UNIVERSIDAD AUSTRAL DE CHILE
SEDE PUERTO MONTT
ESCUELA DE INGENIERIA EN COMPUTACION

Sistema Multiplataforma de Administración de Eventos Mediante la Utilización de Códigos de Barra Bidimensional.

Seminario
de Titulación para optar
al título de Ingeniero en Computación

PROFESOR PATROCINANTE:
Sra. Claudia Gislaine Zil Bontes

Leonardo Eugenio Kusch Gómez
PUERTO MONTT - CHILE
2014

Universidad Austral de Chile
Escuela de Ingeniería en Computación

Los Pinos s/n, Balneario Pelluco
Sede Puerto Montt
Casilla 1327 · Fono: 56 65 2260990
Fax: 56 65 2277156
Email: ecomputa@uach.cl
www.uach.cl

Puerto Montt, 25 de marzo de 2014

COMUNICACIÓN INTERNA N° 036/14

A : Sra. Ximena Oyarzún Pacheco – **DIRECTORA ACADEMICA SEDE PUERTO MONTT**
Sra. Angélica Barrientos –**DEPARTAMENTO REGISTRO ACADEMICO SEDE PTO. MONTT**

DE : Sra. Sandra Ruiz Aguilar
DIRECTORA ESCUELA INGENIERÍA EN COMPUTACIÓN

C.c : Leonardo Eugenio Kusch Gomez
Claudia Zil Bontes
Sandra Ruíz Aguilar
Viviana Alvarado Espinoza

MOTIVO:

Informar a usted, las calificaciones obtenidas por el alumno de Ingeniería en Computación **Sr. Leonardo Eugenio Kusch Gomez** Rut **13520052-2**, en su informe de Titulación **"Sistema Multiplataforma de Administración de Eventos Mediante la Utilización de Códigos de Barra Bidimensional"**

Claudia Zil Bontes	6,0
Sandra Ruíz Aguilar	6,5
Viviana Alvarado Espinoza	6,4
Promedio Seminario	**6,30**

Sin otro particular, le saluda atentamente,

UNIVERSIDAD AUSTRAL DE CHILE
ESCUELA DE INGENIERIA EN COMPUTACION
SEDE PUERTO MONTT

SANDRA RUIZ AGUILAR
DIRECTORA

SRA/mva

PUERTO MONTT, 24 de marzo del 2014

De : Sra. Claudia Zil Bontes
PROFESORA PATROCINANTE

A : Sra. Sandra Ruiz Aguilar
DIRECTORA ESCUELA INGENIERÍA EN COMPUTACIÓN

MOTIVO:

Informar a Usted la calificación obtenida por el alumno **Leonardo Eugenio Kusch Gomez** en su Seminario de Titulación **"Sistema Multiplataforma de Administración de Eventos Mediante la Utilización de Códigos de Barra Bidimensional"**:

NOTA: 6,0

JUSTIFICACION:

Documentación apropiada a tipo de sistema desarrollado. Sistema de mediana complejidad. Producto acorde a requerimientos y necesidades presentados por usuarios.

OTRAS OBSERVACIONES:

Claudia Zil Bontes
PROFESORA PATROCINANTE

PUERTO MONTT, 24 marzo 2014

De : Sra. Sandra Ruíz Aguilar
PROFESORA INFORMANTE

A : Sra. Sandra Ruiz Aguilar
DIRECTORA ESCUELA INGENIERÍA EN COMPUTACIÓN

MOTIVO:

Informar a Usted la calificación obtenida por el alumno **Leonardo Eugenio Kusch Gomez** en su Seminario de Titulación **"Sistema Multiplataforma de Administración de Eventos Mediante la Utilización de Códigos de Barra Bidimensional ":**

NOTA: 6.5

JUSTIFICACION:

Buen desarrollo del informe y del sistema

Sistema que satisface una necesidad de la zona.

OTRAS OBSERVACIONES:

Sandra Ruíz Aguilar
PROFESORA INFORMANTE

PUERTO MONTT, 24 de marzo, 2014

De : Sra. Viviana Alvarado Espinoza
PROFESORA INFORMANTE

A : Sra. Sandra Ruiz Aguilar
DIRECTORA ESCUELA INGENIERÍA EN COMPUTACIÓN

MOTIVO:

Informar a Usted la calificación obtenida por el alumno **Leonardo Eugenio Kusch Gomez** en su Seminario de Titulación "**Sistema Multiplataforma de Administración de Eventos Mediante la Utilización de Códigos de Barra Bidimensional ":**

NOTA: 6,4 (seis coma cuatro)

JUSTIFICACION:

Trabajo bien desarrollado metodológicamente. La aplicación presenta gran potencial y diversos tipos de usos.

OTRAS OBSERVACIONES:

Viviana Alvarado Espinoza
PROFESORA INFORMANTE

INDICE

Figuras

Tablas

Síntesis

El presente Seminario de Titulación tiene como objetivo diseñar un sistema que permita a la empresa SurTicket dedicada a gestionar logísticamente la inscripción y acreditación de asistentes a eventos masivos de terceros, utilizando para estos fines tecnologías web, con la intención de masificar su alcance en conjunto con el uso de códigos de barra bidimensionales para acelerar los procesos de acreditación.

Para la realización de este sistema se escogió una metodología ágil, la que es una guía sobre los procesos a seguir durante la construcción de un software, desde la toma de requerimientos hasta su implementación.

El conjunto de actividades detalladas dentro de la metodología es apoyada por el uso de herramientas que las facilitan, entre las que se pueden mencionar Sybase PowerDesigner 16.1 para el diseño de los modelos de bases de datos, Adobe DreamWeaver CS6 para el desarrollo se los sistemas web, Microsoft Visual Studio 2010 y Eclipse Juno para el desarrollo de las aplicaciones para plataforma Windows y Android que interactúan con la información alojada en las bases de datos.

En resumen la implementación de este sistema permitirá la simplificación y masificación de inscripciones a eventos, además de acelerar los procesos de acreditación, con una solución acotada a las necesidades locales.

ABSTRACT

This Graduation Seminar aims to design a system that enables the company to manage logistically SurTicket registration and accreditation of attending mass events of others, using web technologies for these purposes, with the intention of expanding its reach in conjunction with the use dimensional bar codes to expedite the accreditation process.

For the realization of this system was chosen an agile methodology, which provides guidance on the procedures to be followed during the construction of a software -based requirements gathering to implementation.

The detailed set of activities within the methodology is supported by the use of tools that facilitate , among which may be mentioned Sybase PowerDesigner 16.1 for design models database , Adobe Dreamweaver CS6 for developing web systems , Microsoft Visual Studio 2010 and Juno Eclipse to develop applications for Windows and Android platform to interact with the information stored in databases.

In summary, the implementation of this system will allow simplification and popularization of event registrations, and accelerate the process of accreditation, with a bounded solution to local needs.

1. Introducción

Desde fines del siglo pasado, la producción de eventos, tales como congresos, convenciones y reuniones ha aumentado de manera exponencial en Chile, sobre todo durante los últimos cinco años, convirtiéndose en una importante fuente de ingresos para el país, generando un alto impacto económico, generando nuevas fuentes laborales y captando inversiones. Ha sido tan importante, que se han formado organizaciones a nivel nacional y regional especializadas en la generación de eventos, con el fin atraer el mayor número de potenciales clientes, generando una nueva rama turística llamada turismo de congresos y convenciones.

La producción de eventos masivos conlleva una gran cantidad de tareas de mayor o menor complejidad, por lo que la externalización de parte de las etapas que las comprenden resulta de gran alivio para los organizadores o empresas que desean generar estas actividades. Entre las tareas más comunes se tiene: selección del equipo de trabajo, calendarización, presupuesto, temática y diseño de imagen, búsqueda de patrocinadores, cartas a ponentes, difusión, organización de logística, inscripción y pago de conferencistas, acreditación, inauguración, fiestas y eventos recreativos, entre otros.

El objetivo de este Seminario de Titulación es crear las herramientas necesarias que permitan la administración de eventos, en lo que respecta a inscripción y registro de asistentes a éstos.

Resumen de Capítulos

Capítulo 1. Introducción: En este capítulo se introduce al lector acerca del contenido de este proyecto de tesis.

Capítulo 2. Objetivos: Definición de los objetivos generales y específicos que pretende alcanzar este proyecto.

Capítulo 3. Planteamiento del Problema: Se describen los antecedentes que apoyan el desarrollo de este proyecto.

Capítulo 4. Metodología: Se detalla la metodología que se utilizará para llevar a cabo este proyecto.

Capítulo 5. Recursos: En este capítulo se describen los recursos, tanto de software como hardware utilizado durante el desarrollo.

Capítulo 6. Planificación del Sistema: En esta sección de detallan las etapas de planificación y análisis.

Capítulo 7. Diseño del Sistema: Tiene como objetivo explicar y documentar el proceso de diseño del sistema.

Capítulo 8. Desarrollo del Sistema: En este capítulo se detalla el proceso de codificación del sistema.

Capítulo 9. Pruebas del Sistema: En este capítulo se detallan las pruebas realizadas y los resultados obtenidos.

Capítulo 10. Conclusiones y/o Recomendaciones: Capítulo donde se describen las conclusiones obtenidas al dar término al proyecto, además de dar recomendaciones y posibles mejoras al sistema para una siguiente etapa.

Capítulo 11. Bibliografía: Esta sección detalla las referencias bibliográficas utilizadas para sustentar, complementar y apoyar las etapas del desarrollo del proyecto.

2. Objetivos

2.1. Objetivo general

Desarrollar un sistema web multiplataforma de administración de eventos y de apoyo a la acreditación de sus asistentes.

2.2. Objetivos específicos

- Desarrollar un componente Joomla que permita la generación, mantención de eventos y listado de inscritos, así como la inscripción de sus asistentes.
- Generar códigos QR que permitan la identificación de un asistente a un evento en cuestión.
- Desarrollar las herramientas necesarias para realizar la acreditación, que permitan interactuar con el código QR generado, tanto para plataformas Android, como estaciones de trabajo bajo plataforma Windows.

3. Planteamiento del problema

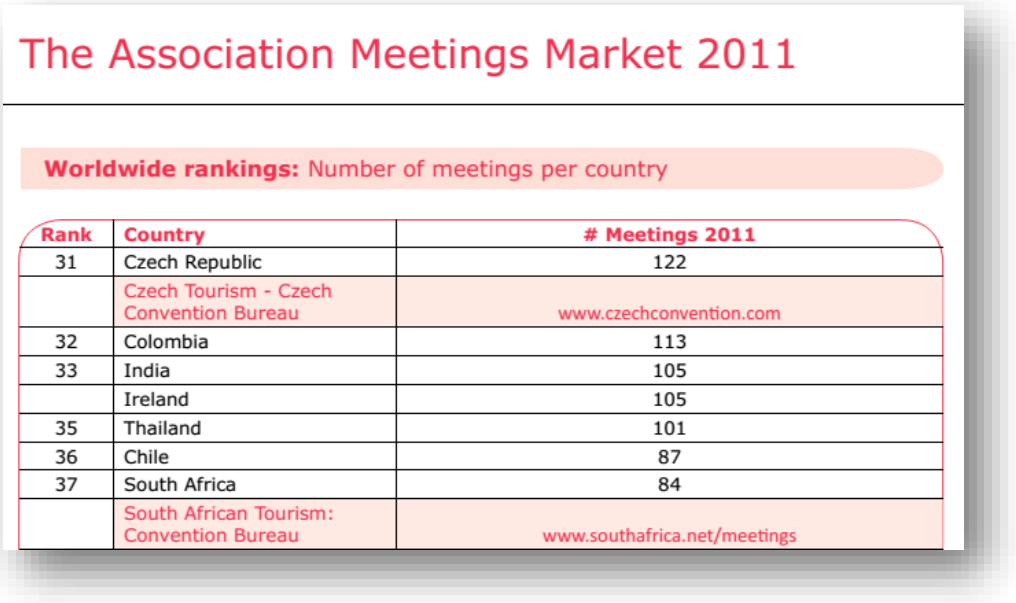
3.1. Antecedentes

3.1.1. Definición del problema

El aumento significativo en el turismo de eventos y convenciones representa un nuevo recurso de alto interés para SERNATUR[1]. Este sector ha sido capaz de generar ingresos por sesenta y seis millones de dólares al año, representando un 4,5% de los ingresos alcanzados por el sector turístico durante el año 2010. Con esto se logra ubicar a Chile en el cuarto lugar de los países organizadores de eventos y reuniones a nivel latinoamericano y 36° a nivel mundial [ICCA2012].

Las estadísticas en la Figura 1, muestran claramente el posicionamiento de Chile a nivel mundial como uno de los lugares preferidos para la realización de eventos.

[1] Servicio Nacional de Turismo



The Association Meetings Market 2011

Worldwide rankings: Number of meetings per country

Rank	Country	# Meetings 2011
31	Czech Republic	122
	Czech Tourism - Czech Convention Bureau	www.czechconvention.com
32	Colombia	113
33	India	105
	Ireland	105
35	Thailand	101
36	Chile	87
37	South Africa	84
	South African Tourism: Convention Bureau	www.southafrica.net/meetings

Figura 1: Ranking de países con mayor número de eventos a nivel mundial.

En la Figura 2 se aprecia la posición de nuestro país a nivel latinoamericano como destino con mayor número de congresos y reuniones.

The Association Meetings Market 2011

Latin- & North America rankings: Number of meetings per country

Rank	Country	# Meetings 2011
1	U.S.A.	759
2	Brazil	304
3	Canada	255
4	Argentina	186
5	Mexico	175
6	Colombia	113
7	Chile	87
8	Peru	55
9	Uruguay	46
10	Paraguay	34
11	Panama	32
12	Ecuador	30

Figura 2: Posición de Chile a nivel latinoamericano como sede de congresos y reuniones.

Como proyección del crecimiento de esta industria, se esperaba un aumento sostenido de un 15% en relación al año 2011 - 2012.

Debido a este gran incremento en el mercado es que han surgido varias empresas que pretenden entregar ventajas y soluciones a la organización de este tipo de eventos. Dentro de estas se encuentran empresas que con el uso de la tecnología y software de administración aspiran a otorgar experiencias

mucho más confortables tanto para los asistentes como a los gestores, simplificando las tareas de inscripción y acreditación.

Como la industria local no ha reaccionado ante este mercado, la empresa SurTicket formada por exalumnos de la carrera de Ingeniería Civil Industrial de la Sede Puerto Montt de la Universidad Austral de Chile, concursó por un capital semilla en el postulan la construcción de un software que permita la administración de estos eventos, mejorando su efectividad dada una mejor sincronización con el mercado local y la puesta a disposición del cliente de los elementos tecnológicos necesarios para acelerar los procesos críticos típicos de estas actividades, como lo son inscripción y check-in. De esta manera, recuperar la inversión de capitales en mercados extranjeros que se pierden por concepto de administración de eventos.

Como se ha mencionado anteriormente, gran parte de las críticas que se realizan a estos tipos de eventos se concentran en lo poco accesible del método de inscripción, y por sobre todo, en los procesos de acreditación, donde por diversas razones logran la disconformidad del cliente y el retraso en el inicio de las actividades. Entre las razones se pueden mencionar, tardía o baja implementación de los puntos de check-in, la poca experiencia en el manejo de las contingencias ocurridas en el proceso, inscripciones de último minuto, extravío de los datos de algunos participantes, reimpresión de identificaciones, entre otras.

La solución a estas complicaciones es el elemento diferenciador que enmarca las características que hicieron acreedores del capital semilla a la empresa SurTicket.

3.1.2. Esfuerzos anteriores

Como se menciona anteriormente, existen en el mercado variadas empresas que abordan este tema crítico dentro de un evento, con enfoques diferentes pero destinados al mismo fin, la inscripción y acreditación de asistentes, pues la primera impresión marca la percepción del cliente para el resto de la jornada.

Dentro de estos sistemas se destacan los siguientes:

- **Ticketforevent (figura 3):** Desarrollado por el grupo Expo Promo (grupo dedicado a la promoción de ferias y eventos), sale al público durante el año 2007, permitiendo masificar la inscripción, pago y publicidad de numerosos eventos en el Reino Unido [Ticketforevent2013]. Dicho sistema se basa en la publicación de eventos, permitiendo la creación de formularios de inscripción y pago.

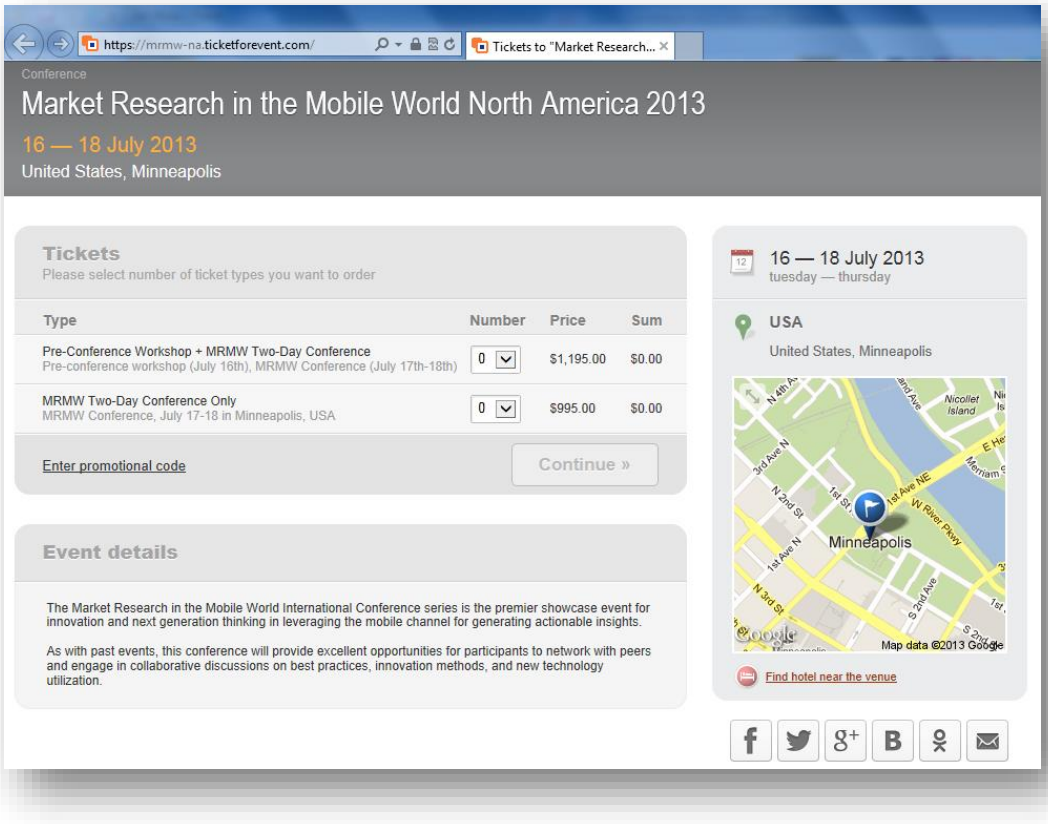


Figura 3: Captura evento publicado por Ticketforevent.

- **Eventbrite (figura 4):** Empresa norteamericana nacida el 2006 [Eventbrite2013], que tiene una plataforma que permite la creación de eventos, ofreciendo la posibilidad de cobro por inscripción. Además, ofrece estadísticas de venta de tickets y la posibilidad de realizar check-in mediante la impresión de la lista de inscritos o la utilización de códigos de barra insertos en el ticket.

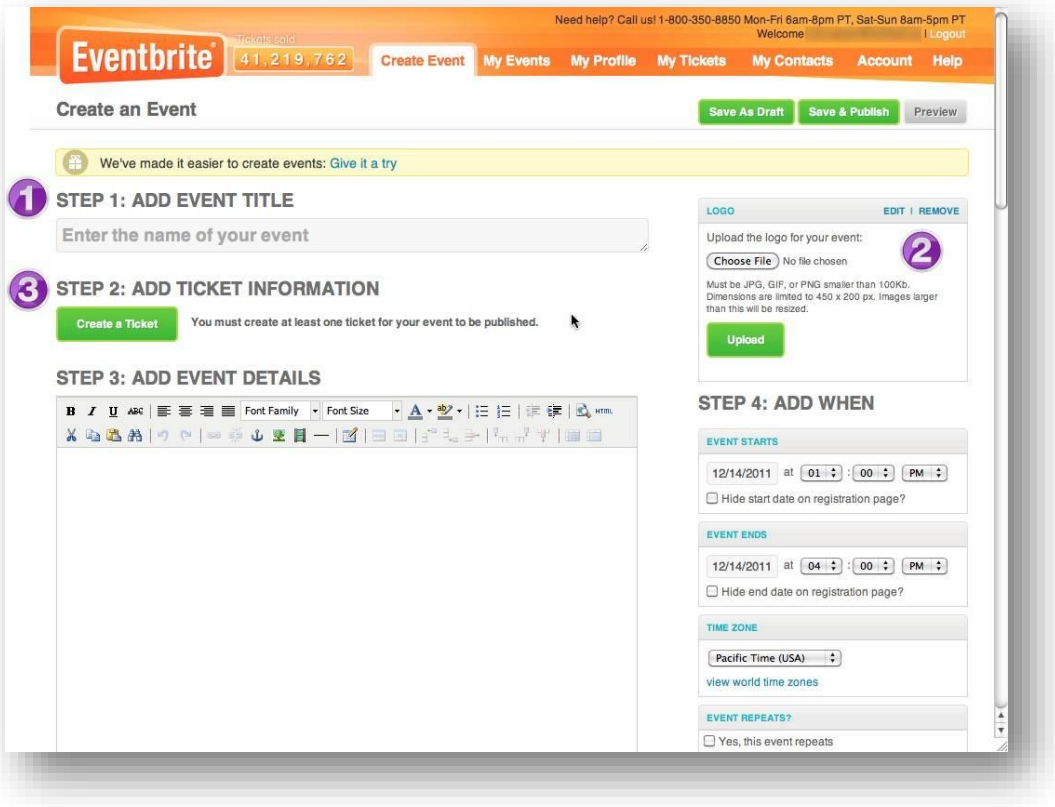


Figura 4: Formulario de creación de evento mediante Eventbrite.

- **Ticket Tailor (figura 5):** Empresa Londinense creada el año 2010 con el fin de administrar la serie de eventos que se originarían a partir de los juegos olímpicos ocurridos el año 2012 en Londres [TicketTailor2013]. Esta aplicación permite la creación de eventos, personalización de tickets, página personalizada para la inscripción, pago por evento mediante paypal y check-in mediante el uso de códigos de barra.

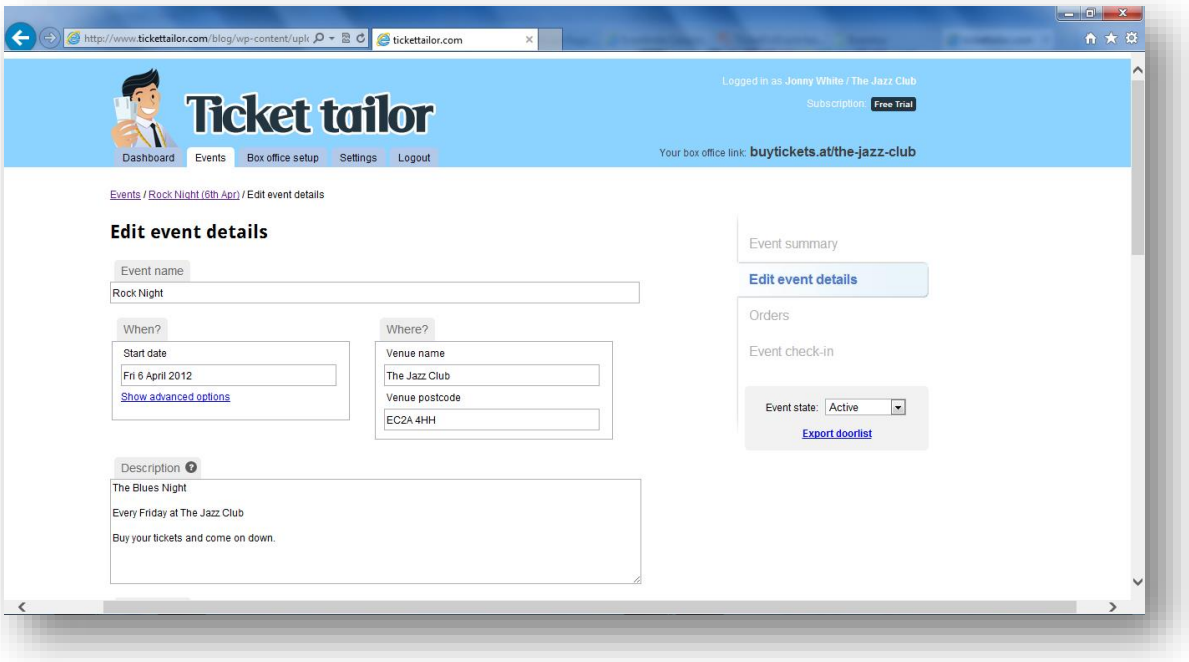


Figura 5: Formulario de creación de evento mediante Ticket Tailor.

- **Eventios (figura 6):** Empresa argentina nacida el 2008. Es una de las aplicaciones más utilizadas en Latinoamérica dada su amplia adaptación a las necesidades latinoamericanas [Eventioz2013]. Este sistema permite la creación de eventos, promoción mediante redes sociales y realizar check-in mediante la impresión del listado de participantes o utilizando móviles para lectura de códigos QR.

Figura 6: Ejemplo de ticket con código QR generado por Eventioz.

3.1.3. Solución propuesta

Si bien en el mercado a nivel mundial existen diferentes sistemas de administración de eventos, con diferentes características dependiendo del enfoque del segmento en que se especializan y generalmente en idioma extranjero, a nivel latinoamericano la participación es mínima, y a nivel de país prácticamente inexistente.

Dada esta situación, es que la empresa SurTicket recoge esta necesidad y decide la implementación de un sistema de administración de eventos enfocado en las necesidades locales de la región, con un enfoque más personal hacia sus clientes, no sólo entregando su plataforma a disposición, sino que

además el personal, equipos y la implementación de tecnología que permita la agilización de estos procesos, que en base a la experiencia de la empresa es necesario mejorar con respecto a sus predecesores.

Con el objetivo de lograr esto, se ha realizado un análisis de la situación actual, con el fin de encontrar los puntos débiles de los sistemas que existen y focalizando estos esfuerzos en las necesidades locales, también se han observado las características tecnológicas a mejorar con el fin de diferenciarse en el mercado.

Para solucionar esto se ha propuesto la construcción de un sitio que permita la fácil inscripción de quienes quieran asistir a un evento, así como el pago online de éstos, además de la implementación de estaciones de acreditación con la tecnología necesaria para acelerar dicha acreditación, realizando un rápido chequeo de inscripción y una ágil generación de las tarjetas de identificación. Se utilizarán códigos QR para acelerar el proceso de búsqueda de los inscritos e impresoras térmicas adaptadas para la generación automática de los gafetes una vez verificada la identidad del asistente.

Esto se pretende desarrollando un componente Joomla que permita en el back-end la generación y mantención de eventos e inscritos, así como en el front-end la inscripción de sus asistentes y el pago de eventos según corresponda.

Al finalizar la inscripción se generarán códigos QR que permitirán la identificación de un asistente a un evento en cuestión. Este estará incluido en el

ticket o invitación que el inscrito recibirá por e-mail y que este deberá llevar para su acreditación.

En el caso del proceso de acreditación, se leerá el código QR para marcar la asistencia y luego automáticamente se imprimirá el documento de identificación. Este proceso se implementará a través de una funcionalidad desarrollada, tanto para plataformas Android, como para estaciones de trabajo bajo plataforma Windows.

Grupo de Trabajo

El equipo de trabajo se compondrá de las siguientes personas:

Leonardo Kusch	Alumno tesista. Tendrá la tarea de Analizar, Diseñar y Construir el sistema. Pruebas de Generación.
Marcos González	Director TI SurTicket. Encargado del mejoramiento tecnológico de la empresa, el cual participa en el análisis de los requerimientos, diseño y construcción del proyecto

Tabla 1: Equipo de Trabajo.

3.2. Justificación

3.2.1. Situación sin proyecto

En base a lo comentado con anterioridad, es clara la existencia de un creciente mercado que a nivel local no se ha cubierto, debiéndose recurrir a empresas extranjeras que permitan el desarrollo de estas actividades. Esto provoca filtraciones de recursos hacia otras latitudes que es justo lo contrario de los esfuerzos de entidades locales apoyadas por organismos gubernamentales tales como SERNATUR, sin mencionar la mejora de la experiencia del usuario que se pretende alcanzar.

3.2.2. Situación con proyecto

Al finalizar este Seminario de Titulación la empresa SurTicket, podrá administrar de manera simple las tareas de inscripción y acreditación de asistentes a eventos masivos, pudiendo ofrecer sus servicios a la comunidad con un toque más personal y focalizado en las necesidades que caracterizan a la zona, lo que lo diferencia principalmente de los otros sistemas.

Una de las ventajas determinantes del proyecto se refiere a la implementación de tecnología que permitirá la aceleración del proceso de acreditación, mencionando por ejemplo la utilización de códigos QR (Quick

Response o código de respuesta rápida por sus siglas en inglés). Con esto se irán minimizando los errores de digitación, habrá una generación automática y rápida de las identificaciones de los asistentes y se tendrá la posibilidad de aumentar rápidamente, de ser necesario, los puntos de acreditación mediante la utilización de smatphones, tablets u otros dispositivos que trabajen con sistema operativo Android y que tengan una cámara que permita la lectura de los códigos QR.

Por último, pero no menos importante, es la ventaja de la implementación del sistema de inscripción mediante un CMS (Content Management System o sistema de administración de contenidos por sus siglas en inglés) lo que permite la fácil implementación de clones del sistema en caso de ser necesario.

3.3. Delimitación

Dada la limitante de tiempo para el desarrollo, implementación y puesta en marcha del proyecto sólo se plantea la creación de un componente de CMS que permita la inscripción a un evento seleccionado desde un grupo de éstos. Los datos requeridos en los formularios no serán personalizables en una primera fase.

Se plantea la implementación de pago por evento con un solo método de pago. El ticket de inscripción que recibirá el cliente tendrá un formato específico.

Se considera la posibilidad de inscripción de eventos complejos, que contengan múltiples actividades a selección de inscriptor, pero no así el pago diferenciado, dependiendo de la actividad seleccionada.

En lo que refiere a la lectura de códigos QR e impresión de identificaciones, estas serán referidas y limitadas al hardware existente durante el momento de construcción del sistema.

Además, se acotarán las pruebas a los sistemas operativos, ya sea plataforma Windows o Android a las versiones con las que fueron diseñadas.

4. Metodología

Basándose en las características del proyecto y composición del grupo del equipo de trabajo, se ha optado por la utilización de la metodología XP (Extreme Programming o Programación Extrema por sus siglas en Inglés). Esta metodología se encuentra dentro del grupo de las llamadas “metodologías ágiles”, éstas de denominan así porque dotan de cierto dinamismo a los proyectos de desarrollo de software, y proporcionan velocidad a la generación de dicho proyecto. Su principal autor es Kent Beck quien creó esta metodología mientras se encargaba del desarrollo de una aplicación de nóminas para la empresa Chrysler Corporation en el año 1996 [ExtremeProgramming2009].

El ciclo de vida de la metodología Extreme Programming consta de cinco etapas o fases: Fase de Exploración, Fase de Planeamiento, Fase de Producción, Fase de Mantenimiento y Fase de Muerte las cuales serán detalladas a continuación [Sánchez2004].

Fase de Exploración: En esta fase se plantean a grandes rasgos los requerimientos del cliente[2], que son la base fundamental para generar

[2] Entiéndase por cliente a los ejecutivos mandantes de la empresa SurTicket.

las herramientas requeridas, al mismo tiempo el desarrollador[3] se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto, tales como Joomla, componentes, patrones de diseño, códigos QR, Android, entre otros. Posteriormente se prueban las tecnologías y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo.

- **Fase de Planeamiento:** Se priorizan los requerimientos del cliente y se acuerda el alcance del release[4], luego se estima cuanto esfuerzo se necesitará para cada requerimiento, y a partir de este punto se define el cronograma.

  Para lograr un release definitivo se deben incluir varias iteraciones. En cuanto al cronograma fijado, este se itera tantas veces como sea necesario, con el fin de satisfacer los requisitos del proyecto. Es el cliente quien selecciona los requerimientos que se utilizarán para cada iteración, y es labor del jefe de proyecto crear en este punto pruebas funcionales para ser ejecutadas al final de cada iteración. Al final de la última iteración el sistema está listo para la fase de producción.

[3] Entiéndase por desarrollador al alumno tesista.
[4] Liberación para el usuario de una versión del software.

- **Fase de Producción:** Requiere prueba y comprobación extra del funcionamiento de las herramientas creadas. Esto debe hacerse antes de que se libere como parte del proyecto. En esta etapa aún se pueden encontrar modificaciones necesarias de realizar, luego se debe decidir si estos cambios se incluyen o no en el release actual, de no incluirse las modificaciones, las ideas y sugerencias se documentan para ponerlas en práctica en la próxima fase.
- **Fase de Mantención:** Es la fase posterior a la entrega del software, y se refiere a realización de todas aquellas modificaciones puntuales requeridas en el proyecto. Es probable que debido a las correcciones solicitadas, la velocidad del desarrollo pueda desacelerarse después de que el sistema esté en producción, alterando las siguientes entregas.
- **Fase de Muerte:** Ocurre cuando el cliente no tiene más solicitudes para ser incluidas en el sistema, para que esto ocurra se deben satisfacer todas las necesidades del proyecto, tales como rendimiento y confiabilidad del sistema.

  Dado que este proyecto se limita hasta una fase funcional mínima, se considerará como fase de muerte cuando se haga efectiva la entrega del primer release.

  Finalmente, se genera la documentación del sistema y no se realizan más cambios en la arquitectura del proyecto.

5. Recursos

5.1. Hardware

Equipo	Descripción	Función
Samsung 530U3B	Intel Core I5 1,6Ghz. 4 GB RAM Disco Duro 500 GB	Contendrá el software para el desarrollo de los componentes y del proyecto de tesis en general.

Tabla 2: Recursos de hardware.

5.2. Software

Dentro de las herramientas que apoyan la fase de construcción, se tiene la siguiente lista.

Software	Función
.Net Framework 3.5	Framework o plataforma de desarrollo, creado para soportar distintos lenguajes de programación como C#, Visual Basic .Net y otros.
Eclipse Juno	IDE gratuita para Windows que permite la edición de código Java para la creación de aplicaciones Android.
Java 7	Lenguaje de programación orientado a objetos.
C# 4.0	Lenguaje de programación orientado a objetos.
PHP 5.3.3	Lenguaje de programación orientado al desarrollo web
Visual Studio 2010	IDE de desarrollo para plataforma Windows de la empresa Microsoft.
Dreamweaver CS6	IDE para múltiples lenguajes de desarrollo web

MySQL 5.5.34	Motor de base de datos

Tabla 3: recursos de Software.

6. Planificación del Sistema

Etapa inicial de la metodología seleccionada en la cual se analiza el problema, así de esta manera se logra plantear una solución, la cual se verá plasmada en sistema que se desarrollará.

6.1. Exploración

Es en esta etapa donde se da inicio al desarrollo del proyecto. Es en donde de adquieren los conocimientos sobre las necesidades del cliente, con las cuales se toman decisiones tales como, las herramientas que se utilizarán para el desarrollo, además se recogen las historias de usuario, las que representan los requisitos del cliente y que posteriormente se verán reflejadas en los casos de uso.

6.1.1. Historias de usuario

Las historias de usuario son la representación de un requisito de software solicitado, es una de las herramientas básicas en la toma de requerimientos de

la metodología XP. A continuación, y a modo de ejemplo se presentan un par de historias recogidas.

Historia de usuario	
Número: **1**	Nombre: **Inscripción de usuario**
Usuario: **Usuario**	
Modificación de Historia Número:	
Prioridad de Negocio: **Alta**	
Descripción: En el sitio web el usuario podrá seleccionar un evento y luego rellenará el formulario de inscripción, al finalizar o enviar el formulario, al usuario le llegará un e-mail con el ticket de invitación con la información del evento suscrito y un código QR en el, dicho QR deberá ser presentado al momento de realizar su acreditación.	

Tabla 4: Historia de usuario sobre la inscripción de eventos.

Historia de usuario	
Número: **2**	Nombre: **Acreditación de usuario**
Usuario: **Administrador**	
Modificación de Historia Número:	
Prioridad de Negocio: **Alta**	
Descripción: El usuario durante el momento de la acreditación entrega su ticket de inscripción, el que contiene el código QR, ese código es leído por el sistema con lo cual se marca su asistencia y se genera su credencial de identificación.	

Tabla 5: Historia de uso referente al proceso de acreditación.

Una vez finalizada la recopilación de historias de usuario se ha obtenido el siguiente listado de requerimientos.

N°	Lista de Requerimientos
1	En lo que refiere al sistema web, en la sección que corresponde a la

	interfaz de administración se deberá poder ver la lista de clientes, eventos, actividades y usuarios. Además de insertar, modificar y eliminar datos en cada una de estas secciones.
2	En la sección que corresponde al usuario que se inscribirá en un evento, se debe ver el listado de eventos a inscribir, un formulario donde este complete sus datos para procedes con la inscripción.
3	Si el evento es de pago, deberá pagar su inscripción online.
4	Si el usuario se ha inscrito con anterioridad debe cargarse automáticamente los datos de él, así disminuirá el tiempo del proceso de inscripción.
5	En cuanto a la aplicación que leerá el código QR, esta deberá buscar los datos que se almacenaron durante el proceso de inscripción e imprimir los datos necesarios en las etiquetas que corresponden a los gafetes, además esta deberá marcar la asistencia al evento. Las funciones de esta aplicación se deben desarrollar para plataforma Windows y plataforma Android.

Tabla 6 : Lista de requerimientos obtenidos del cliente.

6.1.2. Diagramas de casos de usos

Los diagramas de casos de uso describen el comportamiento de los diversos roles que participan en los procesos del sistema.

Según la información obtenida durante el levantamiento de requerimientos el sistema web consta de dos actores, administrador y usuario, en los cuales el Administrador tendrá los privilegios suficientes para lograr la administración de eventos que estarán disponibles, y el usuario que sólo podrá realizar la inscripción a uno o más eventos. Para diagramar los casos de uso se escogió la herramienta Microsoft Visio 2010.

A continuación, se representa el caso de uso del sistema web y las acciones que realizan los actores antes mencionados.

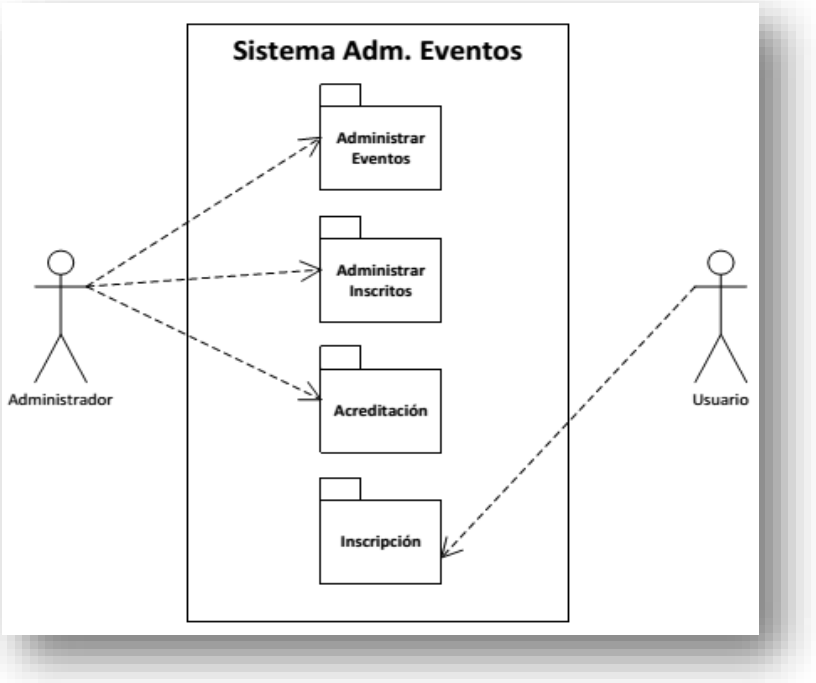


Figura 7 : Diagrama de casos de usos sistema de administración de eventos.

Dada la similitud entre algunas secciones como por ejemplo administrar eventos y administrar inscritos, sólo se explicará uno de ellos como referencia.

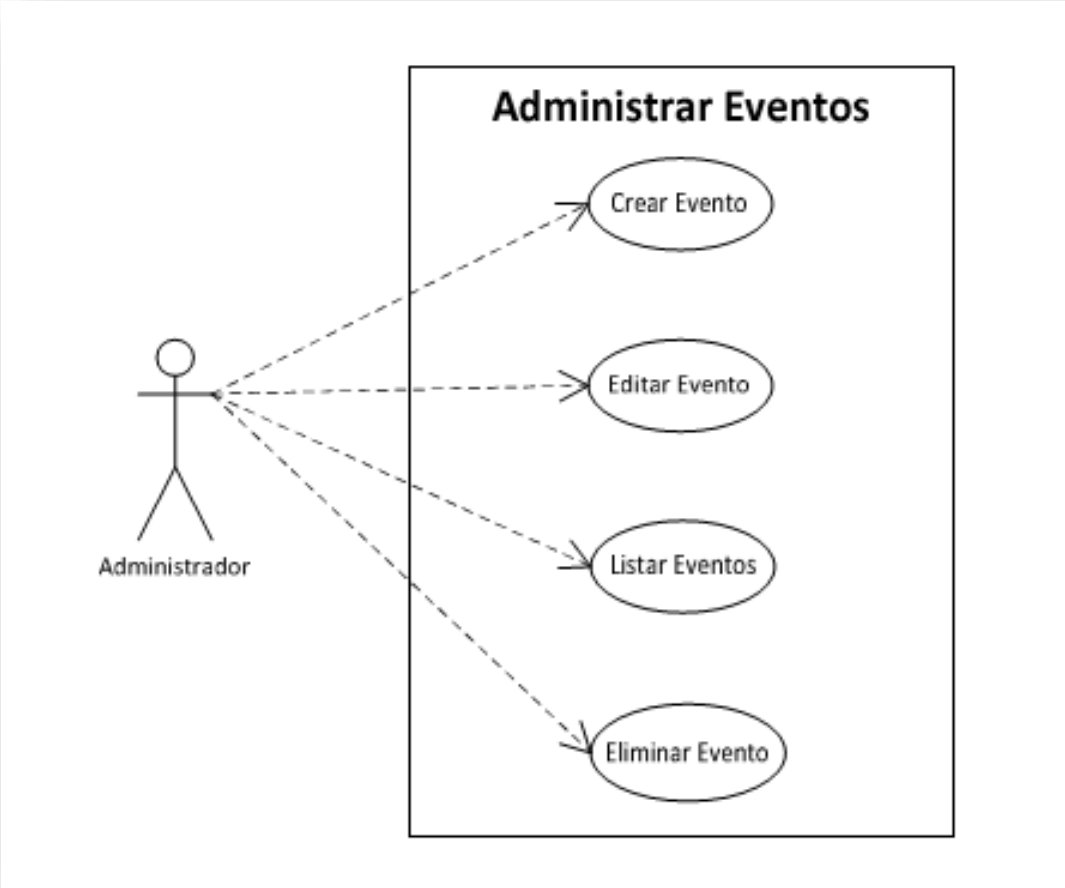


Figura 8: Caso de uso administrar eventos.

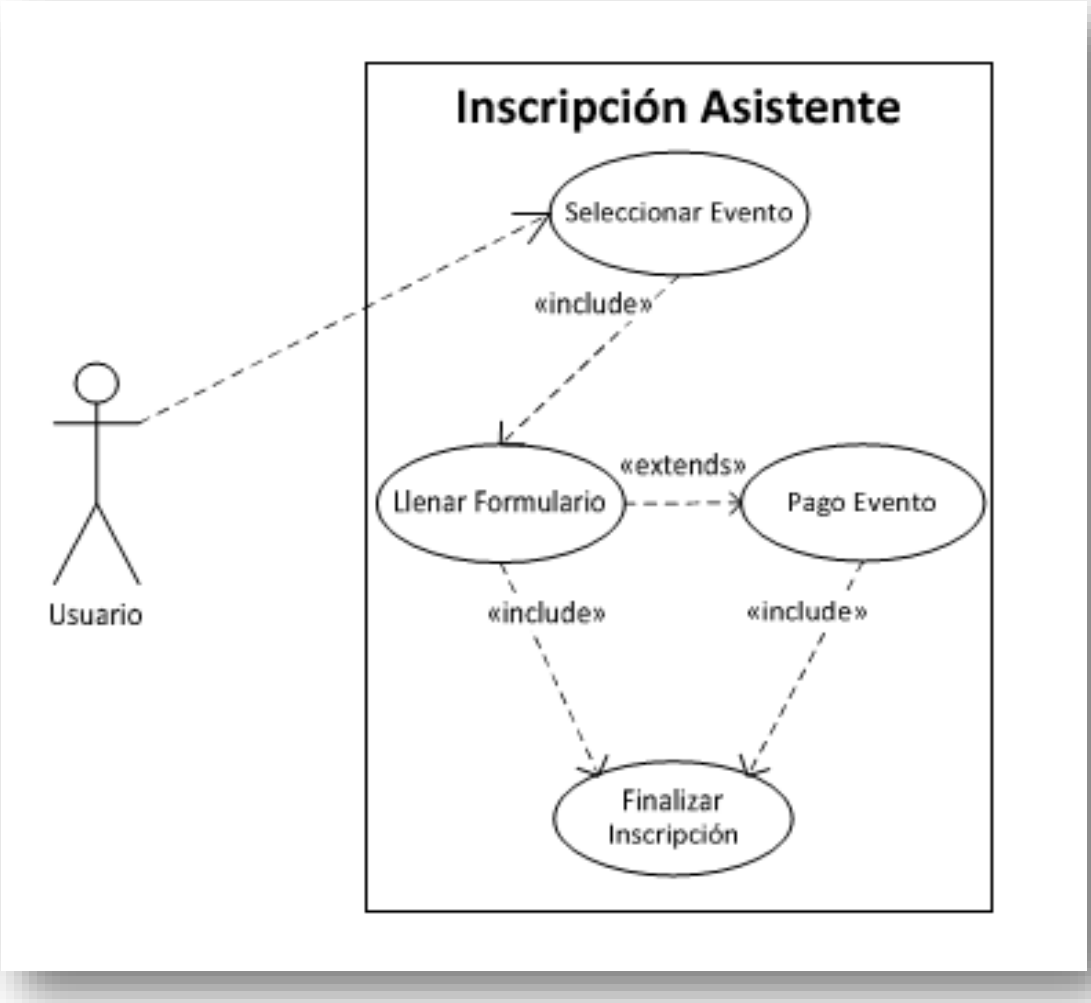


Figura 9: Casos de uso inscripción asistente.

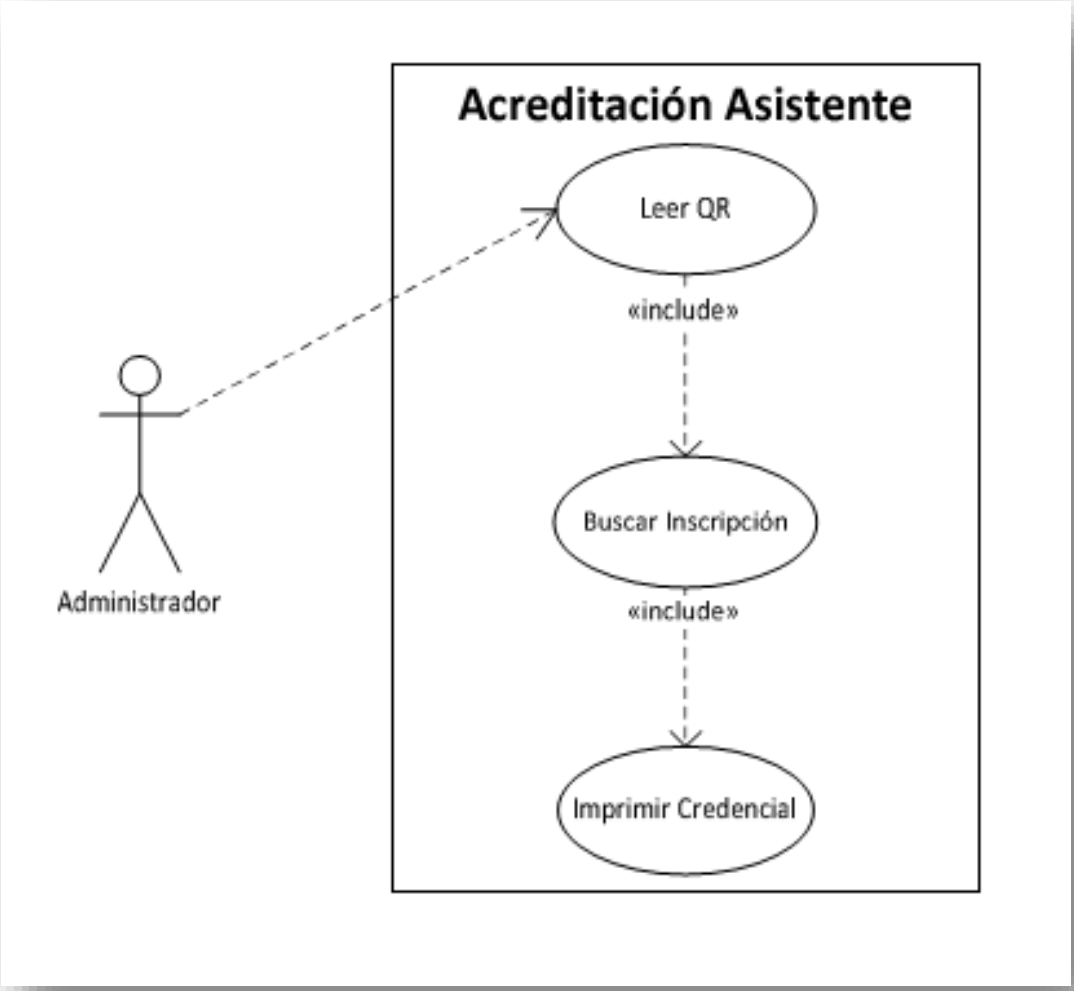


Figura 10: Caso de uso acreditación asistente.

6.1.3. Descripción de casos de usos

Nombre	Crear Evento.
Descripción	Permite la creación de un nuevo evento en el sistema.
Actor	Administrador.
Precondiciones	Administrador debe estar autenticado como tal en Joomla.
Flujo Normal	• El administrador presiona sobre el menú crear evento. • El sistema muestra el formulario correspondiente, con los datos requeridos para un evento. • El administrador introduce los datos requeridos. • El sistema comprueba validez de los datos y lo almacena, los datos requeridos son nombre del evento, fecha del evento, nombre de la actividad y ubicación. Todos los campos son obligatorios.
Flujo Alternativo	• Si el usuario ya ha ingresado sus datos con anterioridad, se auto-completa el formulario, pudiendo actualizar su información. • Si el evento es de pago, se presentan las opciones

	de pago online.
Excepciones	• Al presionar el botón crear el sistema verifica la validez de los datos introducidos y solicita corrección de estos de ser necesario, se requiere que la fecha sea superior a la fecha actual.
Post-condiciones	Un nuevo evento ha sido creado.

Tabla 7: Caso de uso crear evento.

Nombre	Eliminar Evento.
Descripción	Permite la eliminación de un evento, cuidando la integridad de los datos y que no existan usuarios inscritos.
Actor	Administrador.
Precondiciones	Administrador debe estar autenticado como tal en Joomla.
Flujo Normal	• El administrador lista los eventos. • Se selecciona la acción de eliminar del evento seleccionado.

	• Se verifica la no existencia de personas inscritas. • Se elimina el evento.
Excepciones	• Se informa de la existencia de personas inscritas y se cancela la eliminación.
Post-condiciones	Evento eliminado.

Tabla 8: Caso de uso eliminar evento.

Nombre	Inscripción asistente.
Descripción	Permite la inscripción de un asistente a un evento.
Actor	Usuario.
Precondiciones	Ingresar al sistema.
Flujo Normal	• Selecciona el evento a asistir. • Rellena el formulario de inscripción con los datos solicitados como: rut, nombre, e-mail, ocupación, empresa, cargo, teléfono. Nombre, rut, e-mail son datos obligatorios.

	• Se verifican los datos ingresados: rut, e-mail y teléfono. • Finaliza la inscripción. • El usuario recibe en su email un ticket de ingreso.
Flujo Alternativo	• Si el evento es de pago se presentarán las opciones de pago online.
Excepciones	• Si los datos son mal ingresados se le informa al usuario que campos deben ser modificados para cumplir con los requisitos.
Post-condiciones	La inscripción ha sido realizada.

Tabla 9: Caso de uso inscripción asistente.

Nombre	Acreditación Asistente.
Descripción	Permite la verificación de la inscripción de un asistente así como la asistencia al evento y la generación de sus

	credenciales.
Actor	Administrador.
Precondiciones	Configuración correcta de los sistemas de acreditación.
Flujo Normal	• El administrador recibe del asistente el ticket de inscripción y lo pasa por el lector de códigos QR. • Se busca la identificación del asistente asignada en el código QR. • Se marca su asistencia. • Se imprimen las credenciales.
Flujo Alternativo	• Se ingresa manualmente el código de inscripción en caso de no ser legible el código QR.
Excepciones	• No se encuentra inscripción del usuario.
Post-condiciones	El asistente quedará acreditado y su asistencia al evento será marcada.

Tabla 10: Caso de uso acreditación asistente.

6.2. Plan de desarrollo y entrega

Una vez aclaradas las necesidades de la empresa en base a la toma de requerimientos, y son ratificadas por la empresa, se analizan las tecnologías y herramientas a utilizar. Además, se procede a la planificación temporal de las siguientes fases del proyecto, entre las cuales se encuentran:

- Diseño de modelos y diagramas
- Desarrollo de prototipos
- Codificación de fases beta del sistema
- Versión release del sistema
- Correcciones e iteraciones de las etapas anteriores
- Entrega final del proyecto
- Reuniones

6.3. Definición del sistema

Finalizada esta etapa se pueden, concluir a modo de resumen, las características del software que se desarrollará.

Corresponde a un sistema web que permitirá la administración de los asistentes a eventos masivos. Se caracterizará por ser un componente Joomla, el cual permitirá a los usuarios denominados “asistentes” la posibilidad de registrar su intención de asistir a un congreso, exposición, función, etc. y a su

vez registrar la asistencia de los participantes por parte de los “administradores”. Si un evento a inscribir requiere el pago por asistencia, el sistema durante el proceso de inscripción permitirá el pago online, utilizando sistemas de transacciones bancarias de terceros, y así cancelar el costo requerido.

7. Diseño del Sistema

En este segmento se generan variados modelos de los procesos requeridos por el sistema, dentro de los módulos que lo componen. Se pretende lograr una amplia visión de las funcionalidades y el manejo de datos que se necesita, y así apoyar el proceso de generación de las características que se deben desarrollar para satisfacer de manera óptima los requerimientos solicitados y obtenidos de parte del cliente.

Dentro de estos modelos se pueden mencionar los diagramas de actividades, los que detallan los procesos de gestión de la información. Igualmente se explicará el modelo de datos que se obtuvo del resultado del análisis de los requerimientos obtenidos durante el proceso de toma de requerimientos.

7.1. Diagramas de Actividad

Una vez que se han obtenido los diagramas de casos de uso y ratificados los requerimientos solicitados, se requiere el diseño de diagramas de actividad que representarán detalladamente los procesos del sistema.

7.1.1. Diagrama Actividad Crear Evento

Este diagrama representa el proceso a seguir para la creación de un evento por parte del administrador de sistema.

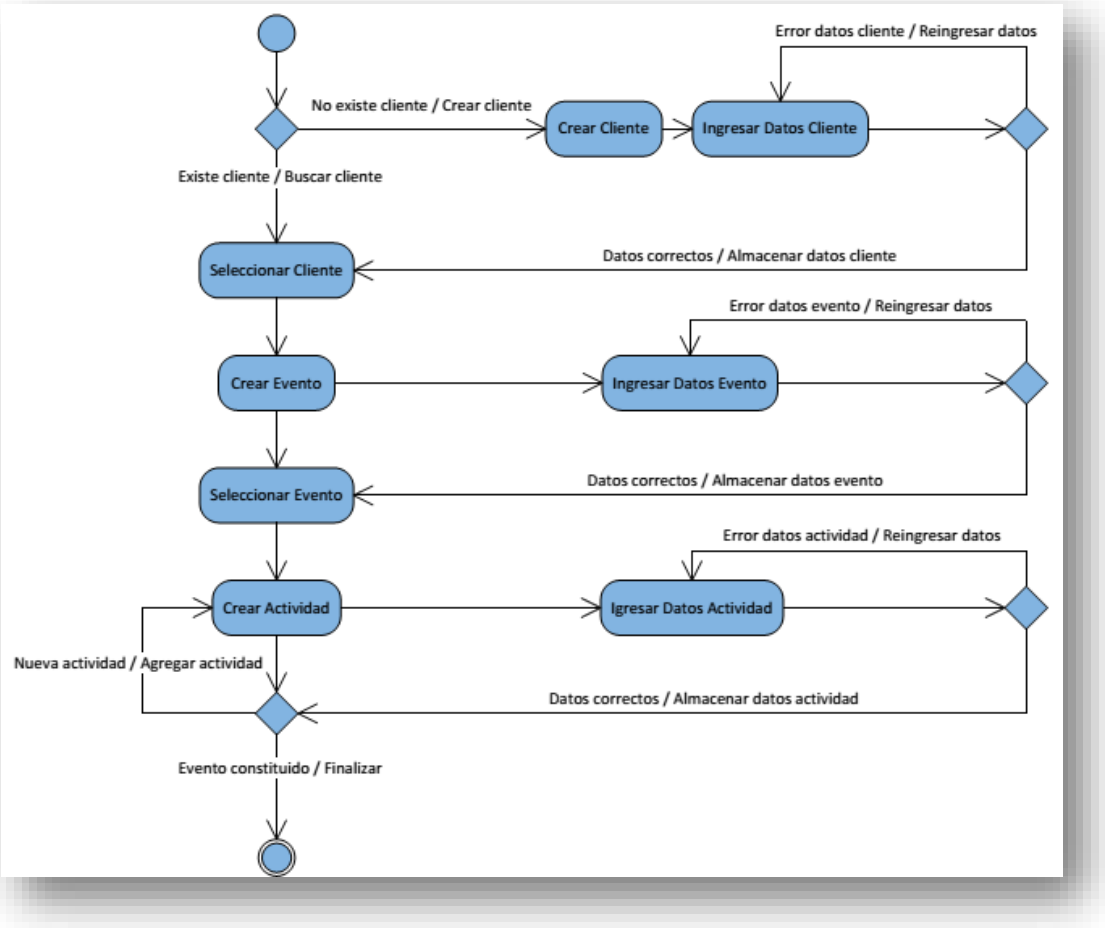


Figura 11: Diagrama de actividad que representa la secuencia de procesos para realizar la creación de un evento.

7.1.2. Diagrama de Actividad Proceso de Acreditación

Este diagrama explica el proceso que debe realizar el ente acreditador al momento de registrar la asistencia.

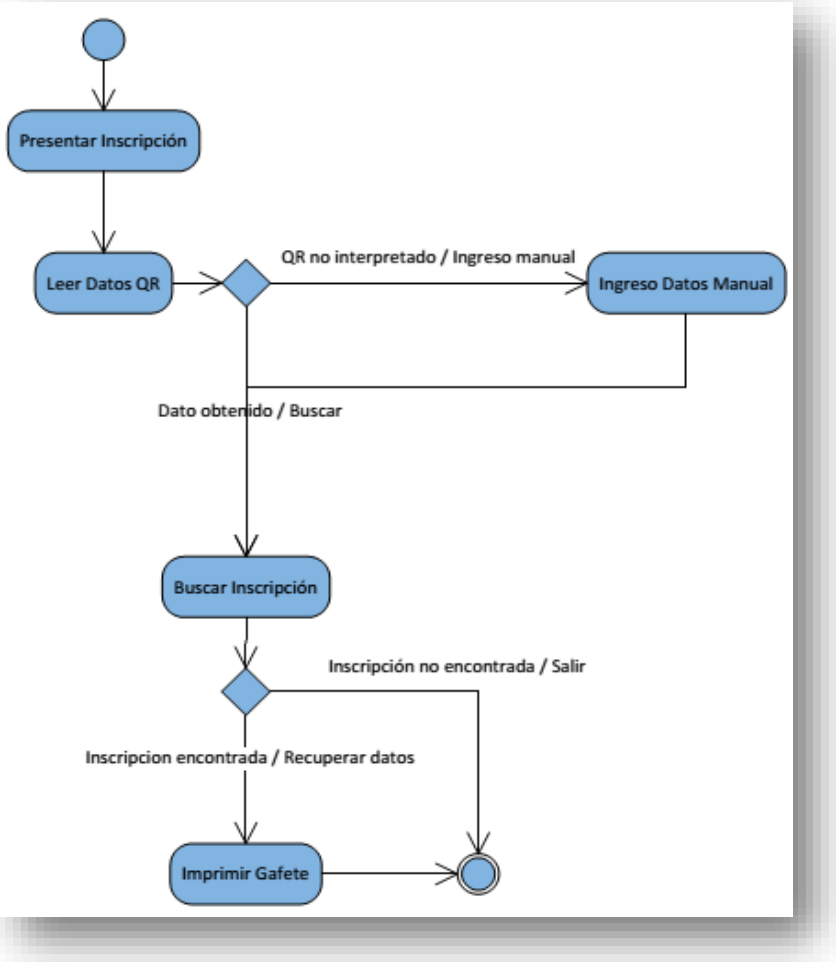


Figura 12 : Proceso de acreditación en un evento.

7.1.3. Diagrama de Actividad Inscripción Evento

Este diagrama muestra el proceso necesario para que un usuario o asistente realice una inscripción.

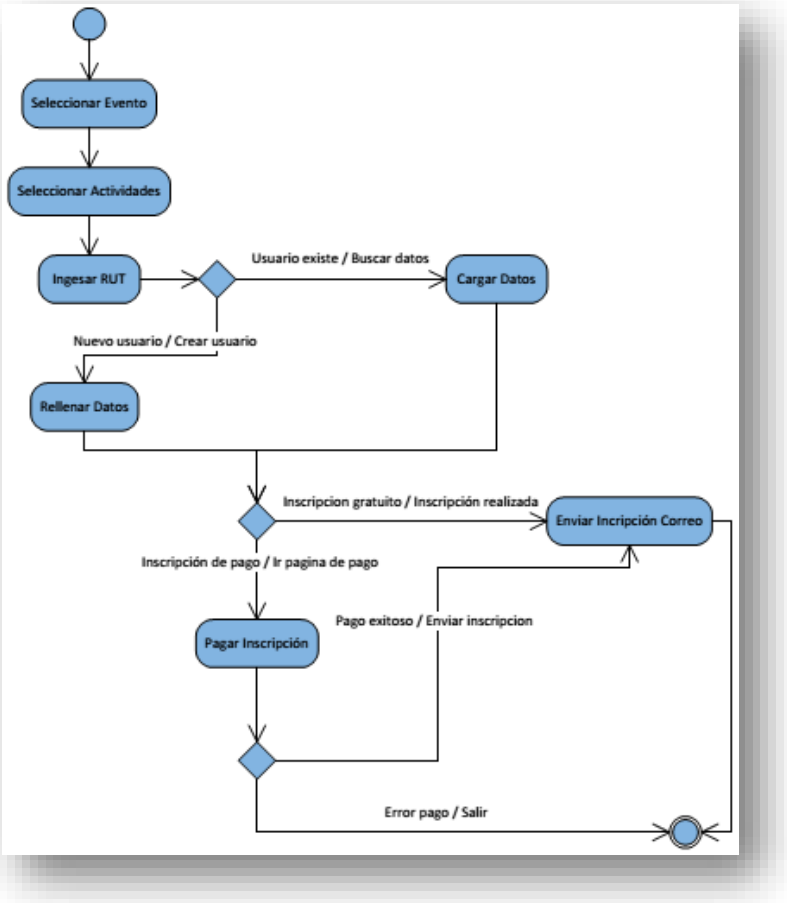


Figura 13 : Proceso de inscripción a un evento.

7.2. Patrones de Diseño

Una característica introducida como buenas prácticas en el desarrollo de componentes Joomla a partir de la versión 1.5 es la utilización del patrón de diseño **Modelo Vista Controlador** o **MVC**, el cual propone separar el desarrollo de un software en tres rasgos: acceso a datos, lógica de negocio e interfaz de usuario.

Este patrón de diseño pretende dar énfasis en la reutilización de código y la segmentación de conceptos, rasgos que buscan facilitar la labor del desarrollo de aplicaciones y su futura mantención.

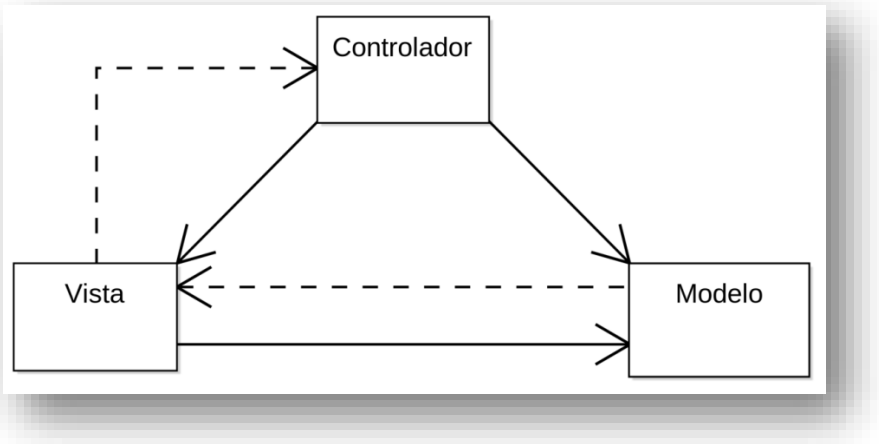


Figura 14 : Diagrama de relaciones entre los distinto componentes MVC.

7.2.1. Descripción de los Componentes de MVC

A continuación, se describirán las características y funciones de los componentes que comprenden el patrón de diseño **MVC**.

Modelo: Es el componente que representa la información que el sistema opera. Gestiona las consultas y la mantención o actualizaciones de los datos. Implementando los privilegios de acceso que se hayan descrito en la lógica de negocios. Entrega a la vista la información solicitada para ser mostrada. Toda solicitud de acceso o manipulación de datos, llega al modelo a través del controlador.

Vista: Exhibe el modelo, lógica de negocio e información, en un formato apropiado para la interacción del usuario.

Controlador: Responde a acciones solicitadas por el usuario, enviando peticiones al modelo cuando se realizan consultas de datos. Así también puede ejecutar acciones sobre la vista, si es que se le solicita un cambio en el modo de presentar los datos. De manera que es posible manifestar que el controlador tiene la función de intermediario entre el modelo y la vista.

En las siguientes secciones se verá la aplicación de este patrón de diseño en la construcción de la aplicación web.

7.3. Diagramas de Clases

Debido a las características del sistema a desarrollar se expondrán tres grupos de diagramas de clases, los que representan a cada uno de los módulos que conforman la totalidad del software.

7.3.1. Componente Joomla

Una de las grandes ventajas de utilizar un CMS como Joomla, así como se explicó anteriormente, es la posibilidad de reutilizar clases previamente desarrolladas, permitiendo un ahorro considerable del tiempo de desarrollo tanto de componentes, módulos o plugins.

Como se mencionó se utilizó el patrón de diseño **MVC** por lo que se presentarán las clases requeridas para el desarrollo del componente en tres secciones: Modelo, Vista y Controlador.

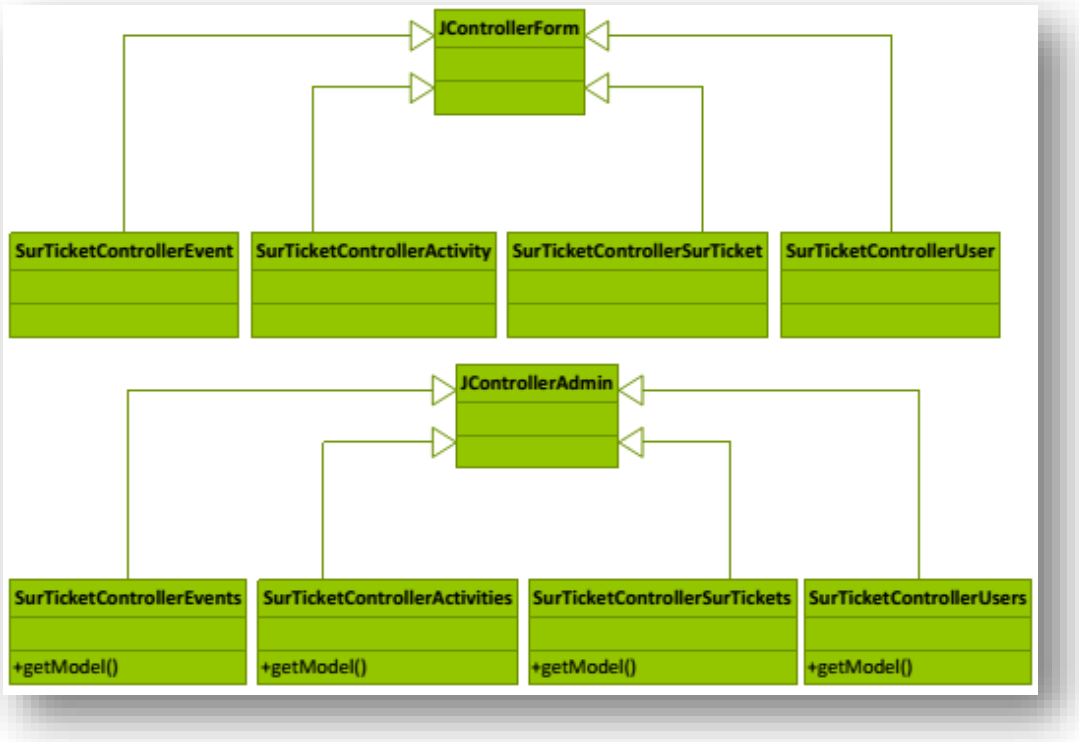


Figura 15 : Diagrama de clases controladores.

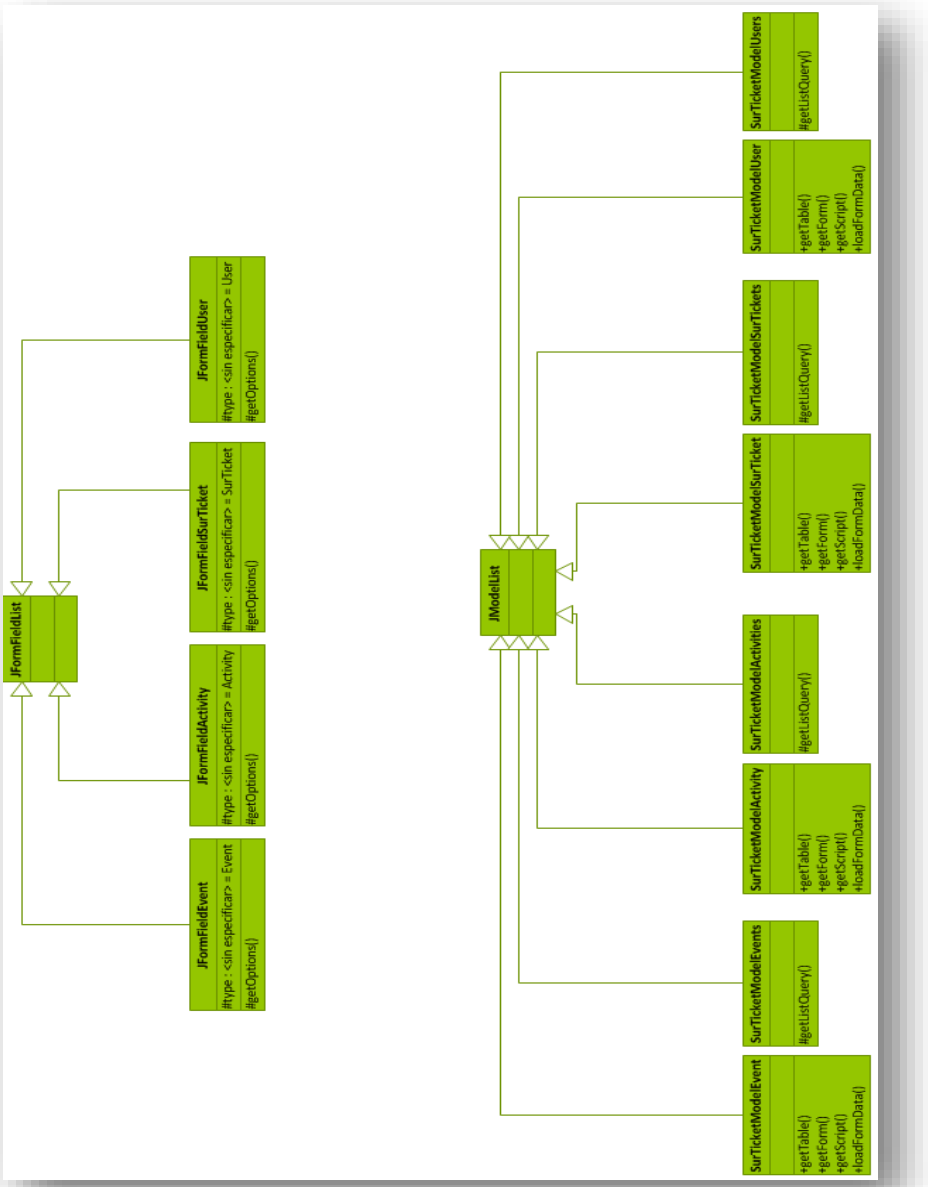


Figura 16 : Diagrama de clases modelos.

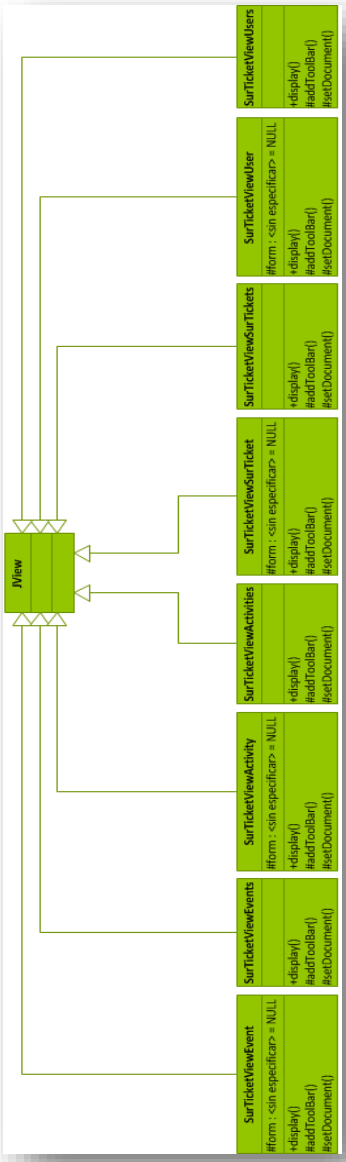


Figura 17 : Diagrama de clases vista.

7.3.2. Lector de Código QR

Una parte fundamental de la aplicación es la lectura del código QR generado durante el proceso de inscripción a un evento. Para esto se requiere de dos versiones diferentes con la misma funcionalidad: una de estas desarrollada bajo plataforma Windows y la otra bajo plataforma Android.

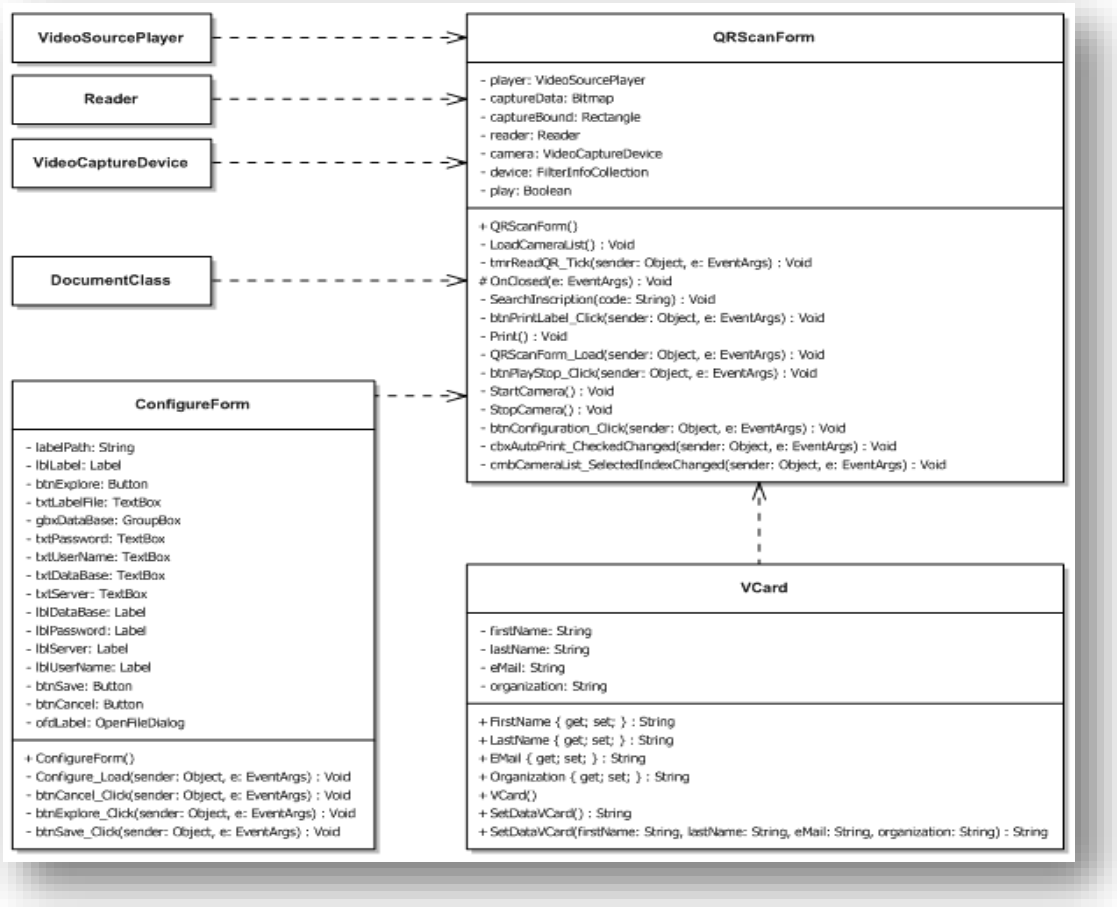


Figura 18 : Diagrama de clases lector QR plataforma Windows.

Esta aplicación permite la lectura del código QR, el cual contiene información para realizar la verificación de la inscripción del asistente, y generar la identificación de éste.

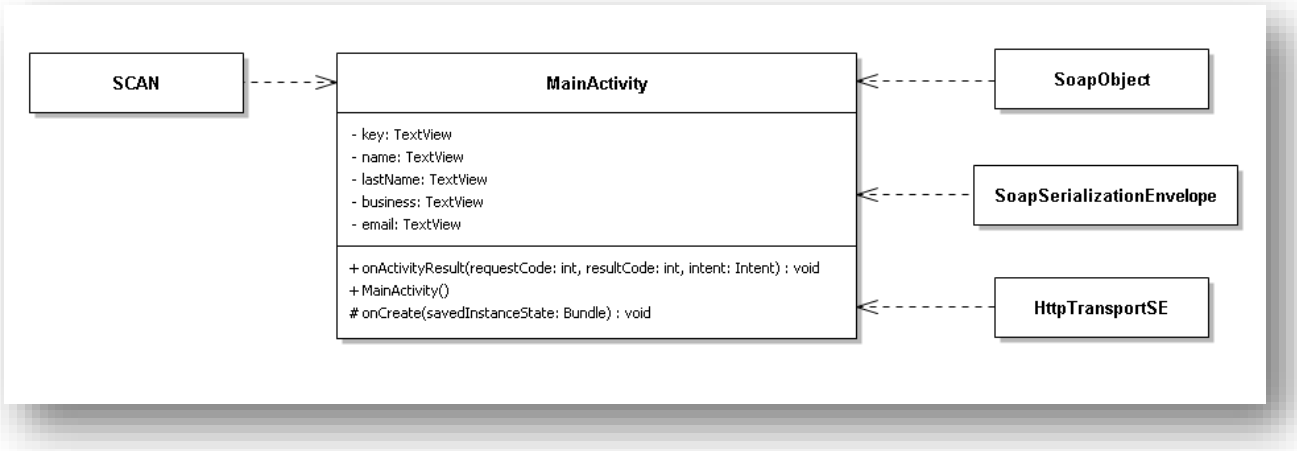


Figura 19 : Diagrama de clases lector códigos QR Android.

Esta versión del lector de códigos QR, diseñada para plataforma Android, logra llevar acabo la función de acreditación del asistente, de manera inalámbrica, logrando una atención más personal.

7.4. Diseño de la Base de Datos

Con el fin de satisfacer las necesidades del sistema, es necesario realizar un diseño de la base de datos, basado en el resultado de los requerimientos obtenidos con anterioridad.

7.4.1. Diseño Conceptual

El diseño conceptual se obtiene de la identificación de las entidades, atributos y sus relaciones, mediante la esquematización de las necesidades reales de una manera fácil de entender.

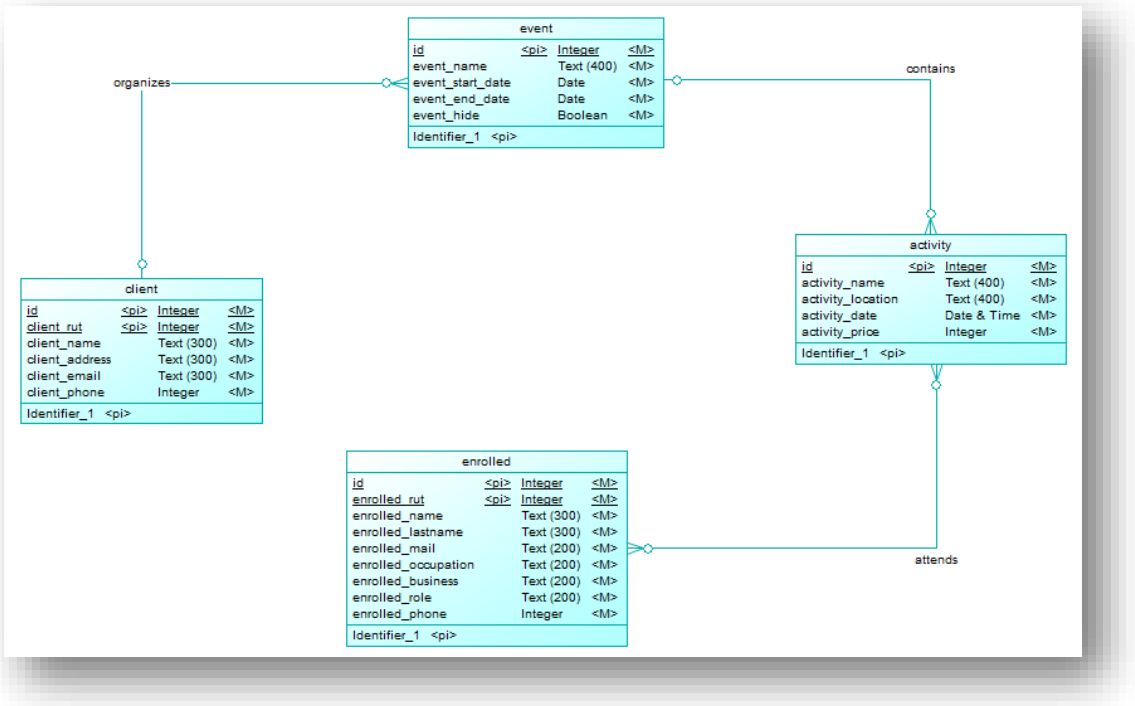


Figura 20 : Diagrama conceptual.

7.4.2. Diseño Físico

El diseño físico es una representación gráfica de la base de datos definitiva, esta se obtiene mediante la utilización de la herramienta Power Designer (donde también se realizó el diseño del modelo anterior de la base de datos), a

través de la conversión del diagrama conceptual, aplicando las reglas necesarias para la perfecta compatibilidad con el motor de base de datos que se utilizó (en este caso MySQL 5.5.34).

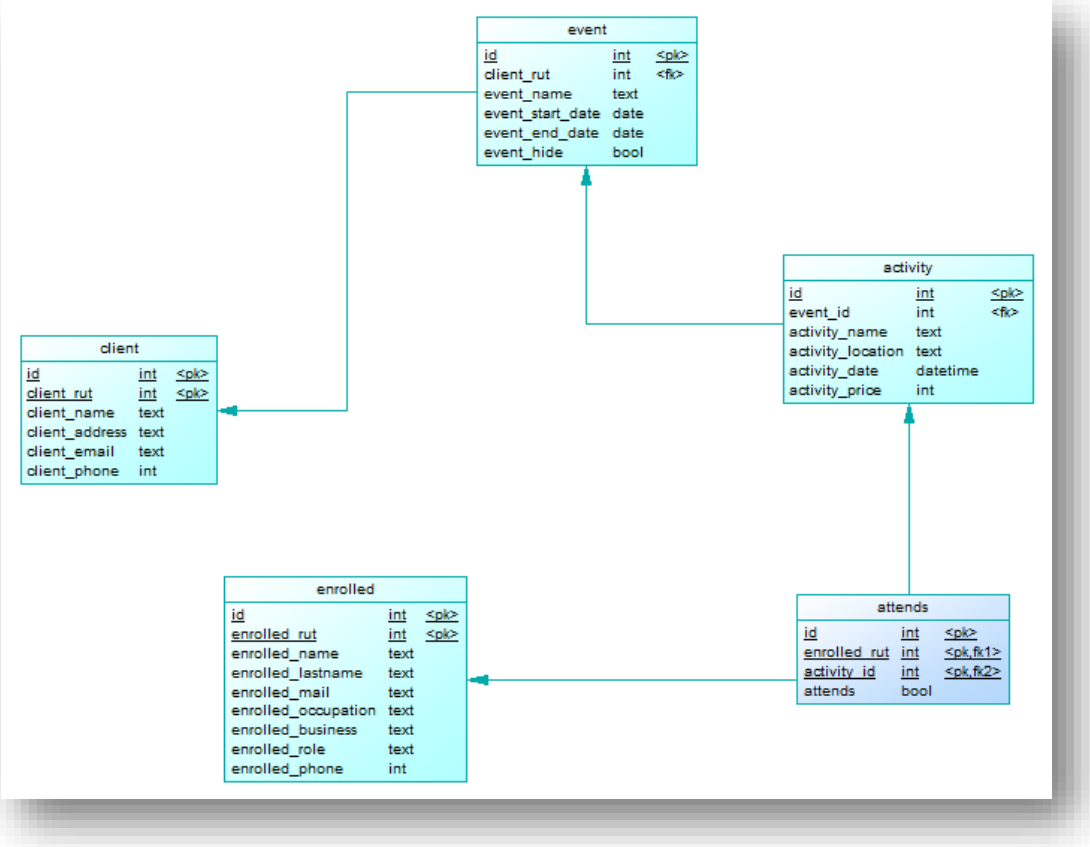


Figura 21 : Diagrama físico.

Tabla	Descripción
Client	Contiene la información relevante sobre las empresas que solicitan los servicios de administración
Event	Contiene información referida al evento que se administrara
Activity	Contiene información correspondiente a las actividades de los eventos
Enrolled	Contiene información sobre las personas que se inscriben en eventos
Attends	Contiene la relación entre el usuario inscrito y el evento.

Tabla 11 : Cuadro de descripción de las tablas de la base de datos.

7.4.3. Script SQL

Mediante la generación del diagrama físico, el cual está normado para el motor de bases de datos MySQL, es posible generar el código **SQL** de aquella representación. A través del uso de Power Designer es posible realizar esta acción de manera simple.

Debido a que el código **SQL** es parte del componente Joomla, éste debe cumplir con cierta característica particular. Joomla antepone un prefijo a cada tabla que utiliza, con el fin de diferenciar a más de una instancia de este **CMS** utilizando la misma base de datos, para lograr esto se debe anteponer un hashtag seguido de doble underscore a cada nombre de tabla (“#__”). Esto se puede lograr indicando a Power Designer que debe realizar esta acción.

A continuación, se presenta un extracto del script SQL generado por Power Designer.

```
drop table if exists `#__attends`;

drop table if exists `#__activity`;

drop table if exists `#__event`;

drop table if exists `#__client`;

drop table if exists `#__enrolled`;

--
-- Table structure for table `#__activity`
--

CREATE TABLE IF NOT EXISTS `#__activity` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `event_id` int(11) NOT NULL,
  `activity_name` varchar(128) NOT NULL,
  `activity_location` varchar(128) NOT NULL,
  `activity_date` date NOT NULL,
  `activity_price` int(11) NOT NULL,
  PRIMARY KEY (`id`,`event_id`),
  KEY `fk_jos25_activity_1` (`event_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO_INCREMENT=1;

--
-- Table structure for table `#__attends`
--
```

```
CREATE TABLE IF NOT EXISTS `#__attends` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `enrolled_rut` char(11) NOT NULL,
  `activity_id` int(11) NOT NULL,
  `attends` tinyint(1) DEFAULT NULL,
  PRIMARY KEY (`id`, `enrolled_rut`,`activity_id`),
  KEY `fk_jos25_attends_1` (`activity_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO_INCREMENT=1;

--
-- Table structure for table `#__client`
--

CREATE TABLE IF NOT EXISTS `#__client` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `client_rut` char(11) NOT NULL,
  `client_name` longtext NOT NULL,
  `client_address` text NOT NULL,
  `client_email` text NOT NULL,
  `client_phone` char(28) NOT NULL,
  PRIMARY KEY (`id`, `client_rut`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO_INCREMENT=1;

--
-- Table structure for table `#__enrolled`
--

CREATE TABLE IF NOT EXISTS `#__enrolled` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `enrolled_rut` char(11) NOT NULL,
  `enrolled_name` varchar(128) NOT NULL,
  `enrolled_lastname` varchar(128) NOT NULL,
  `enrolled_email` varchar(128) NOT NULL,
  `enrolled_occupation` varchar(128) NOT NULL,
  `enrolled_business` varchar(128) NOT NULL,
  `enrolled_role` varchar(128) NOT NULL,
  `enrolled_phone` char(11) NOT NULL,
  PRIMARY KEY (`id`, `enrolled_rut`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO_INCREMENT=1;


--
-- Table structure for table `#__event`
--

CREATE TABLE IF NOT EXISTS `#__event` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `client_rut` char(11) NOT NULL,
  `event_name` varchar(128) NOT NULL,
```

```
  `event_start_date` date NOT NULL,
  `event_end_date` date NOT NULL,
  `published` tinyint(1) NOT NULL,
  PRIMARY KEY (`id`, `client_rut`),
  KEY `fk_jos25_event_1` (`client_rut`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO_INCREMENT=1;
```

Código 1 : Fragmento de código SQL generado por Power Designer.

7.5. Diseño de Interfaz

La interfaz es una parte muy relevante dentro del desarrollo, pues es la manera en que los usuarios interactúan con el sistema. Este debe ser lo más claro e intuitivo posible a fin de rescatar la mejor experiencia por parte de los usuarios.

Este sistema consta de tres partes o módulos que interactúan por lo que se cuenta con más de un tipo de interfaz.

7.5.1. Plataforma Web

A diferencia de otros desarrollos de sitios web donde la interfaz es extremadamente relevante, para el caso de desarrollos de componentes para CMS es menos estricta, debido a que la gran parte de las características gráficas del sitio, están definidas por una plantilla gráfica que ordena y esquematiza el estilo visual del usuario de un sitio implementado con Joomla.

Por el hecho de ser un componente, existen dos tipos de interfaz denominadas front-end y back-end. El front-end es aquella interfaz con la que el usuario del sitio web interactúa, el administrador del sistema puede ubicar ésta en cualquiera de los tres sectores que viene predefinidos por Joomla (centro, lado izquierdo o lado derecho), y la interfaz, como se mencionó anteriormente se ajusta de acuerdo a la plantilla establecida por el administrador.

A continuación, se presenta el prototipo de interfaz definido para el front-end, con una distribución aproximada del resto de los componentes del sitio web.

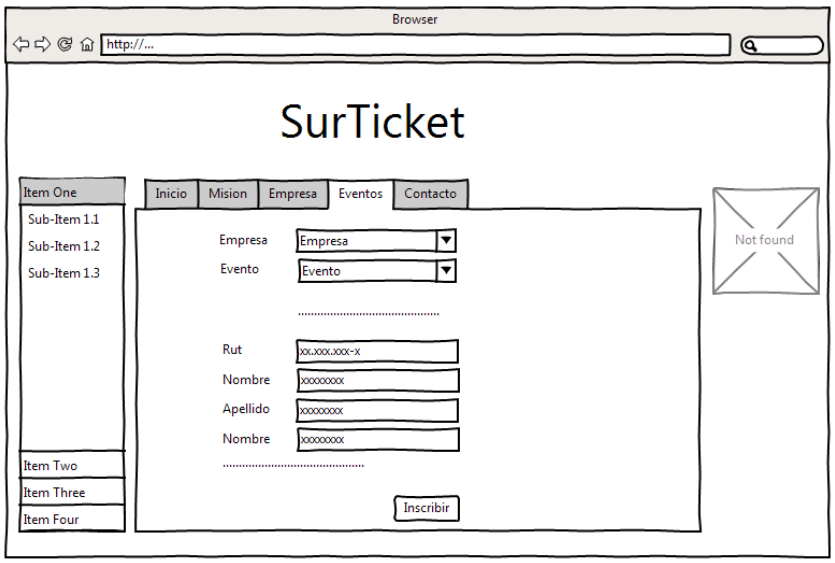


Figura 22 : Front-end o vista del usuario.

7.5.2. Interfaz Plataforma Windows

Esta interfaz depende más del diseño y habilidades del desarrollador, pues no depende gráficamente de los requerimientos del administrador.

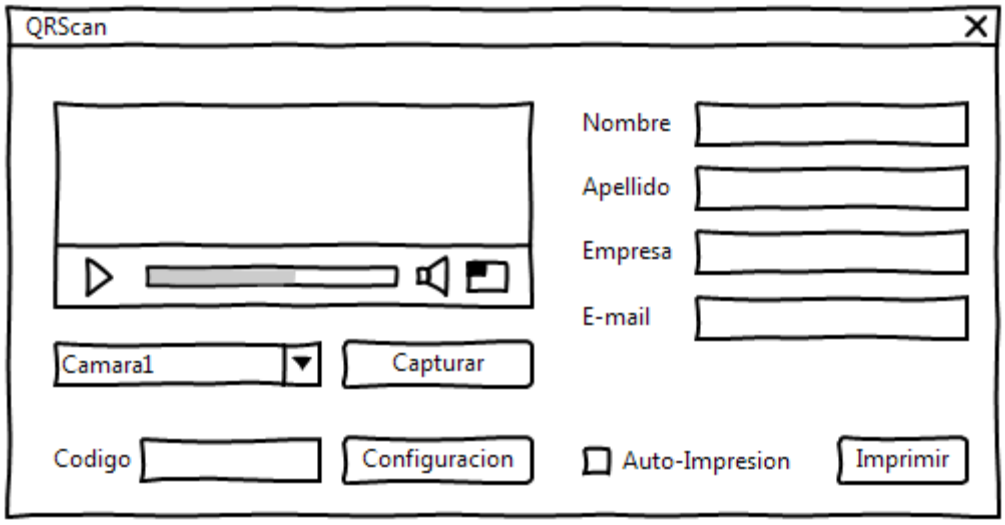


Figura 23 : Interfaz plataforma Windows.

7.5.3. Interfaz Android

Esta interfaz es más reducida por el tamaño de pantalla que puede variar de acuerdo al dispositivo en que se utilizará.

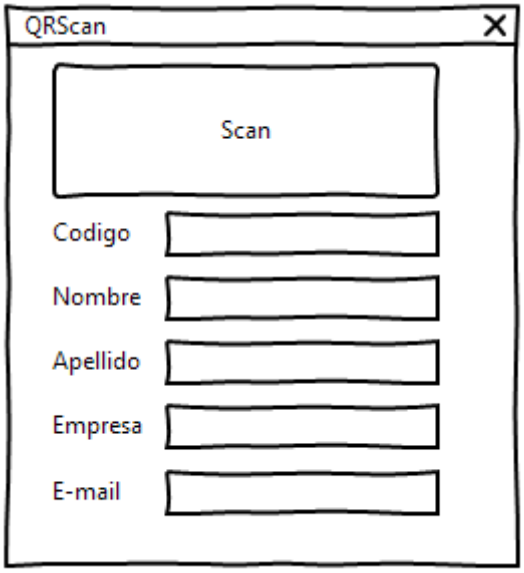


Figura 24 : Interfaz plataforma Android.

8. Desarrollo del Sistema

Finalizado el proceso de análisis y diseño, los cuales fueron descritos en los capítulos anteriores, se procede a la codificación del sistema. Este debe reflejar apropiadamente el conjunto de requerimientos solicitados por quien solicita su construcción.

Entre las herramientas que se utilizaron para su desarrollo se encuentran Adobe Dreamweaver CS6, el cual se utilizó para la codificación del componente Joomla, Visual Studio 2010, en el cual se escribió bajo lenguaje C#, el código correspondiente al lector de códigos QR para plataforma Windows, Eclipse Juno, para el desarrollo bajo plataforma Android. Así como también MySQL Server 5.5 como motor de base de datos.

Con el fin de entender este proceso se explicará cada uno de sus módulos descritos con anterioridad, entregando muestras de las secciones de código más relevantes.

8.1. Desarrollo Componente Joomla

El componente Joomla es uno de los módulos más importante dentro del sistema, por lo que se profundizará un poco más sobre su desarrollo que en el de los otros módulos.

El desarrollo de un componente Joomla requiere de la construcción de un sistema de archivos normado, para su correcto funcionamiento. La carpeta que contiene el componente debe ser nombrado anteponiendo ‘com_’ al nombre que se escoja, internamente contiene dos carpetas y un archivo xml, la carpeta ‘admin’ contiene el back-end del componente y la carpeta ‘site’ contiene el front-end del mismo. El archivo xml contiene la descripción del componente, tales como nombre del componente, versión, autor, etc. esta estructura es mucho más compleja que lo que se ha desglosado por lo que se presentará una captura de la estructura de archivos más comunes.

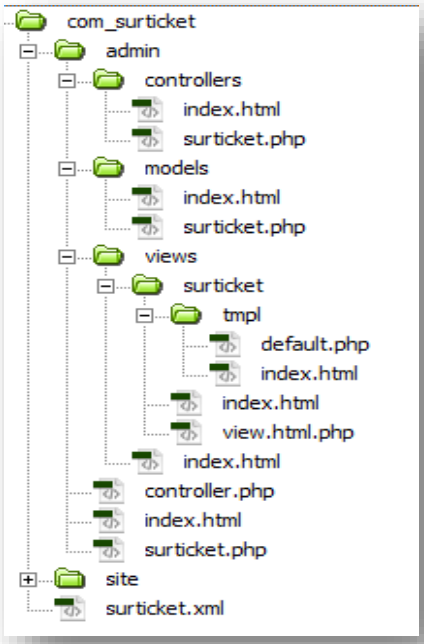


Figura 25 : Estructura de archivos básica de un componente Joomla.

8.1.1. Administración de Clientes

La página de administración de clientes consta de las tres clásicas opciones de mantención, que son crear, editar y eliminar, como se observa en la figura siguiente.

Figura 26 : Captura página de administración.

En esta página de presentación se listan todos los clientes que administra la empresa, la página implementa métodos de paginación por lo que de superar los veinte registros se debe cambiar el rango de muestra. Esta vista está compuesta por varias secciones separadas entre las que se tienen la barra de herramientas, los menús o pestañas de las secciones de administración del componente, cabecera, cuerpo y pie de página.

Para lograr esto, siguiendo la normativa de desarrollo MVC, es necesario un conjunto de archivos de código que representa el resultado final que se aprecia

en la imagen anterior, a continuación se detalla la generación de las interfaz del usuario.

```php
<?php
//Restringe acceso a usuarios autorizados
defined('_JEXEC') or die('Restricted access');
jimport('joomla.application.component.view');
class SurTicketViewSurTickets extends JView
{
	function display($tpl = null) {
		$items = $this->get('Items');
		$pagination = $this->get('Pagination');
		//Manejo de errores en el llamado a la pagina
		if (count($errors = $this->get('Errors'))) {
			JError::raiseError(500, implode('<br />', $errors));
			return false;
		}
		$this->items = $items;
		//Agrega paginacion a la pagina
		$this->pagination = $pagination;
		$this->addToolBar();
		//Despliga el contenido de la pagina
		parent::display($tpl);
		$this->setDocument();
	}
	protected function addToolBar() {
		//Cambia el texto de titulo de la pagina
	JToolBarHelper::title(JText::_('COM_SURTICKET_MANAGER_CLIENTS'),
'surticket');
		//Agrega el boton eliminar a la barra de acciones
		JToolBarHelper::deleteListX('', 'surtickets.delete');
		//Agrega el boton editar a la barra de acciones
		JToolBarHelper::editListX('surticket.edit');
		//Agrega el boton editar a la barra de acciones
		JToolBarHelper::addNewX('surticket.add');
		//Agrega el boton opciones a la barra de acciones
		JToolBarHelper::preferences('com_surticket');
	}//Cambia o define el titulo de la barra titulo del navegador
	protected function setDocument() {
		$document = JFactory::getDocument();
		$document-
>setTitle(JText::_('COM_SURTICKET_ADMINISTRATION'));
	}
}
```

Código 1: Contenido de la vista general archivo view.html.php.

El archivo view.html.php que está ubicado en la carpeta 'views', es el encargado de llamar por defecto al archivo default.php que se encuentra dentro de la carpeta tmpl y el que contiene llamadas a los demás componentes de la vista. A continuación se puede ver el contenido del archivo en cuestión.

```
<?php
defined('_JEXEC') or die('Restricted Access');
JHtml::_('behavior.tooltip');
?>
<form action="<?php echo JRoute::_('index.php?option=com_surticket');
?>" method="post" name="adminForm">
	<table class="adminlist">
		<!-- Llamado al contenido de la cabecera -->
		<thead><?php echo $this->loadTemplate('head');?></thead>
		<!-- Llamado al contenido del pie de pagina -->
		<tfoot><?php echo $this->loadTemplate('foot');?></tfoot>
		<!-- Llamado al contenido del cuerpo -->
		<tbody><?php echo $this->loadTemplate('body');?></tbody>
	</table>
	<div>
		<input type="hidden" name="task" value="" />
		<input type="hidden" name="boxchecked" value="0" />
		<?php echo JHtml::_('form.token'); ?>
	</div>
</form>
```

Código 2 : Contenido del archivo default.php.

Como se puede apreciar en el código, se realizan llamadas a tres plantillas diferentes, 'head', 'foot' y 'body', estas tres plantillas deben ser creadas por el desarrollador con nombres específicos dentro de la carpeta 'tmpl'. Los nombres

de estos son ‘default_head.php’, default_foot.php y default_body.php los que pueden observar a continuación.

```
<?php
defined('_JEXEC') or die('Restricted Access');
?>
<tr>
	<th width="5">
		<?php echo
JText::_('COM_SURTICKET_CLIENT_HEADING_CLIENT_RUT'); ?>
	</th>
	<th width="20">
		<input type="checkbox" name="toggle" value=""
onclick="checkAll(<?php echo count($this->items); ?>);" />
	</th>
	<th>
		<?php echo
JText::_('COM_SURTICKET_CLIENT_HEADING_CLIENT_NAME'); ?>
	</th>
    <th>
		<?php echo
JText::_('COM_SURTICKET_CLIENT_HEADING_CLIENT_ADDRESS'); ?>
	</th>
    <th>
		<?php echo
JText::_('COM_SURTICKET_CLIENT_HEADING_CLIENT_EMAIL'); ?>
	</th>
    <th>
		<?php echo
JText::_('COM_SURTICKET_CLIENT_HEADING_CLIENT_PHONE'); ?>
	</th>
</tr>
```

Código 3 : Contenido del archivo default_head.php.

El código 3 es más claro en cuanto a su contenido, define el encabezado de la tabla que muestra la lista de tuplas almacenadas en la base de datos.

```php
<?php
defined('_JEXEC') or die('Restricted Access');
?>
<tr>
	<td colspan="6"><?php echo $this->pagination->getListFooter();
?></td>
</tr>
```

Código 4 : Contenido del archivo default_foot.php

El código 4 inserta la paginación en la última fila de celdas combinadas, cuyo código viene predefinido dentro de las librerías básicas de Joomla.

```php
<?php
defined('_JEXEC') or die('Restricted Access');
?>

<!-- Llenado de la tabla con el contenido de las tuplas de la base de
datos -->
<?php foreach($this->items as $i => $item): ?>
	<tr class="row<?php echo $i % 2; ?>">
		<td>

			<!-- Obtiene el valor contenido dentro del parametro
contenido en el objeto item -->
			<?php echo $item->client_rut; ?>
		</td>
		<td>
			<?php echo JHtml::_('grid.id', $i, $item->id); ?>
		</td>
		<td>

			<a href="<?php echo
JRoute::_('index.php?option=com_surticket&task=surticket.edit&id=' .
$item->id); ?>">
				<?php echo $item->client_name; ?>
			</a>
		</td>
		<td>
			<?php echo $item->client_address; ?>
		</td>
		<td>
			<?php echo $item->client_email; ?>
		</td>
```

```
        <td>
                <?php echo $item->client_phone; ?>
            </td>
        </tr>
<?php endforeach; ?>
```

Código 5 : Contenido archivo default_body.php.

Finalmente el archivo default_body.php se encarga de rellenar la tabla con el contenido obtenido de la base de datos, el contenido del objeto ítem se llena en la sección que corresponde al modelo. Con este último archivo se termina con la explicación del funcionamiento de la vista para el listado de tuplas contenidas en la tabla cliente, a continuación se describirá el funcionamiento del modelo correspondiente a la vista recién explicada.

```
<?php
defined('_JEXEC') or die('Restricted access');
jimport('joomla.application.component.modellist');
class SurTicketModelSurTickets extends JModelList
{
      protected function getListQuery() {
            $db = JFactory::getDBO();
            $query = $db->getQuery(true);
            //Lista de attributos solicitados en la consulta
            $query-
>select('id,client_rut,client_name,client_address,client_email,client_p
hone');
            //Nombre de la tabla de donde se obtendran los datos
            $query->from('#__client');
            return $query;
      }
}
```

Código 6 : contenido del archivo surtickets.php.

Como se explicó anteriormente el modelo contiene el acceso a datos utilizada por el controlador para obtener los datos a ser representados en la interfaz solicitada por el controlador. Este archivo se encuentra ubicado en la carpeta 'models' perteneciente a la sección que comprende el back-end.

```
<?php
defined('_JEXEC') or die('Restricted access');
jimport('joomla.application.component.controlleradmin');
class SurTicketControllerSurTickets extends JControllerAdmin
{
	//obtiene el modelo
	public function getModel($name = 'SurTicket', $prefix =
'SurTicketModel')
	{
		$model = parent::getModel($name, $prefix,
array('ignore_request' => true));
		return $model;
	}
}
```

Código 7 : Contenido del archivo surticket.php perteneciente al controlador.

El controlador es el que contiene la lógica de negocios, encargado de realizar a las consultas a la base de datos solicitando estos al modelo y entregando los resultados a la vista.

De esta manera trabaja con el patrón MVC utilizado para obtener el listado de datos de la tabla clientes almacenados la base de datos. A continuación, se expondrá el modelo MVC para la inserción, edición y eliminación de datos en la tabla antes mencionada.

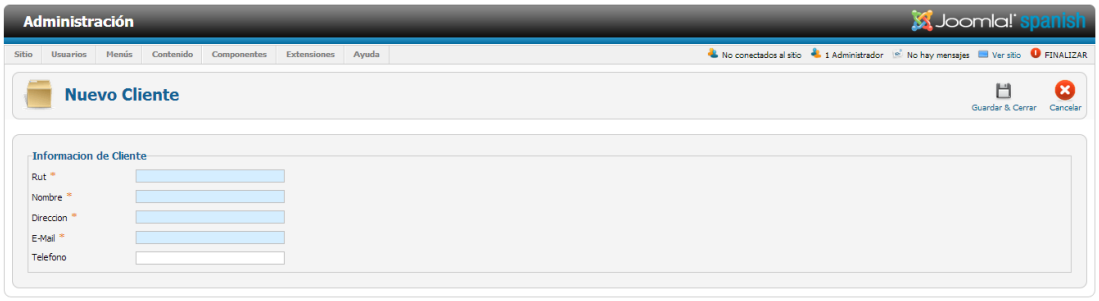


Figura 27 : Captura ventana de creación de nuevo cliente.

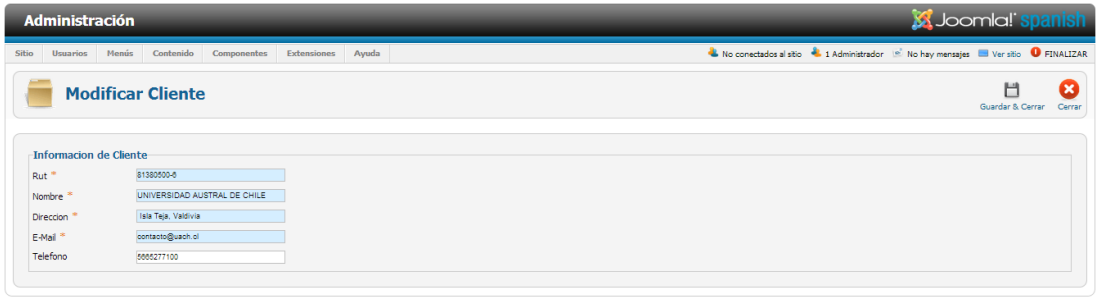


Figura 28 : Captura ventana edición cliente.

Como es posible apreciar la diferencia entre la figura 25 y 26 radica en su función y no en su estética, de esta manera es posible reutilizar código como se explica a continuación.

```php
<?php
defined('_JEXEC') or die('Restricted access');
jimport('joomla.application.component.view');
class SurTicketViewSurTicket extends JView
{
	protected $form = null;
```

```
	public function display($tpl = null)
	{
		$form = $this->get('Form');
		$item = $this->get('Item');
	  $script = $this->get('Script');

		if (count($errors = $this->get('Errors')))
		{
			JError::raiseError(500, implode('<br />', $errors));
			return false;
		}

		$this->form = $form;
		$this->item = $item;
		//Se carga el script de validacion de contenido de texto
	  $this->script = $script;
		//Se carga la barra de opciones de la pagina
		$this->addToolBar();
		parent::display($tpl);
		$this->setDocument();
	}

	protected function addToolBar()
	{
		$input = JFactory::getApplication()->input;
		//Se oculta la barra de opciones anterior
		JRequest::setVar('hidemainmenu', true);
		$isNew = ($this->item->id == 0);
		//Se cambia el titulo en la barra de titulo
		JToolBarHelper::title($isNew ?
JText::_('COM_SURTICKET_MANAGER_CLIENT_NEW') :
JText::_('COM_SURTICKET_CLIENT_EDITING'));
		//Se carga el boton guardar de la barra de opciones
		JToolBarHelper::save('surticket.save');
		//Se carga el boton cancelar de la barra de opciones
		JToolBarHelper::cancel('surticket.cancel', $isNew ?
'JTOOLBAR_CANCEL' : 'JTOOLBAR_CLOSE');
	}

	protected function setDocument()
	{
		$isNew = ($this->item->id < 1);
		$document = JFactory::getDocument();
		//Se cambia el texto de titulo en caso de que sea nuevo
registro o edicion
		$document->setTitle($isNew ?
JText::_('COM_SURTICKET_CLIENT_CREATING') :
JText::_('COM_SURTICKET_CLIENT_EDITING'));
		//Se cargan algunas funciones javascript
	  $document->addScript(JURI::root() . $this->script);
	  $document->addScript(JURI::root() .
"/administrator/components/com_surticket/views/surticket/submitbutton.j
```

```
s");
        JText::script('COM_SURTICKET_SURTICKET_ERROR_UNACCEPTABLE');
      }
}
```

Código 8 : Contenido de archivo view.html.php perteneciente a inserción y edición.

El código 8 contiene la vista para la página crear o editar. Aprovechando la reutilización de código es posible disminuir la cantidad de líneas de código para funciones que son muy similares, pues contienen los mismos campos, su diferencia radica en su funcionamiento.

```
<?php
defined('_JEXEC') or die('Restricted access');
JHtml::_('behavior.tooltip');
JHtml::_('behavior.formvalidation');
?>

<form action="<?php echo
JRoute::_('index.php?option=com_surticket&layout=edit&id='.(int)
$this->item->id); ?>" method="post" name="adminForm" id="surticket-
form" class="form-validate">

      <fieldset class="adminform">
            <legend><?php echo JText::_(
'COM_SURTICKET_CLIENT_DETAILS' ); ?></legend>

            <ul class="adminformlist">

              <!-- Carga los items del formulario -->
              <?php foreach($this->form->getFieldset() as $field): ?>
                  <li><?php echo $field->label;echo $field-
>input;?></li>
              <?php endforeach; ?>
            </ul>
      </fieldset>
```

```
	<div>
		<input type="hidden" name="task" value="surticket.edit" />
		<?php echo JHtml::_('form.token'); ?>
	</div>
</form>
```

Código 9 : contenido del archivo edit.php.

El código anterior representa la carga del formulario, el cual se utiliza tanto para la creación como para la edición de los elementos de la tabla.

```
<?php
defined('_JEXEC') or die('Restricted access');
jimport('joomla.application.component.modeladmin');
class SurTicketModelSurTicket extends JModelAdmin
{
	public function getTable($type = 'SurTicket', $prefix =
'SurTicketTable', $config = array())
	{
		return JTable::getInstance($type, $prefix, $config);
	}
	//Obtiene el formulario
	public function getForm($data = array(), $loadData = true)
	{
		$form = $this->loadForm('com_surticket.surticket',
'surticket', array('control' => 'jform', 'load_data' => $loadData));
		if (empty($form))
		{
			return false;
		}
		return $form;
	}
	//Carga un javascript
	public function getScript()
	{
		return
'administrator/components/com_surticket/models/forms/surticket.js';
	}
	//Obtiene los datos de un registro y los carga en el formulario
	protected function loadFormData()
	{
		$data = JFactory::getApplication()-
>getUserState('com_surticket.edit.surticket.data', array());
```

```
            if (empty($data))
            {
                  $data = $this->getItem();
            }
            return $data;
      }
```

Código 10: Código perteneciente al modelo que se utiliza para la inserción y edición de elementos de la tabla clientes archivo surticker.php.

El modelo anterior corresponde a la implementación de los métodos utilizados para trabajar con la manipulación de elementos de la tabla clientes.

Se han definido en los pasos anteriores las páginas de listar clientes y la página de mantención de esta tabla, completando con esto el patrón modelo vista controlador. Estas implementaciones se replican para el resto de las tablas que necesitan mantención, basta con cambiar los parámetros y la tabla con la cual trabajar en el modelo de administración de cada tabla, así como se puede visualizar en el siguiente modelo.

```
<?php
defined('_JEXEC') or die('Restricted access');
jimport('joomla.application.component.modellist');
class SurTicketModelUsers extends JModelList
{
      protected function getListQuery()
      {
            $db = JFactory::getDBO();
            $query = $db->getQuery(true);
            $query-
>select('id,enrolled_rut,enrolled_name,enrolled_lastname,enrolled_email
```

```
,enrolled_occupation,enrolled_business,enrolled_role,enrolled_phone');
            $query->from('#__enrolled');
            return $query;
        }
}
```

Código 11 : Contenido archivo users.php, modelo utilizado para la administración de inscritos.

Este patrón de diseño se replica tanto para el back-end (página de administración) como para el front-end (página de acceso para el usuario). El front-end es la página a la cual el usuario que desea inscribirse a un evento debe acceder, esta página no contiene ningún tipo de seguridad pues lo que se desea es que este tenga una experiencia grata y simple al utilizarla.

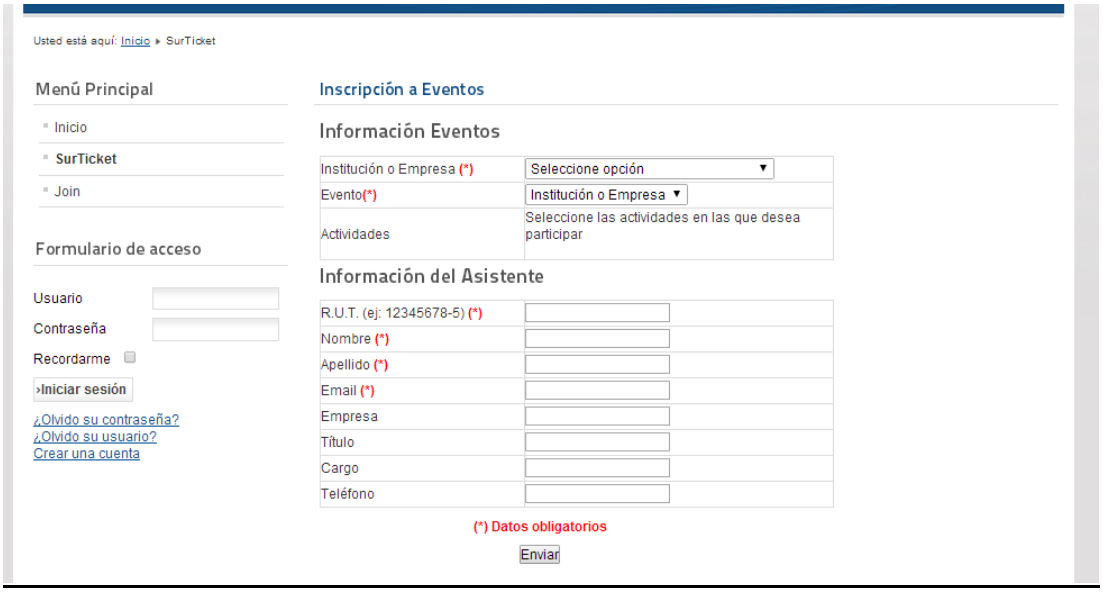


Figura 29 : Front-end o página de inscripción.

Una de las características importantes en esta sección es la utilización de AJAX para algunas de las funciones requeridas en esta vista.

```
function surticket_ajax(){
      var xmlhttp=false;
      try {
            xmlhttp = new ActiveXObject("Msxml2.XMLHTTP");
      } catch (e) {
            try {
                  xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
            } catch (E) {
                  xmlhttp = false;
            }
      }if (!xmlhttp && typeof XMLHttpRequest!='undefined') {
            xmlhttp = new XMLHttpRequest();
      }
      return xmlhttp;
}
//Devuelve la lista de eventos del cliente
function getEventsClient(client_rut)
{
    ajax = surticket_ajax();
    var url =
"index.php?option=com_surticket&view=surticket&task=getEventsClient&cli
ent_rut=" + client_rut;
    ajax.open("post", url, true);
    ajax.onreadystatechange = function() {
        if (ajax.readyState == 4) {
            list_events.innerHTML = ajax.responseText
            list_activities.innerHTML = ""
      }
    }
    ajax.send(null)
}
```

Código 12 : Función AJAX que devuelve la lista de eventos de un cliente.

```
// Retorna la lista de actividades que tiene el evento del cliente
function getActivitiesEvents(event_id)
{
    ajax = surticket_ajax();
    var url =
"index.php?option=com_surticket&view=surticket&task=getActivitiesEvents
&event_id=" + event_id;
    ajax.open("post", url, true);
    ajax.onreadystatechange = function() {
        if (ajax.readyState == 4) {
            list_activities.innerHTML = ajax.responseText
      }
    }
    ajax.send(null)
}
```

Código 13 : Función AJAX que retorna la lista de actividades de un evento.

```
function getUserInfo(enrolled_rut)
{
    ajax = surticket_ajax();
    var url =
"index.php?option=com_surticket&view=surticket&task=getUserInfo&enrolle
d_rut="+enrolled_rut;
    ajax.open("post", url, true);
    ajax.onreadystatechange = function()
    {
        if (ajax.readyState == 4)
        {
            info_user = ajax.responseText.split("#");
            document.getElementById("enrolled_name").value =
(info_user[0] == "") ? "" : info_user[0];
            document.getElementById("enrolled_lastname").value =
info_user[1];
            document.getElementById("enrolled_email").value =
info_user[2];
            document.getElementById("enrolled_business").value =
info_user[4];
            document.getElementById("enrolled_occupation").value =
info_user[3];
            document.getElementById("enrolled_role").value =
info_user[5];
            document.getElementById("enrolled_phone").value =
info_user[6];
```

```
        }
    }
    ajax.send(null)
}
```

Código 14 : Función AJAX que retorna los datos de un usuario que se ha inscrito anteriormente.

Una de las funcionalidades primordiales del sistema es el pago por inscripción, cuyo funcionamiento se ve reflejado en el siguiente código.

```
public function payWithPaypal()
{
      $component = JComponentHelper::getComponent('com_surticket');
      $paypal_email = $component->params->get('paypal_email');
      if($paypal_email != "")
      {
         $currency_code = ($component->params->get('currency_code') ==
'') ? 'USD' : $component->params->get('currency_code');
         $country_code = ($component->params->get('country_code') ==
'') ? 'CL' : $component->params->get('country_code');
        $paypal_amount = $_GET["paypal_amount"];
        switch ($component->params->get('button_language'))
        {
            case "es_ES": $button_language = "es"; break;
            case "en_EN": $button_language = "en"; break;
            case "us_US": $button_language = "en"; break;
            default: $button_language = "es";
        }
        $paypal_form = "<form name=\"com_surticket_paypal\"
id=\"com_surticket_paypal\" action=\"_____" method=\"post\"
target=\"_blank\">
                                <strong style=\"line-height: normal;\">

<label>".JText::_("COM_SURTICKET_PAY")."</label>
                                    <input type=\"hidden\"
name=\"business\" value=\"$paypal_email\">
                                    <input type=\"hidden\" name=\"cmd\"
value=\"_cart\">
                                    <input type=\"hidden\"
```

```
name=\"amount\" id=\"paypal_amount\" value=\"$paypal_amount\">
                                        <input type=\"hidden\"
name=\"item_name\"
value=\"".JText::_("COM_SURTICKET_LIST_ACTIVITIES_SUSCRIBED_BY_USER")."
\">
                                        <input type=\"hidden\" name=\"add\"
value=\"1\">
                                        <input type=\"hidden\"
name=\"currency_code\" value=\"$currency_code\">
                                        <input type=\"hidden\" name=\"lc\"
value=\"$country_code\">
                                        <input type=\"hidden\"
name=\"charset\" value=\"utf-8\">
                                        <input type=\"image\"
name=\"submit\" style=\"border: 0;\"
src=\"".JURI::base()."media/com_surticket/images/".$button_language."_b
tn_surticket_paypal.jpg\" alt=\"PayPal - The safer, easier way to pay
online\" />
                                    </strong>
                                </form>";
        echo $paypal_form;
        }
    else
        echo "";
    exit();
}
```

Código 15 : Función que permite el pago de eventos mediante PayPal.

Como resultado de la inscripción al correo del usuario o inscrito debe llegar un correo con su inscripción como se aprecia en la siguiente figura.

Figura 30 : Ejemplo PDF resultado de la inscripción.

```
public function generateQrCode()
{
	$events_client = $_GET["events_client"];
	$enrolled_rut = $_GET["enrolled_rut"];
	$enrolled_email = $_GET["enrolled_email"];
	$activity_selected = $_GET["activity_selected"];
	$enrolled_name = $_GET["enrolled_name"];

	//--	Obteniendo codigo QR	--//
	echo
"<label>".JText::_("COM_SURTICKET_ATTEND_SAVED")."</label><br/>";
	echo "<iframe

	src=\"".JURI::base()."components/com_surticket/phpqrcode/index.ph
p?enrolled_rut=$enrolled_rut&events_client=$events_client\"
title=\"QR\"
		frameborder=\"0\" border=\"0\" width=\"0\"
height=\"0\" scrolling=\"no\"
		>
		</iframe>";
```

```
	//--	Generar pdf	--//
	$db = JFactory::getDbo();
	// Obtener nombre del evento
	$query = $db->getQuery(true);
	$query->select($db->quoteName(array('event_name')));
	$query->from($db->quoteName('#__event'));
	$query->where($db->quoteName('id') . '=\''. $events_client.'\'');
	$db->setQuery($query);
	$result2 = $db->loadObject();
	// Obtener nombre de las actividades
	$activities = explode(",", $activity_selected);
	$activities_names = "";
	for($j = 0; $j < count($activities); $j++)
	{
		$query = $db->getQuery(true);
		$query->select($db->quoteName(array('activity_name')));
		$query->from($db->quoteName('#__activity'));
		$query->where($db->quoteName('id') .
'=\''.$activities[$j].'\'');
		$db->setQuery($query);
		$result2_activity = $db->loadObject();
		$activities_names .= "(ID:
".$activities[$j].")".$result2_activity->activity_name." ";
		if($j < (count($activities) - 1))
			$activities_names .= ",";
	}

	echo "<iframe

	src=\"".JURI::base()."components/com_surticket/tcpdf/examples/ind
ex.php?img=$enrolled_rut.png&enrolled_rut=$enrolled_rut&id_evento=$even
ts_client&enrolled_name=$enrolled_name&activities_selected_name=$activi
ties_names&event_name=$result2-
>event_name&tag_name=".JText::_("COM_SURTICKET_TAG_NAME")."&tag_rut=".J
Text::_("COM_SURTICKET_TAG_RUT")."&tag_actsub=".JText::_("COM_SURTICKET
_TAG_SUBSCRITED_ACTIVITIES")."&tag_footer=".JText::_("COM_SURTICKET_TAG
_FOOTER")."\"
			frameborder=\"0\" border=\"0\" width=\"0\"
height=\"0\" scrolling=\"no\"
		>
		</iframe>";
	$mailer = JFactory::getMailer();
	$config = JFactory::getConfig();
	$sender = array($config->getValue('config.mailfrom'), $config-
>getValue('config.fromname'));
	$mailer->setSender($sender);
	$mailer->addRecipient($enrolled_email);
	$body   = JText::_("COM_SURTICKET_ATTEND_SEND_QR_EMAIL_BODY");
	$mailer-
>setSubject(JText::_("COM_SURTICKET_ATTEND_SEND_QR_EMAIL_SUBJECT"));
	$mailer->setBody($body);
```

```
      $mailer-
>addAttachment(JPATH_COMPONENT."/tcpdf/examples/data/$enrolled_rut.pdf"
);
      $send = $mailer->Send();
      if ( $send !== true ) {
            echo
"<label>".JText::_("COM_SURTICKET_ATTEND_SEND_QR_EMAIL_SENDED_ERROR")."
</label><br/>";
      } else {
            echo
"<label>".JText::_("COM_SURTICKET_ATTEND_SEND_QR_EMAIL_SENDED")."</labe
l><br/>";
      }
      echo
"<label>".JText::_("COM_SURTICKET_ATTEND_DOWNLOAD_INVITATION")."
                  <a
href=\"".JURI::base()."components/com_surticket/tcpdf/examples/data/$en
rolled_rut.pdf\" target=\"_blank\">

      ".JText::_("COM_SURTICKET_ATTEND_DOWNLOAD_HERE")."
                  </a>
            </label>";exit();
}
```

Código 16 : Función que genera código QR y el PDF que envía por correo electrónico.

8.2. Lector Códigos QR Plataforma Windows

El lector de códigos QR, es otra arista de este sistema, la función de este es la de leer el código QR presente en la invitación, marcar la asistencia del inscrito y generar el gafete imprimiéndolo sobre una etiqueta.

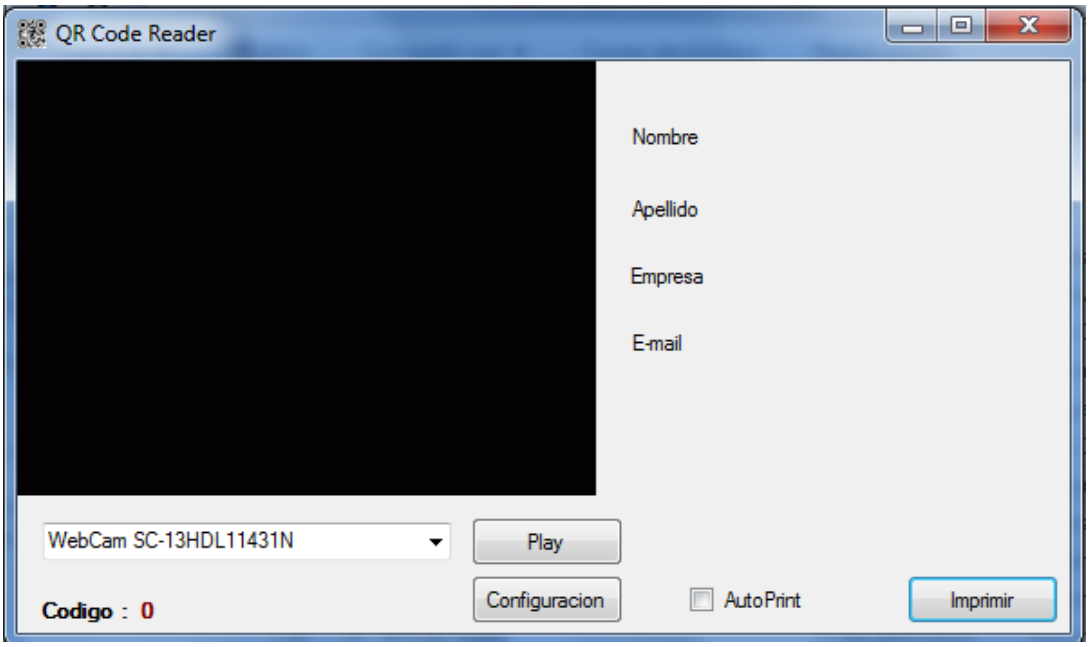


Figura 31 : Interfaz lector código QR.

Para inicializar la lectura de un código QR es necesario la captura de video desde una cámara web, como se aprecia en el código 17 y luego realizar el análisis de los datos capturados. La librería de código abierto Xzing permite el análisis e interpretación de las imágenes capturadas, proceso definido en el código 18.

```
private void StartCamera()
{
    player = new VideoSourcePlayer();
    player.Width = 320;
    player.Height = 240;
    device = new FilterInfoCollection(FilterCategory.VideoInputDevice);
```

```
    camera = new
VideoCaptureDevice(device[cmbCameraList.SelectedIndex].MonikerString);
    player.VideoSource = camera;
    player.Start();
    this.Controls.Add(player);
    captureBound = new Rectangle(player.Location.X, player.Location.Y,
player.Width + 100, player.Height);
    captureData = new Bitmap(player.Width, player.Height);
    Graphics g = Graphics.FromImage(captureData);
    reader = new QRCodeReader();
    hint = new Hashtable();
    hint.Add(DecodeHintType.POSSIBLE_FORMATS, BarcodeFormat.QR_CODE);
    tmrReadQR.Tick += new EventHandler(tmrReadQR_Tick);
}
```

Código 17 : Inicia captura desde la cámara.

```
private void tmrReadQR_Tick(object sender, EventArgs e)
{
    DrawToBitmap(captureData, captureBound);
    RGBLuminanceSource source = new RGBLuminanceSource(captureData,
captureData.Width, captureData.Height);
    BinaryBitmap img = new BinaryBitmap(new GlobalHistogramBinarizer(source));
    Result result = null;
    try {
        result = reader.decode(img, hint);
    }
    catch { }

    if (result == null) {
        CodeReadLBL.Text = "0";
    }
    else
    {
        System.Media.SystemSounds.Beep.Play();
        string readchar = result.Text;
        string[] dataarray = readchar.Split('#');
        CodeReadLBL.Text = readchar;
        SearchInscription(dataarray[3]);
    }
}
```

Código 18 : Obtiene datos desde QR.

```
private void SearchInscription(string code)
{
    MySqlConnection connection = new MySqlConnection();
    string connectionString = "Server=" + Settings.Default.serverAddress + ";
Database=" + Settings.Default.dbName + "; Uid=" + Settings.Default.dbUser + ";
Pwd=" + Settings.Default.dbPassword + ";";
    connection.ConnectionString = connectionString;
    try
    {
        connection.Open();
        MySqlCommand command = connection.CreateCommand();
        command.CommandText = "SELECT * FROM enrolled_rut WHERE scode = '" + code
+ "'";
        MySqlDataReader reader = command.ExecuteReader();
        while (reader.Read())
        {
            tmrReadQR.Stop(); btnPrintLabel.Enabled = true;
            lblFirstNameLoaded.Text = reader['enrolled_name'];
            lblLastNameLoaded.Text = reader['enrolled_lastname'];
            lblCompanyLoaded.Text = reader['enrolled_business'];
            lblMail.Text = reader['enrolled_email'];
        }
        reader.Dispose();connection.Close();
    }
    catch (Exception e) { MessageBox.Show(e.Message); }
    if (cbxAutoPrint.Checked == true)
        Print();
}
```

Código 19 : Recupera datos del usuario desde la base de datos.

En el siguiente código se realiza la impresión del gafete en una impresora de etiquetas, la que contiene datos como nombre, empresa, y un código QR de tipo VCard para facilitar el traspaso de información entre los asistentes al evento. Esta VCard contiene sólo datos como nombre, empresa y e-mail del usuario.

```
void Print()
{
    try
    {
        VCard vcard = new VCard();
```

```
        string strFileLabelPath = "";
        if (Settings.Default.label != null)
            strFileLabelPath = Settings.Default.label.ToString();
        bpac.DocumentClass doc = new bpac.DocumentClass();
        doc.Open(strFileLabelPath);
        doc.GetObject("nombre").Text = lblFirstNameLoaded.Text;
        doc.GetObject("apellido").Text = lblLastNameLoaded.Text;
        doc.GetObject("institucion").Text = lblCompanyLoaded.Text;
        doc.GetObject("qr").Text = vcard.SetDataVCard(lblFirstNameLoaded.Text,
lblLastNameLoaded.Text, lblCompanyLoaded.Text, lblMail.Text);
        doc.StartPrint("", bpac.PrintOptionConstants.bpoDefault);
        doc.PrintOut(1, bpac.PrintOptionConstants.bpoDefault);
        doc.EndPrint();
        doc.Close();
    }
    catch (Exception e)
    {
        MessageBox.Show("Error impresora o etiqueta no encontrada\n" +
e.Message);
    }

}
```

Código 20 : Función que realiza la impresión de la etiqueta.

8.3. Lector Código QR Plataforma Android

Como requisito final de este sistema, corresponde a la creación de una aplicación Android que permita realizar similares funciones que su homólogo de plataforma Windows. Gracias a que la librería usada anteriormente para el análisis e interpretación de códigos QR, existe para múltiples lenguajes, entre ellos Java, es posible una implementación similar a la descrita anteriormente.

Figura 32 : interfaz aplicación Android.

```
scanBtn.setOnClickListener(new OnClickListener() {
@Override
public void onClick(View v) {
	try {
	Intent intent = new Intent("com.google.zxing.client.android.SCAN");
	intent.putExtra("SCAN_MODE", "QR_CODE_MODE,PRODUCT_MODE");
			startActivityForResult(intent, 0);
	} catch (Exception e) {
		e.printStackTrace();
		Toast.makeText(getApplicationContext(), "ERROR:" + e, 1).show();
	}
   }
}
```

Código 21 : Inicio de captura de video.

```
public void onActivityResult(int requestCode, int resultCode, Intent intent)
{
      if (requestCode == 0) {

            if (resultCode == RESULT_OK) {

                  tvResult.setText(intent.getStringExtra("SCAN_RESULT"));
            } else if (resultCode == RESULT_CANCELED) {
                  tvStatus.setText("Press a button to start a scan.");
                  tvResult.setText("Scan cancelled.");
            }
      }
}
```

Codigo 22 : Obtiene datos código QR.

8.4. Control de Versiones

Con el fin de controlar el progreso del proyecto y administrar las versiones que existieron a lo largo de su ciclo de construcción, se implementó la utilización de un repositorio **SVN** remoto. Esto provee no sólo una manera de registrar el progreso del proyecto, sino que también mantener copias de seguridad de cada uno de los puntos de avance en el tiempo, pudiendo retornar a un punto arbitrario de la construcción en caso de ser necesario.

Se utilizó un repositorio para cada módulo en construcción. Debido a que se utilizan herramientas de desarrollo diferentes para cada módulo, se utilizaron las funciones de control de versión que implementa cada una de éstas. En Dreamweaver se utilizó la administración de sitios, herramienta incorporada en este IDE, Eclipse también implementa accesos a **SVN** y para Visual Studio se

utilizó Visual**SVN**, el cual es un plugin que dota a VS de la capacidad de administrar un SVN por proyecto.

También se implementó en el servidor remoto Web**SVN** así cualquier persona podría ver el progreso y el código que existía hasta ese minuto.

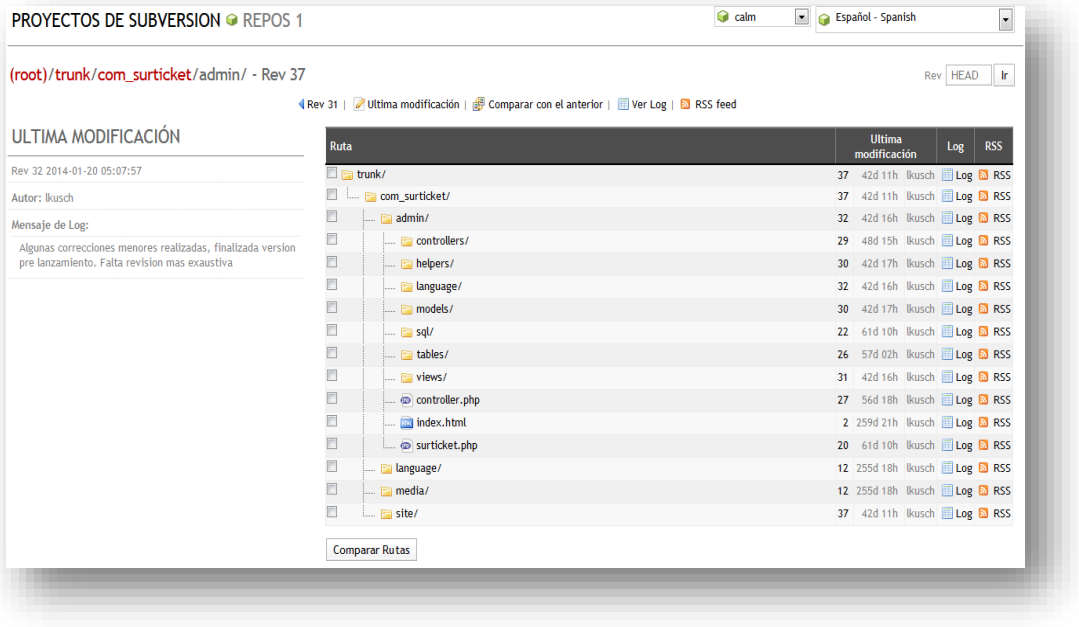


Figura 33 : Captura visualización repositorio componente Joomla mediante websvn.

9. Pruebas del Sistema

Finalizada la etapa de construcción se requiere proceder con un conjunto de pruebas necesarias para verificar el correcto funcionamiento del sistema.

9.1. Pruebas de Caja Blanca

Estas tienen como objetivo probar estatutos condicionales, bucles y ciclos del sistema.

A continuación, se analizará a modo de ejemplo el método que realiza la lectura e interpretación del contenido un código QR.

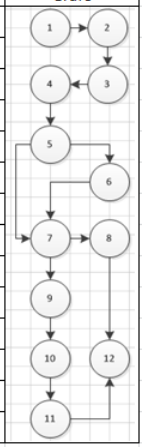
N°	tmrReadQR_Tick	Grafo
	private void tmrReadQR_Tick(object sender, EventArgs e) {	
1	DrawToBitmap(captureData, captureBound);	
2	RGBLuminanceSource source = new RGBLuminanceSource(captureData, captureData.Width, captureData.Height);	
3	BinaryBitmap img = new BinaryBitmap(new GlobalHistogramBinarizer(source));	
4	Result result = null;	
5	try {result = reader.decode(img, hint); }	
6	catch { }	
7	if (result == null) {	
8	CodeReadLBL.Text = "0"; }	
	else {	
9	string[] dataarray = result.Text.Split('#');	
10	CodeReadLBL.Text = result.Text;	
11	SearchInscription(dataarray[3]); }	
12	}	

Tabla 12 : Prueba de caja blanca, método tmrReadQR_Tick.

Los resultados de la prueba de caja blanca aplicada sobre el método mencionado anteriormente se muestran en la siguiente tabla:

N°	Caminos
1	1-2-3-4-5-6-7-8-12
2	1-2-3-4-5-7-8-12
3	1-2-3-4-5-7-9-10-11-12
Resultados	
N° de nodos: 12	
N° de aristas: 14	
Formula 14 - 13 + 2	
Complejidad ciclomatica: 3	

Tabla 13 : Resultado caja blanca, método tmrReadQR_tick.

N°	Camino	Caso Prueba	Resultado Esperado	Resultado Real
1	1-2-3-4-5-6-7-8-12	Error capturando dato result NULL	Definir texto etiqueta en 0	Define texto etiqueta en 0
2	1-2-3-4-5-7-8-12	No error capturando dato result NULL	Definir texto etiqueta en 0	Define texto etiqueta en 0
3	1-2-3-4-5-7-9-10-11-12	No error capturando dato result distinto NULL	Definir texto etiqueta con valor obtenido	Define texto etiqueta con valor obtenido

Tabla 14 : Casos de prueba caja blanca método tmrReadQR_tick.

9.2. Pruebas de Caja Negra

Las pruebas de caja negra consisten en ingresar datos, tanto válidos como no válidos y observar si el comportamiento del módulo es el adecuado. Con el fin de ejemplificar estas pruebas se mostrará el caso correspondiente a la creación de nuevos clientes.

Para realizar estas pruebas es necesario definir la siguiente tabla de equivalencia.

Condición de Entrada	Tipo	Clase Equivalente Valida	Clase Equivalente invalida
Rut	Cadena de Caracteres	1: Cadena de caracteres que cumpla con la expresión de un rut	2: Cadena en blanco 3: Cadena de caracteres que no cumpla con la expresión de un rut
Nombre	Cadena de Caracteres	4: Cualquier cadena de caracteres	5: Cadena en blanco
Dirección	Cadena de Caracteres	6: Cualquier cadena de caracteres	7: Cadena en blanco
E-Mail	Cadena de Caracteres	8: Cadena de caracteres que cumpla con la expresión de un correo electrónico	9: Cadena en blanco 10: Cadena de caracteres que no cumpla con la

			expresión de un correo electrónico
Teléfono	Cadena de Caracteres	11: Cadena en blanco 12: Cadena de caracteres que contenga números o signo '+' al inicio	13: Cadena de caracteres con letras o símbolos distinto de '+' al inicio

Tabla 15 : Clases de equivalencia de crear cliente.

Tomando en cuenta que la interfaz gráfica no permite el ingreso de ciertos campos en blanco, correos electrónicos o números telefónicos que no cumplan el formato se obtienen los siguientes casos de prueba.

Caso	Clase de Equivalencia	Rut	Nombre	Dirección	E-Mail	Teléfono	Resultado
1	3-4-6-8-11	35463-3	Gran Empresa	Sin Calle	gran@empresa.cl		Rut inválido
2	1-4-6-8-12	13520052-2	Leonardo Kusch	Bajo El Puente	lkusch@msn.cl	12345	Guardar Cliente
3	1-4-7-8-11	14226257-6	Verenna Yañez		elcorreo@gmail.com	12323	Faltan Datos
4	1-4-6-10-11	8138050-6	UACh	Los pinos s/n	uach@uach-cl		Mail Inválido

5	1-4-6-8-13	111111 11-1	Test	No se	x@x.cl	#24325	Teléfono Inválido

Tabla 16 : Casos de prueba crear cliente.

Las pruebas ejecutadas fueron de utilidad para poder depurar algunos errores encontrados durante este proceso de pruebas.

10. Conclusiones y/o Recomendaciones

El objetivo de este Seminario de Título era crear un sistema que permita el apoyo a los procesos de inscripción y acreditación de eventos para el uso de la empresa SurTicket.

El sistema desarrollado es capaz de agilizar el proceso de inscripción y acreditación de los usuarios o asistentes de eventos. Gracias a este sistema la empresa SurTicket podrá ofrecer un mejor servicio de administración de eventos, logrando los objetivos de satisfacción del cliente y adquirir una herramienta que le permitirá posicionarse en el mercado local.

La metodología XP, ha servido de gran utilidad para ayudar a mantener un orden y agilizar los procesos de desarrollo en sus diferentes etapas. Además de contribuir con la relación cliente - desarrollador para así lograr un mejor entendimiento de las necesidades o requerimientos que se busca cubrir. Entregando prototipos constantes y de esta manera generar retroalimentación, obteniendo los resultados esperados por el cliente y así contribuir con la generación de un ambiente cálido de trabajo. En lo que refiere a las herramientas utilizadas, contribuyeron de manera significante con la agilización de los procesos de desarrollo. Dentro de las herramientas que se destacan por su contribución con la agilización del desarrollo está Visual Studio .net al contribuir con autocompletación de código y sugerencias al momento de digitar,

otra herramienta que destaca es el uso de repositorios, los que permiten mantener un control histórico sobre los pasos seguidos en la construcción de una aplicación y la seguridad de mantener un respaldo progresivo del sistema, evitando cualquier pérdida o daño durante el proceso de desarrollo.

Se puede concluir desde el punto de vista del desarrollador, que la utilización MVC para el desarrollo de componentes Joomla, puede ser de gran utilidad para la reducción del tiempo de desarrollo, a pesar de ser un poco compleja la absorción del paradigma de desarrollo establecida por éste.

Desde el punto de vista del cliente, es posible estimar que esta aplicación ayudará a la empresa SurTicket a posicionarse como empresa de administración de eventos, contribuyendo con la satisfacción de los usuarios a quienes presten sus servicios.

Se deja planteado para una siguiente etapa, con el fin de entregar un mejor servicio, la inclusión de un medio de pago en moneda nacional a través de los servicios bancarios online, como WebPay, tomando en cuenta que el servicio que se presta, está pensado en el mercado nacional. Otro punto recomendable para analizar a futuro, es la generación de estadísticas a partir de la asistencia a los eventos e implementar la asistencia por actividad, de esta manera se puede entregar mayor información al cliente la cual podría ser relevante para éste.

11. Bibliografía

[ICCA2012] International Congress and Convention Association disponible en : http://www.iccaworld.com/dcps/doc.cfm?docid=1520, 05 de Septiembre 2012.

[Ticketforevent2013] Ticket For Event disponible en: http://ticketforevent.com/, 01 de Mayo 2013.

[Eventbrite2013] Eventbrite disponible en: http://www.eventbrite.com/about/, 01 de Mayo 2013.

[TicketTailor2013] Ticket Tailor disponible en: http://www.tickettailor.com/ 01 de Mayo 2013.

[Eventioz2013] Eventios disponible en: https://eventioz.cl/info/about 01 de Mayo 2013.

[ExtremeProgramming2009] Extreme Programming a gentle introduction disponible en: http://www.extremeprogramming.org/, 28 de Septiembre de 2009.

[Sánchez2004] ONess: un proyecto OpenSource para el negocio textil mayorista desarrollado con tecnologías OpenSource innovadoras. Disponible en: http://oness.sourceforge.net/proyecto/html/index.html, 28 de Septiembre 2004.

[Dexter2012] Joomla! Programming. Mark Dexter. Edición Digital 2012.

[Meier2012] Professional Android 4 Application Development. Reto Meier. Edición Digital 2012.