES ('".$this->id."','".$this->nom."','".$this->pat."',
        '".$this->mat."','".$this->dirr."','".$this->telc."',
        '".$this->telm."','".$password."',0,0)";
        mysql_query($sql);
```

```
        if($tipo=="Cliente")
        {
            $this->dato=trim($_POST['talla_usuario']);
            $fecha=date("Y-m-d");

            $sql="INSERT INTO cliente VALUES ('".$this->id."','".$this->dato."','".$fecha."',
            ".$_SESSION['num'].")";
            mysql_query($sql);

            $sql="SELECT NoIngresP FROM personal WHERE id_P=".$_SESSION['num']."";
            $res=mysql_query($sql);

            if($row=mysql_fetch_array($res))
            {
                mysql_free_result($res);
                $sql="UPDATE personal SET NoIngresP=".($row['NoIngresP']+1)." WHERE
                id_P=".$_SESSION['num']."";
                mysql_query($sql);
?>
                <script type="text/javascript">window.alert("Cliente ingresado exitosamente");</script>
<?php
            }
        }
        elseif($tipo=="Personal")
        {
            $this->dato=trim($_POST['puesto_personal']);
            $sql="INSERT INTO personal VALUES ('".$this->id."', 0, '".$_SESSION['suc']."',
            '".$this->dato."')";
            mysql_query($sql);
?>
            <script type="text/javascript">
                window.alert("Personal ingresado exitosamente");
            </script>
<?php
        }
    }

    public function cambiarPassword($tipo)
    {
        $this->id=$_POST['id'];
        $cont_usuario=$_POST['contraseña_usuario'];
        $confirm_pass1=$_POST['confirmar_usuario1'];
        $confirm_pass2=$_POST['confirmar_usuario2'];

        $sql="SELECT ContU FROM usuario WHERE id_U=".$this->id."";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
            $pass=md5($cont_usuario);
            if($row['ContU']==$pass)
            {
                $pass=md5($confirm_pass1);
                mysql_free_result($res);

                $sql="UPDATE usuario SET ContU='".$pass."' WHERE id_U=".$this->id."";
                mysql_query($sql);
```

```
                if($res)
                {
?>
                    <script type="text/javascript">
                        window.alert('Contraseña actualizada');
                    </script>
                    <meta http-equiv="refresh" content="0;url="<?= $_SERVER['PHP_SELF']?>"?
                    action=VerDetallesDe<?= $tipo?>&amp;id=<?= $this->id?>">
<?php
                }
            }
        }
        else
        {
?>
            <script type="text/javascript">
                window.alert('Contraseña de usuario inválida');
            </script>
<?php
        }
    }

    public function actualizarUsuario($tipo)
    {
        $this->id=trim($_POST['id']);
        $this->nom=trim($_POST['nom_usuario']);
        $this->pat=trim($_POST['ap_pat_usuario']);
        $this->mat=trim($_POST['ap_mat_usuario']);
        $this->dirr=trim($_POST['dir_usuario']);
        $this->telc=trim($_POST['tel_c_usuario']);
        $this->telm=trim($_POST['tel_m_usuario']);

        if($tipo=="Cliente")
        {
            $this->dato=$_POST['talla_usuario'];
        }
        else
        {
            $this->dato=$_POST['puesto_personal'];
        }

        $sql="UPDATE usuario SET NomU='".$this->nom."', ApPatU='".$this->pat."', ApMatU=
        '".$this->mat."', DirU='".$this->dirr."', TelCU='".$this->telc."', TelMU=
        '".$this->telm."' WHERE id_U='".$this->id."'";
        mysql_query($sql);

        if($tipo=="Cliente")
        {
            $sql="UPDATE cliente SET AlturaCliente='".$this->dato."'";
            mysql_query($sql);
        }
        elseif ($tipo=="Personal")
        {
            $sql="UPDATE personal SET id_Pu='".$this->dato."'";
            mysql_query($sql);
        }
```

```
?>
        <script type="text/javascript">
            window.alert("Datos de usuario actualizados exitosamente");
        </script>
<?php
    }

    public function imprimirUsuarios($tipo)
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        if($tipo=="Cliente")
        {
            $usuario="Cliente,";
            $criterio = " WHERE id_P= PersonalQueIngreso AND Cliente.id_C=Usuario.id_U";

            if($_SESSION['puesto']!="Administrador")
            {
                $criterio.=" AND Personal.id_S=".$_SESSION['suc']."";
            }
        }

        if($tipo=="Personal")
        {
            $criterio = ",Puesto WHERE Personal.id_Pu= Puesto.id_Pu AND id_P=id_U";
            if($_SESSION['puesto']!="Administrador")
            {
                $criterio.=" AND Personal.id_S=".$_SESSION['suc']."";
            }

            switch($_SESSION['puesto'])
            {
                case "Gerente":
                    $condicion = "  AND NomPu != 'Gerente' AND NomPu != 'Administrador'";  break;
                case "Supervisor":
                    $condicion = "  AND NomPu != 'Gerente' AND NomPu != 'Supervisor' AND
                    NomPu != 'Administrador'";  break;
            }

            $criterio.=$condicion;
        }

        $txt_criterio = "";
        if ($_POST['buscar_usuario']!="")
        {
            $txt_criterio = $_POST['buscar_usuario'];
            $criterio.= " AND id_U like '%".$txt_criterio."%'";
        }

        $_pagi_sql="SELECT id_U,NomU, ApPatU, ApMatU FROM ".$usuario." Personal, Usuario".$criterio."
        ORDER BY id_U";
        $_pagi_cuantos = 10;
```

```
        $_pagi_nav_num_enlaces=8;

        include("class.paginacion.php");

        if(mysql_num_rows($_pagi_result)!=0)
        {
            $color0="#A4B4C1";
            $color1="#D5E7FB";
            $color2="#E8F2FC";
            $color3="#E0FAC5";
            $color=$color1;
?>
            <table border="1"><tr align="center" bgcolor="<?= $color0 ?>">
            <td width="100"><strong><font size="1">NO. DE CUENTA</font></strong></td>
            <td width="100"><strong><font size="2">NOMBRE(S)</font></strong></td>
            <td width="120"><strong><font size="2">AP. PATERNO</font></strong></td>
            <td width="120"><strong><font size="2">AP. MATERNO</font></strong></td>
            <td width="130"><strong><font size="2">MAS DETALLES</font></strong></td></tr>
<?php

            while($row = mysql_fetch_array($_pagi_result))
            {
                $usuario=$row['id_U'];
?>
                <tr align="center" bgcolor="<?= $color ?>" onMouseOver="this.style.backgroundColor=
                '<?= $color3 ?>';" onMouseOut="this.style.backgroundColor='<?= $color ?>';">
                <td width="100"><?= $row['id_U'] ?></td>
                <td width="100"><?= $row['NomU'] ?></td>
                <td width="120"><?= $row['ApPatU'] ?></td>
                <td width="120"><?= $row['ApMatU'] ?></td>
<?php
                if($tipo=="Cliente")
                {
?>
                    <td width="130"><a href="<?= $_SERVER['PHP_SELF']?>?action=VerDetallesDeCliente
                    &amp;id=<?= $usuario?>"><img src="imagenes/lupa.png" alt="Muestra historial del
                    cliente" width="20" height="15" border="0"></a>

<?php
                }
                elseif($tipo=="Personal")
                {
?>
                    <td width="130"><a href="<?= $_SERVER['PHP_SELF']?>?action=VerDetallesDePersonal
                    &amp;id=<?= $usuario?>"><img src="imagenes/lupa.png" alt="Muestra datos del personal
                    en cuestion" width="15" height="15" border="0"></a>
<?php
                }
?>
                </td></tr>
<?php
                if($color==$color1)
                {
                    $color=$color2;
                }
                else
                {
```

```
                    $color=$color1;
                }
            }
?>
            </table>

            </div>
            <div id="paginador">
            <div class="paginacion" align="center">
            <br/><center><ul><?= $_pagi_navegacion ?></ul></center>
            </div>
            <br /><table border="0" align="center"><tr><td width="400">
            <form method="post" action="<?= $_SERVER['PHP_SELF']?>?action=<?=$tipo?>">
            Ingresa c&oacute;digo de usuario
            <input type="text" name="buscar_usuario" size="10" maxlength="6">
            <input name="submit" class="boton" type="submit" value="Buscar"
            title="Presiona el botón para ver todos los usuarios de la sucursal">
            </form></td>
            <td width="190">
<?php
            if($tipo=="Cliente")
            {
?>
                <a href="<?= $_SERVER['PHP_SELF']?>?action=IngresarCliente">
                <b>INGRESAR CLIENTE</b></a>
<?php
            }

            if($tipo=="Personal")
            {
?>
                <a href="<?= $_SERVER['PHP_SELF']?>?action=IngresarPersonal">
                <b>INGRESAR PERSONAL</b></a>
<?php
            }
?>
            </td></tr>
            </table>
<?php
        }
        else
        {
?>
            <strong>NO HAY USUARIOS EN ESTA CATEGOR&Iacute;A</strong>
<?php
        }
?>
        </div>
<?php
    }

    public function verDetallesUsuario($tipo)
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
```

```
        id_U=".$_SESSION['num']."";
        mysql_query($sql);
?>
        <div id="izquierdo1" align="left">
<?php
        $this->id=$_GET['id'];

        if($tipo=="Cliente")
        {
            $sentencia=", cliente";
            $condicion=" AND id_C=id_U";
        }
        elseif($tipo=="Personal")
        {
            $sentencia=", personal, puesto, sucursal";
            $condicion=" AND personal.id_Pu=puesto.id_Pu AND personal.id_S=sucursal.id_S";
        }

        $sql="SELECT * FROM usuario".$sentencia." WHERE id_U=".$this->id.$condicion."";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
            $color0="#A4B4C1";
            $color1="#D5E7FB";
            $color2="#E8F2FC";
            $color3="#E0FAC5";
            $color=$color1;
?>
            <table border="0">
            <tr>
            <td><strong>NO. DE CUENTA</strong></td>
            <td><?= $row["id_U"] ?></td></tr>
            <tr><td><strong>NOMBRE(S)</strong></td>
            <td><?= $row["NomU"] ?></td></tr>
            <tr>
            <td><strong>APELLIDO PATERNO</strong></td>
            <td><?= $row["ApPatU"] ?></td></tr>
            <tr>
            <td><strong>APELLIDO MATERNO</strong></td>
            <td><?= $row["ApMatU"] ?></td></tr>
            <tr>
            <td><strong>DIRECCI&Oacute;N</strong></td>
            <td><?= $row["DirU"] ?></td></tr>
            <tr>
            <td><strong>TEL&Eacute;FONO (CASA)</strong></td>
            <td><?= $row["TelCU"] ?></td></tr>
            <tr>
            <td><strong>TEL&Eacute;FONO (M&Oacute;VIL)</strong></td>
            <td><?= $row["TelMU"] ?></td></tr>
            <tr>
<?php
            if($tipo=="Cliente")
            {
?>
                <td><strong>TALLA</strong></td>
                <td><?= $row["AlturaCliente"]?> m</td></tr>
```

```
                <tr>
                <td><strong>REGISTR&Oacute;</strong></td>
<?php

                $sql="SELECT NomU, ApPatU, ApMatU FROM personal, usuario WHERE
                id_P=".$row['PersonalQueIngreso']." AND id_U=id_P";
                $res=mysql_query($sql);

                if($row=mysql_fetch_array($res))
                {
?>
                    <td><?= $row["NomU"]." ".$row["ApPatU"]." ".$row["ApMatU"] ?></td>
<?php
                }

                mysql_free_result($res);
            }
            elseif($tipo=="Personal")
            {
?>
                <td><strong>PUESTO </strong></td>
                <td><?= $row["NomPu"] ?></td></tr>
                <tr>
                <td><strong>SUCURSAL</strong></td>
                <td><?= $row["NomS"] ?></td></tr>
                <tr>
                <td><strong>CLIENTES INGRESADOS</strong></td>
                <td><?= $row["NoIngresP"] ?></td>
<?php
            }
?>
            </tr></table>
            <br /><br />
            <table width="369" border="0">
            <tr><td></td></tr>
            <tr align="center"><td>OPERACIONES</td></tr>
            <tr><td></td></tr>
            <tr align="center">
            <td><a href="<?= $_SERVER['PHP_SELF']?>?action=ActualizarDatos<?= $tipo?>&amp;id=
            <?= $this->id?>">Actualizar datos</a></td></tr><tr align="center">
            <td><a href="<?= $_SERVER['PHP_SELF']?>?action=ActualizarPassword<?= $tipo?>&amp;id=
            <?= $this->id?>">Actualizar contraseña</a></td></tr>
<?php
            if(($tipo=="Personal")&&(($_SESSION['puesto']=="Administrador")||
            ($_SESSION['puesto']=="Gerente"))&&($_SESSION['num']!=$this->id))
            {
?>
                <tr align="center"><td><a href="<?= $_SERVER['PHP_SELF']?>?action=Borrar
                <?= $tipo?>&amp;id=<?= $this->id?>">Borrar</a></td></tr>
<?php
            }
            elseif(($tipo=="Cliente")&&($_SESSION['num']!=$this->id))
            {
?>
                <tr align="center"><td><a href="<?= $_SERVER['PHP_SELF']?>?action=Borrar
                <?= $tipo?>&amp;id=<?= $this->id?>">Borrar</a></td></tr>
<?php
```

```
            }

            if($tipo=="Cliente")
            {
?>
                <tr align="center"><td><a href="<?= $_SERVER['PHP_SELF']?>?action=RegistrarAvances
                &amp;id=<?= $this->id?>">Registrar avances de sesion</a></td></tr>
                <tr align="center">
                <td><a href="<?= $_SERVER['PHP_SELF']?>?action=CalendarizarSesion&amp;id=
                <?= $this->id?>">Calendarizar nueva sesi&oacute;n</a></td></tr>
                <tr align="center">
                <td><a href="<?= $_SERVER['PHP_SELF']?>?action=AdquirirTratamiento&amp;id=
                <?= $this->id?>">Adquirir tratamiento</a></td></tr>
<?php
            }
?>
           </table>
           </div>

           <div id="derecho1" align="center">
<?php

            if($tipo=="Cliente")
            {
                $hoy=date("d/m/Y");

                $sql="UPDATE sesion SET StatusSes='Actual' WHERE DATE_FORMAT(FecSes,'%d/%m/%Y')=
                '".$hoy."' AND StatusSes='Calendarizada'";
                mysql_query($sql);

                $_pagi_sql="SELECT CostNPac, NoSes, DATE_FORMAT(FecSes,'%d/%m/%Y') as fecha, DATE_FORMAT
                (HoraSes, '%H:%i') as hora, Torax, Cintura, Cadera, Brazo_I, Brazo_D, Pierna_I, Pierna_D,
                Peso, FAT, H2O, IMC, StatusSes FROM cliente, paquete_cliente, paquete, sesion, medidas,
                horario WHERE cliente.id_C=paquete_cliente.id_C
                AND paquete_cliente.id_Pac=paquete.id_Pac AND paquete_cliente.id_PC=sesion.id_PC AND
                sesion.id_Ses=medidas.id_Ses AND
                horario.id_HS=sesion.id_HS AND cliente.id_C=
                ".$this->id." ORDER BY NoSes";

                $_pagi_cuantos = 20;
                $_pagi_nav_num_enlaces=8;

                include("class.paginacion.php");

                if(mysql_num_rows($_pagi_result)!=0)
                {
?>
                    <table border="1"><tr bgcolor="<?= $color0 ?>" align="center">
                    <td><strong>SESION</strong></td>
                    <td><strong>FECHA</strong></td>
                    <td><strong>HORA</strong></td>
                    <td><strong>TORAX</strong></td>
                    <td><strong>CINTURA</strong></td>
                    <td><strong>CADERA</strong></td>
                    <td><strong>B.I. </strong></td>
                    <td><strong>B.D. </strong></td>
                    <td><strong>P.I. </strong></td>
```

```
                    <td><strong>P.D. </strong></td>
                    <td><strong>PESO</strong></td>
                    <td><strong>FAT</strong></td>
                    <td><strong>H2O</strong></td>
                    <td><strong>IMC</strong></td>
                    <td><strong>STATUS</strong></td></tr>
<?php
                    while($row = mysql_fetch_array($_pagi_result))
                    {
?>
                        <tr align="center" bgcolor="<?= $color ?>" onMouseOver=
                        "this.style.backgroundColor='<?= $color3 ?>';"
                        onMouseOut="this.style.backgroundColor='<?= $color ?>';">
                        <td><?= $row['NoSes'] ?></td>
                        <td><?= $row['fecha'] ?></td>
                        <td><?= $row['hora'] ?></td>
                        <td><?= $row['Torax'] ?></td>
                        <td><?= $row['Cintura'] ?></td>
                        <td><?= $row['Cadera'] ?></td>
                        <td><?= $row['Brazo_I'] ?></td>
                        <td><?= $row['Brazo_D'] ?></td>
                        <td><?= $row['Pierna_I'] ?></td>
                        <td><?= $row['Pierna_D'] ?></td>
                        <td><?= $row['Peso'] ?></td>
                        <td><?= $row['FAT'] ?></td>
                        <td><?= $row['H2O'] ?></td>
                        <td><?= $row['IMC'] ?></td>
                        <td><?= $row['StatusSes'] ?></td></tr>
<?php
                        if($color==$color1)
                        {
                            $color=$color2;
                        }
                        else
                        {
                            $color=$color1;
                        }
                    }
?>
                    </table>
                    <div class="paginacion" align="center">
                    <br/><ul><?= $_pagi_navegacion?></ul>
                    </div>
<?php
                    mysql_free_result($_pagi_result);
                }
                else
                {
?>
                    <h2>NO HAY TRATAMIENTOS ADQUIRIDOS POR ESTE CLIENTE...</h2>
                    <br /><h3>Pulsa el vínculo <em>Adquirir tratamiento</em> para adquirir
                    alguno de nuestros paquetes.</h3>
<?php
                }
            }
            elseif($tipo=="Personal")
            {
```

```
?>
                <img src="imagenes/8.JPG" border="0" width="550" height="350" alt="Svelta -
                Cl&iacute;nica de Reducci&oacute;n de Peso">
<?php
            }
?>
            </div>
            </div>
<?php
        }
    }

    public function cancelarSesion()
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);

        $sesion=$_GET['id'];

        $sql="SELECT DATE_FORMAT(FecSes,'%d/%m/%Y') as fecha, HoraSes FROM sesion, horario WHERE
        sesion.id_HS=horario.id_HS AND id_Ses=".$sesion."";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
            $fecha_hoy = date("d/m/Y");
            list ($dia_hoy, $mes_hoy, $anio_hoy) = split("/", $fecha_hoy);

            $fecha_calen = $row['fecha'];
            list ($dia_calen, $mes_calen, $anio_calen) = split("/", $fecha_calen);

            $hora_hoy=date("H:i:s");
            $hora_calen=$row['HoraSes'];

            mysql_free_result($res);

            if(($anio_hoy < $anio_calen)&&($mes_hoy < $mes_calen)&&($dia_hoy<$dia_calen))
            {
                $sql="UPDATE sesion SET StatusSes='Recalendarizada' WHERE id_Ses=".$sesion."";
                mysql_query($sql);
?>
                <meta http-equiv="refresh" content="0;url=<?= $_SERVER['PHP_SELF']?>">
                <script language="javascript">
                    window.alert("Sesion cancelada: A recalendarizar");
                </script>
<?php
            }
        }

        if(($anio_hoy == $anio_calen)&&($mes_hoy == $mes_calen)&&($dia_hoy==$dia_calen))
        {
            if($hora_hoy<=$hora_calen)
            {
```

```
                $sql="UPDATE sesion SET StatusSes='Recalendarizada' WHERE id_Ses=".$sesion."";
                mysql_query($sql);
?>
                <meta http-equiv="refresh" content="0;url=<?= $_SERVER['PHP_SELF']?>">
                <script language="javascript">window.alert("Sesion cancelada: A recalendarizar");</script>
<?php
            }
            else
            {
                $sql="UPDATE sesion SET StatusSes='Dada' WHERE id_Ses=".$sesion."";
                mysql_query($sql);
?>
                <meta http-equiv="refresh" content="0;url=<?= $_SERVER['PHP_SELF']?>">
                <script language="javascript">window.alert("Sesion cancelada: Dada");</script>
<?php
            }
        }
    }

    public function principal()
    {
        $tiempo_fuera = 300;

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite." AND
        id_U=".$_SESSION['num']."";
        mysql_query($sql);
?>
        <div id="izquierdo" align="center">
        <br />
<?php

        $hoy=date("d/m/Y");

        $sql="UPDATE sesion SET StatusSes='Actual' WHERE DATE_FORMAT(FecSes,'%d/%m/%Y')='".$hoy."' AND
        StatusSes='Calendarizada'";
        mysql_query($sql);

        $sql="SELECT * FROM personal, usuario, puesto, sucursal, zona, estado WHERE
        id_P=".$_SESSION['num']." AND id_P=id_U AND personal.id_S=sucursal.id_S AND
        sucursal.id_Z=zona.id_Z AND zona.id_Est=estado.id_Est";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
            $operacion=new basicas;
            $fecha=$operacion->fecha();
?>
            <table border="0">
            <tr><td width="100"><strong>NOMBRE(S):</strong></td>
            <td width="232"><?= $row['NomU']." ".$row['ApPatU']." ".$row['ApMatU']?></td></tr>
            <tr><td><strong>FECHA:</strong></td>
            <td><?= $fecha?></td></tr>
            <tr><td><strong>SUCURSAL:</strong></td>
            <td><?= $row['NomS']?></td></tr>
            <tr><td><strong>TIPO DE USUARIO:</strong></td>
            <td><?= $_SESSION['puesto']?></td></tr>
```

```
            <tr><td><strong>HORA ACTUAL:</strong></td>
            <td><script type="text/javascript" src="javascript/liveclock.js"></script></td></tr>
            </table>
<?php
            $_SESSION['suc']=$row['id_S'];
            mysql_free_result($res);
        }
?>
        </div>
        <div id="derecho" align="left">
        <br />
<?php
        $sql="SELECT * FROM horario";
        $res=mysql_query($sql);

        if($res)
        {
            $color0="#A4B4C1";
            $color1="#D5E7FB";
            $color2="#E8F2FC";
            $color3="#E0FAC5";
            $color=$color1;
?>
            <table border="1"><tr align="center" bgcolor="<?= $color0 ?>">
            <td width="92"><strong>HORA</strong></td>
            <td width="100" ><strong>CABINA 1</strong></td>
            <td width="20" ><strong>OPCION</strong></td>
            <td width="100"><strong>CABINA 2</strong></td>
            <td width="20" ><strong>OPCION</strong></td></tr>
<?php
            while($row=mysql_fetch_array($res))
            {
?>
                <tr align="center" bgcolor="<?= $color ?>" onMouseOver="this.style.backgroundColor=
                '<?= $color3 ?>';" onMouseOut="this.style.backgroundColor='<?= $color ?>';">
                <td width="92"><?= $row['HoraSes']?></td>
<?php
                $sql="SELECT * FROM sesion,horario WHERE DATE_FORMAT(FecSes,'%d/%m/%Y')='".$hoy."' AND
                sesion.id_HS=".$row['id_HS']." AND HoraSes='".$row['HoraSes']."' AND (StatusSes='Actual'
                OR StatusSes='Calendarizada')";
                $iBandera=0;
                $res2=mysql_query($sql);

                while($row2=mysql_fetch_array($res2))
                {
                    $iBandera=1;
                    $sesion=$row2['id_Ses'];

                    if($row2['StatusSes']=='Actual'||$row2['StatusSes']=='Calendarizada')
                    {
                        if($row2['CabinaSes']==1)
                        {
                            $sql="SELECT CabinaSes,StatusSes FROM sesion,horario WHERE DATE_FORMAT
                            (FecSes,'%d/%m/%Y')='26/08/2008' AND sesion.id_HS=".$row['id_HS']." AND HoraSes=
                            '".$row['HoraSes']."' AND (StatusSes='Actual' OR StatusSes=
                            'Calendarizada') AND CabinaSes=2";
                            $res3=mysql_query($sql);
```

```
                        if(mysql_num_rows($res3)==0)
                        {
?>
                            <td width="100">Ocupada</td>
                            <td width="20"><a href="<?= $_SERVER['PHP_SELF']?>?action=
                            CancelarSesion&amp;id=<?= $sesion?>"><img src="imagenes/x.png"
                            alt="Cancela la sesión" width="15" height="15" border="0"></a></td>
                            <td width="100">Libre</td>
                            <td width="20"><img src="imagenes/palomita.png" alt="Ve al apartado
                            Clientes y calendariza esta hora" width="15" height="15" border="0">
                            </td>
<?php
                        }
                        else
                        {
?>
                            <td width="100">Ocupada</td>
                            <td width="20"><a href="<?= $_SERVER['PHP_SELF']?>?action=
                            CancelarSesion&amp;id=<?= $sesion?>"><img src="imagenes/x.png"
                            alt="Cancela
                            la sesión" width="15" height="15" border="0"></a>
                            </td>
<?php
                        }

                        mysql_free_result($res3);
                    }
                    elseif($row2['CabinaSes']==2)
                    {
                        $sql="SELECT CabinaSes,StatusSes FROM sesion,horario WHERE DATE_FORMAT
                        (FecSes,'%d/%m/%Y')='26/08/2008' AND
                        sesion.id_HS=".$row['id_HS']." AND HoraSes='".$row['HoraSes']."'
                        AND (StatusSes='Actual' OR StatusSes='Calendarizada') AND CabinaSes=1";
                        $res3=mysql_query($sql);

                        if(mysql_num_rows($res3)==0)
                        {
?>
                            <td width="100">Libre</td>
                            <td width="20"><img src="imagenes/palomita.png" alt="Ve al apartado
                            Clientes y calendariza esta hora" width="15" height="15" border="0">
                            </td><td width="100">Ocupada</td>
                            <td width="20"><a href="<?= $_SERVER['PHP_SELF']?>?action=
                            CancelarSesion&amp;id=<?= $sesion?>"><img src="imagenes/x.png"
                            alt="Cancela la sesión" width="15" height="15" border="0"></a></td>
<?php
                        }
                        else
                        {
?>
                            <td width="100">Ocupada</td>
                            <td width="20"><a href="<?= $_SERVER['PHP_SELF']?>?action=
                            CancelarSesion&amp;id=<?= $sesion?>"><img src="imagenes/x.png"
                            alt="Cancela la sesión" width="15" height="15" border="0"></a></td>
<?php
                        }
```

```
                            mysql_free_result($res3);
                        }
                    }
                    else
                    {
                        $iBandera=0;
                    }
                }

                if ($iBandera==0)
                {
?>
                    <td width="100">Libre</td>
                    <td width="20"><img src="imagenes/palomita.png" alt="Ve al apartado
                    Clientes y calendariza esta hora" width="15" height="15" border="0"></td>
                    <td width="100">Libre</td>
                    <td width="20"><img src="imagenes/palomita.png" alt="Ve al apartado
                    Clientes y calendariza esta hora" width="15" height="15" border="0"></td>
<?php
                }

                if($color==$color1)
                {
                    $color=$color2;
                }
                else
                {
                    $color=$color1;
                }
?>
                </tr>
<?php
            }
        }

        mysql_free_result($res2);
?>
        </table>
        </div>
        </div>
<?php
    }
}

?>
```

D.8. index.php

```
<?php

session_start();

require("xajax/xajax.inc.php");
```

```
$xajax=new xajax();
$xajax->setCharEncoding('ISO-8859-1');
$xajax->decodeUTF8InputOn();

include("class.basicas.php");
//Creación de un objeto "DB_mysql" y conexión a la base de datos solicitada
$funcion=new DB;
$funcion->conectar("svelta","localhost","root","");

class index
{
    public function entrada()
    {
?>
        <font color="#3333FF" size="+2">
        <br /><b>&iexcl;Hola, estimado visitante!</b>
        <br />
        <br /><b>Bienvenido a <em>Svelta - Cl&iacute;nica de Reducci&oacute;n de Peso</em>.</b>
        <br />
        <br /><b>Elige una de las opciones</b>
        <br /><b>y empieza a sentirte bien...</b>
        </font>
        </div>
<?php
    }

    public function quienes()
    {
?>
        <div id="izquierdo" align="left">
        <br /><em>Svelta</em> es una cl&iacute;nica de  reducci&oacute;n de peso y moldeamiento de
        figura que nace en septiembre del 2006.
        <br />
        <br />Somos un negocio con experiencia en nuestro campo, y estamos interesados por la salud y
        bienestar de nuestros clientes.
        <br />
        <br />Contamos con equipo avanzado y personal altamente calificado que respaldan la calidad de
        nuestros servicios.
        <br />
        <br/>Nuestros horarios de servicio son:
        <br /><strong>Lunes a viernes: </strong>8:00 AM a 9:00 PM
        <br /><strong>S&aacute;bado: </strong>8:00 AM a 1:00 PM
        <br />
        <br />Conoce nuestros planes y promociones y sorpr&eacute;ndete.
        <br />
        <br />Date la oportunidad de cumplir tus prop&oacute;sitos sinti&eacute;ndote de maravilla.
        <br />&iexcl;Ven y con&oacute;cenos!
        </div>

        <div id="derecho" align="center">
        <br /><img src="imagenes/1.jpg" name="slide" border=0 style="filter:progid:DXImageTransform.
        Microsoft.Pixelate(MaxSquare=15,Duration=1)" width="410" height="340"
        alt="Svelta - Cl&iacute;nica de Reducci&oacute;n de Peso">
        <script type="text/javascript">
            var whichimage=0
            var pixeldelay=(ie55)? document.images.slide.filters[0].duration*1000 : 0
            function slideit()
```

```
            {
                if (!document.images) return;
                if (ie55) document.images.slide.filters[0].apply()
                document.images.slide.src=imageholder[whichimage].src;
                if (ie55) document.images.slide.filters[0].play()
                whichimage=(whichimage<slideimages.length-1)? whichimage+1 : 0;
                setTimeout("slideit()",slidespeed+pixeldelay);
            }
            slideit()
        </script>
        </div>
        </div>
<?php
    }

    public function listaSucursales()
    {
        $sql="SELECT * FROM estado ORDER BY NomEst";
        $res=mysql_query($sql);

        if($row=mysql_fetch_array($res))
        {
?>
            <div id="izquierdo" align="left">
            <form id="sucursales" action="">
            <select name="sucursal" onChange="xajax_cargar_sucursales(xajax.getFormValues
            ('sucursales'))">
            <option>Selecciona un estado...</option>
<?php
            while ($row=mysql_fetch_array($res))
            {
?>
                <option value="<?=$row['id_Est']?>"><?=$row['NomEst']?></option>
<?php
            }

            mysql_free_result($res);
        }
?>
            </select></form></div>
            <div id="derecho" align="left"></div>
            </div>
<?php
    }

    public function login()
    {
        $tiempo_fuera = 300; //tiempo maximo de inactividad en segundos, en este caso 5 minutos

        $limite = time() - $tiempo_fuera;
        $sql = "UPDATE usuario SET Conectado='0', Actividad='0' WHERE Actividad < ".$limite."";
        mysql_query($sql);
?>
        <br /><h2>ENTRADA A <em>SVELTA SYSTEMS</em>.</h2>
        <form name="login" action="" method="post">
        <br />
        <table align="center" width="280" border="0"><tr align="center">
```

```
        <td width="126" align="center">NO. CUENTA</td>
        <td width="144">
        <input type="text" name="no_usuario" onKeyPress="return validarSoloNumeros(event)">
        </td></tr>
        <tr align="center"><td>NIP</td>
        <td><input type="password" name="cont_usuario" onKeyPress="return invalidarAcento(event)">
        </td></tr></table>
        <br /><br />
        <input type="submit" class="boton" name="ingresar_sistema" value="    ENTRAR   "
        onClick="return validarLogin(this.form)">
        </form>
        </div>
<?php

        if(isset($_POST['ingresar_sistema']))
        {
            $usuario = $_POST["no_usuario"];
            $pass=md5($_POST["cont_usuario"]);

            $sql="SELECT id_U, ContU, id_P, NomPu FROM usuario, personal, puesto WHERE
            id_U='".$usuario."' AND ContU='".$pass."' AND id_U=id_P AND
            personal.id_Pu=puesto.id_Pu";
            $res=mysql_query($sql);

            if($res)
            {
                $row=mysql_fetch_array($res);

                if(strcmp($row['ContU'],$pass)==0)
                {
                    $_SESSION['num']=$row["id_U"];
                    $_SESSION['puesto']=$row['NomPu'];

                    mysql_free_result($res);

                    $sql="SELECT Conectado FROM usuario WHERE id_U=".$_SESSION['num']."";
                    $res=mysql_query($sql);

                    if($row=mysql_fetch_array($res))
                    {
                        if($row['Conectado']=="0")
                        {
                            $sql="UPDATE usuario SET Conectado='1', Actividad='".time()."' WHERE
                            id_U=".$_SESSION['num']."";
                            mysql_query($sql);

                            mysql_free_result($res);
?>
                            <script type="text/javascript">
                                window.open("svelta.php","","resizable=no, status=no, location= no,
                                width=960, height=720");
                            </script>
                            <meta http-equiv="refresh" content="0; url=index.php">
<?php
                        }
                        else
                        {
```

```
?>
                            <script type="text/javascript">
                                window.alert("El usuario ya se ha logueado...
                                Acceso denegado");
                            </script>
<?php
                        }
                    }
                }
                else
                {
?>
                    <script type="text/javascript">
                        window.alert("Contraseña inválida");
                    </script>
<?php
                }
            }
            else
            {
?>
                <script type="text/javascript">
                    window.alert("El número de cuenta y/o contraseña son inválidos");
                </script>
<?php
            }
        }
    }

    public function principalIndex()
    {
        $actual="current";
        $clase1="";
        $clase2="";
        $clase3="";
        $clase4="";

        switch($_GET['action'])
        {
            case "QuienesSomos":
                $clase2=$actual;
                break;
            case "Sucursales":
                $clase3=$actual;
                break;
            case "IngresoASistema":
                $clase4=$actual;
                break;
            default:
                $clase1=$actual;
                break;
        }
?>
        <!--En esta sección se incrusta el menu de todo el sistema-->
        <div id="menu" align="center">
        <ul class="glossymenu">
        <li class="<?= $clase1?>"><a href="<?= $_SERVER['PHP_SELF']?>?action=Principal">
```

```
        <b>HOME</b></a></li>
        <li class="<?= $clase2?>"><a href="<?= $_SERVER['PHP_SELF']?>?action=QuienesSomos">
        <b>&iquest;QUIENES SOMOS?</b></a></li>
        <li class="<?= $clase3?>"><a href="<?= $_SERVER['PHP_SELF']?>?action=Sucursales">
        <b>SUCURSALES</b></a></li>
        <li class="<?= $clase4?>"><a href="<?= $_SERVER['PHP_SELF']?>?action=IngresoASistema">
        <b>MIEMBROS</b></a></li>
        </ul>
        </div>
<?php
        $valor1="contenedor";
        $valor2="interno";
        $valor3="contenedorpag";

        switch($_GET['action'])
        {
            case "QuienesSomos": $valor=$valor2; break;
            case "Sucursales": $valor=$valor2; break;
            default: $valor=$valor1; break;
        }
?>
        <div id="<?= $valor?>" align="center">
        <br /><br />
<?php
        switch($_GET['action'])
        {
            case "QuienesSomos": $this->quienes(); break;
            case "Sucursales": $this->listaSucursales(); break;
            case "IngresoASistema": $this->login(); break;
            default: $this->entrada(); break;
        }
    }
}

function cargar_sucursales($form_entrada)
{
    $respuesta = new xajaxResponse('ISO-8859-1');

    $suc=$form_entrada["sucursal"];

    if($suc!="Selecciona un estado...")
    {
        $sql="SELECT * FROM Zona, Sucursal WHERE Zona.id_Est=".$suc." AND Zona.id_Z=Sucursal.id_Z";
        $res=mysql_query($sql);

        if(mysql_num_rows($res)!=0)
        {
            while($row=mysql_fetch_array($res))
            {
                $cont_form='<p><font face="Comic Sans MS" size="2">
                    <br /><b>Sucursal: </b>'.$row['NomS'].'
                    <br /><b>Dirección: </b>'.$row['DirS'].'
                    <br /><b>Teléfono: </b>'.$row['TelS'].'
                    <br /><b>Zona: </b>'.$row['NomZ'].'</font></p>';
            }
        }
        else
```

```
        {
            $cont_form='<br /><br /><h2>A&uacute;n no contamos con sucursales en este estado...
            &iexcl;Esp&eacute;ranos pronto!</h2>';
        }

        mysql_free_result($res);
    }

    $respuesta->addAssign("derecho","innerHTML",$cont_form);
    return $respuesta;
}

$xajax->registerFunction("cargar_sucursales");
$xajax->registerFunction("entrar");
$xajax->processRequests();

?>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" type="text/css" href="css/svelta.css">
<link rel="stylesheet" type="text/css" href="css/menu.css">

<script type="text/javascript" src="javascript/noselection.js"></script>
<script type="text/javascript" src="javascript/typingtext.js"></script>
<script type="text/javascript" src="javascript/svelta.js"></script>
<script type="text/javascript" src="javascript/google.js"></script>
<script type="text/javascript">
    var slidespeed=2500
    var slideimages=new Array("imagenes/1.jpg","imagenes/2.jpg","imagenes/3.jpg", "imagenes/4.jpg",
    "imagenes/5.jpg", "imagenes/6.jpg", "imagenes/7.jpg", "imagenes/8.jpg", "imagenes/9.jpg",
    "imagenes/10.jpg", "imagenes/11.jpg", "imagenes/12.jpg", "imagenes/13.jpg", "imagenes/14.jpg",
    "imagenes/15.jpg", "imagenes/16.jpg", "imagenes/17.jpg", "imagenes/18.jpg", "imagenes/19.jpg",
    "imagenes/20.jpg")
    var imageholder=new Array()
    var ie55=window.createPopup
    for (i=0; i<slideimages.length; i++)
    {
        imageholder[i]=new Image();
        imageholder[i].src=slideimages[i];
    }
</script>

<script type="text/javascript">
    function abrirSistema()
    {
        window.open("svelta.php","","location=no, status=no,width=1000,height=800")
    }
</script>
<title>.:. SVELTA - CLÍNICA DE REDUCCIÓN DE PESO .:.</title>
<?php
$xajax->printJavascript("xajax/");
?>
</head>
```

```
<body id="index">
<script type="text/javascript" src="javascript/norightclick.js"></script>
<div id="encabezado" align="center">
<h1><font face="Comic Sans MS" color="blue"><i>SVELTA - CL&Iacute;NICA DE REDUCCI&Oacute;N
DE PESO</i></font></h1>
</div>
<?php

    $pagina=new index();
    $pagina->principalIndex();
?>
<div id="pie">
<center><font face="Comic Sans MS" color="#FF9900" size="1">
Svelta - Cl&iacute;nica de Reducci&oacute;n de Peso
<br />
Derechos Reservados &reg; <?= $anio=date("Y") ?>
</font></center>
</div>
<script type="text/javascript">disableSelection(document.body)</script>
</body>
</html>
```

D.9. salida.php

```
<?php

session_start();

if(empty($_SESSION['num']))
{
?>
    <script language="javascript">
        window.alert('Acceso no autorizado... Será redirigido a la página principal');
    </script>
    <meta http-equiv="refresh" content="0; url=index.php">
<?php
}
else
{
    include("class.basicas.php");

    $funcion=new DB;
    $funcion->conectar("svelta","localhost","root","");

    $sql="UPDATE usuario SET Conectado='0', Actividad='0' WHERE id_U=".$_SESSION['num']."";
    $res=mysql_query($sql);

    $_SESSION['num']=0;
    $_SESSION['puesto']="";

    session_destroy();
?>
    <script language="javascript">window.close();</script>
<?php
```

```
}
?>
```

D.10. svelta.php

```
<?php

session_start();

if(empty($_SESSION['num']))
{
?>
    <script type="text/javascript">
        window.alert('Acceso no autorizado... Será redirigido a la página principal');
    </script>
    <meta http-equiv="refresh" content="0; url=index.php">
<?php
}

require("xajax/xajax.inc.php");
$xajax=new xajax();
$xajax->setCharEncoding('ISO-8859-1');
$xajax->decodeUTF8InputOn();

include("class.basicas.php");

$funcion=new DB;
$funcion->conectar("svelta","localhost","root","");

include("class.usuarios.php");
include("class.clientes.php");
include("class.personal.php");
include("class.paquetes.php");
include("class.ajax.php");
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<link rel="stylesheet" type="text/css" href="css/calendario.css">
<link rel="stylesheet" type="text/css" href="css/menu.css">
<link rel="stylesheet" type="text/css" href="css/svelta.css">
<script type="text/javascript" src="javascript/noselection.js"></script>
<script type="text/javascript" src="javascript/svelta.js"></script>
<script type="text/javascript" src="javascript/calendar.js"></script>
<script type="text/javascript" src="javascript/calendar-es.js"></script>
<script type="text/javascript" src="javascript/calendar-setup.js"></script>
<title>.:. SVELTA - CL&Iacute;NICA DE REDUCCI&Oacute;N DE PESO .:.</title>
<?php
$xajax->printJavascript("xajax/");
?>
</head>
<body id="home" onLoad="show_clock()">
<script type="text/javascript" src="javascript/norightclick.js"></script>
<div id="encabezado" align="center">
```

```
<h1><font color="blue"><i>SVELTA - CL&Iacute;NICA DE REDUCCI&Oacute;N DE PESO</i></font></h1>
</div>
<?php

    $objUsuario=new usuario();
    $objCliente=new cliente();
    $objPersonal=new personal();
    $objPaquete=new paquete();

    $actual="current";
    $clase1="";
    $clase2="";
    $clase3="";
    $clase4="";
    $clase5="";

    $valor1="contenedorpag";
    $valor2="contenedor";
    $valor3="interno";

    switch($_GET['action'])
    {
        case "Cliente":
            $clase2=$actual;
            $valor=$valor1;
            break;
        case "ActualizarDatosCliente":
            $clase2=$actual;
            $valor=$valor2;
            break;
        case "ActualizarPasswordCliente":
            $clase2=$actual;
            $valor=$valor2;
            break;
        case "BorrarCliente":
            $clase2=$actual;
            $valor=$valor2;
            break;
        case "VerDetallesDeCliente":
            $clase2=$actual;
            $valor=$valor3;
            break;
        case "IngresarCliente":
            $clase2=$actual;
            $valor=$valor2;
            break;
        case "RegistrarAvances":
            $clase2=$actual;
            $valor=$valor3;
            break;
        case "CalendarizarSesion":
            $clase2=$actual;
            $valor=$valor2;
            break;
        case "AdquirirTratamiento":
            $clase2=$actual;
            $valor=$valor2;
```

```
            break;
        case "Personal":
            $clase3=$actual;
            $valor=$valor1;
            break;
        case "VerDetallesDePersonal":
            $clase3=$actual;
            $valor=$valor3;
            break;
        case "IngresarPersonal":
            $clase3=$actual;
            $valor=$valor2;
            break;
        case "ActualizarDatosPersonal":
            $clase3=$actual;
            $valor=$valor2;
            break;
        case "ActualizarPasswordPersonal":
            $clase3=$actual;
            $valor=$valor2;
            break;
        case "BorrarPersonal":
            $clase3=$actual;
            $valor=$valor2;
            break;
        case "Paquetes":
            $clase4=$actual;
            $valor=$valor1;
            break;
        case "DesactivarPaquete":
            $clase4=$actual;
            $valor=$valor1;
            break;
        case "ActivarPaquete":
            $clase4=$actual;
            $valor=$valor1;
            break;
        case "ActualizarPaquete":
            $clase4=$actual;
            $valor=$valor2;
            break;
        case "CargarPaquete":
            $clase3=$actual;
            $valor=$valor2;
            break;
        default:
            $clase1=$actual;
            $valor=$valor1;
            break;
    }
?>
<div id="menu" align="center">
<ul class="glossymenu">
<li class="<?= $clase1?>"><a href="<?= $_SERVER['PHP_SELF']?>?action=Inicio">
<b>PRINCIPAL</b></a></li>
<li class="<?= $clase2?>"><a href="<?= $_SERVER['PHP_SELF']?>?action=Cliente">
<b>CLIENTES</b></a></li>
```

```
<li class="<?= $clase3?>"><a href="<?= $_SERVER['PHP_SELF']?>?action=Personal">
<b>PERSONAL</b></a></li>
<?php

    if($_SESSION['puesto']=="Administrador"||$_SESSION['puesto']=="Gerente")
    {
?>
        <li class="<?= $clase4?>"><a href="<?= $_SERVER['PHP_SELF']?>?action=Paquetes">
        <b>PAQUETES</b></a></li>
<?php
    }
?>
<li><a href="salida.php"><b>SALIR DE SISTEMA</b></a></li>
</ul>
</div>
<div id="<?= $valor?>" align="center">
<br />

<?php

    switch($_GET['action'])
    {
        case "ActualizarDatosCliente":
            $objCliente->actualizarCliente();
            break;
        case "ActualizarPasswordCliente":
            $objCliente->actualizarPassCliente();
            break;
        case "BorrarCliente":
            $objCliente->borrarCliente();
            break;
        case "VerDetallesDeCliente":
            if(isset($_POST['cambiar_pass_Cliente']))
            {
                $objUsuario->cambiarPassword("Cliente");
            }
            elseif(isset($_POST['actualizar_Cliente']))
            {
                $objUsuario->actualizarUsuario("Cliente");
            }
            elseif(isset($_POST['registrar_sesion']))
            {
                $objCliente->cargarAvances();
            }
            $objCliente->verDetallesCliente();
            break;
        case "IngresarCliente":
            $objCliente->ingresarCliente();
            break;
        case "RegistrarAvances":
            $objCliente->registrarAvances();
            break;
        case "CalendarizarSesion":
            $objCliente->calendarizarSesion();
            break;
        case "AdquirirTratamiento":
            $objCliente->adquirirTratamiento();
```

```
    break;
case "Cliente":
    if(isset($_POST['ingresar_Cliente']))
    {
        $objUsuario->cargarUsuario("Cliente");
    }
    $objCliente->mostrarClientes();
    break;
case "VerDetallesDePersonal":
    if(isset($_POST['cambiar_pass_Personal']))
    {
        $objUsuario->cambiarPassword("Personal");
    }
    elseif(isset($_POST['actualizar_Personal']))
    {
        $objUsuario->actualizarUsuario("Personal");
    }
    $objPersonal->verDetallesPersonal();
    break;
case "IngresarPersonal":
    $objPersonal->ingresarPersonal();
    break;
case "ActualizarDatosPersonal":
    $objPersonal->actualizarPersonal();
    break;
case "ActualizarPasswordPersonal":
    $objPersonal->actualizarPassPersonal();
    break;
case "BorrarPersonal":
    $objPersonal->borrarPersonal();
    break;
case "Personal":
    if(isset($_POST['ingresar_Personal']))
    {
        $objUsuario->cargarUsuario("Personal");
    }
    $objPersonal->mostrarPersonal();
    break;
case "Paquetes":
    if(isset($_POST['actualizar_paquete']))
    {
        $objPaquete->cargarActualizacionPaquete();
    }
    $objPaquete->mostrarPaquetes();
    break;
case "DesactivarPaquete":
    $objPaquete->cambiarEstadoPaquete("Activo");
    $objPaquete->mostrarPaquetes();
    break;
case "ActivarPaquete":
    $objPaquete->cambiarEstadoPaquete("No activo");
    $objPaquete->mostrarPaquetes();
    break;
case "ActualizarPaquete":
    $objPaquete->actualizarDatosPaquete();
    break;
case "CargarPaquete":
```

```
            $objPaquete->cargarPaquete();
            break;
        case "CancelarSesion":
            $objUsuario->cancelarSesion();
        default:
            $objUsuario->principal();
            break;
    }
?>
<div id="pie">
<center><font face="Comic Sans MS" color="#FF9900" size="1">
Svelta - Cl&iacute;nica de Reducci&oacute;n de Peso
<br>
<?php
    $anio=date("Y");
?>
Derechos Reservados &reg; <?= $anio ?>
</font></center>
</div>
<script type="text/javascript">disableSelection(document.body)</script>
</body>
</html>
```

Bibliografía y Cibergrafía.

[LED99] Leduc, Daniel / St. Pierre, Armand (1999). *HTML - Creación y difusión de páginas web.* México: Trillas.

[PAP06] Pavón Puertas, Jacobo (2006). *Creación de un portal con PHP y MySQL.* México: Alfaomega Ra-Ma, 2a. Edición.

[VAJ04] Valade, Janet (2004). *PHP para Dummies.* Panamá: ST Editorial.

[ULL04] Ullman, Larry (2004). *MySQL.* Madrid: Pearson Educación S.A.

[ELR04] Elmasri, Ramez A. / Navathe, Shamkant B. (2004). *Fundamentos de Sistemas de Bases de Datos.* España: Pearson Prentice Hall, 3a. Edición.

[ORC02] Orós, Juan Carlos (2002). *Diseño de páginas web interactivas con Javascript y CSS.* México: Alfaomega Ra-Ma, 3a. Edición.

[MAD77] Matthews, Don Q. (1977). *Diseño de Sistemas de Información Administrativa.* Argentina: El Ateneo.

[KIP07] Kimmel, Paul (2007). *Manual de UML.* México: McGraw Hill.

[VIM06] Villegas Lara, Magín (2006). *Desarrollo de un sistema de información orientado a la web. Caso de estudio: Área Académica de Ciencias Computacionales* (Tesis de licenciatura, Universidad Autónoma del Estado de Hidalgo).

[PEJ05] Pérez Sánchez, José Antonio (2005). *Métodos y técnicas para el aseguramiento de la calidad en el diseño de la interfaz de usuario para los desarrollos web.* (Tesis de licenciatura, Universidad Autónoma del Estado de Hidalgo).

[SIA02] Silberschatz, Abraham / Korth, Henry F. / Sudarshan, S. (2002). *Fundamentos de Bases de Datos.* España: McGraw Hill, 4a. Edición.

[RUM05] Ruvalcaba, Mara (2005, abril). *Procesos de Software.* Software Gurú, pp. 50-55.

[BOG99] Booch, Grady / Rumbaugh, James / Jacobson, Ivar (1999). *El Lenguaje Unificado de Modelado.* España: Addison Wesley. 1a. Reimpresión.

[TEL07] Tejada, Luis (2007). *Proceso Unificado de Rational.* Consultado en marzo, 10, 2008 en *http://babotejada.wordpress.com/2007/06/16/proceso-unificado-de-rational/.*

[BD08] Anónimo (2008). *Base de datos.* Consultado en abril, 1, 2008 en *http://es.wikipedia.org/wiki/Base_de_datos.*

[HTM08] Anónimo (2008). *HTML.* Consultado en abril, 1, 2008 en *http://es.wikipedia.org/wiki/Html.*

[MYS08] Anónimo (2008). *MySQL.* Consultado en abril, 3, 2008 en *http://es.wikipedia.org/wiki/MySQL.*

[RUP08] Anónimo (2008). *Proceso Unificado de Rational.* Consultado en marzo, 3, 2008 en *http://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational.*

[POSS05] Pozo, Salvador (2005). *Índice de Sentencias SQL.* Consultado en abril, 28, 2008 en *www.conclase.net/c/mysql/index.php?tab=Sentencias.*

[XAJX08] Anónimo (2008). *Xajax: Ajax para PHP.* Consultado en mayo, 21, 2008 en *www.desarrolloweb.com/articulos/xajax-libreria-php.html.*

[LORC03] López Rodríguez, César (2003). *Desarrollo de un sistema para la gestión de articulos deportivos.* Consultado en febrero, 12, 2008 en *www.dsic.upv.es/asignaturas/facultad/lsi/ejemplorup/index.html.*

[PAOS02] Pastor, Oscar (2002). *OOWS: Una aproximación para el modelado conceptual de aplicaciones web.* Consultado en mayo, 16, 2008 en *www.ing.unlpam.edu.ar/icwe2002/tutoriales/opastor.pdf*

[PINJ05] Pinedo, Jorge (2005). *Paginator versión 1.6.* Consultado en abril, 29, 2008 en *http://jpinedo.webcindario.com/scripts/paginator/.*

[PASS08] Anónimo (2008). *Evitar el uso multiple de un password.* Consultado en junio, 12, 2008 en *www.forosdelweb.com/f18/evitar-uso-multiple-password-460747/.*

[XAJX07] Anónimo (2007). *xaJax - Funciones PHP.* Consultado en mayo, 26, 2008 en *http://www.programacionweb.net/articulos/articulo/?num=500.*

[GOCJ04] González Cornejo, José Enrique (2004). *¿Qué es UML?.* Consultado en marzo, 13, 2008 en *http://www.docirs.cl/uml.htm.*

[RJOR07] R., Jorge (2007). *AMIPCI: Internet en México - Estadisticas 2007.* Consultado en febrero, 18, 2008 en *http://pymes-mexico.blogspot.com/2007/10/amipci-internet-en-mxico-estadisticas.html.*

[UPTG04] Trabajo práctico grupal. Universidad de Palermo (2004). *Ciclos de Vida de proyectos. RUP.* Consultado en marzo, 5, 2008 en *http://writer.zoho.com/public/27201/38205*

[JUMA08] Justo, Miguel Angel (2008). *¿Por que PHP es mas popular en la web?.* Consultado en marzo, 17, 2008 en *http://pwneds.blogspot.com/2007/12/por-que-php-es-mas-popular-en-la-web.html.*

[TASE08] Tarrillo, Sergio (2008). *Mi turno de hablar de AJAX, ventajas y desventajas.* Consultado en mayo, 7, 2008 en *http://geeks.ms/blogs/sergiotarrillo/archive/2007/01/09/8420.aspx.*

[CIJA06] Cid, Jaime (2006). *Estadisticas de uso de AJAX (SDTimes).* Consultado en mayo, 7, 2008 en *http://blogs.sun.com/jaimecd/entry/estadisticas_sdtimes_ajax*

[RACR02] R. Arroyo, Cristian (2002). *PHP supera a ASP como el lenguaje de la Web.* Consultado en marzo, 17, 2008 en *http://www.vivaphp.com.ar/eventos/1507.html*

[POSM05] Pozo, Salvador (2005). *MySQL con Clase*. Consultado en abril, 28, 2008 en *http://mysql.conclase.net/curso/index.php*

[BEFR06] Berzal, Fernando (2006). *El ciclo de vida de un sistema de información*. Consultado en febrero, 4, 2008 en *http://flanagan.ugr.es/docencia/2005-2006/2/apuntes/ciclovida.pdf*

[SIDA01] Silva, Darío Andrés / Mercerat, Bárbara (2001). *Construyendo aplicaciones web con una metodología de diseño orientada a objetos*. Consultado en marzo, 25, 2008 en *http://www.lifia.info.unlp.edu.ar/papers/2001/Silva2001.pdf*

[OSBA08] Osorio Barriga, Arturo (2008). *Ajax*. Consultado en mayo, 8, 2008 en *http://aosorio.wordpress.com/2008/03/03/ajax/*

[MALG06] Martínez Luma, G. L. (2006). *Sistema de información para consultas a investigaciones arqueológicas*. Consultado en abril, 16, 2008 en www.cic.ipn.mx/aguzman/papers/162 %20d %20inf %20p %20consultas %20a %20investig %20arqueolo'gicas.doc