**Universidad de las Ciencias Informáticas**

**Facultad 4**

# Sistema informático para la gestión de la información de los Activos Fijos Tangibles, los Útiles y Herramientas de la residencia de la Universidad de las Ciencias Informáticas

**Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas**

**Autores:**

**Yandriel Padrón Mojena**

**Luis Daniel González Díaz**

**Tutores:**

**Ing. Yusdel Meriño Almaguer**

**Ing. Roanny Lamas López**

**Ciudad de La Habana, junio de 2016**

**"Año 58 de la Revolución"**

**Declaración de autoría**

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) a que haga el uso que estime pertinente con el mismo.

Para que así conste firmamos la presente a los _____ días del mes de _____ del año _____.

Autores:

_____ _____

Yandriel Padrón Mojena Luis Daniel González Díaz

Tutores:

_____ _____

Ing. Yusdel Meriño Almaguer Ing. Roanny Lamas López

**Dedicatorias**

*De Yandriel:*

*Es impresionante la fuerza de la fe, solo con confiar en alguien sin necesidad de probar su veracidad se puede romper toda barrera y dificultad. Gracias por creer en mí,*

*Papá y Mamá*

*De Luis:*

*Gracias por tu apoyo y compresión,*

*Mamá*

*De Yandriel:*

*Agradezco a mis padres Jose Ricardo y Milenis por ceer en mí en todo momento.*

*Agradezco a mi esposa Danae por estar junto a mí y vivir todo mi sufrimiento aliviándolo con amor.*

*Agradezco a mi hermano Yasel por ser tan atento con mis avances en la Universidad*

*Agradezco a toda mi familia en general porque a pesar de estar lejos siempre se preocuparon por mí*

*Agradezco a todos mis amigos que estuvieron cerca y me aconsejaron.*

*Agradezco a todos los profesores y graduados que me ayudaron con la investigación*

*De Luis:*

*Agradezco a mi mamá por todo lo que ha hecho por mí*

*Agradezco a mi hija Valeria por darme fuerzas para continuar*

*Agradezco a Lisandra por creer en mí*

*Agradezco a mis abuelos por confiar en que me graduara*

## Resumen

La gestión de la información de los medios básicos en la Residencia de la Universidad de las Ciencias Informáticas se realiza de forma manual, ello trae consigo que este proceso de gestión sea complejo, debido a la gran cantidad de información sobre esos medios que se maneja en el área de la administración. El presente trabajo consiste en el desarrollo de un sistema multiplataforma concebido con tecnologías libres para facilitar dicho proceso. Este sistema fue guiado por la metodología Extreming Programing. Como tecnologías del lado del cliente se utilizaron: HTML, JavaScript, a través de la librería JQuery, y CSS, apoyado en el framework de diseño Bootstrap. El lenguaje del lado del servidor empleado fue PHP, a través del framework Symfony. Se utilizó el NetBeans IDE como Entorno de Desarrollo Integrado, Apache como servidor web y PostgreSQL para la gestión de bases de datos. Se implementaron funcionalidades en la tramitación de Activos Fijos Tangibles, Útiles, Herramientas, Entidades, Áreas, Usuarios, Cargos, Personas, Movimientos de un medio básico. También se implementaron funcionalidades para importar archivos y reportes, tales como: mostrar listado de movimientos realizados, modelo de un medio y trazas del medio. Fueron realizadas 62 pruebas de software a la propuesta de solución, las cuales ofrecen niveles de validación de las funcionalidades y la satisfacción por parte del cliente.

## Índice

# Introducción

Las grandes transformaciones económicas y sociales de los últimos tiempos han conformado un nuevo modelo socio-técnico, impulsado principalmente por un sorprendente desarrollo tecnológico. Estos eventos han provocado un obligatorio cambio en el propio contenido de la organización de la información en la gestión empresarial y su control.

El control es un plan donde se preveen todas las medidas administrativas dentro de la entidad, para el logro de los objetivos; de ahí que el control interno sea un plan de organización donde se establecen las políticas y procedimientos que persigue la entidad con el fin de salvaguardar los recursos con que cuenta. La efectividad del control interno depende en gran medida de la integridad y de los valores éticos del personal que diseña, administra y vigila este proceso en la entidad.

Es de suma importancia controlar ya que es la única manera de proteger los recursos contra pérdida, fraude o ineficiencia. Además de medir el cumplimiento de las tareas cerciorándose si ha habido adhesión a las políticas generales de la organización. De igual forma promover la exactitud o confiabilidad de los informes contables y administrativos. Es necesario establecer mecanismos de control para verificar la existencia real de los datos que se manejan y que dichos datos se puedan verificar o cotejar con el aspecto contable en cualquier momento o periodo con la satisfacción de que todo está en orden y al día.

La Universidad de las Ciencias Informáticas (UCI), posee una Residencia Universitaria con más de 100 edificios, la residencia está divida en tres Direcciones, la 1, la 2 y la de Profesores y Especialistas. La Dirección de Residencia aloja a estudiantes cubanos y extranjeros, profesores, especialistas, además de Recién graduados de la propia Universidad. Dentro de los procesos que se llevan a cabo en esta Residencia, está el control de los Activos Fijos Tangibles (AFT), los Útiles y las Herramientas, para lo cual existe un aparato administrativo, dirigido por un Director, un Especialista Superior y cuatro Técnicos Generales.

El área administrativa de la Residencia se dedica fundamentalmente al control del inventario de los AFT, los Útiles y las Herramientas y aunque esto parezca sencillo y práctico es de gran dificultad y tiempo de trabajo para las personas que trabajan con dichos medios. Existe un

sistema en la Dirección de Economía de la UCI, llamado Assets Premium, al cual la Dirección de Residencia no tiene acceso.

En la Residencia Universitaria, existe un número elevado de AFT, con más de 30 mil medios, entre AFT, Herramientas y Útiles lo que trae como consecuencia que el proceso de búsqueda de las características y propiedades de los medios sea complejo y tedioso por lo que actualmente en esta área el control de estos medios se realiza de forma manual, con el uso de hojas de cálculo y el proceso de actualización cuando se realizan movimientos, se torna lento y engorroso. Además, debido a los constantes movimientos de medios en la residencia, resulta difícil llevar un control de la información actualizada referida a las existencias de AFT, Útiles y Herramientas. Estas características traen como consecuencia poca protección contra pérdida de información, lo que trae consigo situaciones tales como: el fraude o la ineficiencia laboral del equipo de trabajo.

A partir de la situación problemática anteriormente planteada se define para la presente investigación el siguiente **problema**: ¿Cómo contribuir al control de los Activos Fijos Tangibles, de los Útiles y Herramientas de la Residencia, de la Universidad de las Ciencias Informáticas?

Por lo que se define como **objetivo general**: Desarrollar un sistema informático para la gestión de la información de los Activos Fijos Tangibles, los Útiles y Herramientas de la Residencia de la Universidad de las Ciencias Informáticas.

Para dar cumplimento a las preguntas científicas se trazaron los siguientes **objetivos específicos**:

1. Estudiar los referentes teóricos relacionados con la gestión de la información, en general y de los medios básicos en las organizaciones, en particular
2. Diagnosticar el proceso de gestión de la información de los Activos Fijos Tangibles, Útiles y Herramientas en la Residencia de la Universidad de las Ciencias Informáticas
3. Caracterizar el proceso de gestión de la información de los Activos Fijos Tangibles, Útiles y Herramientas en la Residencia de la Universidad de las Ciencias Informáticas

4. Diseñar un sistema informático como respuesta al problema relacionado con la gestión de la información de los Activos Fijos Tangibles, los Útiles y Herramientas de la Residencia de la Universidad de las Ciencias Informáticas como propuesta de solución
5. Implementar el sistema informático como propuesta de solución
6. Validar las pruebas de software de la propuesta de solución

Este problema enmarca como **objeto de estudio** los procesos para la gestión de la información de medios básicos en una organización.

Delimitando como **campo de acción** el proceso de gestión de la información de los Activos Fijos Tangibles, Útiles y Herramientas en la Residencia de la Universidad de las Ciencias Informáticas.

En correspondencia con lo planteado anteriormente se formulan las siguientes **preguntas de investigación:**

1. ¿Qué fundamentos teóricos son necesarios en el estudio de los procesos para la gestión de la información de los medios básicos en las organizaciones?
2. ¿Cuál es el estado actual del proceso de gestión de la información de los Activos Fijos Tangibles, Útiles y Herramientas en la Residencia de la Universidad de las Ciencias Informáticas?
3. ¿Por qué se caracteriza el proceso de gestión de la información de los Activos Fijos Tangibles, Útiles y Herramientas en la Residencia de la Universidad de las Ciencias Informáticas?
4. ¿Cuál es el diseño informático que puede dar solución a los problemas diagnosticados en el proceso de gestión de la información de los Activos Fijos Tangibles, los Útiles y Herramientas de la Residencia de la Universidad de las Ciencias Informáticas como propuesta de solución?
5. ¿Cómo llevar a cabo la implementación de un sistema informático como propuesta de solución?
6. ¿Cómo verificar la propuesta de solución?

Los **métodos científicos** utilizados en la investigación fueron los empíricos y teóricos. Dentro de los métodos empíricos se utilizó:

- La **Entrevista,** para identificar las necesidades de la Dirección de Residencia de la UCI, que atiende y gestiona los medios básicos para obtener las funcionalidades con que contará el sistema informático
- La **Observación,** que permite constatar el estado de la gestión del proceso a partir de las prácticas cotidianas y la percepción directa en el panorama real

A nivel teórico fueron utilizados los métodos:

- **Analítico–Sintético**, para realizar un estudio bibliográfico profundo de la teoría existente alrededor del objeto de estudio, determinar las características que tendrá la propuesta de solución, y definir las tecnologías y Herramientas más adecuadas para el desarrollo de la propuesta de solución
- **Hipotético–Deductivo,** posibilitó establecer predicciones a partir del sistema de conocimiento existente
- **Histórico–Lógico,** que permite el estudio crítico de soluciones anteriores referidas en la bibliografía científica, para extraer aspectos positivos de ellos, así como el uso de trabajos como punto de referencia para la definición de requisitos preliminares

El presente documento se encuentra estructurado en tres capítulos. En el primero, se desarrolla una fundamentación teórica y se explican los conceptos que conforman el sistema de Gestión de información de AFT, Útiles y Herramientas de la Residencia de la UCI. En el segundo, se muestran las primeras etapas de la metodología XP, se define lo que el cliente requiere, se abordan las necesidades que se obtienen con la interacción del mismo cliente con el equipo de desarrollo, haciéndose así un Historial de Usuario, se presenta al usuario en su importante papel, se define la primacía e importancia de cada Historia de Usuario y se alude al cálculo de duración de la implementación del proyecto y del esfuerzo, según las definiciones del cliente y en el tercero, se muestra un diseño de la arquitectura de la propuesta de solución, la cual es Modelo-Vista-Controlador. Esta arquitectura se encuentra establecida por el uso del framework Symfony, el cual genera los artefactos Tarjetas Clase-Responsabilidades-Colaboradores, y el modelo de datos a través de un Diagrama Entidad Relación.

*Introducción*

Se hace una descripción de los patrones de diseño, los estilos y estándares de codificación utilizados para desarrollar la propuesta de solución. Además, se muestran las tareas de ingeniería necesarias para llevar a cabo el proceso de desarrollo. También se define la estrategia de pruebas a seguir y los casos de pruebas para cumplir las pruebas de aceptación junto al cliente. Por último, se relacionan las conclusiones, recomendaciones, la bibliografía utilizada y los anexos correspondientes.

# Capítulo 1. Fundamentación Teórica

Se desarrolla una fundamentación teórica y se explican los conceptos que conforman el sistema de Gestión de información de AFT, Útiles y Herramientas de la Residencia de la UCI.

## 1.1 Fundamentos teóricos relacionados con el tema de la investigación

En la llamada «Sociedad de la Información y el Conocimiento», internet ha generado un enorme interés en todos los ámbitos de nuestra sociedad. A finales de siglo XX y principios del XXI autores como Allepuz, Bueno, Bustelo, Castro, Fernández, García, Gil y Huang profundizan en la gestión del capital intelectual: nuevos parámetros de análisis para la economía de la información en la metodología del trabajo en equipo durante el proceso de gestión de información, en el tránsito de la sociedad de la información a la del conocimiento y la calidad de la información en los grupos de trabajo.

De acuerdo con varias fuentes consultadas sobre el tema de la gestión de la información prevalece en la teoría que la misma es un conjunto de procesos de las diferentes fases de la información, desde la obtención hasta el accionar en la toma de decisión, también se alude que son premisas que deben ser cumplidas por el capital humano que atiende su control en el área administrativa.

Entre los principales contenidos de la gestión de la información aparecen diferentes conceptos, tipos, se alude a su carácter sistémico englobador, dado por los diferentes elementos que comprende: procesos, clientes, agentes externos, proveedores, estrategia, propiedad intelectual y sistema; comprende entre otros contenidos la importancia en equipos y grupos de trabajo, al decir de Bueno (2010). Además, se refieren varios principios, entre los que se relacionan con indicadores: identificar la información que se quiere medir con el objetivo, la utilidad para el destinatario, con qué variables medir y criterios de comparación; ventajas e inconvenientes que aparecen durante la gestión de la información, así como las implicaciones y el valor en su aplicación práctica.

Del significado extraído de una información, cada individuo evalúa las consecuencias posibles y adecúa sus actitudes y acciones de manera acorde a las consecuencias previsibles que se deducen del significado de la información. Esto se refiere a qué reglas debe seguir el individuo o el sistema experto para modificar sus expectativas futuras sobre cada posible alternativa.

Otro concepto de información es el que la concibe como un conjunto organizado de datos procesados, que constituyen un mensaje que cambia el estado de conocimiento del sujeto o sistema que recibe dicho mensaje, como una medida de la complejidad de un conjunto de datos. Los datos sensoriales una vez percibidos y procesados constituyen una información que cambia el estado de conocimiento, eso permite a los individuos o sistemas que poseen dicho estado nuevo de conocimiento tomar decisiones pertinentes acordes a dicho conocimiento.

A partir de que el dato es la unidad elemental de la información, de que cuando se anexa la estructura a este es que estamos en presencia de la información, es válido apuntar que esta información más la experiencia y el juicio es lo que permite situarse en el conocimiento en sí. Esta teoría lleva a los autores a identificarse con la gestión de ella y considerarla como un conjunto de actividades realizadas con el fin de controlar, almacenar y, posteriormente, recuperar adecuadamente la información producida, recibida o retenida por cualquier organización en el desarrollo de sus actividades. En el centro de la gestión de la información se encuentra la gestión de la documentación (la información que queda plasmada en documentos).

Relacionado con la gestión de la información en esta investigación existe un conjunto de conceptos, definiciones y teorías que los autores han considerado fuente importante para el desarrollo de la propuesta de solución que se presenta y a continuación serán considerados.

Los **Activos Fijos Tangibles,** según los estudios realizados representan propiedades físicamente tangibles que han de utilizarse por un período largo en las operaciones regulares de la entidad y que normalmente no se destinan a la venta.(2) Se registran por su valor de adquisición y los gastos de transportación y montaje, en los casos de los adquiridos y en los ejecutados con medios propios, se valoran de acuerdo con las disposiciones vigentes.

La información es la base del trabajo que realiza el Área Administrativa de la Residencia ya que según los datos de cada medio básico se decide su ubicación y destino como resultado de la información transmitida por esos datos. En la Dirección de Residencia, los AFT se manejan con las categorías de muebles y enseres. Se valoran al precio de adquisición, o a su costo real de

elaboración o de producción o, en su caso, por un valor equivalente cuando se reciban bienes de uso sin contraprestación monetaria. El precio de adquisición incluye el precio neto pagado por los bienes, representado por el monto de efectivo entregado o su equivalente, más todos los gastos necesarios para colocarlos en el lugar y condiciones de uso, tales como fletes, seguros, derechos y gastos de importación y gastos de instalación, hasta su puesta en marcha, o momento de su alta.(3)

El valor de los bienes se actualiza contablemente por el registro de la depreciación, cuando corresponda, incluyéndose el importe de la misma en el resultado del ejercicio económico. Los bienes recibidos en donación sin que exista un valor de origen, deben ser contabilizados a un valor estimado que represente el desembolso que hubiera sido necesario efectuar para adquirirlos en las condiciones en que se reciben. Los bienes adquiridos en moneda extranjera se registrarán en la moneda nacional, aplicándose el tipo de cambio vigente en la fecha de adquisición. Los activos recibidos sin costo alguno o a un costo inadecuado se valoran al precio vigente de adquisición y de no existir este, por estimación efectuada por peritos. Esta regla de valoración también es aplicable a los bienes recibidos como aportaciones de capital.(4)

Los **Útiles** son una especialidad de los Activos Fijos Tangibles, ya que son medios con un periodo de vida útil más corto, no se les calcula depreciación ya que en su mayoría pasan directamente a la cuenta de gastos de la Universidad, sin embargo, generalmente se desgastan con el tiempo por lo que deben reponerse cada cierto periodo.(5) En este caso se pueden tomar como ejemplo los colchones, espejos y soporte de TV en un apartamento u oficina de la Residencia.

Las **Herramientas** son consideradas como un objeto elaborado a fin de facilitar la realización de una tarea mecánica que requiere de una aplicación correcta de energía (siempre y cuando se hable de Herramienta material. El término Herramienta, en sentido estricto, se emplea para referirse a utensilios resistentes (hechos de diferentes materiales, pero inicialmente se materializaban en hierro como sugiere la etimología), útiles para realizar trabajos mecánicos que requieren la aplicación de una cierta fuerza física. En la Residencia se manejan Herramientas tales como las pinzas, destornilladores, entre otros de igual importancia.

Las Herramientas se diseñan y fabrican para cumplir uno o más propósitos específicos, por lo que son artefactos con una función técnica. Por ejemplo, una pinza es una doble palanca cuyo

punto de apoyo está en la articulación central, la potencia es aplicada por la mano y la resistencia por la pieza que es sujetada. Un martillo, en cambio, sustituye un puño o una piedra por un material más duro, el acero, donde se aprovecha la energía cinética que se le imprime para aplicar grandes fuerzas.

Las Herramientas se dividen en dos grandes grupos: manuales y mecánicas. Estas mismas se subdividen según su uso, como por ejemplo de medición, trazado, sujeción, corte, desbaste, golpe y maquinado. Las manuales usan la fuerza muscular humana (como el martillo), mientras que las mecánicas usan una fuente de energía externa, por ejemplo, la energía eléctrica.(6) Las Herramientas que se controlan en la Residencia son subconjuntos de estos dos grandes grupos.

Dentro de la gestión de la información se comprenden también hechos registrados, datos, descripciones y representaciones. Entre las principales implicaciones están las referidas a que la información debe aparecer explícita, almacenada, archivada y distribuida; que la misma existe fuera del contexto de las personas y que requiere para su uso ser interpretada por el que la utiliza. También comprende en su gestión aspectos diversos como: recurso, gestión, necesidades, estrategias, entre otras.

En el Área Administrativa de la Residencia se maneja mucha información relacionada con los medios básicos. Ejemplo, de un AFT es importante conocer su identificador, su valor, en la medida en que en la organización tenga una estrategia, identificado un problema para cuando se tome una decisión se piense en soluciones prácticas a partir del conocimiento que se tiene de la gestión de la información y del estado en que se encuentra para realizar algún movimiento de ese medio básico.

La gestión de la información como denominación convencional atiende un conjunto de procesos por los cuales se controla el ciclo de vida de la información, desde su obtención (por creación o captura), hasta su disposición final (su archivo o eliminación). “*La gestión de información es todo lo que tiene que ver con obtener la información correcta, en la forma adecuada, para la persona indicada, al costo correcto, en el momento oportuno, en el lugar indicado para tomar la*

*acción precisa*"[1] , según la frase esas son las premisas que se deben cumplir a la perfección en el Área Administrativa de la Residencia.

Dentro de la gestión de la información también se comprenden los subprocesos de la extracción, combinación, depuración y distribución de la información a los interesados. El objetivo de la gestión de la información es garantizar la integridad, disponibilidad y confidencialidad de la información(8).

## 1.2 Análisis de soluciones existentes

Actualmente en el mundo los procesos económicos y sociales se encuentran totalmente automatizados. De esta manera se obtiene una mayor calidad del proyecto y sinergia en el equipo de trabajo, así como una increíble velocidad de desplazamiento de la información. Existen diversos sistemas de Gestión de la Información que cumplen funciones específicas en las entidades a las que se asocian. A pesar de que cada entidad funciona de manera particular, el funcionamiento de sus sistemas de gestión de información sirve de apoyo teórico para el cumplimiento del Objetivo General de la presente investigación.

### 1.2.1 Soluciones similares existentes a nivel internacional

El **sistema de Control de Bienes Web- UG** como un funcionalidad que forma parte del sistema Integral de Información Administrativa (SIIA) fue creado en la Universidad de Guanajuato (México) y desarrollado con el propósito de apoyar a las unidades académicas y administrativas de la Universidad en el control de los bienes muebles e inmuebles bajo su cuidado. Es un sistema administrativo y sistemático que permite el registro oportuno, control e inventarios con información veraz, para poder garantizar el resguardo de los bienes y la identificación, registro y etiquetado de los bienes con eficacia, además de la custodia de los documentos que amparan la propiedad de los mismos.

El sistema antes referido permite entre otras funcionalidades:

1. Capturar los bienes y accesorios en una forma adecuada al catálogo institucional de bienes patrimoniales. Capturar y consultar los movimientos de los bienes y accesorios

[1] Frase tomada de: Woodman, 1985.

2. Generar reportes en formato PDF con la información relevante de inventarios, movimientos y resguardos
3. Realizar búsquedas dentro de los catálogos
4. Clasificar los bienes para darles de alta
5. Registrar los datos de los bienes y eliminar bienes en lotes
6. Realizar búsqueda de empleados
7. Imprimir Inventario General(9)

Otros de los sistemas de gestión es el **Universitas XXI–Económico,** que es un sistema informático modular e integrado, antes llamado SOROLLA. Fue desarrollado por la Oficina de Cooperación Universitaria (OCU), con la ayuda de las universidades. Este sistema está orientado a suministrar la información económica, financiera, tributaria, patrimonial, presupuestaria y analítica requerida tanto por los órganos de gobierno, gestión y control, como por los demás miembros de la comunidad universitaria de Zaragoza, España. Además del mantenimiento de las aplicaciones y el área de Gestión, cuenta con servicios de explotación de los datos para la obtención de indicadores y estadísticas, tanto los definidos para el sistema universitario como los requeridos con carácter interno para la toma de decisiones o la auditoría de procedimientos y funcionamiento administrativo.

El sistema Universitas XXI-Económicos dispone de unos módulos que utilizan las unidades de gasto descentralizadas:

1. La funcionalidad Justificantes de Gasto, permite la gestión de las unidades de planificación descentralizadas mediante el sistema de “a justificar”, tanto por Acuerdo de Anticipo de Caja Fija como por Pagos a Justificar
2. A través de la funcionalidad Gestión de Inventario, se realiza la gestión y el control completo de los activos fijos de la Universidad, muebles e inmuebles; así como el registro de bienes históricos e informáticos. La funcionalidad de Avance informa, al nivel que se establezca, el estado de la ejecución del presupuesto de las Unidades
3. Justificantes de Ingresos: Registro y tramitación de los justificantes de ingresos

4. Gestión Centralizada: Para mantenimiento de datos de proveedores y aplicaciones presupuestarias tanto de ingresos como de gastos(10)

La **Fundacite-Mérida** es un sistema de Inventario y Notas de Pedidos, es un sistema de interés a la administración pública del Gobierno Bolivariano de Venezuela para la automatización en el manejo de Notas de Pedidos, Órdenes de Compra y el Inventario de Materiales, Mobiliario y Equipos, que presenta cuatro opciones de manejo:

1. Opciones del Usuario
2. Opciones sobre Inventario de Materiales y Suministros
3. Opciones sobre Inventario de Mobiliario y Equipos
4. Órdenes de Compra

Estas opciones administrativas permiten entre otras cosas:

1. Generar Listas de Materiales y Suministros: muestra un listado de todos los Materiales y suministros que se han ingresado en el inventario de Almacén, con el código presupuestario, nombre, existencia, unidad y el precio de cada artículo
2. Ingresar Nuevo Artículo al Inventario: permite ingresar artículos nuevos al inventario, cada artículo ingresado pertenece a un grupo de artículo por código presupuestario específico
3. Eliminar un Artículo del Inventario y notas de pedidos
4. Modificar Inventario
5. Entregar Pedidos de Almacén
6. Ingreso de bienes al inventario
7. Consultar información del Mobiliario y/o Equipo por Código y Activo Fecha promedio
8. Proveedores: Puede realizar el ingreso y modificación de proveedores de acuerdo a la especialidad de la empresa que provee el servicio
9. Reporte Histórico: Muestra un listado de Órdenes de Compra realizadas(11)

Los sistemas presentados tienen diversidad de funcionalidades, en los que la gestión de AFT es solo una pequeña parte de ellas. Aunque son sistemas pensados para grandes proyectos, con bastante personal y con una complejidad notable, no es conveniente usarlos en la UCI, ni en la Dirección de Residencia, porque complicaría el proceso de gestión de medios básicos, pero constituyen modelos a consultar para diseñar el sistema y personalizarlo al contexto que requiere la propuesta de solución.

### 1.2.2 Soluciones similares existentes a nivel nacional

Entre las soluciones similares existentes en Cuba se encuentra la **Suite Atenas** es un sistema contable creado por la empresa DESOFT. Actualmente lo utiliza la empresa de Software de Camagüey. Se le integra una funcionalidad de activos fijos que permite:

1. Controlar los activos fijos por entidad, actividad, centros de costos y áreas de responsabilidad
2. Controlar los Movimientos de Altas y Bajas
3. Realizar Traslados y Depreciación de Activos
4. Realizar los cambios de Estados
5. Realizar Conteos Físicos
6. Generación de Reportes
7. Creación de usuarios y grupos y permisos para estos
8. Conteo físico de los productos y obtener estados de faltantes y sobrantes
9. Además de estas funcionalidades, posee un sistema de seguridad que otorga permisos por usuarios o grupos de usuarios(12)

Otro de los sistemas similares es el **sistema de Inventario del Patrimonio Cultural y Natural (SIP):** Este sistema fue desarrollado desde 1988 por el Consejo Nacional de Patrimonio Cultural del Ministerio de Cultura de la República de Cuba, con el objetivo de normalizar los términos a emplear en el inventario automatizado del patrimonio cultural, con diversos fines como son:

1. Cumplimentar la legislación nacional e internacional en lo que respecta al registro de los bienes patrimoniales
2. Ejercer el control y priorizar la conservación de los bienes patrimoniales más valiosos del país
3. Facilitar el intercambio de información relacionado con el patrimonio cultural y natural
4. Responder a las necesidades de diferentes usuarios: museólogos, museógrafos, conservadores, investigadores y otros con intereses más generales
5. Está conformado por 16 bases de datos que abarcan las diversas manifestaciones de la cultura
6. Incluye, además, una base de datos relacionada con los bienes muebles, inmuebles y naturales(13)

El **Sistema Informativo para Ejecutivos (SIEWEB)** fue desarrollado en el Instituto Superior Politécnico “José Antonio Echeverría”, CUJAE, con el objetivo principal de mantener informados a los diferentes directivos de la institución sobre la información almacenada por el sistema económico ASSETS. Esta aplicación Web, implementada en el lenguaje PHP, sirve de interfaz entre el usuario y el sistema ASSETS, realizando consultas directamente a la base de datos del sistema económico y de esta forma permite:

1. Mostrar los reportes que brinda el ASSETS mediante interfaz Web, facilitando la toma de decisiones de los directivos de la institución
2. Visualizar la ubicación de los medios a nivel de área
3. Mantener a los Jefes de Áreas y otros directivos informados de los medios existentes en la CUJAE

Las aplicaciones homólogas de Cuba con respecto al sistema a desarrollar también se encuentran en la misma UCI. A continuación, se presentan los sistemas implementados en la UCI que más se asemejan al sistema propuesto.

El **Sistema de Gestión Integral (ASSETS)** actualmente es usado en la UCI, por la Dirección de Economía. ASSETS NS es un sistema de Gestión Integral estándar y parametrizado que

permite el control de los procesos de Compras, Ventas, Producción, Taller, Inventario, Finanzas, Contabilidad, Presupuesto, Activos Fijos, Útiles y Herramientas y Recursos Humanos. Como sistema Integral todos sus módulos trabajan en estrecha relación, generando, automáticamente, al Módulo de Contabilidad los Comprobantes de Operaciones por cada una de las transacciones efectuadas, esto permite que se pueda trabajar bajo el principio de Contabilidad al Día.

Este sistema multiusuario está montado en una plataforma de servidores SQL, dividido en módulos económicos que trabajan en conjuntos para el control de las actividades económicas, financieras y contables sobre los medios materiales y financieros.

La aplicación ASSETS es cliente-servidor, está programada en Visual Basic 6.0 y Microsoft SQL Server 2000, utilizando adicionalmente Crystal Reports 7.0 para la generación de los reportes de salida. Al estar en plataforma SQL, garantiza mayor seguridad y consistencia en los datos, se obliga que sea ilimitado el número de usuarios conectados y hace posible la utilización de servidores remotos. Facilita, además, el ingreso al inventario y también la administración de los códigos asignados y todo el manejo internamente requerido.

Este sistema, aunque es eficiente no fue concebido en la UCI, no es cubano, aunque tiene licencia, no es gratuita y se debe pagar regularmente. En un futuro sería muy conveniente crear un sistema integral propio de la Universidad para no tener que depender de entidades externas.(15)

La **Aplicación Web para la gestión de la información de los Activos Fijos Tangibles y Útiles de la Residencia Universitaria de la Universidad de las Ciencias Informáticas** está contenida en un Trabajo de Diploma y constituye un antecedente de esta investigación. Es una aplicación Web pensada para contribuir al perfeccionamiento del proceso de gestión de la información de los AFT y Útiles en los departamentos económicos de las Direcciones de Residencia de la UCI. Los lenguajes usados en este sistema son PHP, JavaScript, MySQL para la gestión de bases de datos y Apache como servidor Web. La metodología usada para la realización del proyecto es RUP. No se usa ningún framework para el desarrollo del sistema y fue diseñado en el 2008. (16)

Este sistema tiene funciones similares al sistema en realización, pero fue creado para una infraestructura de trabajo de años atrás, diferente a la actual. En el proceso de gestión de

medios básicos cambiaron los roles del personal y la clasificación de los medios básicos por la introducción de las Herramientas y el flujo de movimientos de los medios básicos. El código fuente de la aplicación web, se encuentra obsoleto porque en estos momentos no se corresponde con las necesidades actuales de la Residencia de la UCI.

## 1.3 Estudio de las metodologías y estándares del desarrollo de software para el diseño del sistema informático a implementar

El desarrollo de los sistemas tradicionales de ciclo de vida se originó en la década de los años 60 del siglo XX para desarrollar a gran escala funcional de sistemas de negocio en una época de grandes conglomerados empresariales. La idea principal era continuar el desarrollo de los sistemas de información de una manera deliberada, estructurada y metódica, reiterando cada una de las etapas del ciclo de vida.

La Metodología de Desarrollo de Software tiene como objetivo presentar un conjunto de técnicas tradicionales y modernas de modelado de sistemas que permitan desarrollar software de calidad, incluyendo heurísticas de construcción y criterios de comparación de modelos de sistemas.

Para tal fin se describen, fundamentalmente, Herramientas de Análisis y Diseño Orientado a Objetos (UML), sus diagramas, especificación, y criterios de aplicación de las mismas. Como complemento se describirán las metodologías de desarrollo de software que utilizan dichas Herramientas, ciclos de vida asociados y discusión sobre el proceso de desarrollo de software más adecuado para las diferentes aplicaciones ejemplos que se presentarán. Principalmente, se presentará el Proceso Unificado, el cual utiliza un ciclo de vida iterativo e incremental.

Las metodologías de desarrollo de software denominadas ágiles se sustentan sobre la base de los valores y principios fijados en el Manifiesto Ágil por la Alianza Ágil[2]. La propuesta de solución a desarrollar posee características que hacen que se ajuste a un proceso de desarrollo ágil:

1. La dimensión de la propuesta de solución es pequeña
2. El equipo de desarrollo posee un tamaño pequeño

[2] Tomado de Agile Alliance, 2001.

3. El cliente forma parte del equipo de desarrollo
4. Los requisitos pueden sufrir cambios constantemente a lo largo del proceso de desarrollo
5. Se cuenta con un corto período de tiempo para el desarrollo

Un ejemplo importante de metodología ágil es Agile Unified Process (AUP), en español traducido como Proceso Unificado Ágil de Scott Ambler. Esta metodología es una versión simplificada del Proceso Unificado de Rational (RUP), el cual describe de una manera simplificada la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. AUP aplica técnicas ágiles incluyendo Desarrollo Dirigido por Pruebas.

La metodología AUP tiene varias ventajas como simplicidad, agilidad, centralización en tareas de alto valor, independencia de Herramienta fácil adaptación. Esta metodología se encuentra en plena ejecución en la UCI, debido a que debe existir una uniformidad en la ejecución de metodologías en los proyectos de la Universidad. El entorno del negocio exige que los cambios de requisitos sobre la marcha sean un aspecto natural, inevitable e incluso deseable en el desarrollo del sistema, y en el caso de AUP-UCI no plantea nada acerca de la variabilidad de los requisitos en el escenario No. 4 (es el escenario que se ajusta al negocio).

Dentro de las metodologías ágiles la que más resalta dentro de los parámetros del proyecto y las necesidades del negocio es XP[3]. XP destaca como una de las que mayor índice de agilidad posee. Resalta también por contar con la mayor cantidad de información disponible y es con diferencia la más popular.[4]

Esta es una metodología ágil pensada para proyectos cortos con requerimientos muy cambiantes o poco claros. Además, el cliente tiene un amplio conocimiento y familiarización con XP, por tanto, le es muy fácil interactuar con el equipo de desarrollo en la realización de las funcionalidades del sistema.

Apunta a una alta integración con el usuario. La idea es ofrecer muchas entregas del sistema agregando funcionalidades paulatinamente en lapsos cortos de tiempo, esto permite que el

[3] Del inglés: Extreming Programing, en español Programación Extrema.
[4] Tomado de: Canós, Letelier y Penadés, 2003.

usuario tenga más claro si el sistema hace lo que él quiere, además de comprometerlo en el desarrollo. Programación en parejas, dos personas por máquina; esto hace el desarrollo más llevadero y el surgimiento de ideas, y la estandarización del código, además los grupos se pueden rotar (propiedad colectiva del mismo).(17)

### 1.3.1 Programación Extrema (XP)

La programación extrema o **eXtremeProgramming** es un enfoque de la ingeniería de software formulado por Kent Beck y De Jean, Extreme Programming Explained: Embrace Change (1999). Es la más destacada de los procesos ágiles de desarrollo de software. Al igual que estos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad.

Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

Esta metodología tiene las siguientes características:

1. Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras
2. Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
3. Programación en parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata
4. Frecuente integración del equipo de programación con el cliente o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo

5. Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes
6. Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad, pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo
7. Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada funcionalidad en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados
8. Simplicidad en el código: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo. La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Mientras más simple es el sistema, menos tendrá que comunicar sobre este, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.(20)

El proceso de desarrollo de XP sigue un ciclo, el cual consiste a grandes rasgos en los siguientes pasos[5]:

1. El cliente define el valor del negocio a implementar
2. El programador estima el esfuerzo necesario para su implementación
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo
4. El programador construye ese valor del negocio
5. Vuelve al paso uno

[5] Tomado de: Jeffries, Anderson, y Hendrickson, 2001.

XP clasifica como una metodología ágil que se enfoca en potenciar las relaciones entre los miembros del equipo de desarrollo como la clave para el éxito, promueve el trabajo en equipo, se interesa por la superación de los desarrolladores, y favorece un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, en la comunicación fluida entre todos los participantes, la sencillez de las soluciones desarrolladas y en la audacia para enfrentar los cambios que surjan. Por lo que XP se adecúa perfectamente a proyectos con requisitos imprecisos y propensos al cambio[6].

El ciclo de vida ideal de XP consta de seis fases: Exploración, Planificación de la Entrega, Iteraciones, Producción, Mantenimiento y Muerte del Proyecto[7]. Este ciclo puede apreciarse en la Figura 1.

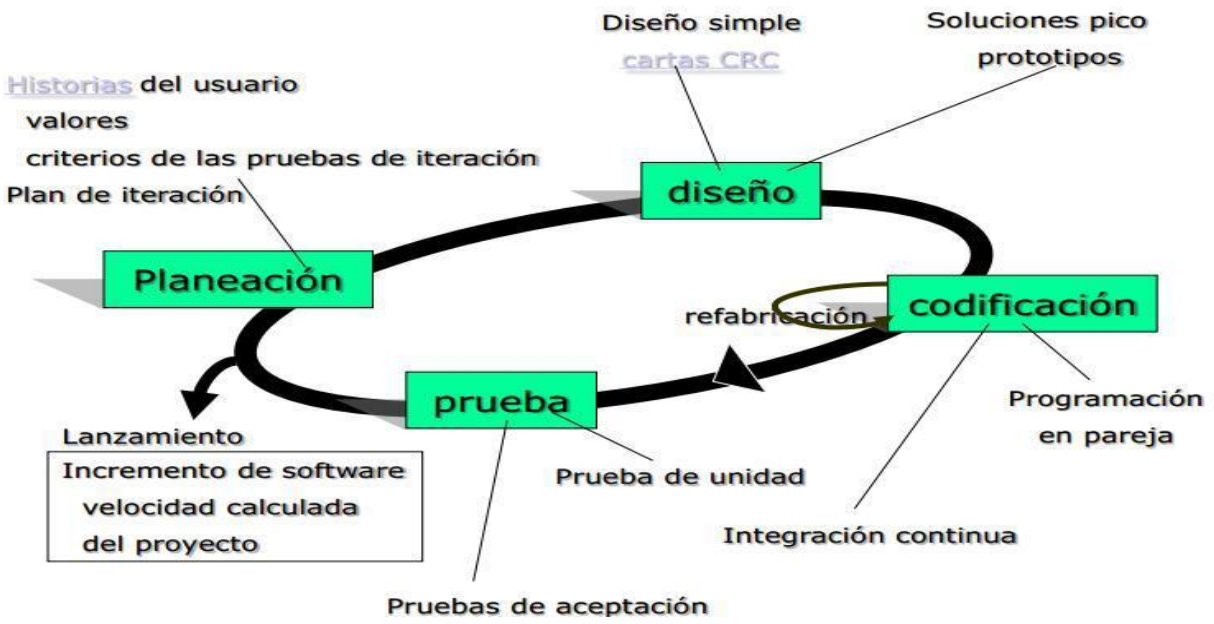


**Figura 1: Ciclo de vida de XP.** (21)

Aunque son múltiples los beneficios que se pueden obtener del uso de todas las prácticas en conjunto, la metodología no obliga, ni exige la aplicación estricta de todas ellas. Es importante destacar que el cliente está altamente familiarizado con la metodología XP y con las

[6] Tomado de: Letelier y Penadés, 2006.
[7] Tomado de: Letelier y Penadés, 2006.

características anteriormente descritas. Se puede concluir que XP será la metodología que guiará el proceso de desarrollo de la propuesta de solución.

### 1.3.2 Lenguaje Unificado de Modelado (UML)

El **UML**[8]: "*…es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software."* (66)

El Lenguaje Unificado de Modelado proporciona a los desarrolladores un vocabulario que incluye tres categorías: elementos, relaciones y diagramas[9].

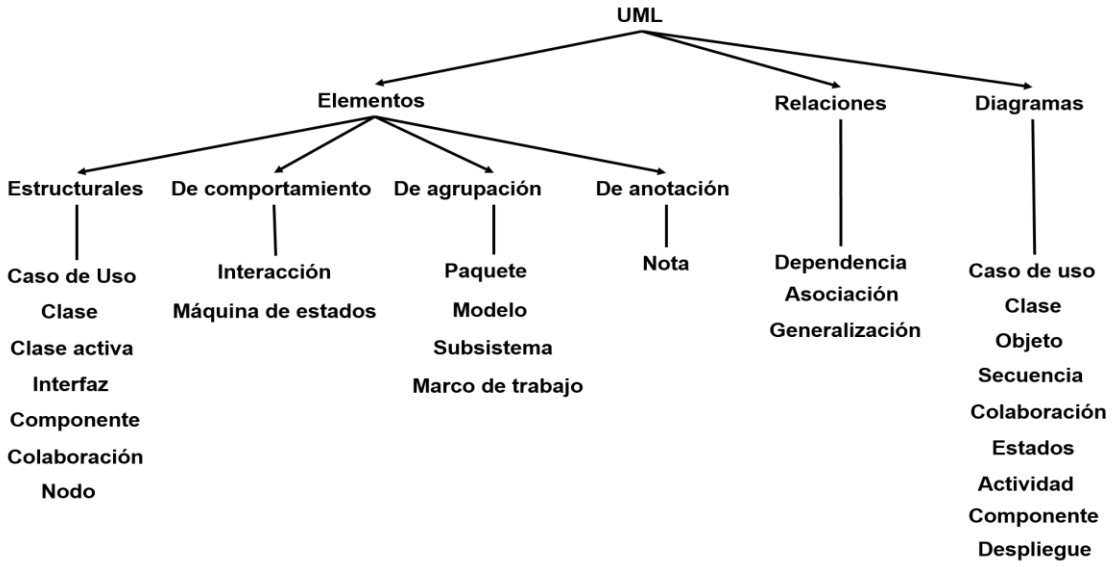


**Figura 2: Vocabulario de UML.**

UML está pensado para ser empleado en Herramientas interactivas de modelado visual que tengan generadores de código y/o generadores de informes. La especificación de UML no define un proceso estándar, pero está ideado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos. (66)

La capacidad de UML para modelar cualquier sistema de *software* y el hecho de que haya sido adoptado por el Grupo de Gestión de Objetos[10]como un estándar desde noviembre de 1997,

[8] Del inglés: Unified Modeling Language.
[9] Tomado de: Jacobson y Rational Software Corporation, 2000.
[10] OMG por sus siglas del inglés Object Management Group.

permitieron determinarlo como el lenguaje de modelado para desarrollar la propuesta de solución.

## 1.4 Tendencias y tecnologías actuales. Selección de las Herramientas y lenguajes de desarrollo

En la actualidad contar con un sitio web es de suma importancia, independientemente del objetivo del sitio, cada minuto que pasa nuestra sociedad está actualizando información de lo que sucede en diferentes lugares del mundo, y en este siglo XXI es de vital importancia estar informado de todos los acontecimientos que nos rodean.

Se determinó desarrollar una **aplicación web** debido a las necesidades del cliente, las cuales requerían versatilidad en el movimiento de datos, y las aplicaciónes web tienen las siguientes ventajas que se acoplan a las características de la Dirección de Residencia:

La Compatibilidad multiplataforma que comprenden las aplicaciones web tienen un camino mucho más sencillo para la compatibilidad multiplataforma que las aplicaciones de software descargables. Varias tecnologías incluyendo Java, Flash, ASP y Ajax permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales.

Las aplicaciones web exigen menos requerimientos de memoria, tienen muchas más razonables demandas de memoria RAM de parte del usuario final que los programas instalados localmente. Al residir y correr en los servidores del proveedor, a esas aplicaciones basadas en web donde se usa en muchos casos la memoria de las computadoras que ellos corren, dejando más espacio para correr múltiples aplicaciones al mismo tiempo sin incurrir en frustrantes deterioros en el rendimiento.

Estas aplicaciones presentan menos Bugs[11] y deberían ser menos propensas a colgarse y crear problemas técnicos debido a software o conflictos de hardware con otras aplicaciones existentes, protocolos o software personal interno. Con aplicaciones web, todos utilizan la misma versión, y todos los bugs pueden ser corregidos tan pronto como son descubiertos. Esta es la razón por la cual las aplicaciones basadas en web deberían tener mucho menos bugs que el software de escritorio descargable tradicional.

[11] Del inglés: Bicho, terminología informática para catalogar errores de programación.

Las **páginas web** tienen como características importantes las siguientes:

1 Presentan contenidos de utilidad ya que la temática y las búsquedas deben ir acordes con el usuario final, si no, lo más seguro es que el visitante abandone la página rápidamente

2 Debe ser intuitivo haciendo fácil la navegación generando que las visitas consigan fácilmente su objetivo. Además, evita abandonos en tu página web

3 Realización un diseño atractivo. En una web, una primera impresión cuenta, ya que representa nuestro negocio, e incluso a nosotros mismos. Si a un usuario no le gusta lo que ve, o no le genera confianza, lo más probable es que abandone el sitio web, en muchos casos sin tan siquiera tener en cuenta el contenido. Por lo cual un buen diseño genera confianza, seriedad y muy buena impresión

4 Creación de un contenido bien estructurado con un desglose del contenido claro y sencillo, bien explicado. La importancia de una estructura clara y objetiva es lo que mantendrá a tu usuario conectado(22)

### 1.4.1 Selección del framework de desarrollo en el servidor

Para el desarrollo de la aplicación es necesario el uso de los frameworks[12], y primero hay que entender su significado y para qué sirve. En el desarrollo de software, un **framework** o **infraestructura digital**, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras Herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Los frameworks[13] tienen como objetivo principal ofrecer una funcionalidad definida, auto contenido, siendo construidos usando patrones de diseño, y su característica principal es su alta cohesión y bajo acoplamiento. Para acceder a esa funcionalidad, se construyen piezas, objetos, llamados objetos calientes, que vinculan las necesidades del sistema con la funcionalidad que este presta. Esta funcionalidad, está constituida por objetos llamados fríos, que sufren poco o ningún cambio en la vida del framework, permitiendo la portabilidad entre

[12] Del inglés: infraestructura, armazón, marco.
[13] Tomado de: Riehle, Dirk (2000)

distintos sistemas.(24) Según la información anterior se puede hacer un bosquejo de algunos de los frameworks más conocidos en el mundo:

El framework **Vaadin** es para el desarrollo de la capa de presentación de aplicaciones web en Java, que ofrece una interfaz de escritorio tradicional, sin necesidad de usar JavaScript ni JSON ni XML ni siquiera HTML (en la sección Leguajes de Programación se explica el significado de estos leguajes). Es un marco de aplicación web de código abierto para aplicaciones de Internet sofisticadas.(25)

En contraste con las bibliotecas de JavaScript y soluciones basadas en navegador plug-in, que cuenta con una arquitectura de servidor, lo que significa que la mayoría de la lógica se ejecuta en los servidores. La tecnología Ajax se utiliza en el lado del navegador para garantizar una experiencia de usuario rica e interactiva. En el lado del cliente Vaadin se construye en la parte superior y se puede ampliar con Google Web Toolkit[14].(26)

El framework **CodeIgnite** se basa en el patrón de desarrollo del Modelo-Vista-Controlador (MVC), el cual es muy popular. Mientras que las clases controlador son una parte necesaria del desarrollo en el marco CodeIgniter, los modelos y puntos de vista son opcionales. Este framework destaca con mayor frecuencia por su velocidad cuando se compara con otros marcos de PHP. Es de los frameworks más seguros. Es uno de los frameworks más rápidos del mercado y su configuración muy fácil y rápida.(27)

Otro framework MVC es **Laravel,** caracterizado por ser el más popular y usado. Su línea de aprendizaje es corta, es ideal tanto para proyectos grandes como pequeños, es uno de los más seguros y cuenta con la mayor comunidad. Su filosofía es desarrollar código PHP de forma elegante y simple, evitando el "código espagueti"[15]. (28) Tiene como objetivo ser un framework que permita el uso de multitud de funcionalidades. Gran parte de Laravel está formado por dependencias, especialmente de **Symfony**, esto implica que el desarrollo de Laravel dependa también del desarrollo de sus dependencias.(29)

[14]Framework creado por Google que permite ocultar la complejidad de varios aspectos de la tecnología AJAX

[15]Término peyorativo para los programas de computación que tienen una estructura de control de flujo compleja e incomprensible.

En la comunidad mundial de programadores existen diferentes criterios del uso de los frameworks. Por tanto, en la Figura 4 se muestra los frameworks más usados en la actualidad.

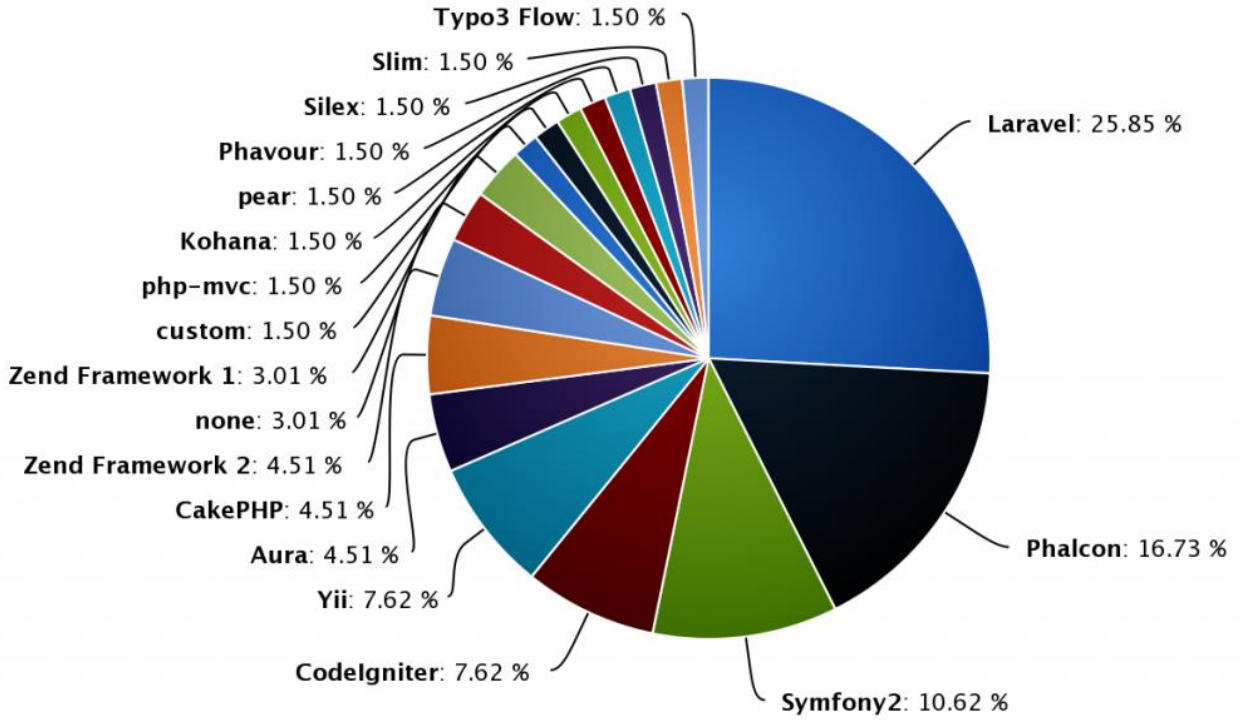


**Figura 4: Frameworks más usados(30)**

Como se puede apreciar el framework **Symfony** se encuentra entre los 3 más usados, por encima de más de 10 frameworks en todo el mundo. Los anteriormente explicados tienen muchas diferencias, pero este framework tiene de una manera explícita características que se apegan a las necesidades del proyecto. Además, en la Tabla 1 se muestran los **mejores frameworks según la cantidad de características** que presenta:

**Tabla 1: Mejores frameworks según sus características.(30)**

BESTWEBFRAMEWORKS

PHP Frameworks

| Framework | PHP 4 | PHP 5 | MVC | Modules | ORM | EDP | Auth | Cache | Validator | Ajax | License | DL | RV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Zend Framework | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | New BSD | | |
| CakePHP | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | MIT | | |
| symfony | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | MIT | | |
| CodeIgniter | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | Apache/BSD | | |
| Seagull | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | BSD | | |
| Prado | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | BSD | | |
| Solar | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | New BSD | | |
| eZ Components | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | New BSD | | |
| Kohana | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | Sort of BSD | | |

De manera resumida se puede determinar que **Symfony** compite sin ningún problema con la élite de frameworks en el mundo. Por tanto, para la realización de la aplicación se escoge este framework y su respectiva versión por las razones anteriores y por las siguientes:

El framewok **Symfony 2.8.7** es completamente diseñado para optimizar el desarrollo de las aplicaciones web basado en el patrón MVC. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias Herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web.(31)

Symfony está desarrollado completamente en PHP 5.3. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y

Microsoft SQL Server. Se puede ejecutar en una amplia gama de plataformas, entre ellas Windows[16].(32)

Las ventajas del uso de Symfony son:

1. Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y Unix-like estándares)
2. Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso de ORM[17], permiten cambiar con facilidad de SGBD[18] en cualquier fase del proyecto
3. Utiliza programación orientada a objetos y características como los espacios de nombres
4. Sencillo de usar en la mayoría de casos, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos
5. Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador solo debe configurar aquello que no es convencional
6. Sigue la mayoría de mejores prácticas y patrones de diseño para la web
7. Código fácil de leer que incluye comentarios y que permite un mantenimiento muy sencillo
8. Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
9. Una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo(33)

### 1.4.2 Selección de la biblioteca JavaScript

La biblioteca **jQuery 2.2.1** es de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM[19], manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Es la biblioteca de JavaScript más utilizada.(34)

[16] Sistema Operativo de la corporación Microsoft.
[17] Del inglés: Object Relacional Maping, en español Objeto de mapeo relacional.
[18] sistema Gestor de Bases de Datos.
[19] Del inglés: Document Object Model, en español Modelo de Objetos del Documento o Modelo en Objetos para la Representación de Documentos.

Esta biblioteca de JavaScript es un software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT[20] y la Licencia Pública General de GNU[21] v2, permitiendo su uso en proyectos libres y privados. jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.(35)

### 1.4.3 Selección del framework CSS

Para la realización del diseño de una aplicación es necesario utilizar un **framework de diseño o CSS,** se realizó una investigación con algunos de los frameworks CSS más usados en el mundo.

El **framework Gumby** fue desarrollado a la base de Saas, que es un pre-procesador potente de CSS. Actualmente Gumby está disponible en la versión 2.6 con muchas características Útiles, estilos predefinidos y elementos de interfaz comunes, tales como palancas, menús desplegables, botones, pestañas etc. Con la ayuda de Gumby, se puede personalizar y formar una cuadrícula (grid) para su diseño. Por otra parte, se puede adaptar a cualquier tamaño y resolución de pantalla.(36)

En el caso de **Kube,** es adaptable y compatible con todo lo necesario para el diseño del sitio web. Presenta la mayoría de las características de Gumby, pero a diferencia del anterior tiene una cuadrícula (grid) flexible revolucionaria con asombrosa tipografía.(37)

La biblioteca **YAML** [22] es ideal para los sitios web flexibles, accesibles y adaptables (responsive). El framework ofrece un conjunto de bloques para la construcción de sitios web. Los elementos incluyen la funcionalidad de tipografía, la navegación, los formularios etc. Este framework contiene un buen Ajax constructor que es una herramienta muy útil para el desarrollo visual del diseño CSS.(38) A pesar de las buenas propiedades de los frameworks anteriormente mencionados se ha elegido como el indicado para el desarrollo de la aplicación el siguiente:

[20] Licencias de software que ha empleado el Instituto Tecnológico de Massachusetts.
[21] Del inglés: General Public License, garantiza a los usuarios finales la libertad de usar, estudiar, compartir (copiar) y modificar el software.
[22] Del inglés: Yet Another Multicolumn Layout, en español OtroDiseñode Varias Columnas.

El framework o conjunto de Herramientas de software libre **Bootstrap** en su versión 3.3.6 es para el diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.(39) Tiene las siguientes ventajas:

1. Ofrece un paquete de elementos web personalizables: con Bootstrap se puede diseñar una web jugando con sus elementos compuestos por diferentes combinaciones de HTML, CSS y Javascript, de manera que las piezas siempre encajan
2. Utiliza componentes vitales para los desarrolladores: como HTML5, CSS3, jQuery o GitHub, entre otros
3. Sus plantillas son de sencilla adaptación. Se desarrolló con la idea de facilitar el proceso de adaptación web a todo tipo de dispositivos
4. Incluye Grid system: muy útil para maquetar por columnas
5. Se integra con librerías JavaScript
6. Usa Less: un lenguaje de las hojas de estilo CSS preparado para enriquecer los estilos de la web
7. Es una herramienta de uso ágil y sencillo: facilita enormemente el diseño de interfaces y además incluye por defecto una plantilla bastante optimizada
8. De manera general Bootstrap es perfecto para los diseñadores de todos los niveles, los proyectos de todos los tamaños, así como dispositivos de todas las formas.(40)

### 1.4.4 Lenguajes de programación

Existen muchos lenguajes de programación en la actualidad, cada uno con sus características y ventajas. A continuación, se muestran los lenguajes que se usan en la realización de la aplicación.

Como lenguaje de programación del lado del cliente es importante **JavaScript** en la implementación debido a que es un lenguaje sencillo, de programación muy liviana. Utiliza poca memoria, sin embargo, tiene gran variedad de efectos visuales. Tiene fácil manejo de datos. Es

ligero de carga y muy fácil de integrar. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.(41)

El lenguaje **PHP** en su versión 7.0.1 es de programación de uso general, de código del lado del servidor, originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con una funcionalidad de procesador de PHP que genera la página Web resultante.

Este lenguaje ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo. PHP puede hacer cualquier tarea que se pueda realizar con un script CGI, como procesar la información de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies. La elección del mismo está determinada por seleccionar a Symfony como framework de desarrollo en el servidor, el cual se encuentra escrito en dicho lenguaje.(42)

Las siglas de **HTML**[23] hacen referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia para la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, entre otros. Es un estándar a cargo de la W3C, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la World Wide Web. Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado. El HTML5 incluye nuevas etiquetas que permiten representar elementos familiares de las páginas, tal es el caso de: <*header*> para el encabezado de las páginas, <*nav*>para la navegación, <*section*> para una sección de la página y <*figure*> para asignar un título a una imagen.

[23] Del inglés: Hyper Text Markup Language, en español Lenguaje de Marcas de Hipertexto.

HTML5 propone estándares para cada aspecto de la web y también un propósito claro para cada una de las tecnologías involucradas. Con HTML5, HTML provee los elementos estructurales, CSS se encarga de volver esa estructura utilizable y atractiva a la vista, y JavaScript tendrá el poder necesario para proveer dinamismo y construir aplicaciones web completamente funcionales y con mayor capacidad de interacción con el usuario.[24]

La selección de HTML5 como el lenguaje para realizar el maquetado de la propuesta de solución está sustentada en las novedosas Herramientas que incorpora para el desarrollo web y el hecho de ser este un estándar establecido por la Wide Web Consortium (W3C).(43)

El lenguaje **CSS**[25] es usado para definir y crear la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación. La información de estilo puede ser definida en un documento separado o en el mismo documento HTML. En este último caso podrían definirse estilos generales con el elemento «style» o en cada etiqueta particular mediante el atributo «style».

Las CSS ofrecen propiedades para ampliar el lenguaje HTML en la representación visual de las páginas web. El lenguaje de programación utilizado por los desarrolladores para codificar dichas hojas, también es denominado CSS. Es el más conocido y utilizado para definir las propiedades de formato de los diferentes elementos HTML[26].

### 1.4.5 Entorno de desarrollo integrado

Un **ambiente de desarrollo integrado** o entorno de desarrollo interactivo, IDE[27], es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software. Normalmente, un IDE consiste de un editor de código fuente, Herramientas de construcción automáticas y un depurador. La mayoría de los IDE

[24] Tomado de: Gauchat, 2012.
[25] Del inglés: Cascading Style Sheets, en español Hoja de Estilo en Cascada.
[26] Tomado de: Lancker, 2009.
[27] Del inglés: Integrated Development Environment, en español Ambiente de Desarrollo Integrado.

tienen auto-completado inteligente de código (IntelliSense). Algunos IDE contienen un compilador, un intérprete, o ambos.(45)

El límite entre un IDE y otras partes del entorno de desarrollo de software más amplio no está bien definido. Muchas veces, a los efectos de simplificar la construcción de la interfaz gráfica de usuario (GUI, por sus siglas en inglés) se integran un sistema controlador de versión y varias Herramientas. Muchos IDE modernos también cuentan con un navegador de clases, un buscador de objetos y un diagrama de jerarquía de clases, para su uso con el desarrollo de software orientado a objetos.(46) Existen muchos IDE usados en el mundo, entre los mejores se encuentran los siguientes:

El IDE **Eclipse** es uno de los más usados, es multiplataforma y multilenguaje. Por muchos programadores es muy aclamado, sin embargo, hay que tomar en cuenta que en Eclipse es necesario agregar varios plugins para que funcione al cien por ciento (dependiendo de lo que se necesite hacer), si se comienza en el mundo de la programación se toma en cuenta. Eclipse dispone de un Editor de texto con un analizador sintáctico. La compilación es en tiempo real.(47)

Si lo que se desea es desarrollar para Windows 8 y sus versiones anteriores utilizando tecnología de Microsoft, la mejor opción es **Visual Studio**. Dentro de este IDE se podrá desarrollar utilizando el entorno .net. También tiene la opción de convertirse en un desarrollador para la tienda de Windows. Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos, consolas, entre otras tecnologías.(48) Existen otros IDEs muy conocidos, pero el más adecuado para la realización de la aplicación según sus desarrolladores debido a su usabilidad y funcionalidades es:

El IDE **Netbeans** en su versión 8.1 es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDEes un producto libre y gratuito sin restricciones de uso**.** La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un funcionalidad es un archivo Java que

contiene clases de java escritas para interactuar con las APIs[28] de NetBeans y un archivo especial (manifest file) que lo identifica como funcionalidad. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que las funcionalidades pueden ser desarrollados. (49)

Este IDE es multiplataforma, debido a que fue desarrollado en Java puede ser instalado y utilizado en cualquier sistema Operativo (Windows, Mac OS, Linux, etc). Es un entorno de desarrollo completo y profesional. Aunque puede ser utilizado para cualquier otro lenguaje de programación (C, C++, Ruby, PHP, etc). Soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB, y aplicaciones). Puede expandirse su funcionalidad a través de la instalación de plugins.(50)

### 1.4.6 Herramienta CASE de modelado con UML

Las Herramientas CASE[29] son diversas aplicaciones informáticas propuestas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas Herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso para realizar un diseño del proyecto, cálculo de costos, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras. Debido a la experiencia del equipo de desarrollo y a su fácil manejo la herramienta CASE que se selecciona en el modelado del sistema usando el lenguaje UML anteriormente explicado es:

**Visual Paradigm for UML 8.0** la cual propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas.(51) Constituye una herramienta privada disponible en varias ediciones, cada una destinada a satisfacer diferentes necesidades. Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque Orientado a Objetos.

[28] Del inglés: Application Programming Interface, del español Interfaz de Programación de Aplicaciones.
[29] Del inglés: Computer Aided Software Engineering, en español Ingeniería de Software Asistida por Computadora.

### 1.4.7 Servidor de Bases de Datos

Los **Servidores de Bases de Datos**. También conocidos como RDBM[30], son programas que permiten organizar datos en una o más tablas relacionadas. Los servidores de Bases de Datos se utilizan en todo el mundo en una amplia variedad de aplicaciones.(53) Para bases de datos con múltiples usuarios sirve un servidor de base de datos. Las bases de datos están situadas en un servidor y se puede acceder a ellas desde terminales o equipos con un programa -llamado cliente- que permita el acceso a la base o bases de datos. Los gestores de base de datos de este tipo permiten que varios usuarios hagan operaciones sobre ella al mismo tiempo.(54) Dentro de los RDBM más conocidos se encuentran los siguientes:

El RDBM **MySQL** es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM[31], pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones. Sea cual sea el entorno en el que va a utilizar MySQL, es importante monitorizar de antemano el rendimiento para detectar y corregir errores tanto de SQL[32] como de programación.(55)

Como la base de datos líder del mercado se encuentra **Oracle Database** ya que soporta todos los tipos de datos relacionales estándares, así como también datos nativos como XML, texto, imágenes, documentos, audio, y datos espaciales. El acceso a la información es realizado a través de interfaces estándares. Se considera a Oracle Database como uno de los sistemas de bases de datos más completos, destacando: soporte de transacciones, estabilidad, escalabilidad, y soporte multiplataforma.(56)

A pesar del profesionalismo Oracle, no es la escogida para ejercer de Servidor de Bases de Datos, debido a que es un Servidor muy robusto destinado a proyectos de gran envergadura. El

[30] Del inglés: Relational Data Base Management System, en español Servidor de Bases de Datos.
[31] Es el mecanismo de almacenamiento de datos usado por defecto por el sistema administrador de bases de datos relacionales MySQL hasta su versión 5.5.
[32] Del inglés: Structured Query Language, en español Lenguaje de Consulta Estructurado. Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

servidor seleccionado por la familiarización del equipo de desarrollo y por su sistema de gestión simplificada es:

El RDBM **PostgreSQL** en su versión 9.3.10, es un sistema de gestión de bases de datos relacional orientado a objetos y libre, publicado bajo la licencia PosgreSQL. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, libre y apoyada por organizaciones comerciales. Dicha comunidad es denominada el PGDG[33].(57) Entre sus ventajas se encuentra:

1. Ampliamente popular e ideal para tecnologías Web
2. Fácil de Administrar
3. Su sintaxis SQL es estándar y fácil de aprender
4. Footprint bajo de memoria, bastante poderoso con una configuración adecuada
5. Multiplataforma
6. Capacidades de replicación de datos
7. Soporte empresarial disponible(58)

### 1.4.8 Servidor Web

Un **servidor web** o servidor HTTP es un programa informático que procesa una aplicación del lado del servidor, realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente y generando o cediendo una respuesta en cualquier lenguaje o Aplicación del lado del cliente. El código recibido por el cliente suele ser compilado y ejecutado por un navegador web. Para la transmisión de todos estos datos suele utilizarse algún protocolo. Generalmente se usa el protocolo HTTP para estas comunicaciones, perteneciente a la capa de aplicación del modelo OSI. El término también se emplea para referirse al ordenador que ejecuta el programa.(59) El servidor web que se usa para desplegar las aplicaciones convenientes es:

[33] Del inglés: PostgreSQL Global Development Group, en español Grupo de Desarrollo Global PostgreSQL.

**Apache** en su versión 2.0 es un servidor de aplicaciones web con licencia de software libre creada por la Apache Software Foundation (ASF). La licencia Apache requiere la conservación del aviso de copyright, no requiere la redistribución del código fuente cuando se distribuyen versiones modificadas. Todo el software producido por la ASF o cualquiera de sus proyectos está desarrollado bajo los términos de esta licencia. Además, algunos proyectos que no pertenecen a la ASF también siguen la licencia Apache.(60)

## 1.5 Conclusiones del capítulo 1

- Se obtuvieron los datos necesarios para el desarrollo de la aplicación.
- Se demostró la importancia y necesidad del desarrollo de este sistema.
- Se estableció la primera conexión entre los desarrolladores y el cliente conformando así las fortalezas y debilidades de la aplicación.

# Capítulo 2 Exploración y Planificación

## 2.1 Descripción de la propuesta de solución

La propuesta de solución tiene como propósito mejorar el proceso de gestión de los AFT, Útiles y Herramientas de la Residencia de la UCI, obteniéndose así un mejor control de estos medios básicos. Los roles que tienen actividad dentro del sistema están en correspondencia con los cargos que se puedan ocupar en la Dirección de Residencia, estos son: Director, Especialista (Este rol incluye los cargos Especialista General y Especialista Superior), Técnico General y Técnico B.

Inicialmente el sistema informático cuenta con un usuario Director en su configuración predeterminada. Este usuario es el encargado de realizar el primer acceso al sistema informático y configurar todo el espacio de trabajo. El Director tiene la responsabilidad de garantizar el acceso al sistema, a través de la gestión de usuarios, roles y permisos. Además, tiene acceso a todas las funcionalidades del sistema sin restricción alguna y es el único usuario que tiene acceso a la funcionalidad aprobar el movimiento de un medio básico. El usuario Especialista tiene acceso limitado solo a gestionar la seguridad del sistema y aprobar el movimiento de un medio básico.

El usuario Técnico General tiene acceso a importar medios básicos, mostrar reportes de medio básicos, mostrar acta de responsabilidad material, mostrar el submayor AFT, mostrar el submayor Útiles y Herramientas, gestionar el movimiento de medios básicos, mostrar el modelo de movimiento de medios básicos, mostrar el listado de movimiento de medios básicos pendientes, mostrar las trazas del movimiento de un medio básico y mostrar el listado de movimientos realizados. El usuario Técnico B es el que menos privilegios posee ya que solo tiene acceso a las funcionalidades mostrar el submayor AFT y mostrar el submayor Útiles y Herramientas.

El sistema podrá ser desplegado como un sistema centralizado en lugares donde exista la infraestructura tecnológica necesaria para conectar varias computadoras, ello no quita la posibilidad de usarlo como un sistema distribuido en computadoras que se encuentren aisladas sin conexión a una red.

## 2.2 Requisitos funcionales

Los requisitos funcionales especifican una acción que debe ser capaz de realizar el sistema (66). Seguidamente se muestran los requisitos funcionales del sistema:

1. Gestionar usuarios, roles y permisos
2. Gestionar activos fijos
3. Gestionar Útiles
4. Gestionar Herramientas
5. Gestionar entidad
6. Gestionar área
7. Gestionar tipos de movimiento
8. Gestionar cargos
9. Gestionar personas
10. Importar los activos fijos del área
11. Importar los Útiles y Herramientas del área
12. Visualizar reporte de activos fijos
13. Exportar a formato PDF reporte de activos fijos
14. Visualizar el acta de responsabilidad material dado un local
15. Exportar a formato PDF el acta de responsabilidad material dado un local
16. Visualizar el submayor de activos fijos del área
17. Exportar a formato PDF el submayor de activos fijos del área
18. Visualizar el submayor de Útiles y Herramientas del área
19. Exportar a formato PDF el submayor de Útiles y Herramientas del área
20. Gestionar el movimiento de un medio

21. Visualizar el modelo de movimiento de medios dado un movimiento
22. Exportar a formato PDF el modelo de movimiento de medios dado un movimiento
23. Aprobar el movimiento de un medio
24. Visualizar el listado de movimientos de medios pendientes a aprobación
25. Exportar a formato PDF el listado de movimientos de medios pendientes a aprobación
26. Visualizar las trazas del movimiento de un medio
27. Exportar a formato PDF las trazas del movimiento de un medio
28. Visualizar el listado de movimientos realizados
29. Exportar a formato PDF el listado de movimientos realizados
30. Generar la planificación anual del control a los activos fijos, Útiles y Herramientas del área
31. Exportar a formato PDF la planificación anual del control a los activos fijos, Útiles y Herramientas del área.

## 2.3 Requisitos no funcionales

Los aspectos no funcionales especifican propiedades del sistema, como restricciones del entorno o de la implementación, rendimiento, dependencias de la plataforma, facilidad de mantenimiento, extensibilidad y fiabilidad (66).

**Software:**

1. Versión mínima de PHP: PHP 5.3.9
2. Como servidor de base de datos: PostgresSQL teniendo como versión 9.3.10
3. Soporta navegadores modernos compatibles con HTML 5 y CSS 3 (Chrome, Firefox, Internet Explorer, Opera, Safari)

**Hardware:**

1. Espacio de disco: 400 MB (mínimo)
2. Procesador: 1GHz (mínimo)

3. Memoria RAM: 1 GB (mínimo)

**Confiabilidad:**

El sistema debe ser preciso con la información que maneja, haciendo énfasis en los resultados de los análisis que ejecutará. Para evitar errores que puedan incidir negativamente en la toma de decisiones.

**Portabilidad:**

El sistema debe ser independiente de la plataforma donde se instale, debe mostrar compatibilidad tanto en Microsoft Windows como en GNU/Linux.

**Usabilidad:**

La interfaz del sistema debe ser amigable a los usuarios finales. Incluirá solo la información y elementos necesarios que hagan intuitivo su uso. Deberá contar con una adecuada combinación de color y tipografía.

## 2.4 Usuarios del sistema

Los usuarios son aquellas personas que tienen contacto directo con el sistema. Existe una jerarquía entre ellos, donde cada rol tiene responsabilidades y restricciones diferentes entre sí. En la tabla 2 se muestran los usuarios con sus respectivas responsabilidades.

**Tabla 2: Usuarios del sistema**

| Usuarios | Responsabilidades |
|---|---|
| **Director** | Este usuario es el encargado de realizar el primer acceso al sistema informático y configurar todo el espacio de trabajo. El Director tiene la responsabilidad de garantizar el acceso al sistema, a través de la gestión de usuarios, roles y permisos. Además, tiene acceso a todas las funcionalidades del sistema sin restricción alguna |

| | |
|---|---|
| **Especialista** | Tiene acceso limitado solo a gestionar la Usuarios, Roles y Permisos, y aprobar el movimiento de un medio básico. |
| **Técnico general** | Tiene acceso a importar medios básicos, mostrar reportes de AFT, mostrar acta de responsabilidad material, mostrar el submayor AFT, mostrar el submayor Útiles y Herramientas, gestionar el movimiento de medios básicos, mostrar el modelo de movimiento de medios básicos, mostrar el listado de movimiento de medios básicos pendientes, mostrar las trazas del movimiento de un medio básico y mostrar el listado de movimientos realizados. |
| **Técnico B** | Solo tiene acceso a las funcionalidades mostrar el submayor AFT y mostrar el submayor Útiles y Herramientas. |

## 2.5 Diagrama conceptual del negocio

El diagrama conceptual del negocio es un artefacto de tipo UML. Este diagrama puede ser utilizado para capturar y expresar el entendimiento ganado sobre el negocio. Fue necesario el uso de este diagrama porque con la identificación de los conceptos y objetos relacionados con el control de los medios básicos en la Residencia, el equipo de desarrollo puede mejorar la comprensión del negocio.

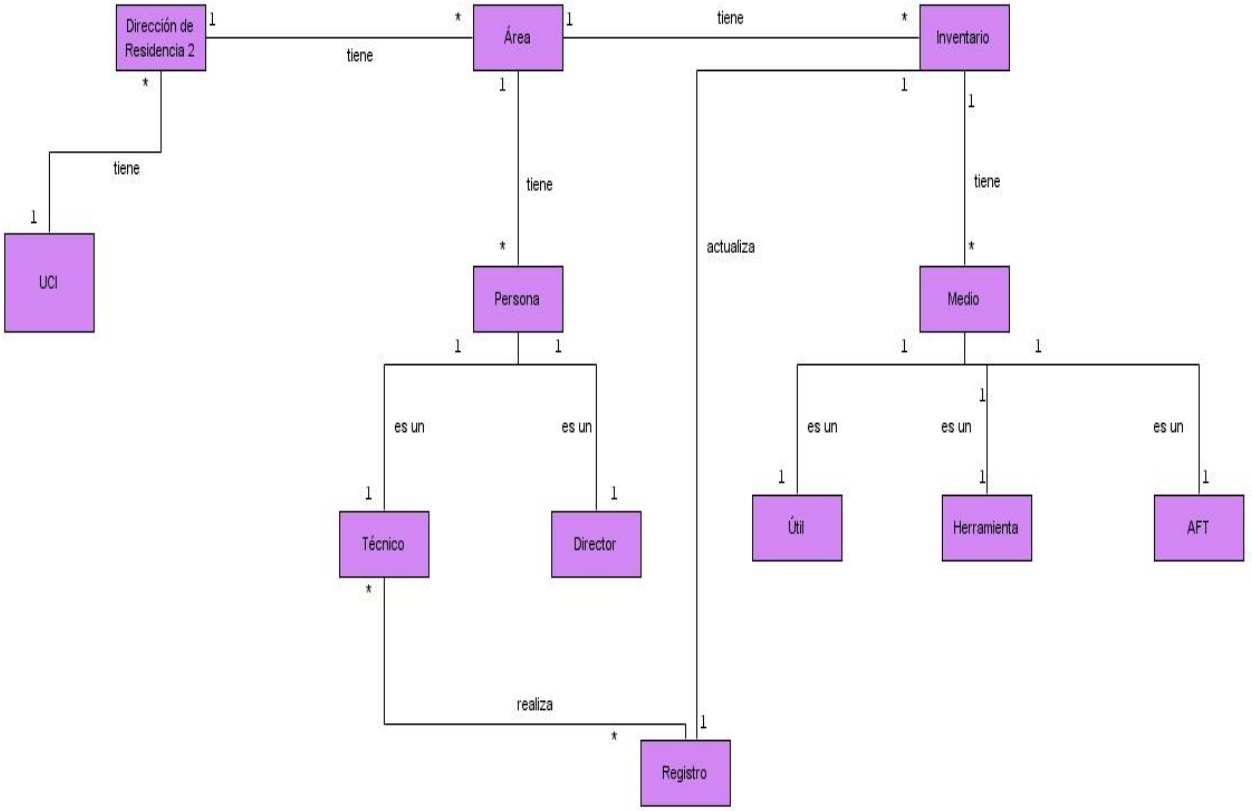


**Figura 4: Diagrama conceptual del negocio.**

## 2.6 Exploración

En esta fase, el cliente plantea a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto y dividirlas o agruparlas en otras más pequeñas o generales. Al mismo tiempo el equipo de desarrollo se familiariza con las Herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo.

Las estimaciones en esta fase son primarias y podrían variar cuando se analice más en detalle cada iteración, logrando como resultado una visión general del sistema y un plazo total estimado. La fase de exploración toma de pocas semanas a pocos meses, dependiendo de la complejidad de los artefactos y conocimientos que tengan los programadores de la tecnología.[34]

[34] Tomado de: Calabria y Píriz, 2003.

### 2.6.1 Historia de usuario

Una historia de usuario (HU) es una representación de un requisito escrito en una o dos frases, utilizando el lenguaje común del usuario. Las historias de usuario son utilizadas en las metodologías de desarrollo ágil para la especificación de requisitos de software. Cada HU debe ser limitada, esta debería poderse escribir sobre una pequeña nota adhesiva de papel.

Dentro de la metodología XP las historias de usuario deben ser escritas por los clientes en tarjetas de papel en las cuales se describen las características del sistema. Las HU son una forma rápida de administrar los requisitos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos. Las historias de usuario permiten responder rápidamente a los requisitos cambiantes.(21)

### 2.6.2 Historias de usuario de la propuesta de solución

Existen varias maneras de representar las HU, porque no existe un estándar o plantilla para la realización de las mismas. En este caso se describen las HU con el nombre, el número, el usuario que realiza la acción dentro del sistema, la estimación del tiempo de desarrollo, el nivel de prioridad en el negocio, el riesgo en caso de no realizarce, la iteración en la que será implementada, una breve descripción y un apartado para agregar alguna observación si es necesario.

**Tabla 3: HU- Gestionar Activos Fijo Tangibles**

| Historia de usuario | |
|---|---|
| **Número:** 2 | **Nombre:** Gestionar Activos Fijos Tangibles |
| **Usuario:** Director, Especialista | |
| **Prioridad:** Alta | **Riesgo:** Alto |
| **Estimación:** 4 días | **Iteración:** 1 |
| **Descripción:** El sistema permitirá crear un AFT, modificar un AFT, ver las características de un AFT y eliminar un AFT. | |
| **Observación:** El usuario deberá estar autenticado con el rol Director o Especialista | |

Existen un total de 21 HU descritas por el cliente, las restantes HU se encuentran en el Anexo 1.

## 2.7 Planificación

En esta fase el cliente establece la prioridad de cada historia de usuario, y en correspondencia, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días (61). A continuación, se muestran artefactos muy importantes generados por la Planificación como son el Plan de Iteraciones y el Plan de Entregas.

### 2.7.1 Plan de entregas

**Tabla 4: Plan de entregas**

| Entregable | Primera entrega: 10 de febrero | Segunda entrega: 10 de marzo | Tercera entrega: 10 de abril | Cuarta entrega: 10 de mayo |
|---|---|---|---|---|
| **Sistema para la gestión de los Activos Fijos Tangibles, Útiles Herramientas de la residencia** | Versión 0.1 | Versión 0.2 | Versión 0.3 | Versión 0.4 |

**Tabla 5: Funcionalidades disponibles por entrega del producto.**

| Historia de Usuario | 1era entrega | 2da entrega | 3era entrega | 4ta entrega |
|---|---|---|---|---|
| HU1. Gestionar Usuarios, Roles y Permisos | X | X | X | X |
| HU2. Gestionar Activos Fijos Tangibles | X | X | X | X |

| | | | | |
|---|---|---|---|---|
| HU3. Gestionar Útiles | X | X | X | X |
| HU4. Gestionar Herramientas | X | X | X | X |
| HU5. Gestionar entidades | X | X | X | X |
| HU6. Gestionar área | | X | X | X |
| HU7. Gestionar tipos de movimientos | | X | X | X |
| HU8. Gestionar cargos | | X | X | X |
| HU9. Gestionar personas | | X | X | X |
| HU10. Importar AFT | | X | X | X |
| HU11. Importar Útiles y Herramientas | | | X | X |
| HU12. Mostrar reporte de activos fijos | | | X | X |
| HU13. Mostrar acta de responsabilidad material | | | X | X |
| HU14. Mostrar el submayor de AFT | | | X | X |
| HU15. Mostrar el submayor de Útiles y Herramientas | | | X | X |
| HU16. Gestionar el movimiento de un medio básico | | | X | X |
| HU17. Mostrar el modelo de | | | | X |

| | | | | |
|---|---|---|---|---|
| movimiento de medios básicos | | | | |
| HU18. Mostrar el listado de movimiento de medios básicos pendientes | | | | X |
| HU19. Mostrar las trazas del movimiento de un medio básico | | | | X |
| HU20. Mostrar el listado de movimientos realizados | | | | X |
| HU21. Generar la planificación anual del control de los medios básicos | | | | X |

### 2.7.2 Iteraciones

En la metodología XP las HU se agrupan en iteraciones que oscilan de 3 a 4 semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no siempre es posible ya que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio). La cantidad de iteraciones depende del número de HU, son agrupadas por orden de prioridad e importancia para el negocio, de tal manera que las HU más importantes se realicen en las primeras iteraciones.

Todo el trabajo de una iteración es expresado en tareas de ingenierías, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores. El tiempo de realización de cada HU debe ser obligatoriamente en la iteración asignada para no afectar el orden del Plan de Entrega y la Planificación de las iteraciones.(62)

**Tabla 6: Planificación de las iteraciones.**

| Iteraciones | Orden de las HU a implementar | Cantidad de tiempo de trabajo |
|---|---|---|
| **Iteración 1** | HU1. Gestionar Usuarios, Roles y Permisos<br>HU2. Gestionar Activos Fijos Tangibles<br>HU3. Gestionar Útiles<br>HU4. Gestionar Herramientas<br>HU5. Gestionar entidades | 3 semanas |
| **Iteración 2** | HU6. Gestionar área<br>HU7. Gestionar tipos de movimientos<br>HU8. Gestionar cargos<br>HU9. Gestionar personas<br>HU10. Importar AFT | 3 semanas |
| **Iteración 3** | HU11. Importar Útiles y Herramientas<br>HU12. Mostrar reporte de activos fijos<br>HU13. Mostrar acta de responsabilidad material<br>HU14. Mostrar el submayor de AFT<br>HU15. Mostrar el submayor de Útiles y Herramientas<br>HU16. Gestionar el movimiento de un medio básico | 3 semanas |
| **Iteración 4** | HU17. Mostrar el modelo de movimiento de medios básicos<br>HU18. Mostrar el listado de movimiento de medios básicos pendientes | 3 semanas |

| | | |
|---|---|---|
| | HU19. Mostrar las trazas del movimiento de un medio básico<br><br>HU20. Mostrar el listado de movimientos realizados<br><br>HU21. Generar la planificación anual del control de los medios básicos | |

## 2.8 Conclusiones del Capítulo 2

- Se estableció el diálogo y entendimiento entre el cliente y los desarrolladores, donde surgen las HU necesarias para satisfacer todas las funcionalidades del sistema.
- Las HU se ordenaron en un número necesario de iteraciones colocados en un Plan de Entrega para una mejor organización en la implementación del sistema.
- Con los artefactos generados en este capítulo se tiene la información necesaria para implementar el sistema de gestión de información de los AFT, Útiles y Herramientas de la Dirección de Residencia de la UCI.

# Capítulo 3 Implementación y Pruebas

## 3.1 Descripción de la arquitectura

La construcción de aplicaciones web mediante el uso de framework constituye una buena práctica, debido a que reducen el tiempo de desarrollo y hace mantenible el código. Dentro de la amplia gama de framework web se utiliza como una solución recurrente el patrón de diseño Modelo-Vista-Controlador (MVC). Symfony no es la excepción y aunque su creador Fabien Potencier plantee que solo garantiza los elementos del controlador y la vista, se puede integrar fácilmente un ORM para la gestión de esta capa. El patrón de diseño MVC organiza el código en base a su función. Est e patrón separa el código en tres capas:

1. El modelo: representa la información relacionada con el dominio de la aplicación, es decir, su lógica del negocio. Abstrae la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes del tipo de gestor de base de datos utilizado. Para el componente una de las tareas más comunes es la persistencia y lectura de la base de datos. En la solución se utiliza Doctrine, una biblioteca cuyo objetivo es brindar poderosas Herramientas para el acceso a una base de datos relacional. El ORM de Doctrine permite asociar objetos a una base de datos tal como MySQL, PostgreSQL o Microsoft SQL. El componente gracias a las funcionalidades de Doctrine, puede persistir y recuperar objetos completos de la base de datos. Esto funciona asociando la clase TbActivoFijo.php a la tabla tb_activo_fijo de la base de datos y las propiedades de esa clase PHP a las columnas de la tabla. Además del uso de un repositorio ActivosFijosRepositorio.php asociado a la entidad que contiene las consultas a la base de datos.
2. La vista: transforma el modelo en interfaces de usuario permitiendo la interacción con el sistema. Solo se encarga de mostrar información. Twig es un motor y lenguaje de plantillas para PHP muy rápido y eficiente. Symfony 2 recomienda utilizar Twig para crear las plantillas de la aplicación. La sintaxis de Twig se ha diseñado para que las plantillas sean concisas y muy fáciles de leer y de escribir. En el componente se utiliza la herencia a tres niveles para reutilizar el máximo código posible. En el primer nivel se encuentra una sola plantilla base de la aplicación llamada base.html. twig, en el segundo nivel una plantilla para la administración del componente llamada layout.html.twig y en el

tercer nivel, plantillas específicas para cada parte del componente por ejemplo index.html.twig para listar los usuarios.

3. El controlador: se encarga de procesar las peticiones realizadas desde el navegador realizando los cambios necesarios en el modelo o en la vista. Este se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comando, etc.). En el componente los controladores son funciones php que toman información de la petición HTTP y construyen una respuesta HTTP como un objeto de Symfony 2. Los controladores tienen la lógica que el componente necesita para reproducir el contenido de las páginas. La clase controladora TbActivoFijoController.php tiene varias acciones, por ejemplo, indexAction que se encarga de mostrar todos los activos fijos de la base de datos. (63)

En la siguiente figura se ejemplifica de manera resumida el funcionamiento de la arquitectura MVC:

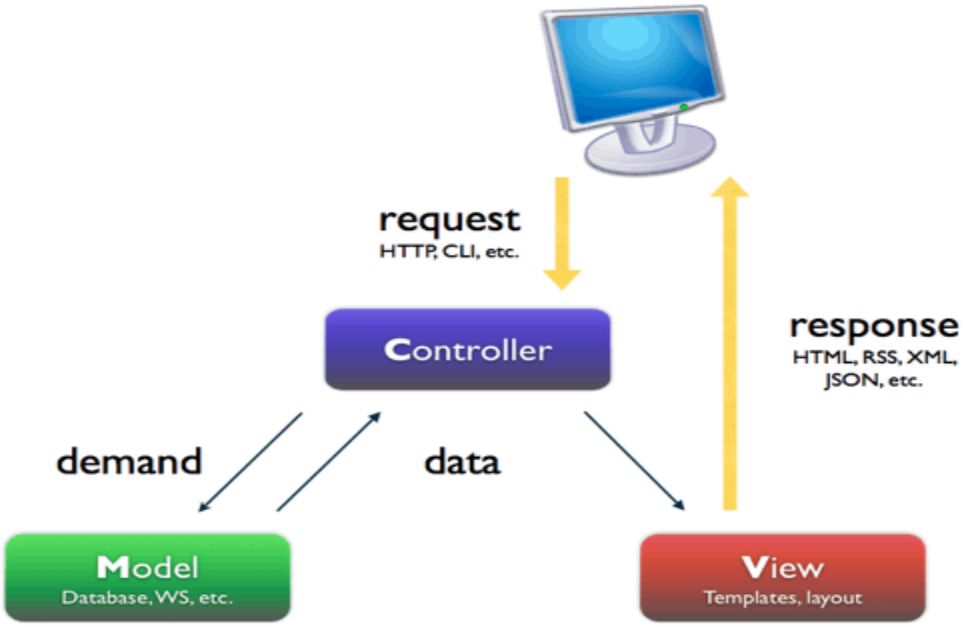


**Figura 5: Arquitectura MVC**

Cuando el usuario solicita ver una página de la herramienta, internamente sucede lo siguiente:

1. El sistema de enrutamiento determina qué controlador está asociado con la página solicitada. Symfony 2 ejecuta el controlador asociado. Un controlador no es más que una clase PHP en la que puedes ejecutar cualquier código que quieras
2. El controlador solicita al modelo los datos necesarios para mostrar. El modelo no es más que una clase PHP especializada en obtener información, normalmente de una base de datos
3. Con los datos devueltos por el modelo, el controlador solicita a la vista que cree una página mediante una plantilla y que inserte los datos del modelo
4. El controlador entrega al servidor la página creada por la vista

La siguiente figura representa de manera resumida el funcionamiento de MVC en Symfony 2:

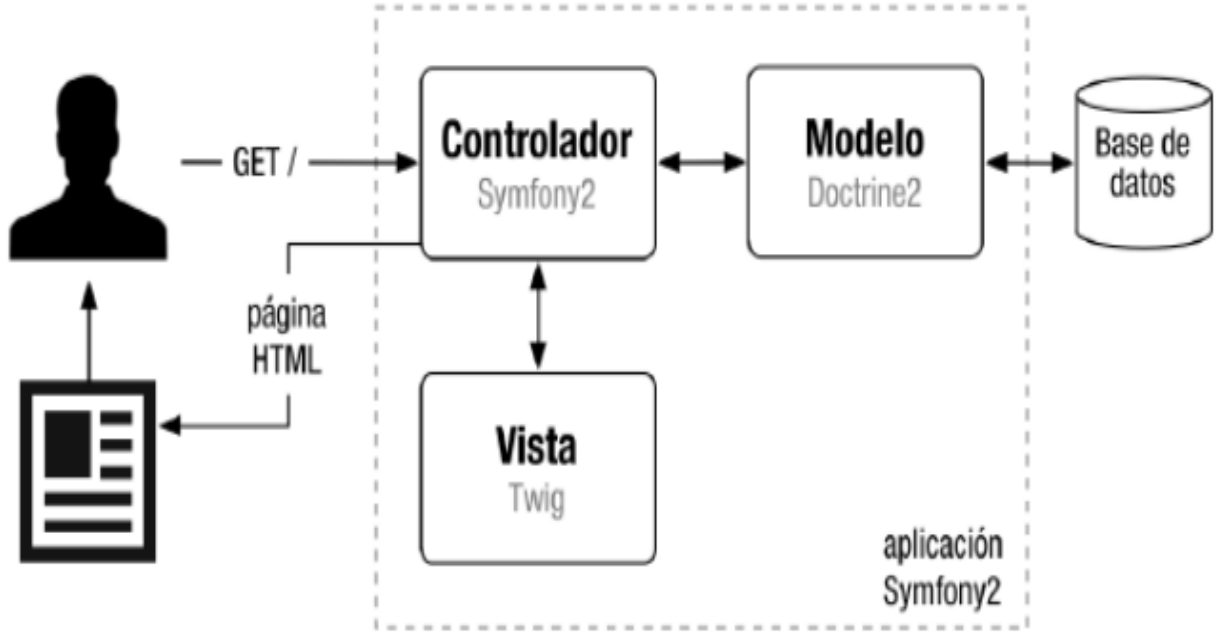


**Figura 6: Patrón arquitectónico Modelo-Vista-Controlador en Symfony 2**

## 3.2 Tarjetas Clase-Responsabilidades-Colaboradores (CRC)

Las tarjetas CRC (Clase-Responsabilidad-Colaboración) son una herramienta de brainstorming[35], usada como metodología para el diseño de software orientado a objetos,

[35] En español: Tormenta de ideas, es una Herramienta de trabajo grupal que facilita el surgimiento de nuevas ideas sobre un tema o problema determinado.

creada por Kent Beck y Ward Cunningham. En estas tarjetas se definen las responsabilidades de la clase, que no son más que su propósito y también se representan las dependencias o colaboraciones de otras clases para realizar sus responsabilidades. A continuación, se representa el formato de plantilla a utilizar para generar este artefacto, el cual representa una clase denominada Clase1 que posee tres responsabilidades llamadas funcion1, funcion2 y función3. La responsabilidad funcion1 necesita para su ejecución un parámetro denominado argumento1 y depende de las clases Clase2 y Clase3 para describir su funcionamiento.

**Tabla 7: Tarjeta CRC Clase1**

| Nombre de la clase: Clase1 | |
|---|---|
| **Responsabilidades** | **Colaboraciones** |
| Funcion1(argumento1) | Clase2, Clase3 |
| Funcion2() | Clase2 |
| Funcion3(argumento2) | Clase2, Clase3 |

El framework Symfony es capaz de generar automáticamente una el CRUD [36] para un determinado modelo de datos, con las características básicas de manipulación (listar, crear, modificar y eliminar), a través de la tarea automatizada mediante la línea de comando doctrine: doctrine-generate-crud. A continuación, se muestran las tarjetas CRC correspondientes a la propuesta de solución:

**Tabla 8 Tarjeta CRC: TbActivoFijoController**

| Nombre de la clase: TbActivoFijoController | |
|---|---|
| **Responsabilidades** | **Colaboraciones** |
| indexAction() | TbActivoFijo |
| newAction() | TbActivoFijoType |
| showAction($entity) | TbActivoFijo |
| editAction($entity) | TbActivoFijoType |
| deleteAction($entity) | TbActivoFijo |
| trazasAction($rotulo) | TbActivoFijo |

Existen un total de 16 tarjetas CRC descritas por el equipo de desarrollo. En el capítulo se muestra un ejemplo, las otras tarjetas CRC se encuentran en el Anexo 4.

[36] Es el acrónimo de Crear, Leer, Actualizar y Borrar.

## 3.3 Diagrama Entidad Relación (DER)

A continuación, se representa el DER, el cual representa el modelo físico de la base de datos del sistema. Este artefacto fue generado por la herramienta CASE[37] Visual Paradingm for UML elegida por el equipo de desarrollo.

[37] Del inglés: Computer Aided Software Engineering, en español Ingeniería de Software Asistida por Computadora.

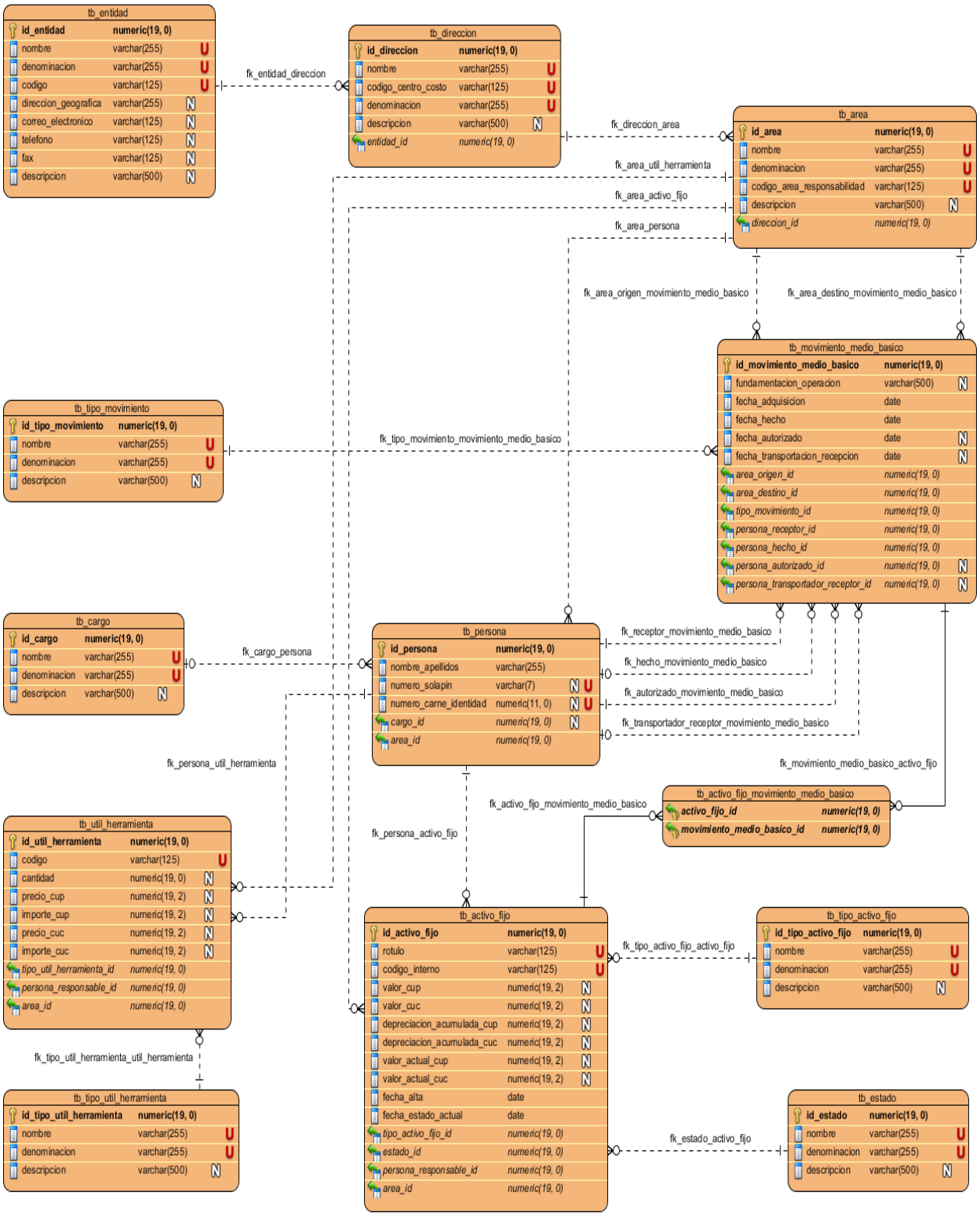


**Figura 7: Diagrama Entidad Relación**

Cada tabla de la base de datos tiene un objetivo específico que la aplicación utiliza para hacer persistir la información que gestiona. A continuación, se describe cada una de ellas:

**tb_area:** La tabla destinada a almacenar la información de las áreas que conforman las direcciones.

**tb_direccion:** La tabla destinada a almacenar la información de las direcciones que conforman las entidades.

**tb_entidad:** La tabla destinada a almacenar la información de las entidades.

**tb_movimiento_medio_basico:** La tabla destinada a almacenar la información de los movimientos de medios básicos.

**tb_tipo_movimiento:** La tabla destinada a almacenar la información de los tipos de movimientos de medios básicos.

**tb_persona:** La tabla destinada a almacenar la información de las personas.

**tb_cargo:** La tabla destinada a almacenar la información de los tipos de movimientos de medios básicos.

**tb_util_herramienta:** La tabla destinada a almacenar la información de los Útiles y Herramientas.

**tb_tipo_util_herramienta:** La tabla destinada a almacenar los tipos de Útiles y Herramientas.

**tb_activo_fijo:** La tabla destinada a almacenar la información de los activos fijos tangibles.

**tb_activo_fijo_movimiento_medio_basico:** La tabla encargada de almacenar las relaciones existentes entre los movimientos de medios básicos y los activos fijos (Un medio puede ser movido en varias ocasiones y un movimiento puede contener varios medios).

**tb_tipo_activo_fijo:** La tabla destinada a almacenar la información de los tipos de activos fijos tangibles.

**tb_estado:** La tabla destinada a almacenar la información de los estados por los cuales puede pasar un activo fijo tangible.

## 3.4 Diagrama de Despliegue

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

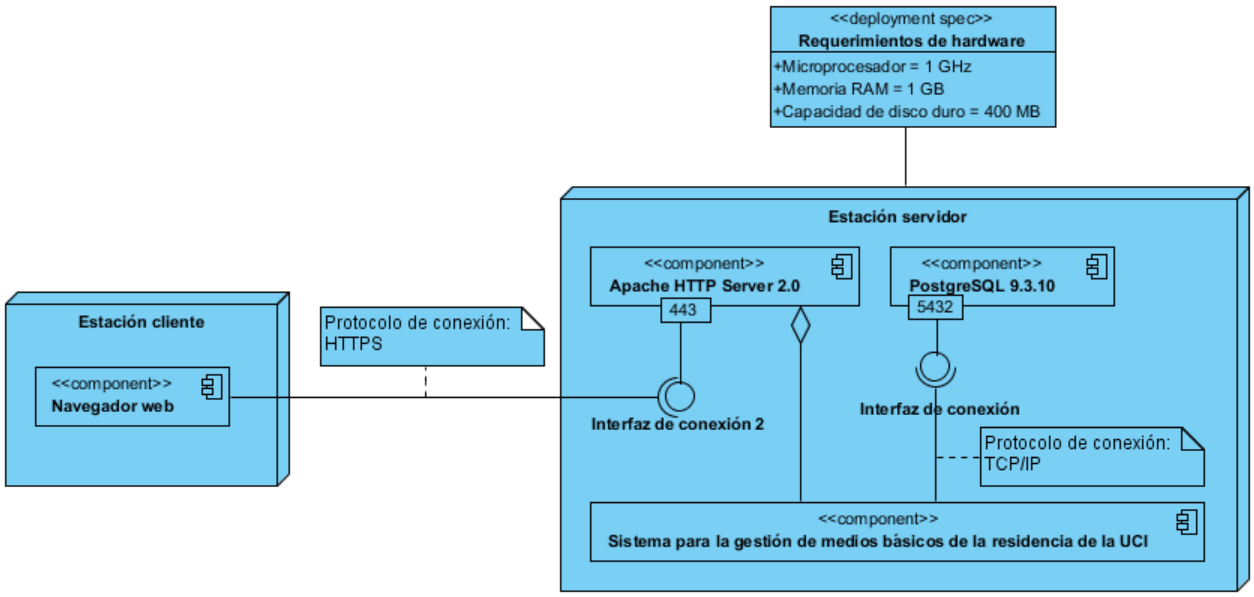


**Figura 8: Diagrama de Despliegue**

## 3.5 Patrones de diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características, una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores, otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias (64). Para el desarrollo de la propuesta de solución que se describe en la investigación solo fueron usados los implementados por el framework, por tal motivo serán los descritos a continuación.

### 3.5.1 Patrones GRASP[38]

1. Experto: Este patrón resuelve el problema de "asignar una responsabilidad al experto en información -la clase que tiene la información necesaria para realizar la responsabilidad -" (63). Es utilizado en la capa de abstracción del modelo de datos. Con el uso del ORM Doctrine, Symfony genera automáticamente las clases que representan las entidades del modelo de datos. Asociado a cada una de estas clases son generadas un conjunto de funcionalidades que las relacionan de forma directa con la entidad que representan. Estas clases contienen toda la información necesaria de la tabla que representan en la base de datos (67)
2. Creador: Este patrón resuelve el problema de asignar responsabilidades relacionadas con la creación de objetos. Es utilizado en los controladores, en ellos se encuentran las acciones definidas para el sistema. En la implementación de las acciones se crean instancias de las clases del modelo y de los formularios que representan a estas clases.[53] En Symfony 2 en la clase Controller.php se definen y ejecutan las acciones. En las acciones se crean los objetos de las clases que representan las entidades, evidenciando de este modo que la clase Controller.php es creador de dichas entidades
3. Alta cohesión: Este patrón resuelve el problema de "asignar una responsabilidad de manera que la cohesión permanezca alta" (63). La cohesión es una medida de la fuerza con que se relacionan y del grado de focalización de las responsabilidades de un elemento[39]. Un elemento con responsabilidades altamente relacionadas, y que no hace una gran cantidad de trabajo, tiene alta cohesión (63). La alta cohesión se evidencia en los controladores que poseen un conjunto de funcionalidades, existiendo estrecha relación entre algunas. Ejemplo de ello lo constituyen las acciones create y update que al crear o actualizar un objeto realizan las validaciones mediante la acción processForm (67)
4. Bajo acoplamiento: Este patrón resuelve el problema de "asignar una responsabilidad de manera que el acoplamiento permanezca bajo" (63). Un elemento con bajo (o débil) acoplamiento no depende de demasiados elementos (63). En la solución la clase DefaultController.php hereda de Controller.php, que es una clase estable en cuanto a su

[38] Del inglés General ResponsibilityAssignment Software Patterns, en español Patrones Generales de Software para Asignar Responsabilidades.
[39] Estos elementos pueden ser clases, subsistemas, etcétera.

implementación por lo que se obtiene un grado bajo de acoplamiento entre las clases. Además, las clases que implementan la lógica del negocio y de acceso a datos no tienen asociaciones con las de la vista o el controlador, lo que proporciona que la dependencia entre las clases, en este caso, sea baja

5. Controlador: Este patrón resuelve el problema de asignar la responsabilidad de recibir o manejar un mensaje de evento del sistema a una clase. Un controlador sirve como intermediario entre una interfaz y la acción que se desee ejecutar. En Symfony 2 el patrón Controlador es utilizado en las clases de la capa del controlador del patrón MVC, como son: TbActivoFijoController.php y AdminController.php

### 3.5.2 Patrones GoF[40]

1. Fachada: Proporciona una interfaz unificada de alto nivel que, representando a todo un subsistema, facilite su uso. La "fachada" satisface a la mayoría de los clientes, sin ocultar las funciones de menor nivel a aquellos que necesiten acceder a ellas. En Symfony 2 se puede acceder al manejador de entidades de las formas siguientes: $this->getDoctrine()->getEntityManager() y $this->container->get(„doctrine")->getEntityManager() lo que evidencia el uso del patrón (67)

2. Decorador: Es un patrón de tipo estructura, ya que permite que clases y objetos sean utilizados para componer estructuras de mayor tamaño. Resuelve el problema de añadir responsabilidades adicionales a un objeto dinámicamente. En Symfony este patrón es utilizado en la capa Vista del patrón arquitectónico MVC. En el componente se utiliza el patrón Decorador para la vista, el layout decora el contenido de la plantilla (67)

3. Observador: Un efecto muy frecuente en aquellos sistemas que se fragmentan en un conjunto de clases que cooperan, es la necesidad de mantener la consistencia entre los distintos objetos interrelacionados. Para no recurrir a soluciones fuertemente acopladas (que reducen la posibilidad de reutilización), este patrón define una dependencia "uno-a-muchos" entre objetos, para que, cuando uno de ellos cambie su estado, los que dependan de él sean avisados y puedan actualizarse convenientemente. (67)

[40] Del inglés: Gang of Four, en español Banda de los Cuatro. Es el nombre con el que se conoce comúnmente a los autores del libro Design Patterns (en español Patrones de Diseño) (ISBN 0-201-63361-2), los mencionados autores son Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides.

## 3.6 Estilos y estándares de codificación.

En la realización del sistema es importante llevar un estándar de desarrollo para evitar el atraso en futuras actualizaciones. Se definen estándares de codificación porque un estilo de programación homogéneo en un proyecto permite que todos los participantes lo puedan entender en menos tiempo y que el código en consecuencia sea conservable. Puede ser que el estándar de documentación definido no sea perfecto para todas las situaciones. Si esto es así y se cree que es necesario salirse de la norma se documentan los motivos. En el diseño del software se usan los siguientes estilos de codificación.

Las variables locales se definen como:

1. Los nombres de algunas variables locales, como los iteradores o los contadores, se especifican en minúscula y de forma abreviada, siempre que su contexto sea específicamente local y su lectura sea intuitiva. Ejemplos: $cont, $i, $j
2. Al hacer asignaciones, existe un espacio a ambos lados del signo igual (=), esto funciona tanto para asignar un valor fijo, de otra variable o del resultado de una función
3. En el caso de un bloque de asignaciones relacionadas entre sí (por ejemplo, al inicializar un script), se alinean los signos (=) agregando espacios extra, para mejorar la legibilidad

La identación y largo de líneas se realizan con 4 espacios, sin tabulador, para que cualquier editor de texto reconozca correctamente la indentación. Por otro lado, si bien existen editores que realizan corte automático de línea, se hace en forma manual a los 75-80 caracteres.

Las llamadas a funciones se realizan sin espacio entre el nombre de la función, el paréntesis de apertura y el primer parámetro. En caso de varios parámetros, se separa con espacios entre la coma y cada parámetro, y sin espacios entre el último parámetro, el paréntesis de cierre y el punto y coma.

La definición de funciones se caracteriza:

1. El nombre es lo más descriptivo posible
2. Se evita el uso de abreviaturas
3. Se utiliza la convención lowerCamelCase

La definición de clases se realiza con:

1. El nombre es descriptivo, evitando abreviaturas
2. La llave de inicio de la clase se coloca en la línea siguiente, indentada correctamente
3. Todos los miembros de la clase son privados, es decir, únicamente accesibles a través de métodos de la misma

Las etiquetas de bloque PHP se utilizan <? php y?> para iniciar y terminar un bloque de código PHP, no las variantes <? y?> o <% y %>.

## 3.7 Tareas de ingeniería

Cada HU se divide en tareas de ingeniería según la metodología XP, para una mejor organización en la distribución del desarrollo del sistema. Estas tareas tienen un rango de días para realizarse, de tal manera que cumplan con el plan de cada iteración. A continuación, se muestra la tarea de ingeniería Adicionar AFT como ejemplo:

**Tabla 9: Tarea #5**

| Tarea | |
|---|---|
| **Número:** 5 | **Número de HU:** 2 |
| **Nombre:** Adicionar AFT | |
| **Tipo de tarea:** Desarrollo | **Estimación:** 1 día |
| **Fecha inicio:**17 de Enero de 2014 | **Fecha fin:** 18 de Enero de 2014 |
| **Programador responsable:** Yandriel Padrón Mojena, Luis Daniel Gonzalez Díaz | |
| **Descripción:** Adicionar un AFT al sistema. | |

Fueron descritas un total de 62 tareas de ingeniería. El resto de las tareas se encuentran en el Anexo 2.

## 3.8 Pruebas

Uno de los pilares de la metodología XP es el uso de pruebas para comprobar el funcionamiento de los códigos que se vayan implementando. Es necesario la realización de un sin números de pruebas a lo largo de todo el desarrollo del software, con el fin de lograr un producto con calidad, reduciendo el número de errores y disminuyendo el tiempo transcurrido entre la aparición de un error y su corrección (46). Existen dos grupos de pruebas en XP:

pruebas unitarias encargadas de verificar el código y pruebas de aceptación que están orientadas a probar las funcionalidades del sistema.

### 3.8.1 Desarrollo dirigido por pruebas

El desarrollo dirigido por pruebas (TDD[41]) es una característica de la metodología XP para comprobar el funcionamiento de los códigos que se van implementando. Este estilo sigue el principio “test-first”, significa que las pruebas se crean antes de comenzar la implementación. La idea es que, al pensar en cómo se probará la funcionalidad, se está pensando en la propia funcionalidad desde el punto de vista de su interfaz (qué métodos tendrá y con qué parámetros), ayudando a desarrollar así un mejor diseño. De esta manera se asegura de que no exista ninguna funcionalidad que no esté probada. Una vez creadas las pruebas, el código no pasa a producción hasta que el resultado de las mismas no sea satisfactorio. Luego se inicia el proceso de refactorización que consiste en limpiar y organizar el código, adaptarlo a los patrones y aumentar su legibilidad, todo esto sin modificar su comportamiento externo (46).

### 3.8.2 Pruebas unitarias

Las pruebas unitarias revisan dentro de las clases de la aplicación todos sus métodos y comprueban su funcionamiento. Esto sirve para asegurar que cada uno de las funcionalidades funcione correctamente por separado. A lo largo de cada iteración se realizaron varias pruebas unitarias desde el comienzo de la aplicación hasta su fin.

A continuación, se muestra el directorio donde se encuentran todas las clases en las que fueron implementadas las pruebas unitarias, creadas para probar el código de la aplicación.

41 Del inglés: Test Driven Development, en español Desarrollo Basado en Pruebas.

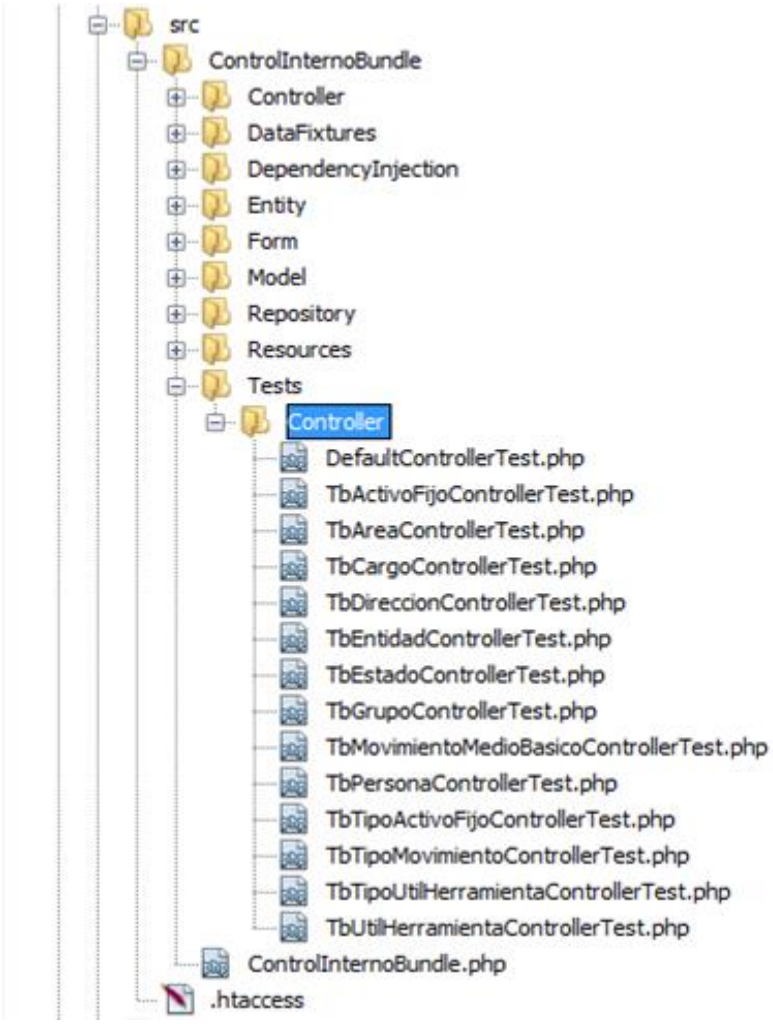


**Figura 8: Directorio de pruebas unitarias en Symfony 2**

### 3.8.3 Pruebas de aceptación

Las pruebas de aceptación son las descripciones del resultado esperado según las funcionalidades de las HU del sistema. El equipo de desarrollo se reúne junto al cliente para analizar las respuestas de las pruebas realizadas y determinar decisiones acerca de los resultados. Es recomendable publicar los resultados de las pruebas de aceptación, para que todo el equipo de desarrollo esté al tanto de esta información.

A continuación, aparecen los casos de pruebas de aceptación referentes a la propuesta de solución:

**Tabla 10: Caso de prueba de aceptación #5**

| Caso de prueba de aceptación | |
|---|---|
| **Código:***HU2_CP5* | **Historia de usuario:** *2* |
| **Nombre:** *Adicionar AFT* | |
| **Condiciones de ejecución:** *El usuario debe estar autenticado con el rol Director o Especialista.* | |
| **Entrada/ Pasos de ejecución:**<br>*Entrada: La entrada consiste en los datos del AFT que se deben introducir para ser insertado en la base de datos.*<br>*Pasos de ejecución: Para adicionar un AFT ir al menú y seleccionar la opción "Activos Fijos". En la ventana que se va a mostrar seleccionar la opción "Crear activo fijo". Una vez introducido los datos, se selecciona la opción "Crear".* | |
| **Resultado esperado:** *El AFT es adicionado correctamente.* | |
| **Evaluación de la prueba:** *Prueba satisfactoria.* | |

Fueron elaborados un total de 62 casos de prueba, para realizar las pruebas de aceptación. Solo se muestra un ejemplo para entender el confeccionamiento. Las restantes se encuentran en el Anexo 3.

## 3.9 Análisis de los resultados de las pruebas

Al concluir cada una de las iteraciones planificadas para el desarrollo de la propuesta de solución, fueron realizadas las pruebas pertinentes para realizar las entregas pactadas con el cliente, la Tabla 12 muestra el resultado de dichas pruebas en cada una de las iteraciones.

**Tabla 11: No conformidades por iteraciones**

| | Iteración 1 | Iteración 2 | Iteración 3 | Iteración 4 |
|---|---|---|---|---|
| **Cantidad** | 16 | 13 | 8 | 0 |

Todas las no conformidades fueron resueltas, lo cual valida en cierto grado la calidad de la propuesta de solución, logrando una mayor satisfacción por parte del cliente.

## 3.10 Conclusiones del capítulo 3

- Se generaron artefactos de gran importancia, como son el diagrama de despliegue, las tarjetas CRC y el diagrama ER.
- Después de un estudio de los patrones de diseño implementados por el framework Symfony se escogieron los necesarios para el desarrollo de la aplicación.

- Se explican los estilos y estándares más significativos utilizados durante el proceso de codificación y las tareas de ingeniería necesarias para desarrollar cada historia de usuario.
- Se plantea las pruebas de aceptación las cuales permiten poner en manos del cliente un software con menor probabilidad de fallo.

# Conclusiones y Recomendaciones

## Conclusiones

- En el proceso de gestión de la información de los Activos Fijos Tangibles, Útiles y Herramientas en la Residencia de la Universidad de las Ciencias Informáticas se diagnosticó que existía una gran cantidad de datos que se operacionalizaban de forma manual con sus consecuentes inconvenientes, que los recursos humanos involucrados en este proceso reconocen lo complejo de la manualidad de este proceso y de la necesidad e importancia que les podía traer poderlo informatizar.
- Se obtuvo un conjunto de datos necesarios para el desarrollo de la aplicación, que en su incorporación al sistema y su manipulación contribuyeron a demostrar la necesidad e importancia el desarrollo del mismo.
- Durante el diseño del sistema informático, como propuesta de solución, se estableció la primera conexión entre los desarrolladores y el cliente conformando así las fortalezas y debilidades de la aplicación, se alcanzó un buen diálogo y entendimiento entre el cliente y los desarrolladores, surgiendo así el diagrama Conceptual del negocio y las HU necesarias para satisfacer todas las funcionalidades del sistema, las que una vez ordenadas y colocadas en un Plan de entrega permitió una mejor organización en la implementación del sistema.
- Para la implementación del sistema informático y el cumplimiento al objetivo principal de mejorar dicho proceso de gestión; se generaron artefactos de gran importancia, como son las tarjetas CRC, el diagrama de Despliegue y el diagrama ER.
- La validación del desarrollo dirigido por pruebas demostró la calidad del sistema informático con las pruebas unitarias y las 62 pruebas de aceptación, las cuales mejoraron el cumplimiento del objetivo general con el análisis de los resultados de dichas pruebas.

## Recomendaciones

- Desarrollar una versión que aproveche las principales tendencias del desarrollo de software actual, como son el desarrollo de software para dispositivos móviles y la computación en la nube.

## *Conclusiones generales y Recomendaciones*

- Se continúe el estudio y análisis en otras áreas de la institución universitaria con vistas a la adecuación e implementación del sistema informático que se propone, por ser un proceso que está presente en cada una de ellas.
- A partir de la propuesta de solución que se muestran en esta investigación, que otros profesionales estudien la adaptabilidad del sistema informático presentado.

## Referencias bibliográficas

1. Belohlavek, Peter. ¿Cómo escribir una fundamentación? - Ensayos - Cloinv. [online]. [Accessed 4 February 2016]. 2005 Available from: http://www.buenastareas.com/ensayos/C%C3%B3mo-Escribir-Una-Fundamentaci%C3%B3n/3748394.html

2. CASTILLO, L. M. H. Ingeniería del Software, 2006. [Disponible en: http://www.monografias.com/trabajos34/ingenieria-software/ingenieria-software.shtml#lenguaje

3. CONTROL DE LOS ACTIVOS FIJOS TANGIBLES EN CUBA. [online]. [Accessed 8 February 2016]. Available from: http://www.eumed.net/cursecon/ecolat/cu/2012/cbg.html

4. Activo fijo tangible | Economía Simple. [online]. [Accessed 8 February 2016]. Available from: http://www.economiasimple.net/activo-fijo-tangible.html

5. Jeremy Bentham, An Introduction to the Principles of Morals and Legislation | Library of Economics and Liberty. [online]. [Accessed 21 January 2016]. Available from: http://www.econlib.org/library/Bentham/bnthPML.html

6. Útiles básicos - ECyT-ar. [online]. [Accessed 21 January 2016]. Available from: http://cyt-ar.com.ar/cyt-ar/index.php/%C3%9Atiles_b%C3%A1sicos

7. Concepto de información — Definicion.de. Definición.de [online]. [Accessed 21 January 2016]. Available from: http://definicion.de/informacion/

8. Gestión de Información en las Organizaciones. Ciudad de La Habana: s.n., 2007. p. 135.

9. Sistema de Control de Bienes Web-UG [online]. Available from: https://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwjE8o3Q1_zKAhUJVz4KHfwnDsgQFggaMAA&url=http%3A%2F%2Fwww.siia.ugto.mx%2Fmanual_bienes_web.pdf&usg=AFQjCNEBqj3vd04yBvjN57goSVvW4TBing&bvm=bv.114195076,d.cWw&cad=rja

10. Universitas XXI-Económico. [online]. [Accessed 16 February 2016]. Available from: http://servicio.us.es/sorolla/

11. FUNDACITE-Mérida. [online]. [Accessed 16 February 2016]. Available from: http://www.fundacite-merida.gob.ve/

12. Desoft. [online]. [Accessed 16 February 2016]. Available from: http://www.desoft.cu/

13. Sistema de Inventario del Patrimonio Cultural y Natural [online]. Available from: http://www.oei.es/cultura2/cuba/07.htm

14. Portada. [online]. [Accessed 16 February 2016]. Available from: http://cujae.edu.cu/

15. ASSETS: sistema de Gestión Integral. [online]. [Accessed 16 February 2016]. Available from: http://www.assets.co.cu/

16. LISANDRO DÍAZ DELGADO. Aplicación Web para la gestión de la información de los Activos Fijos Tangibles y Útiles de la Residencia Universitaria de la Universidad de las Ciencias Informáticas. Ciudad de La Habana: Universidad de las Ciencias Informáticas, 2008.

17. Metodologias de desarrollo. [online]. 03:04:28 UTC. [Accessed 21 January 2016]. Available from: http://es.slideshare.net/MeneRomero/metodologias-de-desarrollo

18. Estilos de codificación (Carlos Fontela) | CyS Ingeniería de Software. [online]. [Accessed 21 January 2016]. Available from: https://cysingsoft.wordpress.com/2009/06/12/estilos-de-codificacion-carlos-fontela/

19. PRESSMAN, Roger S. Ingeniería del software: un enfoque práctico. Séptima Edición. Mcgraw Hill/Interamericana Editores, 2005. ISBN 9789701054734.

20. Programación extrema. [online]. [Accessed 16 February 2016]. Available from: http://www.chuidiang.com/ood/metodologia/extrema.php

21. Extreme Programming: A Gentle Introduction. [online]. [Accessed 16 February 2016]. Available from: http://www.extremeprogramming.org/

22. Desarrollo Web. Lenguajes de lado servidor. Sitio Web de Desarrollo web. [Online] [Cited: mayo 17, 2007.] http://www.desarrolloweb.com/articulos/243.php.

23. Ventajas y Desventajas de Java Desarrollo de Aplicaciones. Storify [online]. [Accessed 13 January 2016]. Available from: http://storify.com/abhorrentp337/ventajas-y-desventajas-de-java-desarrollo-de-aplichAdvantages y Desventajas de Java

24. Framework - English-Spanish Dictionary - WordReference.com. [online]. [Accessed 8 February 2016]. Available from: http://www.wordreference.com/es/translation.asp?tranword=frameworkframework - Translation to Spanish, pronunciation, and forum discussions

25. FINLAND, Invest in. Invest in Finland News (archive). Invest in Finland [online]. [Accessed 8 February 2016]. Available from: http://www.investinfinland.fi/articles/news/2News

26. Releases · vaadin/vaadin · GitHub. [online]. [Accessed 8 February 2016]. Available from: https://github.com/vaadin/vaadin/releases

27. CodeIgniter Web Framework. [online]. [Accessed 8 February 2016]. Available from: http://www.codeigniter.com/

28. Laravel - The PHP Framework For Web Artisans. [online]. [Accessed 8 February 2016]. Available from: https://laravel.com/

29. PHP Framework Laravel ESPAÑOL - Google+. [online]. [Accessed 8 February 2016]. Available from: https://plus.google.com/communities/111797011764886461382

30. Los mejores frameworks php para el 2014 - uno de piera. [online]. [Accessed 8 February 2016]. Available from: https://uno-de-piera.com/los-mejores-frameworks-php-para-el-2014/

31. The technological benefits of Symfony in 6 easy lessons. [online]. [Accessed 8 February 2016]. Available from: http://symfony.com/six-good-technical-reasons

32. Drupal 8 integra los primeros componentes de Symfony2. symfony.es [online]. [Accessed 8 February 2016]. Available from: http://symfony.es/noticias/2011/10/26/drupal-8-integra-los-primeros-componentes-de-symfony2/

33. Symfony.es, el mejor framework PHP para crear aplicaciones web. symfony.es [online]. [Accessed 8 February 2016]. Available from: http://symfony.es/

34. JQUERY.ORG, jquery Foundation-. jQuery. [online]. [Accessed 8 February 2016]. Available from: http://jquery.com/jQuery: The Write Less, Do More, JavaScript Library

35. Ejemplo de ABM usando Ajax - PHP - MySQL. [online]. [Accessed 8 February 2016]. Available from: http://ideaschile.cl/abm/

36. HUMANS.TXT. Gumby - A Flexible, Responsive CSS Framework - Powered by Sass. [online]. [Accessed 16 February 2016]. Available from: http://www.gumbyframework.comCreate rapid and logical page layout and app prototypes with Gumby Framework, a flexible, responsive CSS framework, powered by Sass

37. Kube CSS Framework. [online]. [Accessed 16 February 2016]. Available from: https://imperavi.com/kube/

38. YAML CSS Framework — for truly flexible, accessible and responsive websites. [online]. [Accessed 16 February 2016]. Available from: http://www.yaml.de/

39. Bootstrap en Español (Spanish). [online]. 9 November 2013. [Accessed 8 February 2016]. Available from: http://web.archive.org/web/20131109025010/http://www.oneskyapp.com/docs/bootstrap/es

40. Bootstrap. [online]. [Accessed 8 February 2016]. Available from: http://getbootstrap.com/2.3.2/

41. JavaScript. Mozilla Developer Network [online]. [Accessed 21 January 2016]. Available from: https://developer.mozilla.org/es/docs/Web/JavaScript

42. PHP: Hypertext Preprocessor. [online]. [Accessed 21 January 2016]. Available from: http://php.net/

43. W3C HTML. [online]. [Accessed 21 January 2016]. Available from: http://www.w3.org/html/

44. CSS Snapshot 2015. [online]. [Accessed 21 January 2016]. Available from: http://www.w3.org/TR/CSS/

45. Introduction — Flycheck 0.19-cvs. [online]. 9 May 2014. [Accessed 16 February 2016]. Available from: http://web.archive.org/web/20140509094506/http://flycheck.readthedocs.org/en/latest/manual/introduction.html

46. KOSKELA, LASSE**.** TEST DRIVEN, Practical TDD and Acceptance TDD for Java Developers. Manning Publication Co., 2008. ISBN 1-932394-85-0.

47. Eclipse - The Eclipse Foundation open source community website. [online]. [Accessed 16 February 2016]. Available from: http://www.eclipse.org/

48. Visual Studio - Herramientas para desarrolladores de Microsoft. [online]. [Accessed 16 February 2016]. Available from: https://www.visualstudio.com/es-es

49. Welcome to NetBeans. [online]. [Accessed 9 February 2016]. Available from: https://netbeans.org/

50. Planet NetBeans. [online]. [Accessed 9 February 2016]. Available from: http://www.planetnetbeans.org/xp

51. Software Design Tools for Agile Teams, with UML, BPMN and More. [online]. [Accessed 21 January 2016]. Available from: http://www.visual-paradigm.com/

52. Visual Paradigm. [online]. [Accessed 9 February 2016]. Available from: http://es.scribd.com/doc/65619842/Visual-Paradigm

53. ALBA Software -. [online]. [Accessed 16 February 2016]. Available from: http://www.albasoft.com/web/#/home

54. ASENSIO, Rafael Menéndez-Barzanallana. Artículos y enlaces de interés en informática. Universidad de Murcia. Profesor Rafael Barzanallana. [online]. 13 March 2011.

[Accessed 16 February 2016]. Available from: http://www.um.es/docencia/barzana/DIVULGACION/INFORMATICA/Rafael.

55. MySQL. [online]. [Accessed 16 February 2016]. Available from: http://www.mysql.com/

56. Oracle | Integrated Cloud Applications and Platform Services. [online]. [Accessed 16 February 2016]. Available from: http://www.oracle.com/index.html

57. PostgreSQL: The world's most advanced open source database. [online]. [Accessed 9 February 2016]. Available from: http://www.postgresql.org/

58. PostGreSQL: Ventajas y Desventajas. [online]. [Accessed 13 January 2016]. Available from: http://postgressql-adsi.blogspot.com/2011/11/ampliamente-popular-ideal-para.html

59. World Wide Web Consortium (W3C) - España. [online]. [Accessed 16 February 2016]. Available from: http://www.w3c.es/

60. Apache License, Version 2.0. [online]. [Accessed 9 February 2016]. Available from: http://www.apache.org/licenses/LICENSE-2.0

61. XP - Extreme Programing Ingenieria de Software. [online]. [Accessed 9 February 2016]. Available from: http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html

62. Funcionamiento | Metodología Ágil XP. [online]. [Accessed 21 January 2016]. Available from: https://aps2programacionxtrema.wordpress.com/funcionamiento/

63. Larman, Craig. Applying UML and patterns: an introduction to object-oriented analysis and design and iterative development. New Jersey: Prentice Hall PTR Upper Saddle River, 2004.

64. ADMIN. Desarrollador Senior: Patrón Simple Factory en PHP5. Desarrollador Senior [online]. [Accessed 21 April 2016]. Available from: http://desarrolladorsenior.blogspot.com/2009/09/patron-simple-factory-en-php5.html

65. Pressman, Roger. Ingeniería de software. Un enfoque práctico. 2002.

66. Rumbaugh, James, Jacobson, Ivar y Booch, Grady. El Proceso Unificado de Desarrollo de Software. Madrid: Addison-Wesley, 2000.

67. Potencier, Fabien y Zaninotto, Francois. Symfony, la guía definitiva. s.l.: Librosweb, 2008.