Universidad de las Ciencias Informáticas

Facultad 4

Título:

Royston: Sistema de apoyo a la planeación mercadotécnica del centro FORTES

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

Autores:

Roberto Carlos Aballe Ochoa

Juan Carlos Lomba Fernández

Tutores:

Ing. María de los Angeles Garcia Montero

Lic. Maniuryis Peña Azahares

Ciudad de La Habana, 15 de junio de 2011

“Año 53 de la Revolución”

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los 15 días del mes de junio del año 2011.

Roberto Carlos Aballe Ochoa

_____

Firma del Autor

Ing. María de los Angeles Garcia Montero

_____

Firma del Tutor

Juan Carlos Lomba Fernández

_____

Firma del Autor

Lic. Maniuryis Peña Azahares

_____

Firma del Tutor

DEDICATORIAS

Roberto Carlos:

A mis padres, por haberme formado como un hombre de bien, y en especial a mi madre, por la confianza, las preocupaciones, por ser mi tribunal más severo y mi tutora más exigente.

Juan Carlos:

A mi familia en general, por haberme convertido en el hombre que soy, y en especial a mi madre y abuelos, por haber sido mis guías y mis ejemplos.

AGRADECIMIENTOS

Agradecemos, en primer lugar, a nuestros amigos incondicionales, con los cuales compartimos tantos momentos agradables.

A los que molestamos en busca de ayuda y de los que tanto aprendimos.

A nuestras tutoras, por dedicarnos tanto tiempo, por sus exigencias, sus enseñanzas y el afán de vernos convertidos en profesionales.

A los miembros del tribunal, porque sus señalamientos fueron oportunos y primaba el deseo de enseñar y ayudar.

Queremos agradecer también a los profesores que, durante estos cinco años, han tenido que ver en nuestra formación integral.

A la facultad 8, por habernos acogido como hijos y a la facultad 4, por entregar a la sociedad dos nuevos ingenieros.

RESUMEN

Los procesos de planeación mercadotécnica son vitales para el funcionamiento de las organizaciones que se dedican a la comercialización de productos o servicios. En la actualidad el proceso de planeación mercadotécnica en la Universidad de las Ciencias Informáticas y en particular del centro FORTES se realiza de forma empírica y operativa y no existe una normalización del proceso, ajustado a las características de la institución. El objetivo de la presente investigación es desarrollar una herramienta que automatice los procesos de planeación mercadotécnica del centro FORTES que se enmarca en el subsistema Datos Internos de un Sistema de Información de Marketing (SIM). Para el desarrollo del sistema se empleó como Sistema Gestor de Base de Datos: PostgreSQL, como servidor web se seleccionó Apache HTTP Server. Los lenguajes de programación utilizados fueron PHP para el lado del servidor y JavaScript para el lado del cliente. El *framework* PHP seleccionado fue Symfony con el cual se logró minimizar el tiempo de implementación de las Historias de Usuario. Se implementó sobre el IDE Netbeans. El patrón arquitectónico Modelo-Vista-Controlador se fundamenta a partir de su utilización por el *framework* Symfony. Se obtuvieron los diferentes artefactos a partir de la utilización de la metodología de desarrollo de *software* ágil SXP. El presente trabajo permitió crear una aplicación que automatiza el proceso de planeación mercadotécnica del centro FORTES, contribuyendo a la optimización en tiempo y esfuerzo invertido en este proceso.

Palabras Clave: procesos de planeación mercadotécnica, Sistema de Información de Mercadotecnia, Symfony

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS 3
ÍNDICE DE TABLAS 3
INTRODUCCIÓN: 4
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA 8
1.1. Introducción 8
1.2. La información 9
1.3. Gestión de Información 10
1.4. Sistemas de Información 12
1.4.1. Componentes de un Sistema de Información 13
1.5. Sistemas de Información de Mercadotecnia 15
1.5.1. Estructura de un Sistema de Información de Mercadotecnia 16
1.5.2. Sistemas de Información de Mercadotecnia en Cuba 17
1.6. Análisis de soluciones existentes 19
1.6.1. Sistemas a nivel internacional 19
1.6.2. Sistemas a nivel nacional 21
1.7. Herramientas y tecnologías de desarrollo 22
1.7.1. Lenguaje Unificado de Modelado (UML) 22
1.7.2. Herramienta CASE 23
1.7.3. Metodología de desarrollo de *software* 24
1.7.4. Sistema Gestor de Base de Datos (SGBD) 28
1.7.5. Servidor web 29
1.7.6. Lenguajes de programación 31
1.7.7. *Framework* PHP 34
1.7.8. *Framework* JavaScript 35
1.7.9. Entorno Integrado de Desarrollo (IDE) 37
1.8. Conclusiones parciales 38
CAPÍTULO 2: ANÁLISIS DE LA SOLUCIÓN PROPUESTA 39
2.1. Introducción 39

2.2. Breve descripción del sistema 39
2.3. Arquitectura del sistema 40
2.4. Patrones de diseño 41
2.4.1 Patrones GRASP 42
2.4.2 Patrones GOF 43
2.5. Lista de Reserva del Producto 43
2.6. Historias de Usuario 49
2.7. Modelo de diseño 50
2.8. Modelo de datos 53
2.9. Modelo de despliegue 53
2.10. Conclusiones parciales 55
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA DE LA SOLUCIÓN PROPUESTA 56
3.1. Introducción 56
3.2. Plan de *Releases* 56
3.3. Tareas de Ingeniería 58
3.4. Pruebas de Aceptación 60
3.4.1. Resultado de las pruebas 64
3.5. Estándar de código 64
3.6. Seguridad 68
3.7. Tratamiento de errores 69
3.8. Conclusiones parciales 70
CONCLUSIONES GENERALES 71
RECOMENDACIONES 73
Referencias bibliográficas I

ÍNDICE DE FIGURAS

Figura 1: Componentes de un Sistema de Información....14
Figura 2: Componentes de un Sistema de Información de Mercadotecnia....16
Figura 3: Modelo-Vista-Controlador para Symfony....41
Figura 4: Modelo de Diseño basado en el *framework* Symfony....51
Figura 5: Modelo de clases del diseño de HU01....52
Figura 6: Modelo de despliegue....54

ÍNDICE DE TABLAS

Tabla 1: Lista de Reserva del Producto (LRP)....45
Tabla 2: HU Gestionar servicios externos....50
Tabla 3: Plan de *Releases*....57
Tabla 4: Tarea de Ingeniería 1 de HU01....58
Tabla 5: Tarea de Ingeniería 2 de HU01....59
Tabla 6: Tarea de Ingeniería 3 de HU01....59
Tabla 7: Tarea de Ingeniería 4 de HU01....60
Tabla 8: Caso de Prueba de Aceptación 1 de HU Gestionar servicios externos....61
Tabla 9: Caso de Prueba de Aceptación 2 de HU Gestionar servicios externos....62
Tabla 10: Caso de Prueba de Aceptación 3 de HU Gestionar servicios externos....63
Tabla 11: Caso de Prueba de Aceptación 4 de HU Gestionar servicios externos....64

INTRODUCCIÓN:

La Universidad de las Ciencias Informáticas (UCI) fue creada con el objetivo de que constituyera un centro estratégico para convertir nuestro país en paradigma de la sociedad de la información y el conocimiento para todos y además para insertar al país en nuevos nichos del mercado mundial, a través de la exportación de capital humano, bienes y servicios con alto valor agregado.

En el año 2007 surge en la universidad el perfil de *Software* Educativo, cuyo progreso tuvo un punto de inflexión en el año 2010, con la creación del Centro de Tecnologías para la Formación (FORTES). Esta reestructuración se debe a la búsqueda de una mayor coherencia en las líneas temáticas de desarrollo, con el objetivo de optimizar los esfuerzos productivos y los resultados exportadores de la UCI.

Una de las líneas de desarrollo, por las que apuestan ALBET[1] y la UCI, es la de tecnologías para la formación. Esta abarca aplicaciones educativas, tanto para los niveles curriculares de la enseñanza preescolar, primaria, secundaria, preuniversitaria, como para las tecnologías de formación a distancia y semipresencial, tan en boga hoy día en el nivel superior y de postgrado.

FORTES desarrolla tecnologías que “permiten ofrecer servicios y productos para la implementación de soluciones de formación (…) a todo tipo de instituciones, con diferentes modelos de formación y condiciones tecnológicas (…) a partir de investigaciones que combinan los elementos pedagógicos y tecnológicos más avanzados.”(1)

El campo de la informática educativa (aplicaciones educativas y herramientas de teleformación) tiene un gran potencial de mercado, pues estas soluciones pueden resultar atractivas a instituciones tanto académicas como corporativas, privadas o gubernamentales, interesadas en implementar soluciones de formación con el uso de las Tecnologías de la Información y las Comunicaciones (TIC).

Al madurar los procesos de desarrollo de *software* y la experiencia exportadora de ALBET S.A. en el mercado venezolano, surge la premisa de ampliar el radio de acción a nuevos mercados. Este propósito plantea un gran reto, ya que implica orientar la producción a las demandas del mercado.

[1] ALBET Ingeniería y Sistemas S.A. Empresa que posee los derechos comerciales de todos los productos y servicios que se desarrollan en la UCI.

Sin embargo, no es tarea sencilla encontrar nichos de mercado en un mundo globalizado, donde tanto los adelantos tecnológicos como científicos, se incrementan por día en todas las esferas y las TIC ocupan un lugar protagónico(2). Las soluciones de informática educativa no están libres del exceso de oferta que implica, más que encontrar un cliente, fidelizarlo. En la sociedad moderna, donde la información es poder, un mejor tratamiento de la información implicará mejores decisiones en cuanto a mercados, clientes, productos y servicios.

Es por ello, que puede afirmarse que el éxito de una empresa nueva en el mercado, dependerá de la celeridad con que sea capaz de determinar qué productos o servicios, tendrán un valor agregado significativo para el cliente; así como su diseño integral, en cuanto a presentación, precio y promoción.(3)

A pesar de que hoy la actividad de mercadotecnia se ha organizado mejor con respecto a los años fundacionales de ALBET S.A., la gestión de la información y su correcta utilización en la toma de decisiones en el área es todavía insuficiente. La condición incipiente de estos procesos implica que las propuestas se generen a partir de información dispersa y aleatoria, y por tanto resultan de efectividad inexacta. En tal sentido, resulta imprescindible para FORTES el estudio de los diferentes actores determinantes en el proceso de toma de decisiones de los objetivos, estrategias y acciones necesarias para su concepción, presentación, promoción y distribución, o sea, la planeación mercadotécnica, a fin de ahorrar recursos y optimizar esfuerzo.

Actualmente en la UCI, el proceso de planeación mercadotécnica se realiza de forma empírica y operativa, no existe una normalización del proceso, ajustado a las características de la institución. Situación que trae como consecuencia que este proceso se lleve a cabo con lentitud y en la mayoría de los casos no se logre concretar un plan, en toda su coherencia interna de estrategia-objetivos-acciones.

Se hace, por tanto, necesario encontrar una solución que permita procesar automáticamente información verídica y confiable, necesaria para la definición de estrategias, objetivos y acciones, así como el control de su cumplimiento.

En este sentido se formula el siguiente **problema de investigación**:
¿Cómo reducir el tiempo y esfuerzo necesarios para los procesos de planeación mercadotécnica en el centro FORTES?

Se presenta como **objeto de estudio**: los sistemas de información de mercadotecnia y el **campo de acción**: el subsistema de datos internos para la planeación mercadotécnica del centro FORTES.

El **objetivo general** es: desarrollar una herramienta que automatice los procesos de planeación mercadotécnica del centro FORTES.

Teniendo en cuenta lo anterior, se propone la siguiente **idea a defender**: si se desarrolla una herramienta que automatice los procesos de gestión de información para la planeación mercadotécnica en el centro FORTES, se deberá reducir el tiempo y esfuerzo necesarios para esta actividad.

El objetivo general se articula en los siguientes **objetivos específicos**:

- Analizar el estado del arte sobre los Sistemas de Información de Mercadotecnia (SIM), sus conceptos asociados, así como las herramientas a emplear para su desarrollo.
- Describir la propuesta del sistema.
- Implementar y validar una aplicación que servirá de apoyo al proceso de planeación mercadotécnica.

Para dar cumplimiento a los objetivos se trazan las siguientes **tareas**:

- Revisión crítica de la documentación relacionada con la planeación mercadotécnica (en el mundo, Cuba y la UCI) y sus procesos.
- Revisión crítica de la documentación referente a los sistemas informáticos para la gestión de los procesos mercadotécnicos.
- Análisis crítico de soluciones para realizar la planeación mercadotécnica.
- Análisis crítico de las características principales de las diferentes metodologías de desarrollo de *software* para determinar la adecuada en la solución del problema.
- Selección de las herramientas y de los lenguajes de programación a utilizar.
- Selección del Entorno Integrado de Desarrollo (IDE) a utilizar.
- Definición y descripción de las funcionalidades y características que tendrá la aplicación.
- Realización del análisis y diseño del *software*.
- Implementación de las funcionalidades identificadas.
- Elaboración de los Casos de Prueba de Aceptación.
- Ejecución de Pruebas de Aceptación.

- Documentación de la investigación.

El presente documento está estructurado de la siguiente forma:

Capítulo 1 Fundamentación teórica

Se presentan los fundamentos teóricos que sustentan la investigación. Se realiza un estudio del estado del arte de las herramientas similares usadas para la gestión de información para mercadotecnia y, además, se hace referencia a las herramientas y tecnologías seleccionadas para el desarrollo de la aplicación que da solución al problema planteado.

Capítulo 2 Análisis de la solución propuesta

Se realiza el análisis de la solución propuesta. Se presenta la Lista de Reserva del Producto, en la que se priorizan los requerimientos funcionales a incluir. Además se describen las funcionalidades que el sistema debe ejecutar, mediante las Historias de Usuarios. También se muestra el modelo de datos, el modelo de despliegue, la explicación de la arquitectura del sistema y los patrones de diseño utilizados.

Capítulo 3 Implementación y prueba de la solución propuesta

Contiene el Plan de Release con la planificación y duración que tendrá cada Historia de Usuario. Se expone el contenido referente a las Tareas de Ingeniería. Se explica la realización de las Pruebas de Aceptación y la confección de los Casos de Prueba. Se detallan, además, los temas de tratamiento de errores, seguridad y estándares de codificación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

La *mercadotecnia* es la unión de una serie de tareas que una empresa realiza para lograr mantener o superar las ventas de sus productos en un mercado lleno de clientes, pero también de competidores. Según Philip Kotler (considerado por algunos como *padre de la mercadotecnia*) es: "el proceso social y administrativo por el cual los grupos e individuos satisfacen sus necesidades al crear e intercambiar bienes y servicios"(4). Otras definiciones plantean que *mercadotecnia* es la ciencia de satisfacer a los clientes al mismo tiempo que se obtienen las ganancias de los mismos, o sea, es una relación de intercambio mutuamente beneficiosa.

Como consecuencia del incremento de la competencia entre empresas y la necesidad de vender más productos, teniendo como premisa esencial la calidad y utilidad de los mismos; se hace imprescindible aumentar la capacidad de gestionar información útil para los procesos de mercadotecnia, de forma rápida. Esta ha sido la causa fundamental de la proliferación de los Sistemas de Información de Mercadotecnia (SIM) para recolectar y procesar información con mayor rapidez.

En la década de los 70, con la aparición de las primeras computadoras, que por primera vez permitían realizar varias operaciones al mismo tiempo, dando mayor velocidad al procesamiento de datos, comienza a popularizarse el uso de los SIM. Este auge continuó en los 80, cuando podían emitir extensos reportes llenos de datos, los cuales solo entendían expertos en el tema. Ya en la década del 90 los SIM generaban los reportes haciendo una presentación gráfica de los datos, los cuales podían ser almacenados y utilizados por la propia empresa o por otras en cualquier momento.(5)

Los SIM, en la actualidad, han dado un cambio verdaderamente sustancial. Lo que anteriormente constituían solamente entradas numéricas, ahora también incluyen información acabada, que luego es analizada y usada en los procesos de mercadotecnia.(5)

En el presente capítulo se describen los conceptos relacionados con la información y la gestión de la información que se consideraron pertinentes para respaldar la propuesta de solución, así como nociones acerca de los Sistemas de Información de manera general y en qué consiste un Sistema de Información

de Mercadotecnia en particular. Se expone además el análisis realizado para identificar los sistemas, herramientas y tecnologías utilizadas para la consecución del objetivo de este trabajo.

1.2. La información

En todo sistema de gestión de información, el elemento principal es la misma información, por lo que es necesario definirla y caracterizarla.

El concepto de información se define de diferentes maneras, se puede interpretar la información como un mensaje, como un patrón, como un impulso sensorial, como un efecto que produce una transformación, como una magnitud física, entre otras facetas.

Según la Real Academia de la Lengua Española, se define la información como la comunicación o adquisición de conocimientos que permiten ampliar o precisar los que se poseen sobre una materia determinada. Esta definición hace referencia a la utilidad de la información como fuente de conocimientos.

El autor Noel Angulo Marcial al analizar distintos conceptos la definió como “la significación que adquieren los datos como resultado de un proceso consciente e intencional de adecuación de tres elementos: datos del entorno, propósitos y contexto de aplicación y estructura de conocimientos del sujeto”(6). Manfredo Monforte considera que “la información es un dato o conjunto de ellos que, en un contexto determinado tienen un significado para alguien”(7), mientras que Eduardo Bueno Campos plantea que “es el conjunto de datos estructurados o elaborados que tienen significado para alguien en un momento y lugar”(8). Estas definiciones guardan varios elementos en común, reflejan que la información es un conjunto de datos relacionados, los cuales adquieren significado para alguien en un momento y lugar determinados.

Como ya se ha comentado, la información tiene distintas facetas y tendrá distintos significados en dependencia del punto de vista que se analice. Según la autora Dolores Vizcaya Alonso, haciendo un análisis de la información desde el punto de vista filosófico la define como “una parte de una reflexión, diferente de los factores materiales y energéticos, que es percibida por los sistemas materiales en una etapa organizativa definida y tan voluminosa que puede almacenarse, expresa en mensajes ordenados respecto a la probabilidad de uno u otro hecho entre la multitud de acontecimientos de una naturaleza dada”(9), definición esta que se centra fundamentalmente en el carácter informativo y/o explicativo de la información.

Otro concepto con el que se define a la información es en el ámbito de Sistemas de Información (SI), en el cual Börje Langefors define a la información como “cualquier mensaje o conocimiento que pueda usarse para posibilitar o mejorar una acción o decisión”(10), mencionando el valor de la información en la toma de decisiones en cualquier organización.

A partir de las definiciones anteriores, se puede concluir que la información es un conjunto de datos procesados que tiene importancia para el que la recibe, sirve de ayuda en la toma de decisiones además de influir en la obtención de nuevos conocimientos o reafirmar los ya existentes.

1.3. Gestión de Información

Antes de hablar de la Gestión de la Información (GI), primero se debe hacer alusión al concepto de “gestión”, la cual, el autor Pablo Murray citando a Roberto Faga dice que es la “actividad dirigida a obtener y asignar los recursos necesarios para el cumplimiento de los objetivos de la organización”.(11)

Varios son los autores que coinciden con que la información necesita una buena gestión, o sea, obtenerla de una fuente confiable, analizarla, procesarla y utilizarla en la toma de decisiones de una organización. Según Phil Bartle “la gestión de la información es el proceso de analizar y utilizar la información que se ha recabado y registrado para permitir a los administradores (de todos los niveles) tomar decisiones documentadas”.(12)

Muchas veces se tiene un gran cúmulo de información, pero se debe tener presente que contar con mayor cantidad de información no es sinónimo de estar mejor informado. Por eso es muy importante saber seleccionar (filtrar) la información que se posee, dándole un valor de importancia y fiabilidad, dependiendo de la fuente de la que procede. También se debe obtener distintos puntos de vista de la misma información para poder marcar un criterio propio de la misma.

Un problema que existe en nuestros días es que hay mucha información disponible, por lo que se debe hacer una buena gestión de esta para que resulte realmente útil, que aporte un valor significativo y esté disponible en el momento y a las personas que la necesiten, ya que esto sería de gran beneficio para cualquier organización; pues como se ha dicho, la información puede ser utilizada en la toma de decisiones y podría ayudar a que el desempeño de dichas organizaciones aumente.

A partir de lo descrito anteriormente, se puede decir que la gestión de la información implica:

- Determinar la información que se precisa
- Recoger y analizar la información
- Registrar y recuperar la información cuando sea necesaria
- Utilizarla
- Divulgarla

Eva María Méndez Rodríguez escribió: “Una organización que aspire a competir con éxito en el entorno actual debe aprender a aprender, debe hacer explícitos los procesos que permiten incorporar la información pertinente y relevante de que dispone, debe aplicar con creatividad e iniciativa las experiencias y el saber que ofrecen, en primer término, sus propios integrantes, sus proveedores, los grupos de interés y los clientes, es decir, la sociedad en su conjunto y más específicamente aquellos sectores en los que dicha organización actúa”(13), expresando de esta forma que la información debe ser un recurso del cual se debe valer una organización en su quehacer diario.

Muchas empresas hoy día se preocupan por implantar sistemas de gestión de información que ayuden a evitar tener una cantidad de información innecesaria que no aporte ningún valor y solamente poseer la que realmente se considera útil, contribuyendo de esta manera con un mejor desarrollo en el trabajo y decisiones.

El desarrollo alcanzado en los últimos años en aspectos tecnológicos, ha posibilitado la creación de nuevas soluciones que permiten procesar, almacenar, recuperar y divulgar información, la cual puede mejorar las estrategias de administración y de esta manera, elevar la eficiencia y la eficacia de las empresas.

Haciendo un resumen de las definiciones citadas anteriormente, y desde el punto de vista de la ingeniería, se pudiera entender por Gestión de la Información (GI) al proceso de extracción, manipulación, tratamiento, depuración, conservación y acceso a la información obtenida a través de una o varias fuentes, además de gestionar los derechos de los usuarios a usarla.

En el mundo actual donde la información ocupa un lugar tan importante en el quehacer diario de cualquier organización, se puede decir que la GI es un proceso básico. Al hacer una correcta GI se garantiza: un mayor control de la información existente, acceder a ella de manera más rápida, consultar la información de manera simultánea por distintos miembros de la organización, así como utilizarla en la toma de decisiones de la empresa, entre otros beneficios. Por tanto, hablar de una correcta GI garantiza poder alcanzar la excelencia en la organización.

1.4. Sistemas de Información

Una manera de gestionar la información en las empresas es a través de los Sistemas de Información (*SI*). Muchos son los autores que han dado definiciones sobre los SI, por ejemplo, Telchroew opina que “un sistema de información es una colección de personas, procedimientos y equipos diseñado, construido, operado y mantenido para colecciones, registros, procesar, almacenar, recuperar y mostrar información”(14), mientras que Miguel y Piattini lo definen como “un conjunto de elementos, ordenadamente relacionados entre sí de acuerdo con unas ciertas reglas, que aporta al sistema objeto (es decir a la organización a la cual sirve y que le marca las directrices de funcionamiento) la información necesaria para el cumplimiento de sus fines, para lo cual tendrá que recoger, procesar y almacenar datos, procedentes tanto de la misma organización como de fuentes externas, facilitando la recuperación, elaboración y presentación de los mismos”(14). Estos autores coinciden en que un *SI* está compuesto por personas, las cuales utilizan equipos que son capaces de procesar y llevar registros de información útil para la organización a la cual sirven. Por otra parte Langefors en un concepto más escueto lo define como “sistemas que suministran servicios de información”(14).

Según la Red Escolar Nacional de la República Bolivariana de Venezuela “un sistema manual de información puede llegar a ser ineficiente y frustrante, incluso en organizaciones pequeñas. Un sistema de información automatizado o basado en computadoras, es la integración de *hardware*, *software*, personas y datos”(15), o sea, hablar de un SI eficiente es sinónimo de hablar de un sistema de información computarizado donde intervienen los elementos mencionados anteriormente.

Por tanto, se puede decir que los elementos que componen un *SI* automatizado y que interactúan entre sí son: el equipo computacional (*hardware*), el o los programas que componen el *SI* (*software*), el recurso humano que utiliza el SI (personas) y la información que se procesa (datos).

Rosiry M. López cita a G. Davis, el cual dijo que “Un sistema de información es un sistema hombre/máquina integrado que provee información para el apoyo de las funciones de operación, gerencia y toma de decisiones en una organización”(16).

La autora Aymara Hernández Arias, cita a Murdick y Ross, los cuales definen un *SI* como un “grupo de gente, una serie de manuales y equipo de procesamiento de datos que escogen, almacenan, procesan y recuperan datos para disminuir la incertidumbre de la toma de decisiones, mediante el suministro de información a los gerentes cuando pueden utilizarla más eficientemente”(17). Además, Hernández Arias cita a Burch y Strater, que opinan que los *SI* son un “conjunto sistemático y formal de componentes, capaz de realizar operaciones de procesamiento de datos: dispositivos de entrada, dispositivos de almacenamiento de datos, medios de telecomunicación, equipo de procesamiento de datos, procedimientos y programas, métodos, documentación, modelos de manejo de datos y obviamente, analistas de sistemas de información para utilizar los elementos anteriores”(17); definiciones que plantean que los *SI* son (o deben ser) sistemas integrados por personas y equipos capaces de gestionar información que sea de utilidad para una organización.

Muchos son los autores que coinciden en que los SI son sistemas hombre-máquina, en los cuales las máquinas deben ayudar a las personas a recoger, procesar, almacenar y distribuir la información. También se hace necesario generar reportes de dicha información para que puedan ser utilizados en la planificación, control y toma de decisiones de la organización.

1.4.1. Componentes de un Sistema de Información

Como se ha explicado anteriormente, los *SI* automatizados son un conjunto de componentes (*hardware*, *software*, personas y datos) interrelacionados que permiten capturar, procesar, almacenar y distribuir información, facilitando la coordinación de las actividades de la organización, la toma de decisiones, la fijación de objetivos y el control.

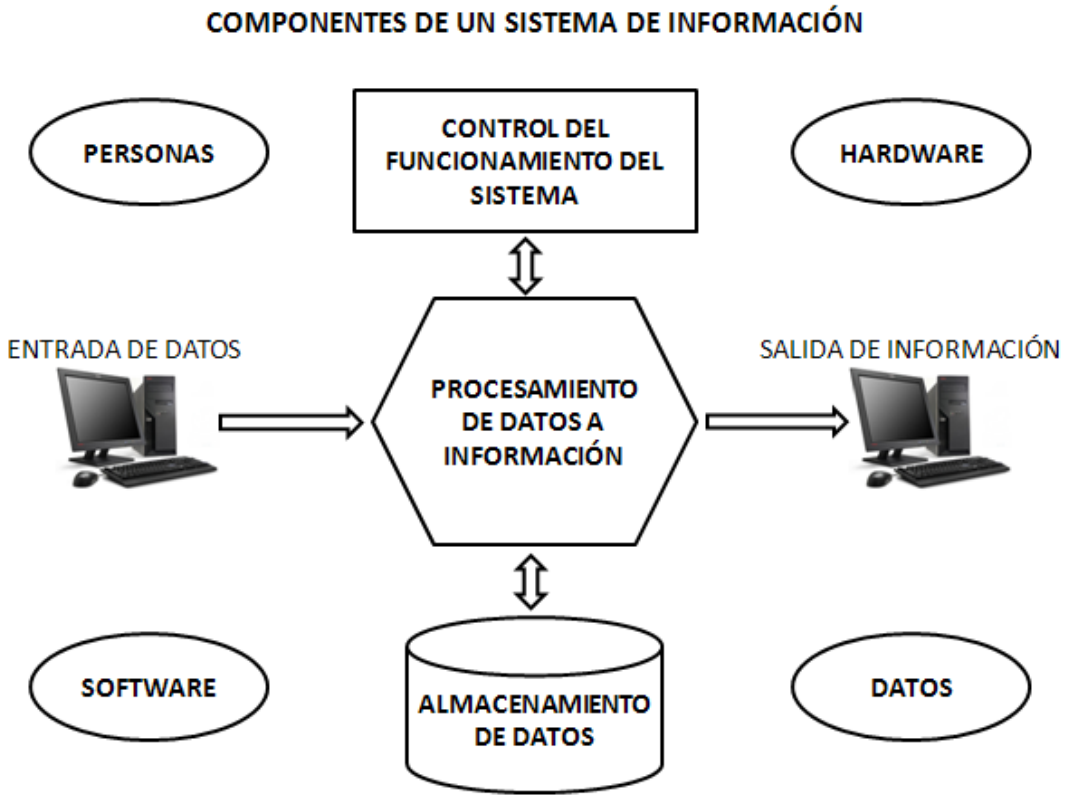


Figura 1: Componentes de un Sistema de Información

Tomado de http://www.rena.edu.ve/cuartaEtapa/Informatica/Tema10.html (2009). Adaptado por los autores

Los *SI*, para poder convertir los datos de entrada en productos de información, necesitan de otros subsistemas. Estos componentes serán descritos a continuación:(15)

Personas: personal encargado de la construcción, mantenimiento y uso del *SI*. Se dividen en dos grupos: **Usuarios finales** y **Especialistas.**

- **Usuarios finales**: son las personas que interactúan con el sistema o que reciben reportes generados por el mismo.

- **Especialistas**: entre los especialistas se encuentran los analistas que están encargados de actualizar, modificar o reconstruir el *SI*; los programadores crean los programas que forman parte

del *SI*; los administradores del sistema encargados de mantener el sistema en buenas condiciones y por último los capacitores que instruyen a los usuarios para utilizar el sistema.

Hardware: equipos, dispositivos y medios necesarios con los cuales el *SI* puede funcionar. Entre ellos se encuentran las computadoras, monitores, impresoras y medios de almacenamiento.

Software: componente lógico (rutinas e instrucciones) que conforman el *SI*. La aplicación que conforma un *SI* completo contiene subconjuntos de programas que se encargan de apoyar las distintas actividades propias de la organización.

Datos: unidades de información que son almacenadas y generadas en el transcurrir de la labor de la empresa. Los datos son almacenados en las bases de datos o bases de conocimientos.

1.5. Sistemas de Información de Mercadotecnia

En una empresa puede haber varios *SI* que gestionen distintos tipos de información, utilizada para diferentes fines (administrativos, gerenciales o comerciales). Uno de estos *SI* es el Sistema de Información de Mercadotecnia (SIM), cuya finalidad es generar, analizar, difundir, almacenar y recuperar la información que se utilizará en la toma de decisiones de mercadotecnia.

Según Kotler un SIM “consta de personal, equipo y procedimientos para reunir, clasificar, analizar, evaluar y distribuir información necesaria, oportuna y exacta para aquellos que toman decisiones de mercadotecnia para mejorar la planeación, ejecución y control”.(4)

El autor Rafael Muñiz González define el SIM como “un conjunto de relaciones estructuradas, donde intervienen los hombres, las máquinas y los procedimientos, y que tiene por objeto el generar un flujo ordenado de información pertinente, proveniente de fuentes internas y externas a la empresa, destinada a servir de base a las decisiones dentro de las áreas específicas de responsabilidad de mercadotecnia”.(18)

Fermín Garmendia y John Romeiro Serna definen un SIM como “una estructura estable y orientada al futuro, cuya finalidad es generar, procesar, almacenar y más tarde recuperar información para contribuir a la toma de decisiones en un programa de mercadotecnia”.(19)

Sergio Serna conceptualiza un SIM como “un conjunto organizado de procedimientos y métodos que continuamente recopila, clasifica, analiza, evalúa, almacena y distribuye información conveniente, oportuna y exacta, para uso de quienes toman decisiones”.(20)

A partir de lo planteado en las definiciones de los autores citados, se define un Sistema de Información de Mercadotecnia como un sistema que proporciona informaciones relacionadas con la actividad y que sirve de apoyo a los directivos en la toma de decisiones. Este SIM, aunque no necesariamente tiene que ser un sistema computarizado, puede constar de soluciones informáticas para la gestión de esta información que garantice una reducción en tiempo y mayor calidad.

1.5.1. Estructura de un Sistema de Información de Mercadotecnia

Según Kotler, un Sistema de Información de Mercadotecnia está compuesto por los siguientes elementos:

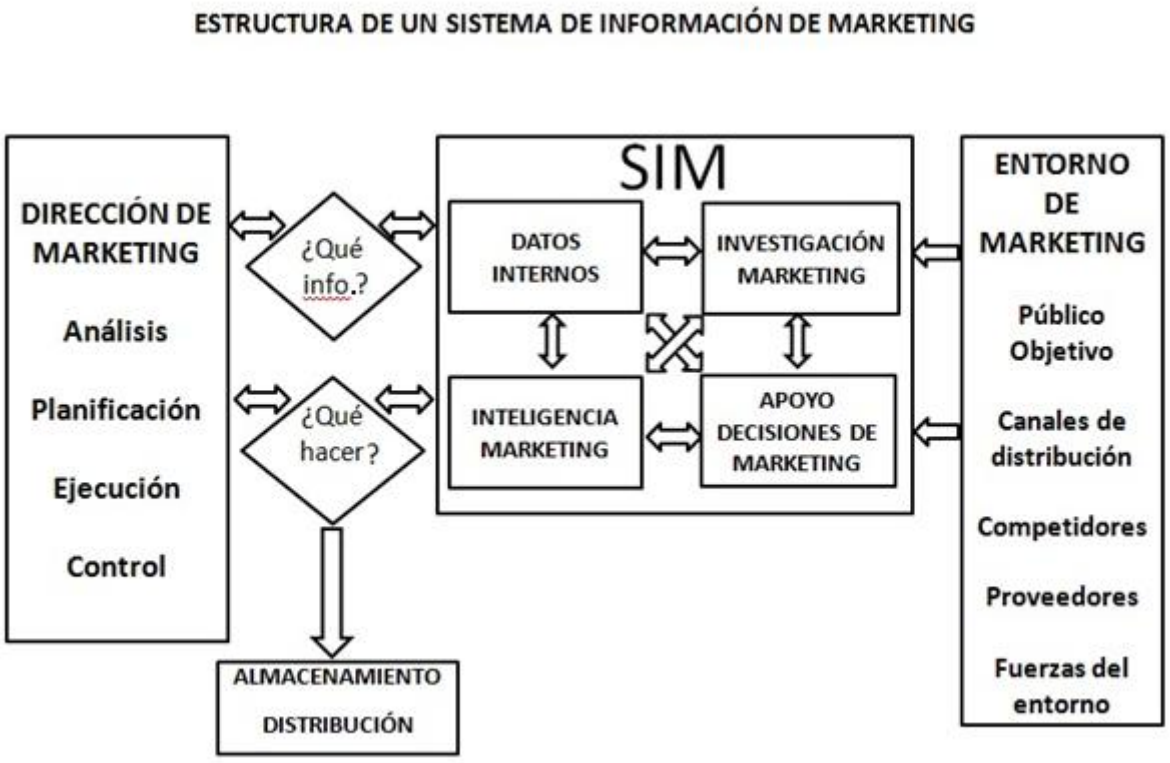


Figura 2: Componentes de un Sistema de Información de Mercadotecnia

Elaborado por los autores a partir de: Kotler, Philip. 2001

Los subsistemas incluidos en el SIM descrito por Kotler son detallados a continuación utilizando la referencia citada anteriormente:

Datos internos: se busca la información en los registros e informes internos de la compañía (estados financieros, registros de ventas, pedidos, inventarios y datos de los productos y de la compañía).

Inteligencia de mercadotecnia: la información es relativa a sucesos, acontecimientos y todo tipo de información cotidiana del entorno (nuevas reglamentaciones, tendencias demográficas y sociales, desarrollos tecnológicos, ambiente macroeconómico y comportamiento de los competidores) que pueda ayudar a tomar decisiones y preparar el Plan de Mercadotecnia.

Investigación de mercadotecnia: permite vincular al consumidor, al cliente y al público en general con la Dirección de Mercadotecnia. La información que se genera se usa para identificar y definir oportunidades y problemas de mercado.

Apoyo de decisiones de mercadotecnia: permite que la información reunida por los subsistemas anteriores, sea analizada con más profundidad, utilizando análisis estadísticos avanzados que estudian las relaciones entre series de datos y su confiabilidad estadística.

1.5.2. Sistemas de Información de Mercadotecnia en Cuba

En nuestro país ha sido una preocupación explícita desde 1987, el perfeccionamiento del sistema empresarial, lo cual se ha visto reflejado en sucesivos decretos leyes que han brindado soporte legal y apoyo normativo al cumplimiento de este objetivo. El último de estos documentos es el Decreto No. 281 del año 2007, del Comité Ejecutivo del Consejo de Ministros. En este, se trazaron los objetivos fundamentales, normas y procedimientos, exigencias y disposiciones para asegurar el desarrollo de la actividad de mercadotécnica. En el Artículo 653 de dicho documento se plantea:

“La implantación del sistema de mercadotecnia en la empresa permite establecer una relación dinámica entre la empresa y su entorno inmediato y mediato. Por esa razón se hace imprescindible, que la empresa estatal ejecute las siguientes acciones para garantizar la implantación del sistema de mercadotecnia:

- Identificar y seleccionar mercado, sociedad y clientes a los cuales están dirigidos sus productos y servicios.

- Elaborar base de datos actualizada de las características de los clientes actuales.
- Conocer y tener documentada las necesidades, deseos, expectativas y demandas de la sociedad y los clientes a la cual están dirigido sus productos y servicios.
- Documentar las opiniones de los clientes actuales sobre los productos y servicios que oferta y presta la empresa.
- Diseñar ofertas acorde a las expectativas de la sociedad y clientes a la cual están dirigidos sus productos y servicios.
- Dirigir su actividad para satisfacer las necesidades de la sociedad y de los clientes en productos y servicios; teniendo documentado e implantado el sistema de atención a clientes.
- Estudiar el tamaño del mercado donde opera la empresa y tener documentados y evaluados las acciones que se realizan para alcanzar un mayor posicionamiento en su mercado meta.
- Evaluar y aprobar los proveedores de la empresa, efectuando tal y como está establecido los pliegos de concurrencia.
- Participar en el proceso de conformación de contratos con los clientes y suministradores de la empresa.
- Garantizar que los productos y servicios que ofrece la empresa se distingan en el mercado y sean reconocidos por la sociedad por sus atributos (precio, prestaciones y calidad).
- Planear y proponer la fijación de precios.
- Planear y ejecutar la promoción y distribución de productos y servicios.
- Dar seguimiento y respuesta a las quejas, reclamaciones y sugerencias de los clientes.”(21)

El documento más adelante, en coherencia con la literatura especializada en el tema, plantea que el sistema deberá abarcar los siguientes aspectos:

- Sistema de datos internos.
- Sistema de inteligencia de mercadotecnia.
- Sistema de investigación de mercadotecnia.

- Sistema de apoyo a las decisiones de mercadotecnia (sistemas que permiten almacenar y recuperar información para la toma de decisiones).

En el presente trabajo de diploma, se hace referencia sólo al subsistema Datos internos de un SIM como un primer paso en la puesta en práctica de estos lineamientos, vinculándolo con el proceso de planeación para obtener, como entregable, un plan mercadotécnico correctamente estructurado. La implementación de todos los subsistemas implicaría un alcance que excede al de la presente investigación, en cuanto a tiempo y esfuerzo, ya que la complejidad del sistema en general, demandaría además la participación de especialistas en otras áreas del conocimiento como por ejemplo, almacenamiento de datos o inteligencia artificial.

1.6. Análisis de soluciones existentes

Para desarrollar una aplicación es necesario analizar las soluciones existentes tanto a nivel internacional como nacional que realicen funcionalidades similares a las de la solución planteada por la presente tesis. La temática de los Sistemas de Gestión de Información, principalmente los que manejan información de mercadotecnia, es poco trabajada en Cuba; por lo que se hace imprescindible la realización del análisis antes planteado.

A continuación se exponen características de soluciones existentes estudiadas.

1.6.1. Sistemas a nivel internacional

Los sistemas que gestionan información de mercadotecnia a nivel internacional, automatizan varios procesos que tienen que ver con esta área. Los cuales, en su mayoría, tienen relación con campañas de publicidad, mercadotecnia para internet y estudios específicos sobre las necesidades del cliente y por tanto, no se incorporan al sistema que propone este trabajo de diploma.

Los sistemas que se explican a continuación, presentan funcionalidades que automatizan procesos que se encuentran entre los que requiere el centro FORTES.

Plan Write for Marketing[2]

Plan Write for Marketing es un *software* encaminado a la elaboración de planes de mercadotecnia. Ayuda a identificar las características de los mercados de destino, los tamaños de mercado y factores competitivos. La guía de expertos ayuda a documentar de manera correcta cómo será el proceso de venta; paso clave para establecer las tácticas que permitan mejorar el presupuesto.(22)

Este sistema ayuda a identificar los competidores y la influencia de la empresa en el mercado. Además, permite describir las características generales de los segmentos de mercado que tienen una necesidad potencial de adquirir los productos o servicios de la empresa.(22)

Con toda la información que recoge, auxilia a los especialistas en la documentación de las estrategias a seguir en el plan, finalizando con la creación del documento final, el cual es impreso por el propio *software* o exportado a un fichero de Microsoft Word.(22)

Difiere de las necesidades del sistema al que hace referencia este trabajo en que no gestiona información relevante acerca de la empresa y los productos, elemento que es esencial para el centro FORTES. Es una aplicación propietaria, con un coste de 129.95 USD[3] para adquirir el sistema básico y 299.95 USD para una versión en la que se incluye una guía para industrias con tecnología de avanzada.

Zend Marketing

El *software* Zend Marketing permite realizar y controlar los planes de mercadotecnia paso a paso en la mitad de tiempo. Además, incluye ayuda e información sobre qué funciona mejor en cada uno. El programa permite realizar un análisis del producto, del cliente y de la competencia, para luego establecer objetivos, estrategias y acciones, lo que tiene como resultado planes efectivos, rápidos y centrados en las claves importantes. También posee una sección para controlar los resultados, ver lo que funciona y lo que no, lo que hay realizado, lo que queda pendiente, modificando fácilmente lo que sea necesario.

Este *software* difiere del sistema que propone esta investigación en que no permite gestionar los datos internos de los productos, servicios y clientes de la empresa. Además, es un software propietario. Solo la

[2] Traducción de los autores de: Business Resource Software Inc. http://www.brs-inc.com/marketing_plan.asp

[3] Siglas del dólar estadounidense.

versión beta es descargable sin costo desde internet, pero no permite usar ninguna de las funciones del módulo de control. La versión gratis disponible solo permite trazar tres objetivos y tres estrategias, lo que limita a la empresa. No exporta el plan de mercadotecnia en formato PDF, elemento fundamental para el centro FORTES. Para realizar las acciones que tiene bloqueadas la versión beta se debe adquirir la versión PRO[4], que tiene un costo de 75 euros.

1.6.2. Sistemas a nivel nacional

Luego de realizar una detallada búsqueda de la aplicación de los SIM en Cuba, sólo se encontraron ejemplos de los mismos en pocas empresas cubanas, principalmente en cadenas de tiendas que ofrecen servicios a minoristas como es el caso de la cadena de tiendas: Tiendas Recuperadoras de Divisas (TRD).

En el caso del SIM aplicado en la cadena TRD(23), incluye los subsistemas detallados en el presente documento: Datos internos, Inteligencia de mercadotecnia, Investigación de mercadotecnia y Apoyo de decisiones de mercadotecnia. De estos subsistemas, el único que consta de aplicaciones informáticas es el subsistema Datos internos. Estas aplicaciones no son *software* relacionados específicamente con el tema de la mercadotecnia, sino que más bien son bases de datos realizadas con el Sistema de Gestor de Base de Datos (SGBD) Microsoft Access del paquete de Microsoft Office. Las bases de datos con las que cuenta el subsistema son detalladas a continuación:

- Golden: es una aplicación que gestiona el control de los inventarios, cobros y pagos en las bases de almacenes, así como las ventas en diferentes puntos (tiendas y kioscos).
- Golden Data WareHouse (GDWH): utilizado para los reportes comerciales a nivel de división y cadena.
- Parte Diario de Ventas: permite el seguimiento de las ventas de la cadena.(23)

El resto de los subsistemas no cuentan con ninguna herramienta automatizada, solamente con procedimientos, procesos y personas que se encargan de gestionar la información, para lo cual se valen de un buzón de correo electrónico al que llegan, a través de diversas fuentes, alertas de amenazas, cambios, deficiencias u oportunidades.

[4] Zend Marketing Professional.

Existe una propuesta de realizar un SIM para el Centro de Informática Médica (CESIM) que permita "proporcionar la información interna y externa necesaria, precisa, oportuna, organizada, con facilidades para su procesamiento y análisis, como respaldo a la toma de decisiones efectivas en el desarrollo y comercialización de productos de Informática Médica".(23)

En la Universidad de la Ciencias Informáticas, la aplicación de las variables de mercadotecnia no se han difundido ampliamente. En los centros de desarrollo creados por la dirección de la Universidad, se cuenta con una persona encargada de la gestión de estos procesos, para los productos desarrollados y/o terminados. Actualmente no existe ninguna aplicación que ayude a esta persona a gestionar de forma automatizada la información necesaria para llevar a cabo los procesos de mercadotecnia, por lo que la recolección, análisis, almacenamiento y distribución de la información se realiza de manera manual o con ayuda de aplicaciones ofimáticas.

Esta situación se comporta de manera similar en la empresa ALBET S.A. de la UCI, en la que el equipo encargado de la mercadotecnia sólo consta de una herramienta informática utilizada para la gestión de los clientes: el CRM VTiger[5]; mientras que para el estudio de mercados, países y competidores no existe una aplicación informática que los asista. El VTiger se encarga de recopilar la mayor cantidad de información posible sobre los clientes para conocer las necesidades de los mismos y así poder adelantar una oferta y mejorar la calidad en la atención. Mejorar la oferta se refiere a brindar soluciones a los clientes que se adecuen a sus necesidades.

1.7. Herramientas y tecnologías de desarrollo

1.7.1. Lenguaje Unificado de Modelado (UML)

El Lenguaje Unificado de Modelado (UML por sus siglas en inglés) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de *software*. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear, configurar, mantener y controlar la información sobre tales sistemas. Sirve de apoyo a la mayoría de los procesos de desarrollo orientados a objetos, captando la información sobre la estructura estática y el comportamiento dinámico de un sistema(24). UML es uno de los lenguajes de modelado más utilizado

[5] Desarrollado por la corporación privada Vtiger. El sitio oficial se creó en el año 2003.

por la mayoría de las herramientas de modelado, tales como Rational Rose, Visual Paradigm, Microsoft Visio y ArgoUML.

1.7.2. Herramienta CASE

Las herramientas CASE (acrónimo de *Computer Aided Software Engineering*, en español: Ingeniería de Software Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la efectividad en el desarrollo de *software* y sistemas, en términos de productividad y calidad, contribuyendo a una reducción de costos de producción. Incluyen programas para el diseño de sistemas, ingeniería inversa, generadores de código y otros(25). Para determinar la herramienta CASE se estudiaron dos de las más utilizadas a nivel internacional: Rational Rose y Visual Paradigm.

Rational Rose

Rational Rose es la herramienta CASE desarrollada por los creadores de UML (Booch, Rumbaugh y Jacobson), que cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables. Permite especificar, analizar y diseñar el sistema antes de codificarlo, así como la generación de documentación en distintos formatos, como PDF y HTML.(26)

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

Visual Paradigm

Herramienta CASE que acelera el desarrollo de aplicaciones, sirviendo de intermediario visual entre arquitectos, analistas y diseñadores de *software*, mediante un ambiente de modelado superior que posibilita un trabajo más fácil y dinámico.(27)

Permite diseñar base de datos y generar sus esquemas (con el DDL del Sistema Gestor de Bases de Datos seleccionado). Es integrable con varios entornos de desarrollo como Eclipse, NetBeans y Microsoft Visual Studio, permitiendo aumentar la velocidad en el análisis, captura, plan, desarrollo, comprobación y

despliegue de los requisitos. Es multiplataforma y cuenta con una versión libre para la comunidad (*Comunity Edition*).(27)

Genera la documentación del sistema en los formatos PDF, HTML y el formato de documentos de Microsoft Word; además, permite importar proyectos de otras herramientas de modelado como Rational Rose, Erwin y Microsoft Visio. Soporta la revisión ortográfica, brindando sugerencias para los idiomas: inglés y español.(27)

Selección de la herramienta

Finalmente se escogió Visual Paradigm, principalmente por ser multiplataforma. Otro criterio de selección es que genera documentación en diferentes formatos, como PDF y HTML. La versión que se utilizará es la 6.4, pues la UCI paga por su licencia de uso.

1.7.3. Metodología de desarrollo de *software*

Se entiende por metodología de desarrollo una colección de documentación formal referente a los procesos, las políticas y los procedimientos que intervienen en el desarrollo del *software*. La finalidad de una metodología de desarrollo es garantizar la eficacia (por ejemplo: cumplir los requisitos iniciales) y la eficiencia (por ejemplo: minimizar las pérdidas de tiempo) en el proceso de generación de *software*.

Metodologías tradicionales

Teniendo en cuenta la filosofía de desarrollo de las metodologías, aquellas con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado, reciben el apelativo de metodologías tradicionales o pesadas. En las metodologías tradicionales se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto de *software*. Se centran especialmente en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada. Además, las metodologías tradicionales no se adaptan adecuadamente a los cambios, por lo que no son métodos adecuados cuando se trabaja en un entorno, donde los requisitos no pueden predecirse o bien pueden variar.(28)

Metodologías ágiles

Este tipo de metodologías están diseñadas para ser utilizadas en proyectos pequeños o de corta duración. Con ellas se consigue simplificar grandemente los procesos de desarrollo de *software* tradicionales, sin que esto afecte las buenas prácticas que garantizan la calidad del producto obtenido.(29)(30)

Son de gran utilidad en muchos proyectos actuales, ya que en varios casos se necesita reducir el tiempo de desarrollo del *software*. Las metodologías ágiles dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del *software* con iteraciones muy cortas.(29)(30)

Principios del manifiesto ágil

- Se valora al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas. Las personas son el principal factor de éxito de un proyecto de *software*.
- Desarrollar un *software* que funcione es más importantes que conseguir una buena documentación. Los documentos deben ser cortos y centrarse en lo fundamental.
- La colaboración con el cliente es esencial, más que la negociación de un contrato. Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo.
- Responder a los cambios, más que seguir estrictamente un plan. La planificación no debe ser estricta, sino flexible y abierta.

En el proceso de desarrollo del sistema al que se refiere este trabajo de diploma, se hace necesaria la obtención de resultados concretos en ciertos períodos de tiempo permitiendo al cliente evaluar el nivel de madurez que va alcanzando la aplicación. Esto le permite obtener ventajas competitivas para adaptar correctamente la aplicación a los cambios que van surgiendo. Los procesos ágiles promueven un desarrollo sostenible basándose en la auto-organización del equipo de desarrollo por lo que es de vital importancia el trabajo unido de los clientes con los desarrolladores. Después de analizar las características del proyecto a realizar, el equipo de desarrollo y el ambiente de trabajo, se decide utilizar metodologías ágiles para el desarrollo del sistema, teniendo en cuenta que se cuenta con un tiempo reducido para la construcción del sistema, que el cliente forma parte del equipo de desarrollo y la necesidad de centrarse más en la programación que en la documentación.

SCRUM

SCRUM es un proceso en el que se aplican de manera regular un conjunto de mejores prácticas para trabajar colaborativamente en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio sobre la manera de trabajar de equipos altamente productivos.(31)

En SCRUM se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados a corto plazo, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.(31)

En esta metodología un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones de un mes natural y hasta de dos semanas, si así se necesita). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite.(31)

Ciclo de vida de SCRUM

- Planeamiento: Establecer la visión, definir expectativas y aseguramiento financiero.
- Montaje: Identificar otros requerimientos y priorizar las tareas para la primera iteración.
- Desarrollo: Implementar un sistema listo para entrega en una serie de iteraciones de treinta días llamadas “corridas” (*sprints*).
- Liberación: El propósito es el despliegue operacional.

XP (eXtreme Programming)

La Programación Extrema es una metodología ligera de desarrollo de *software* que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado.(32)

Características de XP

- Desarrollo iterativo e incremental.
- Establece mejoras en cada una de las iteraciones del proyecto de forma incremental.

- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión.
- Programación por parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto.
- Frecuente interacción del equipo de programación con el cliente o usuario.
- Corrección de todos los errores antes de añadir una nueva funcionalidad y realización de entregas frecuentes.
- Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento.
- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto.
- Simplicidad en el código.

SXP (SCRUM y eXtreme Programming)

SXP es un híbrido cubano de metodologías ágiles que tiene como base las metodologías SCRUM y XP, las que permiten actualizar los procesos de desarrollo de *software* para el mejoramiento de su producción. Consta de cuatro fases: Planificación-Definición, Desarrollo, Entrega y Mantenimiento, cada una desglosada en flujos de trabajo y actividades que generan artefactos. Esta metodología ayuda a fortalecer el trabajo en equipo, enfocados en una misma dirección, permitiendo además seguir de forma clara el avance de las tareas a realizar, a partir de la inserción de procedimientos ágiles que permitan actualizar los procesos de *software* para el mejoramiento de la producción, aumentando el nivel de interés del equipo.(33)

Selección de la metodología a utilizar

La metodología de desarrollo de *software* que se utilizará es SXP. Es un híbrido cubano que tiene como base las metodologías ágiles SCRUM y XP. Ayuda a fortalecer el trabajo en equipo, permitiendo darle seguimiento a las tareas a partir de la inserción de procedimientos ágiles. La metodología SXP funciona bajo los principios del manifiesto ágil, lo que garantiza una gran capacidad de respuesta ante los cambios en los requisitos a lo largo del proceso de desarrollo. Además, se establece una estrecha relación entre el

cliente y el equipo de desarrollo, evitando errores del sistema por mala definición de los requisitos, por lo que se pueden obtener resultados concretos desde la primera iteración. Por otra parte, posibilita la atención continua a realizar un diseño de excelencia y permite que los procesos definidos se vayan mejorando de forma incremental hasta obtener una pieza de *software* con la mayor calidad posible para los clientes, asegurando el cumplimiento de sus necesidades.

Además, SCRUM posibilita una planificación y organización del proyecto centrada en los principales aspectos del sistema, evitando pérdida de tiempo en tareas secundarias. Esta planificación es materializada por la metodología XP, obteniéndose de esta forma un proceso de *software* completo y exitoso.

1.7.4. Sistema Gestor de Base de Datos (SGBD)

Un Sistema Gestor o Manejador de Base de Datos (SGBD) se define como "el conjunto de programas que administran y gestionan la información contenida en una base de datos"(34), o sea, programas que permiten a los usuarios crear, manipular y mantener una base de datos (BD).

PostgreSQL

PostgreSQL[6] es un SGBD multiplataforma ampliamente conocido y usado en entornos de *software* libre. Distribuido bajo la licencia Berkeley Software Distribution (BSD), que permite su libre uso, modificación y redistribución con la única condición de mantener el derecho de autor del *software* original a sus autores.(35)

Es un sistema de base de datos objeto-relacional, que cuenta con una arquitectura que ha ganado gran reputación por su confiabilidad, estabilidad y mantenimiento de la integridad de los datos. Incluye extensiones de orientación a objetos, permitiendo definir un nuevo tipo de tabla a partir de una previamente definida.(35)

Posee interfaces de programación nativas para diversos lenguajes y plataformas como Java, .NET, C/C++, Perl, Python, y es capaz de ejecutar procedimientos almacenados en muchos de estos lenguajes,

[6] Traducción de los autores de: PostgreSQL: *The world's most advanced open source database*. http://www.postgresql.org

y en su propio lenguaje PL/pgSQL. Incluye en su librería estándar cientos de funciones que abarcan desde operaciones matemáticas y de cadenas básicas, hasta criptografía y compatibilidad con Oracle.(35)

Cuenta con un rico conjunto de tipos de datos y provee un *framework* que permite a los desarrolladores definir y crear tipos de datos personalizados, conjuntamente con las funciones que estos soportarán y los operadores que definirán su comportamiento. Presenta otras características como el control de concurrencia multi-versión (MVCC), los puntos de recuperación dentro de las transacciones, replicación asincrónica, conjunto de caracteres internacionales, entre otras.(35)

MySQL

MySQL es un Sistema Gestor de Base de Datos relacional que tiene un comportamiento dual, pues se distribuye bajo una licencia GNU GPL, pero si se desea distribuir alguna aplicación que use MySQL es necesario pagar por la licencia comercial. Este gestor de base de datos es uno más usado en el mundo del *software* libre, debido a su gran rapidez. Posee varias librerías y herramientas además de su fácil instalación y configuración. Tiene un gran aprovechamiento de sistemas multiprocesador por su implementación multi-hilo y dispone de borrado multitablas.

Selección del Sistema Gestor de Base de Datos

Teniendo en cuenta las características del sistema a desarrollar y la importancia de una SGBD robusto para una correcta manipulación de los datos, incluso en situaciones críticas y la seguridad que deben tener los mismos, se escoge el PostgreSQL para el sistema propuesto. Es importante señalar que este SGBD tiene una implementación multiproceso, lo que posibilita una mejor disponibilidad de la información ante cualquier fallo de algún proceso evitando que el servidor deje de brindar servicios. PostgreSQL es el SGBD de código abierto más potente del mercado y es multiplataforma. Para el desarrollo de la aplicación se utilizará la versión 8.4.

1.7.5. Servidor web

Un servidor web es un programa ejecutado continuamente en una computadora (generalmente llamada servidor), el cual se encarga de contestar de forma adecuada las peticiones de ejecución que le hará un usuario mediante la red; entregando como resultado una página web (código HTML) o información de todo tipo de acuerdo a los comandos solicitados, implementando el protocolo HTTP.

Apache HTTP Server

Servidor web desarrollado por la Fundación de Software Apache (*Apache Software Foundation*). Es "un esfuerzo colaborativo de desarrollo de *software* que apunta a crear una implementación del código fuente de un servidor (web) HTTP que sea robusto, comercial, de libre disponibilidad y con muchas funcionalidades"[7].

El servidor HTTP Apache ofrece un mejor uso de las especificaciones de HTTP existentes, es libre y de distribución gratuita, e implementa funcionalidades demandadas con frecuencia, tales como:

- ✓ personalizar las respuestas a los errores y problemas.
- ✓ establecer ficheros que son devueltos por el servidor en las respuestas a los errores y problemas, realizando diagnósticos "en el momento" para los usuarios y el administrador.
- ✓ aceptar un número ilimitado de directivas Alias y Redirect.
- ✓ admitir servidores *"multi-home"*, que permiten distinguir entre las peticiones de diferentes direcciones IP.(36)

En mayo del 2010, Apache hospedaba 112 663 533 sitios web en Internet (54,68% del total); por debajo de él se encontraba Microsoft con un 25,27% del total.(37)

Internet Information Services (IIS)

Internet Information Services o IIS es un servidor web y un conjunto de servicios para el sistema operativo Microsoft Windows. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS. Es propietario y su uso está restringido por una licencia.(38)

Selección del servidor web

Para el desarrollo de la aplicación se utilizará Apache como servidor web, por la eficiencia que este brinda. Además es multiplataforma, tiene una gran comunidad que lo respalda y su uso no está restringido por una licencia. Es un servidor altamente configurable y fácil de usar. Es muy extensible, permitiendo a cualquier usuario implementar módulos para alguna función específica, además de poseer abundante documentación para usuarios de poca experiencia. Se utilizará la versión 2.2.6.

[7] Traducción de los autores de: *Apache Software Foundation: About the Apache HTTP Server Project*. http://httpd.apache.org/ABOUT_APACHE.html

1.7.6. Lenguajes de programación

Lenguaje del lado del servidor

PHP 5

PHP (acrónimo de *Hipertext Preprocesor*) es un lenguaje *script* interpretado que corre del lado del servidor en la Arquitectura Cliente - Servidor. Es uno de los lenguajes preferidos por los desarrolladores de páginas web dinámicas.(39)

Por ser libre y disponer de acceso pleno al código fuente, además de ser gratuita su distribución, ha posibilitado el surgimiento y consolidación de una gran comunidad de programadores que le dan soporte y contribuyen a la mejora de su código. Posee gran eficiencia cuando se combina con el servidor web Apache, soportando millones de visitas diarias sin complicaciones.(39)

PHP dispone de una conexión propia a la mayoría los SGBD. Además de MySQL, puede conectarse directamente a las base de datos de PostgreSQL, Oracle, SyBase, entre otras compatibles con ODBC (del inglés *Open Connectivity Stardard*, Estándar de Conectividad Abierta de Bases de Datos).(39)

Es multiplataforma e incorpora una gran cantidad de funciones integradas para realizar útiles tareas relacionadas con la web. Es gratuito; la sintaxis de PHP se basa en otros lenguajes de programación, principalmente en C y Perl. Está disponible para una gran cantidad de sistemas operativos diferentes. Se puede escribir código PHP en todo los sistemas operativos gratuitos del tipo Unix, como Linux y FreeBSD, versiones comerciales de Unix, como Solaris e IRIS, así como en las diferentes versiones de Windows. Su código funciona sin necesidad de aplicar ninguna modificación a los diferentes sistemas que ejecute PHP.(39)

Perl

Perl es un lenguaje de programación interpretado que toma características de los lenguajes C, shell (sh), AWK, sed, Lisp y, en un grado inferior, de otros lenguajes de programación. Es un lenguaje de propósito general originalmente desarrollado para la manipulación de texto y que ahora es utilizado para un amplio rango de tareas incluyendo administración de sistemas, desarrollo web, programación en red y desarrollo de GUI. Sus principales características son que es fácil de usar, soporta tanto la programación

estructurada como la programación orientada a objetos y la programación funcional, tiene incorporado un poderoso sistema de procesamiento de texto y una enorme colección de módulos disponibles.(40)

Java

Se trata de un lenguaje orientado a objetos, originalmente desarrollado por un grupo de ingenieros de la empresa Sun Microsystems. Este lenguaje fue utilizado por Netscape posteriormente como base para JavaScript. Si bien su uso se destaca en el web, sirve para crear todo tipo de aplicaciones.

Java ha sido concebido, desde sus orígenes, como un lenguaje capaz de producir código totalmente transportable(41). Se ha desarrollado hasta ser un lenguaje totalmente independiente de la plataforma y a la vez potente. Esa filosofía y su facilidad para crear aplicaciones para redes TCP/IP ha hecho que sea uno de los lenguajes más utilizados en la actualidad.

Selección del lenguaje a utilizar

Teniendo en cuenta las funciones que brindan los lenguajes estudiados anteriormente, así como sus ventajas y desventajas se define para el sistema a desarrollar la utilización del lenguaje de programación del lado del servidor PHP 5. Su distribución es gratuita, es libre, multiplataforma y posee gran eficiencia cuando se combina con el servidor web Apache, soportando millones de visitas diarias sin complicaciones. Dispone conexión propia a la mayoría los SGBD.

Lenguaje del lado del cliente

XHTML

XHTML (Lenguaje de Marcado de Hipertexto Extensible) es una versión más estricta y limpia de HTML, que nace precisamente con el objetivo de reemplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML. XHTML extiende HTML 4.0 combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos.(42)

El lenguaje XHTML es muy similar al lenguaje HTML. De hecho, XHTML no es más que una adaptación de HTML al lenguaje XML. Técnicamente, HTML es descendiente directo del lenguaje SGML, mientras que XHTML lo es del XML (que a su vez, también es descendiente de SGML). Las páginas y documentos creados con XHTML son muy similares a las páginas y documentos HTML.(43)

CSS

Las hojas de estilo se basan en un código desarrollado por el World Wide Web Consortium (www.w3c.org) mediante el cual es posible definir anticipadamente el estilo que tendrá cada etiqueta HTML en la página. Es un lenguaje de hojas de estilos creado para controlar la presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para la creación de páginas web complejas.(44)

Hojas de Estilo en Cascada (*Cascading Style Sheets*), es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.(45)

JavaScript

JavaScript es un lenguaje de programación interpretado, es decir, que no requiere compilación, con una sintaxis semejante a la de los lenguajes Java y C. Se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas que se muestran al usuario. Las sintaxis de código escritas en JavaScript se pueden probar directamente en cualquier navegador.

AJAX

Acrónimo de *Asynchronous JavaScript And XML* (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (*Rich Internet Applications*). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.(46)

1.7.7. *Framework* PHP

Los *framework* agilizan el desarrollo de *software*, proporcionando una estructura definida la cual ayuda a crear aplicaciones con mayor rapidez; además, facilitan las tareas de mantenimiento del sitio gracias a la organización durante el desarrollo de la aplicación.

Los *framework* son desarrollados con el objetivo de brindar a los programadores y diseñadores una mejor organización y estructura a sus proyectos, utilizando la Programación Orientada a Objetos (POO), permitiendo la reutilización de código.

Symfony

El *framework* Symfony[8] ha sido un proyecto de código abierto por varios años y se ha convertido en uno de los *framework* PHP más populares gracias a sus excelentes características y amplia documentación. Es un *framework* completo, una biblioteca de clases coherente escrito en PHP. Proporciona una arquitectura, componentes y herramientas a los desarrolladores para crear aplicaciones web complejas en poco tiempo. Su objetivo es acelerar la creación y mantenimiento de aplicaciones web, y para reemplazar las tareas repetitivas de codificación.(47)

El pequeño número de requisitos previos hace a Symfony fácil de instalar en cualquier configuración; solo se necesita Unix o Windows con un servidor web y PHP instalado. Es compatible con casi todos los sistemas de base de datos. Es adaptable a las políticas y arquitecturas propias de cada empresa u organización, lo que permite su uso en aplicaciones empresariales.(47)

Usar Symfony es tan natural y fácil para las personas que usan PHP y los patrones de diseño de aplicaciones de internet que la curva de aprendizaje se reduce significativamente. Los desarrolladores pueden aplicar los principios de desarrollo ágil. Al elegir Symfony se obtienen los beneficios de una activa comunidad de código abierto. Es totalmente libre y está publicado bajo la licencia MIT de *software* libre.(47)

CodeIgniter

Es un *framework* hecho en PHP 4 para la creación de páginas web dinámicas. Es de código abierto y cuenta con una serie de librerías que apoyan el desarrollo de aplicaciones web, además de proponer la

[8] Traducción de los autores de: Symfony | *Web PHP Framework*. http://www.symfony-project.org

mejor manera de usarlas si se quiere obtener los mejores resultados. Este *framework* está implementado siguiendo el patrón arquitectónico Modelo – Vista – Controlador, muy usado en la creación de aplicaciones web.

Zend Framework

Zend Framework se trata de un *framework* para desarrollo de aplicaciones web y servicios web con PHP, brinda soluciones para construir sitios web modernos, robustos y seguros. Además es de código abierto y trabaja con PHP 5.(48)

Selección del *framework*

Por las ventajas que ofrece el *framework* Symfony se decidió utilizarlo para el desarrollo del sistema propuesto. Tiene la capacidad de poner en funcionamiento una gran cantidad de procesos con una sola línea de comando. Está desarrollado en PHP 5 para obtener todas las ventajas que ofrece el lenguaje. El proyecto Symfony incluye un tutorial en el que se explica paso a paso el desarrollo de una aplicación utilizando las técnicas de desarrollo ágil de aplicaciones. Posee una extensa comunidad que lo respalda y abundante documentación. Permite generar de forma automática los formularios de acceso a datos utilizando la base de datos definida para el sistema a desarrollar, formularios que incluyen validación automatizada y relleno automático de datos (*"repopulation"*), lo que asegura la obtención de datos correctos y mejora la experiencia de usuario. Los datos incluyen un mecanismo de escape que permite una mejor protección contra ataques por datos corruptos, además permite hacer cambios en la aplicación sin necesidad de reiniciar el servidor. Igualmente, la solución propuesta se integrará al SIM de la empresa ALBET S. A. (actualmente en desarrollo), cuya arquitectura definió el uso del *framework* Symfony por parte del equipo de desarrollo. Para la implementación se utilizará la versión 1.4.3.

1.7.8. *Framework* JavaScript

Las librerías de los lenguajes de programación son un elemento tan necesario para el programador, como puede ser la sintaxis del lenguaje o el compilador. Es por esta causa que, a veces, un lenguaje bien desarrollado no puede prescindir de un gran número de librerías que ahorran la tarea de escribir funciones comunes, que pueden necesitar los programadores, reduciendo o eliminando la pérdida de tiempo que conllevaría implementarlas. JavaScript, al igual que cualquier otro lenguaje, tiene librerías que varían

desde inmensas bibliotecas como Prototype, jQuery o Ext JS, hasta otras de menor tamaño y aplicación más específica.

jQuery

jQuery es una rápida y concisa librería JavaScript, que simplifica el control de eventos, la animación y las interacciones AJAX para el desarrollo web en corto tiempo. Implementa una serie de clases (de programación orientada a objetos) que permite al programador trabajar sin preocuparse por cuál navegador está usando el usuario, pues funcionan de igual forma en diferentes plataformas.(49)

Así pues, este *framework* JavaScript, ofrece una infraestructura con la que los desarrolladores tienen mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Este *framework* tiene licencia para uso en cualquier tipo de plataforma, personal o comercial.(49)

Es una de las mejores opciones, si de escoger un *framework* JavaScript se trata, pues es un producto con muy buena aceptación por parte de los programadores y un grado de penetración en el mercado muy amplio. Además, es un producto serio, estable, bien documentado y con un gran equipo de desarrolladores a cargo de la mejora y actualización del *framework*. Posee una dilatada comunidad de desarrolladores de *plugins* o componentes, lo que facilita encontrar soluciones ya creadas en jQuery para implementar, por ejemplo, galerías, votaciones y efectos diversos.(49)

Prototype

Prototype es un *framework* de JavaScript que tiene como objetivo facilitar el desarrollo de aplicaciones web dinámicas. Implementa las técnicas AJAX y su potencial es aprovechado al máximo cuando se desarrolla con Ruby on Rails.(50)

Ext JS

Ext JS es una librería JavaScript ligera y de alto rendimiento, compatible con la mayoría de los navegadores, que nos permite crear páginas e interfaces web dinámicas.(51)

Selección del *framework*

Para la solución planteada se escogió jQuery en su versión 1.5. Este *framework* JavaScript, es el más utilizado por los desarrolladores web, es libre y gratuito y además permitirá la integración que se espera, de la aplicación planteada, con el SIM de ALBET S. A.

1.7.9. Entorno Integrado de Desarrollo (IDE)

Integrated Development Environment (Entorno Integrado de Desarrollo): se refiere a los programas o interfaces visuales de programación que se usan para interactuar con el compilador y/o *debugger* de un lenguaje de programación determinado, de forma amigable para el desarrollador. Este brinda una serie de posibilidades que le permiten hacer un mejor uso de sus conocimientos y explotar de forma óptima el lenguaje de programación en el cual está trabajando. Por lo general consta de un editor de texto, con opciones para tratar el código que se edita y para mandar comandos al compilador y al *debugger*, sobre acciones a realizar, sobre el código que se está editando. Por lo general son aplicaciones GUI, aunque también pueden estar basados en ambientes sobre texto. Ejemplos de IDEs conocidos son: Visual Studio, Borland C++ Builder, Borland Delphi, Eclipse, NetBeans, KDevelop y Aptana.

NetBeans

NetBeans[9] es un entorno de desarrollo para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para otros lenguajes de programación. Existe, además, un número importante de módulos para extender el NetBeans IDE, el cual es un producto libre y gratuito sin restricciones de uso.(52)

Principales características:

Permite:

- creación de proyectos PHP.
- integración con Symfony y ZendFramework.
- integración con PHP Unit Testing.
- depuración de PHP.
- integración con Sistemas de Control de Versiones.

Zend Studio

Zend Studio es un completo entorno de desarrollo integrado para el lenguaje de programación PHP. Se ha diseñado para maximizar la productividad de los desarrolladores por lo que les permite desarrollar y mantener el código en el menor tiempo posible, resolver los problemas de aplicación de forma rápida y mejorar la colaboración en equipo. Está escrito en Java, y está disponible para las plataformas Microsoft

[9] Traducción de los autores de: Netbeans. http://netbeans.org/

Windows, Mac OS X y GNU/Linux(53). Se trata de un *software* privativo, lo que contrasta con la idea de que PHP es *software* libre.

Selección del IDE

Para el desarrollo del sistema se determinó utilizar como IDE NetBeans 6.9 por ser de código abierto, es compatible con Symfony y sus funcionalidades son extensibles mediante la instalación de paquetes. Tiene un potente motor para la conexión a bases de datos, es fácil de instalar y se ejecuta en varias plataformas como Windows y Linux. No se utilizará Zend Studio porque Symfony no se integra a este IDE, no siendo así con NetBeans.

1.8. Conclusiones parciales

- Para sentar las bases de la construcción de un SIM, debe definirse la información como un recurso empresarial que debe ser gestionado como tal.

- Los SIM no necesariamente implican la presencia de sistemas informáticos, pero el uso de las nuevas tecnologías en las condiciones actuales, cobra importancia en su concepción.

- Las aplicaciones existentes a nivel internacional son privativas y no satisfacen por completo, las necesidades de la aplicación propuesta; mientras que las que existen a nivel nacional no se ajustan a los requerimientos del cliente.

- Para el desarrollo del sistema se empleará como herramienta CASE Visual Paradigm, la metodología de desarrollo de *software* SXP, como Sistema Gestor de Base de Datos PostgreSQL, como servidor web se seleccionó Apache HTTP Server. Los lenguajes de programación a utilizar serán: PHP para el lado del servidor y JavaScript para el lado del cliente. El *framework* PHP seleccionado es Symfony y para la capa de presentación el *framework* JavaScript jQuery y como IDE se determinó el uso de NetBeans.

CAPÍTULO 2: ANÁLISIS DE LA SOLUCIÓN PROPUESTA

2.1. Introducción

La creación de sistemas informáticos es un proceso en el que se desarrollan artefactos que, luego de ser integrados, posibilitan responder a las necesidades del cliente. En este proceso se identifican varias etapas, que van desde la declaración del problema y los requerimientos del sistema, hasta las pruebas y la liberación del mismo. Las metodologías de desarrollo proveen una guía que ayuda al grupo de desarrollo a organizarse en tiempo, actividades y artefactos a desarrollar. Dentro de las metodologías de desarrollo, las ágiles "dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas (…) mostrando su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad".(30)

En este capítulo se describirá la propuesta de Sistema de apoyo a la Planeación Mercadotécnica, analizando sus funcionalidades y elementos principales utilizando la metodología ágil SXP. De esta última, se incluyen algunos de sus artefactos: la *Lista de Reserva del Producto*, las *Historias de Usuarios* arquitectónicamente significativas y el *Modelo de Diseño*. Igualmente se incluirán la arquitectura y patrones de diseño utilizados, así como la descripción del *Modelo de Datos* del sistema.

2.2. Breve descripción del sistema

El centro de desarrollo de *software* FORTES requiere automatizar los procesos relacionados con el *marketing* que se desarrollan actualmente de forma manual, adicionándole nuevas funcionalidades identificadas por los especialistas en el área. En respuesta a estas necesidades se desarrolla ***Royston***[10], un sistema de gestión de información para mercadotecnia destinado al centro FORTES. El sistema es una aplicación web, elemento que aporta la posibilidad de conexión remota por varios usuarios, quienes pueden acceder desde cualquier terminal con acceso al servidor principal, en el que se hospeda el sistema.

[10] Tomado del nombre científico de la Palma Real cubana, caracterizada por su robustez y fortaleza.

El sistema permite gestionar información referida a elementos internos y externos al centro de desarrollo. De la información interna se recogen datos referidos a los productos, servicios, proyectos, departamentos, marcas de los productos y servicios, licencias y estándares. Los clientes, servicios externos, productos externos, organizaciones sin fines de lucro, empresas, ferias y fuentes de información, forman parte de la información externa: datos relacionados con el estudio de la competencia y el entorno. También se gestionan los planes de mercadotecnia, para lo cual es necesario definir correctamente objetivos, estrategias y acciones. Sobre estos elementos se almacena información de fuentes confiables, que es empleada para mantener actualizados los datos esenciales y garantizar el correcto funcionamiento del proceso de planeación mercadotécnica. El sistema permite la elaboración y generación de reportes e informes, así como imprimir estos informes y exportarlos en formato PDF.

La aplicación es fácil de usar, tiene una interfaz amigable, con colores suaves y pocos elementos visuales, lo que facilita que el usuario centre su atención en los elementos propios del trabajo con el *software* y no distraiga su atención.

2.3. Arquitectura del sistema

"La arquitectura de *software* (...) es la estructura de un sistema, que contiene elementos del *software*, las propiedades externas visibles de esos elementos, y las relaciones entre ellos"(54). Una correcta definición de la arquitectura permite facilitar el diseño y construcción de los componentes y sus relaciones.

Entre los patrones arquitectónicos empleados en la construcción de aplicaciones web se encuentra el Modelo-Vista-Controlador (MVC), que separa el modelo de datos, la lógica de control y las interfaces de usuario. Particularmente el *framework* Symfony utiliza como arquitectura base el patrón MVC (ver Figura 3). Fabien Potencier (2009), creador del *framework*, describe las capas del MVC en Symfony como se describe a continuación:

- La ***capa del modelo*** define la lógica de negocio, incluyendo la base de datos. Symfony guarda todas las clases y archivos relacionados con el modelo en el directorio 'lib/model/'.(55)

- La ***vista*** es lo que utilizan los usuarios para interactuar con la aplicación (los gestores de plantillas pertenecen a esta capa). En Symfony la capa de la vista está formada principalmente por plantillas en PHP. Estas plantillas se guardan en varios directorios llamados 'templates/'.(55)

- El ***controlador*** es un bloque de código que realiza llamadas al modelo para obtener los datos y se los pasa a la vista para que los muestre al usuario. Todas las peticiones se canalizan a través de los controladores frontales, que delegan todo el trabajo en las acciones, las que se agrupan en módulos.(55)

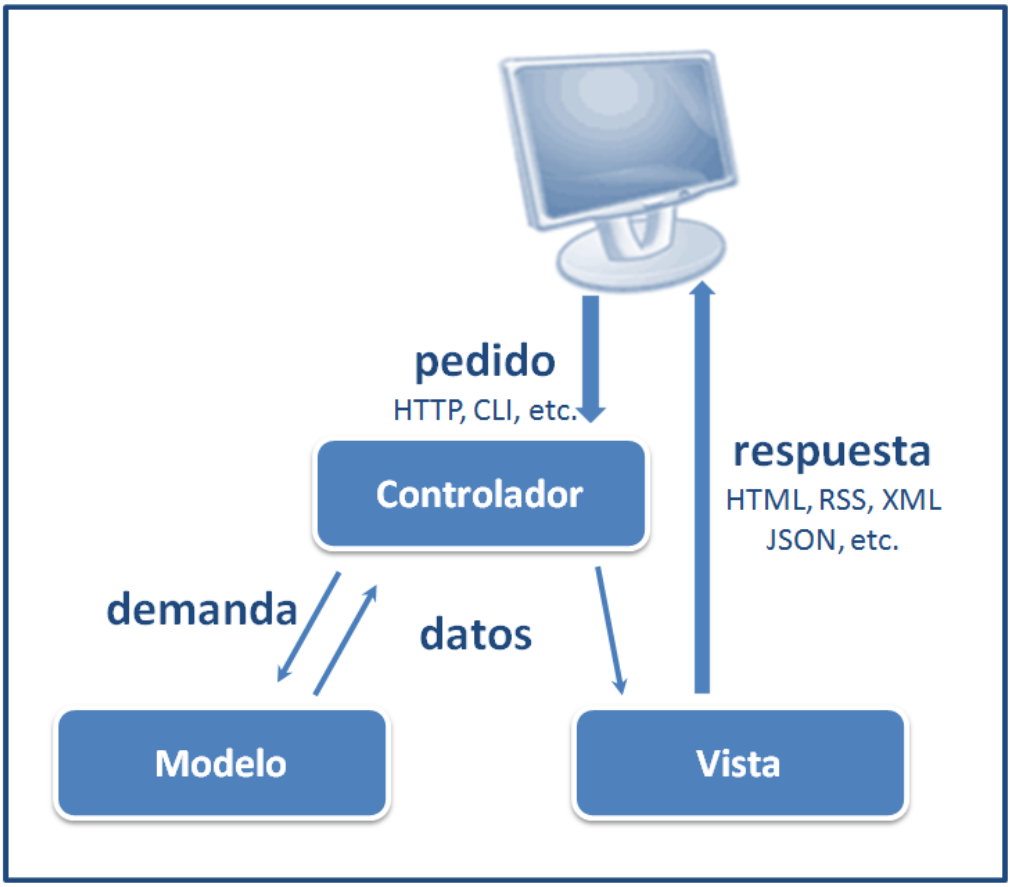


Figura 3: Modelo-Vista-Controlador para Symfony
Traducido por los autores de Potencier, Fabien. *Practical Symfony* (2009).

2.4. Patrones de diseño

Los patrones de diseño nombran, abstraen e identifican los aspectos claves de un diseño estructurado común, que lo hace útil para la creación de diseños orientados a objetos reutilizables. Definen una posible solución correcta para un problema de diseño dentro de un contexto dado, describiendo las cualidades invariantes de todas las soluciones.(56)

2.4.1 Patrones GRASP

GRASP es el acrónimo de *General Responsibility Assignment Software Patterns* (Patrones generales de *software* para asignar responsabilidades), nombre elegido para indicar la importancia de captar (*grasping*) estos principios, si se quiere diseñar eficazmente el *software* orientado a objetos.

Experto: este patrón permite la asignación de una responsabilidad a la clase que cuenta con la información necesaria para cumplirla (experto de información). Expresa la "intuición" de que los objetos realizan operaciones relacionadas con la información que poseen. Brinda beneficios como la conservación del encapsulamiento, el soporte de un bajo acoplamiento y una alta cohesión.(56)

Creador: este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. Su propósito fundamental es encontrar un creador que debe ser conectado con el objeto producido en cualquier evento. El patrón Creador proporciona soporte a un bajo acoplamiento.(56)

Bajo acoplamiento: estimula asignar una responsabilidad a una clase, de modo que su colaboración no incremente tanto el acoplamiento con otras clases, al nivel que produzca los resultados negativos propios de un alto acoplamiento. El bajo acoplamiento soporta el diseño de clases más independientes, que reducen el impacto de los cambios y también reutilizables, que aumentan la oportunidad de una mayor productividad.(56)

Alta cohesión: una clase con alta cohesión posee un número relativamente pequeño de responsabilidades, relacionadas entre sí por sus funcionalidades. Dicha clase colabora con otras clases para compartir el esfuerzo si la tarea es grande. Este patrón mejora la calidad y facilidad del diseño, genera un bajo acoplamiento y promueve la reutilización.(56)

Controlador: la mayor parte de los sistemas reciben eventos de entrada externa. En estos casos hay que elegir controladores que manejen esos eventos de entrada. Este patrón ofrece una guía para tomar decisiones apropiadas en la elección de los controladores de eventos. Su utilización propicia que las operaciones del sistema se manejen en la capa de dominio de los objetos, y no en la de presentación.(56)

2.4.2 Patrones GOF

Los patrones GOF constituyen un catálogo de 23 patrones de diseño básicos publicados por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides en el libro *Design Patterns: Elements of Reusable Object-Oriented Software*. GOF significa Gang of Four (Banda de los Cuatro), forma en que se conoce al grupo de autores.

Algunos de los patrones GOF empleados en la implementación del *framework* Symfony son *Abstract Factory* (Fábrica abstracta), *Factory Method* (Método de fabricación), *Decorator* (Decorador) y *Singleton* (Instancia Única), los que se describen a continuación.

Abstract Factory: proporciona una interfaz que permite trabajar con objetos de distintas familias de manera que las familias no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando.(57)

Factory Method: permite que una clase difiera la instanciación a las subclases (son éstas las que deciden qué clase instanciar).(57)

Decorator: permite asignar nuevas responsabilidades a un objeto dinámicamente. Es una alternativa a la creación de subclases por herencia.(57)

Singleton: el empleo de este patrón permite asegurar que una clase posee solamente una instancia, proporcionando un punto de acceso global al objeto instanciado.(57)

2.5. Lista de Reserva del Producto

La gestión de proyecto en la metodología ágil SXP define artefactos para describir las características del producto, integrando al cliente como parte importante del equipo de desarrollo, debido a las grandes posibilidades de que existan cambios en los requerimientos. Es dentro de la gestión de proyecto que se genera el artefacto Lista de Reserva del Producto (LRP).

La LRP es una lista priorizada que define el trabajo a realizar en el proyecto. En ella se incluyen los requerimientos sobre el producto, por lo que puede crecer y modificarse a medida que se obtiene más

conocimiento acerca del producto y del cliente. El completamiento de la lista (acompañado de los cambios en el entorno y el producto) asegura que el producto sea lo más correcto, útil y competitivo posible.(33)

En la LRP se incluyen los requerimientos que demanda el cliente del sistema, separados por su prioridad (muy alta, alta, media o baja), la que indica la necesidad que tiene el sistema de dicha funcionalidad, siendo las muy altas, funcionalidades indispensables para cubrir las necesidades del cliente; mientras que las bajas son aquellas que se desean incluir, pero sin las que el sistema podría, todavía, ser funcional y útil. Además, se enuncian los requerimientos no funcionales especificados por el cliente como necesarios.

La tabla de la LRP incluye los siguientes campos: ***Prioridad***, que contiene los niveles de prioridad de las funcionalidades incluidas (divididas en 4 grupos: Muy Alta, Alta, Media y Baja); ***Ítem***, que contiene la numeración secuencial de las funcionalidades que están contenidas en cada grupo de prioridad; y ***Descripción***, que contiene la descripción de la funcionalidad. Además, se incluyen los ***Requerimientos No Funcionales*** (que no llevan estimación).

En la LRP del sistema al que hace referencia el presente trabajo, se incluyeron, entre las funcionalidades de prioridad muy alta, aquellas que tienen relación con la gestión de información básica sobre los elementos internos y sobre algunos elementos externos. Estas funcionalidades de prioridad muy alta serán las consideradas como arquitectónicamente significativas. Es válido aclarar que esta LRP sufrió cambios en el transcurso del proyecto, adicionando, modificando y eliminando funcionalidades, hasta quedar como se muestra en la Tabla 1.

Tabla 1: Lista de Reserva del Producto (LRP)

Prioridad	**Ítem ***	**Descripción**
Muy Alta		
	1	Listar servicios externos
	2	Mostrar servicio externo
	3	Adicionar servicio externo
	4	Modificar servicio externo
	5	Eliminar servicio externo
	6	Buscar servicio externo
	7	Listar productos externos
	8	Mostrar producto externo
	9	Adicionar producto externo
	10	Modificar producto externo
	11	Eliminar producto externo
	12	Buscar producto externo
	13	Listar departamentos
	14	Mostrar departamento
	15	Adicionar departamento
	16	Modificar departamento
	17	Eliminar departamento
	18	Buscar departamento
	19	Listar productos internos
	20	Mostrar producto interno
	21	Adicionar producto interno
	22	Modificar producto interno
	23	Eliminar producto interno
	24	Buscar producto interno
	25	Listar clientes
	26	Mostrar cliente
	27	Adicionar cliente
	28	Modificar cliente
	29	Eliminar cliente
	30	Buscar cliente
	31	Listar servicios internos
	32	Mostrar servicio interno
	33	Adicionar servicio interno
	34	Modificar servicio interno

	35	Eliminar servicio interno
	36	Buscar servicio interno
	37	Listar proyectos
	38	Mostrar proyecto
	39	Adicionar proyecto
	40	Modificar proyecto
	41	Eliminar proyecto
	42	Buscar proyecto
	43	Listar planes de *marketing*
	44	Mostrar plan de *marketing*
	45	Crear plan de *marketing*
	46	Modificar plan de *marketing*
	47	Eliminar plan de *marketing*
	48	Autenticar usuario
	49	Listar usuarios
	50	Crear usuario
	51	Modificar usuario
	52	Eliminar usuario
	53	Buscar usuario
	54	Adicionar objetivo
	55	Eliminar objetivo
	56	Adicionar estrategia
	57	Eliminar estrategia
	58	Adicionar acción
	59	Eliminar acción
Alta		
	1	Listar empresas externas
	2	Mostrar empresa externa
	3	Adicionar empresa externa
	4	Modificar empresa externa
	5	Eliminar empresa externa
	6	Buscar empresa externa
	7	Listar mercados externos
	8	Mostrar mercado externo
	9	Adicionar mercado externo

	10	Modificar mercado externo
	11	Eliminar mercado externo
	12	Buscar mercado externo
	13	Listar fuentes de información
	14	Mostrar fuente de información
	15	Adicionar fuente de información
	16	Modificar fuente de información
	17	Eliminar fuente de información
	18	Buscar fuente de información
	19	Listar ferias
	20	Mostrar feria
	21	Adicionar feria
	22	Modificar feria
	23	Eliminar feria
	24	Buscar feria
	25	Listar marcas de los productos y servicios internos
	26	Mostrar marca de los productos y servicios internos
	27	Adicionar marca de los productos y servicios internos
	28	Modificar marca de los productos y servicios internos
	29	Eliminar marca de los productos y servicios internos
	30	Buscar marca de los productos y servicios internos
	31	Exportar plan de marketing
	32	Listar licencias
	33	Mostrar licencia
	34	Adicionar licencia
	35	Modificar licencia
	36	Eliminar licencia
	37	Buscar licencia
	38	Listar estándares
	39	Mostrar estándar
	40	Adicionar estándar
	41	Modificar estándar
	42	Eliminar estándar
	43	Buscar estándar
Media		
	1	Mostrar documento de objetivos

	2	Mostrar documento de estrategias
	3	Mostrar documento de acciones
	4	Mostrar informe de control
Baja		
	1	Imprimir informes generados por los especialistas de mercadotecnia
Requisitos No Funcionales		
	1	Facilidad de uso por parte de los usuarios: el sistema debe presentar una interfaz amigable que permita la fácil interacción con el mismo y llegar de manera rápida y efectiva a la información buscada. Debe, además, ser una interfaz de manejo cómodo que posibilite a los usuarios sin experiencia una rápida adaptación.
	2	Especificación de la terminología utilizada: el sistema debe adaptarse al lenguaje y términos utilizados por los clientes en la rama abordada con vista a una mayor comprensión por parte del cliente de la herramienta de trabajo.
	3	Uso de metáforas intuitivas en las interfaces, lo que hace más fácil la interacción con la herramienta por parte del usuario.
	4	Menús: El sistema debe presentar una serie de menús, que permitan el acceso rápido a la información por parte de los usuarios, aprovechando así las potencialidades de estas estructuras.
	5	La seguridad de la base de datos está a nivel de roles, con el fin de mantener la integridad de los datos en función del acceso de cada uno de ellos, trayendo consigo además la protección de la información.
	6	El sistema debe contar con un grupo de políticas de accesibilidad a las diferentes funcionalidades del mismo en dependencia del nivel de autorización que presente un usuario determinado.
	7	El sistema debe soportar una conexión simultánea de varios usuarios.
	8	El sistema debe brindar como apoyo una ayuda que permita al usuario orientarse sobre lo que debe hacer.
	9	Se utilizará un servidor de base de datos con SGBD PostgreSQL 8.4 o superior bajo un sistema operativo basado en GNU/LINUX.
	10	El servidor de aplicaciones Web será Apache HTTP 2.2 o superior.
	11	Se utilizará la versión 5.x del intérprete de PHP.
	12	El servidor de base de datos deberá tener como mínimo 512 MB de RAM.
	13	Interfaz web: la interfaz debe ser sencilla con colores suaves a la vista y sin cúmulo de imágenes u objetos que distraigan al cliente del objetivo.
	14	La comunicación entre el cliente y el servidor de aplicaciones es a través del protocolo HTTP.

2.6. Historias de Usuario

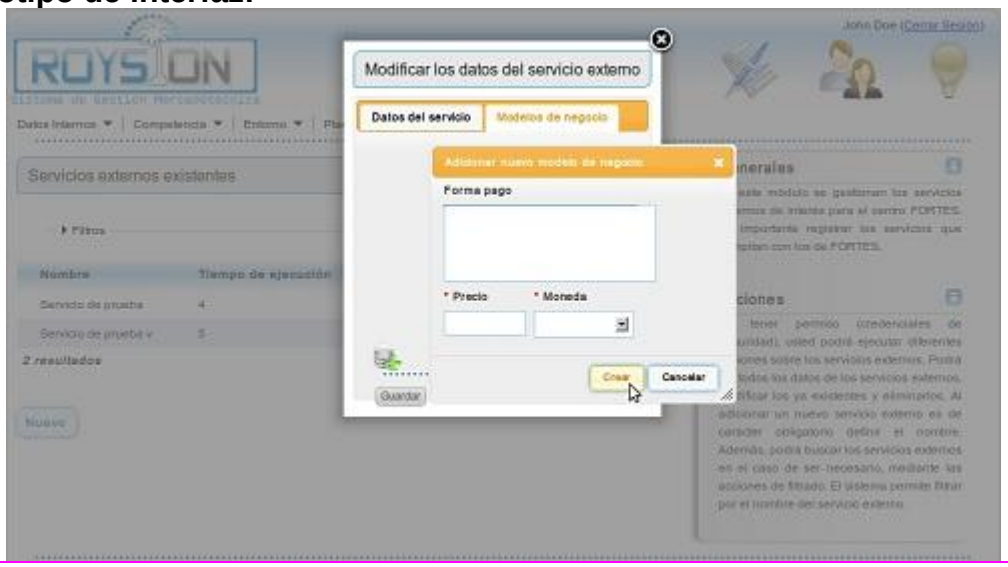
En el Proceso Unificado de Software (RUP), los rectores del desarrollo de los artefactos en el análisis, diseño, implementación y prueba, son los casos de uso. De forma análoga en SXP existen las historias de usuarios (HU), único documento de requisitos generado en la metodología.

Las HU son escritas por los clientes, quienes especifican y describen los requisitos del *software* en forma de tareas que el sistema debe hacer, utilizando un lenguaje natural, sin formato predeterminado y en pocas líneas de texto. Guían la construcción de otros artefactos incluidos en la metodología (Tareas de Ingeniería, Plan de *Releases* y Pruebas de Aceptación), siendo utilizadas para estimar tiempos de desarrollo.(33)

En las tablas se incluyen los siguientes campos: ***Número***, que contiene el identificador de la HU; ***Nombre Historia de Usuario***, que contiene el nombre que identifica a la HU; ***Modificación de Historia de Usuario Número***, que contiene la cantidad de modificaciones que se han realizado a la HU; ***Usuario***, nombre del programador encargado de implementar la HU; ***Iteración Asignada***, número de la iteración en la que se desarrollará la HU; ***Prioridad en Negocio***, tipo de prioridad de la HU (‘Muy Alta’, ‘Alta’, ‘Media’ o ‘Baja’); ***Puntos Estimados***, valor que describe la cantidad de semanas estimadas para completar la HU; ***Riesgo en Desarrollo***, al desarrollar la HU (‘Alto’, ‘Medio’ o ‘Bajo’); ***Descripción***, breve descripción del proceso que define la HU; ***Observaciones***, comentarios aclaratorios relacionados con la HU; y ***Prototipo de Interfaz***, que contiene la imagen de una de las interfaces de usuario relacionadas con la HU.

En la Tabla 2, se muestra la Historia de Usuario Gestionar servicios externos, priorizada como ‘Muy Alta’, es decir: arquitectónicamente significativa. El resto de las HU se incluyen en el Anexo 1.

Tabla 2: HU Gestionar servicios externos

Historia de Usuario	
Número: HU01	**Nombre Historia de Usuario:** Gestionar servicios externos
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Roberto Carlos Aballe O.	**Iteración Asignada:** Primera
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 1 Semana
Riesgo en Desarrollo: Medio	**Puntos Reales:** 1 Semana
Descripción: La Historia de Usuario consiste en listar, mostrar, adicionar, modificar, eliminar y buscar servicios externos.	
Observaciones: Los servicios externos son aquellos ofrecidos por empresas competidoras. La búsqueda se realizará por el nombre del servicio externo.	
Prototipo de interfaz: 	

2.7. Modelo de diseño

A consecuencia de la aplicación del patrón arquitectónico MVC, Symfony 1.4.3 propone un modelo de diseño por el que, por lo general, se rigen las aplicaciones web desarrolladas sobre él (ver Figura 4).

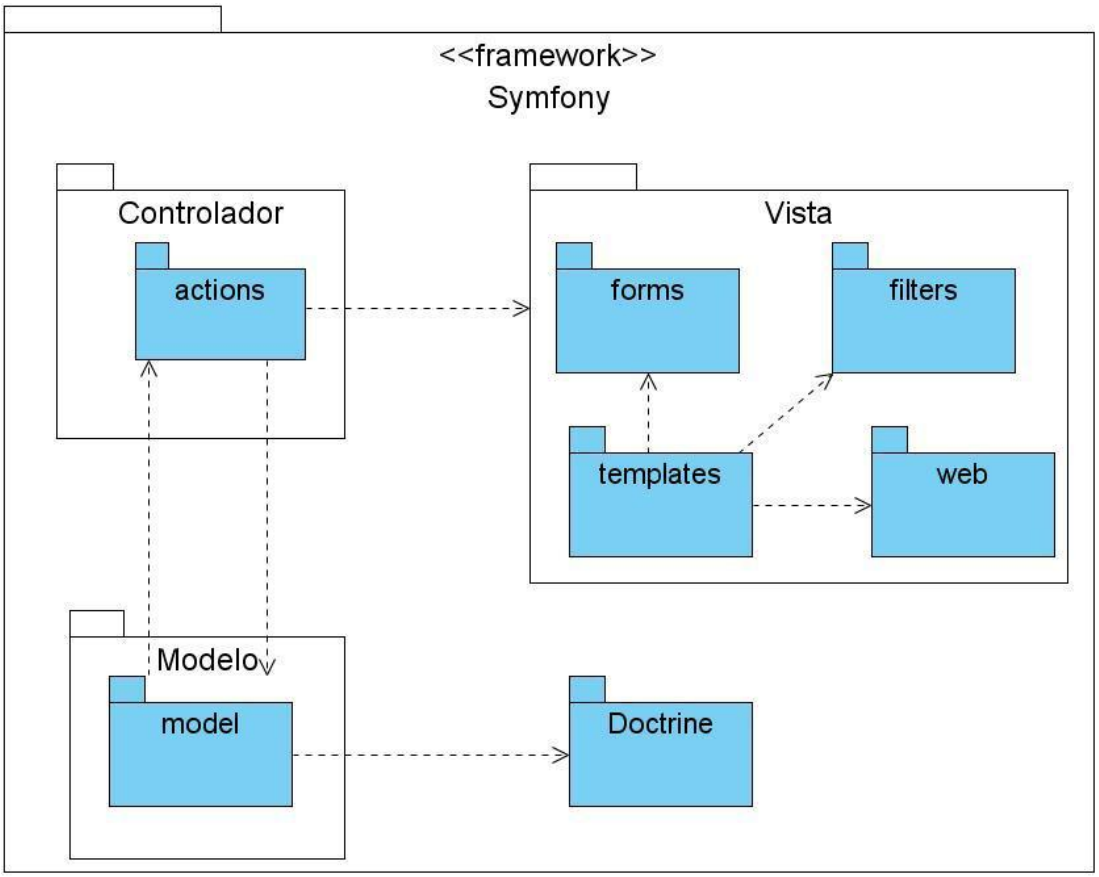


Figura 4: Modelo de Diseño basado en el *framework* Symfony

En la capa de la Vista se tienen varios paquetes: el paquete *'templates'* que agrupa los ficheros encargados de generar la interfaz gráfica, compuestos por código HTML y PHP (conocidos como plantillas), y que se corresponden con una acción del Controlador; los paquetes *'forms'* y *'filters'* que contienen los formularios y filtros, respectivamente, incluidos en las plantillas y procesados en las acciones; y el paquete *'web'* donde se encuentran las imágenes, hojas de estilo en cascada (CSS) y ficheros JavaScript (como los de la librería jQuery) utilizados por las plantillas para su visualización.

La capa del Controlador consta de un paquete llamado *'actions'*, donde se crea una clase 'nombreMóduloActions' que define todas las acciones que debe ejecutar cada módulo creado. Esta clase hereda de sfActions que es el controlador frontal de la aplicación. Dentro de esta clase las acciones son

responsables de actualizar el Modelo, y a su vez generar cambios en la Vista debido a la obtención de nuevos datos procedentes del Modelo.

La capa del Modelo está formada por un paquete llamado *'model'*, donde se encuentra una colección de clases generadas por el ORM del *framework*. Cada tabla de la base de datos posee dos clases en el modelo: 'NombreTabla' (representa un registro) y 'NombreTablaTable' (representa la tabla completa). Estas clases generadas interactúan con el ORM (Doctrine) para llevar a cabo las operaciones necesarias sobre la base de datos.

El modelo de clases del diseño describe la realización física de las historias de usuario, centrándose en cómo los requisitos funcionales y no funcionales tienen impacto en la aplicación a desarrollar. Facilita la abstracción de la implementación del sistema y, es de ese modo, el artefacto fundamental de entrada a las actividades de implementación. A continuación se muestra el modelo de clases del diseño de la HU Gestionar servicios externos:

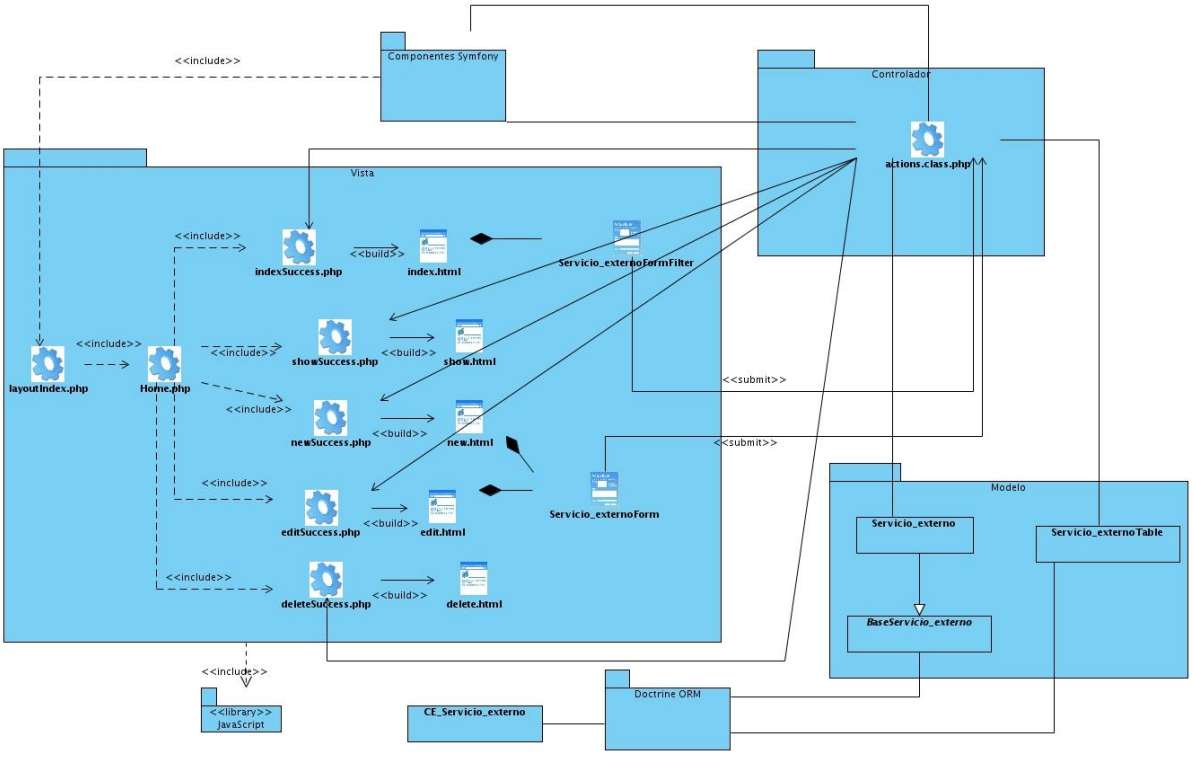


Figura 5: Modelo de clases del diseño de HU01

2.8. Modelo de datos

Aunque la modelación y gestión de la base de datos no se especifica dentro de la metodología SXP, sí es necesario abordar el tema debido a la importancia del mismo en la concepción y desarrollo de un Sistema de Gestión de Información (SGI). El almacenamiento de los datos es una de las actividades fundamentales en los SGI y forman parte, junto a la recopilación, el procesamiento y la distribución, de las tareas fundamentales que persigue. Este almacenamiento se realiza utilizando bases de datos cuyo empleo, a decir de Castilla Plaza, "supone consideraciones sobre su propio diseño, incluyendo la transmisión de los datos y la interacción con el ordenador."(58)

La base de datos fue diseñada utilizando Visual Paradigm 6.4, y en total está integrada por 36 tablas. Para una mejor visualización y entendimiento del modelo relacional se hace necesario remitirse al Anexo 3. Además, se incluye en el Anexo 4 una descripción de las tablas de la base de datos.

2.9. Modelo de despliegue

El modelo de despliegue es un modelo físico que muestra la distribución de los diferentes componentes de *software* que se ejecutan en los distintos nodos en tiempo de ejecución, los cuales pueden tener capacidad de procesamiento (**<<executionEnvironment>>**) o no (**<<device>>**).(56)

El sistema, para su explotación, quedará distribuido de la siguiente manera:

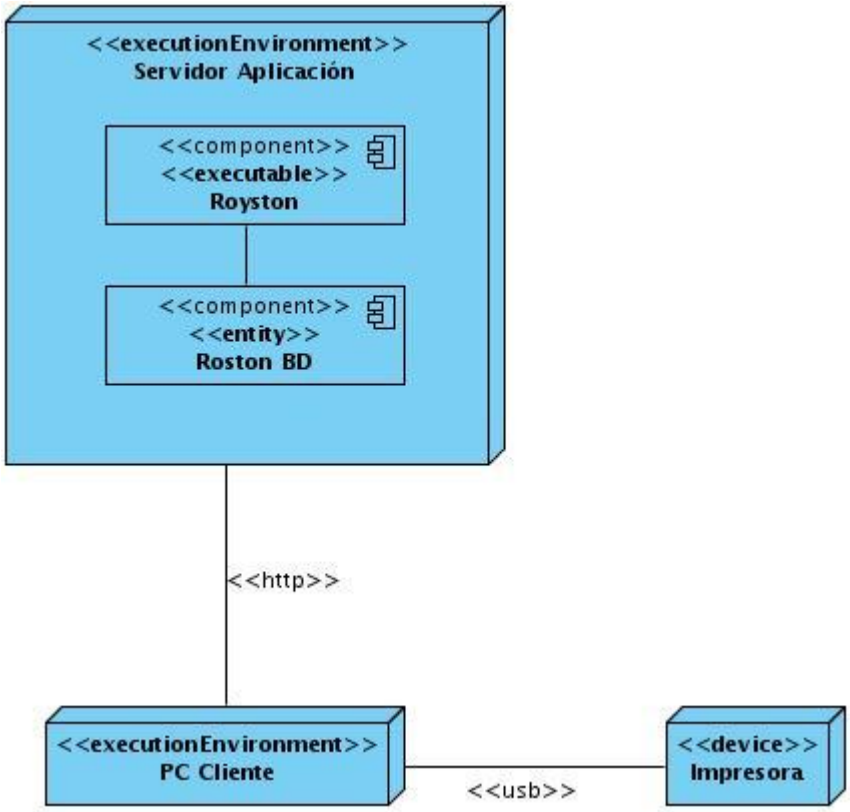


Figura 6: Modelo de despliegue

Descripción de la capacidad que el dispositivo provee al sistema

El único dispositivo que utiliza el sistema es una impresora. El sistema tiene incluida la funcionalidad de impresión de los planes de mercadotecnia, los informes de control asociados a los planes de mercadotecnia, el reporte de los objetivos, reporte de las estrategias y reporte de las acciones de cada plan, por ello se necesita que la PC[11] del usuario tenga conectada una impresora. No se necesita de una impresora en específico, solo requiere estar activa y lista para imprimir cuando el usuario esté trabajando con el *software*.

[11] *Personal Computer*; en español: Computadora Personal

Descripción de la funcionalidad y capacidad del nodo

PC Cliente: ordenador cliente que se conecta a través de un navegador web al servidor central donde reside la aplicación. No se necesita tener instalado el producto localmente.

Servidor Aplicación: servidor central, el cual tiene instalado un servidor web (Apache 2.2), uno de base de datos (PostgreSQL 8.4), el intérprete de PHP 5.x y hospeda todos los componentes necesarios para el funcionamiento del producto. La memoria RAM debe ser de al menos 512 MB.

Descripción de elementos e interfaces de comunicación

<<http>>: representa la conexión que se va a establecer entre una PC Cliente que se va a conectar con el servidor central donde reside el servidor web, el de base de datos y la aplicación; en otras palabras, significa la conexión entre el navegador de usuario y el servidor del sistema.

<<usb>>: conexión que existe entre la impresora y la PC Cliente del usuario. Se modela con usb, pero puede existir otro tipo de conexión, todo depende del tipo de impresora que esté usando el usuario y el tipo de conexión que utilice esta.

2.10. Conclusiones parciales

- En este capítulo se explicó la solución que se provee con ***Royston***, incluyendo la explicación del patrón arquitectónico Modelo-Vista-Controlador para el *framework* Symfony y los patrones de diseño utilizados. Además, se explicaron y presentaron los artefactos de la metodología ágil SXP utilizados: Lista de Reserva del Producto e Historias de Usuario; así como los modelos de diseño y de datos.

- El uso de la metodología de desarrollo de *software* SXP permitió que el equipo de trabajo se centrara en la programación y no en la documentación.

- La correcta gestión de proyecto que propone SXP posibilitó un mayor control de las tareas.

- Con la utilización del *framework* Symfony se logró minimizar el tiempo de implementación de las Historias de Usuario.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA DE LA SOLUCIÓN PROPUESTA

3.1. Introducción

La concepción de la propuesta del sistema y los artefactos generados y presentados en el capítulo anterior, ayudan al programador a entender las funcionalidades que busca el cliente y, por tanto, a ser capaz de llevarlas a código interpretable por la computadora, o lo que es lo mismo: implementar el sistema. Igual de importante para un sistema es el proceso de pruebas, que tienen como objetivo determinar si el *software* se ajusta a los requisitos o a las condiciones impuestas.(59)

Las metodologías de desarrollo denominadas robustas, controlan y documentan de manera exhaustiva los procesos de implementación y de pruebas. Esta detallada especificación de elementos es importante a la hora de integrar lo realizado en grandes proyectos; pero en el caso de los proyectos pequeños, en los que el equipo de desarrollo es también pequeño, se hace complejo crear todos estos artefactos, en especial si el proyecto utiliza un *framework* de trabajo que se ocupa de automatizar muchas operaciones de manera transparente para el propio programador.

La metodología SXP gestiona los procesos de implementación y prueba con pocos artefactos, de forma tal que la comunicación entre todos los miembros del equipo y el cliente, sea la base para la obtención del producto que realmente desea este último. En el presente capítulo se incluyen los artefactos: Plan de *Releases* y Tareas de Ingeniería, que constituyen los rectores del proceso de implementación y las Pruebas de Aceptación, base del proceso de prueba en la metodología SXP. Además, se incluyen los temas de estándares de codificación, tratamiento de errores y seguridad.

3.2. Plan de *Releases*

La metodología SXP basa su funcionamiento en iteraciones, en las que se busca “transformar un subconjunto de la Reserva del producto en un incremento en la funcionalidad del producto que sea potencialmente entregable a los usuarios. Para ello (…) se determina en qué funcionalidad del producto trabajará el equipo durante la próxima iteración”(33). Como resultado de determinar dichas funcionalidades, aparece la plantilla Plan de *Releases*, en la que “se recogen las iteraciones a realizar con

sus características, además del orden de las historias de usuario con su planificación estimada para ser implementadas".(33)

En la tabla Plan de *Releases*, se incluyen los siguientes campos: ***Release***, que contiene el identificador de la iteración que se va a desarrollar; **Descripción de la iteración**, breve descripción del objetivo de la iteración; **Orden de la HU a implementar**, contiene los identificadores de las HU a implementar en la iteración, en el mismo orden en que se deben realizar; y **Duración total**, cantidad de semanas que durará realizar la iteración, la que depende del tiempo estimado de las HU propuestas.

En el caso del sistema al que se hace referencia en este trabajo de diploma, en cada iteración se incluyeron las HU de una misma prioridad, quedando la lista como se muestra en la Tabla 3.

Tabla 3: Plan de *Releases*

Release	Descripción de la iteración	Orden de la HU a implementar	Duración total
Iteración 1	En esta iteración se desarrollan las funcionalidades priorizadas como muy altas.	HU01 HU02 HU03 HU04 HU05 HU06 HU07 HU08 HU09 HU10	10
Iteración 2	En esta iteración se desarrollan las funcionalidades priorizadas como altas.	HU11 HU12 HU13 HU14 HU15 HU16 HU21 HU22	4
Iteración 3	En esta iteración se desarrollan las funcionalidades priorizadas como medias.	HU17 HU18 HU19 HU20	4

3.3. Tareas de Ingeniería

Para alcanzar los objetivos de una iteración es necesario completar las HU que se incluyen en esta, por tanto, se precisa saber cuáles son las tareas que componen la HU, las cuales marcarán el proceso para cumplir con los objetivos de cada HU y, por consiguiente, de la iteración en particular y el sistema en general. Estas tareas son descritas en el documento Tareas de Ingeniería, artefacto generado en la metodología SXP.

En las tablas se incluyen los siguientes campos: **Número Tarea**, que contiene un número consecutivo en base a la historia de usuario correspondiente; **Número Historia de Usuario**, que contiene el identificador de la HU a la que pertenece esta tarea; **Nombre Tarea**, contiene un nombre que identifica a la tarea; **Tipo de Tarea**, contiene el tipo de tarea, que puede ser de 'desarrollo', 'corrección', 'mejora', o la especificación de otra; **Puntos Estimados**, estimación en semanas de la duración de la tarea; **Fecha Inicio**, contiene la fecha de inicio de la tarea; **Fecha Fin**, contiene la fecha de finalización de la tarea; **Programador Responsable,** nombre del programador responsable de desarrollar la tarea; y **Descripción**: que contiene la descripción de la tarea.

En las tablas de la 4 a la 7 se incluyen las tareas de ingeniería de la HU Gestionar servicios externos. Esta se encuentra clasificada en el nivel de prioridad 'muy alta', es decir, arquitectónicamente significativa para el sistema. El resto de las tareas de ingeniería se incluyen en el Anexo 5.

Tabla 4: Tarea de Ingeniería 1 de HU01

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU01
Nombre Tarea: Estudiar el *framework* Symfony.	
Tipo de Tarea : Estudio	**Puntos Estimados:** 1
Fecha Inicio: 16/12/2010	**Fecha Fin:** 30/01/2011
Programador Responsable: Roberto Carlos Aballe Ochoa Juan Carlos Lomba Fernández	
Descripción: estudiar el funcionamiento del *framework* Symfony, en especial el trabajo con los módulos, los formularios y el modelo.	

Tabla 5: Tarea de Ingeniería 2 de HU01

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU01
Nombre Tarea: Estudiar funciones del *framework* jQuery.	
Tipo de Tarea : Estudio	**Puntos Estimados:** 1
Fecha Inicio: 30/01/2011	**Fecha Fin:** 10/02/2011
Programador Responsable: Roberto Carlos Aballe Ochoa Juan Carlos Lomba Fernández	
Descripción: estudiar el funcionamiento del *framework* jQuery y su integración con Symfony.	

Tabla 6: Tarea de Ingeniería 3 de HU01

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** HU01
Nombre Tarea: Desarrollar CRUD de servicios externos.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 0.5
Fecha Inicio: 10/02/2011	**Fecha Fin:** 12/02/2011
Programador Responsable: Roberto Carlos Aballe Ochoa	
Descripción: implementar las funcionalidades de listar, mostrar, adicionar, modificar y eliminar los datos de un servicio externo.	

Tabla 7: Tarea de Ingeniería 4 de HU01

Tarea de Ingeniería	
Número Tarea: 4	**Número Historia de Usuario:** HU01
Nombre Tarea: Buscar servicio externo.	
Tipo de Tarea : Mejora	**Puntos Estimados:** 0.5
Fecha Inicio: 12/02/2011	**Fecha Fin:** 13/02/2011
Programador Responsable: Roberto Carlos Aballe Ochoa	
Descripción: crear filtros de búsqueda para los servicios externos, teniendo como base el campo que se refiere al nombre del servicio externo.	

3.4. Pruebas de Aceptación

Al realizar la planificación de la iteración es necesario definir también el sistema de pruebas que se aplicarán para verificar que el sistema cumpla con las funcionalidades que precisa el cliente. Con este objetivo se construye y entrega el artefacto de la metodología SXP: Pruebas de Aceptación, en las que “el desarrollador escribe las pruebas realizadas según la historia de usuario seleccionada para realizar la comprobación y validar las funcionalidades del sistema, y de esta forma saber si está apto para ser liberado”.(33)

En las tablas de las Pruebas de Aceptación, se incluyen los siguientes campos: **Código Caso de Prueba**, que contiene el identificador de caso de prueba (en el caso de las presentes, se utiliza el identificador de la HU, al que se le adiciona ‘-P’ y un número consecutivo); **Nombre Historia de Usuario**, que contiene el nombre de la HU correspondiente a este caso de prueba; **Nombre de la persona que realiza la prueba**, contiene el nombre del responsable que realiza la prueba; **Descripción de la Prueba**, que contiene una breve descripción de la prueba realizada; **Condiciones de Ejecución**, se incluyen las condiciones necesarias para que se pueda realizar la prueba; **Entrada/Pasos de ejecución**, contiene una serie de pasos enumerados para lograr realizar la prueba de esta HU; **Resultado Esperado**, contiene la descripción de lo que se espera luego de realizar la prueba (cumplimiento de las restricciones del producto); y **Evaluación de la Prueba**, muestra si la prueba fue satisfactoria o insatisfactoria.

A continuación se incluyen las tablas de la 8 a la 11, que contienen los Casos de Pruebas de Aceptación de la HU Gestionar servicios externos, una de las clasificadas en el nivel de prioridad ‘muy altas’, o sea, arquitectónicamente significativa para el sistema. El resto se detallan en el Anexo 6.

Tabla 8: Caso de Prueba de Aceptación 1 de HU Gestionar servicios externos

Caso de Prueba de Aceptación	
Código Caso de Prueba: HU01-P1	**Nombre Historia de Usuario:** Gestionar servicios externos
Nombre de la persona que realiza la prueba: Roberto Carlos Aballe O.	
Descripción de la Prueba: Se insertan los datos de un servicio externo, inicialmente se insertarán incorrectamente para verificar las validaciones del sistema, luego de forma correcta para comprobar que los datos sean almacenados.	
Condiciones de Ejecución: El usuario debe tener los permisos suficientes para realizar esta operación.	
Entrada / Pasos de ejecución: 1. Insertar los datos en blanco. 2. Insertar el nombre en blanco y los otros datos no. 3. Insertar el campo precio en blanco y los otros no. 4. Insertar el campo moneda en blanco y los otros no. 5. Insertar un nombre que ya esté almacenado en la base de datos. 6. Insertar en el campo del nombre el caracter espacio. 7. Insertar en el campo nombre, números. 8. Insertar un nombre que comience con letra minúscula. 9. Insertar un nombre con algún o algunos números incluidos. 10. Insertar en el campo precio datos no numéricos. 11. Insertar los datos correctos. 12. Verificar que el nuevo producto aparece en el listado.	
Resultado Esperado: El sistema debe alertar al usuario cuando se inserten datos en blanco en los campos obligatorios (nombre, precio y moneda); además debe alertar cuando ya exista un elemento con el mismo nombre, almacenado en la base de datos. Cuando se inserten los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrarlos en el listado.	
Evaluación de la Prueba: Satisfactoria	

Tabla 9: Caso de Prueba de Aceptación 2 de HU Gestionar servicios externos

Caso de Prueba de Aceptación	
Código Caso de Prueba: HU01-P2	**Nombre Historia de Usuario:** Gestionar servicios externos
Nombre de la persona que realiza la prueba: Roberto Carlos Aballe O.	
Descripción de la Prueba: Se muestra la lista de servicios externos que están almacenados en la base de datos. Se puede visualizar la información específica de un servicio externo.	
Condiciones de Ejecución: Deben existir varios elementos almacenados en la base de datos. Debe estar abierto algún cliente de base de datos de PostgreSQL en el que esté cargada la base de datos del sistema.	
Entrada / Pasos de ejecución: 1. Contar el número de servicios externos almacenados en la base de datos y ver sus características. 2. Verificar que se muestre un listado en el que aparezca la cantidad de servicios externos anteriormente contados y que sus datos coincidan con los antes vistos. 3. Verificar que se visualizan los datos de un servicio externo específico.	
Resultado Esperado: El sistema debe mostrar todos los servicios externos almacenados en la base de datos con su información, y debe permitir visualizar toda la información relacionada con uno en específico.	
Evaluación de la Prueba: Satisfactoria	

Tabla 10: Caso de Prueba de Aceptación 3 de HU Gestionar servicios externos

Caso de Prueba de Aceptación	
Código Caso de Prueba: HU01-P3	**Nombre Historia de Usuario:** Gestionar servicios externos
Nombre de la persona que realiza la prueba: Roberto Carlos Aballe O.	
Descripción de la Prueba: Se modifican los datos de un servicio externo, inicialmente se modificarán incorrectamente para verificar las validaciones del sistema, luego de forma correcta para comprobar que los datos sean almacenados y cargados.	
Condiciones de Ejecución: El usuario debe tener los permisos suficientes para realizar esta operación.	
Entrada / Pasos de ejecución: 1. Seleccionar la opción Modificar en un servicio externo específico. 2. Modificar los datos en los campos requeridos (nombre, precio y moneda) e insertarlos en blanco. 3. Modificar el nombre, poniéndole uno que ya esté almacenado en la base de datos. 4. Modificar el nombre insertando espacio. 5. Modificar el nombre insertando números. 6. Modificar el nombre insertando un nombre que comience con letra minúscula. 7. Modificar el nombre insertando un nombre con algún o algunos números incluidos. 8. Modificar el precio insertando datos no numéricos. 9. Modificar de forma correcta los datos. 10. Verificar que el elemento aparece en el listado con los nuevos datos.	
Resultado Esperado: El sistema debe alertar al usuario cuando se inserten datos en blanco en los campos obligatorios (nombre, precio y moneda); además debe alertar cuando ya exista un elemento con el mismo nombre almacenado en la base de datos. Cuando se modifiquen los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrarlos en el listado.	
Evaluación de la Prueba: Satisfactoria	

Tabla 11: Caso de Prueba de Aceptación 4 de HU Gestionar servicios externos

Caso de Prueba de Aceptación	
Código Caso de Prueba: HU01-P4	**Nombre Historia de Usuario:** Gestionar servicios externos
Nombre de la persona que realiza la prueba: Roberto Carlos Aballe O.	
Descripción de la Prueba: Se elimina un servicio externo.	
Condiciones de Ejecución: El usuario debe tener los permisos suficientes para realizar esta operación.	
Entrada / Pasos de ejecución: 1. Seleccionar la opción Eliminar en un servicio externo específico. 2. Verificar que el servicio externo eliminado no aparece en el listado.	
Resultado Esperado: El sistema debe mostrar una ventana de aviso al usuario preguntando si realmente desea eliminar. Cuando se acepte, debe eliminar el servicio externo. Después de eliminado, el elemento no debe aparecer en la lista.	
Evaluación de la Prueba: Satisfactoria	

3.4.1. Resultado de las pruebas

Se realizaron dos iteraciones de las pruebas de aceptación con el cliente para conocer en qué grado satisfacía el producto a sus necesidades. Durante la primera iteración se encontraron tres no conformidades, las cuales fueron resueltas antes de la segunda iteración, después de la cual la aplicación quedó aprobada por parte del cliente, evaluando el cien por ciento de los casos de prueba de forma satisfactoria.

A petición del cliente y los desarrolladores y, teniendo en cuenta la función estratégica que cumplirá, se propuso que el *software* fuera validado por el Grupo de Calidad del centro FORTES realizándole pruebas de caja negra. Durante la primera iteración del proceso se encontraron siete no conformidades, tres de prioridad ‘Alta’ y tres de prioridad ‘Media’. Ya en la segunda iteración no se encontraron no conformidades, por lo que la aplicación fue validada por el Grupo de Calidad. El Acta de Liberación se muestra en el Anexo 7.

3.5. Estándar de código

La calidad del código en la programación se mide por varios aspectos: la estabilidad, la tolerancia a los fallos y la reusabilidad son algunos de ellos. Pero otros aspectos importantes a la hora de medir la calidad

del código programado son la legibilidad y la posibilidad de adicionarle nuevas sentencias. Para poder conseguir un código con estas dos últimas características es necesario establecer un conjunto de reglas a la hora de escribir el código del programa, es decir: utilizar un estándar de código.

El estándar de código, también conocido como estándar de programación o estilo de código, "tiene importantes consecuencias en la programación (…) por ejemplo, la investigación ha demostrado que ciertas prácticas estilísticas pueden servir para reducir el número de errores que se presentan durante el desarrollo del programa. Al mismo tiempo, el programa mismo resulta más fácil de leer y de comprender por otros programadores, quienes pueden en algún momento ser llamados para realizar modificaciones del mismo. El mantenimiento del programa, es decir, el "ajuste" de los programas que existen para reunir requerimientos siempre cambiantes, consume gran parte del tiempo del trabajo de los programadores profesionales; es muy común, de hecho, que se emplee más tiempo en el mantenimiento de un programa que en su desarrollo original."(60)

Para implementar el presente sistema, se utilizó el siguiente estándar de código:

1. Todas las etiquetas PHP deben estar completas (<?php ?>), no reducidas (<? ?>).

2. La indentación se basa en el estilo *"hanging paragraph"* (párrafo colgante), en el que las líneas de código anidadas dentro de otra se indentan con espacios. Ejemplo:

```
public function nombreFuncion()
{
$variable = 0;
for ($i = 0; $i < $variable; $i++)
                    if ($i == funcionX())
                                return true;
}
```

3. En los arreglos en los que se insertan varios valores se utiliza la alineación vertical. Ejemplo:

```
arreglo(
        producto => 'nombre',
        precio => 33.15,
        descripcion => 'producto'
);
```

4. Los bloques de código deben estar siempre confinados por llaves, excepto en los casos en que tengan una sola línea. Las llaves siempre estarán al mismo nivel de la sentencia de código de la que proceden. Ejemplo:

```
public function nombreFuncion()
{
        if (funcionX() == 0)
                return 0;
        else
        {
                $valor = 1;
                return $valor;
        }
}
```

5. Las sentencias de código demasiado largas deben ser divididas en tantas líneas como sea necesario para poder ser vista sin necesidad de utilizar desplazamiento horizontal. Esto sucede por lo general en las sentencias donde se definen datos para arreglos y las divisiones debe ser siempre después de las comas.
6. Las cadenas de caracteres deben definirse utilizando comillas simples siempre que sea posible, para obtener un mejor rendimiento.
7. Los nombres de las variables y funciones comenzarán siempre con letra minúscula. Ejemplo: $variable, nombreMetodo().
8. Cuando el nombre de la variable o la función es compuesto (tiene más de una palabra), entonces se utilizará el estilo “encamellado” (*camel case*). Ejemplo: $variableCompuesta, $variable1, nombreMetodo().
9. Las funciones se especificarán de forma tal que el tipo de valor que devuelve sea entendible:
    a) Las funciones para obtener algún dato de un objeto específico se comenzarán utilizando el verbo inglés *get*, para lograr uniformidad con los métodos del mismo tipo que incorpora el *framework* Symfony. Ejemplo: getNombreProducto().

10. Las declaraciones de clases abren llaves en la misma línea de la declaración, y cierran llaves con el mismo nivel de margen de la declaración, una línea después de la última sentencia. Ejemplo:

```
class saveEstrategia {
}
```

11. Cuando sea necesario utilizar comillas, estas serán simples. Ejemplo:

```
<?php echo 'Análisis del cliente'; ?>
```

12. Para la concatenación de cadenas en PHP, siempre se usará un espacio entre el punto y las partes concatenadas para mejorar la legibilidad. Ejemplo:

```
<?php echo 'Total de ingresos previstos: '?><?php echo $suma_ingresos_usd . ' dólares'?>
```

13. Para la concatenación de cadenas en JavaScript no se usará espacio entre el signo '+' y las partes concatenadas. Ejemplo:

```
'[name*='+field+']'
```

14. La terminología usada para escribir código CSS es:

```
selector
{
    propiedad: valor;
}
```

15. En el código CSS los selectores deben estar seguidos con una llave abierta en la próxima línea. Luego de las propiedades debe terminar con una llave cerrada con la misma sangría de la llave abierta. Una línea en blanco debe existir entre cada bloque de selectores. Ejemplo:

```
.button_login
{
   padding: 4px 5px 4px 5px !important;
}

.button_login .box1
{

}
```

16. En el código CSS, cada propiedad debe estar en su propia línea, identada con dos espacios, debe tener un solo espacio entre el nombre de la propiedad y el valor y terminar con un punto y coma. Ejemplo:

```
#contentObj
{
    width: 100%;
    height: 200px;
    overflow: auto !important;
}
```

El estándar de código anteriormente descrito es el utilizado al adicionar nuevas líneas de código a los ficheros que genera el *framework* Symfony; el resto de las clases, funciones u otros elementos generados automáticamente por el *framework* pueden diferir de dicho estándar.

3.6. Seguridad

La computadora ha revolucionado la sociedad, optimizando y acelerando procesos en todos los campos de la ciencia y la técnica. Su uso ha extendido a todas las instituciones o centros que requieren gestionar correctamente información. La difusión del uso de la computación también trae consigo el problema de proteger el elemento más importante: los datos, que son considerados actualmente activos tan importantes como el dinero o el capital humano. Por este motivo es necesario darle al tema de la seguridad informática la importancia que merece, pues “no es solo un tema de moda, es una cultura que debemos conocer para no quedarnos atrás y no ser blanco fácil de quienes quieran hacerle daño a la empresa por medio de la red (…)”.(61)

Para garantizar el funcionamiento correcto y seguro de la aplicación a la que hace referencia este trabajo se establecen como políticas de seguridad: la autenticación de las personas autorizadas y previamente registradas en el sistema, así como la administración de la base de datos por parte del administrador del sistema.

Con el objetivo de proteger la información manejada por el sistema se definirán varios tipos de usuarios, estableciéndoles permisos indispensables de acceso a la información de acuerdo con su rol. Por su parte, el acceso a la base de datos será solamente a través de usuarios con privilegios mínimos, los suficientes para realizar solamente las operaciones básicas de acceso a datos.

En caso de una contingencia, se debe efectuar una recuperación de la información que el sistema tenía almacenada. Para facilitar esta recuperación se deben realizar copias de seguridad periódicas de la información contenida en la base de datos y en las carpetas donde se almacenan los documentos, chequeándose que estas salvas son guardadas en perfecto estado.

En cuanto a la programación de la aplicación, se siguieron muchos de los principios de la programación segura de aplicaciones, específicamente de aplicaciones web. Los mensajes de error que se le muestran al usuario, se elaboran respetando el principio de ofrecer la mínima información. El ORM empleado (Doctrine) permite declarar parámetros en todas las consultas, evitando de esta forma ataques de inyección SQL.

Cuando se genera la aplicación utilizando el *framework* Symfony, automáticamente se activan medidas de seguridad para la protección frente a dos de las vulnerabilidades más extendidas en la web: los ataques de tipo XSS y los de tipo CSRF. Para evitar los ataques XSS, el *framework* activa el mecanismo de escape, que consiste en reemplazar en los datos mostrados en la vista, los símbolos '<' y '>' por sus equivalentes '<' y '>', evitando así la interpretación por parte del navegador de cualquier *script* malicioso. Por su parte, para evitar los ataques CSRF, Symfony genera aleatoriamente una palabra secreta, que es enviada al servidor en cada solicitud proveniente de los formularios (en un campo oculto llamado _csrf_token) o ante una solicitud de eliminación (como un parámetro con el mismo nombre), de forma tal, que se compruebe la autenticidad de la petición.

3.7. Tratamiento de errores

La informática está muy lejos de ser perfecta, y lo demuestran un buen número de mensajes de error que han perseguido y acompañado a los usuarios mientras trabajan con distintos productos de *hardware* y *software*. Para solucionar estos problemas aparece el término tratamiento de errores, elemento que permite identificar, localizar, analizar y eliminar los errores en la implementación de un programa de computadora. Las aplicaciones deben poseer un buen mecanismo de tratamiento de errores, enfocándose no sólo en las entradas suministradas por el usuario, sino también en cualquier error que pueda ser generado por el comportamiento incorrecto de componentes internos, por ejemplo, errores en tiempo de ejecución o en consultas a la base de datos.

En ***Royston***, los datos que introduce el usuario son validados en los formularios de Symfony, mostrando mensajes de error que indican al usuario cuáles datos están incorrectos e impidiendo el envío de la solicitud al servidor mientras los formularios contengan valores incorrectos.

3.8. Conclusiones parciales

- En este capítulo se incluyeron y explicaron los artefactos generados por la metodología SXP para la implementación, las pruebas y su aplicación en el Sistema de Información de Mercadotecnia para el centro FORTES.

- En el artefacto Plan de *Releases* se definieron 4 iteraciones. En cada iteración se incluyeron las HU de una misma prioridad, quedando reservada la primera para las HU arquitectónicamente significativas. El artefacto Tareas de Ingeniería marcó el proceso para cumplir con los objetivos de cada HU, de cada iteración y del sistema en general.

- El proceso de pruebas de aceptación arrojó que un 100% de las pruebas fueran evaluadas de “Satisfactoria” por parte del cliente.

- Se describió e estándar de código utilizado, el cual provee legibilidad y la posibilidad de adicionarle nuevas sentencias por parte de otros programadores, los que en algún momento pueden ser llamados a realizar modificaciones.

- Se establecieron las políticas de seguridad a seguir por parte de los programadores y se explicaron las bondades que en cuanto a seguridad brinda automáticamente Symfony al ser usado y en particular la utilización del ORM Doctrine.

- Se describió la manera de tratar los errores durante la implementación del sistema en cuanto a la validación de formularios.

CONCLUSIONES GENERALES

La investigación desarrollada y los resultados obtenidos permiten a los autores plantear las siguientes conclusiones:

- El análisis de las características de un Sistema de Gestión de Información y las de un Sistema de Información de Marketing, posibilitó el diseño de un Sistema de Gestión de Información para Marketing que se adecua a las necesidades específicas del centro FORTES.
- La búsqueda y análisis de sistemas semejantes demostró, que las herramientas a nivel internacional poseen características no requeridas por el cliente y son propietarias, mientras que a nivel nacional no se encontraron soluciones que automaticen el proceso de gestión de información para *marketing*, evidenciándose la necesidad de crear un nuevo producto de este tipo para el centro FORTES.
- Mediante la aplicación de la metodología ágil SXP se logró efectividad y rapidez en el desarrollo de *software* y la gestión de proyecto.
- Se obtuvo una aplicación web funcional (probada y validada durante su proceso de desarrollo utilizando pruebas de aceptación), desarrollada con tecnologías libres y provista de elementos de seguridad y un entorno amigable, gracias a la utilización del *framework* Symfony y la librería JavaScript jQuery.

Por lo antes expuesto, se considera que la base teórica, la selección de las tecnologías para construir el Sistema de Gestión de Información para Mercadotecnia y su implementación; así como la vinculación en una sola aplicación de la Gestión de Información de Mercadotecnia con la planeación mercadotécnica, son los principales aportes de este trabajo de diploma.

Este tipo de sistema es altamente valioso debido al tipo de información que gestiona. En el desarrollo de la aplicación se utilizaron herramientas libres y herramientas gratuitas, lo cual minimiza los costos.

El contar con una herramienta propia para gestionar los datos que genera la actividad, permite hacer más eficiente el control mercadotécnico, pues garantiza la confiabilidad de la información resultante, reduce el tiempo de las búsquedas, acumula datos que no deben desaparecer y organiza y sistematiza los que mantienen actualidad, todo lo cual es visible a través de informes. Por otra parte, contar con esta herramienta debe reducir el tiempo de elaboración de los planes de mercadotecnia, el diagnóstico de la

capacidad exportable, y otras actividades operativas que afectan la planeación estratégica en el área. Todo ello deberá apoyar la toma de decisiones en cuanto a la esfera de exportación. El sistema desarrollado, se espera que contribuya a agilizar y mejorar la gestión de la mercadotecnia en el centro FORTES, y que influya en el posicionamiento más eficaz de sus productos y/o servicios.

RECOMENDACIONES

- Incorporar en la próxima versión funcionalidades que, haciendo uso de servicios web, permitan obtener y brindar información directamente desde y hacia el SIM que está siendo desarrollado para ALBET S.A.
- Incorporar funcionalidades que permitan agregarle valor a los productos de información que genera la aplicación, como puede ser la visualización de información a través de la minería de datos.

Referencias bibliográficas

1. **FORTES, Universidad de las Ciencias Informáticas.** Documento Visión del centro FORTES. Ciudad de La Habana : s.n., 2010.
2. [En línea] [Citado el: 8 de Enero de 2011.] http://www.itu.int/net/itunews/issues/2010/10/04-es.aspx.
3. **Peña Azahares, Maniuryis.** Diseño de la Gestión de Información para las decisiones de marketing en el Centro de Tecnologías para la Formación, FORTES. [Protocolo de tesis en opción al grado de Máster en Gestión de Información]. La Habana, Cuba : s.n., 2010. inédito.
4. **Kotler, Philip.** *Dirección de Mercadotecnia. Análisis, Planeación, Implementación y Control.* Octava. Lima : Pearson Educación, 2001.
5. **Gómez Vieites, Álvaro y Suárez Rey, Carlos.** *SISTEMAS DE INFORMACIÓN. HERRAMIENTAS PRÁCTICAS PARA LA GESTIÓN EMPRESARIAL.* Segunda edición. s.l. : Microinformática. ISBN: 8478976853.
6. *Información: una nueva propuesta conceptual.* **Angulo Marcial, Noel.** 4, Diciembre de 1996, Biblioteca Virtual de las Ciecias en Cuba, Vol. 27. 959-234-032-3.
7. **Monforte, Manfredo.** *Sistemas de información para la dirección.* Madrid : Pirámide, 1995.
8. **Bueno Campos, Eduardo.** *Dirección Estratégica de la empresa. Metodología, técnicas y casos.* Madrid : Pirámide, 1996.
9. *Lenguaje e información.* **Vizcaya Alonso, Dolores.** 4, Agosto de 2001, Revista de Ciência da Informação, Vol. 2.
10. **Langefors, Börje.** *Teoría de los sistemas de información.* Buenos Aires : El Ateneo, 1976.
11. *Gestión - Información - Conocimiento.* **Murray, Pablo.** 14, Lima : s.n., Noviembre de 2002, BIBLIOS.
12. **Bartle, Phil.** Información para la gestión y gestión de la información. *CEC Community Empowerment Collective.* [En línea] [Citado el: 10 de Enero de 2011.] http://www.scn.org/mpfc/modules/mon-miss.htm.
13. **Méndez Rodríguez, Eva María.** *Metadatos y tesauros: aplicación de XML/RDF a los sistemas de organización del conocimiento en Intranets.* Madrid : s.n., 2000.
14. **Universidad Rey Juan Carlos.** Sistemas de Información y Bases de datos. *Conferencia.*
15. **RENa.** Red Escolar Nacional. [En línea] 2008. [Citado el: 10 de Enero de 2011.] http://www.rena.edu.ve/.
16. **López G., Rosiry M.** Automatización del Sistema de Gestión de la Base de Datos de la Numeración Nacional de CONATEL. *Trabajo presentado como requisito parcial para optar por el título de Ingeniero Electricista.* Mérida, Mérida, Venezuela : s.n., Diciembre de 2004.
17. **Hernández Arias, Aymara.** UCLA. [En línea] [Citado el: 11 de Enero de 2011.] http://www.ucla.edu.ve/dac/investigaci%F3n/compendium9/Sistemas.htm.
18. **Muñiz González, Rafael.** *Marketing en el Siglo XXI.* Tercera edición. Disponible en: http://www.marketing-xxi.com.
19. **Garmendia, Fermín y Romeiro Serna, John.** Sistemas de Información de Marketing-SIM: más que simples cajas tecnológicas. s.l. : ESIC MARKET, Diciembre de 2007. Disponible en: http://www.esic.es/documentos/revistas/esicmk/070905_114835_E.pdf.
20. **Serna, Sergio.** Sistemas de Información Aplicados a la Mercadotecnia. Disponible en: http://www.mercadeo.com/archivos/Sist_Info_Mercadeo.pdf.
21. **Ministros, Comité Elecutivo del Consejo de.** Decreto No. 281. Reglamento para la implantación y consolidación del sistema de dirección y gestión empresarial estatal. 2007.

22. **Inc., Business Resource Software.** Business Resource Software Inc. [En línea] 1994. [Citado el: 12 de Enero de 2011.] http://www.brs-inc.com/marketing_plan.asp.
23. **Osés Sosa, Yoelvis.** Sistema de Información de Mercadotecnia para el Centro de Informática Médica (CESIM). Ciudad de La Habana, Cuba : s.n. Disponible en: http://www.fec.uh.cu/CUGIO/1%20acciones/Proyectos-Protocolos/Yoelvis%20Oses/MIC%20Yoelvis%20Os%C3%A9s%20Sosa%20UCI.pdf.
24. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* s.l. : Addison Wesley, 2000.
25. **Callejas Cuervo, Mauro y Baquero Moreno, Oscar Yovany.** Herramientas libres para modelar software. 2005. Disponible en: http://ervisual.hostoi.com/pdf/er_tecnico.pdf. 0121-1129.
26. **González Blanco, Rubén y Pérez Tobalina, Sergio.** LESE-2. Introducción a Rational Rose. Barcelona. Universidad Politécnica de Cataluña, España : s.n., 2003.
27. UML tool, business, process modeler and database designer for software development team. [En línea] [Citado el: 9 de Enero de 2011.] http://www.visual-paradigm.com.
28. **Brito Acuña, Kerenny.** eumed.net. [En línea] [Citado el: 11 de Febrero de 2011.] SELECCIÓN DE METODOLOGÍAS DE DESARROLLO PARA APLICACIONES WEB EN LA FACULTAD DE INFORMÁTICA DE LA UNIVERSIDAD DE CIENFUEGOS. http://www.eumed.net/libros/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm.
29. **Molpeceres, Alberto.** Procesos de desarrollo: RUP, XP y FDD. 15 de Diciembre de 2002.
30. **H. Canós, José, Letelier, Patricio y Penadés, Mª Carmen.** Metodologías Ágiles en el Desarrollo de Software. Valencia : s.n., 2005.
31. Proyectos Ágiles. [En línea] [Citado el: 10 de Febrero de 2011.] http://www.proyectosagiles.org/que-es-scrum.
32. **Fernández Escribano, Gerardo.** Introducción a Extreme Programming. 9 de Diciembre de 2002. Disponible en: http://www.info-ab.uclm.es/asignaturas/42551/trabajosAnteriores/Presentacion-XP.pdf.
33. **Peñalver, G., Meneses, A. y García, S.** SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE. Ciudad de La Habana, Cuba : s.n., 2010.
34. **Alvarez, Sara.** DesarrolloWeb.com. [En línea] [Citado el: 11 de Enero de 2011.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.
35. **PostgreSQL Global Development Group.** PostgreSQL: The world´s most advanced open source database. [En línea] [Citado el: 6 de Enero de 2011.] http://www.postgresql.org.
36. APACHE HTTP SERVER PROJECT. *Information on the Apache HTTP Server Project.* [En línea] [Citado el: 10 de Enero de 2011.] http://web.archive.org/web/19961028123412/http://www.apache.org/info.html.
37. **Netcraft.** Netcraft. [En línea] [Citado el: 12 de Enero de 2011.] http://news.netcraft.com/archives/category/web-server-survey.
38. IIS. [En línea] 2011. [Citado el: 10 de Febrero de 2011.] http://www.iis.net/.
39. **Gallego Vázquez, José Antonio.** *Desarrollo Web con PHP y MySQL.* Primera edición. Madrid : Anaya, 2003. 84-415-1525-5.
40. The Perl Programming Language. [En línea] 2011. [Citado el: 9 de Febrero de 2011.] http://www.perl.org/.
41. **López, Ángel.** *Java, la programación del futuro.* Buenos Aires : s.n., 2005.
42. **W3C.** *World Wide Web Consortium.* [En línea] 7 de Febrero de 2008. [Citado el: 24 de Enero de 2011.] Guía Breve de XHTML. http://www.w3c.es/divulgacion/guiasbreves/XHTML.

43. **Eguíluz Pérez, Javier.** *Introducción a XHTML.* 2008. Disponible en: http://www.librosweb.es/xhtml.
44. —. *Introducción a CSS.* 2009. Disponible en: http://www.librosweb.es/css.
45. **W3C.** *World Wide Web Consortium.* [En línea] 24 de Marzo de 2005. [Citado el: 29 de Enero de 2011.] Disponible en: http://www.abcdatos.com/webmasters/tutorial/z1056.html. http://www.w3c.es/.
46. **CodeBOX.** [En línea] 2008. [Citado el: 30 de Enero de 2011.] CodeBOX: Glosario. http://www.codebox.es.
47. Symfony. [En línea] Sensio Labs. [Citado el: 9 de Enero de 2011.] http://www.symfony-project.org/.
48. Zend Framework. [En línea] 2011. [Citado el: 9 de Febrero de 2011.] http://framework.zend.com.
49. jQuery. [En línea] [Citado el: 10 de Enero de 2011.] http://www.desarrolloweb.com/articulos/introduccion-jquery.html.
50. Prototype JavaScript framework. [En línea] 2007. [Citado el: 8 de Febrero de 2011.] http://www.prototypejs.org.
51. **Sencha.** Ext JS en español. Comunidad de desarrollo. [En línea] [Citado el: 8 de Febrero de 2011.] http://extjs.es.
52. NetBeans. [En línea] 2011. [Citado el: 13 de Enero de 2011.] Portal del IDE Java de código abierto. http://netbeans.org/community/releases/69/.
53. Zend Studio. [En línea] 2010. [Citado el: 8 de Febrero de 2011.] http://www.zend.com/products/studio.
54. **Bass, Len, Clements, Paul y Kazman, Rick.** *Software Architecture in Practice.* Segunda edición. s.l. : Pearson Education, 2003. Disponible en: http://books.google.com.cu/books?id=mdiIu8Kk1WMC&dq=software+architecture+in+practice&printsec=frontcover&source=bn&hl=es&ei=lrG6S7uqJ8P98AaluKGkCA&sa=X&oi=book_result&ct=result&resnum=4&ved=0CCAQ6AEwAw#v=onepage&q=&f=false. 0-321-15495-9.
55. **Potencier, Fabien.** *Practical Symfony.* [ed.] Sensio Labs. 2009.
56. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* [trad.] Luz María Hernández Rodríguez. Primera edición. s.l. : Prentice Hall, 1999. Traducido de: Applying UML and Pattern. An Introduction to Object-Oriented Analysis and Design. 970-17-0261-1.
57. **Pavón Mestras, Juan.** Patrones de diseño orientado a objetos. Madrid: Universidad Complutense de Madrid, España : s.n., 2004. Disponible en: http://www.fdi.ucm.es/profesor/jpavon/poo/2.14PDOO.pdf.
58. **Castilla Plaza, Carlos.** Implicaciones de las tecnologías de la información en la gestión del sistema empresa. *Tesis Doctoral.* Madrid, España : s.n., 2003. Tesis de la Universidad Complutense de Madrid, Facultad de Ciencias Económicas y Empresariales, Departamento de Economía Financiera y Contabiblidad I. Tutor: Dr. Santodomingo Garachana, A.. 3641.
59. **Polo Usaola, Dr. Macario.** Mantenimiento Avanzado de Sistemas de Información. Pruebas de Software.
60. **González Cornejo, José Enrique.** ACERCA DEL ESTILO EN PROGRAMACIÓN. 18 de Abril de 2009. Disponible en: http://www.docirs.cl/acerca_del_estilo_programacion.htm.
61. *Seguridad informática.* **Velázquez, A.** 3, s.l. : Entrepreneur, 2005, Vol. XIII, págs. 108-109. Disponible en: http://search.ebscohost.com/login.aspx?direct=true&db=zbh&AN=22948879&site=ehost-live. 16655087.