

Universidad de las Ciencias Informáticas

Facultad 4

Título:

Diseño e Implementación del Sistema SGISC

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

Autores: Roy Alberto Castro Aguilera

Arletys Betancourt Alemán

Tutores:

MSc. Mailin Carballosa Infante

Ing. Yordankis Matos López

Declaración de autoría:

Declaramos que somos los únicos autores del presente trabajo y le damos autorización a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma. Para que haya constancia firmamos a los ____ días del mes de _____ del año _____.

Arletys Betancourt Alemán

Firma del autor

Roy Alberto Castro Aguilera

Firma del autor

Ing. Yordankis Matos López

Firma del tutor

MSc. Mailin Carballosa Infante

Firma del tutor

Dedicatoria:

A mis padres, mi hermana y familia en general

A mis tutores, que sin su tiempo nada habría sido posible.

Arletys

A mis padres y hermanas.

A la familia en general, a los amigos de siempre y de ahora.

A nuestros tutores, pues no hay palabras suficientes para expresar nuestra gratitud por su tiempo.

Roy Castro

Agradecimientos:

Arletys

Le agradezco a mis padres, quienes han sido mi ejemplo y guía durante toda la vida. Gracias por brindarme su amor y su apoyo incondicional. A mi adorada hermanita a quien he tratado de inspirar para que sea una mejor persona cada día. A mis abuelos, tíos, primos y tías abuelas, quienes adoro con todo mi corazón y me han alentado siempre para que me supere. A mis amigos y a mis profesores que me han aconsejado y extendido su mano cuando lo he necesitado. A la familia de mi novio que es la mía también y me han acogido como miembro de la suya. A esas personas que por razones adversas hoy no se encuentran entre nosotros pero de las que aprendí mucho y de las que estaré muy orgullosa siempre, así como lo estarían ellos de mí. Les agradezco inmensamente a mis tutores que han invertido su tiempo en la realización de esta tesis, brindando su ayuda incondicional para que el resultado logrado fuese el mejor. Un agradecimiento especial a quien ha sido mi guía constante, mi amigo, mi apoyo en los buenos y malos momentos, y sobre todo quien me ha dado su corazón, su cariño y todo su amor, mi novio, mi compañero de tesis, el amor de mi vida. Sin él, el camino se me habría hecho largo y difícil. Gracias a todos, porque gracias a ustedes hoy estoy aquí, firme y segura de mí, cada día como una mejor persona, más preparada, responsable, con muchas metas trazadas y muchos sueños por cumplir. A cada uno de ustedes que me han hecho crecer, dedico esta tesis.

Roy Castro

A esos que unas veces obviamos, otras reprochamos, pero siempre damos por sentado que estarán ahí más allá del momento triste, la trastada o la equivocación.

A los que no están en nuestro camino porque ya recorrieron el suyo.

A mis padres.

A mis abuelos, que sin título universitario, son decanos de la vida.

Resumen:

La imprenta Sagrado Corazón presenta la necesidad de un mayor control de sus procesos productivos, entre ellos el control de insumos, control básico de las finanzas, control de pedidos y mejores formas para analizar la información generada. A partir del análisis realizado para solventar esta necesidad se decidió la implementación de un sistema capaz de gestionar estos procesos. Como lenguaje de programación del lado del servidor se utilizó PHP apoyándose en la utilización de Symfony y JavaScript utilizando la librería JQuery del lado del cliente. Se utilizaron las herramientas NetBeans 7.4 y Visual Paradigm para el diseño e implementación del sistema. El proceso de desarrollo estuvo guiado por la metodología de desarrollo XP. Se describieron las pruebas de aceptación realizadas para validar la propuesta del sistema.

Palabras Clave: análisis, desarrollo, control, gestión, sistema

ÍNDICE

Introducción	1
1 CAPÍTULO 1: Fundamentación Teórica.....	1
1.1 Introducción.....	1
1.2 Conceptos fundamentales relacionados con el dominio del problema	1
1.2.1 Lenguajes de programación y entornos web.....	1
1.3 Sistemas de Gestión.....	2
1.3.1 Cumulus Software	3
1.3.2 EcoSoftCS.net	4
1.3.3 SAGE ERP X3.....	4
1.3.4 OpenERP	5
1.3.5 Stock Mistral.....	5
1.4 Metodologías de desarrollo de software.....	6
1.4.1 Metodología Extreme Programming.....	7
1.4.2 Selección de metodología.....	9
1.5 UML (Unified Modelling Language o Lenguaje Unificado de Modelado)	9
1.6 Lenguajes de programación.....	10
1.7 Frameworks analizados	10
1.7.1 Del lado del Servidor	10
1.7.2 Del lado del Cliente.....	13
1.8 Entorno de desarrollo: Netbeans 7.4.....	14
1.9 Herramienta de modelado: Visual Paradigm.....	14
1.10 Servidor Web: Apache	15
1.11 Sistema gestor de base de datos.....	15

1.12	Selección de tecnologías	16
1.13	Conclusiones	17
2	CAPÍTULO 2: Características del Sistema, Exploración y Planificación	18
2.1	Introducción	18
2.2	Problemática.....	18
2.3	Objeto de informatización	18
2.4	Propuesta del sistema	18
2.5	Características no funcionales del sistema	20
2.5.2	Apariencia o Interfaz Externa.....	21
2.5.3	Seguridad	21
2.5.4	Confidencialidad	21
2.5.5	Requisitos de Hardware. Especificaciones	22
2.5.6	Requisitos de Software	22
2.5.7	Usuarios relacionados con el sistema.....	22
2.6	Fase de Exploración	23
2.7	Planificación	28
2.7.1	Estimación de esfuerzo por Historia de Usuario.....	29
2.7.2	Plan de duración de las iteraciones	30
2.7.3	Plan de entregas.....	31
2.8	Conclusiones	31
3	CAPÍTULO 3. Análisis y Diseño del Sistema.....	32
3.1	Introducción	32
3.2	Arquitectura	32
3.2.1	Estilo de arquitectura: Modelo Vista Controlador	32
3.3	Estilos de código.....	34

3.4	Patrones de diseño.....	35
3.4.1	Patrones para Asignar Responsabilidades (GRASP).....	35
3.5	Diseño de la Base de Datos.....	37
3.5.1	Modelo Entidad - Relación.....	37
3.6	Tarjetas Clase - Responsabilidad - Colaborador (CRC).....	37
3.7	Conclusiones.....	40
4	CAPÍTULO 4. Implementación y Prueba.....	41
4.1	Introducción.....	41
4.2	Fase de Implementación.....	41
4.2.1	Iteraciones.....	41
4.3	Diagrama de Despliegue.....	49
4.4	Pruebas.....	50
4.4.1	Pruebas Unitarias.....	50
4.4.2	Pruebas de Aceptación.....	50
4.5	Conclusiones.....	59
5	Conclusiones Generales.....	60
6	Recomendaciones.....	61
7	Referencias Bibliográficas.....	62
8	Anexos.....	64
8.1	Anexo 1.....	64
8.2	Anexo 2.....	72
8.3	Anexo 3.....	74

INTRODUCCIÓN

El hombre desde sus inicios vio representada entre sus necesidades el atesoramiento para próximas generaciones del conocimiento adquirido, de manera que no pereciera en el tiempo la sabiduría alcanzada por él mismo y sus antepasados, asegurando de esta forma un desarrollo continuo de su pueblo y, como se ha demostrado posteriormente, de la humanidad toda. A raíz de esta necesidad el hombre acudió a la naturaleza para hacerse de medios que le permitiesen transmitir, sin importar el tiempo mediante, una idea, un concepto, o inclusive la representación gráfica de un momento vivido. Como pruebas imperecederas del alcance de la creatividad y capacidad de inventiva del hombre, aún hoy se pueden apreciar las evidencias de su estancia en cavernas por todo el mundo. De hecho uno de los mayores avances en el entendimiento de los lenguajes utilizados por este en la edad antigua fue el hallazgo de la piedra Rossetta¹ grabada con una diversidad de símbolos que permitió el estudio de estos y su traducción al lenguaje actual.

La evolución del hombre y especialización de sus instrumentos de trabajo estableció nuevas técnicas y tecnologías que fueron transformando y mejorando la forma de transmitir información, llegando a lo que se conoce hoy día como la televisión, fotografía, internet, etc... sin embargo, es indudablemente el momento en el que se inventa la imprenta cuando se establece el punto de ruptura entre una sociedad en que la transmisión de ideas y su tiempo de vida eran determinados prácticamente en su totalidad por la tradición oral a una sociedad, y cuando se habla de sociedad se dice en su concepto más amplio, que es capaz de hacer trascender en el tiempo prácticamente todo conocimiento generado de manera eficiente y efectiva. Muy pronto las imprentas se convirtieron en prósperos negocios capaces de generar grandes dividendos monetarios o simplemente en medios utilizados por minorías para dar a conocer sus ideas sobre los más diversos temas. Sin duda alguna su desarrollo se convirtió en una necesidad para una sociedad cada vez más ávida de información.

En la actualidad la imprenta se mantiene como uno de los principales medios de transmisión de información pese a que el propio desarrollo del hombre haya llevado a este a arribar a nuevas soluciones para este fin. Es por esto que en el mundo pese a la existencia de pequeñas impresoras capaces de llegar a cada oficina u hogar aún existe la necesidad de grandes imprentas capaces de una gran productividad a costos increíblemente bajos pero con una necesidad de insumos y de toda una logística alrededor del proceso de

¹ La piedra Rossetta es uno de los pilares del estudio del lenguaje, pues no fue hasta su hallazgo que se llegó a comprender el sentido y la riqueza alcanzada por el hombre que llegó a tal desarrollo en su lenguaje.

impresión que son realmente retadoras debido a la necesidad de anticipación en la compra de todos los productos necesarios para mantener una producción constante.

En nuestro país se ha recurrido a las grandes imprentas para mitigar la necesidad de este tipo de servicio de impresión a gran escala por lo que también se han encontrado problemas en la gestión de la producción de estos centros. Específicamente la imprenta Sagrado Corazón se ha encontrado con la necesidad de un sistema que le permita controlar la compra oportuna de insumos, así como el control de los costos de operación y otros aspectos de gran importancia. Actualmente el control de estos aspectos se realiza con un sistema de tarjetas que si bien permite dar solución a algunos de estos problemas, es sin duda un método obsoleto y complicado que no es capaz de brindar la seguridad necesaria para el control de unos pedidos cada vez mayores en número y monto de impresión. Además convierte el control económico y la planificación en procesos de una alta complejidad que los hace prácticamente inviable cuando deberían ser estos piedras angulares en la gestión de esta imprenta.

A partir de esta situación se deriva el siguiente **problema a resolver**:

¿Cómo mejorar el control de los procesos de la imprenta Sagrado Corazón?

Por tanto el **objeto de estudio** lo constituyen los sistemas que permitan gestionar procesos similares, enmarcando en el **campo de acción** a los Sistemas de Gestión basados en plataformas web.

Se define como **objetivo general** desarrollar un sistema informático para el control de los procesos de la imprenta Sagrado Corazón. Se derivan de este los siguientes **objetivos específicos**:

- Establecer el marco teórico-referencial de la investigación a partir del estudio de las particularidades de los sistemas de pedidos existentes.
- Diseñar la solución para el control de los procesos de la imprenta Sagrado Corazón.
- Implementar la solución para el control de los procesos de la imprenta Sagrado Corazón.
- Validar la solución propuesta.

Por tanto la **idea a defender** plantea que el desarrollo de un Sistema de Control de Pedidos para la imprenta Sagrado Corazón permitirá mejorar la gestión de los pedidos, pagos e insumos de esta entidad.

Es necesario dar cumplimiento a las siguientes **tareas de investigación** en aras de lograr materializar los objetivos planteados.

- Definir métodos teóricos y empíricos que serán empleados en el diseño metodológico de la investigación.
- Caracterizar aplicaciones que posean características similares a la que se desea implementar.
- Determinar herramientas y tecnologías a utilizar para el desarrollo del sistema.
- Definir la metodología de desarrollo de software a utilizar.
- Realizar la implementación de la propuesta de solución.
- Validar el sistema mediante pruebas de aceptación.

En el transcurso de la investigación se utilizaron distintos métodos que entran dentro de la clasificación de empíricos, destacándose entre ellos la observación. Debido a su utilidad en la comprensión del funcionamiento de la imprenta y como es su flujo de información es en definitiva este método el que permitió conocer los aspectos en los que esta actualización tecnológica permitiría mejorar el proceso productivo.

De acuerdo con las necesidades enfrentadas en la fase investigativa se acudió además a métodos que entran en la clasificación de teóricos como el inductivo-deductivo para comprender cómo el framework de desarrollo Symfony realiza el procesamiento de las peticiones. Además se utilizó el método analítico-sintético para esgrimir las funcionalidades necesarias para cumplimentar los objetivos y satisfacer las necesidades del cliente.

En aras de exponer de manera clara y concisa el proceso de investigación, desarrollo y prueba, el presente trabajo se estructura de la siguiente manera:

Capítulo 1. Fundamentación Teórica: Se realiza un análisis de las tecnologías que a partir de la investigación se encuentran como necesarias para la implementación. Se fundamentan además los conceptos, metodologías, lenguajes, y herramientas utilizadas.

Capítulo 2. Exploración y Planificación: Se profundiza en el sistema a desarrollar y sus características, se definen iteraciones a realizar y su duración, se definen también estimación de esfuerzo de las distintas historias de usuario.

Capítulo 3. Diseño del Sistema: Se realiza un acercamiento a las tarjetas: Tarjetas de Cargo o Clase, Responsabilidad y Colaboración (CRC), también a los patrones de diseño y finalmente se representa la estructura que tendrá la base de datos.



Capítulo 4. Implementación y Pruebas: Aquí se analiza lo relacionado a los procesos de implementación y pruebas del sistema, y más particularmente la identificación de tareas de ingeniería.

1 CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se realizará un análisis de los distintos conceptos y tecnologías que se han analizado en la presente investigación. De estos se pueden mencionar: el framework Symfony, el protocolo HTTP² y el IDE³ NetBeans que de una forma u otra son parte de la solución debido a su utilización directa o indirecta en el desarrollo de la aplicación. De todos se enuncian sus principales características y razones para optar por su uso en la presente investigación.

1.2 Conceptos fundamentales relacionados con el dominio del problema

1.2.1 Lenguajes de programación y entornos web

La Real Academia Española define el término lenguaje como el “conjunto de sonidos articulados con que el hombre manifiesta lo que piensa o siente” o al “conjunto de señales que dan a entender algo” y se define para el ámbito informático como el “conjunto de signos y reglas que permite la comunicación con un ordenador.” (1).

Un lenguaje de programación se puede definir como un idioma artificial diseñado para expresar tareas a realizar por máquinas como las computadoras, que puede usarse para crear programas que controlen el comportamiento físico y lógico de la máquina, para expresar algoritmos con precisión, entre otros.

Los lenguajes de programación son las herramientas necesarias que permiten la creación de los programas informáticos; están formados por un conjunto de símbolos y reglas que definen su estructura y el significado de sus elementos y expresiones.

Un lenguaje de programación de alto nivel se caracteriza por expresar los algoritmos necesarios para la creación de los programas informáticos de una manera adecuada a la capacidad cognitiva humana en lugar de a la capacidad ejecutora de las máquinas, es decir, facilita la comunicación con un computador mediante signos convencionales cercanos a los de un lenguaje natural.

² HTTP: Acrónimo de Hypertext Transfer Protocol o Protocolo de Transferencia de Hipertexto, es el protocolo de comunicación más difundido en la Internet puesto que sobre él descansa la infraestructura de las páginas web.

³ IDE: Integrated Development Environment o Entorno Integrado de Desarrollo.

En la propuesta de solución se utiliza el lenguaje de programación PHP debido a sugerencias del cliente, coincidiendo esta recomendación con el área de experticia de los desarrolladores. Sobre este lenguaje y otros relacionados con la programación web específicamente se realiza un análisis en los subsiguientes epígrafes.

1.2.1.1 PHP (Hypertext Preprocessor)

PHP, acrónimo de "PHP: Hypertext Preprocessor", es un lenguaje de 'scripting' de propósito general y de código abierto que está especialmente pensado para el desarrollo web y que puede ser embebido en páginas HTML. Su sintaxis recurre a C, Java y Perl, además es fácil de aprender. La meta principal de este lenguaje es permitir a los desarrolladores web escribir dinámica y rápidamente páginas web generadas; aunque se puede hacer mucho más con PHP. (2)

1.2.1.2 HTML (Hypertext Markup Language)

HTML fue originalmente desarrollado por Tim Berners-Lee mientras este laboraba en el CERN⁴ y fue popularizado por el navegador Mosaic desarrollado en la NCSA⁵. Durante el curso de los años 90 fue creciendo su popularidad según aumentaba el uso de las redes de interconexión de computadoras para el intercambio de información. En este tiempo HTML fue extendido varias veces para cumplir con su objetivo de manera más efectiva. (3)

1.3 Sistemas de Gestión

Un sistema de gestión sin importar el entorno en que se utilice es sin duda una herramienta que se hace indispensable para mantener un buen funcionamiento del negocio. Sin embargo es el propio sistema de gestión por donde debe siempre comenzar el proceso de mejora, asegurando de esta forma que el control de la actividad que se realiza no derive en costos innecesarios, ya sea de tiempo o monetarios. Es por esto que la informatización de estos sistemas ha presentado un impacto positivo debido a que ha ofrecido la posibilidad de llegar, incluso en los procesos más complejos, a permitir controles en tiempo real con una precisión muchas veces mayor que los antiguos sistemas pese a que estos últimos conllevaban más tiempo de análisis y cálculos.

⁴ Organización Europea para la Investigación Nuclear

⁵ Centro Nacional de Aplicaciones de Supercomputación

Ante una necesidad de este tipo la industria del software no tardó en dar respuesta, por lo que en la actualidad existen varios productos informáticos con este objetivo.

1.3.1 Cumulus Software

Cumulus Software es usado para el control de almacenes, pedidos y distribuciones. Sin dudas una solución poderosa y de un costo económico en comparación a otras soluciones existentes en el mercado, haciendo de este un producto atractivo para almacenadoras, centros de distribución y empresas que manejan su propio almacén.

Cumulus cuenta con dos versiones que le ayudan a maximizar la eficacia y productividad de la entidad, estas son La Estándar y la Profesional con Radiofrecuencia (WiFi). Ambas versiones constan de funciones básicas como: control de inventarios, administración de inventarios, control de pedidos y control de distribución, recibo de productos y surtido de órdenes. La versión Profesional, además incluye funciones como mejora continua en el nivel de servicio de su operación, línea de surtido, administración de back order⁶ y re-abasto de áreas de surtido.

Es un software completamente escalable en funcionalidad y plataforma de Base de Datos. Como consecuencia de esta flexibilidad, una empresa puede pasar de la Versión Estándar a la Profesional cuando las necesidades de su negocio así lo requieran.

Principales ventajas:

- Aumentan el nivel de servicio a sus clientes.
- Reducen gastos operativos.
- Son amigables con los usuarios.
- Mejoran el proceso de toma de decisiones.

Los requerimientos de hardware son Pentium III, 256 Mb de RAM, Pantalla de 15", Resolución mínima 1024x768, 16-bit color video, mouse, 25 Mb de espacio en disco duro para Cumulus y 1 Gb de espacio en

⁶ Un cliente que no pueda ser atendido en el momento debido a que la compañía no cuenta con la capacidad en ese momento para suplir el servicio genera un back order siempre que esté dispuesto a esperar para recibir el servicio o producto. El porcentaje de elementos en back order y el número de días que representa de espera son cifras de gran importancia para un negocio.

disco duro para la base de datos y tarjeta de red, los sistemas operativos soportados son Windows 98 o XP.
(4)

1.3.2 EcoSoftCS.net

EcoSoftCS.net fue especialmente diseñado en aras de facilitar la tarea de introducción de datos, haciendo posible que con el menor número de pasos a realizar pueda cargar su pedido y partiendo de este, terminar el ciclo de compra y gestión de stock. Por supuesto, contempla todas las posibilidades de facturación de proveedores, tales como agrupación de albaranes y abono de mercancía. Dispone de estas potentes herramientas también en el área de facturación de ventas.

Este sistema intercambia mercancía entre sus diferentes sucursales y factura a empresas del grupo con distribución asociada, no tiene que facturar y posteriormente hacer la distribución, sino que puede realizar las dos operaciones automáticamente desde la misma opción.

Los requerimientos de hardware son mínimos, pudiéndose utilizar en casi cualquier equipo, los sistemas operativos soportados son Windows® 98, 98Se, ME, XP, 2000, 2003, Windows® Vista y Windows® 7 (5)

1.3.3 SAGE ERP X3

Es uno de los software de gestión empresarial integral más avanzado para medianas y grandes empresas. Esta solución integra todas las funcionalidades de la empresa en las áreas de finanzas, ventas, compras, CRM, producción y logística, lo que garantiza una gestión consistente de los datos y un control global de la actividad en tiempo real. Siendo estas algunas de las características que hacen de este un producto de calidad que se destaca por su potencia.

Se puede acceder a todas las funcionalidades de Sage ERP X3 (Enterprise Resource Planning⁷) desde un simple navegador web, ya sea a nivel local desde la red interna de la empresa, o con un acceso remoto vía Internet.

Permite que la compañía se extienda de manera segura y controlada con todas las posibilidades del e-business facilitando:

- Optimizar los flujos de gestión internos y externos de la empresa.
- Integrar toda la gestión de las unidades descentralizadas, filiales o sucursales.

⁷ ERP: En español Planificador de Recursos Empresariales

- Facilitar la relación con los proveedores, partners o colaboradores y clientes.

Teniendo todas estas funcionalidades este software tiene solo como gran inconveniente una curva de aprendizaje relativamente alta que exige de sus usuarios conocimientos en temas económicos que van más allá de los objetivos de control económico que presenta la imprenta Sagrado Corazón. (6)

1.3.4 OpenERP

OpenERP, próximamente ODOO, debido a cambios anunciados por el equipo de desarrollo, es una potente herramienta para la gestión de negocios. En el mundo aproximadamente dos millones de personas o negocios utilizan este sistema. Entre sus principales ventajas se pueden destacar (7):

- Soporte y ayuda en el análisis de requisitos para la gestión del negocio por parte del equipo de desarrollo.
- Aproximadamente 4000 aplicaciones utilitarias al alcance de cualquier usuario.
- Posee una licencia libre.

Desventajas:

- Licencia comercial no compatible con software libre.
- Es necesario el acceso a internet para poder utilizar este sistema.
- A pesar de poseer una licencia libre está limitada a solo dos usuarios.

1.3.5 Stock Mistral

Este software resulta de particular utilidad en la toma de decisiones. En Cuba ha sido utilizado en empresas vinculadas a las FAR. Diseñado con el objetivo de disminuir niveles de stock, reducir los recursos inmovilizados e incrementar la liquidez en la empresa. Posee herramientas de enorme utilidad y potencia que permiten crear, en tiempo real, almacenes virtuales a diferentes niveles, con las existencias y movimientos de sus dependencias, permitiendo organizar el proceso de compras, distribución y pedidos. Brinda prestaciones como el completo control de la consignación comercial, generación de ofertas, conciliación con proveedores, así como el tratamiento de los lotes y sus vencimientos (caducidad), logrando su total aplicación para productos perecederos como alimentos y medicamentos. Tiene como principal aspecto negativo que es software privativo y solo puede ser ejecutado sobre la plataforma privativa de Windows.

1.4 Metodologías de desarrollo de software

Dentro de las metodologías existen fundamentalmente dos tipos que contribuyen al desarrollo de software, asegurando en un caso y otro la calidad del producto terminado, estas son: las tradicionales y las ágiles.

Independientemente de las especificidades de una u otra, las metodologías de desarrollo pudieran definirse de la siguiente manera:

“Son un conjunto de métodos, reglas, que sirven como un modelo para desarrollar el software y que guían a la dirección del proyecto y a los componentes del desarrollo a realizar ciertas comprobaciones sistemáticas de modo que el producto o resultado del trabajo final tenga éxito o sea, que se cumplan los objetivos y los requerimientos, se entregue en tiempo, forma y cumpla con la calidad que requiere o que el cliente exija.” (8)

De las categorías anteriormente mencionadas es válido aclarar que las metodologías ágiles están basadas en heurísticas provenientes de prácticas de producción de código, están especialmente preparadas para sufrir cambios durante el proyecto y son un proceso menos controlado y con pocos principios, por lo que desarrollan software en cortos períodos de tiempo y exigen poca documentación. Dentro de las metodologías ágiles se encuentra Extreme Programming (XP). (9)

Por otro lado se pueden encontrar las metodologías tradicionales que responden a otros intereses en cuanto a tiempo de desarrollo y control de este. Pudiendo ser definidas de la siguiente manera:

“Las tradicionales están basadas en normas provenientes de estándares seguidos por el entorno de desarrollo; ofrecen por lo general cierta resistencia a los cambios que puedan producirse durante el ciclo de desarrollo del producto, y es un proceso muy bien controlado con muchas políticas y normas.” (8)

Dentro de este grupo se encuentra Rational Unified Process (en lo adelante RUP), que responde a un proceso iterativo e incremental por lo que se espera una nueva versión del sistema en cada iteración respondiendo a la necesidad de mantener la calidad durante todo el ciclo de vida del proyecto y por tanto asegurando de esta forma la calidad final del producto. No obstante este método de trabajo presenta algunas desventajas que pueden hacerlo inviable para el desarrollo de ciertas aplicaciones, estas son:

La generación de un gran cúmulo de documentación que no genera valor respecto a la calidad del desarrollo y que necesita incluir un mayor número de personas al equipo de desarrollo, tales como especialistas en casos de uso. (10)

De esto último se decanta que en el desarrollo de aplicaciones con necesidad de una rápida implementación o con poco personal para el desarrollo no es aconsejable la utilización de estas metodologías tradicionales.

1.4.1 Metodología Extreme Programming

Es una metodología ágil, que centra su atención en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, para esto promueve el trabajo en equipo, se preocupa por el aprendizaje de los desarrolladores y propicia un buen clima de trabajo para los mismos. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, en la comunicación fluida entre todos los participantes y en el coraje para enfrentar los cambios. Está diseñada para mitigar los riesgos en proyectos con corto tiempo de entrega y continuos cambios de requerimientos. También cuando se exige reducir drásticamente el tiempo de desarrollo manteniendo siempre una alta calidad. (9)

La metodología Extreme Programming (XP) consta de 6 fases:

- Exploración: En esta fase, los clientes plantean a grandes rasgos las historias de usuario (en lo adelante HU) que son de interés para la primera entrega del producto, mientras que el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto.
- Planificación de la entrega: En esta fase el cliente establece la prioridad de cada HU, mientras que los programadores realizan una estimación del esfuerzo necesario de cada una de ellas.
- Iteraciones: Esta fase incluye las iteraciones que se van a realizar sobre el sistema, antes de ser entregado.
- Producción: En esta fase se requieren pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente.
- Mantenimiento: Esta fase contempla el mantenimiento del sistema luego de que la primera versión se encuentre en producción.
- Muerte del proyecto: Esta fase presenta vital importancia ya que el cliente no tiene más historias para ser incluidas en el sistema, por lo que se deben satisfacer otras necesidades, como son: el rendimiento y la confiabilidad del sistema.

Debido a la importancia que estas presentan para la implementación del sistema es válido especificar que las HU se pueden definir como la técnica que utiliza la metodología XP para lograr identificar los requisitos del software. Estas deben escribirse desde el punto de vista del cliente, pues en general identifican las

necesidades de este. No obstante el programador puede ayudar a identificarlas, pues al aportar su experticia es capaz de enfocarlas en los aspectos correctos.

Cada HU debe ser lo más sencilla y concreta posible pues los programadores deben ser capaces de obtener, de cada una de estas, estimaciones con bajos márgenes de riesgo de su tiempo de desarrollo, siendo responsabilidad del cliente asignarle prioridades. De la misma forma es parte del trabajo de los programadores la asignación de costos, que se traducen en las semanas que se desarrollará la implementación de la HU. Es válido destacar que nuevas HU pueden ser descritas e introducidas en el proceso de desarrollo en cualquier momento, siendo este uno de los puntos a favor de la metodología XP.

Las HU son clasificadas de la siguiente forma:

Alto: Las HU que son de carácter imprescindibles para el correcto funcionamiento de la aplicación desde el punto de vista del cliente, quien tiene la responsabilidad de darles la clasificación.

Medio: Son aquellas que se deben de tener en cuenta para el desarrollo sin que influyan decisivamente en el funcionamiento de la aplicación que se desarrolla.

Bajo: Es otorgada esta clasificación a las funcionalidades que le son de ayuda al control de los elementos asociados al equipo de desarrollo sin que tengan relación directa con el sistema desarrollado.

Sobre el riesgo en su desarrollo son clasificadas de la siguiente manera:

Alto: En caso de que se aprecie posibilidades reales altas de la introducción de códigos que aporte inestabilidad en cuanto al tiempo de desarrollo y el correcto funcionamiento.

Medio: En esta clasificación entran las HU que puedan ser causantes de retrasos en la entrega de la versión correspondiente del producto.

Bajo: Entran en esta clasificación las HU que no presentan riesgos de importancia para las entregas de las distintas versiones del producto.

Las HU serán representadas mediante tablas divididas por las siguientes secciones:

- Número: Número de la historia de usuario incremental en el tiempo.
- Nombre de Historia de Usuario: Debe ser un texto relacionado con el objetivo de la HU para una fácil identificación tanto por parte del desarrollador como del cliente.

- Usuario: Personas involucradas en el desarrollo de la HU.
- Iteración Asignada: Número de la iteración.
- Prioridad en negocio: Alto, Medio o Bajo.
- Riesgo en Desarrollo: Alto, Medio o Bajo.
- Puntos estimados: Tiempo estimado de desarrollo de la HU; cada punto es considerado como una semana de trabajo.
- Puntos Reales: Tiempo que se demoró en realidad el desarrollo de la HU.
- Descripción: Breve descripción de la HU.
- Observaciones: Señalamiento o advertencia del sistema.
- Prototipo de interfaz: Prototipo de interfaz si aplica.

1.4.2 Selección de metodología

Para guiar la implementación del sistema se decide emplear la metodología XP por ser este un proyecto pequeño donde todo el trabajo es realizado por una pareja de programadores, en un ambiente de trabajo donde el riesgo de desarrollo es elevado debido al corto tiempo de entrega planteado y a los continuos cambios de requerimientos; además el cliente forma parte del equipo de desarrollo, de esta manera se logra una mejor retroalimentación, corrección de errores y se garantiza la entrega de un producto final con la calidad requerida.

1.5 UML (Unified Modelling Language o Lenguaje Unificado de Modelado)

El Lenguaje Unificado de Modelado fue creado con el objetivo de llenar la brecha existente entre la idea de un proyecto y su implementación, permitiendo modelar uno o varios sistemas de manera que no sean necesarios procesos engorrosos para expresar de manera clara y concisa lo que deberían ser. Basado en tecnología orientada a objetos este es sin duda alguna una herramienta de uso obligatorio al menos en los proyectos de gran alcance, aunque los pequeños proyectos también podrían obtener grandes ventajas de este lenguaje. Es válido decir que involucra todo el ciclo de vida del proyecto y está pensado para varios lenguajes y plataformas.

1.6 Lenguajes de programación

En aras de desarrollar las funcionalidades necesarias para la aplicación se ha decidido utilizar el lenguaje PHP v5.2.5 del lado del servidor, al mismo tiempo del lado del cliente se estima conveniente utilizar los lenguajes HTML y JavaScript. Estos lenguajes permitirán ofrecer una experiencia de usuario adecuada pues dan la posibilidad de intercambio de información dinámico utilizando tecnología AJAX⁸.

También es válido destacar el uso de la librería MetroUI 2.1 en la maquetación del sitio web pues brinda un estilo moderno, minimalista⁹ y sobrio además de usable. Es compatible con los principales navegadores del mercado de manera oficial, aunque no existen motivos que impidan su correcta visualización en otros que cumplan los estándares de HTML5 y CSS3.

1.7 Frameworks analizados

Para lograr un desarrollo acorde a las necesidades existentes en cuanto a tiempo y requisitos del sistema se hizo necesario el análisis de varios frameworks o librerías de desarrollo tanto del lado del cliente como del servidor. En términos generales un framework es definido como un conjunto de códigos o utilidades previamente implementadas siguiendo diversos patrones que permiten a los desarrolladores reutilizar código y como consecuencia directa ahorrar ya sea tiempo de trabajo o esfuerzo. Aunque existen frameworks de desarrollo en todos los lenguajes de programación existentes, en la presente investigación se analizan solamente frameworks basados en el lenguaje PHP que es el seleccionado para la implementación de la propuesta de solución del lado del servidor y del lado del cliente se analizan librerías basadas en JavaScript.

1.7.1 Del lado del Servidor

En general los frameworks a evaluar deben ser capaces de brindar funcionalidades de acceso a datos, generación de pdf de manera nativa o mediante plugins de terceros entre otras funcionalidades genéricas que son de utilidad en cualquier proyecto implementado en PHP, un ejemplo pudiera ser el manejo mediante

⁸ Ajax, siglas de Asynchronous JavaScript and XML, es un término que describe un nuevo acercamiento a usar un conjunto de tecnologías existentes juntas, incluyendo las siguientes: HTML o XHTML, hojas de estilo (Cascading Style Sheets o css), Javascript, el DOM (Document Object Model), XML, XSLT, y el objeto XMLHttpRequest.

⁹ Minimalista: El minimalismo se define como una corriente del diseño donde se prioriza la funcionalidad sin olvidar la estética, pero sí redefiniéndola. Ejemplo de esta corriente es la interfaz Metro de Windows 8.

un sistema basado en POO¹⁰ del pedido HTTP realizado por el cliente. Otra de las razones para utilizar un framework sería para asegurar un código escalable y de fácil mantenimiento, por lo que esto es un aspecto de vital importancia en la selección de este.

1.7.1.1 ORM Doctrine

Doctrine es un conjunto de librerías PHP que tiene como principal objetivo la prestación de servicios de persistencia y sus funcionalidades relacionadas. Más explícitamente, es un ORM¹¹ que funciona como la parte superior de un DBAL¹². Entre sus características es el uso de un lenguaje propio, Orientado a Objetos, capaz de realizar consultas a la Base de Datos, su principal logro. De esta forma con este lenguaje llamado DQL¹³ se logran hacer consultas con rasgos similares al lenguaje SQL, de uso común en los distintos Gestores de Base de Datos, pero que cuenta con la ventaja de una capa de abstracción muy potente. Son estas ventajas las que le han ganado un puesto cimero entre la preferencia de muchos desarrolladores, permitiéndole contar con una comunidad activa tanto en uso como en desarrollo. Tiene una API muy completa que hace su integración con una aplicación desarrollada desde cero muy fácil e intuitiva en general. (11)

1.7.1.2 Framework CodeIgniter

CodeIgniter es un framework de PHP con un tamaño bastante pequeño, construido para programadores de PHP que necesitan una herramienta simple y elegante para crear aplicaciones web con todo tipo de funcionalidades y que cuentan con estrechas fechas de realización y entrega. (12)

Del framework CodeIgniter existen diversas características que se pudieran plantear, no obstante las más distintivas son:

- Un framework de trabajo de pequeño formato.
- Rendimiento excepcional, pues va cargando dinámicamente las librerías en el momento que lo requieran siguiendo un estilo lazy loading.

¹⁰ POO: Programación Orientada a Objetos , en este tipo de programación se programa imitando al mundo real pues se definen objetos con atributos y métodos que asemejan al comportamiento de la naturaleza. En este tipo de programación es también definida la herencia y el polimorfismo.

¹¹ Object-Relational Mapping o Mapeo Relacional de Objetos.

¹² Database Abstraction Layer o Capa de Abstracción de Base de Datos.

¹³ Doctrine Query Language.

- La configuración de este framework es realmente sencilla, algo que es indudablemente uno de sus puntos fuertes.
- Aboga por soluciones simples, haciendo el código de fácil interpretación por el programador aunque este esté ajeno al desarrollo previo que se haya realizado.
- Cuenta con una documentación muy clara y concisa, algo que es una necesidad en todo software de este tipo.

Desventajas:

- No cuenta con soporte para ORM.
- No cuenta con soporte para pruebas unitarias al código fuente.

1.7.1.3 Framework Yii

Yii es un framework PHP de alto rendimiento muy adecuado para el desarrollo de aplicaciones para la web 2.0. Yii viene con excelentes características como:

- MVC: Arquitectura Modelo Vista Controlador
- DAO/Active Record: Como métodos de acceso a datos.
- Configurado para permitir realizar pruebas unitarias.

Además de lo anterior es productivo mencionar que Yii carga solo lo necesario de manera que llega a ser extremadamente rápido y ligero. Otro de sus puntos fuertes es la seguridad, pues viene como estándar protegido de Inyección SQL y XSS¹⁴ entre otras vulnerabilidades conocidas.

Desventajas:

- Yii no aprovecha la compresión de JavaScript para emitir contenidos compactos en JavaScript compresado multi-browser. (13)

1.7.1.4 Framework Symfony

Symfony se destaca entre los frameworks más robustos que existen en el mercado actualmente. Además presenta entre sus principales ventajas su licencia que permite un uso comercial sin necesidad de realizar pagos a ninguna empresa. Entre sus principales características se encuentran una flexibilidad enorme en

¹⁴ Cross Site Scripting: Técnica utilizada para robar información en la web.

cuanto a posibilidad de uso sin sacrificar en ningún momento la organización del código generado. Es este un framework capaz de ser usado virtualmente en cualquier proyecto que involucre tecnología PHP, desde el más pequeño hasta el más abarcador. Entre sus puntos fuertes es imprescindible mencionar la capacidad de sus desarrolladores de cohesionar distintos proyectos como Doctrine y Twig, que por sí solos eran muy útiles, y lograr obtener un framework que brinda las características de los productos antes mencionados pero con un extra aportado por el proyecto Symfony, la posibilidad de utilizarlos en conjunto sin necesidad de re-implementar soluciones para que estos funcionaran en conjunto. En esta filosofía de trabajo de Symfony se basa uno de sus principios medulares alta cohesión, bajo acoplamiento. (14)

1.7.2 Del lado del Cliente

1.7.2.1 Framework JQuery

JQuery es una pequeña, rápida y completa librería JavaScript. Su objetivo principal es simplificar la manera en que son realizadas las distintas acciones en JavaScript y estandarizar el trabajo independientemente del navegador. Esto lo logra debido a su filosofía de mantener compatibilidad con los métodos tradicionales de manipulación de los elementos HTML, el manejo de eventos, las animaciones y AJAX entre los distintos navegadores mediante una intuitiva API. Con una excelente combinación de versatilidad y extensibilidad JQuery ha cambiado la forma de escribir código en JavaScript de millones de programadores que se han vuelto adictos a un framework del que se pueden obtener jugosas ganancias de tiempo y productividad (15).

Este framework cuenta con una comunidad que lo soporta y permite su constante evolución y desarrollo. En la actualidad se puede afirmar que es este la más completa de las soluciones para la programación del lado del cliente, pues cuenta con miles de plugins y cada día se incrementan en número. (15)

1.7.2.2 Framework Dojo Toolkit

Dojo ahorra tiempo y permite incrementar su complejidad de funcionamiento según lo vaya necesitando el proyecto, por lo que es considerado un framework escalable. En aras de lograr esto se basa en los estándares web como su plataforma. Este desarrollo escalable constituye uno de sus puntos distintivos a la hora de desarrollar aplicaciones robustas tanto para entornos web como móviles. De esto se deriva que este sea un framework adecuado para el desarrollo desde pequeñas aplicaciones hasta grandes proyectos empresariales.

Además Dojo cuenta con una librería gráfica 2D llamada GFX que es independiente del navegador y abstrae del uso de tener que acceder a las capacidades de aceleración gráfica nativas del navegador. (16)

1.7.2.3 Framework Ext JS

Sencha Ext JS se precia de ser el estándar para el desarrollo de aplicaciones empresariales de gran alcance y complejidad, algo que aunque ha ido variando con el auge de popularidad de JQuery, no deja de ser cierto debido al uso intensivo que se le ha dado en este tipo de aplicaciones. Entre sus cualidades se pueden contar una gran cantidad de APIs , una documentación muy completa y una gran cantidad de ejemplos que le dan la posibilidad al programador novato acercarse a Ext JS de manera simple, y sin mayores complicaciones. De ahí que este framework tenga una curva de aprendizaje bastante suave respecto a la gran potencia y robustez que brinda.

Entre sus facilidades Ext JS brinda herramientas que facilitan el tratamiento de grandes cantidades de información, como grandes Grids de datos. Esta facilidad también ayuda a la hora de construir aplicaciones siguiendo el patrón MVC (Modelo Vista Controlador). (17)

1.8 Entorno de desarrollo: Netbeans 7.4

El IDE NetBeans es una herramienta que permite escribir, compilar, depurar y ejecutar programas. Pese a estar basado en tecnología Java, que necesita de una máquina virtual, este IDE cuenta con todos los requisitos para estar entre las primeras opciones por las que optar a la hora de buscar un desarrollo rápido de aplicaciones, no solo PHP sino de diversos lenguajes entre ellos el propio Java. Tiene soporte para trabajar con AJAX y la creación de aplicaciones orientadas a servicios (SOA), además de distintas herramientas de depuración que llegan a ser de particular interés una vez que se comienza a desarrollar, por ejemplo, aplicaciones de gran consumo de memoria.

Desde julio de 2006, NetBeans IDE es licenciado bajo la Common Development and Distribution License (CDDL), una licencia basada en la Mozilla Public License (MPL). Lo que hace que cada vez más desarrolladores de aplicaciones, estén utilizando NetBeans como IDE de desarrollo.

1.9 Herramienta de modelado: Visual Paradigm

Esta es una herramienta de tipo Computer Assisted Software Engineering (CASE en lo adelante) que da soporte al ciclo de vida de desarrollo de software en su totalidad. Abarcando desde el análisis y diseño hasta la construcción de pruebas y despliegue. Entre sus principales ventajas está que permite modelar todos los diagramas de clases.

De estas características se pueden destacar:

- Disponibilidad en varias plataformas (Windows, Linux).
- Capacidades de ingeniería directa e inversa.
- Puede ser utilizado en el diseño de aplicaciones web.
- Soporta UML versión 2.1.

Debido a las ventajas claras que ofrece esta potente herramienta y a que la Universidad de las Ciencias Informáticas posee una licencia de tipo comercial de este software, se opta por utilizarlo en aras de lograr disponer de su alta efectividad en el modelado.

1.10 Servidor Web: Apache

El Apache HTTP Server es un servidor web de código abierto; un esfuerzo de desarrollo de software de colaboración, destinado a crear una implementación de código fuente robusto. El proyecto es administrado conjuntamente por un grupo de voluntarios ubicados en todo el mundo que se comunican a través de Internet para planear y desarrollar el servidor y su documentación. Este proyecto forma parte de la Apache Software Foundation. Además, cientos de usuarios han contribuido con ideas, código y documentación para el proyecto. Apache es usado para muchas otras tareas donde el contenido necesita ser puesto a disposición en una forma segura y confiable. Permite la comunicación segura mediante TLS y autenticar usuarios a través de un servidor LDAP, entre otras. Posee una arquitectura modular, de código abierto y multiplataforma y es fácil de conseguir ayuda y soporte por la amplia comunidad de desarrolladores y seguidores en todo el mundo. (18)

1.11 Sistema gestor de base de datos

En este apartado no existe una restricción por cuestiones de incompatibilidad o algún otro asunto relacionado a estos particulares. No obstante debido a la necesidad de mantener el producto final libre de licencias que pudiesen llegar a afectar económica o judicialmente a la entidad destino de la aplicación, se decidió utilizar PostgreSQL. Este sistema ha demostrado ser de gran potencia y estabilidad lo que es sin duda un requisito para confiarle la gestión de la información recopilada por la aplicación.

Este Sistema Gestor de Base de Datos es activamente desarrollado por una enorme comunidad de desarrolladores y apoyado por una gran cantidad de grandes compañías que han delegado en él la responsabilidad de sus datos.

1.12 Selección de tecnologías

Para seleccionar la tecnología del lado del servidor se tuvieron en cuenta las siguientes necesidades:

- El uso de software libre, la tecnología a utilizar debe contar con licencias compatibles con esta filosofía.
- Curva de aprendizaje, en este aspecto es necesaria una tecnología que sea de fácil comprensión debido a la necesidad de ahorrar tiempo de preparación de los desarrolladores. Además de contar con soporte por parte de la comunidad y la existencia de una amplia documentación.
- Acceso a datos, se impone un sistema ORM como una necesidad debido a las facilidades que brinda en el manejo de datos.
- Flexibilidad y escalabilidad, se necesita una tecnología capaz de generar código de fácil mantenimiento y legibilidad.

Teniendo en cuenta los aspectos antes enunciados sería factible la utilización de Symfony o Yii debido a que son los que satisfacen las necesidades anteriores, pero evaluando otros aspectos de interés para el desarrollo e implementación del sistema es Symfony la principal opción debido a que permite la compresión de archivos css y js con el objetivo de ahorrar ancho de banda y disminuir tiempos de respuesta.

En el lado del cliente es definitivamente JQuery la opción que se impone por ser la librería más estable en el momento de realizar el estudio, además es meritorio destacar que este framework cuenta con la mayor comunidad de usuarios y de plugins desarrollados, aspectos estos que reflejan la madurez alcanzada por este framework.

En general se puede afirmar que el uso de las tecnologías de desarrollo seleccionadas permite obtener:

- Soporte para una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, Sybase MySQL, Informix, entre otras.
- Permite generar documentos en PDF.
- Permite generar gráficas de gran calidad mediante la utilización de plugins ya sea del lado del servidor o del cliente
- Un producto de código abierto permitiendo que los fallos de funcionamiento y seguridad se encuentren y reparen rápidamente sin infringir ningún tipo de restricción comercial o licencia.

1.13 Conclusiones

En este capítulo se realizaron análisis de varios sistemas de gestión que actualmente se encuentran en explotación alrededor del mundo y en nuestro país. No obstante el desarrollo e implementación del sistema es viable debido a que la propuesta de solución se enfoca en mantener la curva de aprendizaje del sistema muy inferior a sus competidores, además con el sistema a desarrollar se asegura la posibilidad de futuras mejoras debido a un código simple y escalable gracias a la utilización de Symfony. En una pequeña imprenta la implantación de un gran sistema de gestión sería engorroso debido a la necesidad de adiestramiento a trabajadores con pocos conocimientos económicos, por el contrario el sistema a implementar reflejará las necesidades de control de los procesos de la imprenta Sagrado Corazón de manera más simple y directa, teniendo además un funcionamiento similar a la solución de registro en tarjetas, en cuanto al flujo de trabajo y la información a recopilar. Se determinó además que la metodología más adecuada para el desarrollo del sistema es XP y las herramientas necesarias para su implementación son el IDE NetBeans y el servidor Apache. Entre las tecnologías a utilizar se encuentran Symfony 2.3, JQuery 1.9 y la librería CSS MetroUI 2.1.

2 CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA, EXPLORACIÓN Y PLANIFICACIÓN

2.1 Introducción

En el presente capítulo se describen las distintas funcionalidades que serán implementadas en el sistema. Se enfatiza además las fases de Exploración y Planificación propias de la metodología de desarrollo utilizada. Es en estas fases donde se definen las HU para cada una de las iteraciones definidas por el equipo de desarrollo.

2.2 Problemática

En el mercado existen numerosas aplicaciones web o escritorio que se encuentran destinadas a la gestión, sin embargo, como fue planteado en el capítulo anterior, poseen ciertas características que no los hacen idóneos para su explotación en la Imprenta Sagrado Corazón. Del estudio de algunos de estos sistemas, debidamente mencionados anteriormente, se obtuvo experiencia para realizar la propuesta de solución que este capítulo tiene la responsabilidad de introducir.

2.3 Objeto de informatización

Como parte de la solución se pretende informatizar la generación de comprobantes de los pagos realizados por los clientes, también se realizará esta operación para los pedidos. Otro de los aspectos que es necesario informatizar es la determinación del número de usuarios que presenta algún tipo de deuda con la entidad. Otro objetivo de la implementación es dar la posibilidad de graficar los distintos reportes que puedan ser obtenidos por la aplicación de manera que con un mínimo de esfuerzo el usuario pueda acceder a una valiosa información de forma mucho más fácil de interpretar.

2.4 Propuesta del sistema

Esta investigación tiene como meta agilizar y mejorar los procesos de registro de nuevos pedidos, clientes, pagos o insumos así como su gestión en cuanto a eliminación o modificación. También es de interés mantener el control del costo total de cada unidad impresa. Para esto se decide implementar un sistema web que puede ser representado de la siguiente manera. Ver Figura 2.1

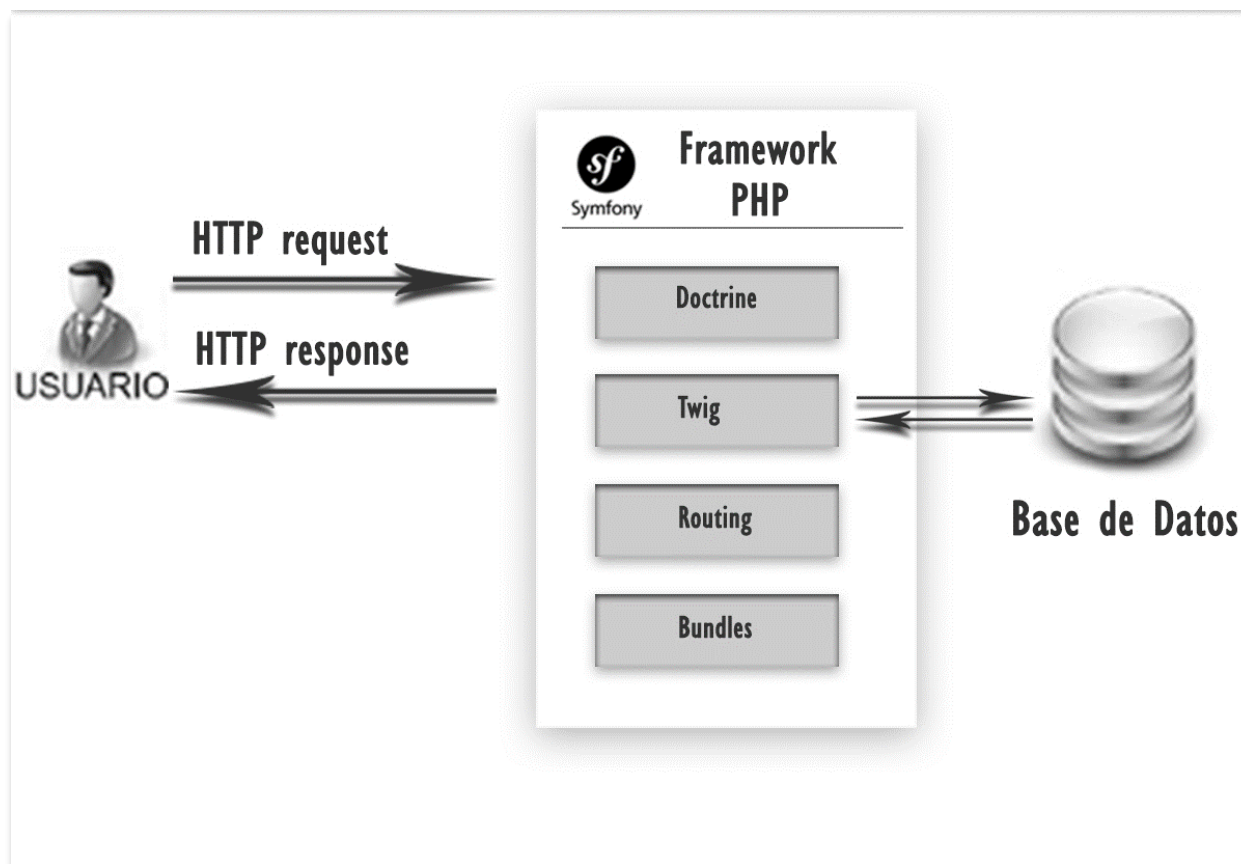


Figura 2.1 Propuesta General del Sistema

Más específicamente se puede precisar la intención de que el sistema maneje y genere la información necesaria para brindar gráficos que permitan un mejor análisis de los distintos datos que son de interés para la administración y gestión de la Imprenta Sagrado Corazón, entre estos datos se encuentran el número de clientes con pedidos en un determinado período de tiempo, la cantidad de pagos realizados en contraste a la cantidad de pedidos y otros datos de este tipo que son de utilidad para la administración del centro. Es también prioritario conocer y poder graficar la información de los clientes que mantienen deudas en la actualidad y su comportamiento históricamente, de manera que se pueda evaluar fácilmente quienes son los más morosos en realizar pagos. Otro aspecto de particular interés es mantener el control de los insumos que se adquieren en el negocio y de la misma forma graficar el comportamiento de las compras de estos a lo largo de distintos períodos de tiempo. De manera general en cada una de las opciones de graficar se hace necesario dar la opción de seleccionar el período de tiempo que se desea establecer para obtener los datos y mostrar para su análisis.

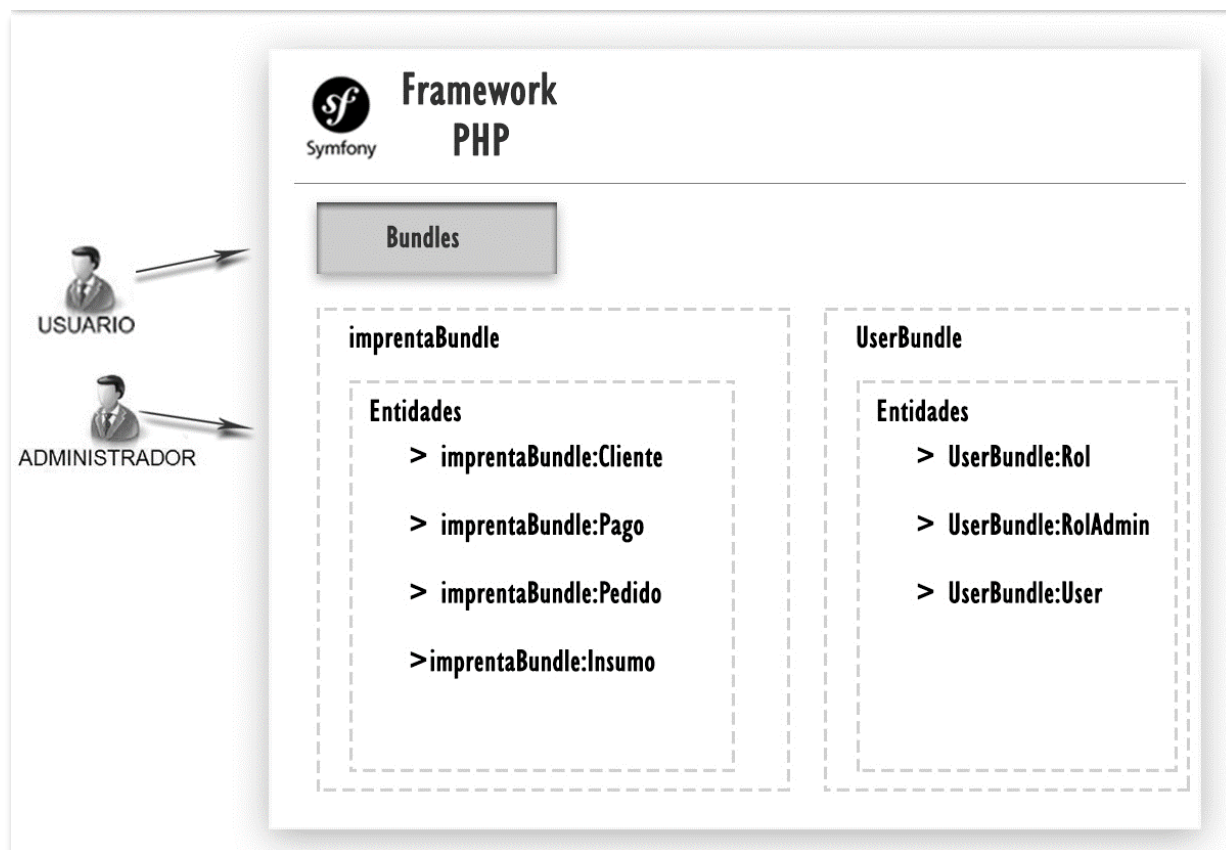


Figura 2.2 Propuesta Específica del Sistema

2.5 Características no funcionales del sistema

Los requisitos no funcionales pueden marcar la diferencia entre un producto con aceptación y uno con poca aceptación, son cualidades o propiedades adicionales que el sistema debe tener proporcionándole al cliente un producto atractivo, usable, rápido y confiable que sea capaz de llenar todas sus expectativas.

2.5.1.1 Apariencia o Interfaz Externa

Se requiere de una pequeña preparación previa para interactuar con el sistema, pues el manejo de la aplicación es sencillo. Para una fácil comprensión y adaptación del cliente con el sistema se realizó el producto semejante al estilo de trabajo manual con el que se trabajaba inicialmente.

2.5.2 Apariencia o Interfaz Externa

La interfaz del sistema deberá ser amigable y sencilla respondiendo a los principios del diseño minimalista y más específicamente a las pautas de la interfaz Metro¹⁵ lo que le asegura una impresión visual moderna e intuitiva, además de mantener una consistencia con las tendencias del diseño web actual.

2.5.3 Seguridad

En este aspecto se destaca que la seguridad del sistema en general está orientada a roles, y por tanto en dependencia de los permisos dados al usuario autenticado en el sistema, se le dará o no acceso a las distintas funcionalidades. Es válido destacar que las configuraciones de seguridad realizadas solo permiten a los usuarios anónimos acceder al formulario de autenticación del sitio, por lo que siempre deberá encontrarse autenticado todo aquel usuario que pretenda hacer uso de determinada funcionalidad del sistema. Para este proceso de autenticación se hizo uso del sistema de seguridad brindado por Symfony, que se encuentra ampliamente probado y del que se han obtenido excelentes resultados no solo por su fácil implementación sino por su robusto diseño y fuerte seguridad. Ninguna contraseña se almacena en texto plano en la Base de Datos por tanto es imposible la obtención de las claves de seguridad incluso teniendo acceso a esta.

Más específicamente en cuanto a permisos dados a usuarios, se puede decir que estos tienen la posibilidad de crear y gestionar Clientes, Pagos y Pedidos una vez autenticados. Además de poder realizar las distintas consultas a la Base de Datos con el objetivo de graficar información o imprimir comprobantes de las distintas operaciones realizadas o de los estados actuales de los distintos pedidos, pagos realizados.

En cuanto a seguridad es destacado el trabajo de los logs de seguridad, que tienen como función registrar las distintas acciones realizadas por los usuarios, de manera que, en caso de existir algún problema se pueda conocer específicamente quién realizó la acción causante de la situación además de otras informaciones que pudieran ser de utilidad. A verificar los logs de seguridad tendrá acceso únicamente el rol Administrador quien además podrá crear y eliminar usuarios.

2.5.4 Confidencialidad

Debido al diseño de la seguridad basada en roles toda la información está debidamente compartimentada de manera que un usuario no pueda llegar a acceder o modificar una información para la que no tiene el

¹⁵ Interfaz Metro: Esta interfaz fue adoptada por Microsoft en su versión más reciente del sistema operativo de esta compañía, Windows 8.

nivel de autorización adecuado. Es válido especificar que el rol Administrador tendrá acceso a toda la información del sistema con excepción de contraseñas y otra información de este tipo.

2.5.5 Requisitos de Hardware. Especificaciones

Tabla 2.1 Requerimientos de Hardware

Servidores	Especificaciones
Servidor de Aplicaciones Web	<ul style="list-style-type: none"> • RAM: 1 GB o superior. • Disco Duro: 160 GB o superior. • UPS: 1.
Servidor de Base de Datos	<ul style="list-style-type: none"> • RAM: 1 GB o superior. • Disco Duro: 160 GB o superior. • UPS: 1.

2.5.6 Requisitos de Software

Tabla 2.2 Requerimientos de Software (Servidor)

Servidores	Especificaciones
Servidor de Aplicaciones Web	<ul style="list-style-type: none"> • Sistema Operativo: Ubuntu 11.04 o superior. • Servidor Web: Apache 2.0 o superior. • Librerías Adicionales: PHP 5 o superior. <li style="padding-left: 20px;">: php5-curl <li style="padding-left: 20px;">: php5-gd <li style="padding-left: 20px;">: libapache2-mod-php5
Servidor de Base de Datos	<ul style="list-style-type: none"> • Sistema Operativo: Windows o Linux o cualquiera compatible con PostgreSQL 8.4.7. • Sistema Gestor de Base de Datos: PostgreSQL 8.4.7 o superior o cualquiera compatible con Symfony 2.3

Tabla 2.3 Requerimientos de Software (Cliente)

Servidores	Especificaciones
Cliente Web	<ul style="list-style-type: none"> • Navegador Web: Mozilla Firefox 2.0+, Safari 3+, Opera 10.6+ Google Chrome 8+.

2.5.7 Usuarios relacionados con el sistema

Es tratado como usuario relacionado con el sistema todo aquel que de alguna forma interactúa con el sistema y obtiene algún resultado.

Tabla 2.4 Usuarios relacionados con el sistema

Usuario	Especificaciones
Administrador	Es el rol de mayor nivel de autorización en el sistema y por tanto tiene acceso a toda la información manejada en el mismo.
Usuario	Es quien realiza las entradas de Pagos, Clientes, Pedidos e Insumos.
Cliente	Puede registrar pedidos y visualizarlos, además puede visualizar sus pagos.

2.6 Fase de Exploración

Esta es la primera fase definida por la metodología XP, y tiene como comienzo la creación de varias HU que definen mediante su redacción qué es lo que desea obtener el cliente según se ha planteado en capítulos anteriores. En este paso es determinado el alcance del sistema y permite a los desarrolladores familiarizarse con las herramientas y tecnologías usadas en la solución. A continuación se muestran algunas HU del sistema, el resto pueden ser consultadas en el Anexo 1 del presente documento.

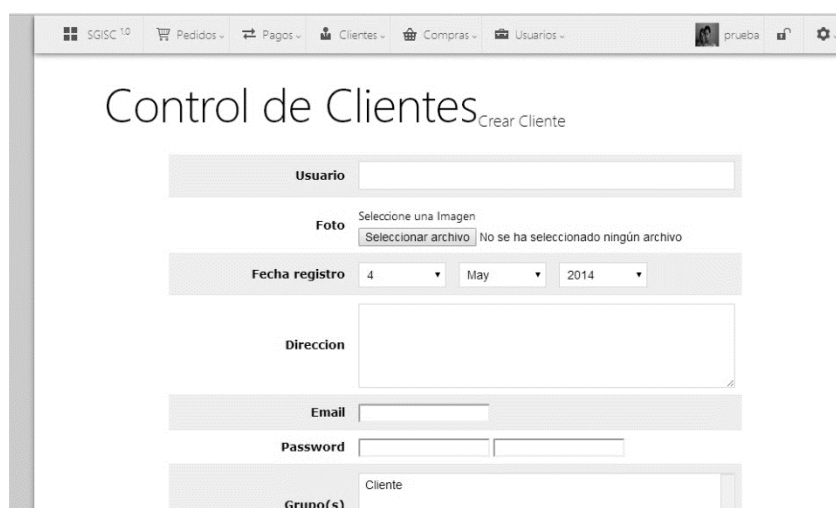
Tabla 2.5 HU: Registrar Cliente en el sistema

HISTORIA DE USUARIO	
Número: 1	Nombre de HU: Registrar cliente en el sistema
Usuario: Arletys Betancourt, Roy Castro	Iteración asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 1
Riesgo en Desarrollo: Alto	Puntos Reales: 1
Descripción: Esta funcionalidad tiene como objetivo la implementación de la parte de la aplicación encargada del registro de los clientes y por consiguiente de la incorporación al sistema de toda información asociada a estos.	

Observaciones:

Deben de recogerse los datos Nombre, Dirección, Fecha de Registro de cada cliente. Todos estos campos deben ser llenados obligatoriamente.

Prototipo de Interfaz:



The screenshot shows a web application interface titled "Control de Clientes" with a sub-link "Crear Cliente". The interface includes a navigation menu at the top with items like "Pedidos", "Pagos", "Clientes", "Compras", and "Usuarios". The main form contains the following fields:

- Usuario:** A text input field.
- Foto:** A section with the label "Seleccione una Imagen" and a button "Seleccionar archivo". Below it, a message states "No se ha seleccionado ningún archivo".
- Fecha registro:** A date picker showing "4", "May", and "2014".
- Dirección:** A large text area for address input.
- Email:** A text input field.
- Password:** Two text input fields for password and confirmation.
- Grupo(s):** A dropdown menu currently showing "Cliente".

Tabla 2.6 HU: Registrar Pedido en el sistema

HISTORIA DE USUARIO	
Número: 3	Nombre de HU: Registrar pedido en el sistema
Usuario: Arletys Betancourt, Roy Castro	Iteración asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 1
Riesgo en Desarrollo: Alto	Puntos Reales: 1

Descripción: Esta funcionalidad tiene como objetivo la implementación del registro de los pedidos y de la incorporación al sistema de toda información asociada a estos.

Observaciones:

Deben de recogerse los datos Cantidad, Precio Unitario vigente, Fecha del Pedido y Cliente que realiza el pedido. Todos estos campos deben ser llenados obligatoriamente.

Prototipo de Interfaz:



The screenshot shows a web application interface titled "Control de Pedidos" with a "Crear Pedido" link. The interface includes a navigation bar with icons for "Pedidos", "Pagos", "Clientes", "Compras", and "Usuarios". The main form contains the following fields:

- Cliente:** A dropdown menu with "prueba" selected.
- Fecha:** Three dropdown menus for day (4), month (May), and year (2014).
- Cantidad:** A numeric input field with "0" and up/down arrows.
- Atendido:** A progress indicator consisting of a black square followed by a grey bar.
- Crear Pedido:** A button at the bottom right of the form.

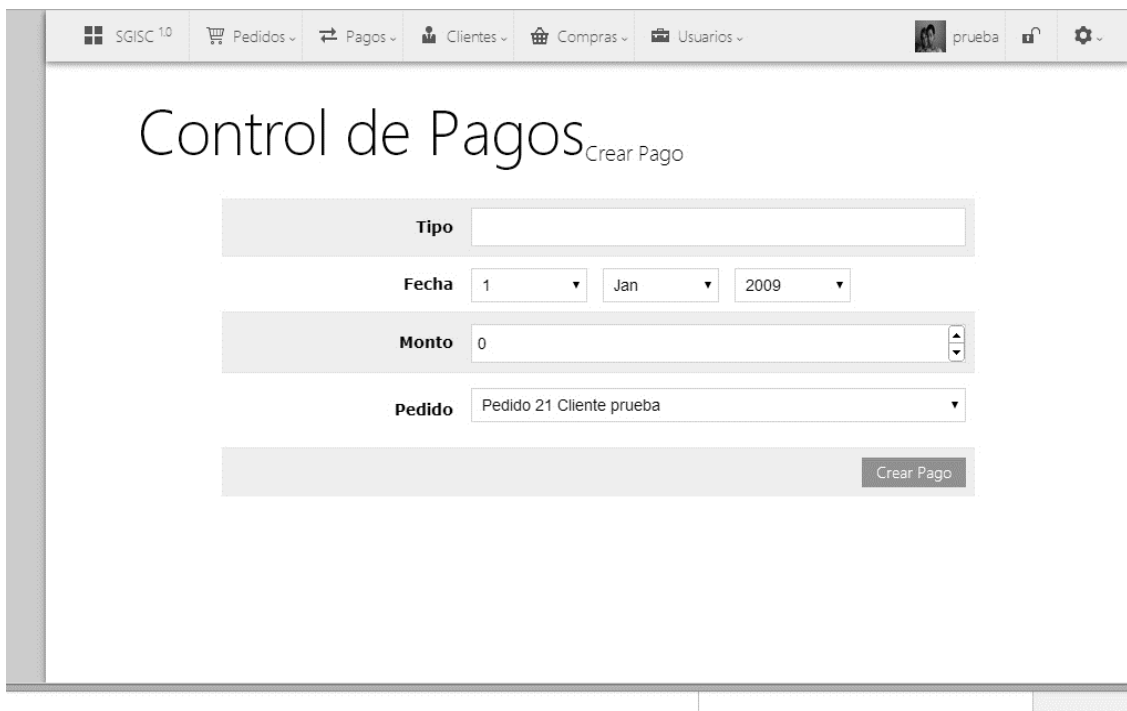
Tabla 2.7 HU: Registrar Pago en el sistema

HISTORIA DE USUARIO	
Número: 2	Nombre de HU: Registrar pago en el sistema



Usuario: Arletys Betancourt, Roy Castro	Iteración asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 1
Riesgo en Desarrollo: Alto	Puntos Reales: 1
Descripción: Esta funcionalidad tiene como objetivo la implementación del registro de los pagos y de la incorporación al sistema de toda información asociada a estos.	
Observaciones: En esta HU se debe recoger los siguientes datos: Tipo de pago (Efectivo o Giro), Fecha en que se realiza, Monto pagado y Pedido al que se le realiza el pago. Todos estos campos son de llenado obligatorio.	

Prototipo de Interfaz:



The screenshot shows a web application interface for 'Control de Pagos'. At the top, there is a navigation bar with icons and labels for 'Pedidos', 'Pagos', 'Clientes', 'Compras', and 'Usuarios'. The main content area features the title 'Control de Pagos' and a sub-label 'Crear Pago'. Below the title is a form with the following fields:

- Tipo:** A text input field.
- Fecha:** Three dropdown menus for day (1), month (Jan), and year (2009).
- Monto:** A numeric input field with a value of 0 and a vertical spinner.
- Pedido:** A dropdown menu with the selected value 'Pedido 21 Cliente prueba'.

A 'Crear Pago' button is located at the bottom right of the form area.

Tabla 2.8 HU: Graficar clientes

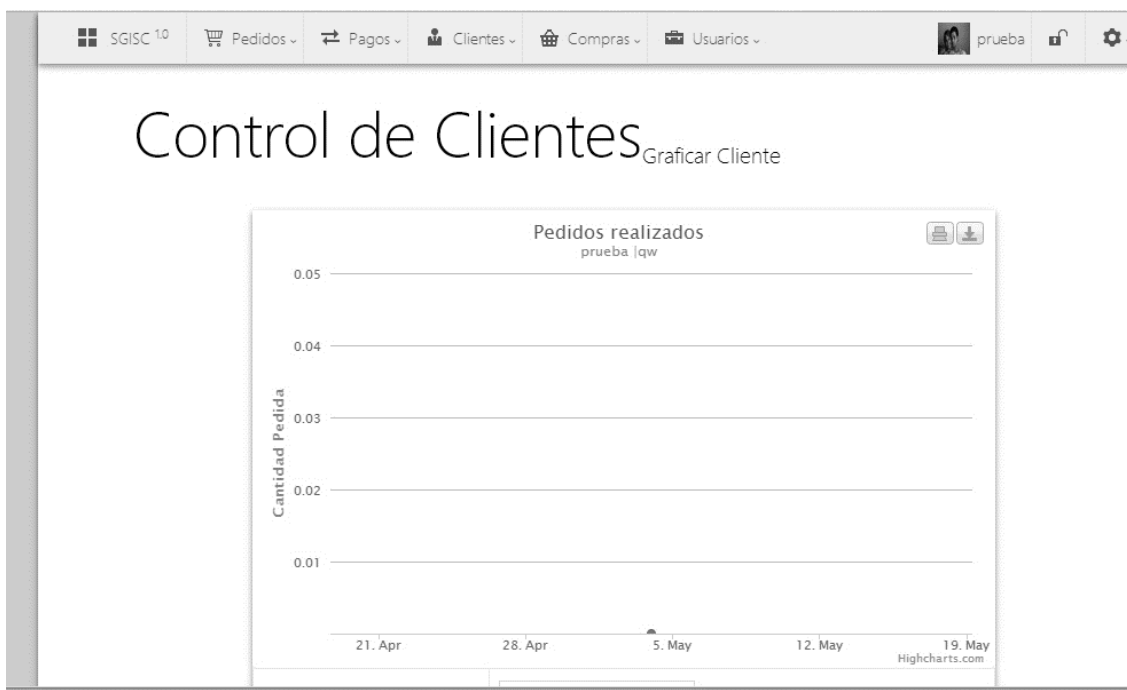
HISTORIA DE USUARIO	
Número: 4	Nombre de HU: Graficar clientes
Usuario: Arletys Betancourt, Roy Castro	Iteración asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 2
Riesgo en Desarrollo: Alto	Puntos Reales: 2

Descripción: Esta funcionalidad tiene como objetivo que de cada cliente se pueda graficar, seleccionando el período deseado, los pagos y pedidos realizados por este.

Observaciones:

En esta HU siempre debe estar disponible el cliente del que se desea graficar la información de pedidos y pagos. Además es necesario recoger el período de tiempo del que se desea conocer la información, aunque por defecto siempre se graficará la información del último año.

Prototipo de Interfaz:



2.7 Planificación

A las diferentes HU se les asigna su prioridad en la fase de planificación de la metodología XP, definiendo de manera clara y concisa las características y funcionalidades necesitadas por el sistema. Es además en este punto donde se le asigna una estimación del esfuerzo que costará la implementación de cada HU, tomando como unidad de medida el punto, al que se le da una equivalencia con una semana de trabajo a

tiempo completo. Es válido aclarar que aquí es incluido todo tipo de esfuerzo necesario para la correcta y completa implementación de cada HU.

2.7.1 Estimación de esfuerzo por Historia de Usuario

En el siguiente apartado se muestran las estimaciones de esfuerzo por cada una de las distintas HU identificadas:

Tabla 2.9 Estimación de esfuerzo por Historias de Usuarios

No	Historia de Usuario (HU)	Estimación de Esfuerzo
1	Registrar Cliente en el sistema	1
2	Registrar Pago en el sistema	1
3	Registrar Pedido en el sistema	1
4	Graficar Clientes	2
5	Graficar Pedidos	1
6	Graficar Pagos	1
7	Generación de comprobantes e informes de Clientes	2
8	Generación de comprobantes e informes de Pagos	2
9	Generación de comprobantes e informes de Pedidos	2
10	Registrar Insumo en el sistema	1
11	Generación de comprobantes e informes de Insumos	2
12	Gestión de Usuarios	1

2.7.1.1 Plan de iteraciones

Una vez identificadas y descritas las HU es necesaria una planificación de la fase de implementación, en la que se realizarán iteraciones. En este caso se realizarán dos iteraciones según se describe a continuación:

Iteración 1:

En esta iteración se contempla como objetivo la implementación de las HU 1, 2, 3, 7, 8, 9, 10 y 11 que son de prioridad alta para el sistema.

Iteración 2:

En esta iteración se contempla como objetivo la implementación de las HU 4, 5, 6 que son de prioridad media para el sistema.

2.7.2 Plan de duración de las iteraciones

Este plan se encarga de mostrar las Historias de Usuarios en el orden en que se implementarán en cada iteración, así como la duración estimada.

Tabla 2.10 Plan de duración de iteraciones

Iteración	Orden de las HU a implementar	Duración Total
Iteración 1	Registrar Cliente en el sistema	13 Semanas
	Registrar Pago en el sistema	
	Registrar Pedido en el sistema	
	Sistema generación de comprobantes e informes de Clientes	
	Sistema generación de comprobantes e informes de Pagos	
	Sistema generación de comprobantes e informes de Pedidos	
	Registrar Insumo en el sistema	

	Sistema generación de comprobantes e informes de Insumos	
	Sistema de gestión de Usuarios	
Iteración 2	Sistema de graficar Clientes	4 Semanas
	Sistema de graficar Pagos	
	Sistema de graficar Pedidos	

2.7.3 Plan de entregas

En este apartado se muestran los detalles de las fechas de inicio y fin de cada iteración, teniendo como resultado de cada iteración una versión funcional de la aplicación según su estado de desarrollo. A continuación se muestra el plan de entrega para cada iteración:

FPA: Funcionalidades con prioridad alta.

FPM: Funcionalidades con prioridad media.

Tabla 2.11 Plan de entrega de las iteraciones

Aplicación	Fin de la Iteración 1 (25 de marzo 2014)	Fin de la Iteración 2 (22 de abril 2014)
Sistema de Gestión de la Imprenta Sagrado Corazón	FPA	FPM

2.8 Conclusiones

En este capítulo quedó formalmente definida la propuesta de sistema realizada por el equipo de desarrollo, para lo que fueron identificadas las funcionalidades y requisitos que se espera este sistema cumpla. También fueron identificadas y definidas las distintas HU que se establecieron como premisas para dar solución a las necesidades del cliente. Es también un resultado la determinación del orden de implementación de estas HU.

3 CAPÍTULO 3. ANÁLISIS Y DISEÑO DEL SISTEMA

3.1 Introducción

Como parte de este capítulo es descrita la fase de diseño del sistema respondiendo de esta forma a la metodología XP. Siguiendo estas pautas, es definida la arquitectura y patrones de diseño que son necesarios seguir durante el desarrollo e implementación del sistema. Es también de carácter prioritario el diseño de la base de datos de manera que esta pueda responder a los requerimientos del sistema. Otro aspecto de vital importancia que queda resuelto es la definición de las tarjetas CRC (Colaborador-Responsabilidad-Clase) como técnica de diseño.

3.2 Arquitectura

En el desarrollo de software es la arquitectura a seguir un aspecto de vital importancia, debido a que esta es la encargada del diseño de componentes de una aplicación generalmente utilizando patrones de arquitectura. Un diseño arquitectónico describe en general el cómo se construirá una aplicación de software. (19)

3.2.1 Estilo de arquitectura: Modelo Vista Controlador

Esta arquitectura tiene como base la división del sistema en tres capas fundamentales, de lo que parte su nombre. De estas capas es importante resaltar la asignación de responsabilidades, bien diferenciadas en cada caso:

- Vista: Es la capa encargada de presentar información generada dinámicamente al usuario, generalmente páginas HTML, CSS y JavaScript aunque es capaz de generar todo tipo de documentos.
- Controlador: Es la capa encargada de responder a acciones del usuario.
- Modelo: Es donde es representada toda la información generada o recogida por el sistema que es necesario guardar para futuras consultas. En esta capa además se pueden hacer consultas para recuperar, modificar o borrar esta información.

La arquitectura utilizada en la presente propuesta de solución puede ser apreciada en la siguiente figura, Figura 3.1

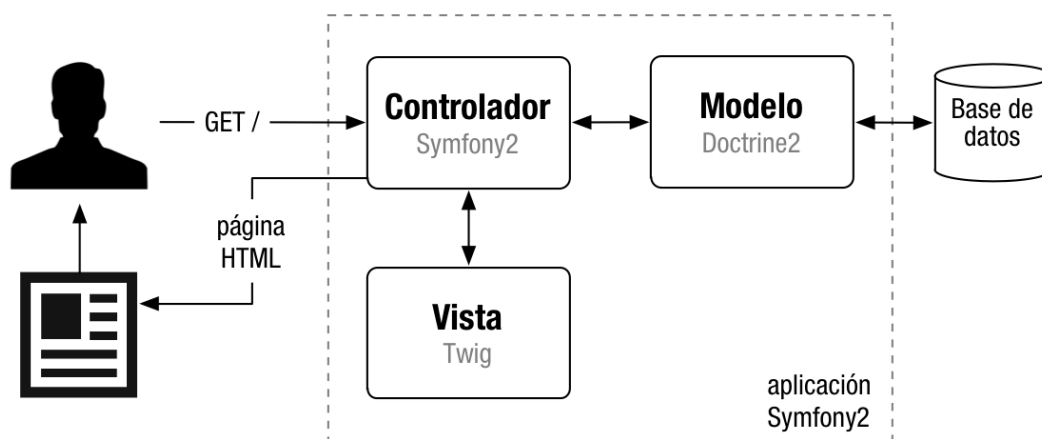


Figura 3.1 Arquitectura de la aplicación.

La arquitectura antes mencionada presenta como principal característica una muy clara separación de contenido y lógica del negocio por lo que brinda, entre otras ventajas, una gran adaptabilidad a requisitos cambiantes, algo muy común en proyectos medianos o pequeños desarrollados bajo la metodología XP. También es válido mencionar que esta arquitectura permite generar una gran cantidad de código reutilizable que deriva en tiempos de desarrollo menores sin sacrificio de calidad del código.

La arquitectura en capas está definida como una organización jerárquica tal que cada capa proporciona servicios a la capa superior y se sirve de las prestaciones que le brinda la inferior (19). De esto se puede inferir que cada capa es independiente del proceso que realiza la que le brinda el servicio, por tanto la escalabilidad y mantenimiento se hacen de manera muy fluida sin grandes impactos en tiempos de desarrollo. Es precisamente este el motivo por el que el resultado de la presente investigación aprovecha estas características y obtiene un bundle en el que es reflejada la arquitectura MVC de manera muy clara, pues fueron generadas diversas clases controladoras, múltiples entidades que tienen como finalidad definir el modelo de datos y todas las vistas necesarias utilizando el motor de plantillas Twig. Quedando una estructura de carpetas como se muestra a continuación:

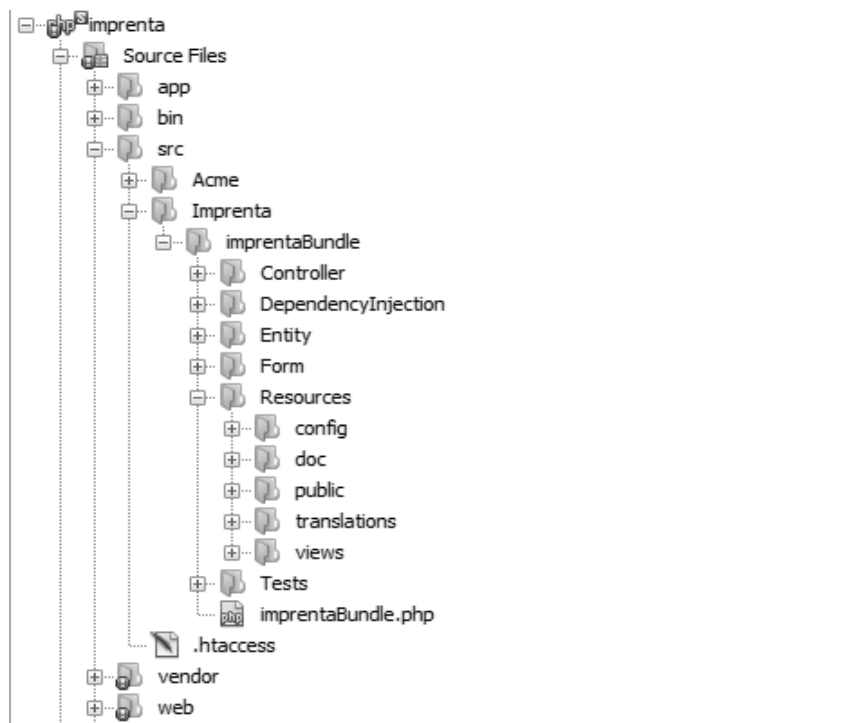


Figura 3. 2 Estructura de carpetas del bundle generado.

3.3 Estilos de código

Symfony define como estilo de código el Camel, por lo que en el desarrollo del sistema se sigue la siguiente estructura de código:

- No utilizar subrayados para nombres de variables, funciones o parámetros.
- Usar namespaces para todas las clases.
- Marcar como abstractas todas las clases que posean esta característica.
- Poner sufijo Interface en todas las interfaces.
- Poner sufijo Exception en todas las excepciones.
- Poner sufijo Controller a las clases que cumplan esta funcionalidad.
- Los nombres de clases comienzan con mayúscula y de tener más de una palabra cada una debe iniciar con mayúscula.
- Los nombres de funciones empiezan con minúscula y de tener más de una palabra cada una debe iniciar con mayúscula.

3.4 Patrones de diseño

En el desarrollo de una aplicación informática la solución de problemas de características similares, en distintos módulos, secciones, etc., es una realidad a la que se enfrentan los programadores comúnmente. En aras de mejorar la capacidad de darle solución a estos, muchas veces pequeños, pero muy recurrentes problemas, surgen los patrones de diseño a partir de soluciones ya probadas. De esta forma cada patrón tiene una descripción de la solución según el problema a tratar, permitiendo emplear la misma solución sin implementarla nuevamente. A continuación se exponen los principales patrones presentes en la solución implementada.

3.4.1 Patrones para Asignar Responsabilidades (GRASP)

Los Patrones GRASP describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades.

3.4.1.1 Bajo Acoplamiento

Este patrón es un principio que asigna la responsabilidad de controlar el flujo de eventos del sistema a clases específicas. Consiste en tener las clases lo menos ligadas entre sí, permitiendo que al producirse una modificación en alguna de ellas, tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. Este patrón es usado en nuestra aplicación al utilizar la inyección de dependencias, pues esta permite asignar responsabilidades a clases específicas en el sistema y utilizarlas según sea necesario. Un ejemplo de su uso puede ser encontrado en la clase ClienteController.

3.4.1.2 Alta Cohesión

Este patrón plantea que la información que almacena una clase debe ser coherente y estar estrechamente relacionada con la clase. De esta forma se realiza un diseño donde las clases mantengan una alta cohesión se mejora la claridad y facilidad con que se entiende el diseño, se simplifica el mantenimiento y las mejoras de funcionalidad, a menudo se genera un bajo acoplamiento, soportando mayor capacidad de reutilización. Este patrón también es reflejado en el trabajo realizado con la inyección de dependencias en la propuesta de solución. Tanto este patrón como el Bajo Acoplamiento pueden ser vistos en toda la aplicación pues la inyección de dependencias se utiliza ampliamente a lo largo de todo el proyecto. Un ejemplo de su uso puede ser encontrado en la clase PedidoController.

3.4.1.3 Patrón Experto

Este patrón se usa con el objetivo de darle a las clases las responsabilidades necesarias siempre que cuenten con la información para cumplirlas. Logrando así un mejor comportamiento entre las clases y haciendo que estas fueran más cohesivas, fáciles de comprender y mantener, permitiendo mayores facilidades de soporte. Un ejemplo claro de este patrón en la aplicación obtenida es el sistema de rutas utilizadas, que asegura que siempre que una clase controladora sea llamada a responder determinado pedido esta cuenta con la información necesaria para satisfacer la petición.

3.4.1.4 Patrón Creador

Este patrón permite tener una política general para la creación de objetos. Se usa en las clases controladoras para crear instancias de las clases del modelo, principalmente las del negocio, así como en las clases del negocio para instanciar las clases del dominio. Este patrón se manifiesta en gran parte de la solución al ser creadas instancias de las clases necesarias para ejecutar las funcionalidades.

3.4.1.5 Patrón Controlador

Es el encargado de asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a clases que representen, un sistema global o la fachada, algo en el mundo real que pueda ejecutar el papel (personas), o un manejador artificial que pueda ejecutar los eventos.

Este patrón se evidencia en la solución cuando las clases controladoras convierten un evento generado por el usuario (una petición HTTP por ejemplo) en una llamada a un método del modelo específico, convirtiéndose así en controladoras del flujo de eventos generados en el sistema. Un ejemplo de su uso puede ser encontrado en la clase InsumoController.

3.4.1.6 Patrón Singleton

El patrón Singleton es un patrón de diseño, específicamente de creación, es utilizado para garantizar que una clase sólo tenga una instancia y proporcione un punto de acceso global a ella. Su funcionamiento puede resumirse a que oculta el constructor de la clase Singleton, para que no se puedan crear otras instancias. Una vez más este patrón es reflejado en su totalidad mediante la inyección de dependencias utilizada en la solución propuesta.

3.5 Diseño de la Base de Datos

Siempre que se maneje grandes volúmenes de información o simplemente se desee conservar en el tiempo es necesario un método para almacenar y recuperar la información de manera organizada, para suplir esta necesidad surgen las bases de datos que no solo son capaces de almacenar elementos de distintos tipos sino que además tienen la capacidad de representar relaciones entre estos.

3.5.1 Modelo Entidad - Relación

A continuación se muestran las distintas entidades y sus relaciones, implementadas para cubrir las funcionalidades del sistema.

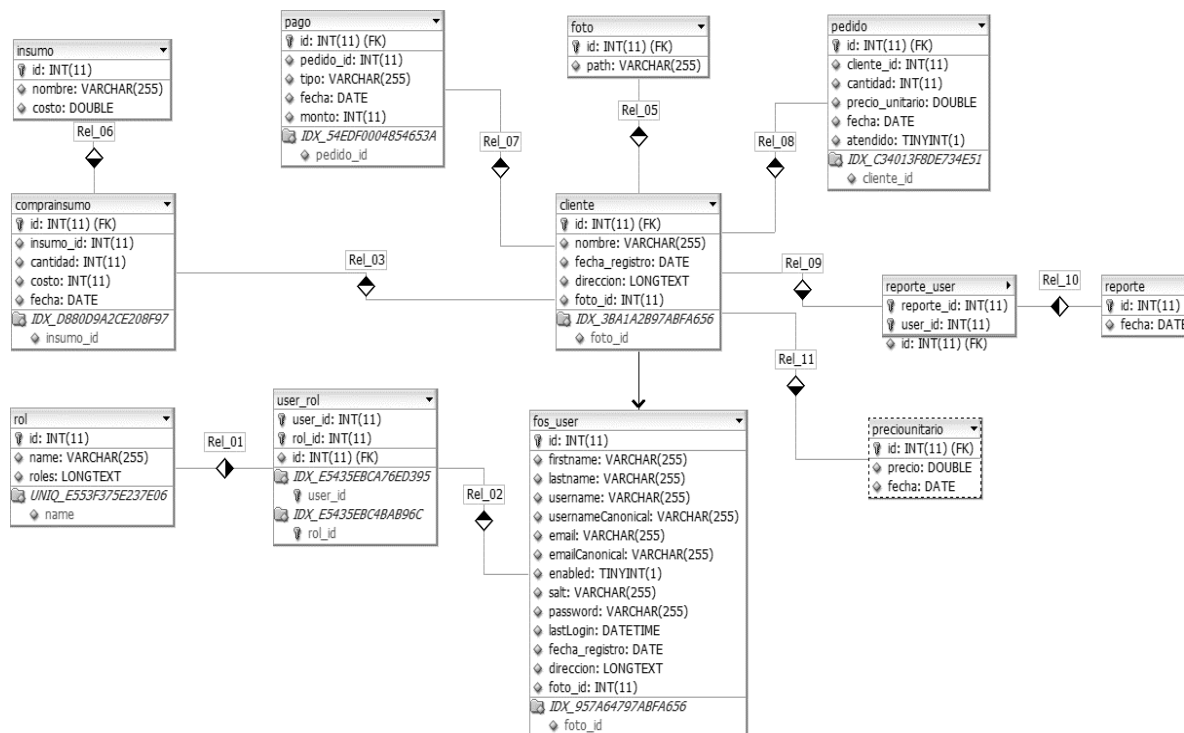


Figura 3.3 Modelo Entidad-Relación

3.6 Tarjetas Clase - Responsabilidad - Colaborador (CRC)

Como parte del diseño en la metodología XP el modelado de Clase-Responsabilidad-Colaborador (CRC) es un paso fundamental que permite organizar las clases más relevantes en el sistema. En este proceso se generan tarjetas representativas de las distintas clases y cuentan con la siguiente información:

- Nombre
- Responsabilidades
- Colaboraciones

Donde se define como responsabilidad lo que conoce o hace la clase. Las colaboraciones son aquellas en las que una clase le brinda la información necesaria a otra para completar una determinada responsabilidad. A continuación se presenta una muestra de las tarjetas identificadas en la implementación del sistema, el resto puede ser consultado en el Anexo 2 de este documento.



Tabla 3. 1 Tarjeta CRC de la clase pedidoController

Clase PedidoController	
Responsabilidad	Colaboraciones
Gestionar los pedidos.	PagoController, ClienteController
Graficar los pedidos	

Tabla 3. 2 Tarjeta CRC de la clase pagoController

Clase PagoController	
Responsabilidad	Colaboraciones
Gestionar los pagos.	PedidoController, ClienteController
Graficar los pagos.	

3.7 Conclusiones

Como parte de este capítulo se realizaron los análisis necesarios sobre el patrón arquitectónico y patrones de diseño utilizados para la implementación del sistema. Fueron analizadas además las necesidades existentes en cuanto a diseño de la Base de Datos para dar solución a los requisitos previamente establecidos y se definieron las distintas tarjetas CRC de las clases pertenecientes a la solución propuesta.

4 CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBA

4.1 Introducción

La dinámica de desarrollo de un software planteada por la metodología ágil XP establece la necesidad de iteraciones para diferenciar las distintas HU en lo que se refiere a su prioridad de implementación. De cada una de las iteraciones que se realicen se debe obtener como resultado un producto funcional, ateniéndose a las HU de usuario y sus limitaciones en cuanto al alcance con respecto al producto final, estos resultados deben ser probados para asegurar un correcto funcionamiento y deben ser presentados al cliente para cumplir con algo que es también una necesidad en la metodología XP, la comunicación y retroalimentación entre el equipo de desarrollo y el cliente. Este capítulo centra su atención en las iteraciones realizadas, las tareas de ingeniería y las pruebas de aceptación ejecutadas.

4.2 Fase de Implementación

En la ejecución de las iteraciones se lleva a cabo la implementación de las HU planificadas para cada iteración. Como parte de cada iteración se incluye la realización de una revisión del plan de iteraciones y se le realizan cambios a este de ser necesarios. Una vez realizados los análisis y cambios necesarios son descompuestas las HU en tareas de desarrollo que son asignadas según se determine a un equipo o persona encargado de su implementación. Debido a que estas tareas son responsabilidad únicamente del equipo de desarrollo pueden ser escritas en lenguaje técnico para una efectiva descripción de las distintas necesidades.

4.2.1 Iteraciones

En la implementación del sistema se siguió la planificación establecida y fueron realizadas dos iteraciones de desarrollo sobre el sistema y de esta forma se obtuvieron sendos productos para la aceptación del cliente, en el primer caso implementadas las HU de prioridad alta y en el segundo las restantes obteniéndose un producto final que da cumplimiento a las necesidades del cliente. A continuación son descritas las iteraciones.

4.2.1.1 Iteración 1

Las HU de prioridad alta fueron abordadas en esta iteración con el objetivo de entregar una primera versión del producto al cliente en la que fuesen solventadas las necesidades principales, de manera que de existir alguna inconformidad esta pudiese ser corregida efectivamente sin impactar negativamente en el tiempo de

entrega. A continuación se muestran las historias de usuario que se desean realizar en esta iteración, su tiempo de realización estimado y el real.

Tabla 4.1 Historias de Usuario abordadas en la primera iteración

Historia de Usuario	Estimación	Real
Registrar Cliente en el sistema	1	1
Registrar Pago en el sistema	1	1
Registrar Pedido en el sistema	1	1
Generar comprobantes e informes de Clientes	2	2
Generar comprobantes e informes de Pagos	2	2
Generar comprobantes e informes de Pedidos	2	2
Registrar Insumo en el sistema	1	1
Generar comprobantes e informes de Insumos	2	2
Gestionar Usuarios	1	1

4.2.1.1.1 Tareas de Ingeniería

Las siguientes tablas corresponden a algunas de las tareas de ingeniería correspondientes a las HU de prioridad alta abordadas en esta iteración, el resto pueden ser consultadas en el Anexo 3 del presente documento.

Tabla 4.2 Tarea #1 de la HU: Registrar Cliente en el Sistema

Tareas de Ingeniería



No. Tarea: 1	No. Historia de Usuario: 1
Nombre de la Tarea: Registrar Cliente en el Sistema	
Tipo de Tarea: Desarrollo	Puntos de estimación: 1
Fecha de Inicio: 11/11/2013	Fecha de Fin: 18/11/2013
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para registrar un cliente en el sistema, asignándole correctamente su rol y registrando la información necesaria.	

Tabla 4.3 Tarea #2 de la HU: Registrar Pago en el Sistema

Tareas de Ingeniería	
No. Tarea: 2	No. Historia de Usuario: 2
Nombre de la Tarea: Registrar Pago en el Sistema	
Tipo de Tarea: Desarrollo	Puntos de estimación: 1
Fecha de Inicio: 18/11/2013	Fecha de Fin: 25/11/2013
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para registrar un pago en el sistema.	

Tabla 4.4 Tarea #3 de la HU: Registrar Pedido en el Sistema

Tareas de Ingeniería	
No. Tarea: 3	No. Historia de Usuario: 3
Nombre de la Tarea: Registrar Pedido en el Sistema	
Tipo de Tarea: Desarrollo	Puntos de estimación: 1
Fecha de Inicio: 25/11/2013	Fecha de Fin: 2/12/2013
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para registrar un pedido en el sistema.	

Tabla 4.5 Tarea #4 de la HU: Generar comprobantes e informes de Clientes

Tareas de Ingeniería	
No. Tarea: 4	No. Historia de Usuario: 7
Nombre de la Tarea: Generar comprobantes e informes de Clientes	
Tipo de Tarea: Desarrollo	Puntos de estimación: 2
Fecha de Inicio: 2/12/2013	Fecha de Fin: 16/12/2013
Programador Responsable: Arletys Betancourt Alemán	

Roy Alberto Castro Aguilera

Descripción:

Se implementarán las funcionalidades necesarias para generar los comprobantes relacionados con uno o varios clientes en el sistema.

Tabla 4.6 Tarea #9 de la HU: Generar comprobantes e informes de Pedidos

Tareas de Ingeniería	
No. Tarea: 5	No. Historia de Usuario: 9
Nombre de la Tarea: Generar comprobantes e informes de Pedidos	
Tipo de Tarea: Desarrollo	Puntos de estimación: 2
Fecha de Inicio: 16/12/2013	Fecha de Fin: 30/12/2013
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para la generación de los comprobantes e informes necesarios, con este objetivo se debe de recoger la información concerniente a los pedidos involucrados.	

Tabla 4.7 Tarea #6 de la HU: Generar comprobantes e informes de Pagos

Tareas de Ingeniería	
No. Tarea: 6	No. Historia de Usuario: 8



Nombre de la Tarea: Generar comprobantes e informes de Pagos	
Tipo de Tarea: Desarrollo	Puntos de estimación: 2
Fecha de Inicio: 30/12/2013	Fecha de Fin: 13/1/2014
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para generar los comprobantes relacionados con uno o varios pagos en el sistema.	

Tabla 4.8 Tarea #7 de la HU: Gestionar Usuarios

Tareas de Ingeniería	
No. Tarea: 7	No. Historia de Usuario: 12
Nombre de la Tarea: Gestionar Usuarios	
Tipo de Tarea: Desarrollo	Puntos de estimación: 2
Fecha de Inicio: 13/1/2014	Fecha de Fin: 27/1/2014
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para gestionar la información relacionada con uno o varios usuarios en el sistema.	

4.2.1.2 Iteración 2

En la presente iteración se concluyó la implementación de las funcionalidades con prioridad media. A continuación se detallan las HU concernientes a esta iteración.

Tabla 4.9 Historias de Usuario abordadas en la segunda iteración

Historia de Usuario	Estimación	Real
Sistema de graficar Clientes	2	2
Sistema de graficar Pagos	1	1
Sistema de graficar Pedidos	1	1

4.2.1.2.1 4.2.1.2.1 Tareas de Ingeniería

Las siguientes tablas corresponden a algunas de las tareas de ingeniería correspondientes a las HU de prioridad media abordadas en esta iteración, el resto pueden ser consultadas en el Anexo 3 del presente documento.

Tabla 4.10 Tarea #1 de la HU: Graficar Pagos

Tareas de Ingeniería	
No. Tarea: 8	No. Historia de Usuario: 6
Nombre de la Tarea: Graficar Pagos	
Tipo de Tarea: Desarrollo	Puntos de estimación: 2
Fecha de Inicio: 11/03/2013	Fecha de Fin: 25/03/2014
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	

Descripción:

Se implementarán las funcionalidades necesarias para graficar la información relacionada con uno o varios pagos en el sistema.

Tabla 4.11 Tarea #1 de la HU: Graficar Clientes

Tareas de Ingeniería	
No. Tarea: 9	No. Historia de Usuario: 4
Nombre de la Tarea: Graficar Clientes	
Tipo de Tarea: Desarrollo	Puntos de estimación: 2
Fecha de Inicio: 25/03/2014	Fecha de Fin: 8/04/2014
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para graficar la información relacionada con uno o varios clientes en el sistema.	

Tabla 4.12 Tarea #1 de la HU: Graficar Pedidos

Tareas de Ingeniería	
No. Tarea: 10	No. Historia de Usuario: 5
Nombre de la Tarea: Graficar Pedidos	

Tipo de Tarea: Desarrollo	Puntos de estimación: 2
Fecha de Inicio: 8/04/2014	Fecha de Fin: 22/04/2014
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para graficar la información relacionada con uno o varios pedidos en el sistema.	

4.3 Diagrama de Despliegue

Este tipo de diagramas responde a la necesidad de representar la distribución física del sistema teniendo en cuenta como se distribuirán las funcionalidades entre los distintos nodos, que representan recursos de cómputo. Para lograr el funcionamiento de la aplicación en este caso es necesario contar con una computadora cliente y un servidor de aplicación como mínimo. No obstante pudiera seguirse otros tipos de arquitecturas como la separación física del servidor HTTP y el de base de datos como se presenta a continuación:

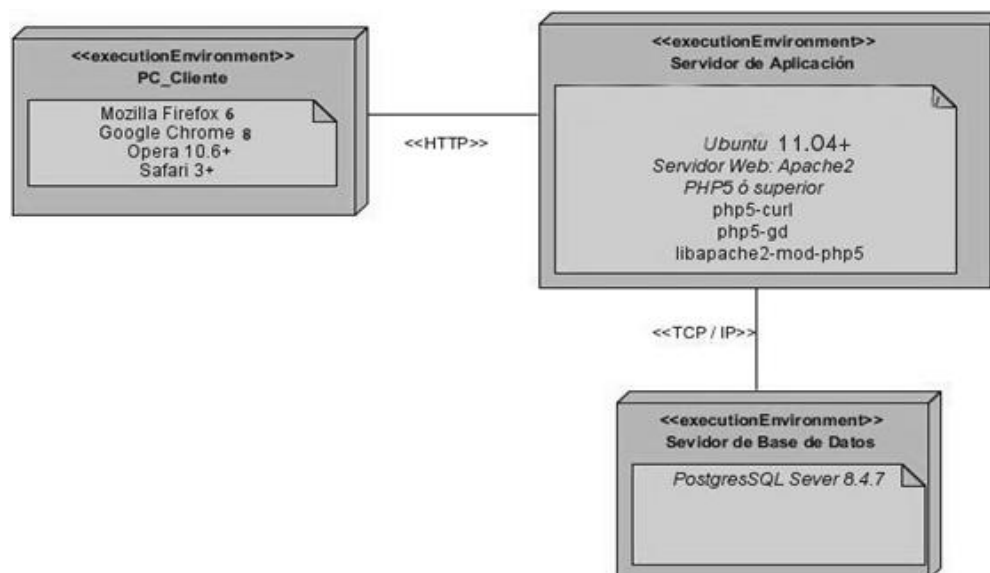


Figura 4.1 Diagrama de Despliegue

4.4 Pruebas

Durante el desarrollo de un software la etapa de pruebas es sin duda alguna un aspecto fundamental a tener en cuenta, pues de su correcta realización depende la entrega de un producto que responde realmente a las necesidades del cliente. En esta etapa el código es examinado exhaustivamente con el objetivo de depurarlo de cualquier error que haya pasado inadvertido durante la etapa de desarrollo.

La metodología XP divide las pruebas en dos grupos: las pruebas unitarias que son implementadas por los programadores y son capaces de ejecutarse de forma automática y de esta forma verificar el código y las pruebas de aceptación que están orientadas a evaluar en el final de cada iteración si se consiguió dar solución a la necesidad del cliente en cada funcionalidad implementada.

4.4.1 Pruebas Unitarias

En la metodología XP se prioriza la agilidad del desarrollo por encima de la generación de documentos no obstante la calidad del producto final no deja de ser el principal objetivo, y en este aspecto incide profundamente el hecho de que se diseñen y realicen pruebas unitarias capaces de verificar la efectividad del código implementado en la solución de las necesidades del cliente.

Las pruebas unitarias son empleadas durante el desarrollo del producto en todas las funcionalidades implementadas y esto permite retroalimentación constante para el programador sobre cómo está realizando su trabajo. En Symfony las pruebas unitarias son soportadas nativamente por el framework que ofrece muchísimas facilidades para su implementación y desarrollo.

4.4.2 Pruebas de Aceptación

En el caso de las pruebas de aceptación es necesario aclarar que se diferencian de las pruebas unitarias en que no son basadas en la evaluación de una funcionalidad en específico sino que se diseñan a partir de las HU. De esta forma se evalúa la implementación de las distintas HU y se le da al cliente la posibilidad de evidenciar el funcionamiento del sistema en un entorno real de funcionamiento, donde se evidencien la estabilidad del sistema, su rendimiento, etc. Es necesario que una HU pase todas las pruebas de aceptación diseñadas para ella antes de que pueda declararse como terminada.

Las pruebas de aceptación correspondiente a cada una de las funcionalidades serán representadas mediante tablas divididas por las secciones siguientes:

- Código de la prueba de aceptación.

- Número de la historia de usuario a la que se le realiza la prueba.
- Nombre de la funcionalidad.
- Descripción de la funcionalidad.
- Condiciones de ejecución de la funcionalidad.
- Entrada y pasos de ejecución que realiza el usuario con el objetivo de obtener el resultado esperado.
- Resultado esperado.
- Evaluación de la prueba.

Tabla 4.13 Prueba de aceptación HU1-P1

Caso de prueba de aceptación	
Código: HU1-P1	No. Historia de Usuario: 1
Nombre de la Tarea: Registrar Cliente en el sistema	
Descripción: Prueba la HU Registrar Cliente en el sistema	
Condiciones de ejecución: El usuario debe estar autenticado y con permisos de registrar clientes.	
Entrada/Pasos de ejecución: Toda la información del cliente a registrar. Una vez entrada la información del cliente en el sistema se procede a crear una entidad que a su vez será persistida en la base de datos.	
Resultado esperado: Cliente registrado correctamente.	
Evaluación: Prueba Satisfactoria	

Tabla 4.14 Prueba de aceptación HU2-P1

Caso de prueba de aceptación

Código: HU2-P1	No. Historia de Usuario: 2
Nombre de la Tarea: Registrar Pedido en el sistema	
Descripción: Prueba la HU Registrar Pedido en el sistema	
Condiciones de ejecución: El usuario debe estar logueado y con permisos de registrar pedidos.	
Entrada/Pasos de ejecución: Toda la información del pedido a registrar. Una vez entrada la información del pedido en el sistema se procede a crear una entidad que a su vez será persistida en la base de datos.	
Resultado esperado: Pedido registrado correctamente.	
Evaluación: Prueba Satisfactoria	

Tabla 4.15 Prueba de aceptación HU3-P1

Caso de prueba de aceptación	
Código: HU3-P1	No. Historia de Usuario: 3
Nombre de la Tarea: Registrar Pago en el sistema	
Descripción: Prueba la HU Registrar Pago en el sistema	
Condiciones de ejecución: El usuario debe estar logueado y con permisos de registrar pagos.	
Entrada/Pasos de ejecución: Toda la información del pago a registrar. Una vez entrada la información del pago en el sistema se procede a crear una entidad que a su vez será persistida en la base de datos.	

Resultado esperado: Pago registrado correctamente.

Evaluación: Prueba Satisfactoria

Tabla 4.16 Prueba de aceptación HU4-P1

Caso de prueba de aceptación	
Código: HU4-P1	No. Historia de Usuario: 4
Nombre de la Tarea: Graficar clientes	
Descripción: Prueba la HU Graficar clientes	
Condiciones de ejecución: El usuario debe estar logueado y con permisos de graficar clientes.	
Entrada/Pasos de ejecución: Una vez seleccionados los clientes se procede a graficar la información relacionada con estos.	
Resultado esperado: Gráfico generado correctamente.	
Evaluación: Prueba Satisfactoria	

Tabla 4.17 Prueba de aceptación HU5-P1

Caso de prueba de aceptación	
Código: HU5-P1	No. Historia de Usuario: 5
Nombre de la Tarea: Graficar pedidos	



Descripción: Prueba la HU Graficar pedidos
Condiciones de ejecución: El usuario debe estar logueado y con permisos de graficar pedidos.
Entrada/Pasos de ejecución: Una vez seleccionados los pedidos se procede a graficar la información relacionada con estos.
Resultado esperado: Gráfico generado correctamente.
Evaluación: Prueba Satisfactoria

Tabla 4.18 Prueba de aceptación HU6-P1

Caso de prueba de aceptación	
Código: HU6-P1	No. Historia de Usuario: 6
Nombre de la Tarea: Graficar pagos	
Descripción: Prueba la HU Graficar pagos	
Condiciones de ejecución: El usuario debe estar logueado y con permisos de graficar pagos.	
Entrada/Pasos de ejecución: Una vez seleccionados los pagos se procede a graficar la información relacionada con estos.	
Resultado esperado: Gráfico generado correctamente.	
Evaluación: Prueba Satisfactoria	

Tabla 4.19 Prueba de aceptación HU7-P1

Caso de prueba de aceptación	
Código: HU7-P1	No. Historia de Usuario: 7
Nombre de la Tarea: Generar comprobantes e informes de Clientes	
Descripción: Prueba la HU Generar comprobantes e informes de Clientes	
Condiciones de ejecución: El usuario debe estar autenticado y con los permisos requeridos para generar los informes y comprobantes de los clientes.	
Entrada/Pasos de ejecución: Una vez seleccionados los clientes se procede a generar la información relacionada con estos.	
Resultado esperado: Reporte o comprobante generado correctamente.	
Evaluación: Prueba Satisfactoria	

Tabla 4.20 Prueba de aceptación HU8-P1

Caso de prueba de aceptación	
Código: HU8-P1	No. Historia de Usuario: 8
Nombre de la Tarea: Generar comprobantes e informes de Pagos	
Descripción: Prueba la HU Generar comprobantes e informes de Pagos	

<p>Condiciones de ejecución:</p> <p>El usuario debe estar autenticado y con los permisos requeridos para generar los informes y comprobantes de los pagos.</p>
<p>Entrada/Pasos de ejecución:</p> <p>Una vez seleccionados los pagos se procede a generar la información relacionada con estos.</p>
<p>Resultado esperado: Reporte o comprobante generado correctamente.</p>
<p>Evaluación: Prueba Satisfactoria</p>

Tabla 4.21 Prueba de aceptación HU10-P1

Caso de prueba de aceptación	
Código: HU10-P1	No. Historia de Usuario: 10
Nombre de la Tarea: Registrar Insumo en el sistema	
Descripción: Prueba la HU Registrar Insumo en el sistema	
Condiciones de ejecución: El usuario debe estar autenticado y con los permisos requeridos para ingresar en el sistema un nuevo tipo de insumo.	
Entrada/Pasos de ejecución: Una vez introducidos en el sistema todos los datos necesarios para registrar un nuevo insumo se procede a generar una entidad que luego es persistida en la base de datos.	
Resultado esperado: Insumo registrado correctamente.	
Evaluación: Prueba Satisfactoria	

Tabla 4.22 Prueba de aceptación HU9-P1

Caso de prueba de aceptación	
Código: HU9-P1	No. Historia de Usuario: 9
Nombre de la Tarea: Generar comprobantes e informes de Pedidos	
Descripción: Prueba la HU Generar comprobantes e informes de Pedidos	
Condiciones de ejecución: El usuario debe estar autenticado y con los permisos requeridos para generar los informes y comprobantes de los pedidos.	
Entrada/Pasos de ejecución: Una vez seleccionados los pagos se procede a generar la información relacionada con estos.	
Resultado esperado: Reporte o comprobante generado correctamente.	
Evaluación: Prueba Satisfactoria	

Tabla 4.23 Prueba de aceptación HU12-P1

Caso de prueba de aceptación	
Código: HU12-P1	No. Historia de Usuario: 12
Nombre de la Tarea: Gestión de Usuarios	
Descripción: Prueba la HU Gestión de Usuarios	
Condiciones de ejecución:	



El usuario debe estar autenticado y con los permisos requeridos para crear nuevos usuarios en el sistema. Estos usuarios son necesariamente administradores.

Entrada/Pasos de ejecución:

Una vez introducidos en el sistema toda la información relacionada al nuevo usuario se procede a crear una entidad de este tipo y persistirla en la base de datos.

Resultado esperado: Usuario generado correctamente.

Evaluación: Prueba Satisfactoria

Luego de realizarse las pruebas de aceptación se obtuvieron un total de seis no conformidades que fueron resueltas satisfactoriamente. Ilustrando estos resultados se presenta la siguiente gráfica:

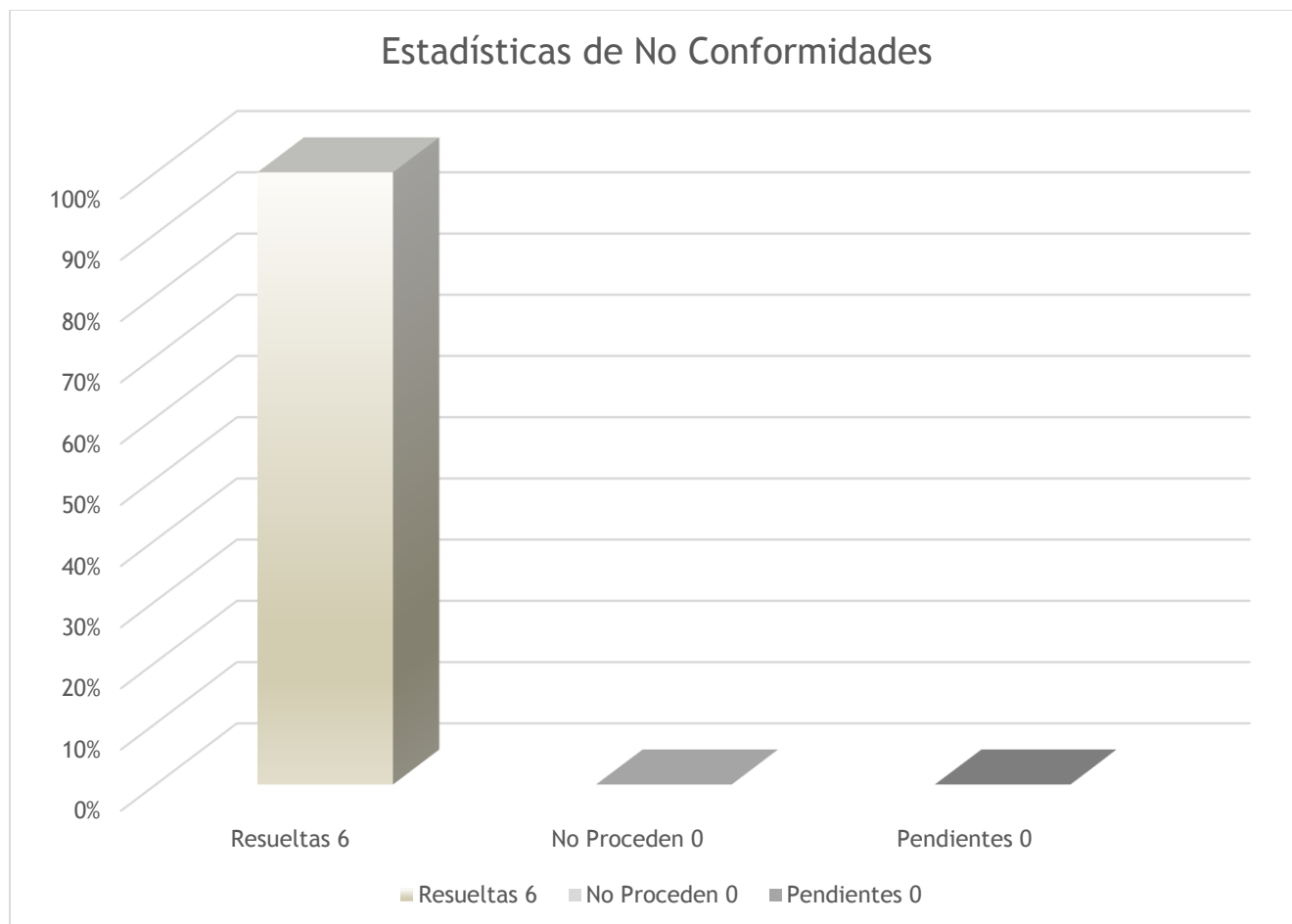


Figura 4.2 Estadísticas de pruebas de no conformidad

En una segunda iteración se ejecutaron nuevamente los casos de prueba sin encontrarse ninguna no conformidad.

4.5 Conclusiones

En el transcurso de este capítulo fueron implementadas las funcionalidades de acuerdo a las HU abordadas y definidas en capítulos anteriores, además las pruebas de aceptación fueron un objetivo cumplido pues se le realizaron distintas pruebas de este tipo al producto obtenido que dieron resultados satisfactorios en cuanto a la solución de problemas.

5 CONCLUSIONES GENERALES

Con la investigación realizada se logró cumplir el objetivo general, desarrollando un sistema capaz de mejorar el control de los procesos de la imprenta Sagrado Corazón. Teniendo en cuenta lo anterior se puede plantear que:

- El estudio de los distintos sistemas de gestión existentes evidenció la carencia de adaptabilidad de estos para un entorno de trabajo en el que no es posible una preparación profunda del personal en temas económicos.
- El uso de XP como metodología de desarrollo además de las tecnologías y herramientas utilizadas permitieron la implementación del Sistema SGISC y de esta forma mejorar el control de los procesos que se desarrollan en la entidad.
- La realización de las pruebas de aceptación permitieron validar la propuesta del sistema.

6 RECOMENDACIONES

- Realizar un levantamiento de información en otras imprentas para validar la posibilidad de la utilización o adaptación del sistema.
- Agregar soporte a pedidos de tipo back order en aras de permitir el registro y posterior seguimiento de estos.

7 REFERENCIAS BIBLIOGRÁFICAS

1. **RAE.** Diccionario de la Lengua Española – Vigésima Segunda Edición. [En línea] RAE. [Citado el: 10 de 2 de 2014.] <http://buscon.rae.es/drae/>.
2. **php.net.** www.php.net. [En línea] The PHP Group, 2001. [Citado el: 25 de 01 de 2014.] <http://es.php.net/manual/es/preface.php>.
3. **w3c.** www.w3c.org. [En línea] w3c. [Citado el: 25 de 01 de 2014.] <http://www.w3.org/TR/html4/intro/intro.html>.
4. **supplychain-software.** <http://www.supplychain-software.com/>. [En línea] supplychain-software, 2008. [Citado el: 01 de 02 de 2014.] <http://www.supplychain-software.com/index.html>.
5. **ecosoftconsulting.** <http://www.ecosoftconsulting.net/index.aspx>. [En línea] ecosoftconsulting, 2013. [Citado el: 01 de 02 de 2014.] http://www.ecosoftconsulting.net/quienes_somos.aspx.
6. **Sage.** <http://www.softwaregestionempresarial.com/>. [En línea] Sage. [Citado el: 02 de 02 de 2014.] <http://www.softwaregestionempresarial.com/>.
7. **Odoo.** About Us.: Odoo. *Odoo, sitio web*. [En línea] Odoo. [Citado el: 18 de 05 de 2014.] <https://www.odoo.com/page/about-us>.
8. **Quispe Carita, Vilma, Huamantuco Solorzano, Dante Harry y Vargas Yupanqui, José Luis.** *METODOLOGIA RUP (RATIONAL UNIFIED PROCESS)*. Lima : s.n., 2011.
9. **Carmen Penadés, María, H. Canós, José y Letelier.** *Métodologías Ágiles en el desarrollo de software*. Madrid : Universidad Politécnica de Valencia, 2004.
10. **Rumbaugh, James, Booch, Grady y Jacobson, Ivan.** *El Proceso Unificado de Desarrollo de . s.l. :* Pearson Addison-Wesley, 2000.
11. **<http://www.doctrine-project.org/>.** <http://www.doctrine-project.org/>. [En línea] doctrine-project.org. [Citado el: 25 de 01 de 2014.] <http://www.doctrine-project.org/blog/index.html>.
12. **sencha.com.** <http://www.sencha.com>. [En línea] sencha.com. [Citado el: 27 de 01 de 2014.] <http://www.sencha.com/products/extjs/>.

13. **trucosdeprogramacionmovil.** <http://trucosdeprogramacionmovil.blogspot.com>. [En línea] trucosdeprogramacionmovil, 10 de 10 de 2012. [Citado el: 12 de 04 de 2014.] <http://trucosdeprogramacionmovil.blogspot.com/2012/11/yii-framework-el-lado-oscuro-de-la-luna.html>.
14. **<http://symfony.es>.** <http://www.symfony.es/que-es-symfony/>. *Symfony*. [En línea] Symfony. [Citado el: 15 de 5 de 2014.] <http://www.symfony.es/que-es-symfony/>.
15. **<http://jquery.com/>.** <http://jquery.com/>. [En línea] <http://jquery.com/>. [Citado el: 27 de 01 de 2014.] <http://jquery.com/>.
16. **<http://dojotoolkit.org/>.** <http://dojotoolkit.org/>. [En línea] <http://dojotoolkit.org/>. [Citado el: 27 de 01 de 2014.] <http://dojotoolkit.org/>.
17. **ellislab.com.** ellislab.com. [En línea] ellislab, 2002. [Citado el: 26 de 01 de 2014.] <http://ellislab.com/codeigniter>.
18. **apache.org.** <http://httpd.apache.org>. [En línea] [apache.org](http://httpd.apache.org). [Citado el: 27 de 01 de 2014.] http://httpd.apache.org/ABOUT_APACHE.html.
19. **César de la Torre, Zorrilla, Unai y Ramos, Miguel Ángel.** *Guía de Arquitectura N-Capas*. s.l. : Krassis Press, 2011.
20. **<http://symfony.es/>.** symfony.es/. [En línea] <http://symfony.es/>. [Citado el: 26 de 01 de 2014.] <http://symfony.es/que-es-symfony>.
21. **Eguiluz, Javier.** *Desarrollo Web Ágil con SYMFONY 2.3*. 2013.



8 ANEXOS

8.1 Anexo 1

HISTORIA DE USUARIO	
Número: 5	Nombre de HU: Graficar Pedidos
Usuario: Arletys Betancourt, Roy Castro	Iteración Asignada: 2
Prioridad en negocio: Media	Puntos estimados: 1
Riesgo en Desarrollo: Alto	Puntos Reales: 1
Descripción: Esta funcionalidad tiene como objetivo la implementación de la parte de la aplicación encargada de graficar la información relacionada con los pedidos registrados en el sistema.	
Observaciones: Debe recogerse la información necesaria para seleccionar los pedidos correctamente, siendo de carácter obligatorio seleccionar el espacio de tiempo que se desea analizar.	



Prototipo de Interfaz:



Anexo 1 HU Graficar pedidos

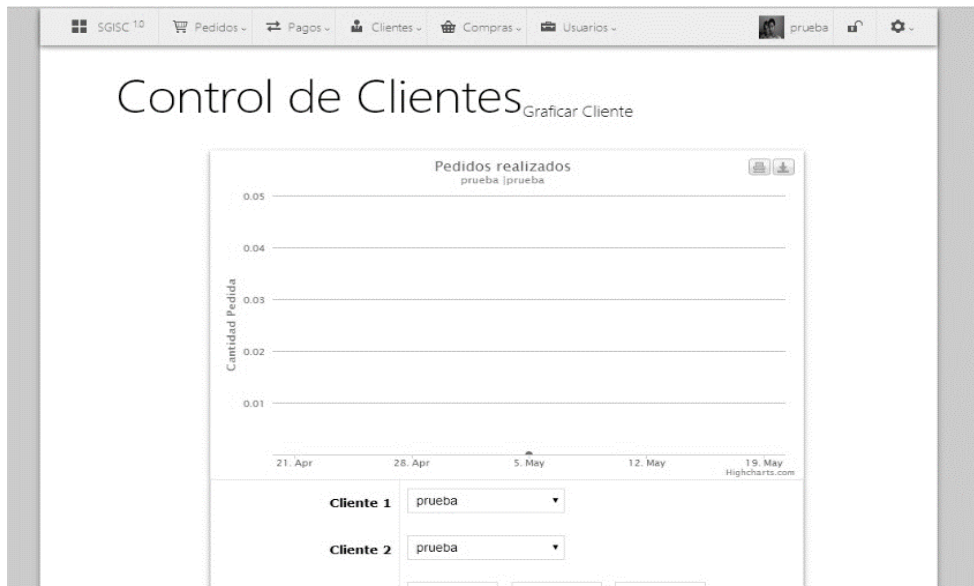
HISTORIA DE USUARIO	
Número: 6	Nombre de HU: Graficar Pagos
Usuario: Arletys Betancourt, Roy Castro	Iteración Asignada: 2
Prioridad en negocio: Media	Puntos estimados: 1
Riesgo en Desarrollo: Alto	Puntos Reales: 1
Descripción: Esta funcionalidad tiene como objetivo la implementación de la parte de la aplicación encargada de graficar la información relacionada con los pagos registrados en el sistema.	



Observaciones:

Debe recogerse la información necesaria para seleccionar los pagos correctamente, siendo de carácter obligatorio seleccionar el espacio de tiempo que se desea analizar.

Prototipo de Interfaz:



Anexo 1 HU Graficar pagos

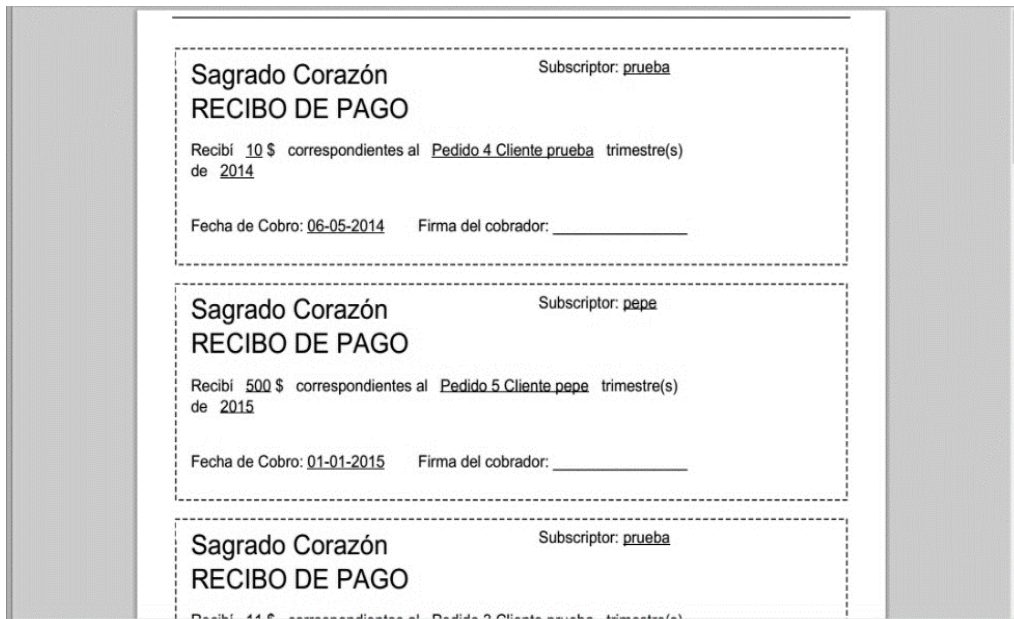
HISTORIA DE USUARIO	
Número: 7	Nombre de HU: Generar Comprobantes e Informes de Clientes
Usuario: Arletys Betancourt, Roy Castro	Iteración Asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 2
Riesgo en Desarrollo: Alto	Puntos Reales: 2
Descripción: Esta funcionalidad tiene como meta la generación de los comprobantes e informes relacionados con los distintos clientes, para ello debe seleccionarse siempre el cliente y este debe de tener los permisos adecuados.	



Observaciones:

Este sistema tiene como principal característica la generación de documentos pdf.

Prototipo de Interfaz:



Anexo 1 HU Generar Comprobantes e Informes de Clientes

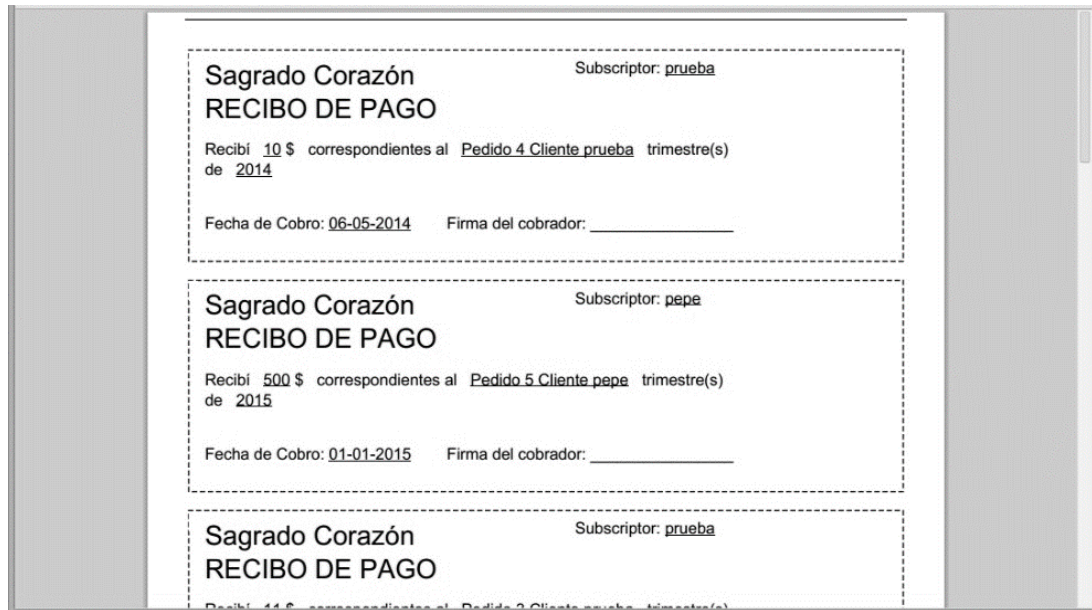
HISTORIA DE USUARIO	
Número: 8	Nombre de HU: Generar Comprobantes e Informes de Pagos
Usuario: Arletys Betancourt, Roy Castro	Iteración Asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 2
Riesgo en Desarrollo: Alto	Puntos Reales: 2
Descripción: Esta funcionalidad tiene como meta la generación de los comprobantes e informes relacionados con los distintos pagos, para ello debe seleccionarse siempre el cliente y este debe de tener los permisos adecuados.	



Observaciones:

Este sistema tiene como principal característica la generación de documentos pdf.

Prototipo de Interfaz:



Anexo 1 HU Generación de Comprobantes e Informes de Pagos

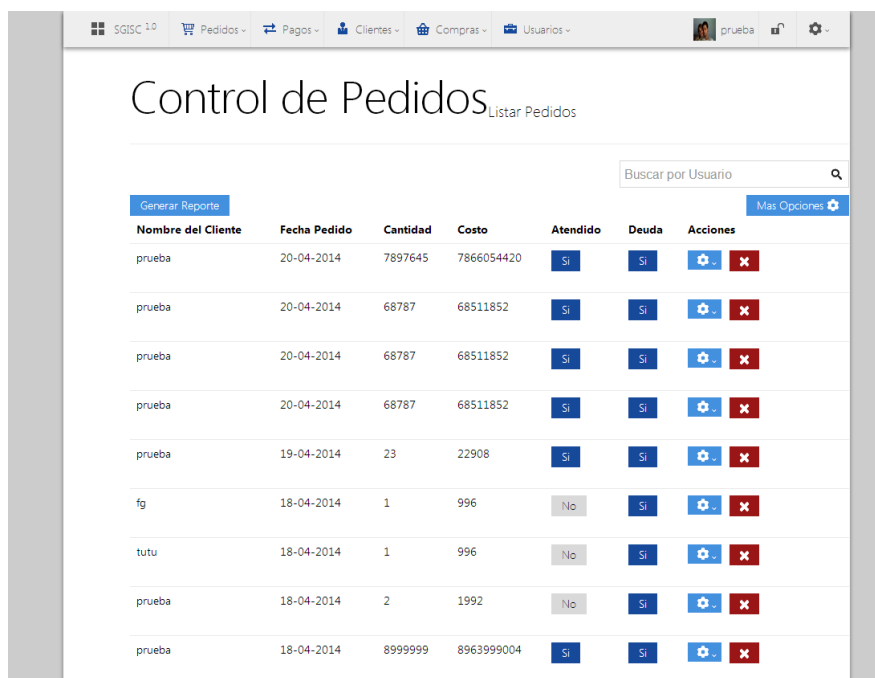
HISTORIA DE USUARIO	
Número: 9	Nombre de HU: Generar Comprobantes e Informes de Pedidos
Usuario: Arletys Betancourt, Roy Castro	Iteración Asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 2
Riesgo en Desarrollo: Alto	Puntos Reales: 2
Descripción: Esta funcionalidad tiene como meta la generación de los comprobantes e informes relacionados con los distintos pedidos realizados por los clientes, para ello debe seleccionarse siempre el cliente y este debe de tener los permisos adecuados.	



Observaciones:

Este sistema tiene como principal característica la generación de documentos pdf.

Prototipo de Interfaz:



Anexo 1 HU Generar Comprobantes e Informes de Pedidos

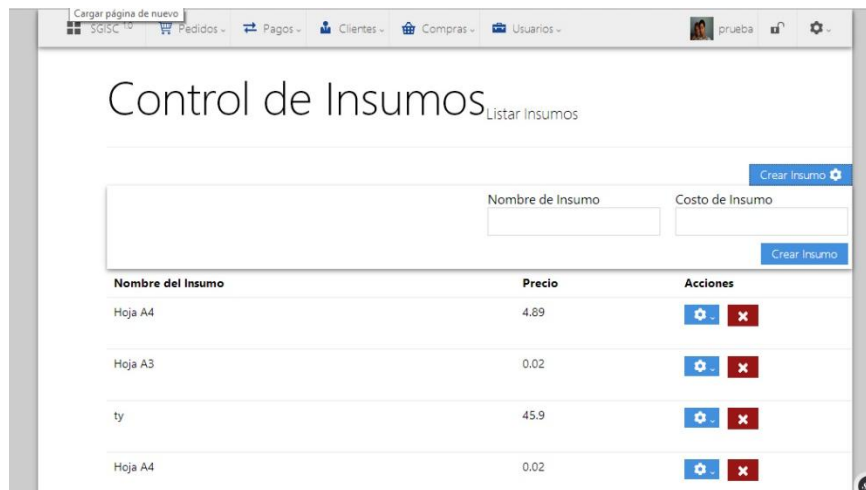
HISTORIA DE USUARIO	
Número: 10	Nombre de HU: Registrar Insumo en el Sistema
Usuario: Arletys Betancourt, Roy Castro	Iteración Asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 1
Riesgo en Desarrollo: Alto	Puntos Reales: 1



Descripción: Esta funcionalidad tiene como meta el registro de nuevos tipos de insumos y de nuevas compras de insumos en el sistema.

Observaciones:
Para registrar un nuevo tipo de insumo se debe contar con privilegios de administración.

Prototipo de Interfaz:



Anexo 1 Tabla 1 Registrar Insumo en el Sistema

HISTORIA DE USUARIO	
Número: 11	Nombre de HU: Generar Comprobantes e Informes de Insumos
Usuario: Arletys Betancourt, Roy Castro	Iteración Asignada: 1
Prioridad en negocio: Alta	Puntos estimados: 2
Riesgo en Desarrollo: Alto	Puntos Reales: 2



Descripción: Esta funcionalidad tiene como meta la generación de los comprobantes e informes relacionados con los distintos insumos registrados en el sistema o las compras relacionadas con estos.

Observaciones:
Los clientes no pueden acceder a los distintos insumos ni a las compras de estos.

Prototipo de Interfaz:



Anexo 1 Generar Comprobantes e Informes de Insumos



8.2 Anexo 2

Clase ClienteController	
Responsabilidad	Colaboraciones
Gestionar los clientes.	PedidoController, PagoController , InsumoController, CompraInsumoController
Graficar datos relacionados con los clientes.	

Anexo 2 Tabla 1 Tarjeta CRC Clase ClienteController

Clase InsumoController	
Responsabilidad	Colaboraciones
Gestionar los insumos.	ClienteController , CompraInsumoController
Graficar datos relacionados con los insumos.	



Anexo 2 Tabla 2 Tarjeta CRC Clase InsumoController

Clase PrecioUnitarioController	
Responsabilidad	Colaboraciones
Gestionar los precios.	ClienteController

Anexo 2 Tabla 3 Tarjeta CRC Clase PrecioUnitarioController

Clase CompralInsumoController	
Responsabilidad	Colaboraciones
Gestionar las compras de insumos.	ClienteController, InsumoController
Graficar datos relacionados con las compras de insumos.	

Anexo 2 Tabla 4 Tarjeta CRC Clase CompralInsumoController

8.3 Anexo 3

Tareas de Ingeniería	
No. Tarea: 11	No. Historia de Usuario: 10
Nombre de la Tarea: Registrar Insumo en el sistema	
Tipo de Tarea: Desarrollo	Puntos de estimación: 1
Fecha de Inicio: 27/1/2014	Fecha de Fin: 3/2/2014
Programador Responsable: Arletys Betancourt Alemán Roy Alberto Castro Aguilera	
Descripción: Se implementarán las funcionalidades necesarias para introducir nuevos insumos en el sistema y sus compras.	

Anexo 3 Tabla 1 Tarea #11 de la HU Registrar Insumo

Tareas de Ingeniería	
No. Tarea: 12	No. Historia de Usuario: 11
Nombre de la Tarea: Sistema generación de comprobantes e informes de Insumos	
Tipo de Tarea: Desarrollo	Puntos de estimación: 2
Fecha de Inicio: 3/2/2014	Fecha de Fin: 17/2/2014
Programador Responsable: Arletys Betancourt Alemán	



Roy Alberto Castro Aguilera

Descripción:

Se implementarán las funcionalidades necesarias para generar los comprobantes e informes según corresponda en uno u otro caso. Debe de haberse seleccionado previamente el o los insumos involucrados.

Anexo 3 Tabla 2 Tarea #12 de la HU Sistema de generación de comprobantes e informes de insumos