UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería
Departamento de Ciencia de la Computación

Portal para el registro de miembros de la Asociación de Scouts de Guatemala

Trabajo de graduación presentado por Ricardo Guerra Chávez para optar al grado académico de Licenciado en Ciencia de la Computación

Guatemala
2012

Portal para el registro de miembros de la Asociación de Scouts de Guatemala

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería
Departamento de Ciencia de la Computación

Portal para el registro de miembros de la Asociación de Scouts de Guatemala

Trabajo de graduación presentado por Ricardo Guerra Chávez para optar al grado académico de Licenciado en Ciencia de la Computación

Guatemala
2012

Vo. Bo. :

Ing. Luis Furlán

Tribunal Examinador:

MSc. Douglas Barrios

Ing. Lynette García

Ing. Luis Furlán

Fecha de aprobación: Guatemala, 6 de junio de 2012.

PREFACIO

La elaboración del presente trabajo surgió de una idea del Ingeniero Luis Furlán, Director del Departamento de Computación de la Universidad del Valle de Guatemala y miembro del Consejo Nacional de la Asociación de Scouts de Guatemala. El pensó en darle a este proyecto de graduación una orientación social y como miembro activo de los Scouts, tenía conocimiento de la falta de un sistema automatizado para el registro de sus miembros.

La Asociación de Scouts de Guatemala es una entidad no lucrativa, cuyos recursos son limitados y que depende en mayor parte del trabajo de personas voluntarias para su funcionamiento. La Asociación cuenta actualmente con varios miles de miembros a nivel nacional, por lo que la existencia de un registro informatizado es una necesidad primordial.

Con la elaboración de este proyecto, se quiere establecer una base de datos de todos los miembros que actualmente forman, o anteriormente han formado parte del movimiento scout de Guatemala y tener un registro histórico de sus miembros y su participación en el escultismo. Se busca, igualmente, que la Asociación de Scouts se inicie en el uso de las tecnologías de la información y que el software realizado continúe desarrollándose más allá del alcance de este proyecto.

No me resta más que agradecer a aquellas personas que con su tiempo y esfuerzo han hecho posible la culminación de este trabajo. Agradezco en particular al señor William Collado, Jefe Scout Nacional, y al señor Sergio Barrientos, Subjefe Scout Nacional, quienes con sus aportes, consejos y explicaciones me permitieron comprender la parte operativa de la Asociación de Scouts y ayudaron a establecer los requisitos funcionales del sistema informático realizado como parte de este trabajo. Quiero expresar mi agradecimiento muy reconocido también para el Ingeniero Luis Furlán por su propuesta de elaborar un proyecto con orientación social y quien como asesor principal, me ayudó durante su preparación con críticas, sugerencias y correcciones.

CONTENIDO

Página

LISTA DE TABLAS

Página

Tabla

LISTA DE FIGURAS

Página

Figura

RESUMEN

La Asociación de Scouts de Guatemala no cuenta con un sistema informatizado para llevar el control de registro e inscripciones de sus miembros. Con el desarrollo del portal para el registro de scouts se busca dar apoyo a las principales funciones relacionadas con la afiliación y organización de los miembros de la Asociación de Scouts, además de establecer una base de datos que permita almacenar la información de todo el proceso.

El portal de registro de scouts es una aplicación Web que emplea *symfony 1.4* como plataforma de desarrollo. *Symfony* está desarrollada en el lenguaje PHP orientado a objetos y permite desarrollar aplicaciones en el mismo lenguaje. Su estructura está diseñada para promover el uso del patrón Modelo-Vista-Controlador y las mejores prácticas para la elaboración de herramientas Web. Como complemento de la aplicación esta la base de datos relacional administrada por *MySQL 5.5*.

Este proyecto cumple con conceptos basados en la teoría de diseño de bases de datos relacionales y fue desarrollado siguiendo metodologías ágiles para el de desarrollo de sistemas. El enfoque ágil utilizado demostró ser el más apropiado, no solamente por tratarse del desarrollo de una aplicación Web, sino también porque la Asociación de Scouts de Guatemala no contaba anteriormente con ningún sistema informático para el registro de sus miembros, creando un mayor grado de incertidumbre para el establecimiento detallado de los requisitos del software por parte de los directivos de la Asociación de Scouts.

I. INTRODUCCIÓN

La Asociación de Scouts de Guatemala cuenta con un gran número de miembros, cuya inscripción debe realizarse anualmente. Además, se organizan eventos y actividades en las cuales pueden participar todos los miembros inscritos. La falta de una base de datos con información acerca de sus miembros exige que la información sea requerida más de una vez provocando una pérdida de tiempo y recursos. Con la creación de un portal para el registro de miembros de la Asociación de Scouts se pretende proporcionar un sistema ágil y seguro para integrar dicha información en la base de datos y además servir de apoyo a otras actividades relacionadas con la participación de sus miembros.

Este proyecto desea promover el uso de sistemas de información en el seno de la Asociación de Scouts con un costo mínimo gracias al uso de software libre y gratuito. Con este trabajo no se pretende informatizar todos los procesos operativos de la Asociación de Scouts, pero se desea que sea solo el inicio de un proceso de automatización más grande.

El desarrollo del portal para el registro de miembros de la Asociación de Scouts tiene como referencia conceptos teóricos de la ingeniería de software y del diseño de bases de datos. Por lo tanto, en el contenido de este documento se hace una reseña histórica sobre la evolución en el proceso de desarrollo de software y se describen los fundamentos teóricos más importantes para la elaboración del presente trabajo.

El diseño de la base de datos se realiza aplicando los conceptos de modelo relacional, se presenta un modelo conceptual a los dirigentes de la Asociación de Scouts para determinar los requisitos de la base de datos. Una vez establecido el modelo lógico del diseño, se aplican las reglas del modelo relacional para obtener una base de datos normalizada.

Para la elaboración del sistema se hace uso de un desarrollo incremental e iterativo, siguiendo la filosofía de las metodologías ágiles de desarrollo de software. Por medio de la presentación de prototipos se permite a los representantes de la Asociación de Scouts obtener una mejor idea de la funcionalidad requerida y así afinar y mejorar los requisitos del software, al mismo tiempo que se descubren y señalan los fallos encontrados.

II. OBJETIVOS

A. Objetivo general

Proporcionar a la Asociación de Scouts de Guatemala un sistema de información que sirva de apoyo a las actividades relacionadas con el registro, formación, recreación, progresión y organización de sus miembros.

B. Objetivos específicos

Desarrollar un portal de Internet para el registro de información de sus miembros con las requeridas medidas de seguridad.

Diseñar y establecer una base de datos para el almacenamiento de forma estructurada y segura de la información de todos sus miembros. La base de datos contendrá una serie de registros variables en el tiempo acerca de sus miembros y su relación con la asociación.

Proveer una herramienta que permita obtener información de manera rápida y eficiente a través de listados según criterios predeterminados.

III. MARCO TEÓRICO

A lo largo de este capítulo se describen las disciplinas utilizadas para la conformación del marco teórico del portal de registro de miembros de la Asociación de Scouts de Guatemala. De estas disciplinas, la ingeniería de software y el diseño de bases de datos, se citan los conceptos con mayor influjo en la construcción del fundamento teórico de este trabajo.

A. Ingeniería de software

Existen diferentes tipos de software con objetivos específicos y ambientes de operación distintos, pero no importa que tan diversa sea su naturaleza, sus campos de aplicación o su uso, todos requieren de:

> «…la aplicación de un enfoque sistemático, disciplinado y cuantificable a su desarrollo, operación y mantenimiento; es decir, de la aplicación de la ingeniería al software.»
>
> IEEE[1], 1990:67

Para Pressman (2010), la ingeniería de software es un una tecnología estratificada, formada por un conjunto de herramientas, métodos y un proceso. Toda la estructura está apoyada en una base enfocada en obtener un resultado de buena calidad.

1. Proceso del software. El proceso es el agente coordinador de las herramientas y las metodologías para la obtención puntual de los productos en cada etapa. Feiler y Humphrey (1992:7) definen un proceso como:

> «…el conjunto de etapas parcialmente ordenadas con la intención de alcanzar un objetivo.»

En el caso de desarrollo de software el objetivo es la elaboración de un producto de software. El proceso de software, también conocido como el ciclo de vida del software, incluye todas las actividades

[1] Instituto de Ingenieros Eléctricos y Electrónicos

necesarias para su desarrollo, estas pueden variar en número y en su especificidad, pero al más alto nivel deben existir al menos las siguientes etapas:

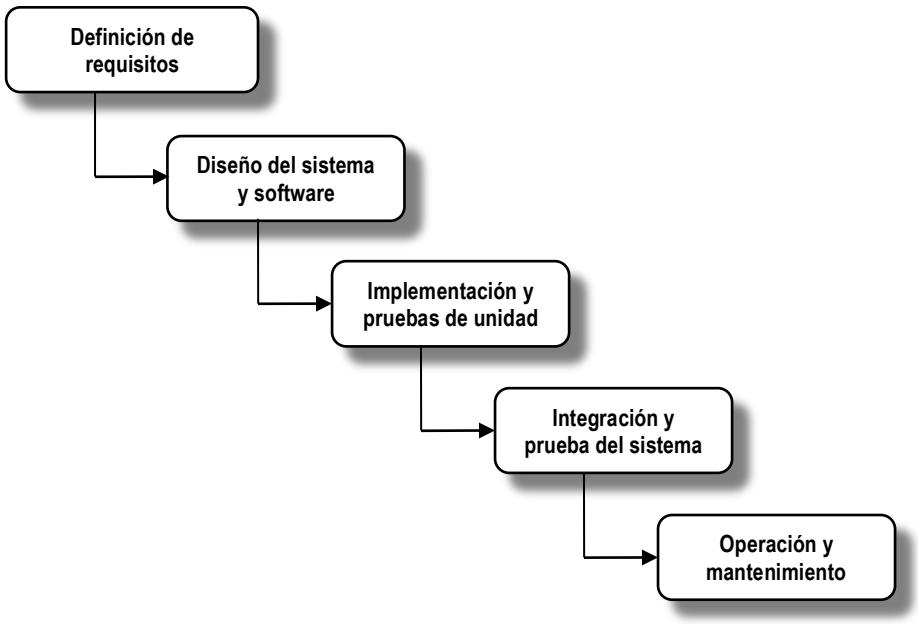
- Especificación de requisitos
- Elaboración e implementación
- Validación y pruebas
- Transición y mantenimiento

Los modelos de proceso de desarrollo definen el orden de las actividades involucradas para la producción del software así como su coordinación e interacción entre ellas. Los diversos modelos se dan según la importancia que adjudican a estas actividades y al flujo de trabajo que definan entre ellas. Así, hay aquellos que conocidos como procesos tradicionales, hacen énfasis en una planificación detallada, definiendo rigurosamente cada una de las actividades y sus resultados. Los otros, los procesos ágiles, en cambio, cuentan con una planificación más flexible, con definiciones simplificadas, enfocados a la realización del software en plazos más cortos, sin por ello sacrificar la calidad del producto e instando a una comunicación continua entre el usuario y el equipo de desarrollo.

a. Procesos tradicionales. Entre los modelos de los procesos de desarrollo de software tradicionales vale la pena mencionar los siguientes:

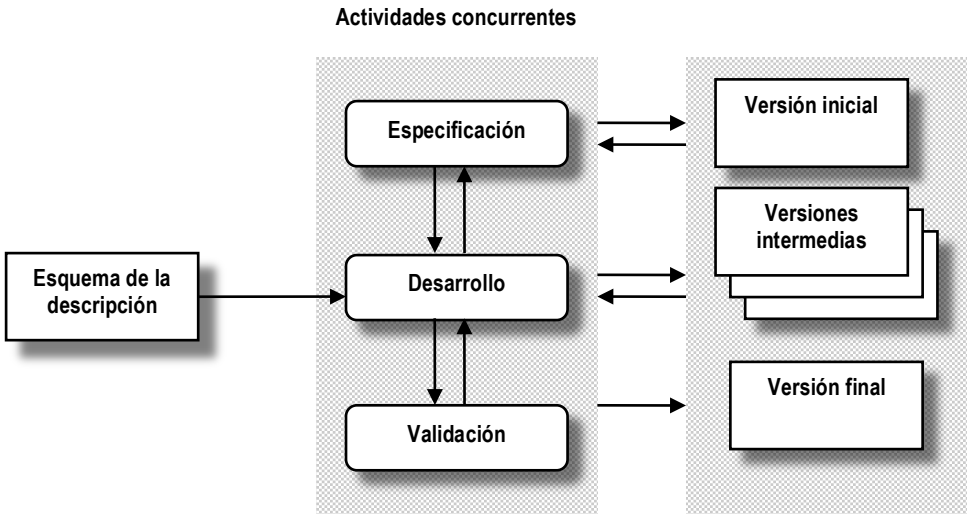
1) Modelo en cascada. Conocido también como el modelo clásico o tradicional, sugiere seguir un enfoque secuencial, con la previa culminación de cada etapa antes de proseguir con la etapa siguiente, como se muestra en la Figura 1. Esto requiere la planificación y definición exacta de cada actividad. No obstante, altamente criticado, debido a su rigidez frente a la naturaleza evolutiva del software y la falta de interacción entre el programador y el usuario, es aún bastante utilizado (Davidson, 2008).

Figura 1: Modelo en cascada



2) Modelo incremental . La idea detrás de este modelo es la de desarrollar un producto básico y mostrarlo al usuario para su evaluación y así afinarlo a través de varias versiones (iteraciones) hasta llegar al producto final. En la Figura 2 puede observarse como las actividades de especificación, desarrollo y validación están intercaladas de manera que se ejecutan repetidamente con cada versión hasta la obtención del producto final. Dentro del modelo incremental e iterativo se pueden citar dos enfoques. El primero, un enfoque donde no hay incertidumbre, el cual parte de especificaciones y requisitos bien establecidos a partir de los cuales se construyen módulos en plazos determinados que incrementan la funcionalidad del software. El segundo, es un enfoque más evolutivo, en el cual los requisitos van siendo descubiertos a medida que se construyen los módulos y por lo tanto las observaciones del usuario son primordiales para establecer dichos requisitos. A pesar de sus diferencias, ambos enfoques buscan evitar el paso único y secuencial por las etapas mostradas en el modelo en cascada (Larman y Basili, 2003:47). La filosofía de estos modelos es también la filosofía adoptada por los procesos de desarrollo ágiles (Pressman, 2010).

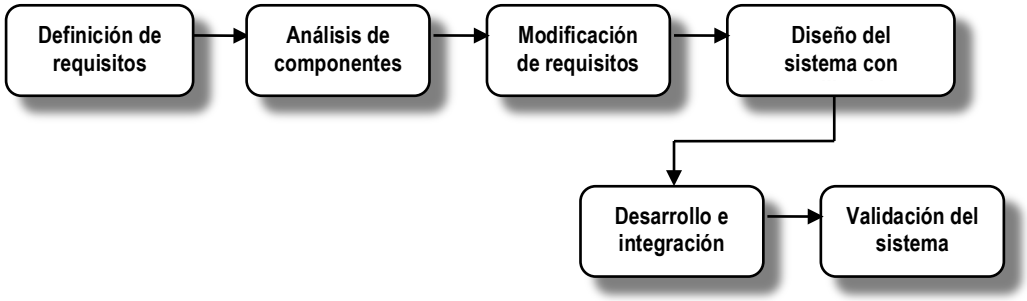
Figura 2: Modelo incremental



3) Modelo basado en componentes . Este modelo se basa en la reutilización de software, debido a que ciertas tareas o funciones pueden estar duplicadas en aplicaciones totalmente diferentes. Sommerville (2011) describe cuatro fases importantes en el desarrollo basado en componentes o de reutilización, que se ilustran en la Figura 3, estas son:

- Análisis de componentes.
- Modificación de requisitos.
- Diseño del sistema con reutilización.
- Desarrollo e integración.

Figura 3: Modelo basado en componentes



En la práctica corriente de la ingeniería de software existen una gran variedad de modelos de desarrollo. Debido a que no existe un proceso universal aplicable a todos los proyectos de desarrollo de

software, la ingeniería de software continua evolucionando y proponiendo modelos que reúnan características que mejor se adapten a las nuevas tecnologías y a los modernos entornos de desarrollo.

b. Procesos ágiles. Estos resultan como una reacción a la rigurosa tarea de planificación, definición detallada de actividades, productos y la extensa documentación obtenida de los métodos tradicionales. Sommerville (2011) menciona que en algunos tipos de proyectos, tales como el desarrollo de sistemas críticos o los que involucran a un gran número de equipos de desarrollo, el uso de métodos basados en una planeación exhaustiva puede estar justificado, sin embargo, su aplicación en proyectos de pequeña o mediana escala, incrementa los gastos generales de manera desproporcionada.

Además, tratar de aplicar procesos convencionales, los cuales requieren una extensa labor inicial para determinar los requisitos, en ambientes donde estos son volátiles o difíciles de establecer, podría resultar con la entrega de un software obsoleto (Sommerville, 2011:57).

Durante la década de 1990 empiezan a surgir modelos de desarrollo con procesos más ligeros, más fáciles de adaptar, dirigidos por pruebas y ciclos de desarrollo más cortos. La idea de desarrollo incremental e iterativo empezaba a ganar notoriedad (Larman y Basili, 2003:53). En 2001, en un taller organizado en Utah, Estados Unidos de América, participan diecisiete expertos de la industria del software, incluyendo creadores de algunas de las metodologías "ligeras" y se redacta el Manifiesto Ágil[2] (Fowler, 2005), una declaración que resume los principios y valores comunes de los procesos ágiles. En este se declara valorar:

- Al individuo y las interacciones sobre procesos y herramientas,
- Software que funcione sobre documentación detallada,
- Colaboración con el cliente sobre negociación de contratos,
- Responder al cambio más que seguir un plan.

De los valores mencionados en el manifiesto se desprenden doce principios fundamentales que sirven actualmente de base a cualquier proceso ágil de desarrollo:

- «Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
- Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al período de tiempo más corto posible.
- Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.

[2] http://agilemanifesto.org/iso/es/

- Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
- El software funcionando es la medida principal de progreso.
- Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
- La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
- La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
- A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.»

Otro de los frutos que resultó del esfuerzo de muchos de los participantes de esta reunión fue la creación de la Alianza Ágil[3], una organización sin fines de lucro que pretende fomentar y estudiar los conceptos de las metodologías ágiles. La Alianza Ágil organiza anualmente una conferencia que reúne a miembros e interesados con el fin de fomentar el intercambio de ideas y mejores prácticas en el área de desarrollo ágil.

Los procesos ágiles de software que existen en la actualidad son varios y no todos cumplen rigurosamente con los doce principios anteriormente mencionados pero todos siguen un enfoque incremental e iterativo para una o más etapas del ciclo de vida de desarrollo de sistemas. Entre los de más renombre están:

1) Programación extrema. También conocido como XP (por sus siglas en inglés). Es una metodología creada por Beck en 1996 (Beck, 2000) y llamada así por llevar las mejores prácticas, como el desarrollo iterativo, a niveles extremos, está basada en cuatro métodos esenciales: la comunicación continua entre el cliente y el equipo de desarrollo, simplicidad en las soluciones propuestas, las reacciones y observaciones de los usuarios ante cada iteración del producto y valor por parte del equipo de desarrollo para aceptar fácilmente los cambios.

Entre las principales prácticas que caracterizan la programación extrema están:

- El desarrollo se realiza haciendo entregas pequeñas y frecuentes del sistema. Los requisitos se establecen por medio de las historias de usuario las cuales sirven de base para agregar nuevas funciones a cada iteración.
- Las pruebas se hacen de forma automatizada en cada iteración y se diseñan antes de escribir el código.

[3] http://www.agilealliance.org

- Diseño simple para únicamente satisfacer los requisitos de cada etapa. Mantenerlo simple (KIS, por sus siglas en inglés) es el principio que rige el ejercicio del diseño en la programación extrema.
- Los programadores trabajan en pareja, verificando y sirviendo de apoyo a lo que hace cada uno, siguiendo estrictos estándares de programación.
- La re-fabricación constante del código para conservarlo simple y fácil de mantener.
- La presencia y completa disponibilidad del cliente para comunicar sus observaciones al equipo de desarrollo y establecer los requisitos.

2) Scrum. Es una metodología desarrollada a principios de la década de 1990 por Jeff Sutherland y su equipo de desarrollo. En 1995, Sutherland y Ken Schwaber formalizan el proceso al presentarlo en OOPSLA'95[4]. Después en 1999 Mike Beedle contribuyó con la estandarización de Scrum.

Basada en un concepto que había sido presentado en un investigación realizada en compañías multinacionales en los Estados Unidos de América y Japón (Nonanka y Takeuchi, 1986). En este estudio se describe la estrategia para administrar el desarrollo de nuevos productos que había sido adoptada por estas compañías para incrementar la rapidez y flexibilidad de producción y se le compara con la estrategia de rugby donde todo el equipo intenta avanzar en el campo de juego como una unidad pasándose el balón unos a otros.

Scrum es un proceso de administración del desarrollo incremental e iterativo que no establece prácticas específicas de la ingeniería de software, es por eso que puede aplicarse a distintas áreas como la administración de proyectos, en general. También se puede combinar con otras metodologías ágiles como la de programación extrema en el área de desarrollo de software.

En Scrum el proceso de desarrollo se lleva a cabo con una sucesión de iteraciones cortas, de duración fija (generalmente de 2 a 4 semanas), denominadas *sprints*. Como resultado de cada *sprint* se obtiene un producto de software funcional que se muestra al usuario. Al inicio de cada *sprint* se escogen de una lista general los requisitos que deben ser incluidos en este ciclo, los cuales no cambian durante este periodo. Cada día se organizan reuniones de no más de 15 minutos para evaluar el avance, descubrir problemas potenciales y así poder adaptar la estrategia y métodos.

Al final del *sprint* se presenta el sistema obtenido al usuario y se recogen sus observaciones y comentarios para establecer requisitos que deban ser integrados en el ciclo siguiente.

[4] Una conferencia anual de la ACM sobre programación, sistemas, lenguajes y aplicaciones orientados a objetos. Por su alto enfoque académico ha sido de suma importancia para la evolución del desarrollo de software. Es aquí donde se presentan y dan a conocer las tecnologías emergentes de la ingeniería de software. A partir de 2010, forma parte de una conferencia más amplia llamada SPLASH.

Un proyecto concluido es visto como una secuencia de *sprints* que producen el software de forma gradual hasta obtener el resultado final. Este enfoque permite abordar proyectos cuyos requisitos son impredecibles o que pueden cambiar en cualquier momento y parte del principio que el problema a solucionar no esté del todo definido.

Existen otras metodologías ágiles entre las cuales algunas fueron creadas por los signatarios del manifiesto ágil y por lo tanto coinciden en filosofía y principios.

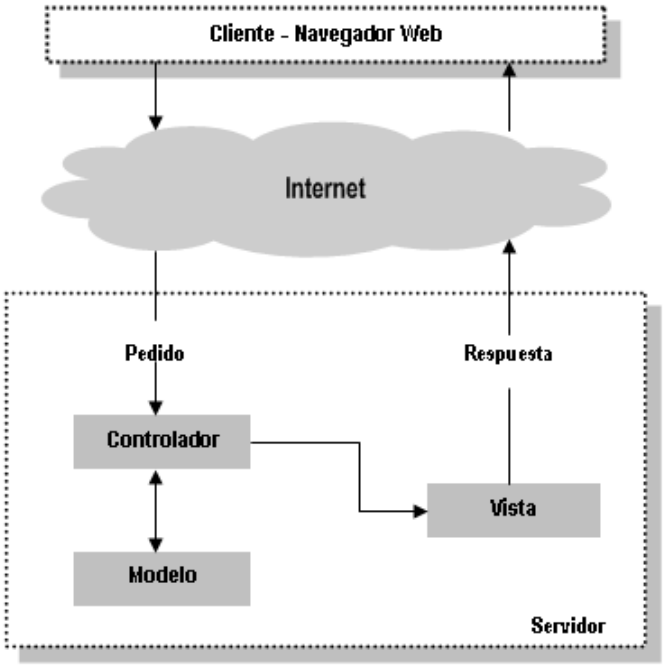
2. Arquitectura del software. Se define la arquitectura del software como el establecimiento de la estructura de un sistema de software por medio de componentes o elementos, y sus relaciones entre sí. Esta estructura debe ser diseñada tomando en cuenta las funciones del software.

La ingeniería de software nos proporciona diversos patrones de diseño, de manera que se puedan adoptar soluciones ya probadas a problemas encontrados previamente en el diseño de software. Es así como el patrón Modelo-Vista-Controlador (MVC) se menciona frecuentemente en la literatura, la cual enumera y examina los distintos patrones de diseño. El patrón MVC juega un rol importante dentro del contexto del presente trabajo debido a su uso generalizado en el desarrollo de sistemas web, a tal punto, que han surgido numerosas plataformas (*framework,* en inglés) que implantan dicho esquema para su desarrollo. Tal es el caso de *symfony*, la plataforma de desarrollo empleada para la elaboración del portal para el registro de scouts.

En la Figura 4 se ilustra de forma simplificada la arquitectura clásica de una aplicación web y su naturaleza de ejecución distribuida. Esto quiere decir que, por un lado, tenemos software que se ejecuta en el cliente y que interactúa con el usuario (el navegador) y por el otro, un código que se ejecuta en el servidor y que interactúa con la información almacenada o datos (llamado simplemente servidor). El patrón MVC acá mostrado, es del lado del servidor[5] lo que significa que la ejecución de sus tres componentes se lleva a cabo en este.

[5] El patrón MVC existe también en el lado del cliente, ya sea incorporado en el navegador o bien a través de aplicaciones de Internet sofisticadas (RIA, por su nombre en inglés: *Rich Internet Applications*).

Figura 4: Arquitectura de una aplicación web



En el patrón MVC, el modelo representa la información con la cual trabaja el sistema, es decir, la información y las operaciones asociadas con estos (lógica de negocio). La vista es la que transforma el modelo en un formato tal que permita al usuario una fácil interacción. El controlador interpreta las acciones del usuario convirtiéndolas en notificaciones para la vista y el modelo, las cuales pueden a su vez, provocar cambios en estos dos últimos. Tal es el caso, por ejemplo, cuando un usuario presiona la función de 'borrar' en una lista de artículos. La acción se convierte en una orden al modelo, el cual va a suprimir el artículo de la base de datos y luego da una orden a la vista para actualizarse y no mostrar más el artículo eliminado.

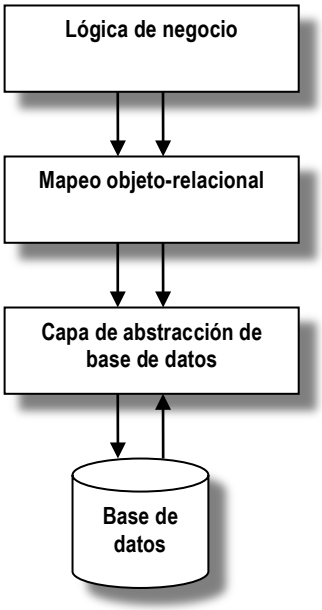
Así Fowler (2002), reflexionando acerca del patrón MVC, ve dos separaciones principales: la separación entre la vista y el modelo; y la separación entre la vista y el controlador. De estas, la primera es de fundamental importancia para el buen diseño de software (Fowler, 2002:330). Una aplicación que permita ser ejecutada desde diferentes dispositivos y con una clara separación entre la vista y el modelo, requerirá solamente la creación de las distintas vistas permitiendo emplear el mismo modelo y controlador.

En el esquema MVC, el modelo se muestra simplificado, cuando en realidad un vista más detallada, como se muestra en la Figura 5, permite ver las varias capas que lo componen. Partiendo de abajo hacia arriba, la primera capa son los datos, es decir, toda aquella información que persiste y que se encuentra almacenada, generalmente, en una base de datos relacional.

Luego, está la capa de abstracción de base de datos (DBAL, por sus siglas en inglés), la cual permite un primer nivel de abstracción de la base de datos proporcionando operaciones básicas en esta, tales como: crear, obtener, actualizar y borrar. Esta capa permite también obtener independencia con respecto a las particularidades de la base de datos[6] que se utilice, de tal manera que el resto de capas no requieran ninguna modificación si se cambia la base de datos.

A continuación se encuentra la capa de asignación objeto-relacional (ORM, por sus siglas en inglés), la cual tiene como función la conversión del modelo relacional de la base de datos al modelo orientado a objetos utilizados en los lenguajes de programación modernos, y viceversa. Por último, la lógica de negocio, la cual contiene y refleja las reglas establecidas por los requisitos del software. En términos prácticos, esta consiste en la definición de métodos en las clases que conforman el modelo conceptual.

Figura 5: Capas que conforman el modelo



B. Bases de datos

El concepto de base de datos tiene su origen en la necesidad de los sistemas de información de almacenar la información que transita por ellos de manera que pueda ser reutilizada. Se puede definir como un conjunto de datos almacenados de forma estructurada y que están relacionados entre sí, los cuales

[6] Desde un punto de vista puramente teórico, la capa DBAL permite la manipulación de datos de cualquier proveedor de datos, sea este una base de datos relacional, un archivo xml o un archivo texto.

cumplen con un modelo de datos. El modelo de datos es la representación de datos del mundo real, siguiendo un conjunto de conceptos y reglas, en un sistema informático. Existen diversos modelos de datos para el diseño de bases de datos, pero el más utilizado para sistemas de información es el modelo de datos relacional, inicialmente formulado por E.F. Codd en 1970 (Connolly y Begg, 2005:70). En los párrafos siguientes se tratará únicamente sobre los conceptos y diseño de bases de datos relacionales.

Un sistema de gestión de bases de datos (RDBMS, por sus siglas en inglés) es el software que permite el acceso a la base de datos, es decir que maneja las estructuras físicas y coordina el almacenamiento de los datos. Este esta compuesto de un lenguaje, por lo general y para las bases de datos relacionales es SQL (por su nombre en inglés: *Structured Query Language*), el cual permite realizar consultas y operaciones en la base de datos; pero también, de programas para afinar el desempeño, corregir irregularidades o recuperar información perdida.

1. Diseño de bases de datos. Las fases del ciclo de vida de la base de datos descritas por Teorey *et al*. (2006) son:

- El análisis de requisitos para la base de datos,
- El diseño lógico,
- El diseño físico,
- Implementación y modificación.

Aquí se limitará el estudio a las primeras dos fases, debido a que las otras dependen del sistema de gestión de bases de datos y de su implementación en la práctica.

El análisis de los requisitos para la base de datos consiste en identificar los datos necesarios para el funcionamiento del sistema y los datos que requieran ser almacenados. Esta actividad se realiza por medio de entrevistas entre las partes interesadas, tales como analistas, clientes y usuarios.

El diseño lógico requiere elaborar un diagrama conceptual que muestre cómo los datos identificados en el análisis de requisitos se relacionan entre sí. Este diagrama conceptual puede realizarse usando el modelo entidad-relación presentado por P. Chen en 1976. Teorey *et al*. (2006:13-20) mencionan y describen los tres elementos básicos que existen en el modelo entidad-relación y cuyas definiciones se resumen a continuación:

- Entidades: son aquellos elementos de los cuales queremos guardar alguna información y representan un objeto del mundo real. Para Chen (1976:10) entidad es «algo que puede ser identificado de forma inequívoca». Por ejemplo, ‘persona’ puede ser una entidad.

- Atributos: son las propiedades o características que definen la entidad. Un conjunto de atributos es lo que permite distinguir una entidad de otras del mismo tipo. Por ejemplo, los nombres y apellidos definen y distinguen a las entidades de tipo 'persona'. Se llama dominio al conjunto de los posibles valores que pueden ser asignados a un atributo (Connolly y Begg, 2005:351). Aquel atributo que permita identificar de forma inequívoca la ocurrencia de una entidad es denominado clave primaria. Cuando la clave está compuesta por un conjunto de atributos se le llama clave compuesta. En muchos casos es conveniente definir un atributo artificial, o sea que no forma parte de las características naturales del objeto representado por la entidad, para identificar la ocurrencia de la entidad de manera única y de esta forma ser utilizado como clave primaria.

- Correlaciones[7]: son las asociaciones que existen entre entidades. Estas están determinadas por el grado, cardinalidad y existencia. Se le llama grado al número de entidades que participan en una correlación, por lo general, suelen ser dos o tres. La cardinalidad está definida por el número de ocurrencias de las entidades involucradas en la asociación, entre estas se puede mencionar: una a una, una a varias o varias a varias (Connolly y Begg, 2005:363). La existencia es la que indica si la ocurrencia de la entidad es obligatoria u opcional.

2. Modelo relacional. Cuando el modelo conceptual se implementa usando el modelo relacional de bases de datos, es necesario cumplir con las reglas que dicho modelo relacional establece. Connolly y Begg (2005:72) definen que, en el modelo relacional, una relación es el conjunto de datos que se refieren a una entidad y están organizados de forma tabular, es decir contiene filas (registros) y columnas (campos o atributos). Es por esto que una relación también se denomina tabla.

a. Restricciones de integridad. La integridad impuesta en una base de datos se refiere a la calidad y exactitud de la información que esta contiene. Existen dos reglas principales en el modelo relacional (Connolly y Begg, 2005:81):

- Integridad de entidad: esta requiere que cada fila de una tabla pueda ser identificada de forma única. Esto implica la existencia de una clave primaria que no acepte valores duplicados y por lo tanto el valor nulo no es permitido.
- Integridad referencial: esta exige que los valores de una clave foránea (campo o conjunto de campos de una relación que es la clave primaria de otra relación) coincidan con los valores de la clave primaria a la que hace referencia.

[7] Se usa el término correlación en lugar de relación para evitar ambigüedades. Relación en el modelo relacional tiene otro sentido y es sinónimo de tabla.

b. Normalización. Codd (1970) define el concepto de normalización cuando menciona que «es necesario simplificar los dominios de las relaciones con el fin de almacenarlas en estructuras que puedan ser representadas por arreglos de dos dimensiones». Más adelante escribe: «De hecho, existe un procedimiento simple para eliminar los dominios no simples, el cual llamaremos normalización» (Codd, 1970).

La normalización de una base de datos consiste en una serie de reglas que pueden ser usadas para evaluar las tablas de forma individual y establecer si llenan los requisitos que estas imponen. Cuando un requisito no es cumplido por causa de una tabla, esta debe ser dividida en otras tablas de manera que estas cumplan individualmente con los requisitos de la normalización (Connolly y Begg, 2005: 401).

En términos más simples, la normalización asegura que no exista duplicación de información y que no resulten anomalías o inconsistencias en la base de datos al realizar en esta operaciones de creación, actualización o supresión.

El proceso de normalización se lleva a cabo aplicando transformaciones sucesivas denominadas 'formas normales'. En la práctica se aplican las primeras tres 'formas normales', las cuales se detallan a continuación:

Primera forma normal (1FN): Una relación esta en primera forma normal si todos sus atributos son atómicos, es decir, que la intersección de filas y columnas de una tabla contiene un solo valor y nunca una lista de valores (Teorey et al., 2006:109).

Segunda forma normal (2FN): Una relación esta en segunda forma normal si está en primera forma normal y se han eliminado las dependencias parciales (Connolly y Begg, 2005:402). Una dependencia parcial es cuando los atributos no clave dependen solo de una parte de una clave compuesta. El cumplimiento de esta forma normal implica eliminar los atributos con dependencias parciales para que formen parte de una nueva relación.

Tercera forma normal (3FN): Una relación esta en tercera forma normal si está en segunda forma normal y se han eliminado las dependencias transitivas (Connolly y Begg, 2005:402). Es decir, que cada atributo que no formen parte de la clave debe depender únicamente de la clave primaria. Una dependencia transitiva existe cuando un atributo que no forma parte de la clave depende de uno o más atributos que no forman parte de la clave.

IV. ESTRUCTURA DEL SISTEMA

La estructura del portal para registro de miembros de la Asociación de Scouts de Guatemala puede describirse desde dos perspectivas. La primera, una estructura física, concierne la organización del código fuente, archivos de configuración y otros, la cual es establecida por la plataforma de desarrollo *symfony*[8]. La segunda, una estructura conceptual, corresponde al resultado del análisis del problema siguiendo una metodología determinada y está formada por componentes que, interrelacionados por operaciones, establecen la lógica de negocio.

A. Estructura física

Un sistema desarrollado en *symfony* adopta una estructura arborescente para la organización de directorios y archivos que lo conforman. Así, al nivel de la raíz esta lo que se denomina un proyecto *symfony*, este está formado por una o varias aplicaciones y estas a su vez abarcan uno o varios módulos.

1. Proyecto. Un proyecto en *symfony* se define como:

> «un conjunto de operaciones y servicios disponibles bajo un mismo dominio y compartiendo el mismo modelo [del patrón MVC].»
>
> (Zaninotto y Potencier, 2010)

La Figura 6 ilustra la estructura de directorios existente en un proyecto de *symfony* (Zaninotto y Potencier, 2010:30).

[8] El término *symfony* se utiliza a lo largo de este documento para referirse a la versión 1 de *symfony* (http://www.symfony-project.org), específicamente a la version 1.4, la cual fue empleada en el desarrollo del presente trabajo. La versión más reciente de *symfony* es la 2 por lo que se usa el término *symfony2* para referirse a ella.

Figura 6: Estructura de directorios de un proyecto *symfony*

En la Tabla 1 (Zaninotto y Potencier, 2010:30) se describe la función de los directorios más importantes anteriormente delineados.

Tabla 1: Descripción de directorios de un proyecto *symfony*

Directorio	**Descripción**
apps/	Contiene un directorio por cada aplicación definida en el proyecto. En symfony, generalmente, son dos: 'frontend' y 'backend'. En el portal de registro de scouts solo se ha sido definido una: 'admin', para la administración y gestión del registro de miembros con la idea que en un futuro forme parte del sitio Internet de la Asociación de Scouts de Guatemala.
config/	Contiene los archivos de la configuración general del proyecto.
data/	Contiene los archivos relacionados con la definición de la base de datos, tal como las órdenes SQL para crearla.
doc/	Directorio predeterminado para guardar la documentación.
lib/	Sirve para colocar las librerías externas requeridas por el proyecto. Clases comunes a todas las aplicaciones se encuentran también en este directorio.
plugins/	Contiene los módulos accesorios utilizados por el proyecto. Ver más adelante para una descripción detallada.
web/	Es el directorio raíz para el servidor web. Es el único directorio (y los archivos aquí contenidos) que se puede acceder a través Internet.

2. Aplicaciones. Las aplicaciones se encuentran al interior de un proyecto. Estas se pueden ejecutar, por lo general, independientemente unas de otras, pero compartiendo la misma base de datos. La estructura de directorios al interior de una aplicación *symfony* se ilustra en la Figura 7 (Zaninotto y Potencier, 2010:31).

Figura 7: Estructura de directorios de una aplicacíon *symfony*

La Tabla 2 (Zaninotto y Potencier, 2010:31) describe la función de cada uno de los directorios ubicados al interior de una aplicación *symfony*.

Tabla 2: Descripción de directorios de una aplicación *symfony*

Directorio	**Descripción**
config /	Contiene la mayoría de archivos de configuración de la aplicación. Los parámetros predeterminados pueden ser modificados dentro de estos archivos.
i18n/	Contiene los archivos que sirven para la internacionalización de la aplicación, principalmente los archivos para la traducción de mensajes y textos del interfaz de la aplicación.
lib/	Contiene las librerías y clases específicas a la aplicación.
modules/	Directorio para guardar los distintos módulos que componen la aplicación.
templates/	Contiene las plantillas usadas por la aplicación y que son compartidas por todos sus módulos. Contiene un archivo predeterminado 'layout.php', el cual define el estilo de página principal en el que se agregan las demás plantillas.

3. Módulos. Un conjunto de módulos forman una aplicación *symfony*. Un módulo equivale a una página web o un grupo de páginas web con un mismo objetivo. Los módulos contienen acciones, las cuales representan las operaciones que pueden ser ejecutadas en un módulo. La estructura de directorios de un módulo *symfony* está representada en la Figura 8 (Zaninotto y Potencier, 2010:32).

Figura 8: Estructura de directorios de un módulo *symfony*

```
apps/
  [nombre de la aplicación]/
    modules/
      [nombre del módulo]/
          actions/
            actions.class.php
          config/
          lib/
          templates/
            indexSuccess.php
```

4. Módulos accesorios. La plataforma de desarrollo *symfony* permite la reutilización de software a través de módulos accesorios o *plugins*. Estos son normalmente puestos a disposición por terceras personas o por el equipo de desarrollo de *symfony* para su uso libre y gratuito, permitiendo ganar tiempo y esfuerzo. A continuación se resumen algunos de dichos módulos empleados por el portal de registro de scouts.

- sfDoctrineGuardPlugin 1.0.4: este módulo provee las funciones de seguridad a las aplicaciones *symfony*. Permite la creación y modificación de los usuarios del sistema, así también permite la creación y modificación de permisos, los cuales permiten restringir el acceso a los usuarios a realizar ciertas operaciones. Las funciones de conexión y desconexión del sistema son facilitadas por este módulo.

- sfTCPDFPlugin 1.6.3: facilita un interfaz para el acceso a las funciones de la librería TCPDF[9], la cual permite la generación de documentos PDF por medio de clases escritas en PHP. Este módulo es utilizado para la generación de informes.

- ioMenuPlugin 0.7.0: permite la creación y manejo de menús en *symfony* de manera simplificada.

- sfThumbnailPlugin 2.0.1: provee funciones para la creación de fotografías en miniatura.

- stOfcPlugin 1.0.2: facilita un interfaz para el acceso a las funciones de la librería OFC[10] (por las siglas en inglés de Open Flash Chart), la cual permite generar gráficas estadísticas a partir de un arreglo de datos.

[9] http://www.tcpdf.org
[10] http://teethgrinder.co.uk/open-flash-chart/

B. Estructura conceptual

En la Figura 9 se puede observar el esquema general del portal, que consiste en 16 módulos principales. También se muestran las operaciones o acciones que pueden realizarse en cada uno de los módulos. Con el fin de poder realizar estas operaciones es necesario que los usuarios tengan los permisos requeridos para cada módulo.

1. Roles. Inicialmente se han identificado dos tipos de usuarios:

- Administrador: es aquel que tiene acceso a todas las operaciones existentes en todos los módulos. Es este el único con acceso a las operaciones de administración, tales como gestión de usuarios, asignación de permisos; y a las funciones de configuración, es decir a la gestión de los datos de referencia..

- Operador: tiene accesos únicamente a las operaciones de registro del menú principal, las cuales permiten el registro de scouts, la gestión de inscripciones anuales y las inscripciones a eventos y cursos.

2. Módulos. A medida que se establece el modelo conceptual, siguiendo la metodología que será descrita en el capítulo siguiente, se logra definir una serie de módulos reunidos en tres grupos:

a. Módulos de configuración. Son los módulos con las funciones de mantenimiento de los datos de referencia, es decir, los datos que sirven de base a otros datos. A continuación se ofrece una descripción de cada uno de estos módulos.

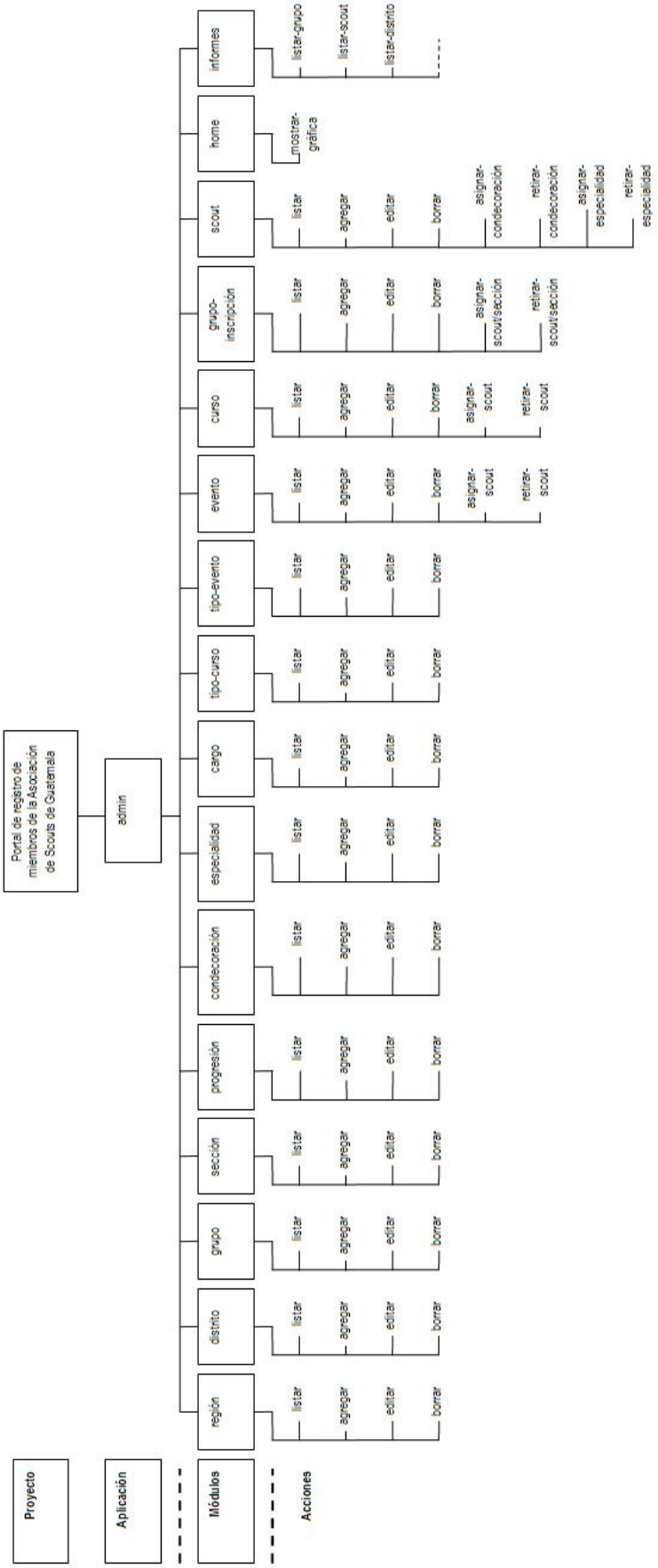
1) Regiones. Permite la creación o modificación de las regiones en las cuales está dividida la Asociación de Scouts de Guatemala. Los Procedimientos, Organización y Reglamentos (P.O.R.) de la Asociación de Scouts de Guatemala la definen de la siguiente manera:

> «La Región Scout es la conformación de varios distritos scout, en su (sic) sector geográfico determinado, para promover el desarrollo de la Asociación de Scouts de Guatemala y la aplicación del Programa Scout.»

2) Distritos. Permite la creación de nuevos distritos o la modificación o supresión de los distritos ya existentes. Un distrito pertenece a una región. Los P.O.R. le dan la siguiente definición:

> «Es la conformación de varios Grupos Scouts, en un sector geográfico determinado para promover el desarrollo de la Asociación de Scouts de Guatemala, y la aplicación del Programa Scout; está compuesto de un mínimo de cuatro (4) y un máximo de siete (7) Grupos, coordinado por un Comisionado de Distrito.»

Figura 9: Esquema general del portal de registro de scouts



3) Grupos. Permite la creación de nuevos grupos de scouts, o la modificación o supresión de grupos ya existentes. Un grupo pertenece a un distrito. Los P.O.R. lo definen así:

> «El Grupo Scout, es el organismo encargado de aplicar el Programa Scout.
> El Grupo Scout, está formado por un conjunto de : Muchachos, Muchachas y Dirigentes Adultos; registrados en la Asociación, y organizado de acuerdo con éste Reglamento.»

4) Secciones. Permite la creación de nuevas secciones, o la modificación o supresión de secciones ya existentes. Una o más secciones existen en un mismo grupo. De acuerdo con los P.O.R. la sección es: «un conjunto de muchachos (as) agrupados según edades, en unión con Scouters[11].»

5) Progresiones. Permite la creación de nuevas progresiones, o la modificación o supresión de las progresiones ya existentes. La progresión representa un adelanto o superación de cada miembro dentro de una sección.

6) Condecoraciones. Permite la creación de nuevas condecoraciones, o la modificación o supresión de las ya existentes. Las condecoraciones son reconocimientos otorgados a los miembros por mérito.

7) Especialidades. Permite la creación de nuevas especialidades, o la modificación o supresión de las ya existentes. Las especialidades son habilidades que desarrollan los miembros, según sus intereses, dentro de una sección.

8) Cargo. Permite la creación de nuevos cargos, o la modificación o supresión de los ya existentes. Los cargos son las funciones que pueden desempeñar, tanto jóvenes como adultos, dentro del organigrama de la Asociación de Scouts de Guatemala.

Un miembro puede tener como máximo dos cargos simultáneamente, excepto los siguientes, que sólo pueden tener uno:

- Jefe Scout Nacional
- Sub-Jefe Scout Nacional
- Miembros de Consejo del Consejo Scout Nacional
- Miembros de la Corte Nacional de Honor

[11] Adulto a quien la Asociación de Scouts le ha conferido un cargo.

9) Tipo de cursos. Permite la creación de nuevos tipos de cursos, o la modificación o supresión de los ya existentes. Los tipos de curso son aquellos que definen los diversos cursos que se imparten en el seno de la Asociación de Scouts sin tomar en consideración el lugar o fecha. Estos cursos existen para la formación de los miembros que quieran optar a un cargo, como parte de su plan de adelanto o para difundir los principios fundamentales del escultismo.

10) Tipo de eventos. Permite la creación de nuevos tipos de eventos, o la modificación o supresión de los ya existentes. Los tipos de evento representan las distintas actividades organizadas por la Asociación para el desarrollo y recreación de sus miembros independientemente del lugar o fecha cuando se llevan a cabo. La realización de estas actividades permite la aplicación del programa scout.

b. Módulos de registro. Los módulos de registro, hacen uso de los datos representados por los módulos anteriormente mencionados para realizar la operaciones de mantenimiento en el registro de scouts y otras actividades. Estos módulos se describen a continuación:

1) Eventos. Este módulo permite las operaciones de creación y mantenimiento de eventos organizados en un lugar y fecha determinados, para ello propone una lista con los tipos de eventos definidos en el módulo 'Tipo de evento' de la cual el usuario debe escoger uno. Al mismo tiempo, permite asignar (o retirar) los scouts que tomarán parte de dicha actividad.

2) Cursos. De manera semejante al módulo anterior, este permite las operaciones de creación y mantenimiento de los cursos impartidos en un lugar y fecha determinados. Este también propone una lista con los cursos definidos en el módulo 'Tipo de cursos' para poder escoger uno de ellos. La asignación (o retiro) de los scouts que se han inscrito para participar en dicha capacitación se realiza a través de este módulo.

3) Inscripciones. El módulo de inscripciones es ligeramente más complicado que los hasta aquí expuestos, debido a que además de permitir las operaciones de creación y mantenimiento de las inscripciones de grupos de scouts, estos últimos deben agruparse por secciones. Este módulo permite asignar o retirar scouts en las respectivas secciones de un grupo. Los scouts son propuestos por listas como se mencionó en el caso de tipos de eventos y cursos, por lo cual es necesario que los miembros sean preliminarmente creados en el módulo 'Scouts', descrito a continuación.

4) Scouts. Este módulo es el que acepta el ingreso de nuevos miembros scouts con toda su información personal requerida durante el establecimiento de los requisitos del software. Las operaciones realizadas en este módulo son de creación y mantenimiento de miembros scouts, además de la asignación de condecoraciones, progresiones y especialidades obtenidas durante su participación en el escultismo.

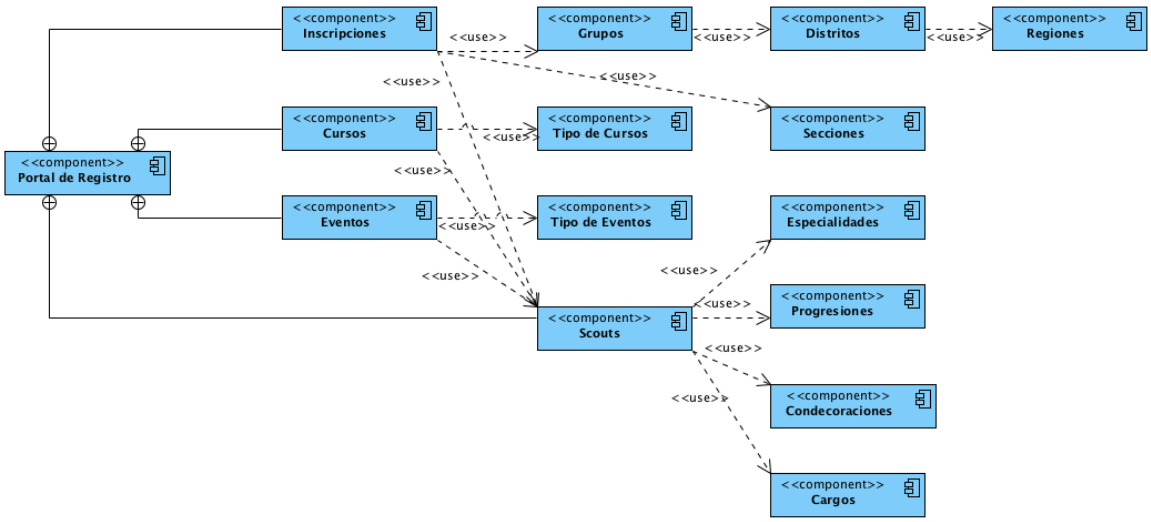
c. Otros módulos. Finalmente, dos módulos con funciones muy específicas:

1) Bienvenida. Este módulo tiene como función la de servir de área predeterminada una vez el usuario realiza una conexión con éxito. Al ejecutarse la única acción definida en este módulo muestra una gráfica circular representando el número de miembros inscritos por sección.

2) Informes. Este módulo presenta las diversas opciones para la generación de informes definidos en el sistema, con el fin de presentar de una forma clara y útil los datos almacenados en la base de datos.

Los módulos anteriormente descritos y su relación entre sí, se resumen en la Figura 10, la cual representa el diagrama de componentes del modelo conceptual del portal de registro de scouts.

Figura 10: Diagrama de componentes del modelo conceptual del portal de registro de scouts



V. METODOLOGÍA Y DESARROLLO

Con el fin de conocer la problemática presentada por el proceso de registro de miembros de la Asociación de Scouts de Guatemala y comenzar a obtener elementos que permitan la identificación y establecimiento de los requisitos del software, se debieron resolver los siguientes objetivos preliminares:

- Estudiar y analizar el proceso manual de registro de miembros.
- Familiarizarse con la organización y actividades de la Asociación de Scouts de Guatemala. Discernir las diferentes agrupaciones en el seno de la Asociación.
- Enumerar los datos recolectados en el proceso de registro e identificar aquellos que deben ser agregados en el sistema automatizado.
- Seleccionar la plataforma de utilización del producto final.

Estos puntos fueron resueltos por medio de una entrevista inicial con los directivos de la Asociación de Scouts de Guatemala, concretamente con el Jefe Scout Nacional y Subjefe Scout Nacional. En esa ocasión se obtuvieron algunos documentos relativos al funcionamiento de la Asociación y al registro de scouts, a saber:

- Procedimientos, Organización y Reglamentos de la Asociación de Scouts de Guatemala (P.O.R.),
- Ficha de inscripción 2011 en formato Excel,
- Software desarrollado por la Asociación de Scouts de Estonia y proporcionado por la Oficina Scout Mundial con sede en Ginebra, Suiza y
- Nota preparada por el Subjefe Nacional Scout detallando las diferentes secciones, cargos, plan de adelanto, especialidades de la Asociación de Scouts. También se definían los datos requeridos para el registro de los miembros y bosquejos de algunos informes requeridos.

A. Evaluación del software obtenido de la Oficina Scout Mundial

Como parte del análisis del sistema de registro de scouts, se procedió a la instalación de un software obtenido de la Oficina Scout Mundial. Una vez instalado, se organizó una teleconferencia con los directivos de la Asociación de Scouts para evaluar sus prestaciones y funcionalidad.

Fue obvio que el software no cumplía a cabalidad con las necesidades expresadas por los representantes de la Asociación de Scouts de Guatemala, por lo cual se indicaron una serie de modificaciones necesarias para ser adaptado a los requisitos locales. Luego de una evaluación técnica detallada, se concluyó que era mejor diseñar un sistema nuevo según las exigencias enunciadas por los directivos. A continuación se presentan algunas de las limitaciones descubiertas durante la evaluación:

- El software no fue desarrollado con un enfoque orientado a objetos.
- El uso de la librería *Smarty*[12] sólo se ocupa de la 'vista' y deja de lado el 'modelo' y el 'controlador' del patrón MVC haciendo su mantenimiento más laborioso.
- El diseño sumamente simple de la base de datos, a tal extremo que varios datos se encuentran definidos dentro del código fuente.
- La falta de una capa de abstracción de base de datos.

B. Desarrollo del nuevo software

La decisión de desarrollar un nuevo software para el portal de registro de scouts implica adoptar una metodología adecuada que permita, según las condiciones del entorno de trabajo, alcanzar los objetivos propuestos inicialmente. Teniendo en cuenta que la Asociación de Scouts de Guatemala no contaba anteriormente con un sistema automatizado y debido a la distancia geográfica entre el autor[13] y los representantes de la Asociación de Scouts de Guatemala, un desarrollo incremental e iterativo considerando los preceptos de la programación extrema se presentaba como la estrategia más adecuada a adoptar. De igual manera, Sommerville (2011:13), considera el enfoque de las metodologías ágiles como el más apropiado para el desarrollo de aplicaciones basadas en Internet. Sin embargo, no existe un equipo completo de programación y la distancia geográfica y los diferentes husos horarios de los involucrados en el proyecto, obligaron a adaptar algunas de las consignas de la programación extrema o simplemente no pudieron ser adoptadas.

Entre las características de las metodologías ágiles empleadas con más frecuencia[14] en el ámbito de desarrollo de software y que fueron aplicadas, se pueden citar las siguientes:

- Planificación iterativa
- Entregas pequeñas y periódicas
- Integración continua
- Pruebas unitarias

[12] http://www.smarty.net
[13] El autor reside en Ginebra, Suiza.
[14] Según la encuesta de http://www.versionone.com/state_of_agile_development_survey/11/

- Re-fabricación del código
- Estándares de programación

Para el establecimiento de las etapas a seguir para el desarrollo del portal para el registro de scouts, se usó como base las definidas en la programación extrema (Letelier y Penadés, 2006).

1. Exploración. En esta etapa se investiga sobre las tecnologías y herramientas a utilizar, las cuales deben pertenecer a la categoría de software libre con el fin de mantener los costos al mínimo y que la Asociación de Scouts de Guatemala no incurra en gastos adicionales. Así mismo, se discute a grandes rasgos los requisitos generales del software.

a. Herramientas. Luego de una breve indagación sobre las diversas herramientas disponibles de forma libre para la elaboración de aplicaciones Web, se optó por *symfony* como plataforma de desarrollo. Entre las características que motivaron su selección están:

- Conforme con la mayoría de mejores prácticas para el desarrollo de aplicaciones web.
- Implantación del patrón MVC.
- Enfocado a la aplicación de metodologías ágiles.
- Desarrollo orientado a objetos.
- Software libre.
- Excelente documentación y una amplia comunidad de desarrolladores.
- Incorporación de mapeo objeto-relacional (Doctrine 1.2).
- Escrito completamente en PHP, haciéndolo apto para funcionar en la mayoría de sistemas operativos existentes[15].
- Compatible con una gran variedad de manejadores de bases de datos.

El manejador de bases de datos seleccionado es *MySQL*, uno de los manejadores de uso libre más utilizado en las aplicaciones de Internet por su simplicidad y facilidad de configurar. Debido a su inmensa popularidad, su desarrollo continúa mejorando sus características de base de datos relacional, permitiéndole introducirse en el mercado corporativo. La firma Forrester Research Inc., una de las empresas líderes en investigación de mercado, considera *MySQL* un base de datos apropiada para el mercado de pequeñas y medianas empresas (Yuhanna, 2009).

[15] El desarrollo del sistema se realizó usando el sistema operativo OS X de Apple Inc., a medida que se producían nuevas versiones del software se desplegaban al servidor de pruebas, el cual funciona usando el sistema operativo Linux Ubuntu.

b. Establecimiento de requisitos generales. Con base a lo discutido en la primera reunión con los directivos de la Asociación de Scouts de Guatemala y el estudio de los documentos obtenidos, se puede establecer el siguiente enunciado:

> *Desarrollar un sistema de registro de miembros de la Asociación de Scouts de Guatemala para poder contar de forma electrónica con la información personal de sus miembros, así como también de sus especialidades, condecoraciones y progresiones obtenidas en el transcurso de su vida activa en el escultismo.*
>
> *La organización de los miembros se hace por sectores geográficos. Así, al más alto nivel, están las regiones, estas contienen distritos y en estos se encuentran los grupos scouts. Los miembros deben pertenecer a un grupo scout, están organizados por secciones y deben inscribirse de forma anual. El sistema debe permitir el ingreso de la inscripción de los grupos y cada uno de sus miembros indicando la sección a la que pertenecen, la fecha de inscripción y el número de recibo emitido por el cobro de la cuota de inscripción.*
>
> *Los miembros activos de la Asociación de Scouts pueden participar en cursos y/o eventos organizados en el seno de la organización. Dicha participación requiere también de una inscripción, la cual será igualmente ingresada y almacenada en el sistema.*

Este enunciado plantea los requisitos generales del proyecto. De él se deducen las historias de usuario principales, las cuales se describen en la siguiente sección.

2. Planificación. En esta etapa se definen las historias de usuario y se establece el orden de prioridad. Una vez establecidas las historias de usuario se realiza el diseño de la base de datos a través del modelo entidad-relación.

a. Historias de usuario. A continuación se describen las historias de usuario que fueron definidas a partir de los requisitos generales del software:

- H1: Como usuario registrado quiero dar de alta a un nuevo miembro y crear una ficha con sus datos personales siguientes:

  - Nombres y apellidos,
  - Dirección,
  - Teléfonos,
  - Fecha de nacimiento,

- Dirección de correo electrónico,
- Número de carné,
- Foto,
- Tipo de sangre,
- Género,
- Persona a contactar en caso de emergencia,
- Profesión,
- Cargo y
- Fecha de ingreso a la Asociación de Scouts.

- H2: Como usuario registrado quiero realizar la inscripción anual de un grupo de scouts, previamente registrados, organizados por secciones, indicando fecha y número de recibo de inscripción.

- H3: Como usuario registrado quiero asignar una nueva progresión obtenida por un miembro de la Asociación y registrar cuándo la obtuvo.

- H4: Como usuario registrado quiero asignar una nueva condecoración obtenida por un miembro de la Asociación y registrar cuándo la obtuvo.

- H5: Como usuario registrado quiero asignar una nueva especialidad obtenidas por un miembro de la Asociación y registrar cuándo la obtuvo.

- H6: Como usuario registrado quiero realizar la inscripción de un miembro activo a un curso proporcionado por la Asociación de Scouts.
- H7: Como usuario registrado quiero realizar la inscripción de un miembro activo a un evento realizado dentro de la Asociación de Scouts.

De las historias de usuario anteriormente expuestas se deducen otras que aunque no provean mayor funcionalidad para el usuario, son necesarias para la realización de las primeras. Tales como:

- H8: Creación, modificación y supresión de usuarios.
- H9: Creación, modificación y supresión de permisos.
- H10: Creación, modificación y supresión de regiones.
- H11: Creación, modificación y supresión de distritos.
- H12: Creación, modificación y supresión de grupos.
- H13: Creación, modificación y supresión de condecoraciones.

- H14: Creación, modificación y supresión de especialidades.
- H15: Creación, modificación y supresión de progresiones.
- H16: Creación, modificación y supresión de secciones.
- H17: Creación, modificación y supresión de cargos.
- H18: Creación, modificación y supresión de tipos de cursos.
- H19: Creación, modificación y supresión de tipos de eventos.

Las historias de usuario anteriores (H8 a H19) tienen en común que deben ser realizadas por un usuario con permisos de administrador.

b. Modelo entidad-relación. Antes de iniciar la programación de las historias de usuario, es necesario identificar las entidades, los atributos y las relaciones. Un primer esbozo del esquema entidad-relación fue elaborado a partir de los objetos identificados en las historias de usuario y enviado a los dirigentes de la Asociación de Scouts para sus comentarios. Aplicando los conceptos teóricos sobre el diseño de bases de datos y tomando en cuenta algunas restricciones impuestas por symfony, se definió el esquema entidad-relación ilustrado en la Figura 11. Seguidamente, el modelo fue convertido a un archivo de sintaxis YAML para ser utilizado por Doctrine 1.2, la herramienta de mapeo objeto-relacional incorporada en symfony 1.4, la cual genera las clases correspondientes para permitir el acceso y manipulación del modelo de datos en el código fuente.

Entre las tablas mostradas en el esquema de la Figura 12, existen dos con contenido estático, por lo que se dispuso no incluirlas en la gestión de datos de referencia. Sin embargo, la decisión de crear las tablas 'departamento' y 'tipo de sangre' fue estrictamente de diseño, con el fin de homogeneizar y agilizar su ingreso al momento de registrar los datos de los scouts através de menús desplegables html. La carga de los datos de estas dos tablas se realiza por medio de los archivos con sintaxis YAML ubicados en el directorio `data/fixtures/` del proyecto.

El esquema entidad-relación utilizado por sfDoctrineGuardPlugin, el módulo accesorio que provee las funciones de seguridad, se ilustra en la Figura 11, y se incluye solamente para proporcionar una visión integral de la base de datos del portal para el registro de scouts.

Figura 11: Esquema entidad-relación del módulo sfDoctrineGuardPlugin

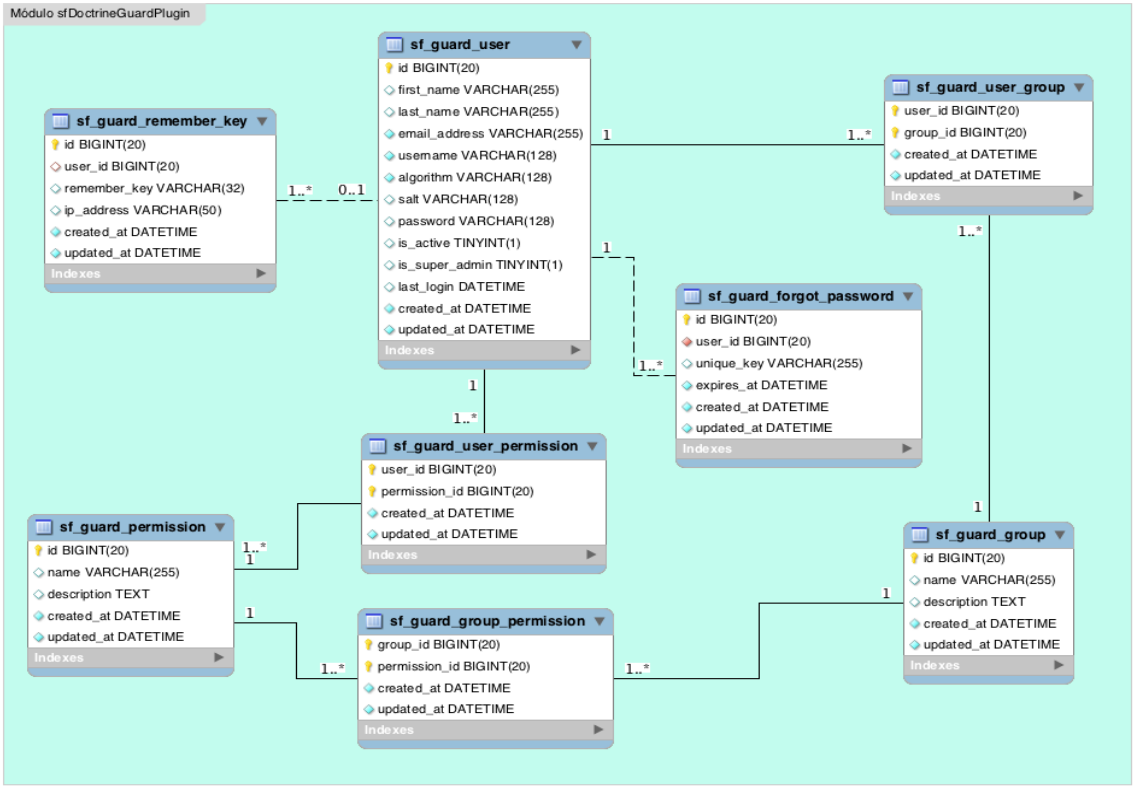
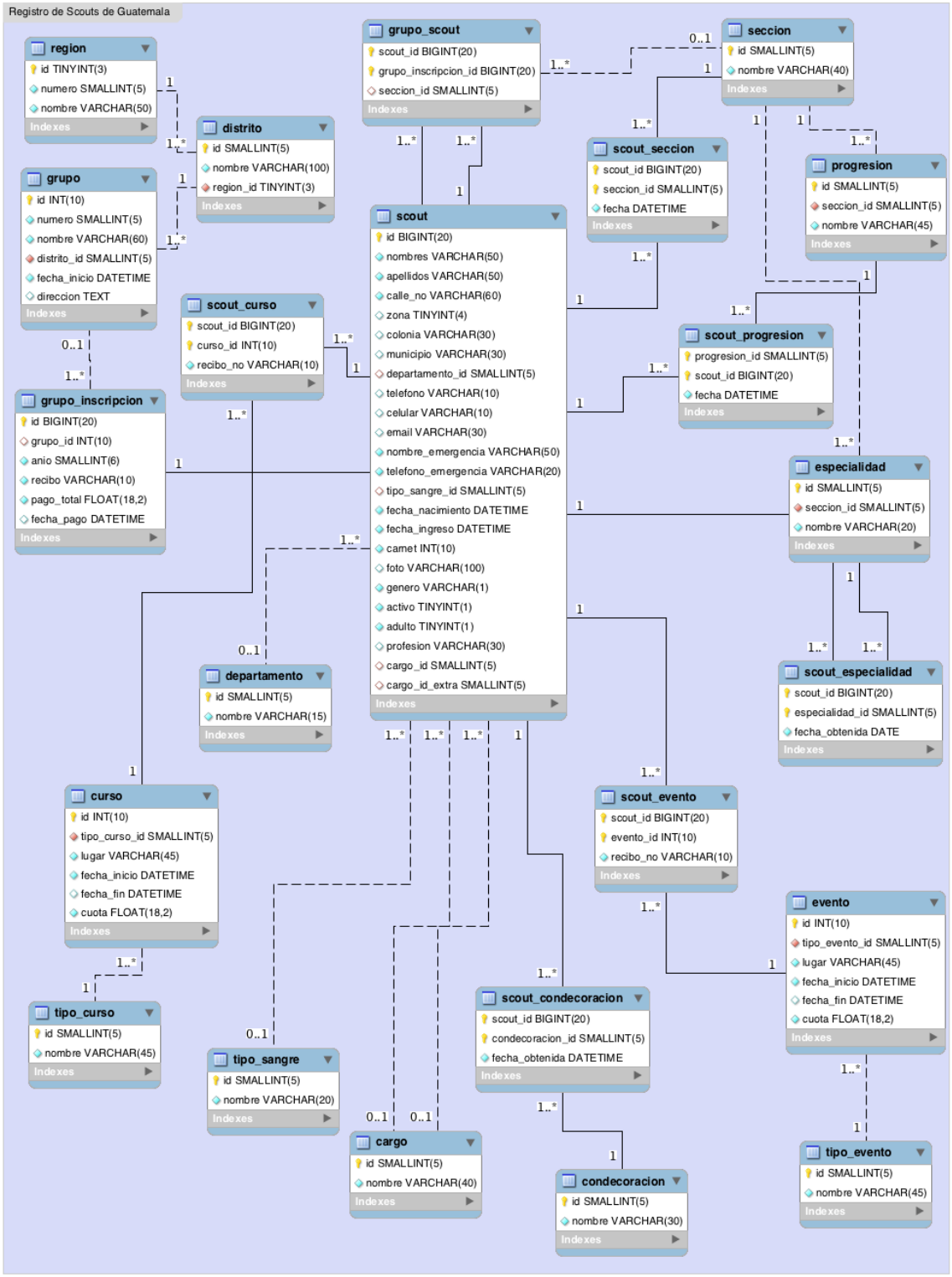


Figura 12: Esquema entidad-relación del portal de registro de scouts



3. Iteraciones. En la Tabla 3 se puede observar un resumen del plan de iteraciones de esta etapa. Cada iteración tiene programada una duración de aproximadamente 3 semanas. Una primera iteración es ejecutada al completar la programación de las historias de usuario H8 a H19, de manera que se pueda asegurar que la información respectiva es completa y exacta. Se preparan datos iniciales para cargar en la base de datos, a través de archivos con sintaxis *YAML*[16] ubicados en el directorio `data/fixtures/` del proyecto.

Con la aplicación desplegada en el servidor de pruebas, se informa a los representantes de la Asociación de Scouts de su disponibilidad. Las observaciones y retroalimentación acerca de esta iteración son recibidas por correo electrónico. Modificaciones a las historias de usuario de la primera iteración debido a fallos o ajustes en los requisitos se realizan antes de darla por finalizada y pasar a la ejecución de la iteración siguiente.

Tabla 3: Plan de iteraciones

Iteración	**Historia de usuario**	**Módulo resultante**
Iteración 1	H8: Gestión de usuarios H9: Gestión de permisos H10: Gestión de regiones H11: Gestión de distritos H12: Gestión de grupos H13: Gestión de condecoraciones H14: Gestión de especialidades H15: Gestión de progresiones H16: Gestión de secciones H17: Gestión de cargos H18: Gestión de tipos de cursos H19: Gestión de tipos de eventos	usuario permiso región distrito grupo condecoración especialidad progresión sección cargo tipo-curso tipo-evento
Iteración 2	H1: Registro de scouts ~~H2: Asignación de grupo~~ H3: Asignación de progresión H4: Asignación de condecoración H5: Asignación de especialidad	scout ~~scout~~ scout scout scout
Iteración 3	H2: Inscripción de grupo H6: Inscripción a cursos H7: Inscripción a eventos	grupo-inscripción curso evento

La segunda iteración consiste en la programación de las historias de usuario H1 a H5, las cuales afectan únicamente al módulo ‘scout’. Siguiendo el procedimiento de la primera iteración, la nueva funcionalidad resultante de esta iteración es desplegada al servidor de pruebas para obtener la aceptación de los funcionarios de la Asociación de Scouts de Guatemala. Con el fin de recibir las observaciones y

[16] http://www.yaml.org/

comentarios sobre el desarrollo de esta iteración, se organiza en esta ocasión una videoconferencia por medio de la aplicación *skype*[17].

La historia de usuario H2 no logra pasar las pruebas de aceptación de parte de los directivos de la Asociación de Scouts y por lo tanto debe ser analizada y desarrollada de nuevo. La razón por lo cual no fue aceptada se debió a que la inscripción de grupos se programó asignando a cada miembro scout un grupo previamente creado. Luego de discutir con más detalle acerca de este requisito, se concluyó que la forma de realizar la inscripción de grupos debía ser agregando los miembros scouts previamente registrados al grupo en cuestión, o sea de forma inversa a la inicialmente programada. Es así como la historia de usuario H2 pasará a la tercera y última iteración.

La tercera y última iteración es realizada con la programación de la historia de usuario H2, pendiente de la iteración anterior, y las historias H6 y H7. Con el desarrollo de las historias de usuario anteriores se completa el proceso de inscripciones de los miembros scouts. Se finaliza la iteración una vez incorporados los comentarios recibidos por correo electrónico de los representantes de la Asociación de Scouts.

4. Producción. La última fase del proyecto consiste en transferir e instalar el sistema en las instalaciones de la Asociación de Scouts de Guatemala. Para esto es necesario cumplir con los requerimientos técnicos siguientes:

Servidor con sistema operativo Linux o MS Windows,
Instalación de Apache 2.2.17 o más reciente,
Instalación de PHP 5.2.4 o más reciente,
Instalación de la base de datos MySQL 5.1 o más reciente,
Instalación de *symfony* versión 1.4.x

Es necesario definir el nombre del dominio con el que va a funcionar el servidor Web Apache y hacer los ajustes a los archivos de configuración del portal de registro de scouts para su correcto funcionamiento.

Así mismo, es necesario establecer sesiones de formación en el uso, operación y administración del sistema al personal necesario.

[17] http://www.skype.com

VI. CONCLUSIONES

A. El portal de registro de miembros de la Asociación de Scouts de Guatemala viene a llenar un inmenso vacío tecnológico en el seno de la organización. Con el registro automatizado de los miembros se racionaliza y agiliza el procedimiento de inscripciones; además permite contar con un registro histórico de la vida de sus miembros en el escultismo. De cierta manera, la base de datos alimentada a través del sistema informático será una herramienta transcendental para el establecimiento de la memoria institucional de la Asociación de Scouts.

B. La existencia de una base de datos en la Asociación de Scouts facilitará enormemente la producción y preparación de informes, reduciendo el trabajo manual y el tiempo necesario para ello. Por consiguiente, las tareas relacionadas con la organización de eventos o cursos, resultarán favorecidas.

C. Los requisitos se establecieron de manera simplificada y con objetivos a corto plazo con el fin de proporcionar un sistema de fácil uso con posibilidades de evolucionar y aumentar sus capacidades en el futuro. Debido a que este es el primer sistema informático para el registro de miembros con el que cuenta la Asociación de Scouts, era importante tomar un enfoque simple y facilitar, de esta manera, su adopción e integración dentro de los procedimientos efectuados en esta.

D. Desde una perspectiva más personal, la realización de un proyecto con enfoque social complementa satisfacciones personales, se alcanzan objetivos que no fueron inicialmente definidos, se concluye que la experiencia fue enriquecedora.

E. El desarrollo de un sistema informático a distancia presenta ciertos obstáculos difíciles de evitar: no siempre es posible organizar en corto plazo una videoconferencia, alguna pregunta queda sin responder en un correo electrónico y otros casos más. Un trabajo de colaboración a distancia presupone un compromiso firme de parte de todas las personas involucradas.

VII. RECOMENDACIONES

Tal como se mencionó en el capítulo seis, el desarrollo del proyecto se realizó con el fin de establecer un sistema simple pero funcional con miras a una expansión futura. Una vez se logre alcanzar un cierto grado de familiaridad y confianza en el sistema, es necesario pensar en su evolución y a continuación se describen algunas ideas que surgieron durante su desarrollo, o bien forman parte del proceso normal de mejoramiento. Se recomienda:

A. Efectuar una evaluación del desempeño de la base de datos a medida que aumente el volumen de la información almacenada. Se mencionó con anterioridad que el diseño físico y mantenimiento de una base de datos forman parte de su ciclo de vida, que ambos dependen del sistema de gestión de base de datos utilizado y que deben realizarse durante la implementación del sistema. La identificación de procesos que requieran una mayor eficiencia permitirá determinar con mayor precisión la creación de índices necesarios en las tablas de mayor volumen o más frecuentemente solicitadas por el sistema.

B. Reestructurar el módulo de seguridad. Una de las ventajas de una aplicación web es su acceso distribuido, de esta manera es posible que cada jefe de grupo scout pueda en el futuro realizar por sí mismo la inscripción de los miembros de su grupo. Con el fin de restringir el acceso únicamente a la información de sus miembros, es necesario agregar permisos al nivel de grupo scout, de manera que los jefes de grupo tengan acceso a crear, modificar y listar exclusivamente la información de los miembros que forman parte del grupo que ellos dirigen.

C. Considerar el desarrollo de un módulo que proporcione un interfaz para el software utilizado en la emisión de carnés de identificación de los miembros scouts. Con esto se evitaría la necesidad de ingresar nuevamente información ya almacenada y agilizar aún más el proceso de inscripciones.

D. Considerar el desarrollo de un módulo inteligente de consultas *ad hoc* de manera que los directivos de la Asociación de Scouts de Guatemala tengan acceso a la información almacenada de manera oportuna y exacta sin recurrir al diseño y programación específicos de un informe para obtenerla.

E. Aplicar las mejores prácticas en la operación del portal de registro y su ambiente. Los ataques e intentos de acceso ilegítimo a aplicaciones Web son eventos que ocurren diariamente, por lo que es necesario tomar las medidas necesarias para asegurar el servidor y la base de datos. La actualización del sistema operativo, del software en general es primordial para disminuir los riesgos.

F. Elaborar y establecer un plan simple de recuperación en caso de incidentes lo cual permita la continuidad de las operaciones y sobretodo la seguridad de la información almacenada.

VIII. BIBLIOGRAFÍA

Basili, Victor R. y C. Larman. 2003. «Iterative and Incremental Development: A Brief History». *IEEE Computer*. 36(6), p.47-56. en http://csdl.computer.org/dl/mags/co/2003/06/r6047.pdf, [consultada el 15 de febrero, 2012]

Chen, Peter Pin-shan. 1976. «The Entity–Relationship model – Toward a unified view of data». *ACM Transactions on Database Systems*, 1(1), 9–36 en http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.123.1085, [consultada el 28 de marzo, 2012]

Codd, E.F. 1970. «A Relational Model of Data for Large Shared Data Banks» *Communications of the ACM,* 13(6): p.377–87 en http://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf, [consultada el 11 de julio, 2012]

Connolly, Thomas y C. Begg. 2005. *Database Systems: A Practical Approach to Design, Implementation, and Management.* 4ª ed. Harlow, Addison-Wesley/Pearson Education. 1374 págs.

Davidson, Michelle. 2008. «Survey: Agile interest high, but waterfall still used by many». *Agile Trends Survey* [Estados Unidos de América]. 27 de junio. en http://searchsoftwarequality.techtarget.com/news/1318992/Survey-Agile-interest-high-but-waterfall-still-used-by-many, [consultado el 26 de febrero, 2012]

DeGrace, Peter, L. Hulet Stahl. 1990. *Wicked Problems, Righteous Solutions: A Catalogue of Modern Software Engineering Paradigms*. Englewood Cliffs, Prentice Hall. 244 págs.

Fowler, Martin. 2002. *Patterns of Enterprise Application Architecture*. Boston, Addison-Wesley. 560 págs.

IEEE Standard Glossary of Software Engineering Terminology. 1990. IEEE. Nueva York. 83 págs.

Letelier P. y M.C. Penadés. 2006. «Metodologías ágiles en el desarrollo de software: eXtreme Programming (XP)». *Técnica Administrativa*. [Argentina]. 5(26).

Potencier, Fabien y F. Zaninotto. 2010. *A Gentle Introduction to symfony 1.4*. Clichy, Sensio SA. 540 págs.

Pressman, Robert. 2009. *Software Engineering: A Practitioner's Approach*. 7ª Ed. Nueva York, McGraw-Hill. 928 págs.

Procedimientos, Organización y Reglamentos de la Asociacion de Scouts de Guatemala (P.O.R.). 1990. Asociación de Scouts de Guatemala. Guatemala. 74 págs.

Sommerville, Ian. 2010. *Software Engineering*. 9ª Ed. Boston, Addison-Wesley. 792 págs.

Takeuchi, Hirotaka y I. Nonaka. 1986. «The new new product development game». *Harvard Business Review*, January-February, 1-11.

Teorey, Toby, *et al*. 2006. *Database Modeling & Design: Logical Design*, 4ª ed. San Francisco, Morgan Kaufmann Press. 275 págs.

Yuhanna, Noel. 2009. «Enterprise Database Management Systems». *The Forrester Wave*, 30 de junio, págs. 1-14 en http://download.microsoft.com/download/E/1/B/E1BF9F58-7B7C-43AE-B70B-B52A6A595E17/EnterpriseDatabaseManagementSystems.pdf, [consultada el 30 de marzo, 2012]