Master Oficial en Software Libre

Desarrollo de Aplicaciones en Software Libre

Sistema de Administración de Propiedades Proyecto de Software Libre Wakodi

Jorge Mauricio Arancibia Patzi

UOC Universitat Oberta de Catalunya
14 de Junio de 2010

Dedicado a mi Padre

Agradecimientos

A los profesores y compañeros de la UOC, gracias por compartir sus conocimientos.
A mi tutor de Proyecto Gregorio por todo el apoyo brindado.
A mi familia que sin su apoyo no habría sido posible realizarlo.

Resumen

Las nuevas tendencias tecnologías y sociales en un mundo cada vez más globalizado, están permitiendo un crecimiento de las soluciones basadas en Software Libre, donde el beneficio social se antepone al beneficio económico. Esto está demostrado con las herramientas y aplicaciones disponibles en la actualidad cuya calidad es igual o mayor a sus pares comerciales.

El éxito en el desarrollo de aplicaciones basadas en Software libre no solo depende de la solución propuesta sino también del grado de participación de la comunidad para apoyar al mismo.

En este sentido el presente documento plantea un proyecto de desarrollo de un Software de Administración de Propiedades bajo la filosofía del Software Libre como una necesidad a la falta de soluciones existentes. El aumento de los negocios en la administración de propiedades es una realidad y es imprescindible proporcionar una alternativa de solución bajo esta filosofía.

El trabajo se centra en proporcionar un solución informática haciendo uso de nuevas tecnologías y a su vez dotar de las herramientas necesarias para la construcción de una comunidad.

El desarrollo de la solución pasa por la descripción de las herramientas empleadas y las etapas para su construcción, que incluyen el análisis, diseño, implementación y posterior implantación. Se hace énfasis en el Framework elegido para demostrar las ventajas de su aplicación.

En la construcción del comunidad se describen las herramientas utilizadas para la difusión del proyecto que incluyen la publicación de una página del proyecto, uso de redes sociales y páginas publicitarias y la puesta en marcha de un software de colaboración para la administración del desarrollo del proyecto.

Índice general

Índice de figuras

Capítulo 1

Introducción

La tecnología de la información se ha convertido en una fuente de ventajas competitivas sostenibles y un arma estratégica en la mayoria de las empresas y en el sector de la administración de propiedades no es la excepción. El término "administración de propiedades" se refiere a la operación de administrar los bienes inmuebles comerciales, industriales y de servicios (hoteles, residenciales), el cual es un proceso que puede ser relacionado con la administración de cualquier tipo de negocio. El término "administración de propiedades" también es utilizado para describir la práctica del manejo activo fijos y la propiedad personal que no son necesariamente bienes inmuebles, como por ejemplo los equipos, maquinaria y herramientas.

Como una forma de reemplazar los métodos antiguos e ineficaces de administración es que aparecen los Sistemas de Administración de Propiedades llamados también Software PMS (Property Management System – PMS), que son un tipo particular de software que como su nombre indica facilitan la administración de bienes. Esta técnologia ha permitido mejoras en la operación y administración, incluso su uso llega a ser un requisito para poder lograr alianzas como la integración hacia cadenas de suministros o el desarrollo de sistemas de distribución y de comunicación con los clientes.

Un Software PMS controla básicamente las propiedades, arrendatarios, contabilidad, mantenimiento, seguros y propietarios. Este tipo de sistema provee la característica del mantenimiento completo de la descripción, segregación y clasificación de cada propiedad (inventario) relacionada con el negocio. Admite la creación de perfiles individuales de cada propiedad en el que no solo se incluyen información de dimensiones, si no información detallada de lo que contiene (color, puertas, ventanas, características, funcionalidades, etc.). Junto con el inventario físico, los Software PMS incluyen perfiles de los arrendatarios con información personal, reportes de créditos, disponibilidades, copias de correspondencias, hábitos de pago y cualquier otra información que el

propietario vea necesario. Algunos PMS tienen la capacidad de interactuar con cuentas bancarias permitiendo de esta forma transacciones automáticas. Estas transacciones pueden incluir pagos de rentas como también pagos hacia vendedores.

El ámbito de trabajo de los Software PMS son variados como por ejemplo en el sector de hotelería donde es utilizado para manejar listados de clientes, disponibilidad de habitaciones, reservaciones en línea, puntos de ventas, telefonía entre otros. En el sector de los gobiernos locales son utilizados para administrar propiedades pertenecientes al estado como por ejempo escuelas, centros de salud, casas sociales y las industrias. En el sector de la manufactura son utilizados para manejar, controlar y contabilizar la propiedad personal, que componen generalmente de equipos, herramientas y capital físico que son adquiridos y usados para construir, reparar y mantener productos terminados.

1.1. Motivación

El uso de tecnologías de información y en concreto el uso del software en el mundo empresarial permite un mejor y mayor control sobre todos los procedimientos necesarios para la obtención de beneficios. El desarrollo de las empresas está muy relacionado con la capacidad de alcanzar niveles de competitividad mayores, lo que es facilitado mediante la adopción de buenas de prácticas de gestión y soluciones de software apropiadas. La situación de los Software PMS no son la excepción, pero el alcance a este tipo de tecnologías se ve limitado por aspectos como el costo el cual es habitualmente elevado y la falta de soluciones a distintos rubros que tengan relación con la administración de propiedades.

Para las pequeñas y medianas empresas o PYMEs que componen el grueso del conjunto de las empresas de nuestra región, el uso de tecnologías de información para apoyar sus procesos productivos y de gestión es bajo, esto principalmente por un tema de costos, puesto que no lo ven como una inversión sino mas bien como un gasto. Si bien existen soluciones más económicas éstas no cubren con todos los aspectos y procesos exigidos para definirse como un Software PMS.

En los últimos años el Software Libre ha buscado romper con los esquemas clásicos del desarrollo de software comercial, donde sus libertades permiten a cualquier tipo de empresa encontrar una verdadera competencia en un mercado con tendencia al monopolio, ya que no depende del fabricante de software y pone a disposición un abanico de empresas que puedan hacer negocio mantenimiento sus programas libres o

simplemente tienen la libertad de modificar el programa para su propio uso y adaptarlo a sus necesidades .

La cantidad de proyectos de Software Libre y el número de personas implicadas demuestran que existe capacidad para que se desarrolle de formas muy diferentes y complementarias a la vez. Esta realidad se traduce en que el Software Libre es un instrumento con fuerza suficiente como para generar muchas y diversas ideas de negocio. Los niveles de especialización del software libre no tiene limites, la posibilidad de contar con los códigos fuente de los productos permiten dar un mejor servicio a los clientes, contando además con el apoyo y colaboración desinteresada de las comunidades de todo el mundo.

1.2. Justificación

La falta de desarrollo de un Software PMS de código abierto con las características de un softwares PMS no ha sido presentando ni muchos menos implementado para sistemas GNU/Linux lo que justifica su desarrollo.

El aumento de los negocios en la administración de propiedades es una realidad y es imprescindible proporcionar una alternativa de solución bajo la filosofía del Software Libre. Esta probado que el Software libre es una opción viable para las PYMEs por que posibilita un acceso masivo y sin distinciones económicas a las nuevas tecnologias de información y ofrece nuevas oportunidades crecimiento.

Finalmente creemos que no solo basta con producir una solución libre, es también importante proporcionar herramientas para formar una comunidad que beneficie a las personas, a las empresas y al mismo proyecto.

1.3. Tecnologías relacionadas

Las soluciones PMS basadas en Web están creciendo en popularidad y aceptación. La razón principal es que no solo proveen mayor libertad a los operadores y propietarios sino que incrementan su rentabilidad utilizando este tipo tecnología. El resultado es que no se requiere de mucha experiencia en tecnologías de información ya que el manejo e instalación del software es simple, además que los propietarios pueden hacer uso de su equipamiento existente ahorrando de esta forma en compras de nuevos equipos. A

continuación se realiza un descripción de aquellas soluciones PMS que consideramos importantes.

- RentManager. Es un Software de Administración de propiedades Residenciales y Comerciales, diseñado especificamente para ayudar en el trabajo de los bienes inmuebles de forma más eficiente y efectiva. Rent Manager incluye un completo sistema de contabilidad que incluye libros mayores, cuentas por pagar y cobrar, presupuestos, reportes financieros los cuales se integran con el sistemas de administración de propiedades. Más información se puede obtener de la pagina oficial del producto: http://www.rentmanager.com.

- Web PMS. Es un sistema comercial de administración de propiedades basado en WEB que se encuentra en una creciente popularidad y aceptación como una alternativa para grandes cadenas de hoteles así como tambien hoteles independientes. El software cubre muchas de las características para llevar a cabo el proceso de administración de hoteles. Más información es posible obtener de la página oficial: http://www.webpms.com.

- Phpresidence. Es un programa de código abierto para la gestión hotelera desarrollado por DigitalDruid.Net. Gracias a la gran versatilidad de su interfaz web puede satisfacer una amplia gama de exigencias, desde las de casas de verano con pocos apartamentos hasta las de hoteles con centenares de habitaciones. El programa se encuentra enteramente desarrollado en PHP, disponible en varios idiomas y descargable desde su página principal. Más información se puede obtener de su página oficial: http://www.digitaldruid.net/phpresidence/es/.

- MICROS OPERA: Complete Enterprise Software Solutions. Desarrollada por la empresa Micros, es un sistema comercial que ofrece a los operadores la posibilidad de compartir información entre varias aplicaciones y propiedades en una simple base de datos mientras provee toda las funcionalidades necesarias para la correcta operación. Provee soluciones técnicas y operacionales para hoteles independientes así como para cadenas hoteleras. Más información se puede obtener en su página oficial: http://www.micros.com/Products/OPERA/.

- Property Manager. Desarrollada por la empresa Buildium es un completo sistema de administración de propiedades basado en web, el cual ofrece todas funcionalidades para la correcta operación de una propiedad. Es un sistema comercial de alquiler es decir, que se debe cancelar un cierto monto de dinero mensualmente por su uso. El costo varia según el número de habitaciones que se desean administrar y de algunas funcionalidades extras que se deseen añadir. Más información

se puede recabar del sitio oficial: http://propertymanagement.buildium.com/PM index.shtml.

Capítulo 2

Objetivos

El objetivo principal del proyecto es el de desarrollar un Sistema de Administración de Propiedades PMS adaptable y flexible a cualquier tipo de administración de propiedad creando al mismo tiempo una comunidad entorno al mismo.

Los objetivos especificos de proyecto se detallan a continuación:

- Desarrollar un Sistema de Administración Propiedades (PMS) bajo una licencia libre.
- Elaborar y utilizar herramientas para la difusión difusión del proyecto, que incluyen la construcción y publicación de una página web oficial del proyecto, la promoción a través del uso de redes sociales y su publicación en comunidades virtuales.
- Proporcionar un software de colaboración para la participación de la comunidad en el proyecto, que incluya zona ficheros descargables del programa, foros y listas de mensajes, sistema de seguimiento de fallos y sistema de control de versiones.

A continuación hacemos una breve descripción de cada objetivo específico:

2.1. Sistema de Administración Propiedades Libre

La administración de propiedades requiere de un manejo adecuado en cada uno de sus procesos. A medida que los diferentes tipos de negocios se expanden, las tareas de organización y mantenimiento se vuelven cada vez mas tediosas. Es aquí donde un Sistema

de Administración de Propiedades denominado tambien PMS (Property Management System) tiene su participación, puesto que es un software que integra varios módulos para cubrir los diferentes procesos del manejo de propiedades. En general el PMS debe contemplar los siguientes módulos:

- Módulo de clientes, para registrar la información de todo cliente que se hospeda, aloja, vive, radica o utiliza una propiedad. El sistema debe permitir la creación de perfiles del cliente, con información sobre registros demográficos, cuentas, negocios, grupos, agentes y recursos.
- Modulo de propiedades, para registrar la información sobre bienes inmuebles o muebles sujetos al proceso del negocio y de administración, por ejemplo los datos del propietario, ubicación, valor, galería de imágenes, datos detallados de la propiedad y por supuesto su disponibilidad.
- Módulo de mantenimiento, para controlar y planificar el mantenimiento de las propiedades. Esto implica disponer de un control sobre las adquisición de materiales y servicios para reposición o reparación.
- Módulo de inventarios, para verificar los stocks de insumos o materiales disponibles en propiedades y para su uso en mantenimiento.
- Módulo de utilización y reservas, para asignar o reservar un cliente a una propiedad. Dependiendo del tipo de negocio el módulo deberá asignar una propiedad disponible y crear cuentas o contratos a clientes.
- Modulo de contabilidad, que integra los diferentes módulos del sistema y permite registrar los asientos contables necesarios para generar reportes básicos entre los cuales mencionamos libros diarios, libros mayores, estado de resultados y balance general.

2.2. Herramientas de difusión del proyecto

2.2.1. Sobre la página web del proyecto

La estructura y contenido de la página del proyecto deberá contener las siguientes secciones:

- Sección con información del proyecto, que muestre sus objetivos, características y su disponibilidad para diferentes tipos de plataformas.
- Sección de descargas, con enlaces para la descarga del programa en sus diferentes versiones. Estos archivos deberán contener tanto el código binario como también el código fuente.
- Sección de noticias, para la publicación de los últimos acontecimientos relacionados con el avance del proyecto.
- Sección de documentación, con información sobre la instalación y guías de uso.
- Sección de colaboración, con información sobre la infraestrucutra de apoyo al proyecto.

2.2.2. Redes sociales como herramientas de difusión

Gracias a la revolución tecnológica, la publicidad ha dado un giro de 180°, expandiendo sus horizontes creativos hacia el mundo cibernético, ofreciendo una mayor visibilidad de lo que una empresa busca transmitir en sus mensajes publicitarios. El auge del uso de las redes sociales en el mundo ha provocado que los medios de comunicación centren su interés en ellos. Estos espacios permiten mantener una conexión con sus integrantes a través de una cuenta activa llamada perfil, en la cual el usuario coloca cierta información sobre su persona, gustos, aficiones políticas, religión e intereses; dando como resultado un banco de datos que se puede llegar a convertir en oro molido para cualquier empresa que busque adentrarse en las preferencias de sus clientes potenciales.

Es por esta razón que estas Redes Sociales son consideradas herramientas clave como medios de difusión y este motivo hace necesario su utilización en el proyecto.

2.3. Software de colaboración del proyecto

El exitoso desempeño de las empresas se debe al modo de adaptación y evolución de las diferentes tecnologías orientadas a enfrentar de forma rápida y eficiente las necesidades de los clientes. Para esto, los equipos de trabajo necesitan de un software denominado colaborativo que permita trabajar en conjunto, intercambiar ideas, información y tareas. El software de colaboración a introducir en el proyecto debe satisfacer las siguientes funciones:

- La Comunicación, siendo la función más importante, debe permitir compartir la información.
- La Colaboración, para unir la cooperación y resolver problemas del proyecto.
- La Coordinación, para asegurar que el equipo trabaje eficientemente y en conjunto para alcanzar la meta trazada.

Capítulo 3

Descripción Informática

En ese capitulo se describe la solución del desarrollo del Sistema de administración de Propiedades. En primer lugar se describirán las tecnologías y herramientas empleadas detallando el lenguaje de programación empleado, el framework, tecnologías, bases de datos y entornos para el desarrollo del proyecto (IDE). Posteriormente hacemos una descripción de las diferentes etapas del desarrollo que incluyen el análisis de los requerimientos de la solución y cuestiones de diseño. En la etapa de implementación se realizan descripciones sobre la estructura del codigo fuente, su codificación basado en el patrón MVC (Modelo, Vista, Controlador) del Framework elegido asi como tambień las pruebas o tests del programa, para finalizar con la generación de documentos, descripción de la implantación e instalación del programa.

3.1. Herramientas y tecnologías empleadas

3.1.1. Lenguaje de programación Ruby

Actualmente, existe una gran cantidad de lenguajes de programación que son utilizados en diversas áreas, desde el desarrollo de aplicaciones administrativas hasta el campo de la inteligencia artificial donde su selección se efectúa en base a gustos o inquietudes.

Ruby se presenta como una alternativa interesante ya que es un lenguaje sencillo y flexible que atrae a programadores de todos los sectores. A pesar de tener muchos años en el mercado, el auge del lenguaje llegó de la mano de un framework para aplicaciones web denominado Rails. Esto hizo que muchos desarrolladores web migraran desde sus lenguajes más tradicionales, como PHP o ASP, a la nueva y fascinante opción.

Ruby es un lenguaje de scripts, multiplataforma, netamente orientado a objetos y es software libre, fue creado por Yukihiro Matsumoto. Ruby hereda varias caracaterísticas de lenguajes como: Perl, Smalltalk, Eiffel, Ada y Lisp. Como lo indica su propio autor, es un lenguaje "aparentemente sencillo pero internamente complejo".

Su sintaxis simple y su curva de aprendizaje lo sitúan como una alternativa excelente para el desarrollo del proyecto. La libre disponibilidad del lenguaje hace que sea una herramienta para tener en cuenta en entornos empresariales. Ruby además permite a los desarrolladores que utilicen términos como elegante, interesante y divertido para describir la experiencia de utilizarlo en el trabajo diario.

3.1.2. Desarrollo de aplicaciones Web

El impacto del internet y las tecnologías móviles ha tenido sus consecuencias sobre el desarrollo de aplicaciones de software, donde aspectos como la compatibilidad y la disponibilidad de acceso se ha un convertido en algo a tomar cuenta a la hora de encarar un nuevo proyecto de software. Estas nuevas necesidades tecnológicas han permitido también el nacimiento del término "*usabilidad*", que consiste en la facilidad con que las personas pueden utilizar una herramienta particular o cualquier otro objeto fabricado por humanos con el fin de alcanzar un objetivo concreto. Históricamente, las técnicas de usabilidad era aplicadas principalmente a las aplicaciones de escritorio (Desktop), simplemente por que las herramientas no se encontraban disponibles en aplicaciones web. Sin embargo, a medida que el Internet se ha vuelto más maduro, la tecnología web se ha nutrido de herramientas que hacen posible la creación de aplicaciones más rápidas, ligeras y robustas.

Las ventajas que nos proporciona una aplicación basada en web son:

- Compatibilidad multiplataforma, las aplicaciones web tienen un camino mucho más sencillo para la compatibilidad multiplataforma que las aplicaciones de software de escritorio, ya que permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales.
- No requieren instalación, pues usan tecnología Web, lo que permite el aprovechamiento de todas las características del Internet.
- Son fáciles de usar, no requieren conocimientos avanzados de computación.
- Alta disponibilidad, ya que es posible realizar consultas en cualquier parte del mundo donde exista acceso a Internet y a cualquier hora.

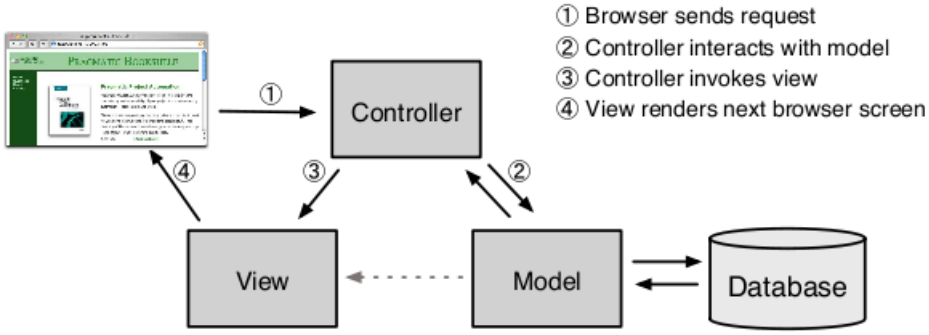


Figura 3.1: Modelo Vista Controlador MVC

En este sentido aparecen los denominados frameworks, que son estructuras de software conformados por componentes personalizables e intercambiables, que permiten un diseño reutilizable y que facilitan y agilizan el desarrollo de un sistema. Si bien en la actualidad existen muchos Frameworks para el desarrollo de aplicaciones web nos centramos el Framework Ruby On Rails por su facilidad de desarrollo, despliegue y mantenimiento.

3.1.3. Framework Ruby On Rails

También conocido como RoR o Rails es un framework de aplicaciones web de código abierto escrito en el lenguaje de programación Ruby, siguiendo el paradigma de la arquitectura Modelo Vista Controlador (MVC ver Figura 3.1). Ruby On Rails es un framework que combina simplicidad con la posibilidad de desarrollar aplicaciones del mundo real escribiendo poco código respecto a otros frameworks y con un mínimo de configuración. Estas características le han permitido pasar de ser un juguete desconocido a un fenómeno a nivel mundial, llegando a convertirse en una elección de un Framework para muchas aplicaciones Web.

En los últimos años, Ruby On Rails han cobrado mucha fuerza, y ya existe una buena cantidad de recursos en la WEB, ademas de que las grandes compañías como Sun Microsystems, están apostando fuertemente a este lenguaje que ya es considerado como parte de la WEB 2.0. Este crecimiento está dado por una necesidad de simplificar y de hacer una experiencia placentera y ágil.

Ruby on Rails es simple y funcional, su uso de Active Record es eficiente, que simplifica y agiliza el desarrollo de forma notable. Minimiza el trabajo con la base de datos y emplea un único lenguaje para todo el desarrollo, lo que consigue acortar los tiempos

de desarrollo. Estos motivos lo convierten en el Framework ideal para el desarrollo del proyecto.

3.1.4. Base de datos MySQL

MySQL es un sistema de administración para bases de datos relacionales (RDBMS[1]) de código abierto que provee una solución robusta a los usuarios con poderosas herramientas multi-usuario, soluciones de base de datos SQL (Structured Query Language) multi-threaded. Es rápido, robusto y facil de utilizar.

Inicialmente, MySQL carecía de algunos elementos esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de esto, atrajo a los desarrolladores de páginas web con contenido dinámico, debido a su simplicidad, de tal manera que los elementos faltantes fueron complementados por la vía de las aplicaciones que la utilizan. Poco a poco estos elementos faltantes, están siendo incorporados tanto por desarrolladores internos, como por desarrolladores de software libre.

La elección como base de datos para el proyecto se basa en que MySQL es el estándar defacto para sitios web de gran tráfico por su motor de consultas de alto rendimiento, su escalabilidad y flexibilidad, fuerte protección de datos, facilidad de gestión y su alta disponibilidad.

3.1.5. Entorno de desarrollo Netbeans

NetBeans es un Entorno de Desarrollo Integrado o IDE(Integrated Development Environment) para desarrolladores de software. Es un proyecto de código abierto el cual provee todas las herramientas para la creación de aplicaciones usando el lenguaje JAVA, como también C/C++, PHP, JavaScript, Groovy y Ruby. Es fácil de instalar y de usar en diferentes sistemas operativos incluyendo Windows, Linux, Mac OS X y Solaris.

Netbeans es una de las mejores herramientas para la programación en Ruby y el desarrollo con el FrameWork Ruby On Rails, razón por la cual se emplea para el desarrollo del proyecto. Algunas de sus características son:

- Editor de Codigo Ruby, identifica, completa y resalta el código Ruby sintática y semánticamente, lo que facilita la refactorización, inferencia y la navegación del código fuente.

[1]Relational Data Base Management System

- Manejo de proyectos Ruby, soporta archivos Ruby, archivos bajo la especificación RSpec y archivos YAML. Netbeans provee herramientas de integración y proporciona acceso al Shell Interacitivo de Ruby (IRB).
- Manejo de proyectos Ruby on Rails, soporta las migraciones de objetivos y bases de datos (Rake). Los proyectos mantienen la estructura separando claramente los controladores de los modelos, vistas y las migraciones de las bases de datos.
- Manejo de Gemas de Ruby, que extienden las capacidades del entorno.

3.2. Análisis y Diseño

En base a los módulos necesarios para conformar un software PMS que fueron definidos en el capítulo dos, es que se definen las funcionalidades principales del sistema. Si bien existen varios diagramas para describir la funcionalidad y diseñar el programa nos concentramos en los Diagramas de Casos de Uso (DCU) y el Digrama de Clases ya que éstos nos permiten proporcionar una idea general del solución propuesta. Por un lado los diagramas de casos nos permiten describir las principales funcionalidades y requisitos de la aplicación mientras que el diagrama de clases nos da una idea clara de la estructura de las clases del sistema incluyendo las propiedades y metodos de cada clase. A continuación detallamos los diagramas comentados.

3.2.1. Diagrama de Casos de Uso

- DCU Módulo Cliente (Figura 3.2)

  Este módulo comprende las funcionaliades de registro de toda la información concerniente a las personas que se registran y usan las propiedades disponibles en el programa. Las funcionalidades que incluye son: el registro de información del cliente, información adicional de la persona y localización geográfica (pais y ciudad).

- DCU Módulo Propiedades (Figura 3.3)

  Cómo su nombre lo indica permite el registro de las características de las propiedades, tipos (casas, habitaciones, maquinas, autos) y su disponibilidad (ocupado, disponible, en mantenimiento, etc.).

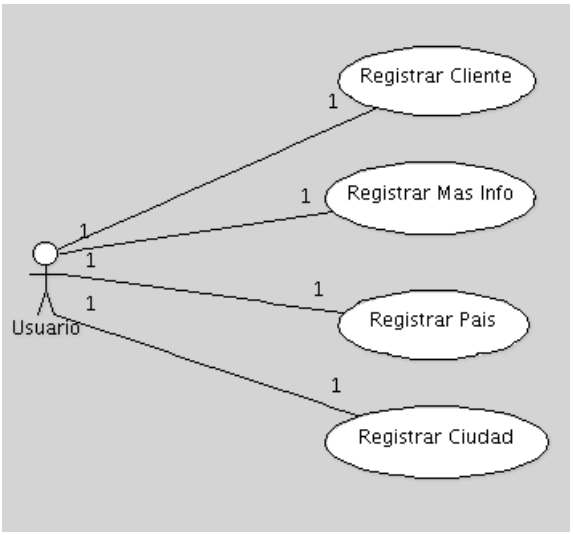


Figura 3.2: DCU Modulo Cliente

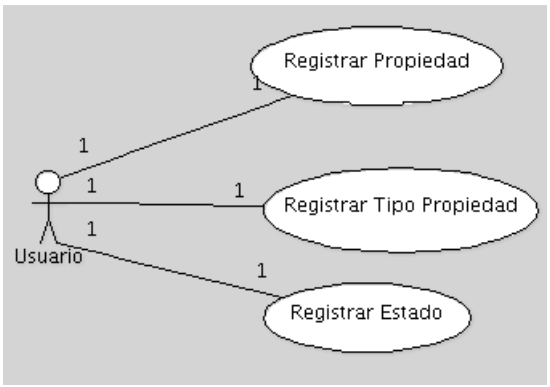


Figura 3.3: DCU Módulo Propiedad

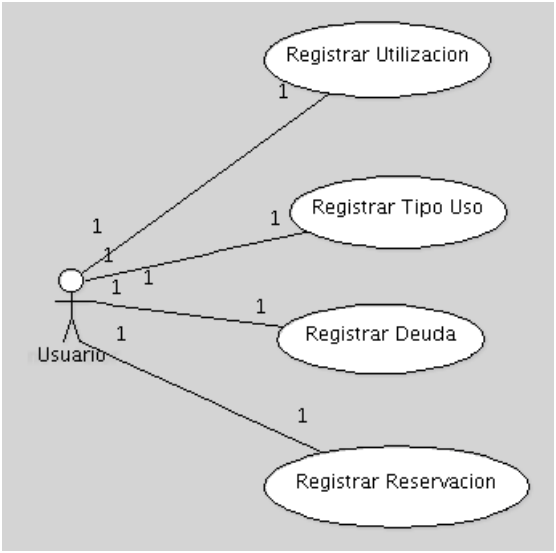


Figura 3.4: DCU Módulo Utilización

- DCU Módulo Utilización (Figura 3.4)

  El módulo de utilización asigna a un cliente una propiedad disponible. El proceso registra un tipo de utilización (renta, alquiler, hospedaje), así como también el precio de uso, fecha de ingreso (checkin) y fecha de salida (checkout) para obtener los ingresos por el servicio prestado. Otra funcionalidad incorporada es la reservación que permite una planificación de disponibilidad a futuro. El módulo también integra un control de deudas sobre clientes que tienen servicios pendientes de pago.

- DCU Módulo de Inventario (Figura 3.5)

  La funcionalidad del módulo de inventarios se concentra en el control de inventariar, ingresar y retirar productos. Este módulo se conforma básicamente del registro de unidades (lts, piezas, gr, kg, etc.), grupos de productos (muebles, electrodomésticos, camas, etc,) y el registro de un producto.

  El módulo también incorpora la funcionalidad de control de movimientos de productos (ingresos y salidas) categorizadas bajo un tipo de movimiento que define las reglas ingresos y salidas de cada producto (ingresos/salidas normales, ajustes de stock, ingresos por stocks iniciales, etc.).

- DCU Módulo de Contabilidad (Figura 3.6)

  Respecto al módulo de contabilidad se conforma de funcionalidades básicas para el registro de cuentas, tipos de cuentas y movimientos de las cuentas.

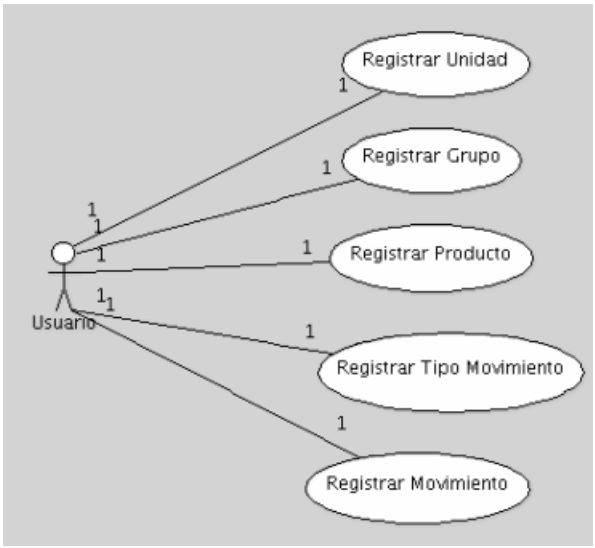


Figura 3.5: DCU Módulo Inventario

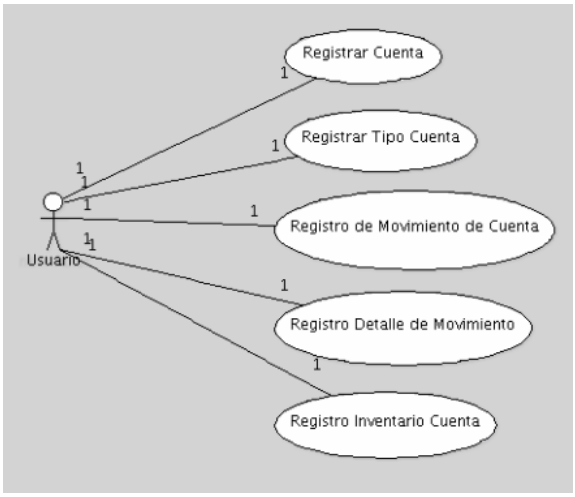


Figura 3.6: DCU Módulo Contabilidad

El módulo también integra el flujo de inventarios y contabilidad a través de una relación entre cuentas y productos, esto con el objetivo de generar y automatizar movimientos en contabilidad que reflejen cambios en stocks de productos.

3.2.2. Diagrama de clases

El diagrama de clases es uno de los diagramas más importantes durante el proceso de diseño puesto que nos permite establecer las clases y las relaciones que existen entre ellas, además de definir los atributos y funcionalidades a través de sus métodos. Existen numerosas herramientas que automatizan la generación de código lo que lo ha convertido en un diagrama muy popular.

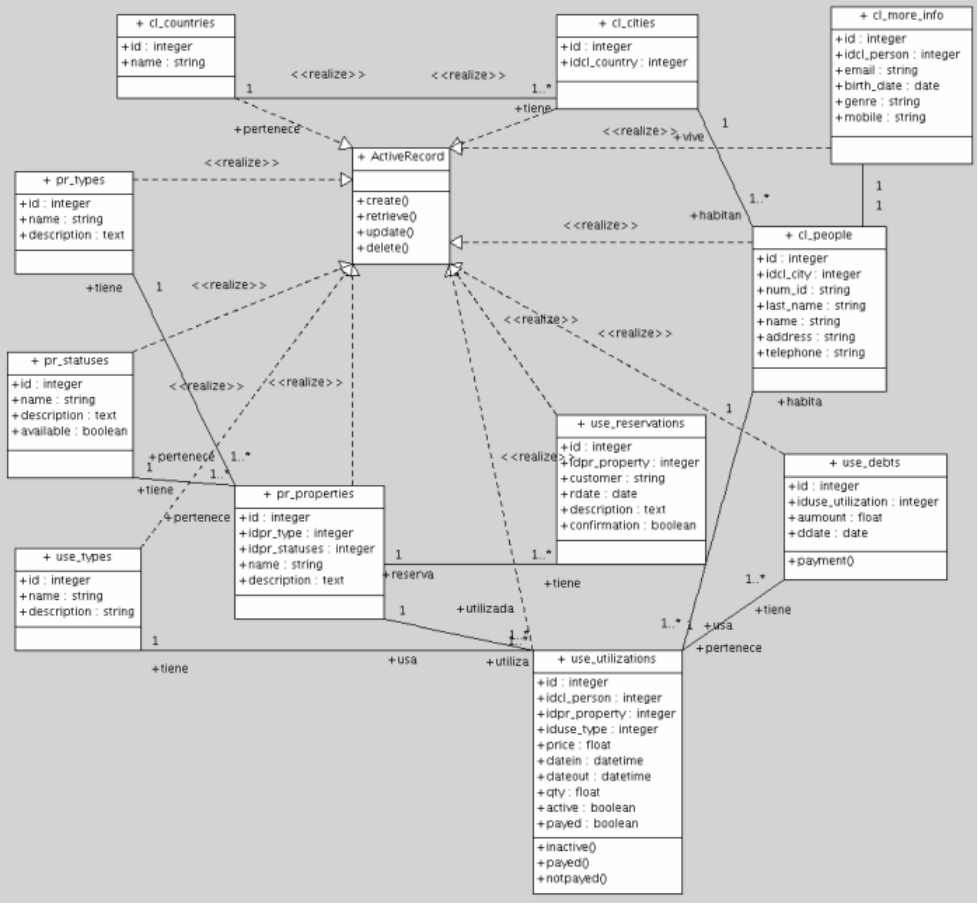


Figura 3.7: Diagrama de Clases: Propiedades, Clientes y Utilización

En las Figuras 3.7-9 se muestra los diagramas de clases de la aplicación. Para una mejor apreciación y diferenciación cada clase se antepone el nombre del módulo, por ejemplo para las clases pertenecientes al módulo de clientes su nombre inicia con "cl_" y para las clases del módulo propiedades se antepone el nombre "pr_".

Es posible observar también que todas las clases heredan de la clase "ActiveRecord" que es un patrón construido para Ruby on Rails, y hace que sea fácil la implementación de CRUD (Create, Retrieve, Update, Delete). En los siguientes apartados se describe más sobre este patrón.

3.2.3. Diseño de interfaces

Desde el punto de ingeniería de software, la interfaz de usuario juega un papel preponderante en el desarrollo y puesta en marcha de todo sistema. La aceptación final del software por parte del usuario depende en gran manera de la percepción que éste tenga del sistema y esta percepción se logra mediante la interfaz del sistema.

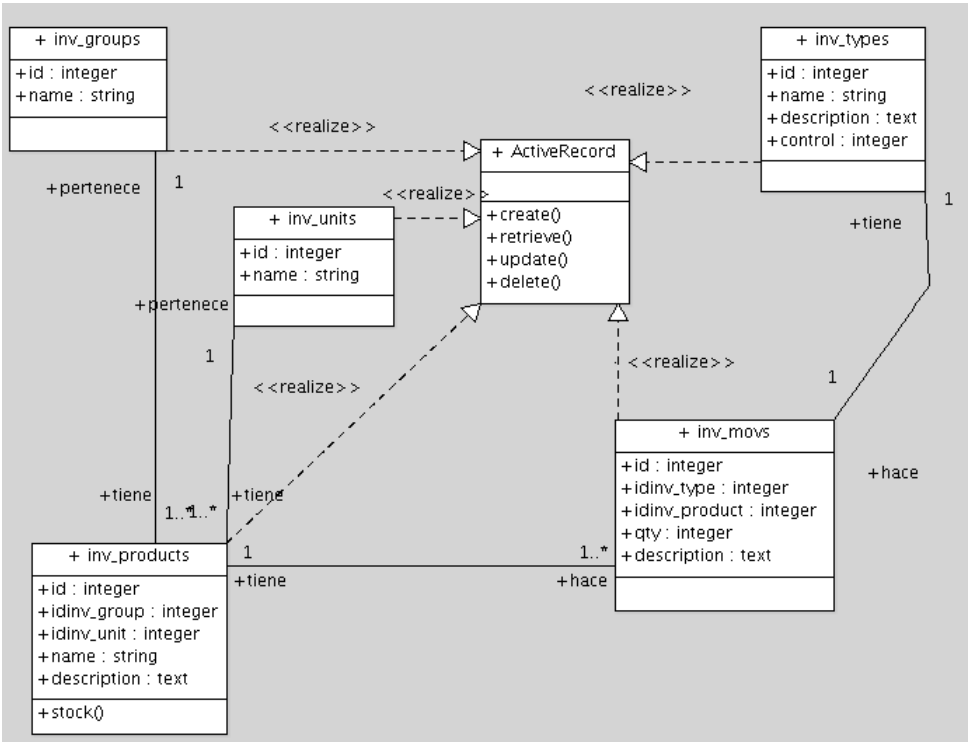


Figura 3.8: Diagrama de Clases: Módulo Inventario

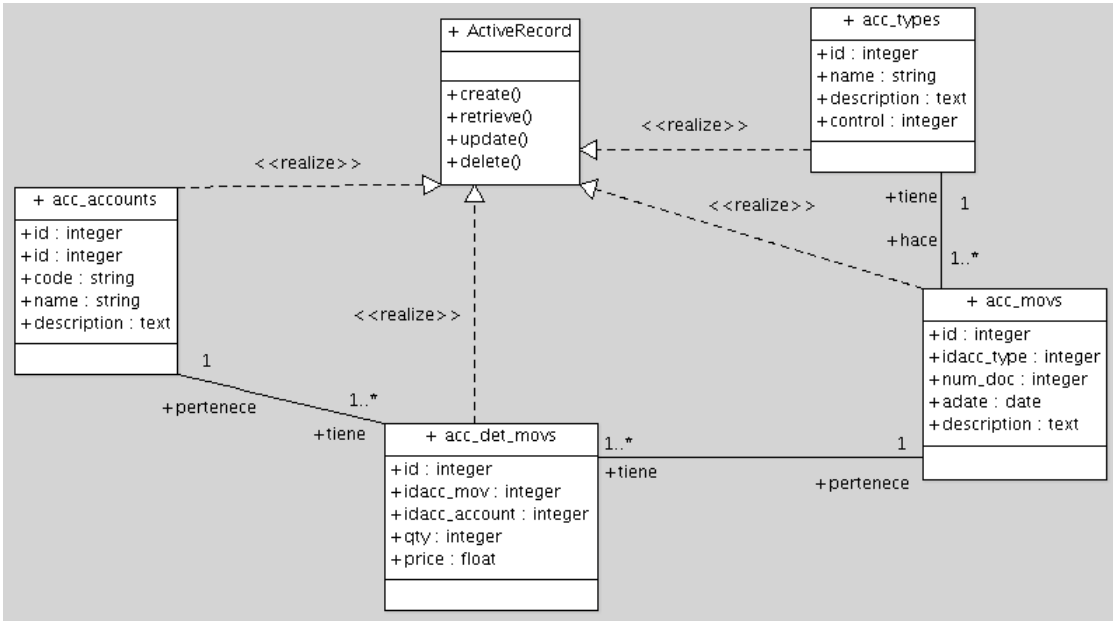


Figura 3.9: Diagrama de Clases: Módulo Contabilidad

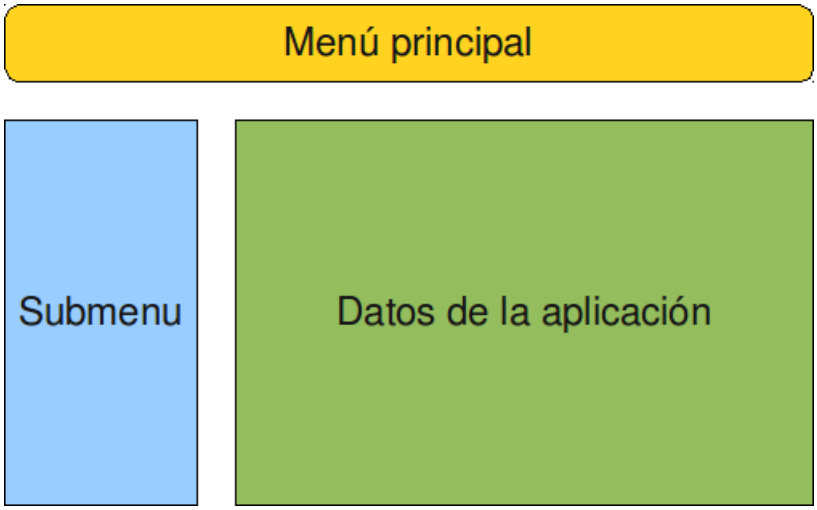


Figura 3.10: Interfaz de la aplicación web

En los ultimos tiempos los diferentes paradigmas de desarrollo de sistemas siguen una tendencia que involucra a la interfaz de usuario como una parte muy importante en el proceso de desarrollo, uno de estos paradigmas es el conocido con el nombre de MVC (Modelo Vista Controlador) utilizado ampliamente en todo sistema que integre facilidades para la construcción de interfaces usando la metodología orientada a objetos.

Lo deseable es que las interfaces de las aplicaciones para la Web sean fáciles de usar, fáciles de navegar, agradables al usuario, que tenga los elementos bien distribuidos, de tal forma que no se haga una saturación de la página y que la intrefaz contenga la información que el usuario espera de ella, además de que por supuesto debe seragradable visualmente.

Desde este punto de vista en la Figura 3.10 se muestra la distribución de la interfaz de nuestra aplicación. En la sección de "Menú principal" se ubican las opciones de acceso a los módulos, en la sección "Submenu" se encuentran las opciones de registro de acuerdo al módulo seleccionado y en la sección de "Datos de la aplicación" contiene las opciones de listado, creación, actualización y eliminación de cada clase. En la Figura 3.11 se muestra la interfaz de la aplicación en funcionamiento.

3.3. Implementación

3.3.1. Arquitectura de la aplicación

La rapidez en el desarrollo de proyectos con Rails está fundamentada en la idea de construir la aplicación separando de forma clara las capas de Modelo (Datos), Vista

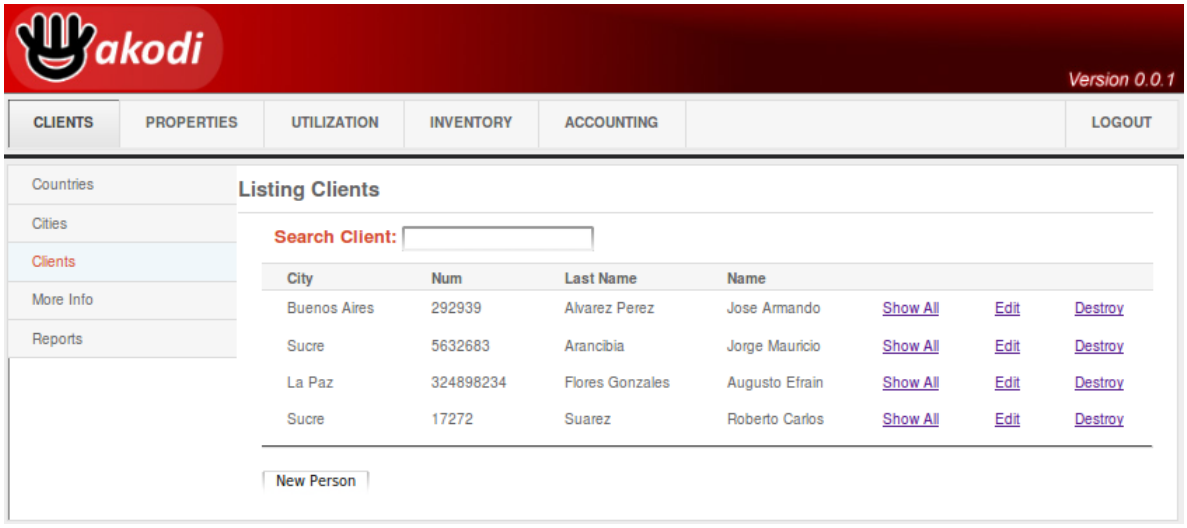


Figura 3.11: Interfaz del proyecto

(Presentación) y Controlador (Funciones y métodos) para reducir el acoplamiento entre la lógica de negocios y la de presentación. De modo que, antes de empezar a describir el trabajo realizado con Rails, no está de mas tener claro los conceptos que engloba la arquitectura Modelo Vista Controlador (MVC).

- Modelo

  En las aplicaciones web orientadas a objetos sobre bases de datos, el Modelo consiste en las clases que representan a las tablas de la base de datos. El modelo es responsable de mantener el estado de la aplicación. Algunas veces este estado es transitorio, durando solo para un par de interacciones con el usuario. En otras ocasiones es permanente y será almacenado a veces fuera del catión del application, a menudo en una base de datos.

  En Ruby on Rails, las clases del Modelo son gestionadas por ActiveRecord. Por lo general, lo único que tiene que hacer el programador es heredar de la clase ActiveRecord::Base, y el programa averiguará automáticamente qué tabla usar y qué columnas tiene.

- Vista

  La Vista es el responsable de generar la interface de usuario, normalmente basada en los datos contenidos en el modelo. Cualquier variable creada en una instancia de la acción del Controlador estará disponible en la vista como también en la aplicación y las funciones de ayuda (helpers). Con frecuencia en las aplicaciones web la vista consiste en una cantidad mínima de código incluido en HTML.

- Controlador

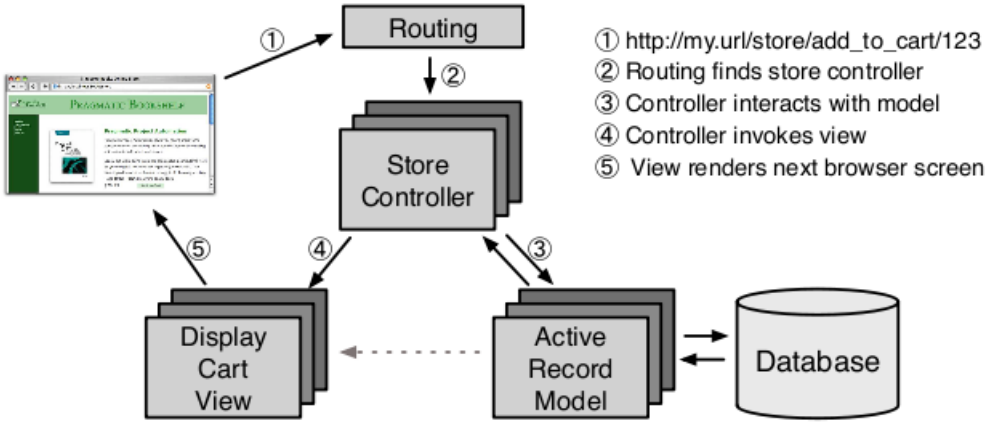


Figura 3.12: MVC en Rails

> En MVC, las clases del Controlador responden a la interacción del usuario e invocan a la lógica de la aplicación, que a su vez manipula los datos de las clases del Modelo y muestra los resultados usando las Vistas. En las aplicaciones web basadas en MVC, los métodos del controlador son invocados por el usuario usando el navegador web.

La arquitectura de nuestra aplicación utiliza la arquitectura MVC del Framework Rails, tal como se muestra en el la Figura 3.12. Una solicitud es enviada en primer lugar a un enrutador, el cual realiza el trabajo de buscar, donde en nuestra la aplicación la solicitud debe ser enviada. Este enrutador identifica un metodo (action) en particular en alguna parte del controlador. La acción en el controlador puede obtener datos de la solictud o posiblemente interactuar con el modelo para eventualmente preparar la información para la vista.

Sobre el active "ActiveRecord" de Rails podemos mencionar que proporciona la capa objeto-relacional que sigue rigurosamente el estándar ORM[2]: Tablas en Clases, Registros en Objetos, y Campos en Atributos. Facilita el entendimiento del código asociado a base de datos y encapsula la lógica especifica haciéndola más fácil de usar para el programador.

Ventajas del ActiveRecord:

- Se trabajan las entidades del Modelo más naturalmente como objetos.

[2]Object-Relational-Mapping:técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional

- Las acciones como Insertar, Consultar, Actualizar, Borrar, etc. de una entidad del Modelo están encapsuladas así que se reduce el código y se hace más fácil de mantener.
- Código más fácil de Entender y Mantener.
- Reducción del uso del SQL, con lo que se logra un alto porcentaje de independencia del motor de base de datos.
- Menos "detalles" más practicidad y utilidad .
- ActiveRecord protege en un gran porcentaje de ataques de SQL inyection.

3.3.2. Creación de la aplicación y estructura de carpetas

Una de las políticas en el desarrollo de aplicaciones bajo Rails es la "Convención antes de de Configuración", esto quiere decir que no hay que perder tiempo en tediosos archivos de configuración y otros similares, ya que con esta estructura Rails ya sabe donde se encuentra todo. Para la construcción de la aplicación se emplea el siguente comando:

"$ rails wakodi"

El comando genera una serie de carpetas tal como se muestra en la Figura 3.13 que compone la estructura de nuestra aplicación. A continuación se describe brevemente el objetivo que cumple cada una de ellas.

- **app**, es la carpeta principal donde radica el código de nuestra aplicación y donde se situa el comentado Modelo Vista Controlador con subcarpetas referidas a este término.
- **config,** donde se encuentran varios archivos que como su nombre lo indica, son pequeños archivos en los cuales se configura el comportamiento de la aplicación (rutas URL y base de datos)
- **db,** guarda los esquemas (schemas) y migraciones de la base de datos.
- **doc**, contiene la documentación mas concretamente el API de los modelos, vistas, controladores y librerías generados por RDoc.
- **lib**, contiene las librerías que utiliza nuestra aplicación

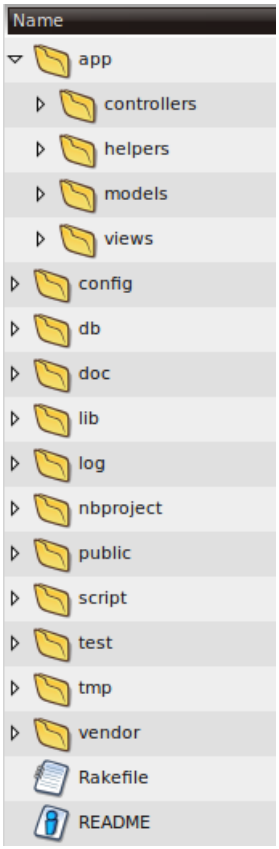


Figura 3.13: Estructura de carpetas

- **log**, en esta carpeta se encuentran diferentes archivos que guardan a modo de bitácora todo lo sucede en el momento exacto de nuestra aplicación. Uno de los más empleados es el "development.log", donde es posible observar las consultas que se hacen a la base de datos.
- **nbproject**, no es una carpeta propia de una aplicación rails, sino más bien una carpeta de configuración que manipula nuestro IDE (Netbeans) para manejar el proyecto.
- **public**, aquí se encuentran todo los archivos que se mostrarán en el navegador del cliente, esta carpeta incluye directorios para la manipulación javascript y stylesheets además de unos cuantos archivos como el index principal de la página y mensajes errores
- **script**, en esta carpeta se encuentran los comandos disponibles para la construcción de la aplicación
- **test**, en esta carpeta se ubican clases para realizar las pruebas.
- **tmp**, que contiene los archivos temporales, como por ejemplo, variables de sesión e información almacenada en el cache entre otras.

- **vendor**, donde se ubica cualquier archivo, componente, script o plugin que provenga de algun proveedor externo.

3.3.3. Construcción de la solución

La facilidad y rapidez que ofrece el Framework Rails para la construcción de una aplicación ha sido comentado en anteriores apartados, pero es importante proporcionar un ejemplo válido que nos permita dar una idea del trabajo realizado así como también el de demostrar el verdadero potencial que representa el uso de esta herramienta. Es por eso que en esta sección hacemos una descripción general de la construcción de la aplicación tomando como referencia el módulo "propiedades".

Si bien cada módulo del sistema presenta su propias particularidades en cuanto a funcionalidades se refiere, la descripción de un solo módulo es suficiente para entender la solución global.

- **Configuración de la base de datos**

  En muchos lenguajes-script de aplicaciones web, la información de como conectarse a la base de datos se encuentra embebida directamente en el código fuente. es posible encontar una llamada a un método de conexión (connect), en el que se transfieren el nombre del "host" y el nombre de la base de datos, acompañados de nombre del usuario y la contraseña. Esto representa problemas de seguridad ya que la información de la contraseña se ubica en un archivo que es accesible via web.

  Rails mantiene la información de la conexión a la base de datos fuera del código en un archivo plano, el cual puede ser localizado en "config/database.yml". Este archivo contiene tres secciones, cada una con una conexion a una base de datos de desarrollo, test y producción. Por defecto Rails construye este archivo con una conexión a una base de datos SQLite[3]. Para nuestro caso se requiere de una conexión a una base de datos MySQL, los cambios efectuados a este archvio se muestran en la Figura 3.14.

  Con el archivo de configuración listo solo resta crear la base de datos y probar su conexión. Para la creación de la base de datos se presenta dos alternativas, la primera que consiste en construir a partir de comandos en MySQL, la segunda

[3] **SQLite** es un sistema de gestión de bases de datos relacional compatible con ACID, y que está contenida en una relativamente pequeña (~275 kiB)1 biblioteca en C. SQLite es un proyecto de dominio público creado por D. Richard Hipp.

```
development:
  adapter: mysql
  encoding: utf8
  database: wakodi-0.0.1_development
  username: user
  password: passwd
  host: localhost
  socket: /var/run/mysqld/mysqld.sock
test:
  adapter: mysql
  encoding: utf8
  database: wakodi-0.0.1_test
  username: user
  password: passwd
  host: localhost
  socket: /var/run/mysqld/mysqld.sock
production:
  adapter: mysql
  encoding: utf8
  database: wakodi-0.0.1_production
  pool: 5
  username: user
  password: passwd
  host: localhost
  socket: /var/run/mysqld/mysqld.sock
```

Figura 3.14: Configuración de base de datos

es a través de comandos que ofrece rails, por razones obvias se emplea la segunda opción.

Utilizamos al comando "rake"[4] para crear la base de datos, el comando completo desde consola es el siguiente:

"$ rake db:create RAILS_ENV='development"'

Finalmente se comprueba la conexión a la nueva base de datos y que Rails posee los permisos necesarios, en consola se ejecuta el siguiente comando:

"$ rake db:migrate"

- **Generación del MVC con scaffold**

  Una de las características más interesantes de Rails es su script denominado "Scaffold" que traducido al español significa andamio. Esta característica permite tener las funcionalidades básicas de administración de datos de un modelo en un controlador. Las funcionalidades proporcionadas son las denominadas CRUD (Create, Retrieve, Update, Delete), típicas de cualquier sistema transaccional.

  Para su aplicación en nuestro programa obtenemos las funcionalidades y atributos a partir del diagrama de clases (Ver Figura 3.15), en el que se identifican tres clases que conforman el módulo "Propiedades".

[4] **Rake** es el equivalente a make para Ruby. Sirve para crear y automatizar tareas de mantenimiento.

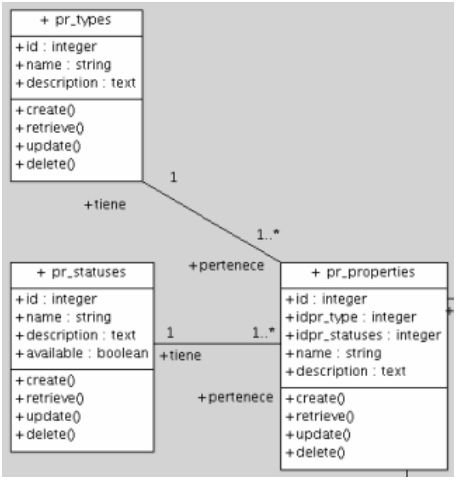


Figura 3.15: Diagrama de Clase: Módulo propiedades

Entonces se ejecutan los siguientes comandos:

$ ./script/generate scaffold pr_type id:integer name:string description:text

$ ./script/generate scaffold pr_status id:integer name:string description:text available:boolean

$ ./script/generate scaffold pr_property id:integer idpr_type:integer idpr_status:integer name:string description:text

Tras ejecución de cada comando, Rails crea los archivos necesarios para una funcionalidad básica CRUD de cada clase. Solo resta hacer fisicas las tablas en la base de datos, para esto Rails dispone de opciones de migración. Cada migración representa un cambio que se quiere hacer a la base de datos, expresado en un archivo independiente del SGBD empleado. Estos cambios pueden actualizar tanto el esquema de la base de datos como los datos en las tablas. Se ejecuta el siguiente comando para llevar a cabo lo anteriormente comentado.

$ rake db:migrate

Para acceder a la nuevas funcionalidades recientemente generadas desde el navegador ingresamos a la siguiente dirección:

http://localhost:3000/pr_properties

En las Figura 3.16 - 18 se muestran los resultados.

Type	Status	Name	Description			
Cars	Not Available	Ford Focus	Color: White, 5 doors, V6.	Show	Edit	Destroy
Machines	Not Available	Machine 2	Features:	Show	Edit	Destroy
Machines	Available	Molino	No provided.	Show	Edit	Destroy
Apartment	Not Available	Room 1	All room description should.	Show	Edit	Destroy
Apartment	Available	Room 10	No TV, no bathroom	Show	Edit	Destroy
Apartment	Not Available	Room 11	TV, Jacuzzi	Show	Edit	Destroy
Apartment	Available	Room 12	TV and Jacuzzi	Show	Edit	Destroy
Apartment	Available	Room 13	Jacuzzi	Show	Edit	Destroy
Apartment	Not Available	Room 2	TV-Cable	Show	Edit	Destroy
Single Room	Not Available	Room 3	No TV.	Show	Edit	Destroy

Figura 3.16: Opción “index”

Figura 3.17: Opción “new” y “update”

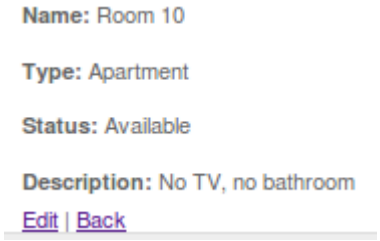


Figura 3.18: Opción “show”

Figura 3.19: Validación con Rails

- **Mejoras en apariencia y funcionalidad**

  Una vez construidas todas las clases se dispone de una aplicación con funcionalidad básica, ahora es necesario dar algunos ajustes con el objetivo de mejorar la apariencia y extender las funcionalidades de la aplicación.

  Respecto a la apariencia, en la carpeta "app/views/layouts" y "public/stylesheets" se localizan los archivos y hojas de estilos necesarios para realizar esta tarea.

  Respecto a la extension de funcionalidades, Rails ofrece muchas opciones que simplifican las tareas del programador. Una herramienta interesante por ejemplo son las validaciones, puesto que ya se tiene definidas reglas de control las cuales deben ser asignadas al modelo. A continuación hacemos mención de algunos de los validadores más empleados en la aplicación:

  - **validates_uniqueness_of :attribute**, que valida la existencia única del atributo.
  - **validates_presence_of :attribute**, que valida y exige un valor para el atributo.
  - **validates_numericality_of :attribute**, que valida que el atributo sea numérico

  Para verificar el funcionamiento de los validores, cuando se ingresa un valor que no cumple con algunas de reglas establecidas, el sistema notifica con un mensaje el error (Ver Figura 3.19)

3.4. Pruebas

La calidad del software es una preocupación a la que se dedican muchos esfuerzos. sin embargo, el sotware casi nunca es perfecto.Todo proyecto tiene como objetivo producir el software de la mejor calidad posible, que cumpla, y si puede se supere las expectativas de los usuarios. Existen varias técnicas para la comprobación de tests tanto de caja blanca "white-box testing" como caja negra "black-box testing". Este último nos permite

comprobar la funcionalidad de los componentes de nuestra aplicación. Se establecen ciertos parámetros entrada y debe producir los resultados esperados. Este tipo de test también es muy útil cuando se desea asegurar la funcionalidad a medidad que nuestra aplicación evoluciona.

Una técnica de "black-box" muy empleada es la prueba unitaria, que consiste en probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. Luego, con las pruebas de integración, se podrá asegurar el correcto funcionamiento del sistema o subsistema en cuestión.

Ruby on Rails utiliza la librería "Test::Unit" como estándar para crear pruebas de unidad. Al utilizar los generadores de Rails para crear cualquier módulo, ya sea modelos o controladores, rails automáticamente crea una prueba de unidad para dicho controlador o modelo en el directorio "test", incluyendo una que otra prueba básica. Además, el comando "rake" en una aplicación de rails ejecuta las pruebas. De esta manera es posible añadir las pruebas de unidad y ejecutarlas con un sólo comando.

3.4.1. Pruebas unitarias con rails

En este apartado se muestra las pruebas realizas a la aplicación. En primer se crea la base de datos para la ejecución de las pruebas, de acuerdo a la información proporcionada en el archivo de configuración "database.yml", se ejecuta el siguiente comando en caso de que la base de datos no estuviera construida.

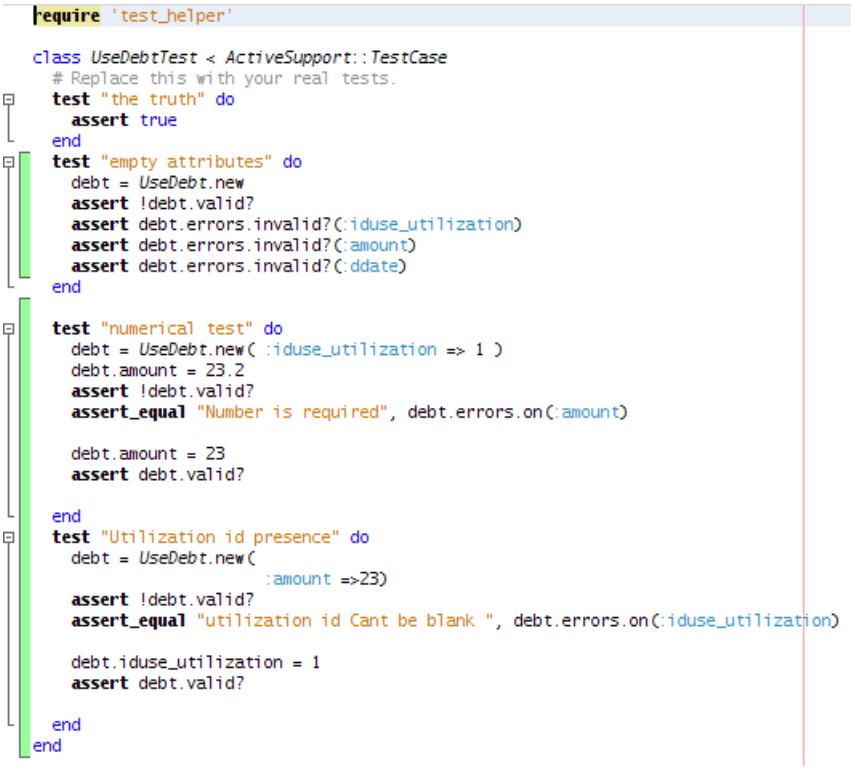
"$ rake db:create RAILS_ENV='test"'

Posteriormente se prepara el esquema de la base de datos de prueba para emparejar con la estructura de base de datos de desarrollo, para esto recurrimos al siguiente comando:

"$ rake db:test:prepare"

Tomando como ejemplo el modelo "use_debt.rb" en la carpeta "test/units" se encuentra el archivo "use_debt_test.rb" que es un archivo donde se definen las pruebas al modelo (Ver Figura 3.20). Para realizar la prueba desde consola ejecutamos

"$ ruby -I test test/unit/use_debt_test.rb"



```
require 'test_helper'

class UseDebtTest < ActiveSupport::TestCase
  # Replace this with your real tests.
  test "the truth" do
    assert true
  end
  test "empty attributes" do
    debt = UseDebt.new
    assert !debt.valid?
    assert debt.errors.invalid?(:iduse_utilization)
    assert debt.errors.invalid?(:amount)
    assert debt.errors.invalid?(:ddate)
  end

  test "numerical test" do
    debt = UseDebt.new( :iduse_utilization => 1 )
    debt.amount = 23.2
    assert !debt.valid?
    assert_equal "Number is required", debt.errors.on(:amount)

    debt.amount = 23
    assert debt.valid?

  end
  test "Utilization id presence" do
    debt = UseDebt.new(
                        :amount =>23)
    assert !debt.valid?
    assert_equal "utilization id Cant be blank ", debt.errors.on(:iduse_utilization)

    debt.iduse_utilization = 1
    assert debt.valid?

  end
end
```

Figura 3.20: Pruebas unitarias en rails

Rails tambien nos proporciona “Fixtures”, que en el mundo de las pruebas es un ambiente en el cual se pueden ejecutar pruebas. Un “Test Fixture” es simplemente una especificación de valores iniciales de un modelo para pruebas. Para especificar los valores del “fixture” Rails maneja archivos con formato YAML los cuales se ubican en “text/fixtures/”.

3.5. Documentación

Una parte fundamental del software es la documentación que lo acompaña. La documentación es una pieza tan importante de un programa que en ocasiones éste será inútil sin ella.

Rails hace fácil la construcción la documentación para el programador, con su utilidad RDoc automatiza la construcción de la documentación en formato HTML a partir de los comentarios vertidos en el código de la aplicación.

Para la generación de documento se ejecuta el siguiente comando:

“$rake doc:app”

3.6. Instalación y configuración

3.6.1. Instalación

Previa a la instalación de la aplicación es importante establecer las dependencias de aplicaciones y herramientas. A continuación detallamos los mismos.

- Ruby >= 1.8.6
- Rails 2.3.5
- MySQL >=5.1.37

Plugins necesarios:

- Will Paginate
- MySQl con soporte InnoDB

El proceso de instalación de la aplicación se bastante sencillo. En primer lugar debemos obtener una copia de la última versión la cual se encuentra disponible en la página del proyecto.

Una vez descargado el archivo y extraido los archivos dentro de la carpeta donde desea instalar el programa, se siguen los siguientes pasos:

1. Creamos una base de datos en MySQL para el sitio Web. Solo se necesita crear la base de datos "production", pero si se desea desarrollar extensiones o ejecutar tests es conveniente crear las bases de datos "development" y "test".
2. Modificamos "config/database.yml" para la conexión.
3. Se ejecuta el bootstrap de la base de datos con el comando rake:

   "$ rake db:migrate RAILS_ENV="production"

   (Si se desea hacer un rake a la base de datos de desarrollo se cambia la variable por "development")
4. El proceso de inicio de la aplicación es al igual que cualquier aplicación basada en Rails.

   "$ script/server -e production"
5. Abrimos en nuestro navegador en el puerto 3000 (http://localhost:3000).

Capítulo 4

Desarrollo en Comunidad

En el anterior capítulo se han descrito las diferentes herramientas y tecnologías empleadas para la construcción de la solución. Hasta este punto disponemos de una aplicación con funcionalidades básicas para la adminsitración de propiedades. Pero el objetivo del proyecto no solo es el dar una solución a nivel de programa sino el de crear una comunidad entorno al proyecto. Para llevarlo a cabo esta tarea es importante proporcionar las herramientas que faciliten la creación de una comunidad que beneficie a las personas, a las empresas y al mismo proyecto.

Este capítulo precisamente describe las herramientas empleadas para crear la comunidad. Primeramente se realiza una descripción de las ventajas de las comunidades virtuales y como éstas pueden aportar al proyecto., definimos la licencia de la aplicación, más adelante se describen las herramientas de difusión que incluyen la construcción de la página oficial de proyecto, uso de redes sociales y páginas que publicitan software de código abierto. Finalizamos con la descripción del software de colaboración empleado para gestionar el desarrollo del proyecto.

4.1. Comunidades Virtuales y el proyecto

Con el crecimento de la red y los nuevos lenguajes de desarrollo web, hemos llegado a un punto en el que la cantidad de servicios ofrecidos en portales para los desarolladores e ingenieros de software es enorme, y no solo abarca herramientas que ayudan a éstos en su desarrollo diario, sino que ponen en contacto a patrocinadores de proyectos con desarrolladores, de modo que el software libre y nuestro proyecto pueda crecer y expandirse. Las ventajas del uso de comunidades virtuales son prácticamente ilimitadas, a continuación hacemos mención de alguna de ellas:

Ventajas

- Incrementa la experiencia social puesto que permiten las exposición de las personas a otras culturas y entornos que de otra forma nunca sería posible. De esta forma las comunidades en el ciberespacio se encuentra mas concentradas en la experiencia social mas que en las características individuales.
- Permite que un grupo de usuarios interelacionados y entusiastas ayudarse mutuamente en el uso de un producto o servicio.
- Si se esta interesado en el desarrollo de un producto o servicio es posible obtener conocimiento de otras personas apasionadas con el tema de tu interés y discutir sobre nuevos temas de investigación.
- Ofrecen muchas nuevas oportunidades no solo para compañias sino también para sitios de caridad, voluntariado, organizaciones sin fines de lucro y gubernamentales que buscan que las personas se interesen en sus objetivos y visiones.
- Libera a los individuos de las limitaciones geográficas y temporales de su comunidad física, ya que puede ser parte de cualquier comunidad virtual con la cual se identifique.
- Las bolsas de trabajo también se van configurando como entornos donde se agrupan los profesionales y las empresas dinamizadoras de la nueva economía, las comunidades virtuales van más allá de responder a necesidades puntuales de inserción; se constituyen en entornos referentes de ocupación que acompañan a los profesionales a lo largo de su vida laboral y donde pueden encontrar información, recursos y servicios que les permitan estar al día y adaptarse a la evolución permanente del mercado.
- Desde el punto de vista tecnológico, las comunidades virtuales mejoran la comunicación haciéndola más rápida y barata. Las comunidades virtual proveen la compartición de archivos, acceso público a servicios, comunicación por voz o chat, conferencias de audio/video y experiencias en realidad virtual.
- Desde el punto de vista comercial, esta claro que las empresas se dirigen al mercado, y el mercado está en las comunidades virtuales. Las comunidades virtuales garantizan la competencia ya que les permite abaratar costos, innovar y diversificar.
- A nivel educativo, las comunidades virtuales permiten conectar a las escuelas al Internet y brindar acceso a los estudiantes al Internet a través de ellas es la

forma más barata de garantizar el acceso universal y la participación de las nuevas generaciones en la sociedad digital del futuro.

A nivel de software libre:

- Las comunidades virtuales son herramientas que nos permiten dar a conocer un proyecto de una forma sencilla. Nos ofrecen las herramientas necesarias para la creación y gestión de un proyecto de software libre.
- La compartición del conocimiento es vital para un proyecto de software libre y las comunidades virtuales no solo nos permiten llevarlo a cabo sino también se logra obtener una retroalimentación en pos de la mejora del mismo.
- A la hora de solucionar un problema no existe mejor fuente de información que las comunidades virtuales ya que existe la posibilidad de encontrarnos con situaciones similares.

4.2. Licencia del proyecto

El proyecto adoptará una licencia GNU Public License (GPL) desde su primera versión. El proyecto nace libre y de distribución gratuita y la mejor forma de asegurar esa libertad es mediante una licencia GPL. Dos aspectos motivan la adopción de una licencia GPL, en primer lugar por que se pretende asegurar que terceros mantengan el espíritu y la filosofía de libertad con la que ha sido concebido el proyecto. En segundo lugar son la cantidad de proyectos bajo esta licencia, ya que la comunidad es mayor en comparación con otros proyectos libres, lo que incrementa la probabilidad de captura de futuros participantes del proyecto.

4.3. Herramientas de difusión del proyecto

4.3.1. Sobre el nombre y el logo del proyecto

Una de las partes más importante al crear un proyecto es la elección del nombre porque la describe y la distingue al otorgarle una identidad. El nombre es la única manera de llamar al proyecto constituyendo entonces en el atributo más visible. Algunos de los detalles a tomar en cuenta son:

Figura 4.1: Logo del Proyecto

- Identidad.

  El nombre debe evocar en alguna forma al producto en cuanto a su género, función o beneficio, también es válido acuñar una palabra sin ningún significado pero con sonoridad lograda con base a la rima entre sus silabas y que, con el tiempo, adquirirá un significado generalizado.

- Pronunciabilidad

  El nombre elegido debe ser más o menos corto y fácil de pronunciar, preferentemente incluyendo en otros idiomas.

- Recordabilidad

  Antes de aceptar un determinado nombre debemos asegurarnos que el público lo recuerde con facilidad, de lo contrario la labor de posicionamiento será doblemente difícil.

El nombre elegido a su vez es el nombre que generalmente se emplea en el registro del dominio, entonces su importancia es doble por que de ello puede depender la facilidad con la que nuestro sitio sea encontrado por personas que deseen conocer y participar del proyecto.

Bajo estas consideraciones se elige como nombre del proyecto a la palabra “Wakodi” que procede del idioma africano Swahili cuyo significado es: arrendatarios, inquilinos o huéspedes que pagan. El nombre seleccionado es interesante por que cumple con muchos de los requisitos ya que es simple, fácil de pronunicar y recordar y representa los objetivos del proyecto.

Respecto al logotipo generalmente todo proyecto de software requiere de uno, ya que este representa al proyecto antes los ojos del público. Por otra parte lado establece en la mayoría de los casos la apariencia de un página Web. En la Figura 4.1 se muestra el logotipo elegido para el proyecto.

4.3.2. Sistema de gestión de contenidos (CMS)

El costo de tiempo que supone realizar un proyecto de desarrollo de una página web completamente nueva y programada de manera personalizada ya no es viable en la actualidad. Sobretodo si la estructura y contenido de la página se limita a la provisión de información básica. En Internet se encuentran disponibles los denominados Sistemas de Gestión de contenidos (en inglés, Content Managment System, abrevido CMS), que son programas que permiten crear una estructura de soporte (Framework) para la creación y administración de contenidos, principalmente de páginas web, por parte de los participantes. Los CMS tienen la ventaja de que cuesta menos ponerlos en funcionamiento y su curva de aprendizaje es pequeña.

Si bien existen muchos software CMS libres disponibles en Internet para el proyecto se empleó Radiant CMS ya que está construido sobre "Ruby on Rails" lo que facilita a los desarrolladores web extenderlo para otros fines. Radiant se encuentra liberado bajo una licencia MIT y se caracteriza por su simpleza y versatilidad.

Radiant CMS cuenta con una interfaz de usuario simple que es elegante y se centra en diseños, snippets y páginas. Tiene una estructura muy flexible que le permite organizar las páginas en cualquier orden. Además, tiene su propio lenguaje de macros llamado "radius template" con lo que es fácil añadir contenido de otras páginas. También dispone de filtros de texto personalizados y cacheo inteligente. Mas información sobre esta aplicación se puede obtener en la página principal del proyecto (http://radiantcms.org/).

4.3.3. Página Oficial del Proyecto

El sitio oficial web de un proyecto de software libre es fundamental, ya que debe dar a conocer a los usuarios y a los desarrolladores los objetivos perseguidos por el proyecto. El sitio web del proyecto es una ventana hacia el mundo para alcanzar una base de usuarios, colaboradores y desarrolladores suficientes para dar sostenibilidad al proyecto.

Para la publicación del página se ha optado el registro del dominio wakodi.org y se ha adquirido el servicio de WebHosting con soporte Rails para dar alojamiento a nuestra página. La Página Oficial del Proyecto puede ser accedida desde el siguiente: http://www.wakodi.org.

En la Figura 4.2 se muestra la página principal del proyecto, según los objetivos definidos en el cápitulo dos referidos a los contenidos básico.

Figura 4.2: Pagina Web Wakodi

4.3.4. Difusión en Redes Sociales

De las redes sociales existentes nos concentramos en Facebook y Twitter que son dos aplicaciones que han llegado a tener gran aceptación y popularidad por la comunidad de usuarios. El proyecto se beneficia porque es posible obtener contactos con la industria u otras personas que comparten una misma opinión, recabar información sobre comentarios del proyecto, estar al tanto de los avances del mismo, proponer nuevas ideas y consejos para su mejora y por supuesto el intercambio de experiencias. Básicamente estos aspectos facilitarán también la creación de una comunidad entorno al proyecto.

En las Figuras 4.3 y 4.4 se muestra las páginas creadas en estas dos redes sociales.

4.3.5. Publicitar en freshmeat.net

Cada día, es necesario tanto para encontrar colaboradores como patrocinadores publicitar nuestros proyectos con las herramientas adecuadas. casi ninguna cumple todas espectativas, pero sí que podemos lograr, usando varias de ellas, el objetivo deseado.

Una de las páginas más utilizadas para publicitar es Freshmeat.net cuyo objetivo es el de servir como medio de información de las últimas actualizaciones y versiones de software publicadas, además de leer o escribir análisis y artículos relacionados, enviar o

Figura 4.3: Wakodi en Facebook

Figura 4.4: Wakodi en Twitter

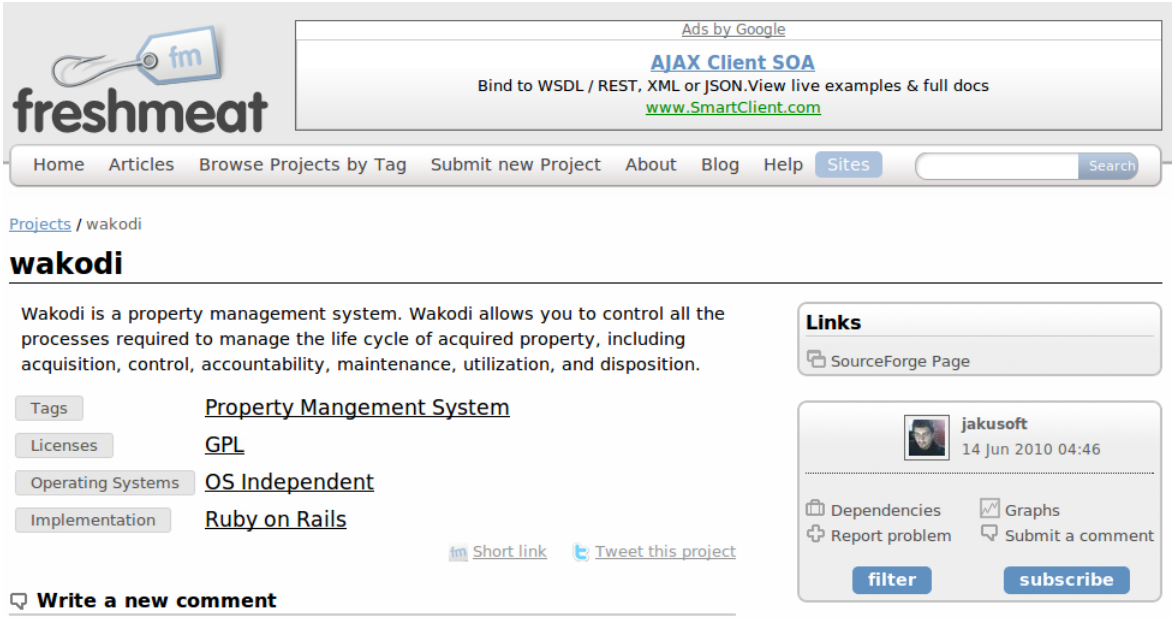


Figura 4.5: Wakodi en Freashmeat.net

recibir comentarios, etc. Aunque la mayor parte es software libre, también informa de software privativo y para múltiples plataformas como Linux o Windows. En la Figura 4.5 se muestra la publicación del proyecto.

4.4. Software de Colaboración SourceForge

En el desarrollo de cualquier proyecto o en la gestión del soporte en cualquier ambito de los sistemas de información (tanto si se trata de soporte interno o a clientes), se requiere el uso de herramientas apropiadas que nos permitan la gestión de dicho soporte, permitiendonos hacer un seguimiento de los procesos, realizar tareas de control o reporting, así como documentar adecuadamente las acciones realizadas.

Sin lugar a duda SourceForge es el sitio más famoso como software de colaboración para el desarrollo de proyectos de Software Libre y es elegido como el sitio desarrollo del proyecto. Lo más importante de SourceForge es que gracias a las herramientas que ofrece es posible realizar toda la tarea de coordinación y comunicación entre desarrolladores, para que éstos no acaben duplicando trabajo y caminen coordinados, es decir para que personas en distintas partes del mundo trabajen juntos como si estuvieran dentro de la misma oficina.

Por otra parte, si quien accede a SourceForge es un usuario que lo que desea es descargarse la última versión de un determinado programa, la web pone a su disposición un

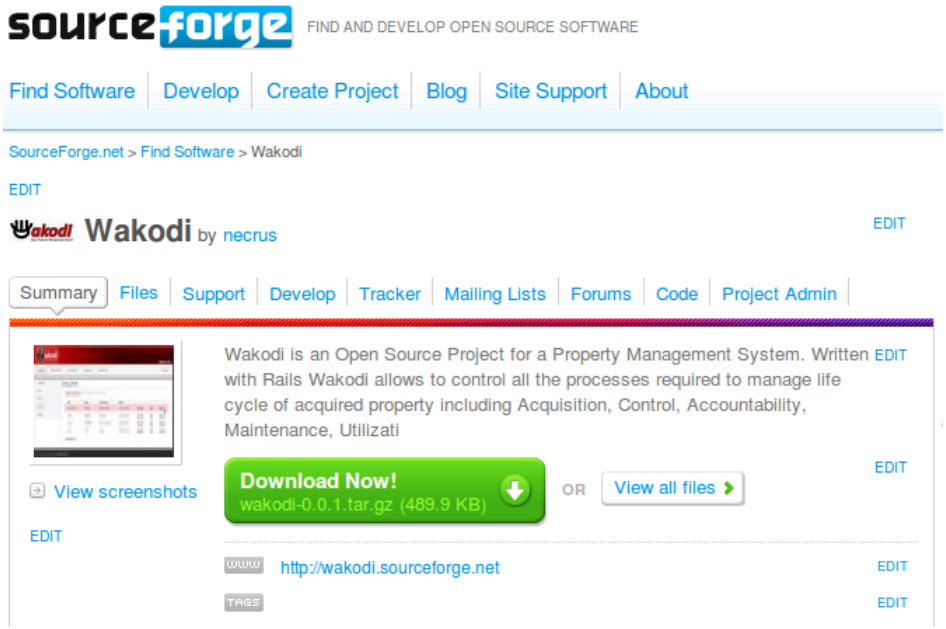


Figura 4.6: Wakodi en SourceForge

buscador para localizarlo rápidamente. Además, ofrece un amplio directorio por el que navegar, y en el que están clasificados de forma temática todos los programas alojados.

En la Figura 4.6 se muestra la pagina del proyecto en SourceForge, el enlace es: http://sourceforge.net/projects/wakodi/

4.4.1. Control de Versiones con Subversion

Para el manejo del código del proyecto SourceForge pone a disposición Subversion. Antes de acceder al repositorio se debe instalar un cliente Subversion disponible para casi todos los sistemas operativos.

Se puede obtener una copia del proyecto a través de SVN con la siguiente instrucción:

"$ svn co https://wakodi.svn.sourceforge.net/svnroot/wakodi wakodi"

4.4.2. Seguimiento de Fallos

El sistema de seguimiento de fallos de SourceForge se denomina Tracker, y no está ideado sólo para la gestión de fallos. A continuación se listan trackers(seguimientos) disponibles en Sourceforge (Ver Figura 4.7), donde es posible crear, añadir o editar los artefactos de cada tracker.

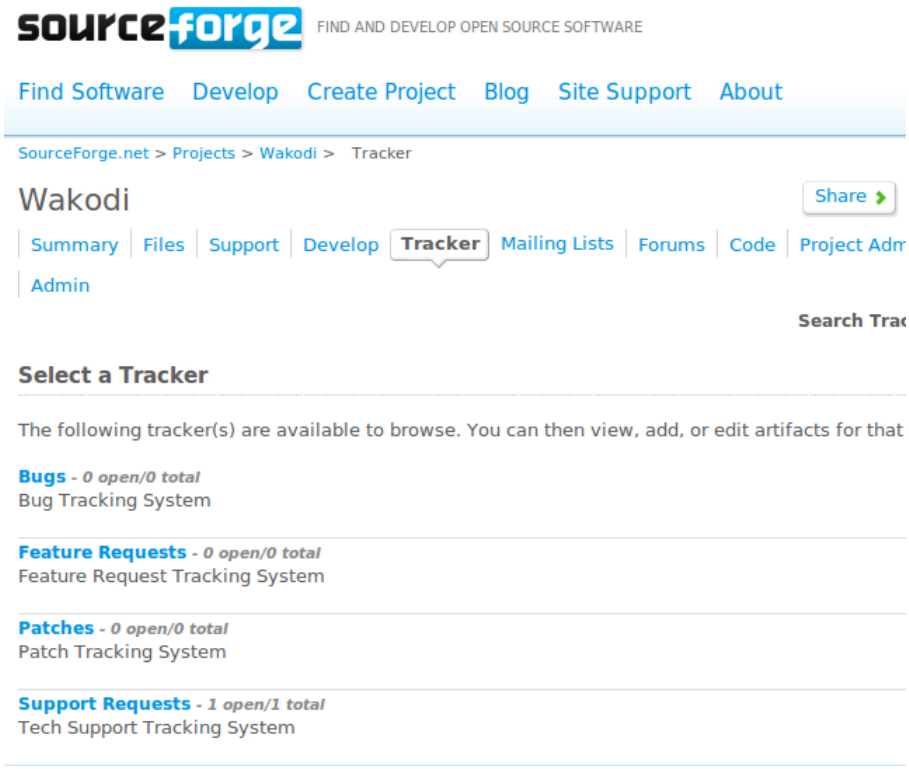


Figura 4.7: Trackers disponibles para el proyecto

- Support requests [Peticiones de soporte]
- Bug reports [Informes de fallos]
- Feature requests [Peticiones de funcionalidad]
- Patches [Parches]

4.4.3. Listas de correo

Para mejorar la comunicación y la coordinación entre programadores, la divulgación del proyecto, así como también la comunicación e intercambio entre los usuarios, se han creado una serie de lista de correo del proyecto de acuerdo a la información que se desea manipular. A continuación hacemos detalle de cado uno de ellas.

- **Lista General**, para uso diario y para el intercambio de todo tipo de información. Enlace creado para su subscripción es:

  https://lists.sourceforge.net/lists/listinfo/wakodi-general

- **Lista de Desarrollo**, con información para los desarrolladores del proyecto. El enlace creado para su subscripción es:

  https://lists.sourceforge.net/lists/listinfo/wakodi-development

- **Lista de la Página Web**, para tratar todo tipo de temas relacionados con la estructura, diseño y contenido del sitio web. El enlace creado para su subscripción es:

  https://lists.sourceforge.net/lists/listinfo/wakodi-webpag

- **Lista de documentación**, para tratar temas relacionados con la documentación, manuales de usuario y materiales derivados. El enlace creado para su subscripción es:

  https://lists.sourceforge.net/lists/listinfo/wakodi-documentation

4.4.4. Descargas del Proyecto

Si bien es posible disponer en la página oficial del proyecto de una sección para las descargas del software, la herramienta de descargas de SourceForge nos ofrece otras opciones como el manejo de estadísticas de descarga, historiales de publicaciones (releases) y la posibilidad réplicas para agilizar el proceso de descarga.

Capítulo 5

Conclusiones

5.1. Conclusiones

El trabajo presentado es un Proyecto de Desarrollo de Aplicaciones en Software Libre, donde se ha generado una aplicación empleando herramientas basadas en esta filosofía y se ha creado un ambiente que facilite la formación de una comunidad. Creemos que se han cumplido con muchos de los objetivos planteados, los cuales podemos mencionar:

- A nivel del Software de Administración de Propiedades.

  Se ha planteado el desarrollo de una aplicación de la cual no existen muchas alternativas en Software Libre.

  La solucion ha sido desarrollada utilizando nuevas tecnologías Web que agilizan el proceso implementación.

  Respecto a la funcionalidades del software PMS, se han desarrollado los módulos de Clientes, Propiedades, Disposición, Inventarios y Contabilidad. Sin bien presentan funcionalidades básicas éstos se encuentran construidos de tal forma que son fáciles de extender.

- A nivel de la difusión del proyecto

  Se ha elaborado y publicado la página oficial del proyecto lo que nos permite disponer de una ventana hacia el mundo para dar a conocer el proyecto y sus objetivos.

  También se ha publicado el proyecto tanto en las redes sociales más populares así como en páginas que fomentan y publicitan el uso de software libre.

- A nivel del software de colaboración

  Se han creado y configurado todas las herramientas necesarias para que la comunidad pueda participar del proyecto. Estas herramientas incluyen un sistema de control de versiones, sistema de seguimiento de fallos, foros, listas de correo y área de descargas del proyecto.

Como conclusión podemos mencionar que el abanico de herramientas disponibles para la creación un Proyecto de Software Libre es inmenso, ya sean estas para el desarrollo de nuevas aplicaciones o simplemente para su utilización, donde el futuro del proyecto dependerá principalmente de la aceptación y la respuesta de los usuarios que son básicamente los artífices del éxito del Software Libre.

5.2. Limitaciones

Sobre las limitaciones del proyecto nos centramos en las funcionalidades del software, donde:

- En esta primera versión el acceso a la aplicación es pública, la seguridad no ha sido implementada, el sistema no dispone de un registro y autentificación de usuarios.
- La información considera solo la necesaria para su funcionamiento, la aplicación no contempla el manejo de imágenes sobre módulos de clientes y propiedades, la interconexión del módulo de contabilidad con inventarios o la generación de reportes.
- La interfaces si bien presentan una estructura que falicita su aprendizaje, queda limitada su interactividad con el usuario a falta de desarrollo de interfaces utilizando RIA (Rich Intelligent Application).
- El sistema se encuentra disponible solo en idioma “Inglés”, la internacionalización no ha sido tomada en cuenta, lo que puede provocar cierta limitación en la captura de usuarios.
- Respecto a la documentación si bien se ha generado un API Documentation, hace falta un mayor detalle y especificación sobre los clases desarrolladas para facilitar el entendimiento de los desarrolladores.

5.3. Trabajos futuros

A partir de las limitaciones planteadas en el punto 5.2 presentamos a continuación trabajos futuros a realizar:

- Implementación del registro y autentificación de usuarios utilizando plugins disponibles en Rails para dotar de forma segura y sencilla esta funcionalidad. Algunos ejemplos de estos plugins podemos mencionar a Restful Authentication y Authlogic.
- Implementación de nuevas funcionalidades a los diferentes módulos para extender sus capacidades además de mejoras en el control de los datos para reducir problemas de integridad.
- A nivel de las interfaces de los usuarios es importante diseñar sistemas intuitivos de interacción e informar a los usuarios la respuesta que tienen sus acciones en la pantalla. La incorporación de AJAX se presenta como una solución interesante.
- Respecto a la internacionalización, si lo que se desea es ampliar el universo de posibles usuarios y participantes del proyecto es importante disponer de la aplicación en difernes idiomas.
- Respecto a la documentación, mejoras en los comentarios del código para generar información más útil al desarrollador.

Bibliografía

[1] Breve Introducción a los Sistemas Colaborativos: Groupware & Workflow. G. Gerónimo, V. Canseco. Mexico

[2] Redes sociales, la nueva era de la publicidad. Mtra. Rocío Flores R., Basado en el estudio de consumo en medios digitales 2009. Mexico 2009. (http://www.infosol.com.mx/espacio/cont/trinchera/nueva_era.html)

[3] Web 2.0: El negocio de las Redes Sociales. Fundación de la Innovación Bankinter. 2007

[4] The Modern Property Management System. Jon Inge. Hospitality Upgrade. 2003

[5] Property Management Guide: A Guide to Implementing a Property Management System in an Organization. Waldo O. Smeby

[6] Assessing Property Management for Affordable Housing. Marc Diaz. Fellowship Program for Emerging Leaders in Community and Economic Development. 2004.

[7] Desarrollo de Aplicaciones en Software Libre. Juan José Amor Iglesias, Israel Herraiz Tabernero, Gregorio Robles Martínez. Fundació per a la Universitat Oberta de Catalunya. 2007.

[8] Wikipedia. The Free Encyclopedia. (http://www.wikipedia.org)

[9] Netbeans. Integrated Development Environment. (http://netbeans.org/features/index.html)

[10] Radiant CMS. Open source content management system designed for small teams. (http://www.radiantcms.org)

[11] MySQL. The world's most popular open source database. (http://www.mysql.com)

[12] Agile Web Development with Rails 3rd Edition. Sam Ruby Dave, Thomas, David Heinemeier Hansson. The Pragmatic Programmers LLC. 2009

[13] Patrones de Interacción para el Diseño de Interfaces WEB usables. Ma. Elena Hernández Hdz., Guillermo Alvarez Carrión, Jaime Muñoz Arteaga. Universidad Autónoma de Tlaxcala (UAT). Departamento de Ingeniería y Tecnología. Instituto Nacional de Astrofísica Óptica y Electrónica (INAOE), Departamento de Ciencias Computacionales. Puebla México