

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

CENTRO UNIVERSITARIO UAEM TEXCOCO  
INGENIERÍA EN COMPUTACIÓN



Desarrollo de un sistema de información para el registro a  
eventos académicos universitarios

TESIS

Que para obtener el Título de  
Ingeniero en Computación

Presenta

Luis Adrián Morales Cruz

Director de tesis

M en I.S.C. Irene Aguilar Juárez

Revisores de tesis

Dr. en C. Alfonso Zarco Hidalgo

Dr. en Ed. Joel Ayala de la Vega

Texcoco, Estado de México, a 14 de marzo de 2016

## **RESUMEN**

El Centro Universitario UAEM Texcoco suele realizar eventos académicos dentro de sus instalaciones. Estos eventos incluyen comúnmente congresos, ponencias y talleres de actividades.

El proceso de inscripción a estos eventos es realizado manualmente por los profesores organizadores, pero no sólo el proceso de inscripción, también el registro de asistencia, entre otras actividades realizadas para obtener información de las actividades realizadas en el evento.

En la presente tesis se presenta una solución al proceso de inscripción y generación de documentación mediante un sistema de información para eventos académicos. En este sistema se podrá realizar preinscripción e inscripción de alumnos, registrar actividades, ponentes, asistencia, y generar la documentación que es requerida al finalizar el evento.

# ÍNDICE

Dedicatoria.....	3
Agradecimientos.....	4
Resumen.....	5
Índice.....	6
Índice de Imágenes.....	12
Índice de Tablas.....	17
I. Introducción.....	19
II. Problemática.....	21
III. Justificación.....	22
IV. Objetivo.....	23
<i>Capítulo 1</i> Ingeniería de Software.....	24
1.1 ¿Qué es el software?.....	24
1.2 ¿Qué es ingeniería del software?.....	24
1.2.1 Objetivo primario de la ingeniería del software.....	25
1.3 Metodología.....	26
1.3.1 Metodología de cascada.....	26
1.4 UML.....	29
1.4.1 ¿Qué es UML?.....	29
1.4.2 Modelo UML.....	29
<i>Capítulo 2</i> Bases de Datos.....	35
2.1 ¿Qué es una base de datos?.....	35

2.2	¿Qué es un Sistema Gestor de Bases de Datos?.....	36
2.2.1	Objetivo de los SGBD .....	36
2.2.2	Características de los SGBD.....	37
2.3	Arquitectura de tres esquemas.....	37
2.3.1	Nivel interno .....	38
2.3.2	Nivel conceptual .....	38
2.3.3	Nivel de vista o externo .....	38
2.4	Independencia de los datos.....	38
2.4.1	Independencia lógica de datos .....	39
2.4.2	Independencia física de datos .....	39
2.5	Modelo Entidad-Relación (ER).....	39
2.5.1	Entidades y atributos.....	40
2.5.2	Relaciones.....	44
2.5.3	Símbolos del modelo ER .....	49
2.6	Modelo relacional.....	50
2.6.1	¿Qué es el modelo relacional? .....	50
2.6.2	Conceptos.....	51
2.6.3	Características.....	52
2.6.4	Restricciones del modelo relacional .....	53
2.7	Transformación del modelo ER al modelo relacional.....	55
2.7.1	Algoritmo de mapeado ER a relacional .....	55
<i>Capítulo 3</i> Sistemas de Información.....		58
3.1	Definición de sistemas de información .....	58
3.2	Elementos de los sistemas de información.....	58
3.3	Ejemplos de sistemas de información .....	59

3.4	Actividades de los Sistemas de Información.....	59
3.4.1	Entrada.....	59
3.4.2	Almacenamiento de información.....	60
3.4.3	Procesamiento.....	60
3.4.4	Salida.....	60
3.5	Beneficios de los Sistemas de Información.....	60
<i>Capítulo 4</i> Tecnologías a implementar.....		63
4.1	Java.....	63
4.1.1	Historia de Java.....	63
4.1.2	¿Qué es Java?.....	65
4.1.3	Características.....	66
4.1.4	Ventajas.....	67
4.2	NeatBeans.....	68
4.2.1	¿Qué es NeatBeans?.....	68
4.2.2	Historia.....	68
4.2.3	Características.....	69
4.3	MySQL.....	70
4.3.1	¿Qué es MySQL?.....	70
4.3.2	Características de MySQL.....	71
4.3.3	Ventajas.....	71
4.3.4	Desventajas.....	72
4.3.5	Workbench.....	73
<i>Capítulo 5</i> Eventos Académicos.....		76
5.1	¿Qué es un evento?.....	76
5.2	Tipo de eventos.....	76

5.2.1	Congreso .....	76
5.2.2	Ponencia.....	78
5.2.3	Seminario .....	80
5.2.4	Taller.....	80
<i>Capítulo 6</i> Desarrollo del Sistema .....		82
6.1	Análisis de requerimientos .....	82
6.1.1	Requerimientos funcionales.....	82
6.1.2	Requerimientos no funcionales.....	83
6.1.3	Identificación de requerimientos.....	83
6.2	Requerimientos del Sistema de Inscripción .....	84
6.2.1	Requerimientos Funcionales.....	85
6.2.2	Requerimientos no funcionales.....	85
6.3	Diagramas de casos de uso.....	86
6.3.1	Diagrama de caso de uso Inicializa evento .....	86
6.3.2	Diagrama de caso de uso Registra actividades .....	86
6.3.3	Diagrama de caso de uso Registra datos de ponente .....	87
6.3.4	Diagrama de caso de uso Realiza preinscripción.....	88
6.3.5	Diagrama de caso de uso Completa inscripción de alumno .....	89
6.3.6	Diagrama de caso de uso Registra asistencia.....	90
6.3.7	Diagrama de caso de uso Genera documentación.....	91
6.4	Diagramas de clases .....	92
6.4.1	Diagrama de clase Actividad .....	92
6.4.2	Diagrama de clase Administrador.....	93
6.4.3	Diagrama de clase Alumno.....	95
6.4.4	Diagrama de clase Paquete .....	96

6.4.5	Diagrama de clase Ponente .....	98
6.4.6	Diagrama de clase Agradecimiento .....	100
6.4.7	Diagrama de clase ConstAlumno.....	100
6.4.8	Diagrama de clase ReporteAsis .....	101
6.4.9	Diagrama de clase PrintMe.....	101
6.4.10	Diagrama de clase Portada.....	102
6.4.11	Diagrama de clase Inicio.....	102
6.4.12	Diagrama de clase Preins .....	103
6.4.13	Diagrama de clase InformAct .....	104
6.4.14	Diagrama de clase FrmInsAlumno .....	105
6.4.15	Diagrama de clase FrCod.....	106
6.4.16	Diagrama de clase RegisPon.....	107
6.4.17	Diagrama de clase ActivaPaquete.....	108
6.4.18	Diagrama de clase RegAct.....	108
6.4.19	Diagrama de clase Asis.....	109
6.4.20	Diagrama de clase AsisPon.....	111
6.4.21	Diagrama de clase Constancias.....	112
6.4.22	Diagrama de clase AlumConst.....	113
6.4.23	Diagrama de clase PonAgr.....	114
6.4.24	Diagrama de clase Asistencia .....	115
6.5	Diagrama Entidad-Relación .....	117
6.6	Modelo Relacional .....	118
6.7	Diagrama del modelo relacional .....	120
<i>Capítulo 7</i> Resultados .....		121
7.1	Resultados del Sistema.....	121

7.1.1	Preinscripción de alumnos .....	122
7.1.2	Inscripción de alumnos .....	123
7.1.3	Registro de ponente.....	125
7.1.4	Activar paquete .....	126
7.1.5	Registro de actividades .....	127
7.1.6	Asistencia de alumnos.....	128
7.1.7	Asistencia de ponente .....	129
7.1.8	Generar documentación .....	130
Conclusiones .....		136
Bibliografía .....		138
<i>Anexo A</i> Manual de Usuario .....		140
A.1	Realizar preinscripción de alumnos .....	141
A.2	Ingreso l sistema como administrador.....	145
A.3	Completar inscripción de alumnos.....	147
A.4	Registro de ponentes .....	150
A.5	Activar paquete .....	151
A.6	Registro de actividades.....	152
A.7	Asistencia de alumnos.....	153
A.8	Asistencia de ponente.....	154
A.9	Generar constancia .....	155
A.9.1	Constancia de asistencia de alumno.....	156
A.9.2	Agradecimiento ponente .....	157
A.9.3	Lista de asistencia .....	158

# ÍNDICE DE IMÁGENES

Imagen 1.	Componentes del software.....	24
Imagen 2.	Arquitectura von Neumann.....	25
Imagen 3.	Modelo de cascada.....	27
Imagen 4.	Ciclo de vida en cascada.....	27
Imagen 5.	Diagrama de casos de uso.....	30
Imagen 6.	Diagrama de clases pedido.....	33
Imagen 7.	Arquitectura de tres esquemas.....	33
Imagen 8.	Ejemplo Modelo ER.....	40
Imagen 9.	Jerarquía de atributos compuestos.....	42
Imagen 10.	Ejemplo de relación binaria.....	45
Imagen 11.	Ejemplo de relación ternaria.....	45
Imagen 12.	Ejemplo de cardinalidad 1:1.....	47
Imagen 13.	Ejemplo de cardinalidad N:M.....	48
Imagen 14.	Ejemplo de modelo ER con razón de cardinalidad.....	48
Imagen 15.	Símbolos del modelo ER.....	50
Imagen 16.	Atributos y tuplas de una relación estudiante.....	52
Imagen 17.	Logo Java.....	68
Imagen 18.	Logo de NeatBeans.....	70
Imagen 19.	Logo de MySQL.....	72

Imagen 20.	Logo de WorkBench.....	73
Imagen 21.	Diagrama de caso de uso Inicializa evento.....	86
Imagen 22.	Diagrama de caso de uso Registra actividades.....	86
Imagen 23.	Diagrama de caso de uso Registra datos de ponente.....	87
Imagen 24.	Diagrama de caso de uso Realiza preinscripción.....	88
Imagen 25.	Diagrama de caso de uso Completa inscripción de alumno.....	89
Imagen 26.	Diagrama de caso de uso Registra asistencia.....	90
Imagen 27.	Diagrama de caso de uso Genera documentación.....	91
Imagen 28.	Diagrama de clase Actividad.....	92
Imagen 29.	Diagrama de clase Administrador.....	93
Imagen 30.	Diagrama de clase Alumno.....	95
Imagen 31.	Diagrama de clase Paquete.....	96
Imagen 32.	Diagrama de clase Ponente.....	98
Imagen 33.	Diagrama de clase Agradecimiento.....	100
Imagen 34.	Diagrama de clase ConstAlumno.....	100
Imagen 35.	Diagrama de clase ReporteAsis.....	101
Imagen 36.	Diagrama de clase PrintMe.....	101
Imagen 37.	Diagrama de clase Portada.....	102
Imagen 38.	Diagrama de clase Inicio.....	102
Imagen 39.	Diagrama de clase Preins.....	103
Imagen 40.	Diagrama de clase InformAct.....	104
Imagen 41.	Diagrama de clase FrmInsAlumno.....	105
Imagen 42.	Diagrama de clase FrCod.....	106

Imagen 43.	Diagrama de clase RegisPon.....	107
Imagen 44.	Diagrama de clase ActivaPaquete.....	108
Imagen 45.	Diagrama de clase RegAct.....	108
Imagen 46.	Diagrama de clase Asis.....	109
Imagen 47.	Diagrama de clase AsisPon.....	111
Imagen 48.	Diagrama de clase Constancias.....	112
Imagen 49.	Diagrama de clase AlumConst.....	113
Imagen 50.	Diagrama de clase PonAgr.....	114
Imagen 51.	Diagrama de clase Asistencia.....	115
Imagen 52.	Modelo Entidad-Relación.....	117
Imagen 53.	Diagrama del modelo relacional.....	120
Imagen 54.	Interfaz de preinscripción de alumnos.....	122
Imagen 55.	Interfaz de consulta de actividades.....	123
Imagen 56.	Tabla "alumno".....	123
Imagen 57.	Interfaz inscripción de alumnos.....	124
Imagen 58.	Gafete con código de barras.....	124
Imagen 59.	Interfaz de registro de ponentes.....	125
Imagen 60.	Tabla "ponente".....	125
Imagen 61.	Interfaz activar paquete.....	126
Imagen 62.	Tabla "paquete".....	126
Imagen 63.	Interfaz registro de actividades.....	127
Imagen 64.	Tabla "actividad".....	127
Imagen 65.	Interfaz asistencia de alumnos.....	128

Imagen 66.	Tabla "asistencia".....	128
Imagen 67.	Interfaz asistencia de ponente.....	129
Imagen 68.	Tabla "asistencia_p".....	129
Imagen 69.	Interfaz de botones para generar documentación.....	130
Imagen 70.	Interfaz de constancia de asistencia de alumno.....	131
Imagen 71.	Tabla "constancia".....	131
Imagen 72.	Constancia de asistencia.....	132
Imagen 73.	Interfaz de agradecimiento de ponente.....	133
Imagen 74.	Tabla "agradecimiento".....	133
Imagen 75.	Agradecimiento de ponente.....	134
Imagen 76.	Interfaz de listas de asistencia.....	134
Imagen 77.	Tabla "lista_asis".....	135
Imagen 78.	Lista de asistencia.....	135
Imagen 79.	Ventana principal del sistema.....	140
Imagen 80.	Acceso al sistema como alumno.....	141
Imagen 81.	Selección del botón de preinscripción de alumnos.....	141
Imagen 82.	Mensaje de conexión exitosa para preinscripción.....	141
Imagen 83.	Ventana de preinscripción.....	142
Imagen 84.	Ventana de consulta de actividades.....	143
Imagen 85.	Consulta de actividades.....	143
Imagen 86.	Registro de datos para preinscripción.....	144
Imagen 87.	Mensaje de registro de datos exitoso.....	145
Imagen 88.	Acceso al sistema como administrador.....	145

Imagen 89.	Introducir clave.....	146
Imagen 90.	Mensaje de verificación de privilegios.....	146
Imagen 91.	Botones de administrador activados.....	147
Imagen 92.	Ventana inscripción de alumnos.....	147
Imagen 93.	Mensaje de error “No hay datos”.....	148
Imagen 94.	Consulta de inscripción de alumnos.....	148
Imagen 95.	Gafete de inscripción.....	149
Imagen 96.	Gafete de inscripción con código de barras.....	149
Imagen 97.	Ventana de registro de datos de ponentes.....	150
Imagen 98.	Registro de ponente.....	151
Imagen 99.	Ventana de activar paquete.....	151
Imagen 100.	Ventana de registro de actividades.....	152
Imagen 101.	Ventana de asistencia de alumnos.....	153
Imagen 102.	Ventana de asistencia de ponente.....	154
Imagen 103.	Ventana de generar constancias.....	155
Imagen 104.	Ventana de generar constancia de alumno.....	156
Imagen 105.	Ventana de agradecimiento de ponente.....	157
Imagen 106.	Ventana de lista de asistencia.....	158

# ÍNDICE DE TABLAS

Tabla 1.	Tabla de terminología de diagramas de clase .....	34
Tabla 2.	Evolución de Java .....	65
Tabla 3.	Diagrama de caso de uso Inicializa evento .....	86
Tabla 4.	Diagrama de caso de uso Registra actividades .....	87
Tabla 5.	Diagrama de caso de uso Registra datos de ponente.....	87
Tabla 6.	Diagrama de caso de uso Realiza preinscripción.....	88
Tabla 7.	Diagrama de caso de uso Completa inscripción de alumno.....	89
Tabla 8.	Diagrama de caso de uso Registra asistencia.....	90
Tabla 9.	Diagrama de caso de uso Genera documentación.....	91
Tabla 10.	Diagrama de clase Actividad.....	92
Tabla 11.	Diagrama de clase Administrador.....	93
Tabla 12.	Diagrama de clase Alumno.....	95
Tabla 13.	Diagrama de clase Paquete.....	97
Tabla 14.	Diagrama de clase Ponente.....	98
Tabla 15.	Diagrama de clase Agradecimiento.....	100
Tabla 16.	Diagrama de clase ConstAlumno.....	100
Tabla 17.	Diagrama de clase ReporteAsis.....	101
Tabla 18.	Diagrama de clase PrintMe.....	101
Tabla 19.	Diagrama de clase Portada.....	102

Tabla 20.	Diagrama de clase Inicio.....	102
Tabla 21.	Diagrama de clase Preins.....	103
Tabla 22.	Diagrama de clase InformAct.....	104
Tabla 23.	Diagrama de clase FrmInsAlumno.....	105
Tabla 24.	Diagrama de clase FrCod.....	106
Tabla 25.	Diagrama de clase RegisPon.....	107
Tabla 26.	Diagrama de clase ActivaPaquete.....	108
Tabla 27.	Diagrama de clase RegAct.....	109
Tabla 28.	Diagrama de clase Asis.....	110
Tabla 29.	Diagrama de clase AsisPon.....	111
Tabla 30.	Diagrama de clase Constancias.....	112
Tabla 31.	Diagrama de clase AlumConst.....	113
Tabla 32.	Diagrama de clase PonAgr.....	114
Tabla 33.	Diagrama de clase Asistencia.....	116

# I. INTRODUCCIÓN

Los eventos culturales o académicos en la vida estudiantil de los alumnos son importantes porque tratan temas actuales y tienen una manera de participación diferente a las clases en las escuelas.

Dentro de los congresos se imparten ponencias, seminarios y talleres de actividades, éstos son de gran utilidad, pues permiten estar actualizados en los temas que nos competen, ampliar los conocimientos de manera general o especializada, y serán de ayuda para el desarrollo de actividades profesionales.

Los conocimientos adquiridos pueden ser compartidos en la escuela, en el trabajo o en la vida personal; así se puede dar una mejor imagen de uno mismo y ser mejor reconocido.

Es importante aprovechar estos eventos para crear relaciones sociales con los demás asistentes o ponentes. El crear contactos profesionales puede ser un gran impulso para cuando quieras desarrollar cualquier proyecto. Ser parte de una comunidad con la cual compartir información y siempre estar al tanto del tema son puntos de los cuales vas a obtener gran ventaja tanto académica como profesional. Como universitario se puede informar sobre las actividades que quieren emprender, en qué área les gustaría desarrollarse profesionalmente y en qué empresas podrían trabajar.

En el Centro Universitario UAEM Texcoco se realiza la inscripción manual a estos eventos. Ésta consiste en realizar el registro con la persona asignada, dar los datos que son requeridos para la inscripción, los cuales se anotan en una hoja de registro, para finalmente, realizar el pago del evento.

La inscripción manual a los eventos es delicada y sujeta a riesgos de pérdida de información pues los datos de la persona que se registra pueden ser anotados erróneamente o extraviar la información. Esta situación puede impedir que se genere la documentación necesaria de forma adecuada. El proceso de inscripción también se complica debido a que debe realizarse un pago para dicho trámite.

Muchas veces los usuarios tienen desconfianza del registro, pues el mal manejo de la información genera cierta incertidumbre sobre la conclusión correcta del trámite, quedan dudas sobre si se garantiza su puesto dentro del evento y si se entregará la documentación en tiempo y forma.

En esta investigación se analizará la información requerida para realizar un sistema de información que permita minimizar los riesgos de este proceso de inscripción, como son los requerimientos para dicho sistema, cómo manejar el registro de asistencias, y cómo generar la documentación de la manera más apropiada.

## II. PROBLEMÁTICA

Los eventos culturales o académicos en la vida estudiantil de los alumnos son importantes porque pueden asistir a diversos talleres para desarrollar varias actividades, o asistir a diferentes congresos donde pueden informarse de proyectos novedosos que se están llevando a cabo para resolver alguna problemática.

El asistir a los talleres y desarrollar alguna actividad es importante para los alumnos, ya que puede valer como experiencia y servir a futuro para ellos.

La inscripción manual consiste en realizar el registro con la persona asignada, dar los datos que son requeridos para la inscripción, los cuales son anotados en una hoja de registro, y finalmente, realizar el pago del evento.

La inscripción manual a los eventos dentro del centro universitario UAEM Texcoco es delicada y sujeta a riesgos de pérdida de información pues los datos de la persona que se registra pueden ser anotados erróneamente.

Esta situación impide que se genere la documentación necesaria de forma adecuada.

El proceso de inscripción se complica debido a que debe realizarse pago para dicho trámite.

Muchas veces los usuarios tienen desconfianza del registro, pues el mal manejo de la información genera cierta incertidumbre sobre la conclusión correcta del trámite, quedan dudas sobre si se garantiza su puesto dentro del evento y si se entregará la documentación en tiempo y forma.

### **III. JUSTIFICACIÓN**

Los beneficios que el desarrollo de este sistema de información ofrece son diversos, uno de ellos es facilitar el registro a eventos, recopilando información de las personas que se registran, almacenar la información como el nombre, número de cuenta, si realizó pago o no dentro de una base de datos.

Un beneficio para el administrador es que hay menos riesgo de pérdida de información de los clientes, se realizan las consultas necesarias sobre los eventos y se genera la documentación requerida en forma apropiada.

Realizar un sistema de información genera mayor confianza en los usuarios, pues existe mayor garantía de que su trámite se haya realizado de buena forma y que sus datos se hayan registrado correctamente

El sistema de información beneficia tanto a los administradores, como a los clientes, permitiendo una mayor confianza hacia la captura de datos, ya que el almacenamiento de datos hará que la información sea más confiable, integra y segura con respecto a la inscripción manual.

#### **Hipótesis o supuesto:**

El sistema de información nos permitirá generar documentación con mayor confiabilidad, integridad, seguridad y eficiencia con respecto al sistema de registro manual que es utilizado.

# **IV. OBJETIVO**

## **Objetivo General**

Desarrollar un sistema de información que permita el registro a los eventos académicos universitarios (congresos), por ejemplo, ponencias, foros, talleres, seminarios en el Centro Universitario UAEM Texcoco y que genere la documentación, como son, constancias, agradecimientos y listas de asistencia.

## **Objetivos Particulares**

1. Identificar los procesos involucrados en los eventos universitarios.
2. Identificar las características de la documentación a generar.
3. Diseño del sistema en base a los requerimientos.
4. Diseño de la base de datos en base a los requerimientos.

# Capítulo 1 INGENIERÍA DE SOFTWARE

## 1.1 ¿Qué es el software?

Según el Webster's New Collegiate Dictionary (1975), "software es un conjunto de programas, procedimientos y documentación relacionada asociados con un sistema, especialmente un sistema informático" (Laboratorio Nacional de Calidad del Software, 2009).

El software se puede definir como el conjunto de tres componentes (véase Imagen 1):

- Programas (instrucciones): este componente proporciona la funcionalidad deseada y el rendimiento cuando se ejecute.
- Datos: este componente incluye los datos necesarios para manejar y probar los programas y las estructuras requeridas para mantener y manipular estos datos.
- Documentos: este componente describe la operación y uso del programa.



Imagen 1. Componentes del software (Laboratorio Nacional de Calidad del Software, 2009)

## 1.2 ¿Qué es ingeniería del software?

El término ingeniería del software apareció por primera vez en la conferencia de ingeniería de software de la OTAN en 1968 y fue mencionado para provocar el pensamiento sobre la crisis de software del momento. Desde entonces, ha continuado como una profesión y campo de estudio dedicado a la creación de software de alta calidad, barato, con capacidad de mantenimiento y rápido de construir. Mientras que el término ingeniería del software fue acuñado en una conferencia en 1968, los problemas que intentaba tratar empezaron mucho antes. La historia de la ingeniería del software está entrelazada con las historias contrapuestas de hardware y software. (Laboratorio Nacional de Calidad del Software, 2009).

Cuando el ordenador digital moderno apareció por primera vez en 1941, las instrucciones para hacerlo funcionar estaban conectadas dentro de la máquina. Las personas relacionadas con la ingeniería rápidamente se dieron cuenta de que este diseño no era flexible e idearon la arquitectura de programa almacenado o arquitectura von Neumann (véase Imagen 2) (Laboratorio Nacional de Calidad del Software, 2009).

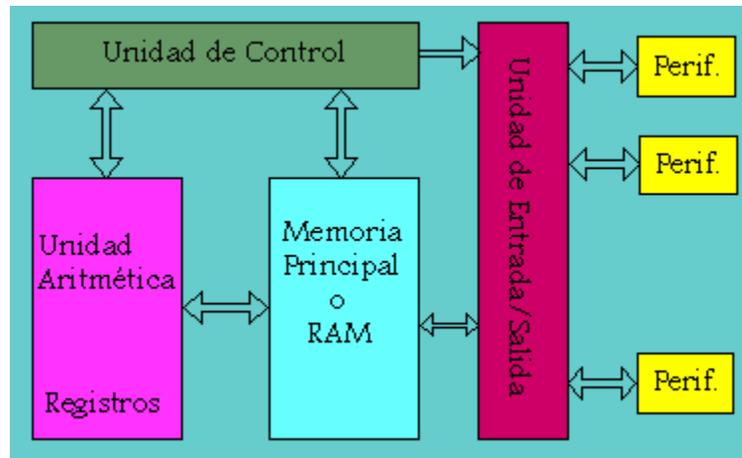


Imagen 2. Arquitectura von Neumann

### 1.2.1 Objetivo primario de la ingeniería del software

El objetivo primario de la ingeniería del software es construir un producto de alta calidad de una manera oportuna. Trata de conseguir este objetivo primario usando un enfoque de ingeniería. Para conseguir el objetivo de construir productos de alta calidad dentro de la planificación, la ingeniería del software emplea una serie de prácticas para:

- Entender el problema
- Diseñar una solución
- Implementar la solución correctamente
- Probar la solución
- Gestionar las actividades anteriores para conseguir alta calidad

La ingeniería del software representa un proceso formal que incorpora una serie de métodos bien definidos para el análisis, diseño, implementación y pruebas del software y sistemas. Además, abarca una amplia colección de métodos y técnicas de gestión de proyectos para el aseguramiento

de la calidad y la gestión de la configuración del software (Laboratorio Nacional de Calidad del Software, 2009).

### 1.3 Metodología

La ingeniería del software establece y se vale de una serie de modelos que establecen y muestran las distintas etapas y estados por los que pasa un producto software, desde su concepción inicial, pasando por su desarrollo, puesta en marcha y posterior mantenimiento, hasta la retirada del producto. A estos modelos se les denomina “Modelos de ciclo de vida del software” (Laboratorio Nacional de Calidad del Software, 2009).

Un modelo de ciclo de vida de software es una vista de las actividades que ocurren durante el desarrollo de software, intenta determinar el orden de las etapas involucradas y los criterios de transición asociados entre estas etapas (Laboratorio Nacional de Calidad del Software, 2009).

Las actividades de los modelos de ciclo de vida son las siguientes:

- Describe las fases principales de desarrollo de software
- Define las fases primarias esperadas de ser ejecutadas durante esas fases
- Ayuda a administrar el progreso del desarrollo
- Provee un espacio de trabajo para la definición de un proceso detallado de desarrollo de software

El sistema de información será realizado mediante el modelo de cascada.

#### **1.3.1 Metodología de cascada**

En Ingeniería de software el desarrollo en cascada, también llamado modelo en cascada (véase Imagen 3), es el enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior (Blogspot, 2013).

# Modelo Cascada

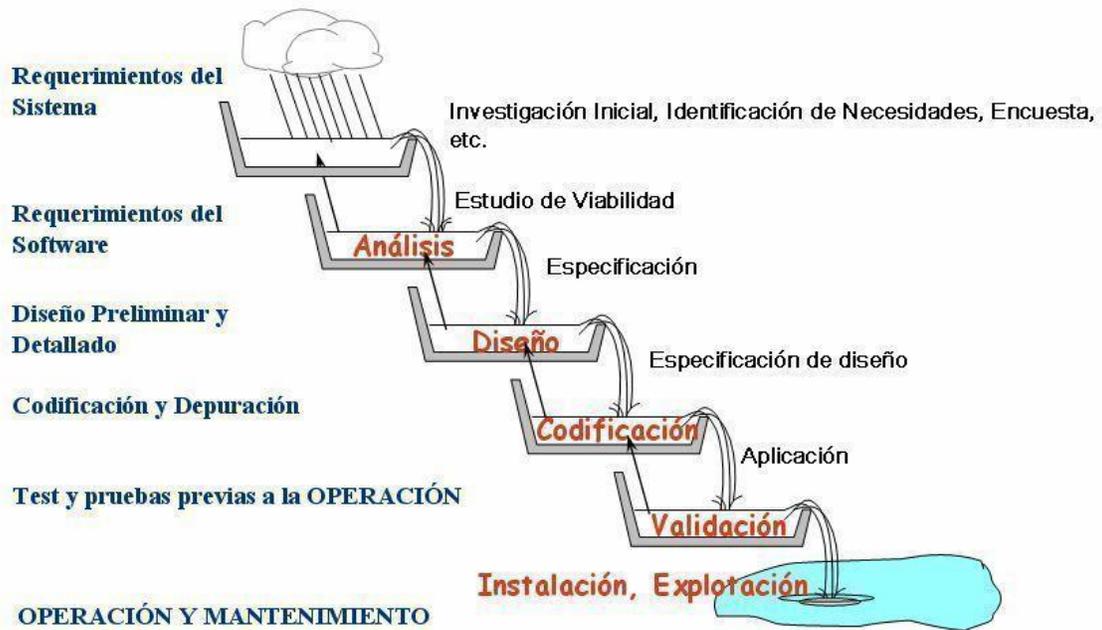


Imagen 3. Modelo de cascada

La Imagen 4 muestra el ciclo de vida en cascada.

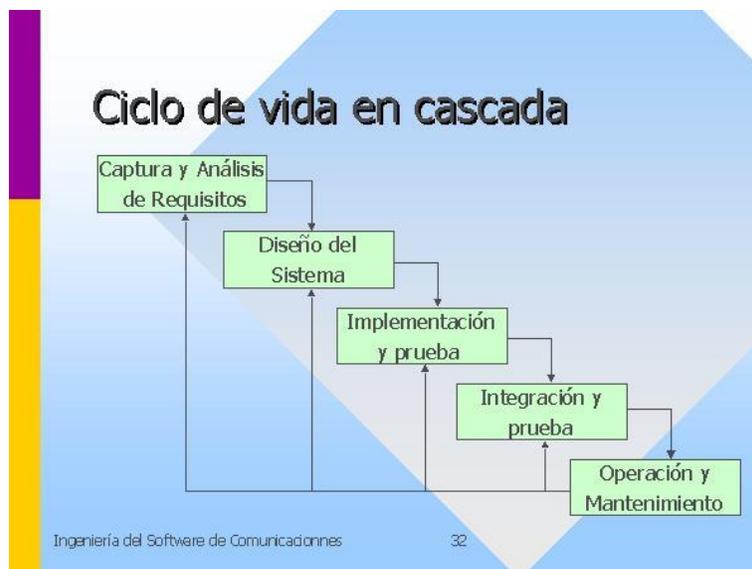


Imagen 4. Ciclo de vida en cascada

### *1.3.1.1 Fases del modelo de cascada*

- Análisis de requisitos

En esta fase se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. De esta fase surge una memoria llamada SRD (Documento de Especificación de Requisitos), que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos. Es importante señalar que en esta etapa se debe consensuar todo lo que se requiere del sistema y será aquello lo que seguirá en las siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software de una manera.

- Diseño del Sistema

Descompone y organiza el sistema en elementos que puedan elaborarse por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el SDD (Documento de Diseño del Software), que contiene la descripción de la estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras.

- Diseño del Programa

Es la fase en donde se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario así como también los análisis necesarios para saber qué herramientas usar en la etapa de codificación.

- Codificación

Es la fase en donde se implementa el código fuente, haciendo uso de prototipos así como de pruebas y ensayos para corregir errores. Dependiendo del lenguaje de programación y su versión se crean las bibliotecas y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.

- Pruebas

Los elementos, ya programados, se ensamblan para componer el sistema y se comprueba que funciona correctamente y que cumple con los requisitos, antes de ser entregado al usuario final.

- Verificación

Es la fase en donde el usuario final ejecuta el sistema, para ello el o los programadores ya realizaron exhaustivas pruebas para comprobar que el sistema no falle.

En la creación de desarrollo de cascada se implementa los códigos de investigación y pruebas del mismo.

- Mantenimiento

Una de las etapas más críticas, ya que se destina un 75% de los recursos, es el mantenimiento del Software ya que al utilizarlo como usuario final puede ser que no cumpla con todas nuestras expectativas.

## 1.4 UML

### 1.4.1 ¿Qué es UML?

El lenguaje unificado de modelado o UML (Unified Modeling Language) es el sucesor de la oleada de métodos de análisis y diseño orientados a objetos que surgió a finales de la década de 1980 y principios de la siguiente (Fowler & Scott, 1999).

El UML es un lenguaje de modelado, y no un método. La mayor parte de los métodos consisten, al menos en principio, en un lenguaje y en un proceso para modelar. El lenguaje de modelado es la notación (principalmente gráfica) de que se valen los métodos para expresar los diseños. El proceso es la orientación que nos dan sobre los pasos a seguir para hacer el diseño (Fowler & Scott, 1999).

### 1.4.2 Modelo UML

El modelo UML es un conjunto de diagramas que definen los procesos de una interfaz. UML maneja diferentes diagramas, a continuación se mencionan algunos:

- Diagrama de casos de uso
- Diagrama de clases
- Diagrama de paquetes
- Diagrama de actividades

- Diagrama de emplazamiento

Para este proyecto solo manejaremos dos diagramas, el diagrama de casos de uso y el diagrama de clases. Éstos se explican a continuación:

### 1.4.2.1 Diagrama de casos de uso

Jacobson (1994), además de introducir los casos de uso como elementos primarios del desarrollo de software, también diseñó un diagrama para la representación gráfica de los casos de uso (véase Imagen 5). El diagrama de casos de uso es ya también parte del UML (Fowler & Scott, 1999).

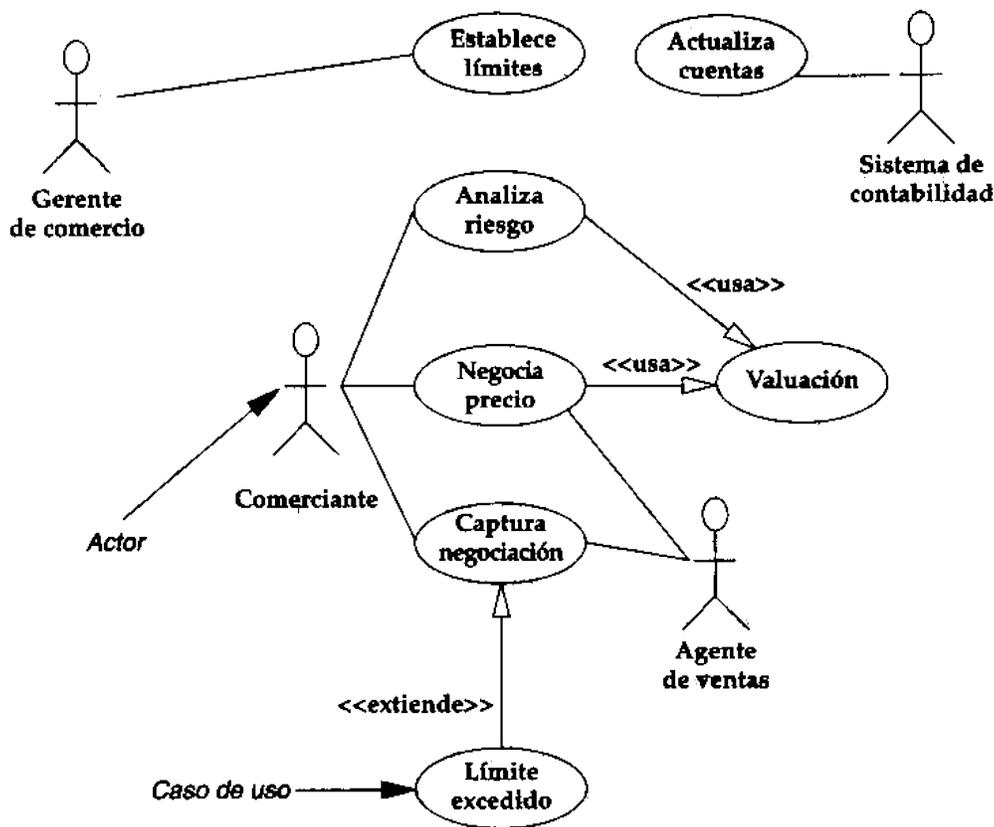


Imagen 5. Diagrama de casos de uso (Fowler & Scott, 1999)

#### 1.4.2.1.1 Actores

Empleamos el término actor para llamar así al usuario, cuando desempeña ese papel con respecto al sistema. Hay cuatro actores en la Imagen 5: el gerente de comercio, el comerciante, el agente de ventas y el sistema de contabilidad. Un usuario puede desempeñar varios papeles. Por ejemplo, un

comerciante de edad madura podría desempeñar el papel de gerente de comercio y además ser un comerciante normal. Los actores llevan a cabo casos de uso. Un mismo actor puede realizar muchos casos de uso; a la inversa, un caso de uso puede ser realizado por varios actores (Fowler & Scott, 1999).

#### *1.4.2.1.2 Los casos de uso*

Los casos de uso realizan las siguientes funciones:

- El caso de uso capta alguna función visible para el usuario.
- El caso de uso puede ser pequeño o grande.
- El caso de uso logra un objetivo discreto para el usuario.

En su forma más simple, el caso de uso se obtiene hablando con los usuarios habituales y analizando con ellos las distintas cosas que deseen hacer con el sistema. Se debe abordar cada cosa discreta que quieran, darle un nombre y escribir un texto descriptivo breve (Fowler & Scott, 1999).

#### *1.4.2.1.3 Uses y Extends*

Se utilizan las relaciones uses (usa) y extends (extiende) como se muestra en la Imagen 5. Se usa la relación extends cuando se tiene un caso de uso que es similar a otro, pero que hace un poco más. Las relaciones uses ocurren cuando se tiene una porción de comportamiento que es similar en más de un caso de uso y no se quiere copiar la descripción de tal conducta (Fowler & Scott, 1999).

Aplique las siguientes reglas:

- Utilice extends cuando describa una variación de conducta normal.
- Emplee uses para repetir cuando se trate de uno o varios casos de uso y desee evitar repeticiones.

#### *1.4.2.2 Diagrama de clases*

El diagrama de clases describe los tipos de objetos que hay en el sistema y las diversas clases de relaciones estáticas que existen entre ellos. Hay dos tipos principales de relaciones estáticas:

- Asociaciones (por ejemplo, un cliente puede rentar diversas videocintas).
- Subtipos (una enfermera es un tipo de persona).

Los diagramas de clase también muestran los atributos y operaciones de una clase y las restricciones a que se ven sujetos, según la forma en que se conecten los objetos (Fowler & Scott, 1999).

A continuación se muestra la estructura de una clase y sus características:

#### Clase

Nombre Clase
Atributos
Operaciones o Métodos

Nombre Clase: Nombre de la clase a desarrollar

Atributos: son las variables de la clase, éstas pueden ser públicas, privadas o protegidas (Representa el estado que un objeto va a tener).

Operaciones o Métodos: son los procesos que una clase sabe llevar a cabo (Representa las posibles acciones que puede tener un objeto).

En la Imagen 6 se muestra un ejemplo de diagrama de clase para realizar pedidos.

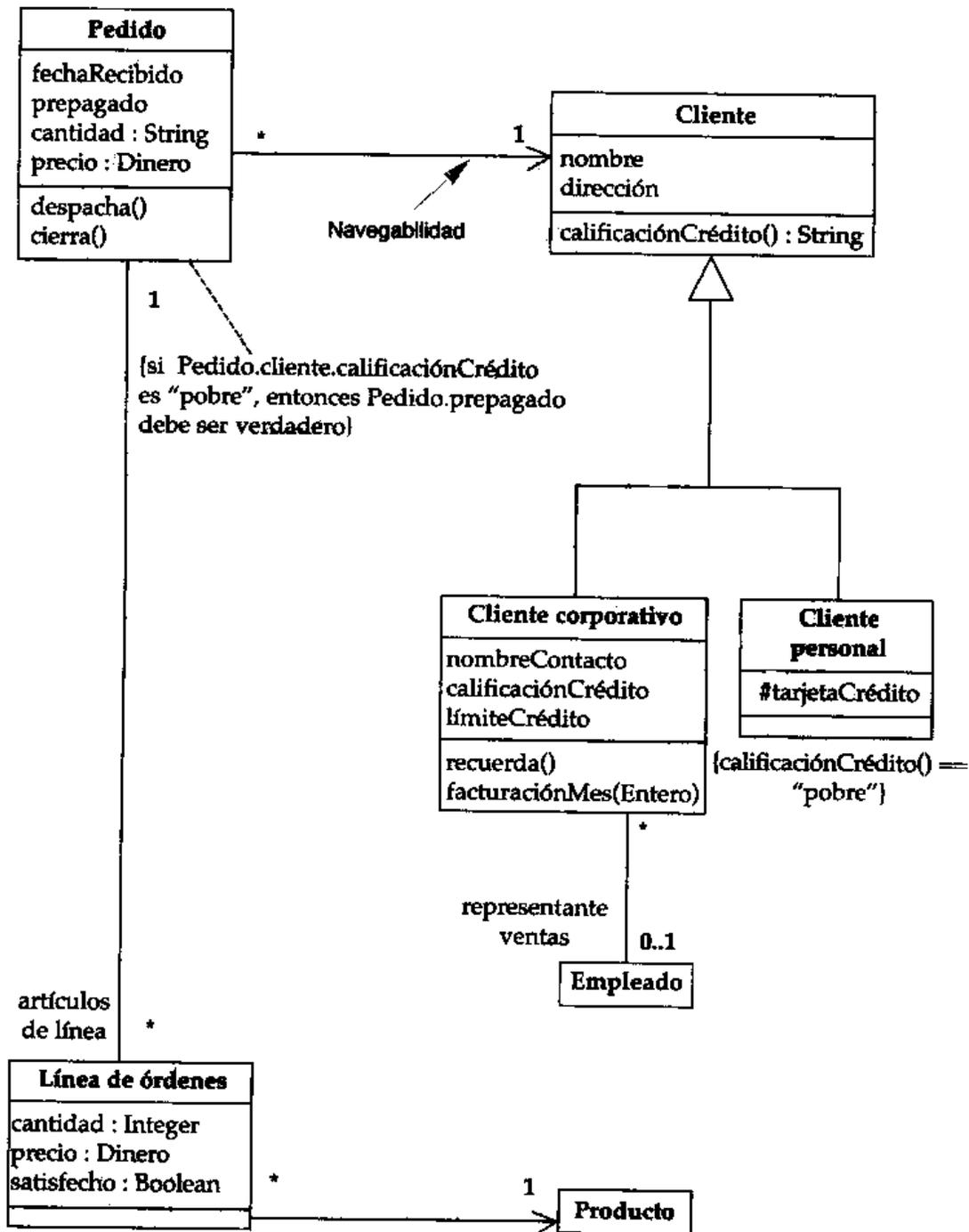


Imagen 6. Diagrama de clases Pedido (Fowler & Scott, 1999)

Los diversos métodos orientados a objetos (OO) utilizan terminologías diferentes para estos conceptos. Se trata de algo sumamente frustrante pero inevitable, dado que los lenguajes OO son tan desconsiderados como los métodos. Es en esta área que el UML aportará algunos de sus mayores beneficios, al simplificar estos diferentes diagramas (Fowler & Scott, 1999).

En la Tabla 1 se muestra la terminología de diagramas de clase:

*Tabla 1. Tabla de terminología de diagramas de clase (Fowler & Scott, 1999)*

<b>UML</b>	<b>Clase</b>	<b>Asociación</b>	<b>Generalización</b>	<b>Agregación</b>
<b>Booch</b>	Clase	Usa	Hereda	Contiene
<b>Coad</b>	Clase y objeto	Conexión de instancias	Espec-gen	Parte-todo
<b>Jacobson</b>	Objeto	Asociación por reconocimiento	Hereda	Consiste en
<b>Odell</b>	Tipo de objeto	Relación	Subtipo	Composición
<b>Rumbaugh</b>	Clase	Asociación	Generalización	Agregación
<b>Shlaer/Mellor</b>	Objeto	Relación	Subtipo	N/A

# Capítulo 2 BASES DE DATOS

## 2.1 ¿Qué es una base de datos?

Una base de datos es una colección de datos relacionados. Con la palabra datos nos referimos a los hechos (datos) conocidos que se pueden grabar y que tienen un significado implícito. Por ejemplo, piense en los nombres, números de teléfono y direcciones de las personas que conoce. Puede tener todos estos datos grabados en un libro de direcciones indexado o los puede tener almacenados en el disco duro de un computador mediante una aplicación como Microsoft Access o Excel. Esta colección de datos relacionados con un significado implícito es una base de datos.

La definición anterior de base de datos es muy genérica; por ejemplo, podemos pensar que la colección de palabras que compone esta página de texto es una colección de datos relacionados y que, por tanto, constituye una base de datos. No obstante, el uso común del término base de datos es normalmente más restringido. Una base de datos tiene las siguientes propiedades implícitas:

- Una base de datos representa algún aspecto del mundo real, lo que en ocasiones se denomina mini mundo o universo de discurso (UoD, Universe of discourse). Los cambios introducidos en el mini mundo se reflejan en la base de datos.
- Una base de datos es una colección de datos lógicamente coherente con algún tipo de significado inherente. No es correcto denominar base de datos a un surtido aleatorio de datos.
- Una base de datos se diseña, construye y rellena con datos para un propósito específico. Dispone de un grupo pretendido de usuarios y algunas aplicaciones preconcebidas en las que esos usuarios están interesados.

En otras palabras, una base de datos tiene algún origen del que se derivan los datos, algún grado de interacción con eventos del mundo real y un público que está activamente interesado en su contenido. Los usuarios finales de una base de datos pueden efectuar transacciones comerciales,

por ejemplo, un cliente que compra una cámara, o se pueden producir unos eventos, por ejemplo, un empleado tiene un hijo, que provoquen un cambio en la información almacenada en la base de datos. Al objeto de que una base de datos sea en todo momento precisa y fiable, debe ser un reflejo exacto del mini mundo que representa; por consiguiente, en la base de datos deben reflejarse los cambios tan pronto como sea posible (Elmasri & Navathe, 2007).

## 2.2 ¿Qué es un Sistema Gestor de Bases de Datos?

Un sistema gestor de bases de datos (DBMS, database management system) es una colección de programas que permite a los usuarios crear y mantener una base de datos. El DBMS es un sistema de software de propósito general que facilita los procesos de definición, construcción, manipulación y compartición de bases de datos entre varios usuarios y aplicaciones (Elmasri & Navathe, 2007).

Se trata de un conjunto de programas no visibles al usuario final que se encargan de la privacidad, la integridad, la seguridad de los datos y la interacción con el sistema operativo. Proporciona una interfaz entre los datos, los programas que los manejan y los usuarios finales. Cualquier operación que el usuario hace contra la base de datos está controlada por el gestor (Alvarez, 2007).

Ayuda a realizar las siguientes acciones:

- Definición de los datos
- Mantenimiento de la integridad de los datos dentro de la base de datos
- Control de la seguridad y privacidad de los datos
- Manipulación de los datos

### 2.2.1 Objetivo de los SGBD

El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado (Bravo, 2006).

Los SGDB brindan facilidad a la hora de elaborar tablas y establecer relaciones entre las informaciones contenidas en ellas. Pueden mantener la integridad de una base de datos permitiéndole a más de un usuario actualizar un registro al mismo tiempo y también puede impedir registros duplicados en una BD (Bravo, 2006).

### 2.2.2 Características de los SGBD

Los SGBD cuentan con algunas características que se mencionan a continuación:

- Permite crear y gestionar base de datos de forma fácil, cómoda y rápida.
- Ofrece una gran flexibilidad para el trabajo con base de datos relacionales.
- Ofrece un ambiente agradable dado por su interfaz gráfica.

### 2.3 Arquitectura de tres esquemas

El objetivo de la arquitectura de tres esquemas, ilustrada en la Imagen 7, es separar las aplicaciones de usuario y las bases de datos físicas.

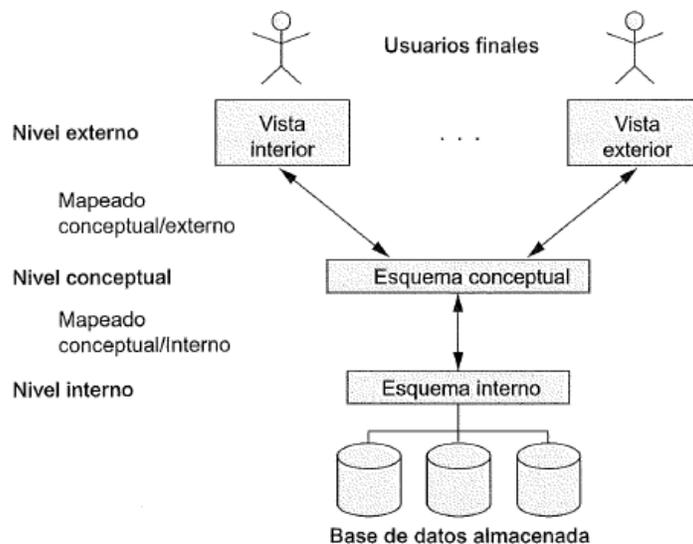


Imagen 7. Arquitectura de tres esquemas (Elmasri & Navathe, 2007)

En esta arquitectura se pueden definir esquemas en los siguientes tres niveles:

### **2.3.1 Nivel interno**

Tiene un esquema interno, que describe la estructura de almacenamiento físico de la base de datos. El esquema interno utiliza un modelo de datos físico y describe todos los detalles del almacenamiento de datos y las rutas de acceso a la base de datos (Elmasri & Navathe, 2007).

### **2.3.2 Nivel conceptual**

Tiene un esquema conceptual, que describe la estructura de toda la base de datos para una comunidad de usuarios. El esquema conceptual oculta los detalles de las estructuras de almacenamiento físico y se concentra en describir las entidades, los tipos de datos, las relaciones, las operaciones de los usuarios y las restricciones. Normalmente, el esquema conceptual se describe con un modelo de datos representativo cuando se implementa un sistema de bases de datos. Este esquema conceptual de implementación se basa a menudo en un diseño de esquema conceptual en un modelo de datos de alto nivel (Elmasri & Navathe, 2007).

### **2.3.3 Nivel de vista o externo**

Incluye una cierta cantidad de esquemas externos o vistas de usuario. Un esquema externo describe la parte de la base de datos en la que un grupo de usuarios en particular está interesado y le oculta el resto de la base de datos. Como en el caso anterior, cada esquema externo se implementa normalmente mediante un modelo de datos representativo, posiblemente basado en un diseño de esquema externo de un modelo de datos de alto nivel.

La arquitectura de tres esquemas es una buena herramienta con la que el usuario puede visualizar los niveles del esquema de un sistema de bases de datos. La mayoría de los DBMS's no separan completa y explícitamente los tres niveles, pero soportan esta arquitectura en cierta medida (Elmasri & Navathe, 2007).

## **2.4 Independencia de los datos**

La arquitectura de tres esquemas se puede utilizar para explicar el concepto de independencia de los datos, que puede definirse como la capacidad de cambiar el esquema en un nivel de un sistema de bases de datos sin tener que cambiar el esquema en el siguiente nivel más alto. Se pueden definir dos tipos de independencia de datos:

### **2.4.1 Independencia lógica de datos**

Es la capacidad de cambiar el esquema conceptual sin tener que cambiar los esquemas externos o los programas de aplicación. Es posible cambiar el esquema conceptual para expandir la base de datos (añadiendo un tipo de registro o un elemento de datos), para cambiar las restricciones o para reducir la base de datos (eliminando un tipo de registro o un elemento de datos). En el último caso, no deben verse afectados los esquemas externos que sólo se refieren a los datos restantes.

Una vez que el esquema conceptual sufre una reorganización lógica, los programas de aplicación que hacen referencia a las estructuras de esquema externo deben funcionar como antes. En el esquema conceptual se pueden introducir cambios en las restricciones sin que se vean afectados los esquemas externos o los programas de aplicación (Elmasri & Navathe, 2007).

### **2.4.2 Independencia física de datos**

Es la capacidad de cambiar el esquema interno sin que haya que cambiar el esquema conceptual. Por tanto, tampoco es necesario cambiar los esquemas externos. Puede que haya que realizar cambios en el esquema interno porque algunos archivos físicos fueran reorganizados (por ejemplo, por la creación de estructuras de acceso adicionales) de cara a mejorar el rendimiento de las recuperaciones o las actualizaciones. Si en la base de datos permanecen los mismos datos que antes, no hay necesidad de cambiar el esquema conceptual.

Por regla general, la independencia física de datos existe en la mayoría de las bases de datos y de los entornos de archivos en los que al usuario se le ocultan la ubicación exacta de los datos en el disco, los detalles hardware de la codificación del almacenamiento, la colocación, la compresión, la división, la fusión de registros, etcétera. Las aplicaciones siguen obviando estos detalles. Por el contrario, la independencia lógica de datos es muy difícil de conseguir porque permite los cambios estructurales y restrictivos sin afectar a los programas de aplicación, un requisito mucho más estricto (Elmasri & Navathe, 2007).

## **2.5 Modelo Entidad-Relación (ER)**

Es un modelo de datos conceptual de alto nivel. Este modelo y sus variaciones se utilizan con frecuencia para el diseño conceptual de las aplicaciones de base de datos, y muchas herramientas de diseño emplean estos conceptos. Describimos los conceptos básicos de la estructura de datos y

las restricciones del modelo ER, así como su uso en el diseño de esquemas conceptuales para las aplicaciones de base de datos. También presentamos la notación esquemática asociada con el modelo ER, conocida como diagramas ER. La Imagen 8 muestra un ejemplo del modelo ER. (Elmasri & Navathe, 2007).

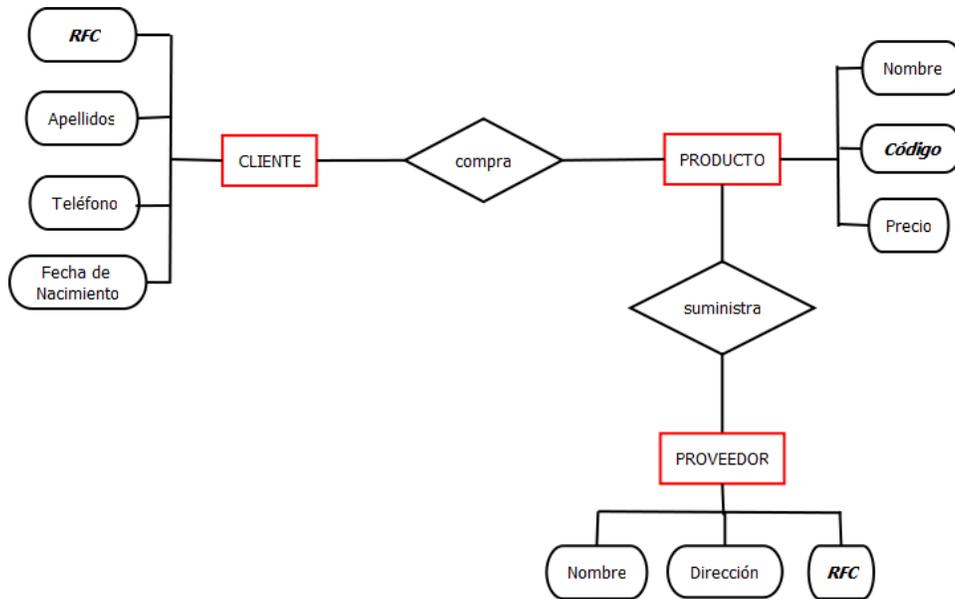


Imagen 8. Ejemplo de modelo ER

## 2.5.1 Entidades y atributos

El modelo ER describe los datos como entidades, relaciones y atributos. En las siguientes secciones se mencionan los conceptos de entidades y sus atributos, los tipos de entidad y los atributos clave.

### 2.5.1.1 Entidades

El objeto básico representado por el modelo ER es una entidad, que es una cosa del mundo real con una existencia independiente. Una entidad puede ser un objeto con una existencia física (por ejemplo, una persona en particular, un coche, una casa o un empleado) o puede ser un objeto con una existencia conceptual (por ejemplo, una empresa, un trabajo o un curso universitario). Cada entidad tiene atributos (propiedades particulares que la describen). Por ejemplo, una entidad “Empleado” se puede describir mediante el nombre, la edad, la dirección, el sueldo y el trabajo que desempeña. Una entidad en particular tendrá un valor para cada uno de sus atributos. Los

valores de los atributos que describen cada entidad se convierten en la parte principal de los datos almacenados en la base de datos (Elmasri & Navathe, 2007).

Las categorías de entidades y los ejemplos incluyen:

- **Personas:** agencia, contratista, cliente, departamento, división, empleado, instructor, estudiante, proveedor. Observe que una clase de entidad de persona puede representar a los individuos, los grupos, o las organizaciones (Bentley, 2008).
- **Lugares:** Región de ventas, edificio, cuarto, oficina regional, campus (Bentley, 2008).
- **Objetos:** libro, máquina, parte, producto, materia prima, licencia del software, paquete de software, herramienta, modelo del vehículo, vehículo. Una entidad del objeto puede representar objetos reales o especificaciones para un tipo de objeto (Bentley, 2008).
- **Eventos:** solicitud, premio, cancelación, clase, vuelo, factura, orden, inscripción, renovación, requisición, reservación, venta, viaje (Bentley, 2008).
- **Conceptos:** cuenta, bloque de tiempo, bono, curso, fondo, capacitación, suministros (Bentley, 2008).

### 2.5.1.2 *Atributos*

Si queremos almacenar datos acerca de una entidad, entonces necesitamos identificar qué piezas específicas de datos queremos almacenar acerca de cada instancia de una entidad dada. Llamamos atributos a estas piezas de datos. Por ejemplo, cada instancia de la entidad estudiante podría estar descrita por los siguientes atributos: nombre, dirección, número de teléfono, fecha de nacimiento, sexo, raza, especialidad, promedio general, y otros (Bentley, 2008).

Algunos atributos pueden agruparse lógicamente en superatributos, llamados atributos compuestos. Por ejemplo, el nombre de un estudiante es de hecho un atributo compuesto que consta de apellido paterno, apellido materno, nombre(s) (Bentley, 2008).

En el modelo ER se dan varios tipos de atributos: simple frente a compuesto, mono valor frente a multivalor, y almacenado frente a derivado (Elmasri & Navathe, 2007).

- **Atributos compuestos frente a atributos simples (atómicos):** Los atributos compuestos se pueden dividir en sub partes más pequeñas, que representan atributos más básicos con

significados independientes. Por ejemplo, el atributo Dirección de la entidad Empleado de se puede subdividir en DirCalle, Ciudad, Provincia y CP. Los atributos que no son divisibles se denominan atributos simples o atómicos. Los atributos compuestos pueden formar una jerarquía. Por ejemplo, DirCalle se puede subdividir en tres atributos simples: Número, Calle y NumApto, como se muestra en la Imagen 9. El valor de un atributo compuesto es la concatenación de los valores de sus atributos simples.

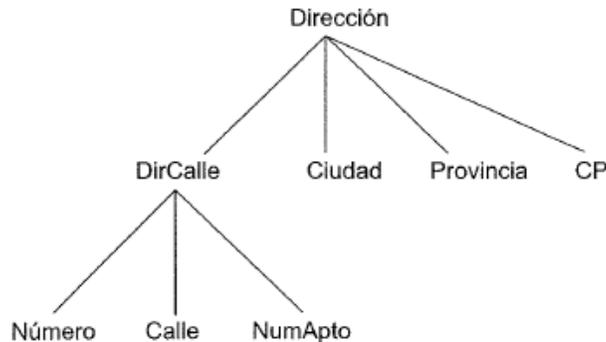


Imagen 9. Jerarquía de atributos compuestos (Elmasri & Navathe, 2007)

- **Atributos monovalor y multivalor:** la mayoría de los atributos tienen un solo valor para una entidad en particular; dichos atributos reciben el nombre de monovalor o de un solo valor. Por ejemplo, Edad es un atributo monovalor de una persona. En algunos casos, un atributo puede tener un conjunto de valores para la misma entidad (por ejemplo, un atributo Colores para un coche, o un atributo Licenciaturas para una persona). Un atributo multivalor puede tener límites superior e inferior para restringir el número de valores permitidos para cada entidad individual.
- **Atributos almacenados y derivados:** en algunos casos, dos (o más) valores de atributo están relacionados (por ejemplo, los atributos Edad y FechaNac de una persona). Para una entidad de persona en particular, el valor de Edad puede determinarse a partir de la fecha actual (el día de hoy) y el valor de FechaNac de esa persona. El atributo Edad se denomina entonces atributo derivado y se dice que se ha derivado del atributo FechaNac, que es el denominado atributo almacenado. Algunos valores de atributo se pueden derivar de entidades relacionadas.

- **Valores NULL (nulos):** en algunos casos, es posible que una entidad en particular no tenga un valor aplicable para un atributo. Por ejemplo, el atributo NumApto de una dirección sólo se aplica a las direcciones correspondientes a edificios de apartamentos, y no a otros tipos de residencias, como las casas unifamiliares. De forma parecida, un atributo Licenciaturas sólo se aplica a las personas con carrera universitaria. Para estas situaciones se ha creado un valor especial denominado NULL (nulo).
- **Atributos complejos:** los atributos compuestos y multivalor se pueden anidar arbitrariamente. Podemos representar el anidamiento arbitrario agrupando componentes de un atributo compuesto entre paréntesis () y separando los componentes con comas, y mostrando los atributos multivalor entre llaves {}. Dichos atributos se denominan atributos complejos.

### 2.5.1.3 Tipos de entidades

Un tipo de entidad define una colección o conjunto de entidades que tienen los mismos atributos. Un tipo de entidad se representa en los diagramas ER como un rectángulo con el nombre del tipo de entidad en su interior. Los nombres de los atributos se encierran en óvalos y están unidos a su tipo de entidad mediante líneas rectas. Los atributos compuestos están unidos a sus atributos componente mediante líneas rectas.

Un tipo de entidad describe el esquema o la intención de un conjunto de entidades que comparten la misma estructura. La colección de entidades de un tipo de entidad en particular está agrupada en un conjunto de entidades, que también se denomina extensión del tipo de entidad (Elmasri & Navathe, 2007).

Tipos de entidades débiles: Los tipos de entidad que no tienen atributos clave propios se denominan tipos de entidad débiles. En contraposición, los tipos de entidad regulares que tienen un atributo clave se denominan tipos de entidad fuertes. Las entidades que pertenecen a un tipo de entidad débil se identifican como relacionadas con entidades específicas de otro tipo de entidad en combinación con uno de sus valores de atributo. Podemos llamar a este otro tipo de entidad tipo de entidad identificado o propietario, y al tipo de relación que relaciona un tipo de entidad débil con su propietario lo podemos llamar relación identificativa del tipo de entidad débil (Elmasri & Navathe, 2007).

#### *2.5.1.4 Atributos clave de un tipo de entidad.*

Una restricción importante de las entidades de un tipo de entidad es la clave o restricción de unicidad de los atributos. Un tipo de entidad normalmente tiene un atributo cuyos valores son distintos para cada entidad individual del conjunto de entidades. Dicho atributo se denomina atributo clave, y sus valores se pueden utilizar para identificar cada entidad sin lugar a dudas. En la notación diagramática ER, cada atributo clave tiene su nombre subrayado dentro del óvalo.

### **2.5.2 Relaciones**

Es una asociación natural de negocios que existe entre una o más entidades. La relación puede representar un evento que enlaza a las entidades o meramente una afinidad lógica que existe entre las entidades. Por ejemplo, considere las entidades estudiante y plan de estudios. Podemos hacer las siguientes aseveraciones de negocios que enlazan a los estudiantes y los cursos:

- Un estudiante se registra en uno o más planes de estudios.
- Un plan de estudios está siendo estudiado por cero, uno o más alumnos.

Las expresiones subrayadas definen las relaciones de negocios que existen entre las dos entidades (Bentley, 2008).

En las siguientes secciones se mencionan los conceptos de tipos de relaciones, conjuntos de relaciones e instancias (también conocidas como ejemplares u ocurrencias) de relaciones, grado de relación, nombres de rol, relaciones recursivas, las restricciones estructurales en las relaciones (por ejemplo, como las razones de cardinalidad y las dependencias existentes).

#### *2.5.2.1 Tipos de relaciones*

Un tipo de relación R define un conjunto de asociaciones (o un conjunto de relaciones) entre las entidades. Como en el caso de los tipos de entidades y los conjuntos de entidades, normalmente se hace referencia a un tipo de relación y su correspondiente conjunto de relaciones con el mismo nombre, R (Elmasri & Navathe, 2007).

### 2.5.2.2 Grado de relación, nombres de rol y relaciones recursivas

El grado de un tipo de relación es el número de tipos de entidades participantes. Un tipo de relación de grado dos se denomina binario, y uno de grado tres, ternario. En la Imagen 10 se muestra un ejemplo de relación binaria.

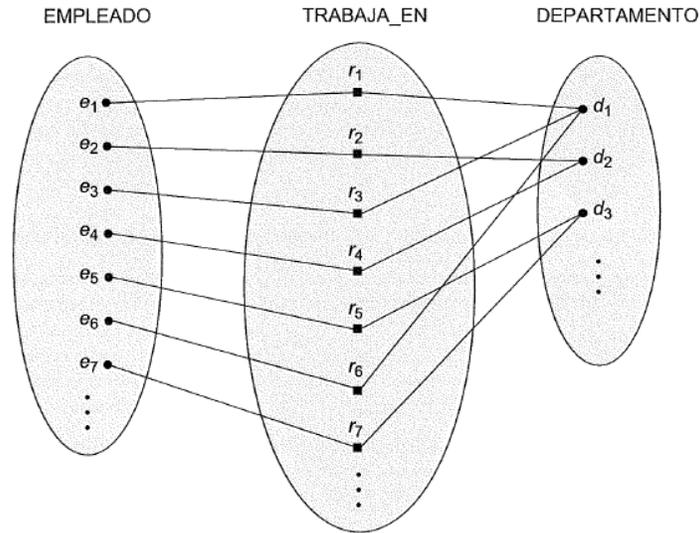


Imagen 10. Ejemplo de relación binaria (Elmasri & Navathe, 2007).

La relación TRABAJA\_PARA es de grado dos. En la Imagen 11 se muestra un ejemplo de relación ternaria.

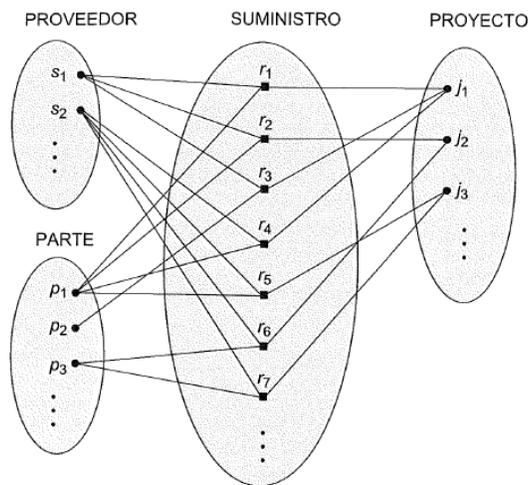


Imagen 11. Ejemplo de relación ternaria (Elmasri & Navathe, 2007).

Las relaciones pueden ser generalmente de cualquier grado, pero las más comunes son las relaciones binarias. Las relaciones de grado más alto son, por lo general, más complejas que las relaciones binarias.

- Relaciones y atributos. A veces es conveniente imaginar un tipo de relación en términos de atributos. Considere el tipo de relación TRABAJA\_PARA de la Imagen 10. Uno puede pensar en un atributo denominado Departamento del tipo de entidad EMPLEADO donde el valor de Departamento por cada entidad EMPLEADO es (una referencia a) la entidad DEPARTAMENTO para la que ese empleado trabaja. Por tanto, el conjunto de valores para este atributo Departamento es el conjunto de todas las entidades DEPARTAMENTO. No obstante, cuando pensamos en una relación binaria como si fuera un atributo, siempre tenemos dos opciones. En este ejemplo, la alternativa es pensar en un atributo Empleado multivalor del tipo de entidad DEPARTAMENTO cuyos valores por cada entidad DEPARTAMENTO es el conjunto de entidades EMPLEADO que trabajan para ese departamento. El conjunto de valores de este atributo Empleado es el conjunto potencia del conjunto de entidades EMPLEADO. Cualquiera de estos dos atributos (Departamento de EMPLEADO o Empleado de DEPARTAMENTO) puede representar el tipo de relación TRABAJA\_PARA. Si se representan ambas, se restringen para ser mutuamente inversas.
- Nombres de rol y relaciones recursivas. Cada tipo de entidad que participa en un tipo de relación juega un papel o rol particular en la relación. El nombre de rol hace referencia al papel que una entidad participante del tipo de entidad juega en cada instancia de relación, y ayuda a explicar el significado de la relación. Por ejemplo, en el tipo de relación TRABAJA\_PARA, EMPLEADO juega el papel de empleado o trabajador y DEPARTAMENTO juega el papel de departamento o empleador.

### *2.5.2.3 Restricciones en los tipos de relaciones*

Los tipos de relaciones normalmente tienen ciertas restricciones que limitan las posibles combinaciones entre las entidades que pueden participar en el conjunto de relaciones correspondiente. Estas restricciones están determinadas por la situación del minimundo representado por las relaciones. Por ejemplo, en la Imagen 10, si la empresa tiene por norma que cada empleado debe trabajar únicamente para un departamento, entonces tendríamos que describir

esta restricción en el esquema. Podemos distinguir dos tipos principales de restricciones de relación: razón de cardinalidad y participación (Elmasri & Navathe, 2007).

- **Razones de cardinalidad para las relaciones binarias.** La razón de cardinalidad de una relación binaria especifica el número máximo de instancias de relación en las que una entidad puede participar. Por ejemplo, en el tipo de relación binaria TRABAJA\_PARA, DEPARTAMENTO: EMPLEADO tiene una razón de cardinalidad de 1:N, que significa que cada departamento puede estar relacionado con (es decir, emplea a) cualquier cantidad de empleados, pero un empleado puede estar relacionado con (trabajar para) un solo departamento. Las posibles razones de cardinalidad para los tipos de relación binaria son 1:1, 1:N, N:1 y M:N. Un ejemplo de relación binaria 1:1 es ADMINISTRA (véase la Imagen 12), que relaciona una entidad departamento con el empleado que dirige ese departamento. Esto representa las restricciones del minimundo, según las cuales, en cualquier momento del tiempo, un empleado puede dirigir un solo departamento y un departamento sólo puede tener un director. El tipo de relación TRABAJA\_EN (véase la Imagen 13) tiene una razón de cardinalidad de M:N, porque la norma del minimundo es que un empleado puede trabajar en varios proyectos, y un proyecto puede tener varios empleados. Las razones de cardinalidad de las relaciones binarias se representan en los diagramas ER mediante 1, M Y N en los rombos (véase la Imagen 14) (Elmasri & Navathe, 2007).

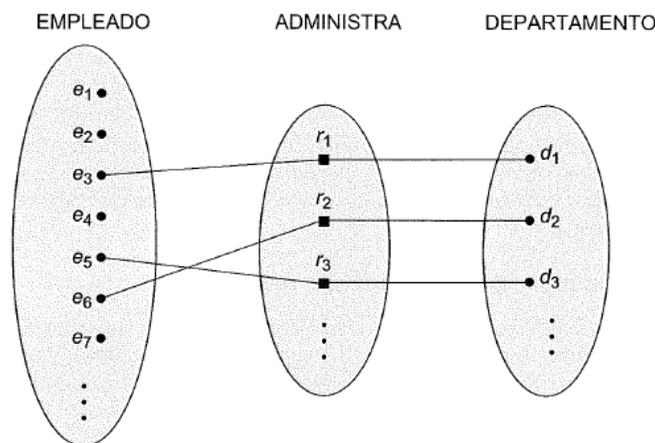


Imagen 12. Ejemplo de cardinalidad 1:1 (Elmasri & Navathe, 2007).

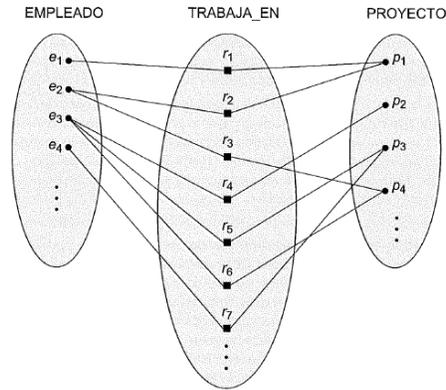


Imagen 13. Ejemplo de cardinalidad M:N (Elmasri & Navathe, 2007).

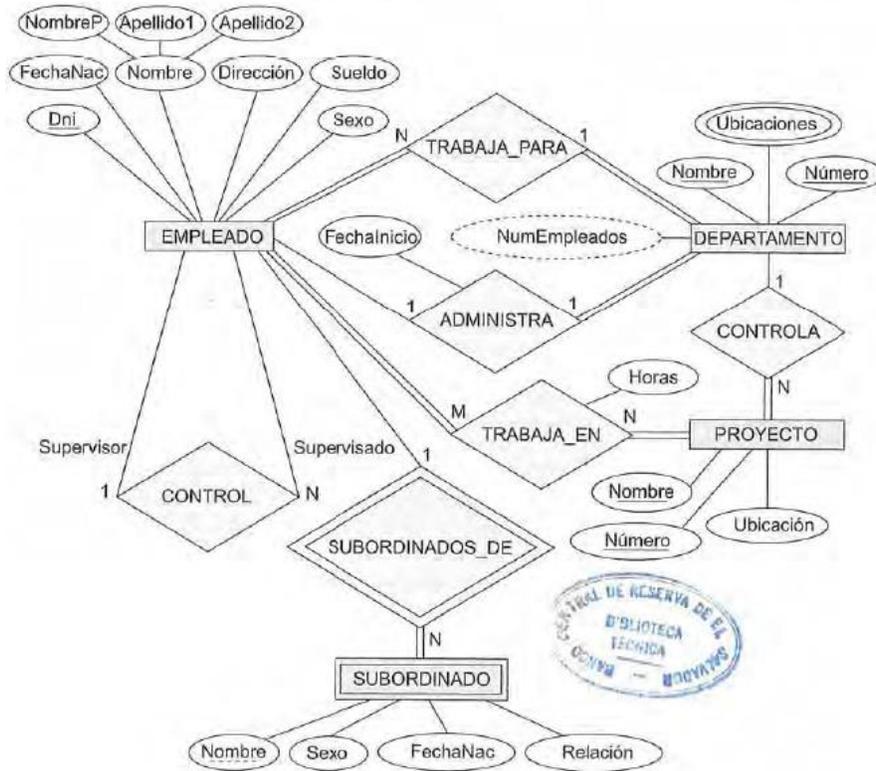


Imagen 14. Ejemplo de modelo ER con razón de cardinalidad (Elmasri & Navathe, 2007).

- Restricciones de participación y dependencias de existencia. La restricción de participación especifica si la existencia de una entidad depende de si está relacionada con otra entidad a través de un tipo de relación. Esta restricción especifica el número mínimo de instancias de relación en las que puede participar cada entidad, y en ocasiones recibe el nombre de

restricción de cardinalidad mínima. Hay dos tipos de restricciones de participación, total y parcial, que ilustramos con un ejemplo. Si una política de la empresa dice que cada empleado debe trabajar para un departamento, entonces una entidad de empleado sólo puede existir si participa en al menos una instancia de relación TRABAJA\_PARA (véase la imagen 10). De este modo, la participación de EMPLEADO en TRABAJA\_PARA se denomina participación total, es decir, cada entidad del conjunto total de entidades empleado debe estar relacionada con una entidad departamento a través de TRABAJA\_PARA. La participación total también se conoce como dependencia de existencia. En la imagen 12 no esperamos que cada empleado dirija un departamento, de modo que la participación de EMPLEADO en el tipo de relación ADMINISTRA es parcial; esto significa que algo o parte del conjunto de entidades empleado está relacionado con alguna entidad departamento a través de ADMINISTRA, pero no necesariamente con todas. Nos referiremos a la razón de cardinalidad y a las restricciones de participación, en conjunto, como restricciones estructurales de un tipo de relación (Elmasri & Navathe, 2007).

### **2.5.3 Símbolos del modelo ER**

El modelo ER representa la estructura lógica de una base de datos de forma gráfica. Los símbolos utilizados se mencionan a continuación:

- Rectángulos: representan a las entidades.
- Elipses: representan a los atributos. Si el atributo es una clave primaria va subrayado, si es multivaluado estará dentro de dos elipses. Si el atributo tiene otras elipses, el atributo es compuesto.
- Líneas: unen atributos a tipos de entidades y de tipos de entidades a tipos de relaciones. Su participación puede ser parcial (cuando no todas las entidades de un tipo de entidad participan en el tipo de relación) y total (cuando todas las entidades de un tipo de entidad participan en un tipo de relación).

La Imagen 15 muestra los símbolos utilizados en el modelo ER.

Símbolo	Significado
	Entidad
	Entidad débil
	Relación
	Relación de identificación
	Atributo
	Atributo clave
	Atributo multivalor
	Atributo compuesto
	Atributo derivado

Imagen 15. Símbolos del modelo ER (Elmasri & Navathe, 2007).

## 2.6 Modelo relacional

### 2.6.1 ¿Qué es el modelo relacional?

El modelo relacional representa la base de datos como una colección de relaciones. Informalmente, cada una de estas relaciones se parece a una tabla de valores o, de forma algo más extensa, a un fichero plano de registros. Sin embargo, existen importantes diferencias entre las relaciones y los ficheros planos.

Cuando una relación está pensada como una tabla de valores, cada fila representa una colección de valores relacionados. En el modelo relacional, cada fila de la tabla representa un hecho que, por lo general, se corresponde con una relación o entidad real. El nombre de la tabla y de las columnas se utiliza para ayudar a interpretar el significado de cada uno de los valores de las filas.

En la terminología formal del modelo relacional, una fila recibe el nombre de tupla, una cabecera de columna es un atributo y el nombre de la tabla una relación. El tipo de dato que describe los

valores que pueden aparecer en cada columna está representado por un dominio de posibles valores (Elmasri & Navathe, 2007).

## 2.6.2 Conceptos

El modelo relacional maneja los siguientes conceptos:

- **Dominio:** un dominio  $D$  es un conjunto de valores atómicos. Por atómico queremos decir que cada valor de un dominio es indivisible en lo que al modelo relacional se refiere. Una forma habitual de especificar un dominio es indicar un tipo de dato desde el que se dibujan los valores del mismo. También resulta útil darle un nombre que ayude en la interpretación de sus valores.
- **Relación:** también especificado como  $r(R)$  es un conjunto de  $n$ -tuplas  $r = \{t_1, t_2, \dots, t_m\}$ .
- **Tupla:** es una lista ordenada de  $n$  valores  $t = \langle v_1, v_2, \dots, v_n \rangle$ , donde  $v_i$   $1 \leq i \leq n$ , es un elemento de  $\text{dom}(A_i)$  o un valor especial NULL. Cada línea excepto la primera recibe el nombre de tupla y almacena ítems concretos para cada columna pertenecientes al dominio del atributo, todas las filas deben ser diferentes.
- **Cardinalidad:** es el número de filas de la tabla.
- **Grado:** es el número de columnas de la tabla.
- **Valor:** es representado por la inserción entre fila y una columna.
- **Valor Null:** representa la ausencia de la información.
- **Claves:** en una relación no hay tuplas repetidas. Para esto se requiere el concepto de clave:

**Clave primaria (Primary Key):** Es el conjunto de atributos mínimo que permite identificar plenamente a cada tupla en una relación.

**Clave alternativa:** aquellas claves candidatas que no han sido escogidas como primarias.

**Clave foránea:** se le llama clave foránea de una relación  $RI$  al conjunto de atributos cuyos valores han de coincidir con los valores de la clave primaria de otra relación. Su principal función es llevar el contenido semántico del tipo de relación del modelo ER.

Ambas claves (la clave primaria y la clave foránea) estarán definidas sobre el mismo dominio y son muy importantes en el estudio de la restricción de integridad del modelo relacional.

La Imagen 16 muestra un ejemplo de una relación ESTUDIANTE que se corresponde con el esquema del mismo nombre recién especificado. Cada tupla de la relación representa a un estudiante en particular.

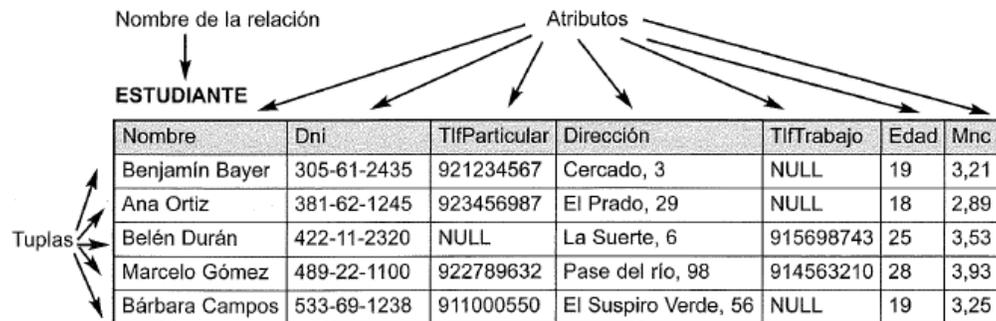


Imagen 16. Atributos y tuplas de una relación estudiante (Elmasri & Navathe, 2007).

Mostramos la relación como una tabla en la que cada tupla aparece como una fila y cada atributo como un encabezamiento de columna que indica la interpretación que habrá que dar a cada uno de los valores de la misma. Los valores NULL representan atributos cuyos valores no se conocen, o no existen, para una tupla ESTUDIANTE individual (Elmasri & Navathe, 2007).

### 2.6.3 Características

Las relaciones del modelo relacional cuentan con algunas características que se mencionan a continuación:

- **Ordenación de tuplas en una relación:** una relación está definida como un conjunto de tuplas. Matemáticamente, los elementos de un conjunto no guardan un orden entre ellos; por tanto, las tuplas en una relación tampoco la tienen. En otras palabras, una relación no es sensible al ordenamiento de las tuplas. Sin embargo, en un fichero, los registros están almacenados físicamente en el disco (o en memoria), por lo que siempre hay establecido un orden entre ellos. De forma análoga, cuando mostramos una relación como una tabla, las filas aparecen con un cierto orden.
- **Ordenación de los valores dentro de una tupla y definición alternativa de una relación:** según la definición anterior de relación, una  $n$ -tupla es una lista ordenada de  $n$

valores, por lo que el orden de valores dentro de una de ellas (y por consiguiente de los atributos de un esquema de relación) es importante. Sin embargo, a nivel lógico, el orden de los atributos y sus valores no es tan importante mientras se mantenga la correspondencia entre ellos.

- **Valores y NULL's en las tuplas:** cada valor en una tupla es un valor atómico, es decir, no es divisible en componentes dentro del esqueleto del modelo relacional básico. Por tanto, no están permitidos los atributos compuestos y multivalor. Un concepto importante es el de los valores NULL (nulo), que se utilizan para representar los valores de atributos que pueden ser desconocidos o no ser aplicables a una tupla.
- **Interpretación (significado) de una relación:** el esquema de relación puede interpretarse como una declaración o un tipo de aserción.

#### 2.6.4 Restricciones del modelo relacional

Generalmente, existen muchas restricciones en los valores de un estado de base de datos. Estas restricciones están derivadas de las reglas del minimundo que dicha base de datos representa. Éstas pueden dividirse generalmente en tres categorías principales:

1. Restricciones que son inherentes al modelo de datos y que reciben el nombre de restricciones implícitas o inherentes basadas en el modelo.
2. Restricciones que pueden expresarse directamente en los esquemas del modelo de datos, por lo general especificándolas en el DDL (Lenguaje de definición de datos, Data Definition Language). Las llamaremos restricciones explícitas o basadas en el esquema.
3. Restricciones que no pueden expresarse directamente en los esquemas del modelo de datos, y que por consiguiente deben ser expresadas e implementadas por los programas. Las llamaremos restricciones semánticas, basadas en aplicación o reglas de negocio.

Otra categoría importante de restricciones son las dependencias de datos, las cuales incluyen las dependencias funcionales y las dependencias multivalor. Suelen emplearse para comprobar la corrección del diseño de una base de datos relacional y en un proceso llamado normalización.

A continuación se describen los tipos de restricciones principales que pueden aplicarse en el modelo relacional:

#### 2.6.4.1 Restricciones de dominio

Las restricciones de dominio especifican que dentro de cada tupla, el valor de un atributo  $A$  debe ser un valor atómico del dominio  $\text{dom}(A)$ . Los tipos de datos asociados a ellos suelen incluir valores numéricos estándar para datos enteros (como entero corto, entero o entero largo) y reales (de coma flotante de simple y doble precisión). También están disponibles tipos de datos para el almacenamiento de caracteres, valores lógicos, cadenas de longitud fija y variable, fechas, horas y moneda. Es posible describir otros dominios como un sub rango de valores de un tipo de dato, o como un tipo de dato enumerado en el que todos sus posibles valores están explícitamente listados (Elmasri & Navathe, 2007).

#### 2.6.4.2 Restricciones de clave y restricciones en valores NULL

Una relación está definida como un conjunto de tuplas. Por definición, todos los elementos de un conjunto son distintos; por tanto, todas las tuplas en una relación también deben serlo. Esto significa que dos tuplas no pueden tener la misma combinación de valores para todos sus atributos. Habitualmente existen otros subconjuntos de atributos de una relación  $R$  con la propiedad de que dos tuplas en cualquier relación  $r$  de  $R$  no deben tener la misma combinación de valores para estos atributos. Una superclave puede contar con atributos redundantes, por lo que un concepto más importante es el de clave, que no tiene redundancia. Una clave  $K$  de un esquema de relación  $R$  es una superclave de  $R$  con la propiedad adicional que eliminando cualquier atributo  $A$  de  $K$  deja un conjunto de atributos  $K$  que ya no es una superclave de  $R$ . Por tanto, una clave satisface dos restricciones:

1. Dos tuplas diferentes en cualquier estado de la relación no pueden tener valores idénticos para (todos) los atributos de la clave.
2. Es una superclave mínima, es decir, una superclave de la cual no podemos eliminar ningún atributo y seguiremos teniendo almacenada la restricción de exclusividad de la condición 1.

La primera condición se aplica tanto a las claves como a las superclaves, mientras que la segunda sólo a las claves (Elmasri & Navathe, 2007).

## 2.7 Transformación del modelo ER al modelo relacional

En esta sección se explican los pasos de un algoritmo convertir las estructuras de un modelo ER básico en relaciones.

### 2.7.1 Algoritmo de mapeado ER a relacional

1. Por cada entidad (fuerte) regular  $E$  del modelo ER, crear una relación  $R$  que incluya todos los atributos simples de  $E$ . Incluir únicamente los atributos simples que conforman un atributo compuesto. Seleccionar uno de los atributos clave de  $E$  como clave principal para  $R$ . Si la clave elegida de  $E$  es compuesta, entonces el conjunto de los atributos simples que la forman constituirán la clave principal de  $R$ .
2. Por cada tipo de entidad débil  $W$  del modelo ER con el tipo de entidad propietario  $E$ , crear una relación  $R$  e incluir todos los atributos simples (o componentes simples de los atributos compuestos) de  $W$  como atributos de  $R$ . Además, incluir como atributos de la clave foránea de  $R$ , el(los) atributo(s) de la o las relaciones que correspondan al o los tipos de entidad propietarios; esto se encarga de identificar el tipo de relación de  $W$ . La clave principal de  $R$  es la combinación de la(s) clave(s) principal(es) del o de los propietarios y la clave parcial del tipo de entidad débil  $W$ , si la hubiera. Si hay un tipo de entidad débil  $E_2$  cuyo propietario también es un tipo de entidad débil  $E_1$ , entonces  $E_1$  debe asignarse antes que  $E_2$  para determinar primero su clave principal.
3. Por cada tipo de relación 1:1 binaria  $R$  del modelo ER, identificar las relaciones  $S$  y  $T$  que corresponden a los tipos de entidad que participan en  $R$ . Hay tres metodologías posibles: (1) la metodología de la clave foránea, (2) la metodología de la relación mezclada y (3) la metodología de referencia cruzada o relación de relación. La primera metodología es la más útil y la que debe seguirse salvo que se den ciertas condiciones especiales, como las que explicamos a continuación:
  - **Metodología de la clave foránea:** seleccionar una de las relaciones (por ejemplo,  $S$ ) e incluir como llave foránea en  $S$  la clave principal de  $T$ . Lo mejor es elegir un tipo de entidad con participación total en  $R$  en el papel de  $S$ . Incluir todos los atributos simples (o los componentes simples de los atributos compuestos) del tipo de relación 1:1  $R$  como atributos de  $S$ .

- **Metodología de la relación mezclada:** una asignación alternativa de un tipo de relación 1:1 es posible al mezclar los dos tipos de entidad y la relación en una sola relación. Esto puede ser apropiado cuando las dos participaciones son totales.
  - **Metodología de referencia cruzada o relación de relación:** la tercera opción consiste en configurar una tercera relación  $R$  con el propósito de crear una referencia cruzada de las claves principales de las relaciones  $S$  y  $T$  que representan los tipos de entidad. Como veremos, esta metodología es necesaria para las relaciones M:N binarias. La relación  $R$  se denomina relación de relación, y, en algunas ocasiones, tabla de búsqueda, porque cada tupla de  $R$  representa una instancia de relación que relaciona una tupla de  $S$  con otra de  $T$ .
4. Por cada relación 1:N binaria regular  $R$ , identificar la relación  $S$  que representa el tipo de entidad participante en el lado  $N$  del tipo de relación. Incluir como llave foránea en  $S$  la clave principal de la relación  $T$  que representa el otro tipo de entidad participante en  $R$ ; hacemos esto porque cada instancia de entidad en el lado  $N$  está relacionada, a lo sumo, con una instancia de entidad del lado 1 del tipo de relación. Incluir cualesquiera atributos simples (o componentes simples de los atributos compuestos) del tipo de relación 1:N como atributos de  $S$ .
  5. Por cada tipo de relación M:N binaria  $R$ , crear una nueva relación  $S$  para representar a  $R$ . Incluir como atributos de la clave foránea en  $S$  las claves principales de las relaciones que representan los tipos de entidad participantes; su combinación formará la clave principal de  $S$ . Incluir también cualesquiera atributos simples del tipo de relación M:N (o los componentes simples de los atributos compuestos) como atributos de  $S$ . No se puede representar un tipo de relación M:N con un atributo de llave foránea en una de las relaciones participantes debido a la razón de cardinalidad M:N; se debe crear una relación de relación  $S$  separada.
  6. Por cada atributo multivalor  $A$ , crear una nueva relación  $R$ . Esta relación incluirá un atributo correspondiente a  $A$ , más el atributo clave principal  $K$ , como clave foránea en  $R$ , de la relación que representa el tipo de entidad o tipo de relación que tiene  $A$  como un atributo. La clave principal de  $R$  es la combinación de  $A$  y  $K$ . Si el atributo multivalor es compuesto, se incluyen sus componentes simples.

7. Por cada tipo de relación  $n$ -ario  $R$ , donde  $n > 2$ , crear una nueva relación  $S$  para representar  $R$ . Incluir como atributos de la clave foránea en  $S$  las claves principales de las relaciones que representan los tipos de entidad participantes. Incluir también cualesquiera atributos simples del tipo de relación  $n$ -ario (o los componentes simples de los atributos compuestos) como atributos de  $S$ . Normalmente, la clave principal de  $S$  es una combinación de todas las claves foráneas que hacen referencia a las relaciones que representan los tipos de entidad participantes. No obstante, si las restricciones de cardinalidad en cualquiera de los tipos de entidad  $E$  que participan en  $R$  es 1, entonces la clave principal de  $S$  no debe incluir el atributo de la clave foránea que hace referencia a la relación  $E$  correspondiente a  $E$  (Elmasri & Navathe, 2007).

# Capítulo 3 SISTEMAS DE INFORMACIÓN

## 3.1 Definición de sistemas de información

Un sistema de información (SI) es un conjunto de elementos interrelacionados con el propósito de prestar atención a las demandas de información de una organización, para elevar el nivel de conocimientos que permitan un mejor apoyo a la toma de decisiones y desarrollo de acciones (Ayala, 2006).

También puede definirse como un conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio (Quito, nd).

## 3.2 Elementos de los sistemas de información

Los componentes más importantes de un sistema de información son los siguientes:

- **Financieros:** es el aspecto económico que permite la adquisición, contratación y mantenimiento de los demás recursos que integran un sistema de información.
- **Administrativos:** es la estructura orgánica de objetivos, lineamientos, funciones, procedimientos, departamentalización, dirección y control de las actividades; que sustenta la creación y uso de los sistemas.
- **Humanos:** está compuesto por dos grupos; el técnico, que posee los conocimientos especializados en el desarrollo de sistemas, siendo estos los administradores, líderes de proyecto, analistas, programadores, operadores y capturistas. El usuario, representado por las personas interesadas en el manejo de información vía cómputo, como apoyo al mejor desempeño de sus actividades, siendo estos los funcionarios, contadores, ingenieros, empleados, público, etc.
- **Materiales:** son aquellos elementos físicos que soportan el funcionamiento de un sistema de información, por ejemplo: local de trabajo, instalaciones eléctricas y de aire acondicionado, medios de comunicación, mobiliario, maquinaria, papelería, etc.

### 3.3 Ejemplos de sistemas de información

Existe gran cantidad y variedad de sistemas de información orientados a diversos campos de la actividad humana, algunas de sus aplicaciones son las siguientes:

- Científicas: control de vuelos espaciales, estudios sobre energía nuclear, etc.
- Militares: control y operación de armas, sistemas de defensa, etc.
- De Ingeniería: diseño industrial de autos y maquinaria; control de procesos de producción y siderúrgicos, etc.
- Comunicaciones y Transportes: diseño de carreteras, control de tráfico aéreo y terrestre, transmisión remota de datos, etc.
- Administrativas: nómina, inventarios, almacenes, contabilidad, presupuesto, ventas, etc.
- En la educación: apoyo en la enseñanza, referencias bibliográficas, etc.
- Médicas: diagnóstico, terapéuticas, aparatos biónicos, etc.
- Artísticas: música, poesía, pintura, etc.

### 3.4 Actividades de los Sistemas de Información

Los sistemas de información cuentan con cuatro actividades que producen la información que las organizaciones necesitan para tomar decisiones, controlar operaciones, analizar problemas y crear nuevos productos o servicios. Estas actividades son entrada, almacenamiento, procesamiento y salida.

#### 3.4.1 Entrada

Captura o recolecta datos en bruto tanto del interior de la organización como de su entorno externo. Por ejemplo:

- Datos generales del cliente: nombre, dirección, tipo de cliente, etc.
- Políticas de créditos: límite de crédito, plazo de pago, etc.
- Facturas (interface automático).
- Pagos, depuraciones, etc.

### **3.4.2 Almacenamiento de información**

Es una de las actividades o capacidades más importantes que tiene una computadora, ya que el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos. Por ejemplo:

- Movimientos del mes (pagos, depuraciones).
- Catálogo de clientes.
- Facturas.
- Salidas
- Reporte de pagos.
- Estados de cuenta.
- Pólizas contables (interface automática)
- Consultas de saldos en pantalla de una terminal.

### **3.4.3 Procesamiento**

Convierte esa entrada de datos en una forma más significativa. Por ejemplo:

- Cálculo de antigüedad de saldos.
- Cálculo de intereses moratorios.
- Cálculo del saldo de un cliente.

### **3.4.4 Salida**

Transfiere la información procesada a la gente que la usará o a las actividades para las que se utilizará.

## **3.5 Beneficios de los Sistemas de Información**

En nuestros días es importantísimo que las empresas integren este tipo de tecnología a través de plataformas que con un solo movimiento repercuten en otros departamentos de la empresa. Así por ejemplo una orden de producción implica la orden de compra de materiales al proveedor, descontar de los materiales disponibles el inventario, generar información de costos, generar una cuenta por pagar y finalmente descontar el flujo del efectivo cuando esta orden de materiales sea

pagada en el futuro, todo esto integrado en una infraestructura que pueda ser utilizado por varios usuarios simultáneamente; haciendo consultas, capturas y finalmente arrojando reportes para la toma de decisiones (Bracho, 2011).

Los beneficios que se pueden obtener usando sistemas de información son los siguientes:

- Acceso rápido a la información y por ende mejora en la atención a los usuarios.
- Mayor motivación en los mandos medios para anticipar los requerimientos de las directivas.
- Generación de informes e indicadores, que permiten corregir fallas difíciles de detectar y controlar con un sistema manual.
- Posibilidad de planear y generar proyectos institucionales soportados en sistemas de información que presentan elementos claros y sustentados.
- Evitar pérdida de tiempo recopilando información que ya está almacenada en bases de datos que se pueden compartir.
- Impulso a la creación de grupos de trabajo e investigación debido a la facilidad para encontrar y manipular la información.
- Soluciona el problema de falta de comunicación entre las diferentes instancias. A nivel directivo se hace más efectiva la comunicación.
- Organización en el manejo de archivos e información clasificada por temas de interés general y particular.
- Generación de nuevas dinámicas, utilizando medios informáticos como el correo electrónico, teleconferencia, acceso directo a bases de datos y redes.
- Acceso a programas y convenios e intercambios institucionales.
- Aumento de la productividad gracias a la liberación de tiempos en búsqueda y generación de información repetida.

La implementación de un Sistema de Información representa el primer paso a la integración de la Cadena de Valor, ya que integra la parte interna-administrativa de los negocios, para mejorar los procesos internos, generando ganancias tangibles e intangibles. Muchas compañías que deciden

crecer ven como atractivo las ventajas de utilizar “algo de la otra compañía”, por ejemplo la manera de administrar, la experiencia de mercado, el uso de la información que ellas mismas generan y por este motivo es muy importante tener un equipo de expertos en la administración del cambio de plataforma de sistemas de información para poder implementar la congruencia de dicha tecnología y se logre el éxito (Bracho, 2011).

# Capítulo 4 TECNOLOGÍAS A IMPLEMENTAR

## 4.1 Java

### 4.1.1 Historia de Java

Java no se pensó originalmente como lenguaje para internet, Sun Microsystems, la empresa estadounidense creadora del lenguaje y de la plataforma, comenzó a desarrollarlo con el objetivo de crear un lenguaje independiente de la plataforma y del sistema operativo que permitiera su diseño y construcción en la floreciente electrónica de consumo (dispositivos como televisores, reproductores de video, equipos de música, etcétera).

El proyecto original, denominado Green, comenzó apoyándose en C++ pero a medida que progresaba su desarrollo, el equipo creador se encontró con dificultades, especialmente de portabilidad; para evitar esto decidieron desarrollar su propio lenguaje, y en agosto de 1991 nació uno nuevo orientado a objetos y al cual llamaron Oak. En 1993, Green se renombró First Person Juc.; Sun invirtió, sin mucho éxito, un gran presupuesto y esfuerzo fundamentalmente humano para intentar vender esta tecnología, hardware y software.

A mitad de 1993 se lanzó Mosaic, el primer gran navegador web, y comenzó a crecer el interés por internet (y en particular por la World Wide Web); después se rediseñó el lenguaje para desarrollar internet y, en enero de 1995, Oak se convirtió en Java. Sun lanzó el entorno JDK 1.0 (java development kit) en 1996 como primera versión del kit de desarrollo de dominio público y se convirtió en la primera especificación formal de la plataforma Java; desde entonces se han lanzado diferentes versiones, aunque JDK 1.1, la primera versión comercial, se lanzó a principios de 1997.

En diciembre de 1998, Sun lanzó JDK 1.2 pero la renombró como Java 2 y comenzó a utilizarse el nombre de J2SE (Java 2 Platform, Standard Edition) para diferenciar las plataformas base de J2EE (Java 2 Platform, Enterprise Edition) y J2ME (Java 2 Platform, Micro Edition); además de

la versión estándar SE, Sun lanzó otras dos ediciones populares: Micro Edition (ME) para dispositivos empotrados (embebidos) tales como teléfonos celulares (móviles) y la edición empresarial (Enterprise Edition, EE) para procesamiento desde el servidor. Este libro se centra esencialmente en la edición SE. En mayo de 2000 se lanzó J2SE 1.3, y en febrero de 2002, la J2SE 1.4; ambas trajeron consigo un gran número de clases e interfaces a las bibliotecas estándar de Java.

Sin embargo fue la versión 5.0, la primera después de la versión 1.1, la que implicó una actualización de Java de modo significativo; dicha versión originalmente se nombró 1.5, pero el número se cambió a 5.0 en la conferencia JavaOne de 2004. Después de años de investigación se añadieron tipos genéricos similares a las plantillas de C++, también se agregaron propiedades de C# (el lenguaje creado por Microsoft), precisamente para competir con Java en internet, como el bucle “for each”, así como el manejo de metadatos para su uso, principalmente, en bases de datos, y otras características como enumeraciones, varargs (número de argumentos variables), entre otras.

La versión 6 (sin el sufijo 0) se lanzó en diciembre de 2006 y hoy día es la más utilizada y recomendada para su descarga del sitio web de Sun o de Oracle (su actual propietario), cuyas direcciones web se indican en el apartado 1.12. Esta versión no ha traído cambios al lenguaje, sino mejoras adicionales al rendimiento y a la interfaz gráfica, así como un nuevo marco de trabajo y API (interfaces de programación de aplicaciones) junto con soporte para servicios web e implementaciones de JavaScript, fundamentalmente para los buscadores, tales como Firefox de la fundación Mozilla.

Al poco tiempo de liberar la versión 6, Sun4 comenzó un nuevo proyecto cuyo nombre clave es Dolphin, aunque ha comenzado a denominarse Java 7 y se espera que a mediados de 2011 se lance Java SE 7 y a finales de 2012, Java SE 8. Al parecer traerá una nueva biblioteca de clases, junto con soporte para XML, un nuevo concepto de súper paquete, introducción de anotaciones estándar para detectar fallos en software, nuevas API para manejo de calendario de fechas y días, etcétera (Joyanes Aguilar & Zahonero Martínez, 2011).

En la Tabla 2 se muestra la evolución de Java desde la primera versión hasta la 6.0. En la tabla no se incluye la versión Java 7 porque todavía no se conocen sus especificaciones finales.

Tabla 2. Evolución de Java (Joyanes Aguilar & Zahonero Martínez, 2011)

Versión	Año	Nuevas características del lenguaje	Número de clases e interfaces
1.0	1996	Definición del lenguaje	211
1.1	1997	Clases internas	477
1.2	1998	Ninguna	1524
1.3	2000	Ninguna	1840
1.4	2004	Aserciones	2723
5.0	2004	Clases genéricas, bucles <i>for each</i> , <i>varargs</i> , autoboxing, metadatos, enumeraciones, importancia estática	3279
6	2006	Ninguna	3777

#### 4.1.2 ¿Qué es Java?

Java es un lenguaje de programación de propósito general, posiblemente, uno de los más populares y más utilizados en el desarrollo de programas de software, especialmente para internet y web; actualmente se encuentra en numerosas aplicaciones, dispositivos, redes de comunicaciones, etcétera, como:

- Servidores web.
- Bases de datos relacionales.
- Sistemas de información geográfica (SIG/GIS, Geographical Information System).
- Teléfonos celulares (móviles).
- Sistemas de teledetección.
- Asistentes digitales personales (PDA).
- Sistemas medioambientales.

Pero Java no sólo es un lenguaje de programación, sino que también constituye una plataforma completa para el desarrollo de software; posee una biblioteca gigantesca de clases y aplicaciones con numerosos códigos reutilizables y un entorno de programación que proporciona servicios tales como seguridad, portabilidad entre sistemas operativos y recolección automática de basura (automatic garbage collection) (Joyanes Aguilar & Zahonero Martínez, 2011).

### **4.1.3 Características**

En C, C++ o Pascal, el compilador traduce el código fuente directamente al lenguaje máquina de su computadora, entendible por su CPU; estos lenguajes necesitan un compilador diferente para cada tipo de CPU y en consecuencia, cuando los programas escritos en estos lenguajes se traducen a código máquina no son portables, de modo que el código fuente debe ser recompilado para cada tipo de máquina o CPU. Para solucionar este problema y hacer que los programas de Java sean independientes de la máquina y se puedan transportar o “portar” fácilmente a otras máquinas y también puedan ejecutarse en navegadores web, los creadores de Java introdujeron el concepto, citado anteriormente, de máquina virtual Java y el bytecode como el lenguaje máquina de la CPU específica, donde la máquina virtual Java los ejecuta o interpreta. Existen numerosas máquinas virtuales disponibles para un gran número de plataformas que permiten a los programas ser independientes de la máquina, de modo que un programa compilado en una estación UNIX puede ejecutarse en un sistema operativo Macintosh o en Windows 7; esta característica se debe a que el intérprete de Java traduce y ejecuta una instrucción de bytecode cada vez sin traducir el código completo como sucede con otros compiladores. Un compilador Java traduce un programa fuente en Java a bytecode, el lenguaje máquina de la máquina virtual Java (JVM) (Joyanes Aguilar & Zahonero Martínez, 2011).

Los creadores de Java escribieron un artículo en el que definen el lenguaje y recogen sus once características más sobresalientes:

- Sencillo.
- Orientado a objetos.
- Distribuido (características de red, especialmente internet).
- Portable.

- Interpretado.
- Robusto.
- Seguro.
- Arquitectura neutra.
- Alto rendimiento.
- Multihilo (multithreaded).
- Dinámico.

Otra fortaleza de Java, como se verá más adelante al estudiar paquetes y bibliotecas, es que incluye bibliotecas de clases incorporadas; dichos paquetes vienen con los entornos de desarrollo JDK (Java development kit) y contienen centenares de clases integradas con millares de métodos.

#### **4.1.4 Ventajas**

Es orientado a objetos, y esto resulta muy conveniente para la mayoría de las aplicaciones, y es esencial para los videojuegos. Ser orientado a objetos permite un gran control sobre el código y una mejor organización, dado que basta con escribir una vez los métodos y las propiedades de un objeto, independientemente de la cantidad de veces que se utilicen.

Es muy flexible, ya que Java es un lenguaje especialmente preparado para la reutilización del código. Permite a sus usuarios tomar un programa que hayan desarrollado tiempo atrás y actualizarlo con mucha facilidad, sea que necesiten agregar funciones o adaptarlo a un nuevo entorno.

Funciona en cualquier plataforma, a diferencia de otros programas que requieren de versiones específicas para cada sistema operativo. Las aplicaciones desarrolladas en Java funcionan en cualquier entorno, dado que no es el sistema quien las ejecuta, sino la máquina virtual, conocida como Java Virtual Machine o JVM.

Su uso no tiene ningún costo ni es necesario adquirir alguna licencia, sino simplemente descargar el kit de desarrollo.

Java ofrece el código de casi todas sus librerías nativas para que los desarrolladores puedan conocerlas y estudiarlas en profundidad, o bien ampliar su funcionalidad (Definicion.de, 2015).



*Imagen 17. Logo Java*

## 4.2 NeatBeans

### 4.2.1 ¿Qué es NeatBeans?

Netbeans es un entorno de desarrollo gratuito y de código abierto que en el momento de escribir este artículo está en su versión 7.4. Permite el uso de un amplio rango de tecnologías de desarrollo tanto para escritorio, como aplicaciones Web, o para dispositivos móviles. Da soporte a las siguientes tecnologías, entre otras: Java, PHP, Groovy, C/C++, HTML5,... Además puede instalarse en varios sistemas operativos (GenBeta dev, 2014).

### 4.2.2 Historia

Nos tenemos que remontar a 1996 e ir hasta Praga, en la república Checa, donde nace un proyecto llamado Xelfi en el ámbito universitario (Facultad de Matemáticas y Física). Se pretendía escribir un IDE para Java que se pareciera al que tenía el lenguaje Delphi (de ahí el nombre de Xelfi). El código fue escrito en Java, y se convirtió en el primero escrito en dicho lenguaje con la publicación de su primera pre-release en el año 1997.

Los autores vieron suficiente interés por el proyecto, para formar una empresa y convertirlo en un proyecto comercial, con la inversión del empresario Roman Stanek. En 1999, en primavera, vería la luz Netbeans DeveloperX2, con soporte para Swing, que posteriormente se vería modificado para adaptarse al JDK 1.3, y hacerlo más modular.

Sun Microsystems se interesará por el proyecto en 1999, firmando un acuerdo en otoño de ese año. En Junio del año 2000, la empresa Sun Micro Systems funda el proyecto Netbeans bajo los auspicios del software de código abierto (GenBeta dev, 2014).

### 4.2.3 Características

A continuación se enlistan algunas características de Netbeans:

- Suele dar soporte a casi todas las novedades en el lenguaje Java. Cualquier preview del lenguaje es rápidamente soportada por Netbeans.
- Asistentes para la creación y configuración de distintos proyectos, incluida la elección de algunos frameworks.
- Buen editor de código, multilenguaje, con el habitual coloreado y sugerencias de código, acceso a clases pinchando en el código, control de versiones, localización de ubicación de la clase actual, comprobaciones sintácticas y semánticas, plantillas de código, coding tips, herramientas de refactorización, entre otras.
- Simplifica la gestión de grandes proyectos con el uso de diferentes vistas, asistentes de ayuda, y estructurando la visualización de manera ordenada, lo que ayuda en el trabajo diario.
- Herramientas para depurado de errores. El debugger que incluye el IDE es bastante útil para encontrar dónde fallan las cosas.
- Optimización de código. El Profiler nos ayuda a optimizar nuestras aplicaciones e intentar hacer que se ejecuten más rápido y con el mínimo uso de memoria. Podemos igualmente configurarlo a nuestro gusto.
- Acceso a base de datos. Desde Netbeans podemos conectarnos a distintos sistemas gestores de bases de datos, como pueden ser Oracle, MySql y demás, y ver las tablas, realizar consultas y modificaciones, y todo ello integrado en el propio IDE.
- Se integra con diversos servidores de aplicaciones, de tal manera que podemos gestionarlos desde el propio IDE (inicio, parada, arranque en modo debug, despliegues). Entre otros

podemos usar Apache Tomcat, GlassFish, JBoss, WebLogic, Sailfin, Sun Java System Application Server.

- Es fácilmente extensible a través de plugins.



*Imagen 18. Logo de NetBeans*

## 4.3 MySQL

### 4.3.1 ¿Qué es MySQL?

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario (Castell Delgado, 2014).

El servidor está proyectado tanto para sistemas críticos en producción soportando intensas cargas de trabajo como para empotrarse en sistemas de desarrollo masivo de software. El software MySQL tiene licencia dual, pudiéndose usar de forma gratuita bajo licencia GNU o bien adquiriendo licencias comerciales de MySQL AB en el caso de no desear estar sujeto a los términos de la licencia GPL. MySQL es una marca registrada de MySQL AB (Mastermagazine, 2010).

El servidor de bases de datos MySQL es muy rápido, seguro, y fácil de usar. El servidor MySQL fue desarrollado originalmente para manejar grandes bases de datos mucho más rápido que las soluciones existentes y ha estado siendo usado exitosamente en ambientes de producción sumamente exigentes por varios años. Aunque se encuentra en desarrollo constante, el servidor

MySQL ofrece hoy un conjunto rico y útil de funciones. Su conectividad, velocidad, y seguridad hacen de MySQL un servidor bastante apropiado para acceder a bases de datos en Internet (Inform@tica, s. f.).

El software de bases de datos MySQL consiste de un sistema cliente/servidor que se compone de un servidor SQL multihilo, varios programas clientes y bibliotecas, herramientas administrativas, y una gran variedad de interfaces de programación (APIs). Se puede obtener también como una biblioteca multihilo que se puede enlazar dentro de otras aplicaciones para obtener un producto más pequeño, más rápido, y más fácil de manejar (Inform@tica, s. f.).

### **4.3.2 Características de MySQL**

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas web con contenido dinámico, justamente por su simplicidad (Castell Delgado, 2014).

Poco a poco los elementos de los que carecía MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre. Entre las características disponibles en las últimas versiones se puede destacar:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponibilidad en gran cantidad de plataformas y sistemas.
- Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferentes velocidades de operación, soporte físico, capacidad, distribución geográfica, transacciones.
- Transacciones y claves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda e indexación de campos de texto.

### **4.3.3 Ventajas**

MySQL cuenta con varias ventajas, éstas se mencionan a continuación:

- MySQL software es Open Source.

- Velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento.
- Bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Facilidad de configuración e instalación.
- Soporta gran variedad de Sistemas Operativos
- Baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que está.
- Su conectividad, velocidad, y seguridad hacen de MySQL Server altamente apropiado para acceder bases de datos en Internet
- El software MySQL usa la licencia GPL

#### 4.3.4 Desventajas

A continuación se enlistan las desventajas de MySQL:

- Un gran porcentaje de las utilidades de MySQL no están documentadas.
- No es intuitivo, como otros programas (ACCESS).



*Imagen 19. Logo de MySQL*

### 4.3.5 Workbench

MySQL Workbench es una herramienta visual de diseño de bases de datos que integra desarrollo de software, Administración de bases de datos, diseño de bases de datos, creación y mantenimiento para el sistema de base de datos MySQL. Es el sucesor de DBDesigner 4 de fabFORCE.net, y reemplaza el anterior conjunto de software, MySQL GUI Tools Bundle (Castell Delgado, 2014).



*Imagen 20. Logo de Workbench*

#### 4.3.5.1 Características de Workbench

##### **General:**

- Conexión a base de datos & Instance Management.
- Asistente de puntos de acción.
- Completamente programables con Python y Lua.
- Soporte para plugins personalizados.

##### **Editor de SQL:**

- Esquema de objeto navegación.
- Resaltador de sintaxis SQL y analizador declaración.
- Múltiple, conjuntos de resultados editables.
- Colecciones de fragmentos SQL.
- Conexión tune SSH.
- Soporte Unicode

### **Modelado de datos:**

- Diagrama entidad relación.
- Arrastrar y soltar modelado visual.
- Ingeniería inversa a partir de secuencias de comandos SQL y bases de datos en directo.
- Ingeniería directa de secuencias de comandos SQL y bases de datos en directo.
- Sincronización de esquema.
- Diseña y administra base de datos avanzada.
- Impresión de modelos.
- Importa de fabFORCE.net DBDesigner4.

### **Administración de bases de datos:**

- Iniciar y detener la base de datos de instancias.
- Configuración de instancia.
- Gestión de la cuenta de base de datos.
- Variables de instancia de navegación.
- Registrar la navegación de archivos.
- Volcado de datos de exportación/importación.

#### *4.3.5.2 Ventajas*

Workbench cuenta con algunas ventajas que se enlistan a continuación:

- Brinda libertad a los usuarios.
- Puede ser usado, copiado, estudiado, modificado y redistribuido.
- Ahorros multimillonarios en la adquisición de licencias
- Tiende a ser muy eficiente (porque mucha gente lo optimiza).

#### 4.3.5.3 *Desventajas*

A continuación se mencionan algunas desventajas de Workbench:

- El software libre y el software no comercial son en realidad incompatibles con el software comercial.
- El software libre crea riesgos legales.
- El software libre no tiene garantía proveniente del autor.
- Disminuye el índice de software “pirata”.

# Capítulo 5 EVENTOS ACADÉMICOS

## 5.1 ¿Qué es un evento?

Un evento académico es un espacio creado por las empresas, profesionales, etc., para complementar la formación adquirida en la universidad, instituto y/o colegio. Es una oportunidad que tienen los interesados de realizar un trabajo de calidad tanto en su contenido, como en su forma; de tal manera que su contenido sea positivo para cada uno de los participantes y se puedan cumplir los objetivos propuestos por dicho evento. Los eventos académicos se realizan para que profesionales de ciertas áreas, lugar o afición lo aprovechen y enriquezcan su formación integral (Bedoya García, 2012).

## 5.2 Tipo de eventos

Los eventos académicos abarcan varios tipos, desde congresos hasta talleres. A continuación se definen algunos de éstos:

### 5.2.1 Congreso

Reunión, generalmente periódica, de varias personas para deliberar y tratar sobre alguna materia o algún asunto previamente establecido. (word reference, 2015).

También puede definirse como una reunión o conferencia, generalmente periódica, en que los miembros de una asociación se reúnen para debatir cuestiones previamente fijadas (Cruz, 2011).

#### 5.2.1.1 Características

Entre las características principales de los congresos podemos destacarlas siguientes:

- La exposición y debate de múltiples ponencias
- Asistencia de personas con un alto nivel profesional
- Intereses comunes

- Presentaciones de nuevos avances o descubrimientos en determinadas materias

Según los expertos debe al menos durar de 3 a 5 días, para poderle llamar congreso. Su finalidad pueden ser múltiples y diversas. Desde dar a conocer nuevos avances en cualquier campo de la ciencia, la tecnología o la medicina, hasta tener un carácter cultural, social o político. Todo depende de la finalidad que los organizadores del mismo deseen tener como objetivo (Infante, Arrieta, & Cruz, 2011).

### *5.2.1.2 Tipo de congresos*

#### *5.2.1.2.1 Según su periodicidad*

Estos congresos son los celebrados cada determinado tiempo.

- Congresos ordinarios: son aquellos, que con independencia de sus objetivos, se celebran con una cierta periodicidad. Pueden ser anuales, bianuales, etc. En algunos países también se les llaman congresos periódicos.
- Congresos extraordinarios: son aquellos que celebran de forma extraordinaria, por diversas razones, bien por hechos fuera de lo común o por razones extraordinarias. Este tipo de congresos se pueden convertir en ordinarios, si luego tienen una cierta periodicidad.

#### *5.2.1.2.2 Según el contenido de los congresos*

Éstos son los que hablan de algún campo en específico.

- Congresos Científicos: aquellos en los que se tratan todo tipo de avances o novedades en el campo de las ciencias. También pueden tener un contenido académico.
- Congresos Académicos: son aquellos en los que se tratan distintos aspectos relacionados con el mundo de la enseñanza (ya sean científicos, sociales, etc.).
- Congresos Culturales: son aquellos que suelen tratar disciplinas de la cultura y las letras.
- Congresos Artísticos: pudiendo quedar englobados en los congresos culturales, son aquellos que tratan alrededor de las disciplinas de las artes y las letras.
- Congresos Tecnológicos: de forma genérica, tratan de los avances y novedades que se dan en los campos de la tecnología, en cualquiera de sus especialidades: medicina, informática, etc.

### *5.2.1.2.3 Según la finalidad del congreso*

En estos congresos por lo general tratan problemas a solucionar.

- **Congresos sociales:** son aquellos cuya finalidad es transmitir a la sociedad algún tipo de avance o mejora para la misma, en cualquiera de los campos de lo social.
- **Congresos privados:** la finalidad de este tipo de congresos es un interés particular por mejorar aspectos privados, generalmente de un colectivo, y que no suele tener una relevancia para la sociedad. Son los congresos de empresa, que tratan de mejorar su fuerza de ventas, por ejemplo.
- **Congresos generales o mixtos:** en este tipo de congresos, partiendo de aspectos privados o no, se buscan soluciones o mejoras que pueden revertir mejoras tanto en el campo privado como en el social (Infante, Arrieta, & Cruz, 2011).

## **5.2.2 Ponencia**

Una ponencia es un texto que se utiliza básicamente para ser expuesto por su autor y ser escuchado, en lugar de leído, por el destinatario final. Se utiliza en su mayoría para presentarse en algún evento científico, seminario congreso, simposio etc. Suelen ser trabajos breves, que se destinan a la discusión colectiva. En cuanto a su organización interna, no son tan decisivos los aspectos de forma, es usual que tenga un límite de entre diez y treinta hojas escritas a doble espacio. Asimismo, es un texto que se presta para ser escrito en primera persona, todo esto con el fin de facilitarle al lector la comunicación oral, que es un aspecto de suma importancia en una ponencia (Centro de escritura Javeriano, 2009).

### *5.2.2.1 Características*

Las características de las ponencias son las siguientes:

- Análisis del tema objeto de estudio.
- Es un trabajo breve pero no por ello falto de exhaustividad.
- Se deben tener en cuenta aspectos que faciliten su lectura y, por ende, la comunicación con el auditorio que la escucha.

### *5.2.2.2 Estructura*

Las ponencias manejan una estructura que se explica a continuación:

- **Resumen:** Esta sección se desarrolla para indicar claramente la temática de la investigación. Se debe tener en cuenta que tiene un rango establecido, no se deben escribir más de 15 renglones, que corresponden más o menos a unas 200 a 500 palabras. Igualmente, en esta sección no se hacen referencias bibliográficas.
- **Introducción:** debe tener información que llame la atención de la audiencia, puede ser una cita. Ésta debe anunciar claramente los objetivos del texto, así como también la secuencia cronológica de los temas que se realizarán más adelante en el desarrollo de la ponencia. Es muy importante que la introducción vaya de lo general a lo específico. Puede también incluir algunos datos relevantes que capten la atención de la audiencia, que cuestionen el tema presentado, que revelen una estadística o que relaten un momento revelador de la investigación. La extensión de la introducción dependerá de la importancia del trabajo.
- **Desarrollo o cuerpo:** debe seguir la misma línea planteada en la introducción, es decir, seguir la secuencia cronológica de los temas de la investigación. Es importante también seguir manteniendo el interés de la audiencia, para lo que se pueden utilizar ejemplos, citas interesantes, generalizaciones, etc. En esta sección, no se recomienda hacer muchas enumeraciones, pues debemos tener en cuenta que es un texto oral. Por otro lado, si la investigación amerita otras fuentes, como lo son las ayudas de PowerPoint, es recomendable reproducir pocas diapositivas, (según la extensión de la ponencia) y colocar poco texto. Asimismo, no se recomienda utilizar notas de pie de página en la ponencia, ya que el lector puede confundirse fácilmente a la hora de escucharla.
- **Conclusión:** a la hora de escribir la conclusión, es importante darle a entender a la audiencia que llegamos a ella. Es decir, en esta sección se debe sintetizar todo lo mencionado anteriormente, resaltando la importancia del tema y/o investigación. Una forma de sintetizar, podría ser, incluir frases breves y concisas que extracten lo expuesto y nombrar las aplicaciones prácticas que tiene su trabajo. La conclusión, debe además corresponder con los objetivos o preguntas mencionadas al inicio en la introducción.
- **Referencias bibliográficas:** se recomienda que las referencias bibliográficas estén organizadas en orden alfabético. Otra forma puede ser nombrarlas de acuerdo al orden en que han venido apareciendo en la ponencia. Estas referencias se deben regir según las normas APA y/o ICONTEC (Centro de escritura Javeriano, 2009).

### **5.2.3 Seminario**

Del latín *seminariŭs*, es una clase o encuentro didáctico donde un especialista interactúa con los asistentes en trabajos en común para difundir conocimientos o desarrollar investigaciones (Definicion.de, 2015).

El seminario es una reunión especializada, de naturaleza técnica o académica, que intenta desarrollar un estudio profundo sobre una determinada materia. Por lo general, se establece que un seminario debe tener una duración mínima de dos horas y contar con, al menos, cincuenta participantes (Definicion.de, 2015).

#### *5.2.3.1 Características*

Los seminarios cuentan con algunas características importantes. A continuación se mencionan algunas de ellas:

- Los miembros tienen intereses comunes en cuanto al tema, y un nivel semejante de información acerca del mismo.
- El tema o material exige la investigación o búsqueda específica en diversas fuentes. Un tema ya elaborado o expuesto en un libro no justifica el trabajo de seminario.
- El desarrollo de las tareas, así como los temas y subtemas por tratarse son planificados por todos los miembros en la primera sesión de grupo.
- Los resultados o conclusiones son responsabilidad de todo el grupo.
- Todo seminario concluye con una sesión de resumen y evaluación del trabajo realizado.
- El seminario puede trabajar durante varios días hasta dar por terminada su labor. Las sesiones suelen durar dos o tres horas.

### **5.2.4 Taller**

Proviene del francés *atelier* y hace referencia al lugar en que se trabaja principalmente con las manos. En el campo de la educación, se habla de talleres para referirse a una cierta metodología de enseñanza que combina la teoría y la práctica. Los talleres permiten el desarrollo de investigaciones y el trabajo en equipo. Algunos son permanentes dentro de un cierto nivel educativo mientras que otros pueden durar uno o varios días y no estar vinculados a un sistema específico (Definicion.de, 2015).

Es un evento donde el profesor o instructor transmite la información la cual debe estar relacionada con lo que el participante realiza habitualmente por medio de asesorías y por medio de varias técnicas de aprendizaje., hecho o problema, conducido por un coordinador.

# Capítulo 6 DESARROLLO DEL SISTEMA

## 6.1 Análisis de requerimientos

Un requerimiento es una característica que el sistema debe tener o es una restricción que el sistema debe satisfacer para ser aceptada por el cliente (Grady Booch, 1999).

Los requerimientos se dividen en dos grupos; los requerimientos funcionales y los no funcionales.

### 6.1.1 Requerimientos funcionales

Los requerimientos funcionales Describen la interacción entre el sistema y su ambiente independientemente de su implementación. El ambiente incluye al usuario y cualquier otro sistema externo que interactúa con el sistema. (Grady Booch, 1999).

Los requerimientos funcionales se clasifican en requerimientos de estructura y requerimientos del prototipo, éstos se explican a continuación:

- Requerimientos funcionales de estructura: por medio de métodos acelerados, se debe elaborar o actualizar uno o más modelos de sistemas para ilustrar el requerimiento funcional. Pueden incluir cualquier combinación de datos, procesos y modelos de objetos que describan con precisión el negocio y los requerimientos de los usuarios. Los modelos de sistemas no están completos hasta que todos los requerimientos funcionales proporcionados han sido modelados. Los modelos con frecuencia se complementan con especificaciones lógicas detalladas que describen los atributos de datos, reglas y políticas de negocio (Bentley, 2008).
- Requerimientos funcionales del prototipo (alternativa): la elaboración de prototipos es una alternativa (y a veces un requerimiento previo) para la elaboración de sistemas. Una alternativa complementaria a la elaboración de modelos de sistemas es construir prototipos de identificación. La elaboración de prototipos normalmente se utiliza en la fase de análisis de requerimientos para construir entradas y salidas de ejemplos. Estas entradas y salidas

ayudan a construir la base de datos y los programas para introducir y generar datos hacia y desde la base de datos (Bentley, 2008).

- Validación de requerimientos funcionales: tanto los modelos de sistemas como los prototipos son representaciones de los requerimientos de los usuarios. Éstos deben ser validados para que estén completos y correctos. Los analistas de sistemas facilitan la tarea de asignación de prioridades al comprometer en forma interactiva a los usuarios del sistema para la identificación de errores, omisiones y aclaraciones (Bentley, 2008).
- Definir casos de prueba de aclaración: los modelos y prototipos de sistemas definen muy eficazmente los requerimientos de proceso, reglas de datos y reglas de negocio para el nuevo sistema. En consecuencia, estas especificaciones pueden ser utilizadas para definir casos de prueba que pueden finalmente ser utilizados para probar que los programas estén correctos (Bentley, 2008).

### **6.1.2 Requerimientos no funcionales**

Los requerimientos no funcionales describen aspectos del sistema que son visibles por el usuario que no incluyen una relación directa con el comportamiento funcional del sistema. Los requerimientos no funcionales incluyen restricciones como el tiempo de respuesta (desempeño), la precisión, recursos consumidos, seguridad, etc. (H.Dutoit, 2000).

### **6.1.3 Identificación de requerimientos**

Para desarrollar un sistema debemos ser capaces de identificar correctamente, analizar y entender cuáles son los requerimientos de los usuarios o lo que los usuarios quieren que haga el sistema. El proceso y las técnicas que un analista de sistemas usa para identificar, analizar y entender requerimientos de sistema, son referidos como la identificación de requerimientos. La identificación de requerimientos involucra principalmente a los analistas de sistemas que trabajan con los usuarios de sistemas y con los propietarios durante las primeras fases de desarrollo del sistema, con el fin de obtener una comprensión detallada de los requerimientos del negocio de un sistema de información. Los requerimientos del sistema especifican lo que el sistema de información deberá hacer o cuál propiedad o cualidad debe de tener éste (Bentley, 2008).

### *6.1.3.1 Proceso de identificación de requerimientos*

El proceso de identificación de requerimientos consiste de las siguientes actividades:

- Identificación del problema y análisis
- Identificación de los requerimientos
- Requerimientos de documentación y de análisis: éste incluye documentación de los requerimientos del borrador, análisis de los requerimientos y formalización de los requerimientos.
- Administración de los requerimientos

### *6.1.3.2 Técnicas de exploración*

A continuación se muestran siete técnicas de exploración:

- Muestreo de la documentación, los formatos y los archivos existentes
- Investigación y visitas al sitio
- Observación del ambiente de trabajo
- Cuestionarios
- Entrevistas
- Elaboración de prototipos de identificación

Generalmente un analista aplica varias de estas técnicas durante un proyecto de sistemas individual. Para poder seleccionar la técnica más adecuada, los analistas de sistemas necesitan aprender las ventajas y desventajas de cada una de las técnicas de exploración (Bentley, 2008).

## **6.2 Requerimientos del Sistema de Inscripción**

Para obtener los requerimientos del sistema a elaborar se realizaron unos cuestionarios, para organizadores del evento, que al mismo tiempo serían los administradores del sistema, y para los alumnos que participan en dichos eventos. Los requerimientos obtenidos son los siguientes:

### **6.2.1 Requerimientos Funcionales**

- Se darán a conocer los temas que serán expuestos en cada ponencia y taller, para saber las actividades a realizar, de esta forma, las personas que asistan tendrán una idea del contenido de éstas, y así identificar si el tema es interesante para ellos.
- Para cada actividad a realizar, se dará a conocer el número de participantes que pueden ingresar, es decir, verificar disponibilidad de lugares.
- Si más de una ponencia es impartida en una misma sala, se tomará asistencia cada vez que inicie una nueva ponencia.
- Se darán a conocer los diferentes paquetes para el evento, que ponencias y talleres incluye, los horarios, el lugar en que se impartirán, y el costo que tendrá cada uno.
- El tiempo en que se generará la documentación debe ser en un plazo que no exceda las tres semanas después de finalizar el evento, preferiblemente en un plazo no mayor a diez días, para comunicar a las personas involucradas la fecha aproximada en que tendrán su documentación.
- Para realizar la inscripción, el alumno puede realizar una preinscripción. Para completar su inscripción, deben acudir con el administrador para generar su gafete.
- El tipo de pago será en efectivo en una sola emisión, además se entregará al asistente un gafete, que también servirá como comprobante de pago, este gafete incluye el código de barras, el cual será usado para registrar asistencia.
- Para generar las constancias, se determinará el número de ponencias mínimas a las que se debe asistir, es decir, si no se asiste a cierto número de ponencias, no se entregará constancia de asistencia. Para los agradecimientos de los ponentes, con registrar su asistencia a la actividad que imparten será entregado su documento.

### **6.2.2 Requerimientos no funcionales**

El sistema contará con una interfaz entendible para todos los usuarios. Ningún alumno deberá tener problemas al interactuar con el sistema para realizar su preinscripción, ni el administrador al completar la inscripción.

### 6.3 Diagramas de casos de uso

#### 6.3.1 Diagrama de caso de uso Inicializa evento



Imagen 21. Diagrama de caso de uso Inicializa evento

Tabla 3. Diagrama de caso de uso Inicializa evento

**Caso de uso:** **Inicializa evento**

<b>Actores:</b>	Organizador
<b>Propósito</b>	Iniciar el evento.
<b>Resumen</b>	El organizador proporciona información del evento que iniciará.
<b>Precondiciones</b>	<ul style="list-style-type: none"><li>• El organizador debe tener una clave de acceso al sistema.</li></ul>
<b>Flujo Principal</b>	<ol style="list-style-type: none"><li>1. El organizador dará de alta sus datos en la base de datos para administrar el sistema.</li></ol>
<b>Excepciones</b>	<ul style="list-style-type: none"><li>• No hay información del evento.</li></ul>

#### 6.3.2 Diagrama de caso de uso Registra actividades



Imagen 22. Diagrama de caso de uso Registra actividades

Tabla 4. Diagrama de caso de uso Registra actividades

Caso de uso: <b>Registra actividades</b>	
<b>Actores:</b>	Organizador
<b>Propósito</b>	Registrar las actividades a realizar y los horarios de éstas.
<b>Resumen</b>	El organizador da de alta las actividades (ponencia o taller) del evento, los paquetes que incluye y los temas de exposición.
<b>Precondiciones</b>	<ul style="list-style-type: none"> <li>• El organizador debe tener una clave de acceso al sistema.</li> <li>• El ponente debe estar registrado.</li> </ul>
<b>Flujo Principal</b>	<ol style="list-style-type: none"> <li>1. El organizador ingresará al sistema.</li> <li>2. Registrará las actividades (tema, hora, fecha, lugar y el tipo, es decir, si es ponencia o taller).</li> <li>3. Registrará los paquetes (folio, costo, temas de ponencias y temas de talleres)</li> </ol>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>• Clave del organizador incorrecta.</li> <li>• Error de conexión con la base de datos.</li> </ul>

### 6.3.3 Diagrama de caso de uso Registra datos de ponente

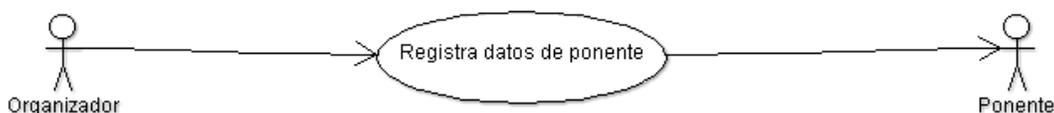


Imagen 23. Diagrama de caso de uso Registra datos de ponente

Tabla 5. Diagrama de caso de uso Registra datos de ponente

Caso de uso: <b>Registra datos de ponente</b>	
<b>Actores:</b>	Organizador, Ponente
<b>Propósito</b>	Registrar a los ponentes que asistirán al evento.
<b>Resumen</b>	El organizador da de alta la información proporcionada por el ponente.
<b>Precondiciones</b>	<ul style="list-style-type: none"> <li>• El organizador debe tener una clave de acceso al sistema.</li> <li>• El ponente debe confirmar su asistencia.</li> </ul>

<b>Flujo Principal</b>	<ol style="list-style-type: none"> <li>1. El ponente envía sus datos al organizador (RFC, nombre, apellidos, grado académico, una breve descripción de su CV y el tema de exposición).</li> <li>2. El organizador registra en el sistema los datos del ponente.</li> </ol>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>• Clave del organizador incorrecta.</li> <li>• Error de conexión con la base de datos.</li> </ul>

### 6.3.4 Diagrama de caso de uso Realiza preinscripción

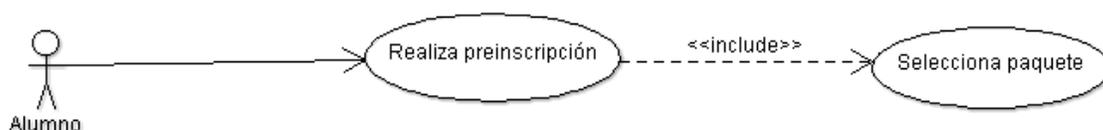


Imagen 24. Diagrama de caso de uso Realiza preinscripción

Tabla 6. Diagrama de caso de uso Realiza preinscripción

#### Caso de uso: **Realiza preinscripción**

<b>Actores:</b>	Alumno
<b>Propósito</b>	El alumno realiza una preinscripción.
<b>Resumen</b>	El alumno ingresa sus datos y selecciona el paquete que pagará.
<b>Precondiciones</b>	<ul style="list-style-type: none"> <li>• Inicio del evento</li> <li>• Activación de paquetes completo.</li> <li>• Registro de actividades completo.</li> </ul>
<b>Flujo Principal</b>	<ol style="list-style-type: none"> <li>1. El alumno ingresa al sistema.</li> <li>2. Ingresa a preinscripción de alumnos.</li> <li>3. Ingresa sus datos (número de cuenta, nombre, apellidos, semestre y licenciatura).</li> <li>4. Consulta los paquetes y actividades del evento.</li> <li>5. Selecciona el paquete que va a pagar.</li> </ol>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>• Error de conexión con la base de datos.</li> </ul>

### 6.3.5 Diagrama de caso de uso Completa inscripción de alumno

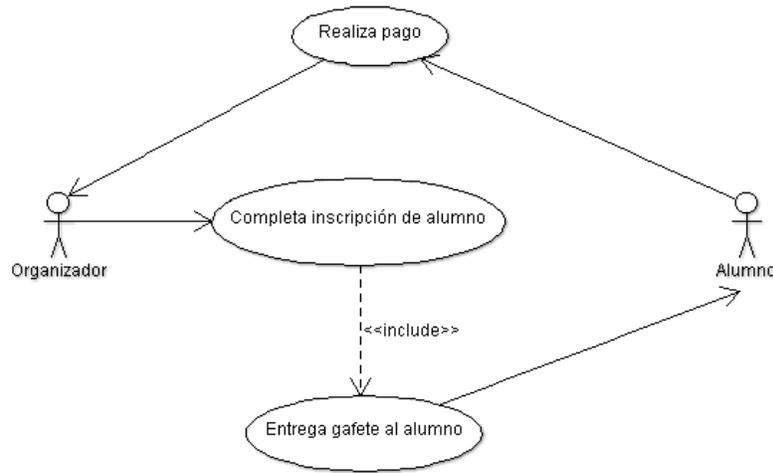


Imagen 25. Diagrama de caso de uso Completa inscripción de alumno

Tabla 7. Diagrama de caso de uso Completa inscripción de alumno

**Caso de uso:** **Completa inscripción del alumno**

<b>Actores:</b>	Organizador, Alumno
<b>Propósito</b>	El organizador completa la inscripción del alumno.
<b>Resumen</b>	El organizador realiza una consulta con el número de cuenta del alumno para verificar su preinscripción, de existir, entrega un gafete con código de barras al alumno y a cambio recibe el pago en efectivo.
<b>Precondiciones</b>	<ul style="list-style-type: none"> <li>• El organizador debe tener una clave de acceso al sistema.</li> <li>• El alumno debe estar preinscrito.</li> </ul>
<b>Flujo Principal</b>	<ol style="list-style-type: none"> <li>1. El organizador entra al sistema.</li> <li>2. Ingresa a inscripción de alumnos.</li> <li>3. Realiza consulta con el número de cuenta del alumno.</li> <li>4. Genera gafete con código de barras.</li> <li>5. al alumno realiza pago en efectivo al organizador.</li> </ol>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>• Clave del organizador incorrecta.</li> <li>• Error de conexión con la base de datos.</li> <li>• El alumno no está preinscrito.</li> </ul>

### 6.3.6 Diagrama de caso de uso Registra asistencia

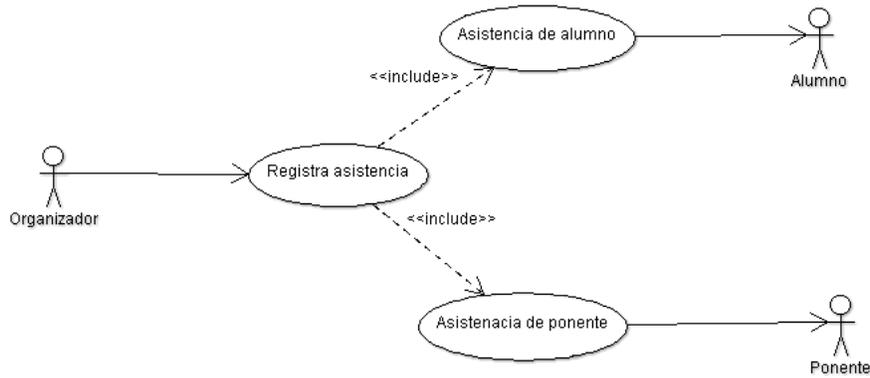


Imagen 26. Diagrama de caso de uso Registra asistencia

Tabla 8. Diagrama de caso de uso Registra asistencia

**Caso de uso:** **Genera documentación**

<b>Actores:</b>	Organizador, Alumno, Ponente
<b>Propósito</b>	Registrar la asistencia de alumnos y de ponentes.
<b>Resumen</b>	El organizador registrará la asistencia de alumnos por medio de su gafete, y la de ponentes, ingresando su RFC.
<b>Precondiciones</b>	<ul style="list-style-type: none"> <li>• El organizador debe tener una clave de acceso al sistema.</li> <li>• El alumno está inscrito.</li> <li>• El ponente está inscrito.</li> </ul>
<b>Flujo Principal</b>	<ol style="list-style-type: none"> <li>1. El alumno asiste al evento.</li> <li>2. El organizador registra la asistencia del alumno escaneando su gafete.</li> <li>3. Realiza una consulta para verificar que el alumno está inscrito.</li> <li>4. El ponente asiste al evento.</li> <li>5. El organizador registra la asistencia del ponente ingresando su RFC.</li> <li>6. Realiza una consulta para verificar que el ponente está inscrito.</li> </ol>
<b>Excepciones</b>	<ul style="list-style-type: none"> <li>• Error de conexión con la base de datos.</li> <li>• El alumno no está inscrito.</li> <li>• El ponente no está inscrito.</li> </ul>

### 6.3.7 Diagrama de caso de uso Genera documentación

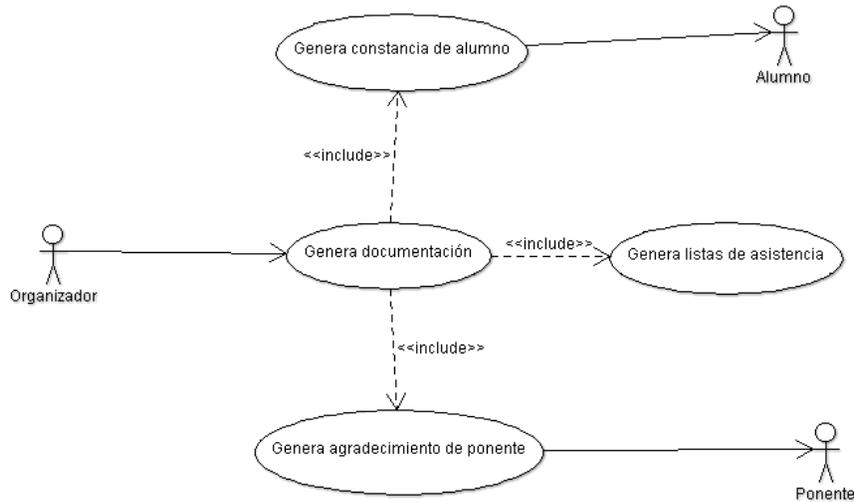


Imagen 27. Diagrama de caso de uso Genera documentación

Tabla 9. Diagrama de caso de uso Genera documentación

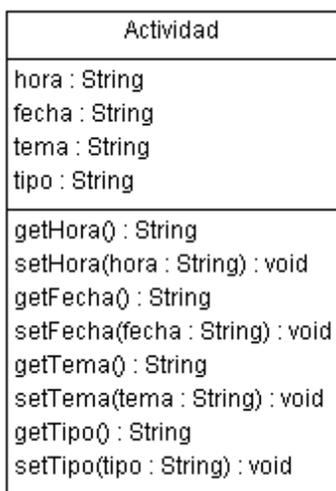
**Caso de uso: Genera documentación**

<b>Actores:</b>	Organizador, Alumno, Ponente
<b>Propósito</b>	Generar la documentación del evento (constancias de asistencia, agradecimientos y listas de asistencia).
<b>Resumen</b>	El organizador generará la documentación del evento basado en la asistencia tanto de los alumnos como de los ponentes.
<b>Precondiciones</b>	<ul style="list-style-type: none"> <li>• El organizador debe tener una clave de acceso al sistema.</li> <li>• El alumno asistió al evento.</li> <li>• El ponente asistió al evento.</li> <li>• Fin del evento.</li> </ul>
<b>Flujo Principal</b>	<ol style="list-style-type: none"> <li>1. El organizador realiza una consulta con el número de cuenta del alumno para saber si asistió al evento y generar su constancia; también ingresa datos (folio de constancia, fecha de generación, fecha de entrega, RFC del organizador que genera y el número de cuenta del alumno) para tener control de entrega.</li> <li>2. El organizador realiza una consulta con el RFC del ponente para saber si asistió al evento y generar su agradecimiento; también ingresa datos (folio de agradecimiento, fecha de generación, fecha de entrega, RFC del organizador que genera y RFC del ponente) para tener control de entrega.</li> <li>3. El organizador genera las listas de asistencia de los alumnos.</li> <li>4. El alumno recibe su constancia de asistencia al evento.</li> <li>5. El ponente recibe su agradecimiento por participar en el evento.</li> </ol>

<b>Excepciones</b>	<ul style="list-style-type: none"> <li>• Error de conexión con la base de datos.</li> <li>• El alumno no asistió al evento.</li> <li>• El ponente no asistió al evento.</li> </ul>
--------------------	--

## 6.4 Diagramas de clases

### 6.4.1 Diagrama de clase Actividad



*Imagen 28. Diagrama de clase Actividad*

*Tabla 10. Diagrama de clase Actividad*

Elemento	Nombre	Descripción
<b>Clase</b>	Actividad	Se ingresan los datos de la actividad
<b>Atributo</b>	hora : String	Obtiene la hora de inicio de la actividad
<b>Atributo</b>	fecha : String	Obtiene la fecha de la actividad
<b>Atributo</b>	tema : String	Obtiene el tema de la actividad
<b>Atributo</b>	tipo : String	Obtiene el tipo de la actividad
<b>Operación</b>	getHora() : String	Muestra el atributo hora
<b>Operación</b>	setHora(hora : String) : void	Modifica el atributo hora
<b>Operación</b>	getFecha() : String	Muestra el atributo fecha
<b>Operación</b>	SetFecha(fecha : String) : void	Modifica el atributo fecha
<b>Operación</b>	getTema() : String	Muestra el atributo tema

<b>Operación</b>	setTema(tema : String) : void	Modifica el atributo tema
<b>Operación</b>	getTipo() : String	Muestra el atributo tipo
<b>Operación</b>	setTipo(tipo : String) : void	Modifica el atributo tipo
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

## 6.4.2 Diagrama de clase Administrador

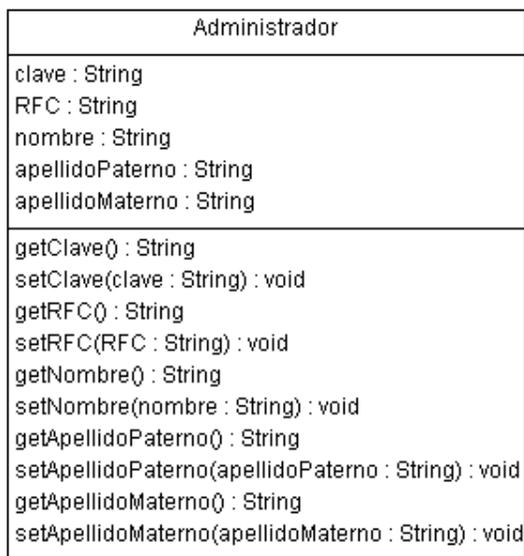


Imagen 29. Diagrama de clase Administrador

Tabla 11. Diagrama de clase Administrador

Elemento	Nombre	Descripción
<b>Clase</b>	Administrador	Se ingresan los datos del administrador (organizador)
<b>Atributo</b>	clave : String	Obtiene la clave de inicio del administrador
<b>Atributo</b>	RFC : String	Obtiene el RFC del administrador
<b>Atributo</b>	nombre : String	Obtiene el nombre del administrador

<b>Atributo</b>	apellidoPaterno : String	Obtiene el apellido paterno del administrador
<b>Atributo</b>	apellidoMaterno : String	Obtiene el apellido materno del administrador
<b>Operación</b>	getClave() : String	Muestra el atributo clave
<b>Operación</b>	setClave(hora : String) : void	Modifica el atributo clave
<b>Operación</b>	getRFC() : String	Muestra el atributo RFC
<b>Operación</b>	SetRFC(RFC : String) : void	Modifica el atributo RFC
<b>Operación</b>	getNombre() : String	Muestra el atributo nombre
<b>Operación</b>	setNombre(nombre : String) : void	Modifica el atributo nombre
<b>Operación</b>	getApellidoPaterno() : String	Muestra el atributo apellidoPaterno
<b>Operación</b>	setApellidoPaterno (aApellidoPaterno : String) : void	Modifica el atributo apellidoPaterno
<b>Operación</b>	getApellidoMaterno() : String	Muestra el atributo apellidoMaterno
<b>Operación</b>	setApellidoMaterno(apellidoMaterno : String) : void	Modifica el atributo apellidoMaterno
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

### 6.4.3 Diagrama de clase Alumno

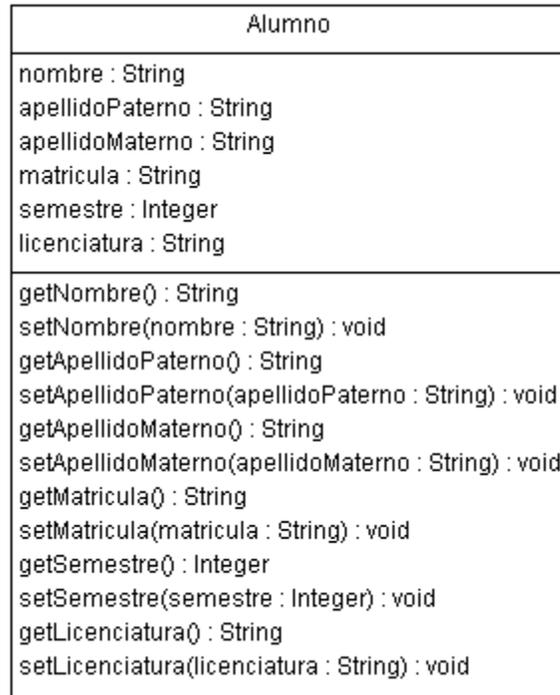


Imagen 30. Diagrama de clase Alumno

Tabla 12. Diagrama de clase Alumno

Elemento	Nombre	Descripción
<b>Clase</b>	Alumno	Se ingresan los datos del alumno
<b>Atributo</b>	nombre : String	Obtiene el nombre del alumno
<b>Atributo</b>	apellidoPaterno : String	Obtiene el apellido paterno del alumno
<b>Atributo</b>	apellidoMaterno : String	Obtiene el apellido materno del alumno
<b>Atributo</b>	matricula : String	Obtiene la matrícula (número de cuenta) del alumno
<b>Atributo</b>	semestre : Integer	Obtiene el semestre del alumno
<b>Atributo</b>	licenciatura : String	Obtiene la licenciatura del alumno
<b>Operación</b>	getNombre() : String	Muestra el atributo nombre
<b>Operación</b>	setNombre(nombre : String) : void	Modifica el atributo nombre

<b>Operación</b>	getApellidoPaterno() : String	Muestra el atributo apellidoPaterno
<b>Operación</b>	setApellidoPaterno (aApellidoPaterno : String) : void	Modifica el atributo apellidoPaterno
<b>Operación</b>	getApellidoMaterno() : String	Muestra el atributo apellidoMaterno
<b>Operación</b>	setApellidoMaterno(apellidoMaterno : String) : void	Modifica el atributo apellidoMaterno
<b>Operación</b>	getMatricula() : String	Muestra el atributo matricula
<b>Operación</b>	setMatricula(matricula : String) : void	Modifica el atributo maticula
<b>Operación</b>	getSemestre() : Integer	Muestra el atributo semestre
<b>Operación</b>	setSemestre (semestre : Integer) : void	Modifica el atributo semestre
<b>Operación</b>	getLicenciatura() : String	Muestra el atributo licenciatura
<b>Operación</b>	setLicenciatura (licenciatura : String) : void	Modifica el atributo licenciatura
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

#### 6.4.4 Diagrama de clase Paquete



Imagen 31. Diagrama de clase Paquete

Tabla 13. Diagrama de clase Paquete

<b>Elemento</b>	<b>Nombre</b>	<b>Descripción</b>
<b>Clase</b>	Paquete	Se ingresan los datos del paquete
<b>Atributo</b>	folio : String	Obtiene el folio de inicio del paquete
<b>Atributo</b>	costo : Float	Obtiene el costo del paquete
<b>Atributo</b>	ponencia : String	Obtiene las ponencias del paquete
<b>Atributo</b>	taller : String	Obtiene los talleres del paquete
<b>Operación</b>	getFolio() : String	Muestra el atributo folio
<b>Operación</b>	setFolio (folio : String) : void	Modifica el atributo folio
<b>Operación</b>	getCosto() : Float	Muestra el atributo costo
<b>Operación</b>	SetCosto(costo : Float) : void	Modifica el atributo costo
<b>Operación</b>	getPonencia() : String	Muestra el atributo ponencia
<b>Operación</b>	setPonencia(ponencia : String) : void	Modifica el atributo ponencia
<b>Operación</b>	getTaller() : String	Muestra el atributo taller
<b>Operación</b>	setTaller (taller : String) : void	Modifica el atributo taller
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

## 6.4.5 Diagrama de clase Ponente

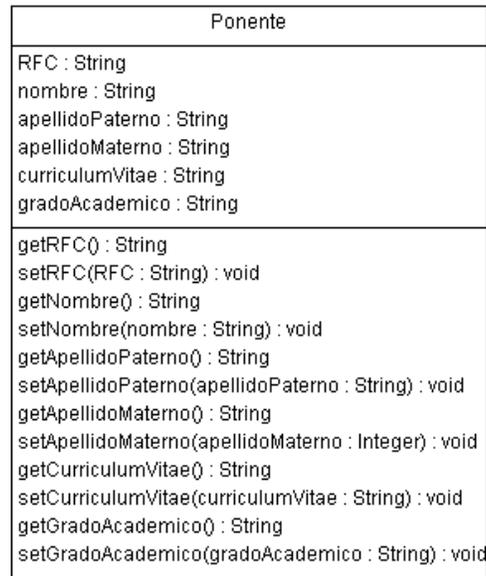


Imagen 32. Diagrama de clase Ponente

Tabla 14. Diagrama de clase Ponente

Elemento	Nombre	Descripción
<b>Clase</b>	Ponente	Se ingresan los datos del ponente
<b>Atributo</b>	RFC : String	Obtiene el RFC del ponente
<b>Atributo</b>	nombre : String	Obtiene el nombre del ponente
<b>Atributo</b>	apellidoPaterno : String	Obtiene el apellido paterno del ponente
<b>Atributo</b>	apellidoMaterno : String	Obtiene el apellido materno del ponente
<b>Atributo</b>	curriculumVitae : String	Obtiene el CV del ponente
<b>Atributo</b>	gradoAcademico : String	Obtiene el grado académico del ponente

<b>Atributo</b>	temaExposicion : String	Obtiene el tema de exposición del ponente
<b>Operación</b>	getRFC() : String	Muestra el atributo RFC
<b>Operación</b>	setRFC (RFC : String) : void	Modifica el atributo RFC
<b>Operación</b>	getNombre() : String	Muestra el atributo nombre
<b>Operación</b>	setNombre(nombre : String) : void	Modifica el atributo nombre
<b>Operación</b>	getApellidoPaterno() : String	Muestra el atributo apellidoPaterno
<b>Operación</b>	setApellidoPaterno (aApellidoPaterno : String) : void	Modifica el atributo apellidoPaterno
<b>Operación</b>	getApellidoMaterno() : String	Muestra el atributo apellidoMaterno
<b>Operación</b>	setApellidoMaterno(apellidoMaterno : String) : void	Modifica el atributo apellidoMaterno
<b>Operación</b>	getCurriculumVitae() : String	Muestra el atributo curriculumVitae
<b>Operación</b>	setCurriculumVitae (curriculumVitae : String) : void	Modifica el atributo curriculumVitae
<b>Operación</b>	getGradoAcademico () : String	Muestra el atributo gradoAcademico
<b>Operación</b>	setGradoAcademico (gradoAcademico : String) : void	Modifica el atributo gradoAcademico
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

## 6.4.6 Diagrama de clase Agradecimiento



Imagen 33. Diagrama de clase Agradecimiento

Tabla 15. Diagrama de clase Agradecimiento

Elemento	Nombre	Descripción
<b>Clase</b>	Agradecimiento	Se crea el agradecimiento para los ponentes
<b>Atributo</b>	cn : Connection	Hace la conexión con la base de datos
<b>Operación</b>	mostrarReporte (titulo : String) : void	Llama al reporte creado para agradecimientos
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

## 6.4.7 Diagrama de clase ConstAlumno

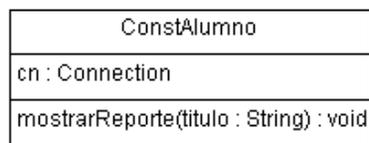


Imagen 34. Diagrama de clase ConstAlumno

Tabla 16. Diagrama de clase ConstAlumno

Elemento	Nombre	Descripción
<b>Clase</b>	ConstAlumno	Se crea la constancia para los alumnos
<b>Atributo</b>	cn : Connection	Hace la conexión con la base de datos
<b>Operación</b>	mostrarReporte (titulo : String) : void	Llama al reporte creado para constancias
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

## 6.4.8 Diagrama de clase ReporteAsis

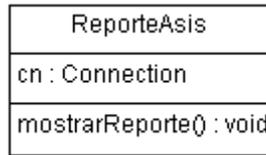


Imagen 35. Diagrama de clase ReporteAsis

Tabla 17. Diagrama de clase ReporteAsis

Elemento	Nombre	Descripción
<b>Clase</b>	ReporteAsis	Se crea la lista de asistencia de los alumnos
<b>Atributo</b>	cn : Connection	Hace la conexión con la base de datos
<b>Operación</b>	mostrarReporte () : void	Llama al reporte creado para la lista de asistencia
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

## 6.4.9 Diagrama de clase PrintMe

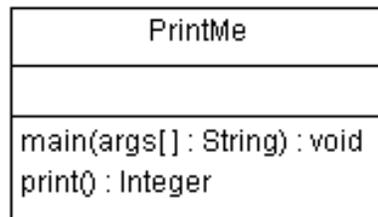


Imagen 36. Diagrama de clase PrintMe

Tabla 18. Diagrama de clase PrintMe

Elemento	Nombre	Descripción
<b>Clase</b>	PrintMe	Permite imprimir a la interfaz que la invoque
<b>Operación</b>	main (args[] : String) : void	Función principal de la clase. Permite que se ejecute
<b>Operación</b>	print() : Integer	Imprime la interfaz que la invoque
<b>Interfaz</b>	No existen Interfaz en esta clase	No existen Interfaz en esta clase

### 6.4.10 Diagrama de clase Portada

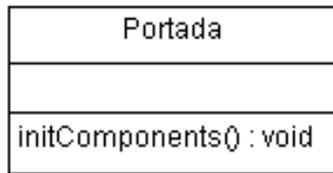


Imagen 37. Diagrama de clase Portada

Tabla 19. Diagrama de clase Portada

Elemento	Nombre	Descripción
<b>Clase</b>	Portada	Esta ventana muestra información de la convocatoria del evento
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Interfaz</b>	Constancias	Está compuesta por etiquetas

### 6.4.11 Diagrama de clase Inicio

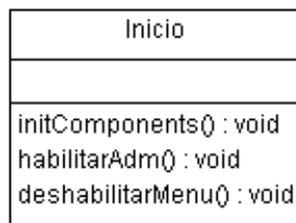


Imagen 38. Diagrama de clase Inicio

Tabla 20. Diagrama de clase Inicio

Elemento	Nombre	Descripción
<b>Clase</b>	Inicio	Es la ventana principal del sistema. Se puede acceder como alumno o como administrador
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto

<b>Operación</b>	habilitarAdm() : void	Habilita los botones a los que el administrador tiene acceso
<b>Operación</b>	deshabilitaMenu() : void	Deshabilita el menú cuando se accesa al sistema
<b>Interfaz</b>	Inicio	Está compuesta por botones, una barra de menú y un panel

#### 6.4.12 Diagrama de clase Preins

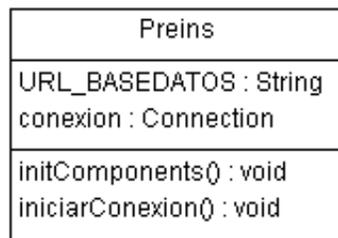


Imagen 39. Diagrama de clase Preins

Tabla 21. Diagrama de clase Preins

<b>Elemento</b>	<b>Nombre</b>	<b>Descripción</b>
<b>Clase</b>	Preins	Los alumnos realizan la preinscripción al evento introduciendo sus datos
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Atributo</b>	conexion : Connection	Hace la conexión con la base de datos
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Interfaz</b>	Preins	Está compuesta por botones, cajas de texto y etiquetas

### 6.4.13 Diagrama de clase InformAct

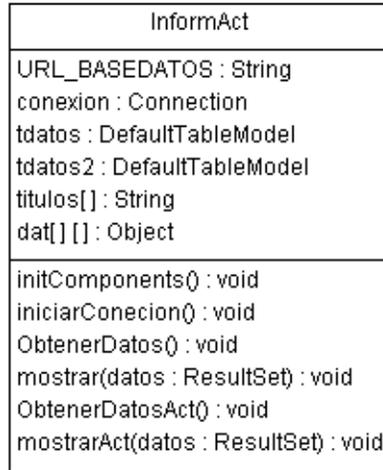


Imagen 40. Diagrama de clase InformAct

Tabla 22. Diagrama de clase InformAct

Elemento	Nombre	Descripción
<b>Clase</b>	InformAct	Los alumnos revisan la información de las actividades del evento y de los paquetes
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Atributo</b>	conexion : Connection	Hace la conexión con la base de datos
<b>Atributo</b>	tdatos : DefaultTableModel	Se crea un objeto tabla
<b>Atributo</b>	tdatos2 : DefaultTableModel	Se crea un segundo objeto tabla
<b>Atributo</b>	titulos[] : String	Obtiene los títulos de la tabla a la que se realiza la consulta
<b>Atributo</b>	dat[][] : Object	Obtiene los datos de la tabla consultada
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Operación</b>	obtenerDatos() : void	Realiza una consulta y obtiene los datos de los paquetes

<b>Operación</b>	mostrar(datos : ResultSet) : void	Muestra la información de la consulta de los paquetes
<b>Operación</b>	obtenerDatosAct() : void	Realiza una consulta y obtiene los datos de las actividades
<b>Operación</b>	mostrarAct(datos : ResultSet) : void	Muestra la información de la consulta de las actividades
<b>Interfaz</b>	InformAct	Está compuesta por botones, tablas y etiquetas

#### 6.4.14 Diagrama de clase FrmInsAlumno

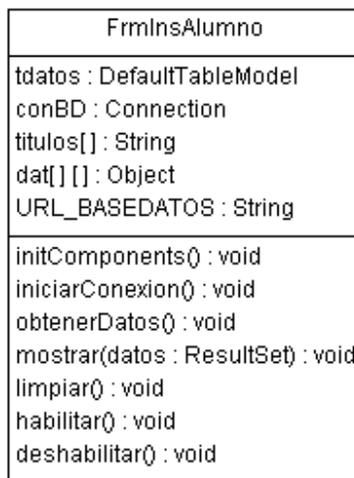


Imagen 41. Diagrama de clase FrmInsAlumno

Tabla 23. Diagrama de clase FrmInsAlumno

<b>Elemento</b>	<b>Nombre</b>	<b>Descripción</b>
<b>Clase</b>	FrmInsAlumno	El administrador completa la inscripción del alumno
<b>Atributo</b>	tdatos : DefaultTableModel	Se crea un objeto tabla
<b>Atributo</b>	conBD : Connection	Hace la conexión con la base de datos
<b>Atributo</b>	titulos[] : String	Obtiene los títulos de la tabla a la que se realiza la consulta
<b>Atributo</b>	dat[][] : Object	Obtiene los datos de la tabla consultada
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql

<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Operación</b>	obtenerDatos() : void	Realiza una consulta y obtiene los datos de la tabla alumno
<b>Operación</b>	mostrar(datos : ResultSet) : void	Muestra la información de la consulta de la tabla alumno
<b>Operación</b>	limpiar() : void	Limpia la tabla cuando se realiza una nueva consulta
<b>Operación</b>	habilitar() : void	Habilita los botones cuando se realiza la consulta
<b>Operación</b>	deshabilitar() : void	Deshabilita los botones cuando la consulta no tiene datos
<b>Interfaz</b>	InformAct	Está compuesta por botones, tablas y etiquetas

#### 6.4.15 Diagrama de clase FrCod

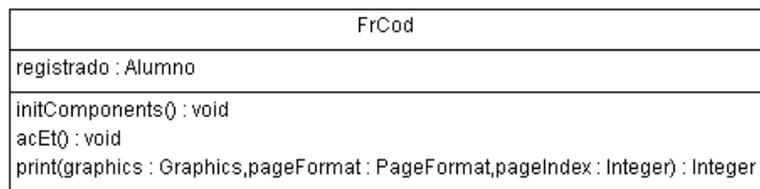


Imagen 42. Diagrama de clase FrCod

Tabla 24. Diagrama de clase FrCod

<b>Elemento</b>	<b>Nombre</b>	<b>Descripción</b>
<b>Clase</b>	FrCod	Se genera el gafete con código de barras para el alumno
<b>Atributo</b>	registrado : Alumno	Se guardan los datos del alumno obtenidos en la consulta de preinscripción
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	acEt () : void	Guarda los datos del alumno en el gafete y

		agrega "0000" al número de cuenta para poder generar el código de barras
<b>Operación</b>	print(graphics : Graphics, pageFormat : PageFormat, pageIndex : Integer)	Permite imprimir el gafete para el alumno
<b>Interfaz</b>	FrCod	Está compuesta por botones, y etiquetas

#### 6.4.16 Diagrama de clase RegisPon

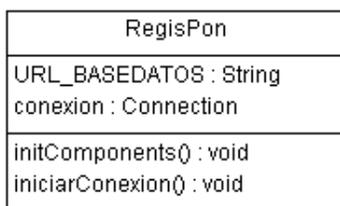


Imagen 43. Diagrama de clase RegisPon

Tabla 25. Diagrama de clase RegisPon

Elemento	Nombre	Descripción
<b>Clase</b>	RegisPon	Se registra a los ponentes
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Atributo</b>	conexion : Connection	Hace la conexión con la base de datos
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Interfaz</b>	RegisPon	Está compuesta por botones, cajas de texto, áreas de texto y etiquetas

### 6.4.17 Diagrama de clase ActivaPaquete

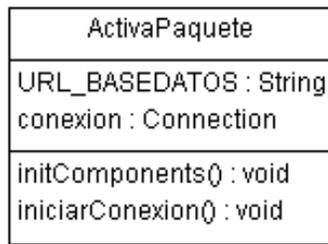


Imagen 44. Diagrama de clase ActivaPaquete

Tabla 26. Diagrama de clase ActivaPaquete

Elemento	Nombre	Descripción
<b>Clase</b>	ActivaPaquete	Se ingresa la información de los paquetes
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Atributo</b>	conexion : Connection	Hace la conexión con la base de datos
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Interfaz</b>	ActivaPaquete	Está compuesta por botones, cajas de texto, áreas de texto y etiquetas

### 6.4.18 Diagrama de clase RegAct

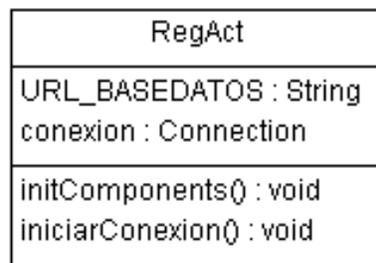


Imagen 45. Diagrama de clase RegAct

Tabla 27. Diagrama de clase RegAct

Elemento	Nombre	Descripción
<b>Clase</b>	RegAct	Se ingresa la información de las actividades a realizar (temas de ponencias y talleres)
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Atributo</b>	conexion : Connection	Hace la conexión con la base de datos
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Interfaz</b>	RegAct	Está compuesta por botones, cajas de texto, cajas de combo, dateChooserCombo (para seleccionar fecha) y etiquetas

#### 6.4.19 Diagrama de clase Asis

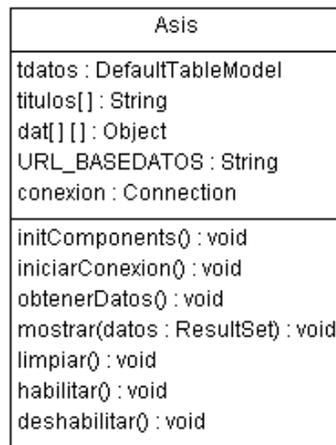


Imagen 46. Diagrama de clase Asis

Tabla 28. Diagrama de clase Asis

<b>Elemento</b>	<b>Nombre</b>	<b>Descripción</b>
<b>Clase</b>	Asis	El administrador registra la asistencia del alumno por medio de su gafete
<b>Atributo</b>	tdatos : DefaultTableModel	Se crea un objeto tabla
<b>Atributo</b>	titulos[] : String	Obtiene los títulos de la tabla a la que se realiza la consulta
<b>Atributo</b>	dat[][] : Object	Obtiene los datos de la tabla consultada
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Atributo</b>	conexión : Connection	Hace la conexión con la base de datos
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Operación</b>	obtenerDatos() : void	Realiza una consulta y obtiene los datos de la tabla alumno
<b>Operación</b>	mostrar(datos : ResultSet) : void	Muestra la información de la consulta de la tabla alumno
<b>Operación</b>	limpiar() : void	Limpia la tabla cuando se realiza una nueva consulta
<b>Operación</b>	habilitar() : void	Habilita los botones cuando se realiza la consulta
<b>Operación</b>	deshabilitar() : void	Deshabilita los botones cuando la consulta no tiene datos
<b>Interfaz</b>	Asis	Está compuesta por botones, una tabla, cajas de texto, dateChooserCombo y etiquetas

## 6.4.20 Diagrama de clase AsisPon

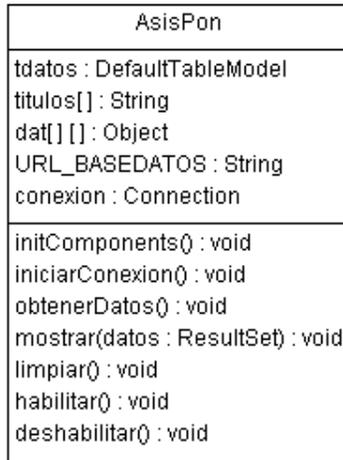


Imagen 47. Diagrama de clase AsisPon

Tabla 29. Diagrama de clase AsisPon

Elemento	Nombre	Descripción
<b>Clase</b>	AsisPon	El administrador registra la asistencia del ponente
<b>Atributo</b>	tdatos : DefaultTableModel	Se crea un objeto tabla
<b>Atributo</b>	titulos[] : String	Obtiene los títulos de la tabla a la que se realiza la consulta
<b>Atributo</b>	dat[][] : Object	Obtiene los datos de la tabla consultada
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Atributo</b>	conexión : Connection	Hace la conexión con la base de datos
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Operación</b>	obtenerDatos() : void	Realiza una consulta y obtiene los datos de la tabla alumno
<b>Operación</b>	mostrar(datos : ResultSet) : void	Muestra la información de la consulta de la tabla alumno
<b>Operación</b>	limpiar() : void	Limpia la tabla cuando se realiza una nueva consulta

<b>Operación</b>	habilitar() : void	Habilita los botones cuando se realiza la consulta
<b>Operación</b>	deshabilitar() : void	Deshabilita los botones cuando la consulta no tiene datos
<b>Interfaz</b>	AsisPon	Está compuesta por botones, una tabla, cajas de texto, dateChooserCombo y etiquetas

#### 6.4.21 Diagrama de clase Constancias

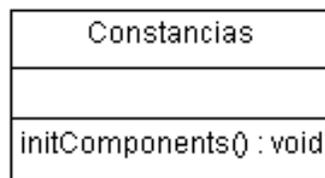


Imagen 48. Diagrama de clase Constancias

Tabla 30. Diagrama de clase Constancias

<b>Elemento</b>	<b>Nombre</b>	<b>Descripción</b>
<b>Clase</b>	Constancias	En esta ventana se muestran las opciones para generar la documentación
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Interfaz</b>	Constancias	Está compuesta por tres botones

## 6.4.22 Diagrama de clase AlumConst

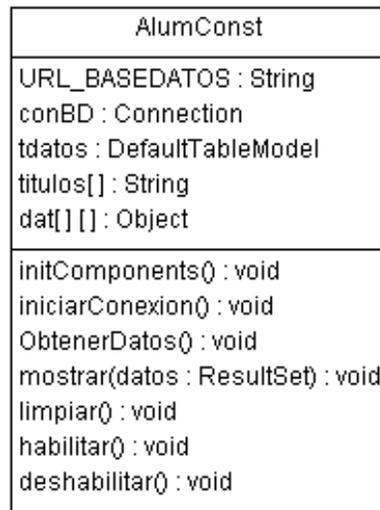


Imagen 49. Diagrama de clase AlumConst

Tabla 31. Diagrama de clase AlumConst

Elemento	Nombre	Descripción
<b>Clase</b>	AlumConst	Se generan las constancias de los alumnos que asistieron al evento
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Atributo</b>	conBD : Connection	Hace la conexión con la base de datos
<b>Atributo</b>	tdatos : DefaultTableModel	Se crea un objeto tabla
<b>Atributo</b>	titulos[] : String	Obtiene los títulos de la tabla a la que se realiza la consulta
<b>Atributo</b>	dat[][] : Object	Obtiene los datos de la tabla consultada
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Operación</b>	obtenerDatos() : void	Realiza una consulta y obtiene los datos de los paquetes
<b>Operación</b>	mostrar(datos : ResultSet) : void	Muestra la información de la consulta de los paquetes

<b>Operación</b>	limpiar() : void	Limpia la tabla cuando se realiza una nueva consulta
<b>Operación</b>	habilitar() : void	Habilita los botones cuando se realiza la consulta
<b>Operación</b>	deshabilitar() : void	Deshabilita los botones cuando la consulta no tiene datos
<b>Interfaz</b>	AlumConst	Está compuesta por botones, una tabla, cajas de texto, dos dateChooserCombo y etiquetas

### 6.4.23 Diagrama de clase PonAgr

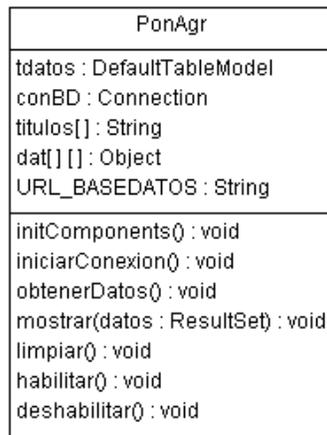


Imagen 50. Diagrama de clase PonAgr

Tabla 32. Diagrama de clase PonAgr

<b>Elemento</b>	<b>Nombre</b>	<b>Descripción</b>
<b>Clase</b>	PonAgr	Se generan los agradecimientos de los ponentes que asisitieron al evento
<b>Atributo</b>	tdatos : DefaultTableModel	Se crea un objeto tabla
<b>Atributo</b>	conBD : Connection	Hace la conexión con la base de datos
<b>Atributo</b>	titulos[] : String	Obtiene los títulos de la tabla a la que se realiza la consulta
<b>Atributo</b>	dat[][] : Object	Obtiene los datos de la tabla consultada

<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Operación</b>	obtenerDatos() : void	Realiza una consulta y obtiene los datos de los paquetes
<b>Operación</b>	mostrar(datos : ResultSet) : void	Muestra la información de la consulta de los paquetes
<b>Operación</b>	limpiar() : void	Limpia la tabla cuando se realiza una nueva consulta
<b>Operación</b>	habilitar() : void	Habilita los botones cuando se realiza la consulta
<b>Operación</b>	deshabilitar() : void	Deshabilita los botones cuando la consulta no tiene datos
<b>Interfaz</b>	PonAgr	Está compuesta por botones, una tabla, cajas de texto, dos dateChooserCombo y etiquetas

#### 6.4.24 Diagrama de clase Asistencia

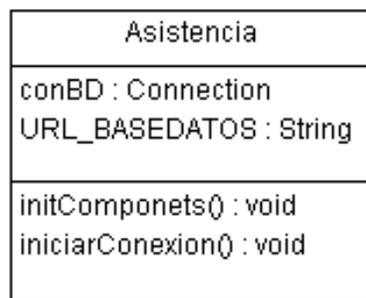


Imagen 51. Diagrama de clase Asistencia

Tabla 33. Diagrama de clase Asistencia

<b>Elemento</b>	<b>Nombre</b>	<b>Descripción</b>
<b>Clase</b>	Asistencia	El administrador genera las listas de asistencia de los alumnos
<b>Atributo</b>	conBD : Connection	Hace la conexión con la base de datos
<b>Atributo</b>	URL_BASEDATOS : String	Obtiene la ruta para hacer la conexión con mysql
<b>Operación</b>	initComponents() : void	Contiene los componentes como botones y cajas de texto
<b>Operación</b>	iniciarConexion() : void	Inicia la conexión con la base de datos
<b>Interfaz</b>	Asistencia	Está compuesta por botones, cajas de texto, dateChooserCombo y etiquetas



## 6.6 Modelo Relacional

Alumno

<u>No_cuenta</u>	Nombre	ApellidoPaterno	ApellidoMaterno	Semestre	Licenciatura	Folio_Paquete
------------------	--------	-----------------	-----------------	----------	--------------	---------------

Organizador

<u>RFC</u>	Clave	Nombre	ApellidoPaterno	ApellidoMaterno
------------	-------	--------	-----------------	-----------------

Ponente

<u>RFC</u>	Nombre	ApellidoPaterno	ApellidoMaterno	CurriculumVitae	GradoAcademico	TemaExposicion	RFC_Ponente
------------	--------	-----------------	-----------------	-----------------	----------------	----------------	-------------

Paquete

<u>Folio</u>	Costo	Ponencia	Taller	RFC_Organizador
--------------	-------	----------	--------	-----------------

Actividad

<u>Tema</u>	Hora_de_inicio	Fecha	Tipo	Lugar	Folio_Paquete	RFC_Ponente	RFC_Organizador
-------------	----------------	-------	------	-------	---------------	-------------	-----------------

Asistencia

Fecha	Hora	No_cuenta_Alumno	RFC_Organizador
-------	------	------------------	-----------------

Asistencia\_p

Fecha	Hora	RFC_Ponente	RFC_Organizador
-------	------	-------------	-----------------

Constancia

<u>Folio</u>	FechaGeneracion	FechaEntrega	No_cuenta_Alumno	RFC_Organizador
--------------	-----------------	--------------	------------------	-----------------

Agradecimiento

<u>Folio</u>	FechaGeneracion	FechaEntrega	RFC_Ponente	RFC_Organizador
--------------	-----------------	--------------	-------------	-----------------

Lista\_asis

<u>Folio</u>	FechaGeneracion	RFC_Organizador
--------------	-----------------	-----------------

## 6.7 Diagrama del modelo relacional

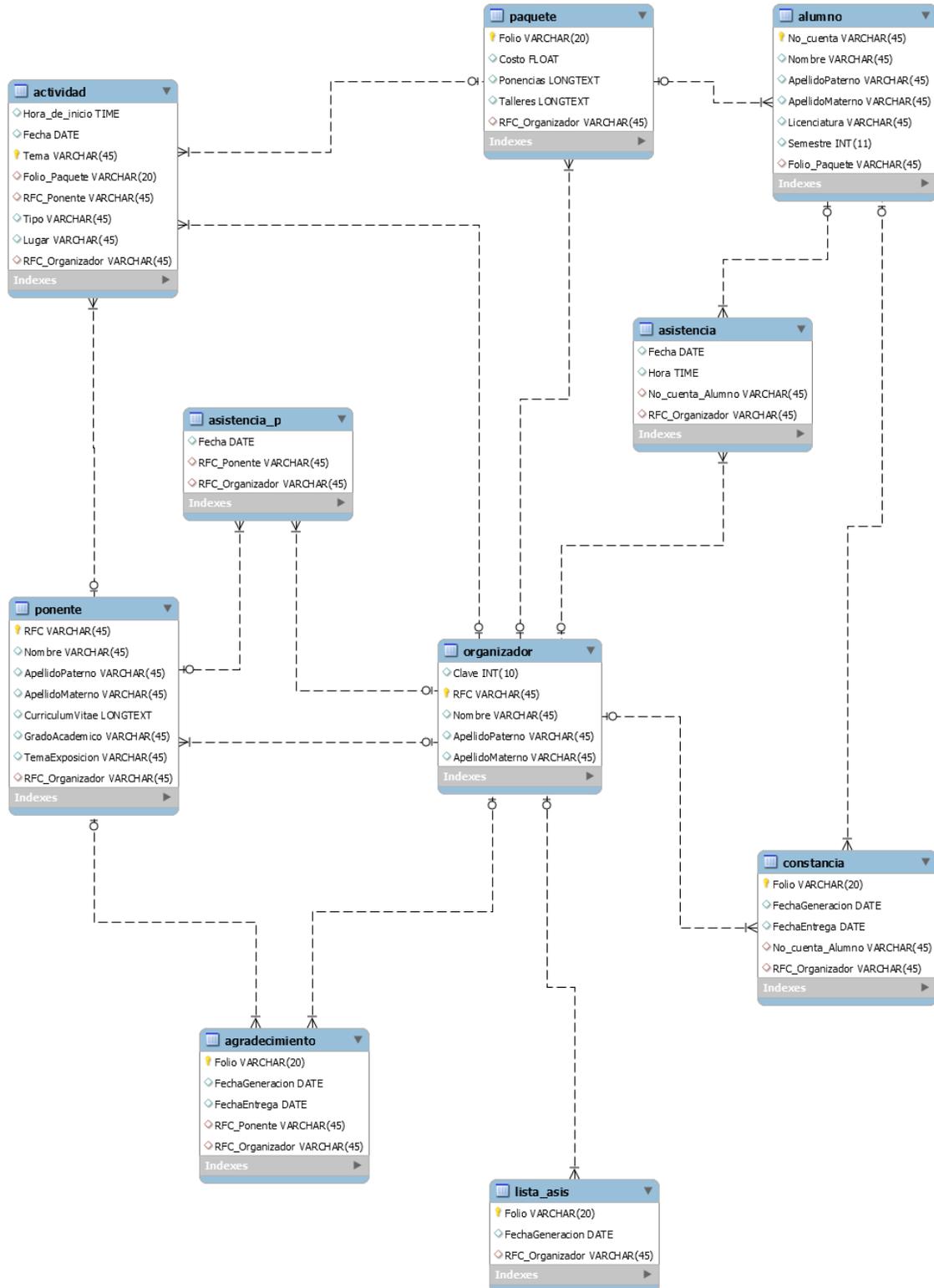


Imagen 53. Diagrama del modelo relacional

# Capítulo 7 RESULTADOS

## 7.1 Resultados del Sistema.

En base a los requerimientos se realizó el desarrollo del sistema y su ejecución finalizando el desarrollo del prototipo establecido en los requerimientos, los resultados obtenidos están fundamentados en la implementación de las herramientas para el desarrollo y diseño del software, analizado por cada etapa del ciclo de vida que nos brinda Cascada. El resultado de este estudio es el cumpliendo los objetivos particulares:

1. Identificar los procesos involucrados en los eventos universitarios.
2. Identificar las características de la documentación a generar.
3. Diseño del sistema en base a los requerimientos.
4. Diseño de la base de datos en base a los requerimientos.

Se implementaron los siguientes procesos en base a la metodología de cascada:

- Análisis de requisitos
- Diseño del Sistema
- Diseño del Programa
- Codificación
- Pruebas
- Verificación
- Mantenimiento

Como resultado se obtuvo un sistema apoyándonos de las tecnologías siguientes: Java, NetBeans, MySQL Workbench. Se realizó una base de datos en la cual se almacena la información registrada en el evento.

La interacción de la bases de datos es por medio del software desarrollado en Java, los cuales son operados por el organizador principalmente, y por el alumno para realizar su preinscripción.

El diseño del sistema está hecho en base a los requerimientos. Inicialmente se pensaba realizar el sistema en forma de kiosco, pero por falta de presupuesto no pudo implementarse de esa forma. El sistema está hecho para que el organizador tenga el control de las actividades realizadas en el

evento, y que nadie pueda agregar información innecesaria a la base de datos. La única excepción son los alumnos, que realizan su preinscripción, ésta es completada por el organizador al recibir el pago en efectivo.

En las siguientes secciones se muestran los resultados de cada parte del sistema. Cabe mencionar que el RFC del organizador (administrador) sirve como llave foránea de casi todas las tablas de la base de datos para tener conocimiento de quien realiza cada registro.

### 7.1.1 Preinscripción de alumnos

Para realizar la preinscripción de los alumnos se creó una interfaz para que introdujera sus datos (véase Imagen 54). Esa interfaz también puede consultar a otra (véase Imagen 55) para consultar las actividades y paquetes del evento. Las consultas se realizan a las tablas “actividad” (véase Imagen 64) y “paquete” (véase Imagen 62) respectivamente.

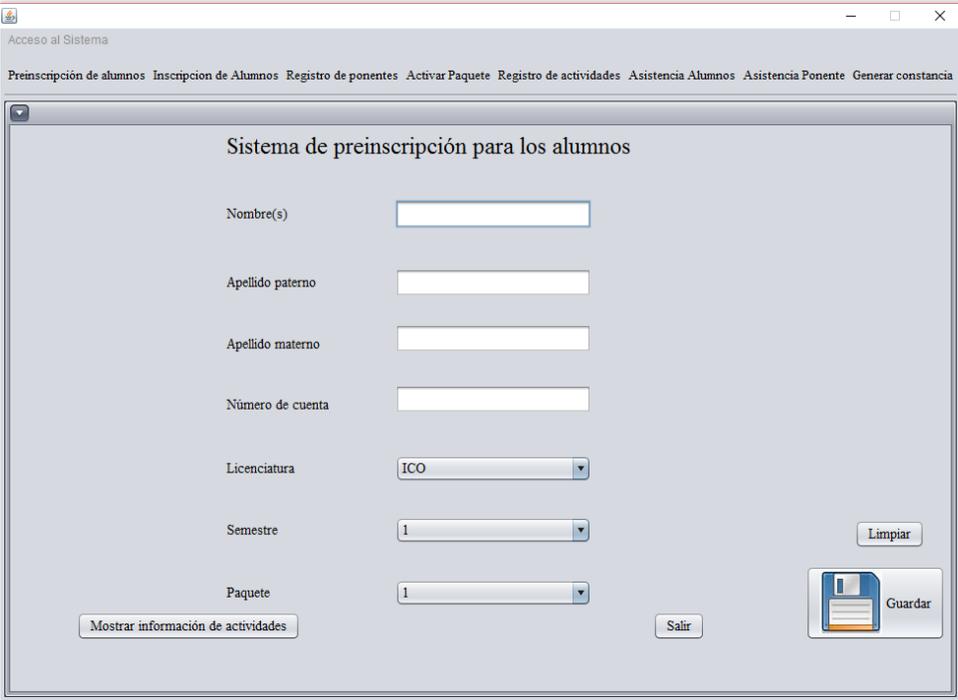


Imagen 54. Interfaz de preinscripción de alumnos



Imagen 55. Interfaz de consulta de actividades

Los datos del alumno quedan guardados en la base de datos en la tabla “alumno” (véase Imagen 56).

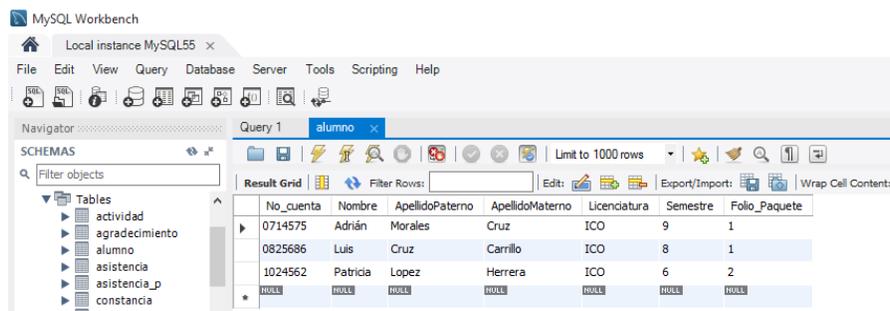


Imagen 56. Tabla “alumno”

## 7.1.2 Inscripción de alumnos

Para completar la inscripción del alumno se creó una interfaz (véase Imagen 57) para realizar una consulta a la tabla “alumno” (véase Imagen 56) de la base de datos y generar su gafete con código de barras (véase Imagen 58).



Imagen 57. Interfaz inscripción de alumnos



Imagen 58. Gafete con código de barras

### 7.1.3 Registro de ponente

Para el registro de ponente se creó una interfaz similar a la de preinscripción de alumnos (véase Imagen 59), pero en ésta, es el administrador quien ingresa la información del ponente y se guarda en la tabla “ponente” (véase Imagen 60) de la base de datos.

Registro de ponentes

Nombre(s)

Apellido paterno

Apellido materno

RFC Ponente

Grado académico

Tema de exposición

Curriculum Vitae

RFC Administrador

Limpiar Salir Registrar

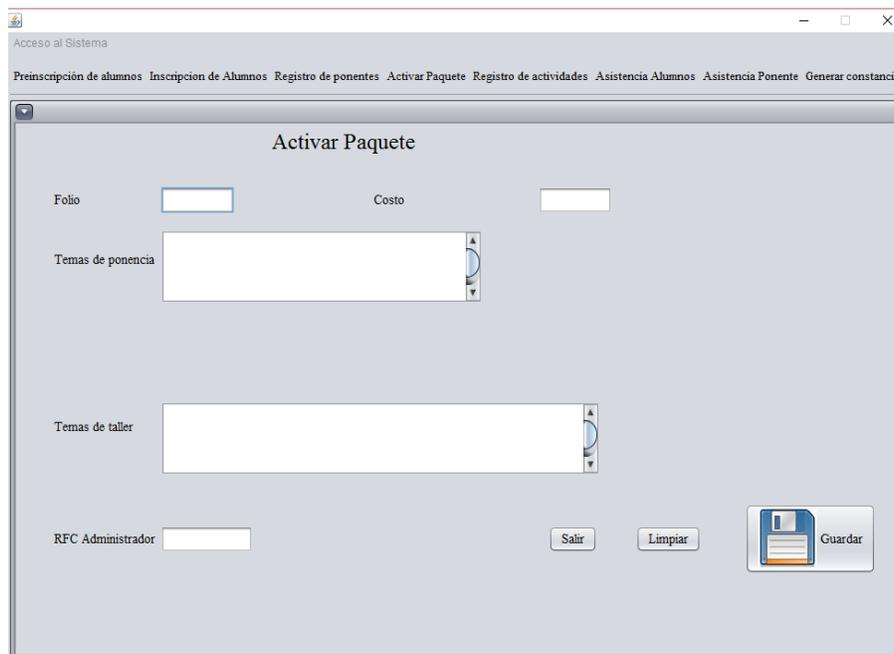
Imagen 59. Interfaz de registro de ponentes

RFC	Nombre	ApellidoPa	ApellidoMaterno	CurriculumVitae	GradoAcademico	TemaExposicion	RFC
ALFD870913	Daniel	Alfaro	Facio	Profesor de tiempo completo	Maestro en computacion	Mantenimiento de servidores	CRM
COMI870118	Ivan	Contreras	Maldonado	Profesor de tiempo completo	Ingeniero en sistemas computacionales	Beneficios de la robotica	CRM
CRMA920414	Adrian	Cruz	Mora	Profesor de tiempo completo	Ingeniero en computacion	Servidores	CRM
CURC860522	Carlos	Cuevas	Rios	Profesor de tiempo completo	Ingeniero en computacion	Sistemas digitales	CRM
GOJA701203	Alejandro	Gonzalez	Juarez	Profesor de tiempo completo	Doctor en educacion	Procesamiento digital de imagenes	CRM
VERA830908	Antonio	Vega	Ramirez	Profesor de tiempo completo	Licenciado en ciencias de la computacion	Reparacion de computadoras	CRM

Imagen 60. Tabla “ponente”

## 7.1.4 Activar paquete

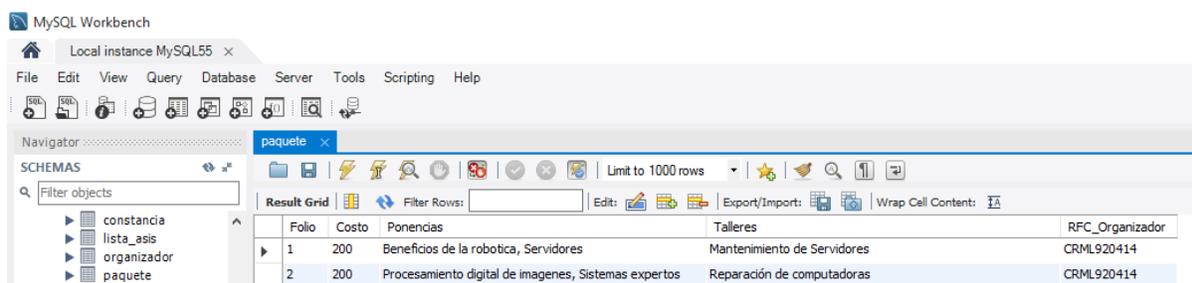
Para dar de alta un paquete, se creó una interfaz para agregar su costo, folio y temas que incluirá (véase Imagen 61). La información se guarda en la tabla “paquete” (véase Imagen 62) de la base de datos.



The screenshot shows a web browser window titled 'Acceso al Sistema'. The navigation menu includes: Preinscripción de alumnos, Inscripción de Alumnos, Registro de ponentes, Activar Paquete, Registro de actividades, Asistencia Alumnos, Asistencia Ponente, and Generar constancia. The main content area is titled 'Activar Paquete' and contains the following form elements:

- Folio:
- Costo:
- RFC Administrador:
- Temas de ponencia:
- Temas de taller:
- Buttons: Salir, Limpiar, and Guardar (with a floppy disk icon).

Imagen 61. Interfaz activar paquete



The screenshot shows the MySQL Workbench interface with the 'paquete' table selected in the Navigator. The Result Grid displays the following data:

Folio	Costo	Ponencias	Talleres	RFC_Organizador
1	200	Beneficios de la robotica, Servidores	Mantenimiento de Servidores	CRML920414
2	200	Procesamiento digital de imagenes, Sistemas expertos	Reparación de computadoras	CRML920414

Imagen 62. Tabla “paquete”

## 7.1.5 Registro de actividades

Para dar de alta una actividad, deben estar registrados los paquetes y los ponentes, para agregar el RFC del ponente que impartirá la actividad y el paquete al que pertenecerá. La información se guarda en la tabla “actividad” (véase Imagen 64) de la base de datos. La interfaz para registrar actividades se muestra en la Imagen 63.

Registro de actividades

Tema

Hora de inicio

RFC Ponente

Tipo

Paquete

Fecha de ponencia

Lugar

RFC Administrador

Imagen 63. Interfaz registro de actividades

MySQL Workbench

Local instance MySQL55

paquete actividad

Hora_de_inicio	Fecha	Tema	Folio_Paquete	RFC_Ponente	Tipo	Lugar	RFC_Organizador
10:00:00	2016-02-24	Beneficios de la robotica	1	COM1870118	Ponencia	Aditorio 1	CRML920414
12:00:00	2016-02-24	Mantenimiento de servidores	1	ALFD870913	Taller	Laboratorio 1	CRML920414
12:00:00	2016-02-25	Procesamiento digital de imagenes	2	GOJA701203	Ponencia	Auditorio 2	CRML920414
14:00:00	2016-02-25	Reparacion de computadoras	2	VERA830908	Taller	Laboratorio 2	CRML920414
11:00:00	2016-02-24	Servidores	1	CRMA920414	Ponencia	Auditorio 1	CRML920414
13:00:00	2016-02-25	Sistemas digitales	2	CURC860522	Ponencia	Auditorio 2	CRML920414
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Imagen 64. Tabla “actividad”

## 7.1.6 Asistencia de alumnos

Para registrar la asistencia de los alumnos se creó una interfaz (véase Imagen 61) que realiza una consulta a la tabla “alumno” (véase Imagen56), escaneando el gafete con código de barras. Los datos ingresados se guardan en la tabla “asistencia” (véase Imagen 66); en esta tabla, el número de cuenta del alumno es una llave foránea.

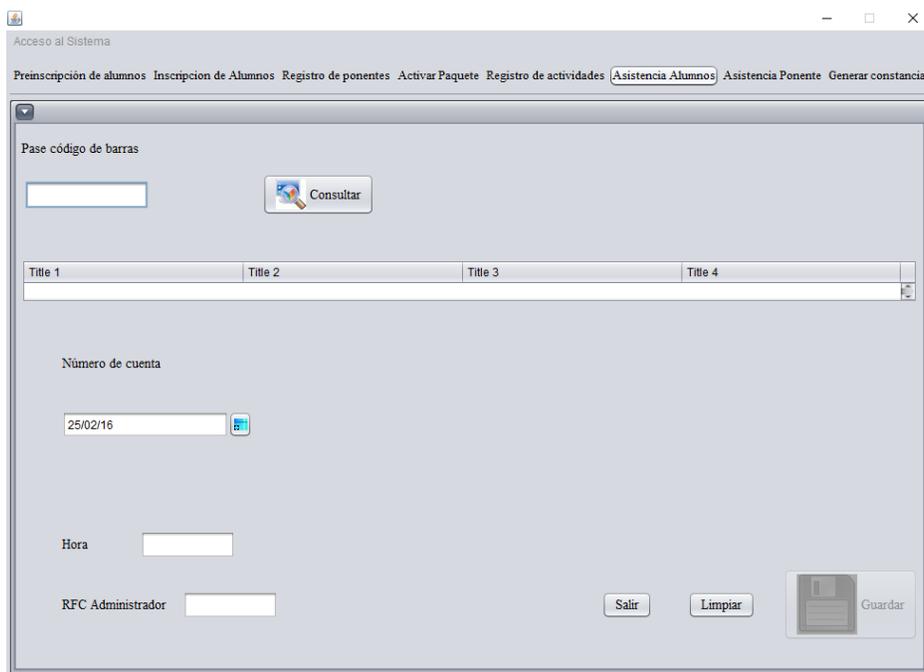


Imagen 65. Interfaz asistencia de alumnos

Fecha	Hora	No_cuenta_Alumno	RFC_Organizador
2016-02-26	12:00:00	0825686	CRML920414
2016-02-27	13:00:00	1024562	CRML920414

Imagen 66. Tabla “asistencia”

## 7.1.7 Asistencia de ponente

Para registrar la asistencia del ponente se creó una interfaz (véase Imagen 67) que realiza una consulta a la tabla “ponente” (véase Imagen 60), ingresando el RFC del ponente. Los datos ingresados se guardan en la tabla “asistencia\_p” (véase Imagen 68); en esta tabla, el RFC del ponente es una llave foránea.

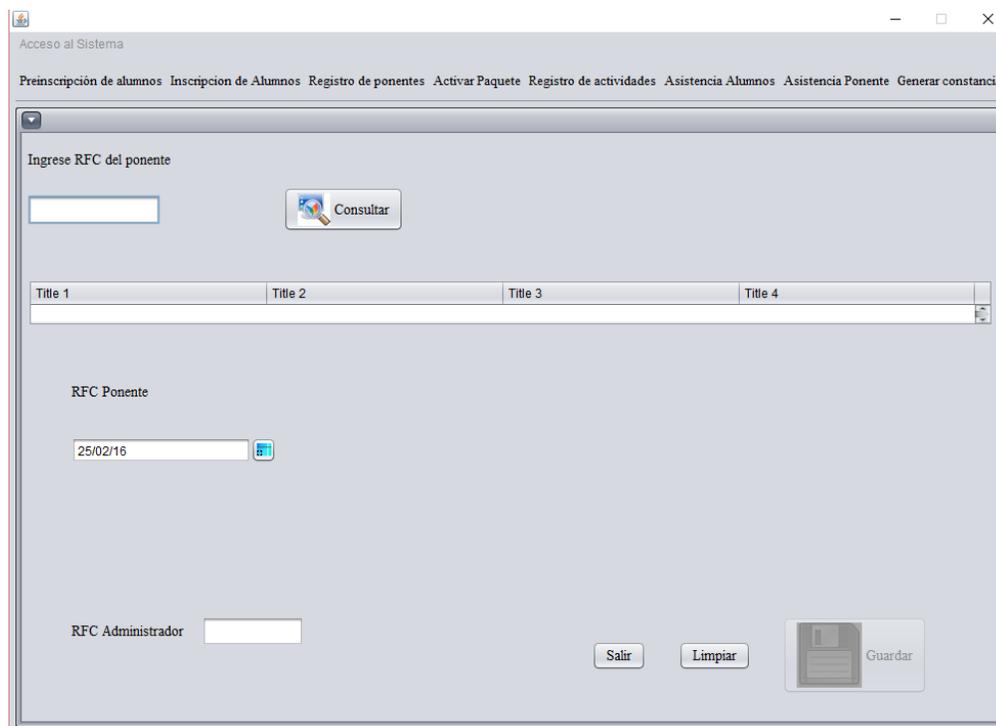


Imagen 67. Interfaz asistencia de ponente

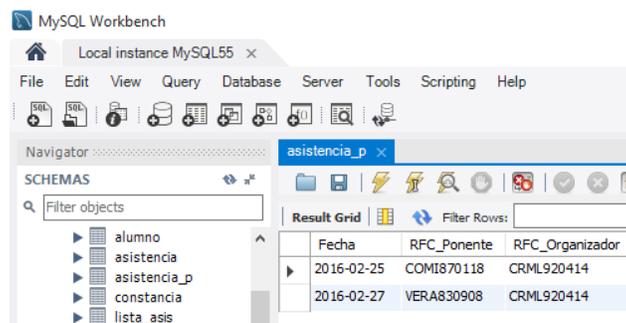
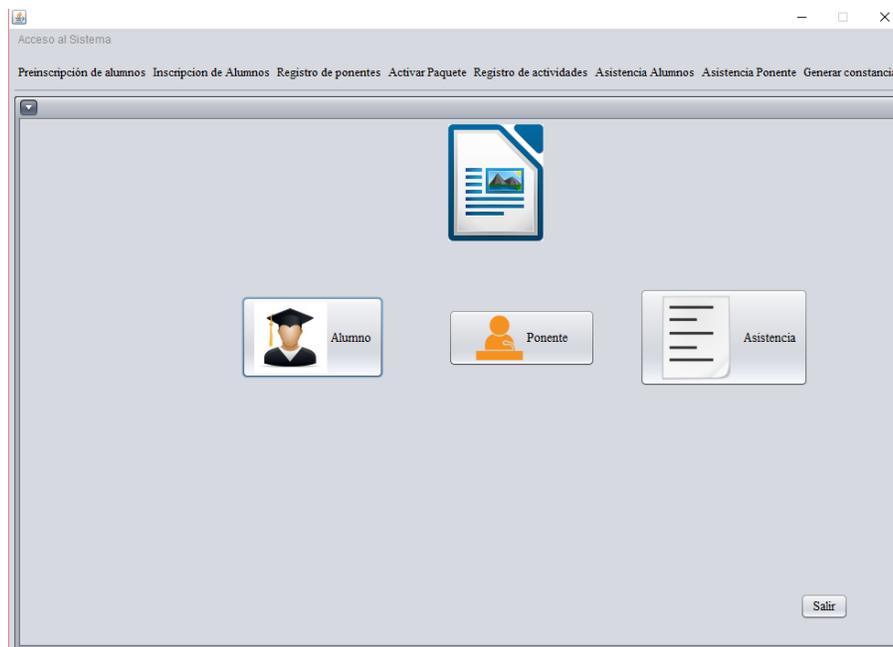


Imagen 68. Tabla “asistencia\_p”

## 7.1.8 Generar documentación

Para generar la documentación (constancias de asistencia, agradecimientos de ponente, y listas de asistencia) se creó una interfaz (véase Imagen 69) con tres botones para cada tipo de documento. En la interfaz de constancia (véase Imagen 70) se hace una consulta a la tabla “alumno” (véase Imagen 56) con el número de cuenta del alumno, se obtienen los datos en conjunto con la tabla “asistencia” (véase Imagen 66) y se ingresan los datos requeridos. Los datos se guardan en la tabla “constancia” (véase Imagen 71); en esta tabla, el número de cuenta del alumno es una llave foránea. Finalmente, se genera el documento (véase Imagen 72).



*Imagen 69. Interfaz de botones para generar documentación*

Número de cuenta

Campo1	Campo2	Campo3	Campo4	Campo5	Campo6
--------	--------	--------	--------	--------	--------

Folio

RFC Administrador

Fecha de generación

Fecha de entrega

Imagen 70. Interfaz de constancia de asistencia de alumno

MySQL Workbench

Local instance MySQL55

File Edit View Query Database Server Tools Scripting Help

Navigator constancia

SCHEMAS

Filter objects

Tables

- actividad
- agradecimiento
- alumno
- asistencia
- asistencia\_p
- constancia

Result Grid

	Folio	FechaGeneracion	FechaEntrega	No_cuenta_Alumno	RFC_Organizador
▶	1	2016-02-27	2016-02-28	0825686	CRML920414
*	NULL	NULL	NULL	NULL	NULL

Imagen 71. Tabla "constancia"

## Constancia de asistencia

Congreso 2016



Número de cuenta	Nombre	Apellido Paterno	Apellido Materno	Licenciatura	Semestre
0825686	Luis	Cruz	Carrillo	ICO	8

*Imagen 72. Constancia de asistencia*

Para los agradecimientos de ponentes se creó una interfaz similar a la de constancias de alumnos. En esta interfaz (véase Imagen 73) se realiza una consulta a la tabla “ponente” (véase Imagen 60), se obtienen los datos en conjunto con la tabla “asistencia\_p” (véase Imagen 68) y se ingresan los datos requeridos. Los datos se guardan en la tabla “agradecimiento” (véase Imagen 74); en esta tabla, el RFC del ponente es una llave foránea. Finalmente, se genera el documento (véase Imagen 75).

RFC Ponente

Campo1	Campo2	Campo3	Campo4	Campo5	Campo6
--------	--------	--------	--------	--------	--------

Folio

RFC Administrador

Fecha de generación

Fecha de entrega

*Imagen 73. Interfaz de agradecimiento de ponente*

MySQL Workbench

Local instance MySQL55 x

File Edit View Query Database Server Tools Scripting Help

agradecimiento x

Result Grid Filter Rows: Edit: Export

Folio	FechaGeneracion	FechaEntrega	RFC_Ponente	RFC_Organizador
1	2016-02-27	2016-02-28	COMI870118	CRML920414
*	NULL	NULL	NULL	NULL

*Imagen 74. Tabla "agradecimiento"*

## Agradecimiento

Congreso 2016



Nombre	Apellido Paterno	Apellido Materno
Ivan	Contreras	Maldonado

Imagen 75. Agradecimiento de ponente

Finalmente, para las listas de asistencia se creó una interfaz (véase Imagen 76) en la que solo se introducen los datos que se solicitan; los datos se guardan en la tabla “lista\_asis” (véase Imagen 77). Finalmente, el administrador generará una lista de asistencia de alumnos (véase Imagen 78).

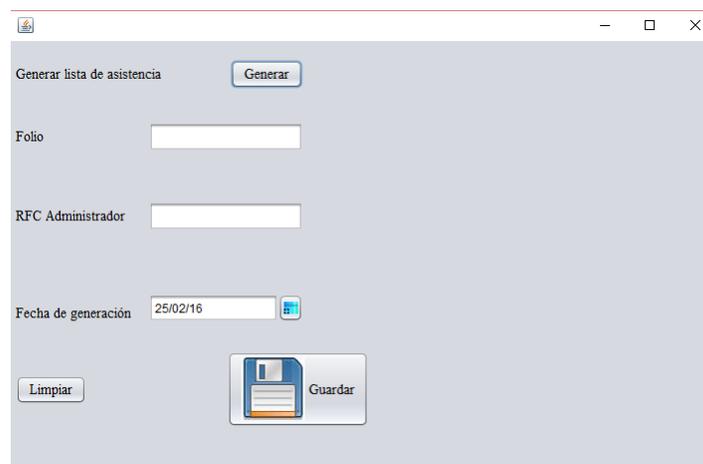
A screenshot of a web application window titled "Generar lista de asistencia". The window contains a "Generar" button at the top right. Below it are three input fields: "Folio" (empty), "RFC Administrador" (empty), and "Fecha de generación" (containing "25/02/16" with a calendar icon). At the bottom left is a "Limpiar" button, and at the bottom center is a "Guardar" button with a floppy disk icon.

Imagen 76. Interfaz de listas de asistencia

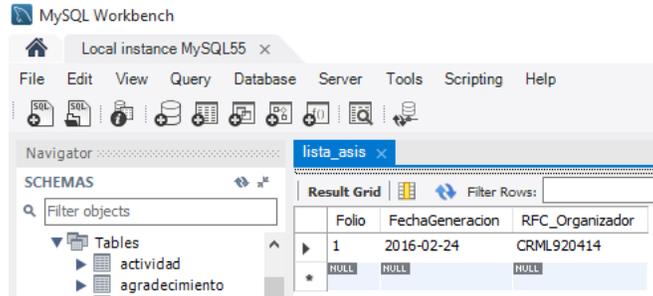


Imagen 77. Tabla “lista\_asis”

## Lista de asistencia

Congreso 2016



Número de cuenta	Nombre	Apellido Paterno	Apellido Materno	Licenciatura	Semestre
0825686	Luis	Cruz	Carrillo	ICO	8
1024562	Patricia	Lopez	Herrera	ICO	6

Imagen 78. Lista de asistencia

# CONCLUSIONES

El desarrollo del sistema de inscripción a eventos académicos universitarios para el Centro Universitario UAEM Texcoco cumple con los requerimientos obtenidos, siendo los más importantes el registro de asistencia y la generación de la documentación.

El requerimiento importante para los alumnos que asisten a los eventos es el de garantizar su lugar, algo que el proceso de inscripción manual no garantiza al 100 por ciento, pues la información puede perderse. Es por eso que pueden realizar una preinscripción guardando sus datos en la base de datos; la preinscripción garantiza que no haya pérdida de datos. También podrán consultar la información de los eventos (temas y horarios de actividades). Para completar su preinscripción tienen que realizar su pago en efectivo y recibirán su gafete con el que se les tomará asistencia para poder generar su constancia.

La asistencia de ponentes será mediante su RFC, que será agregado junto con su información general (nombre, apellidos, etc.). La asistencia de los ponentes también será el factor para generar su agradecimiento.

El registro de actividades y de paquetes ahora será más eficiente, pues se podrán agregar temas, horarios y lugares en que serán impartidas.

Todos los registros a realizar serán con ayuda del sistema. Éste comprueba que el registro de información a los eventos universitarios mediante el sistema de información es realmente más confiable y eficiente con respecto al registro manual realizado por los organizadores. Con el sistema es más difícil que la información se pierda o que no sea registrada correctamente.

La documentación también será generada con el sistema, realizando las consultas dependiendo del documento que se va a generar. Esto también supera a la forma manual, pues la documentación será generada automáticamente en lugar de estar revisando información de cada persona a la que se le dará el documento, además de que ahora se cuentan con listas de asistencia de los alumnos, así se podrá hacer una comparación entre las constancias y las listas en caso de ser necesario.

Como ya fue mencionado, el sistema estaba pensado para ser realizado en un kiosco, pero por falta de presupuesto no pudo ser así. También hubiera sido útil, aunque el sistema realizado obtiene los mismos beneficios en cuanto al registro de información.

# BIBLIOGRAFÍA

- Alvarez, S. (31 de Julio de 2007). *desarrolloweb.com*. Obtenido de desarrolloweb.com:  
<http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html>
- Ayala, A. P. (2006). *Ingeniería de software: Una guía para crear sistemas de información*. México.
- Bedoya García, Y. (12 de Septiembre de 2012). *SlideShare*. Obtenido de SlideShare:  
<http://es.slideshare.net/yeisonandrey19/eventos-academicos>
- Bentley, W. (2008). *Análisis de sistemas, Diseño y Métodos*. México: Mc Graw Hill.
- Blogspot. (3 de Marzo de 2013). Obtenido de <http://modelo-cascada.blogspot.mx/>
- Bracho, D. (11 de Octubre de 2011). *slideshare*. Obtenido de slideshare:  
<http://es.slideshare.net/sylem123/beneficios-de-los-sistemas-de-informacion>
- Bravo, I. M. (2006). *Inform@tica*. Obtenido de Inform@tica: <http://indira-informatica.blogspot.mx/2007/09/qu-es-un-sistema-de-gestin-de-base-de.html>
- Castell Delgado, H. S. (30 de Septiembre de 2014). *Prezi*. Obtenido de Prezi:  
<https://prezi.com/923yydsinkww/concepto-caracteristicas-ventajas-y-desventajas-de-mysql-y-workbench/>
- Centro de escritura Javeriano*. (10 de Octubre de 2009). Obtenido de Centro de escritura Javeriano:  
[http://centrodescritura.javerianacali.edu.co/index.php?option=com\\_content&view=article&id=122:ponencia&catid=44:tipos-de-textos&Itemid=66](http://centrodescritura.javerianacali.edu.co/index.php?option=com_content&view=article&id=122:ponencia&catid=44:tipos-de-textos&Itemid=66)
- Cruz, D. (29 de Marzo de 2011). *Diana Eventos*. Recuperado el Marzo de 2015, de Diana Eventos:  
<https://dianaeventos.blogia.com/2011/033001-definicion-de-congreso-su-planificacion-y-su-tipo-logia.php>

*Definicion.de.* (2015). Recuperado el Marzo de 2015, de Definicion.de:  
<http://definicion.de/seminario/>

Elmasri, R., & Navathe, S. B. (2007). *Fundamentos de sistemas de bases de datos*. Madrid: Pearson Addison Wesley.

Fowler, M., & Scott, K. (1999). En M. Fowler, & K. Scott, *UML gota a gota* (pág. 224). México: Addison Wesley Longman de México. S.A. de C.V.

*GenBeta dev.* (09 de Enero de 2014). Obtenido de  
<http://www.genbetadev.com/herramientas/netbeans-1>

Grady Booch, J. R. (1999). *El Lenguaje Unificado de Modelado*. Addison Wesley.

H.Dutoit, B. B. (2000). *Object Oriented Software Engineering*. Prentice Hall.

Infante, A., Arrieta, L., & Cruz, D. (29 de Marzo de 2011). *DIANAEVENTOS*. Obtenido de DIANAEVENTOS: <http://dianaeventos.blogia.com/2011/033001-definicion-de-congreso-su-planificacion-y-su-tipo-logia.php>

*Inform@tica.* (s. f.). Obtenido de Inform@tica: <http://indira-informatica.blogspot.mx/2007/09/ques-mysql.html>

Joyanes Aguilar, L., & Zahonero Martínez, I. (2011). *Programación en Java 6. Algoritmos, programación orientada a objetos e interfaz gráfica de usuario*. México: Mc Graw Hill.

Laboratorio Nacional de Calidad del Software. (2009). *Ingeniería del software: metodologías y ciclos de vida*. México.

Mastermagazine. (2010). Obtenido de <http://www.mastermagazine.info/termino/6051.php>

Quito, I. R. (nd). *Informática aplicada*. Obtenido de <http://irfeyal.wordpress.com/investigaciones/sistemas-de-informacion/>

*word reference.* (2015). Recuperado el Marzo de 2015, de word reference:  
<http://www.wordreference.com/definicion/congreso>

# Anexo A MANUAL DE USUARIO

La ventana principal del sistema, permite acceder al sistema a los usuarios, alumnos para realizar la preinscripción, y organizadores (administradores), para registrar las actividades realizadas durante el evento, todos los botones están deshabilitados, con excepción del de acceso al sistema.



Imagen 79. Ventana principal del sistema

## A.1 Realizar preinscripción de alumnos

Primero acceden a la ventana de acceso al sistema, y seleccionan la opción alumno.

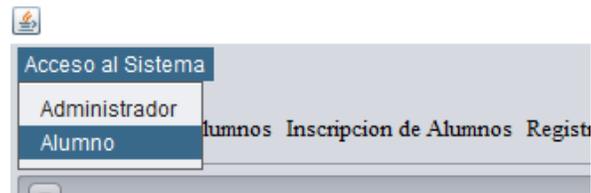


Imagen 80. Acceso al sistema como alumno

El botón de acceso al sistema se deshabilita, y el único botón que queda habilitado es el de preinscripción de alumnos.

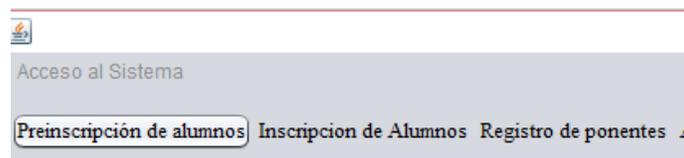


Imagen 81. Selección del botón de preinscripción de alumnos

Al seleccionar se muestra un panel con el mensaje de conexión exitosa.



Imagen 82. Mensaje de conexión exitosa para preinscripción

Se muestra ventana de preinscripción para que el alumno introduzca sus datos.

The screenshot shows a web browser window with the title "Acceso al Sistema". The main content area is titled "Sistema de preinscripción para los alumnos". The form contains the following fields and controls:

- Nombre(s): Text input field.
- Apellido paterno: Text input field.
- Apellido materno: Text input field.
- Número de cuenta: Text input field.
- Licenciatura: Dropdown menu with "ICO" selected.
- Semestre: Dropdown menu with "1" selected.
- Paquete: Dropdown menu with "1" selected.
- Buttons: "Limpiar", "Guardar" (with a floppy disk icon), "Salir", and "Mostrar información de actividades".

Imagen 83. Ventana de preinscripción

El botón de mostrar información de actividades abre una ventana para hacer una consulta sobre las actividades y los paquetes. También muestra un mensaje de conexión exitosa.



Imagen 84. Ventana de consulta de actividades

El alumno puede consultar la información y de los paquetes en cada tabla pulsando el botón para cada una.



Imagen 85. Consulta de actividades

Ya que el alumno conoce la información de las actividades puede realizar su preinscripción.

The screenshot shows a web browser window with the title "Acceso al Sistema". The navigation menu includes: "Preinscripción de alumnos", "Inscripción de Alumnos", "Registro de ponentes", "Activar Paquete", "Registro de actividades", "Asistencia Alumnos", "Asistencia Ponente", and "Generar constancia". The main content area is titled "Sistema de preinscripción para los alumnos" and contains the following form elements:

- Nombre(s): Text input field containing "Adrián".
- Apellido paterno: Text input field containing "Morales".
- Apellido materno: Text input field containing "Cruz".
- Número de cuenta: Text input field containing "0714575".
- Licenciatura: Dropdown menu with "ICO" selected.
- Semestre: Dropdown menu with "9" selected.
- Paquete: Dropdown menu with "1" selected.
- Buttons: "Mostrar información de actividades" (bottom left), "Limpiar" (right side), "Guardar" (bottom right, with a floppy disk icon), and "Salir" (bottom center).

Imagen 86. Registro de datos para preinscripción

Finalmente el alumno completa su preinscripción al seleccionar el botón guardar. Se muestra un panel con el mensaje "Se agregaron los datos".



Imagen 87. Mensaje de registro de datos exitoso

## A.2 Ingreso l sistema como administrador

Se selecciona la opción de administrador del botón de acceso al sistema.

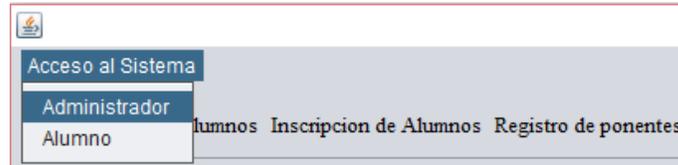


Imagen 88. Acceso al sistema como administrador

Se mostrará una venta para introducir una clave.



*Imagen 89. Introducir clave*

En caso de que la clave sea incorrecta se mostrará un panel con el mensaje “Verifique sus privilegios”.



*Imagen 90. Mensaje de verificación de privilegios*

Al ingresar correctamente al sistema, se deshabilita el botón de acceso al sistema, y se habilitan todos los botones, con excepción del botón de preinscripción de alumnos.

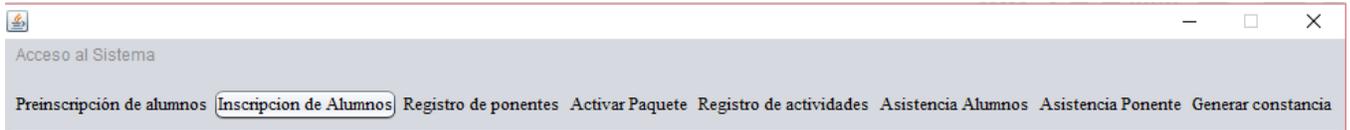


Imagen 91. Botones de administrador activados

### A.3 Completar inscripción de alumnos

Se ingresa mediante el botón de Inscripción de alumnos a la ventana.



Imagen 92. Ventana inscripción de alumnos

El administrador realiza una consulta usando el número de cuenta del alumno. En caso de no existir datos se muestra un mensaje de error.

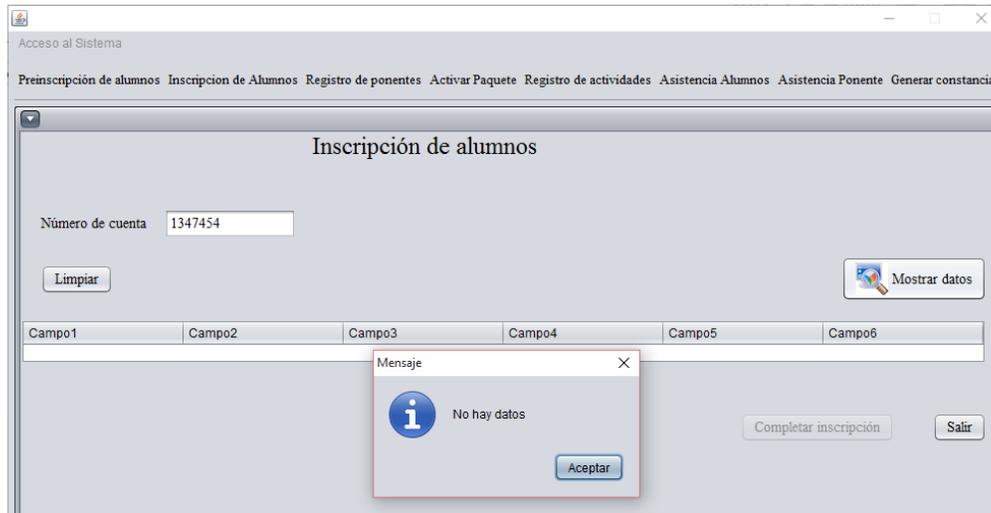


Imagen 93. Mensaje de error “No hay datos”

Al realizar la consulta correcta, se muestran los datos del alumno en la tabla y se activa el botón de completar inscripción.

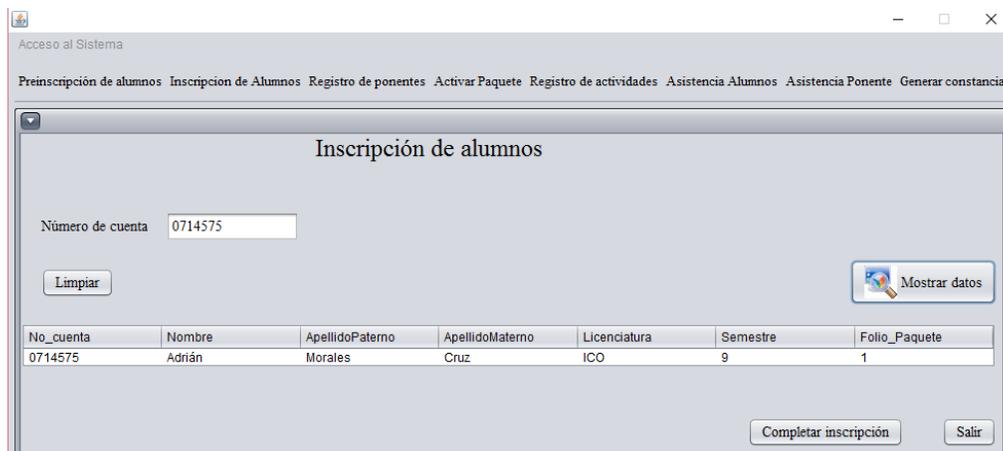


Imagen 94. Consulta de inscripción de alumnos

El botón de completar inscripción nos dirige a una ventana que servirá como gafete y recibo de pagos para los alumnos.



Imagen 95. Gafete de inscripción

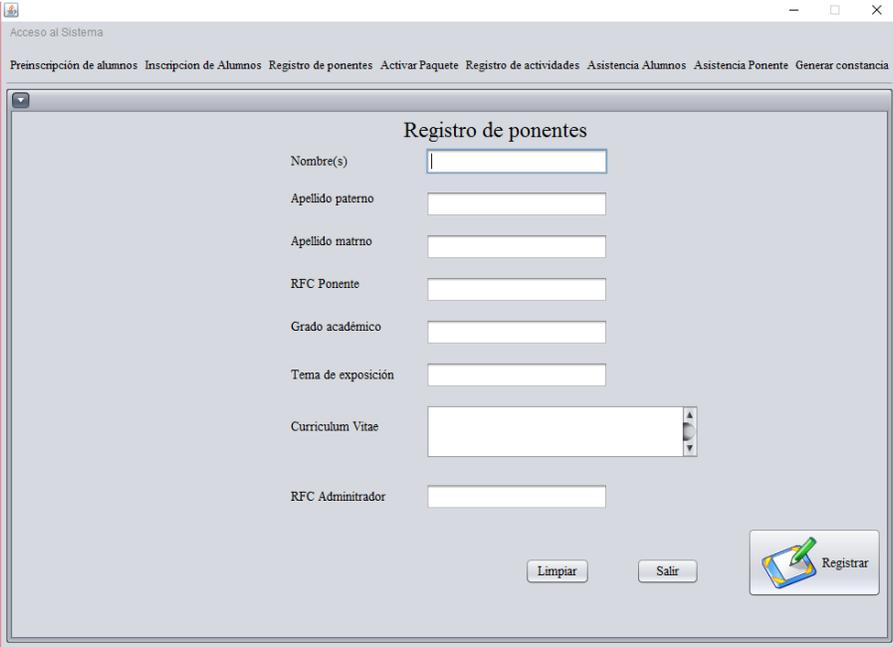
El botón generar crea un código de barras con el número de cuenta del alumno y se habilitará el botón de imprimir para poder entregarlo al alumno. Con ese código de barras se registrará la asistencia del alumno a los eventos.



Imagen 96. Gafete de inscripción con código de barras

## A.4 Registro de ponentes

El botón de registro de ponentes muestra la ventana para ingresar los datos del ponente.



The image shows a screenshot of a software application window titled "Acceso al Sistema". The window contains a menu bar with the following items: "Preinscripción de alumnos", "Inscripcion de Alumnos", "Registro de ponentes", "Activar Paquete", "Registro de actividades", "Asistencia Alumnos", "Asistencia Ponente", and "Generar constancia". The main content area is titled "Registro de ponentes" and contains several input fields for data entry:

- Nombre(s)
- Apellido paterno
- Apellido materno
- RFC Ponente
- Grado académico
- Tema de exposición
- Curriculum Vitae
- RFC Administrador

At the bottom right of the form, there are three buttons: "Limpiar", "Salir", and "Registrar". The "Registrar" button features a small icon of a notepad and a pencil.

*Imagen 97. Ventana de registro de datos de ponentes*

El administrador ingresa los datos de las personas que impartirán alguna actividad. Ingresar el RFC del administrador es para llevar el control de quien es la persona que registra.

Acceso al Sistema

Preinscripción de alumnos Inscripción de Alumnos Registro de ponentes Activar Paquete Registro de actividades Asistencia Alumnos Asistencia Ponente Generar constancia

### Registro de ponentes

Nombre(s)

Apellido paterno

Apellido materno

RFC Ponente

Grado académico

Tema de exposición

Curriculum Vitae

RFC Administrador

Imagen 98. Registro de ponente

## A.5 Activar paquete

El botón activar paquete muestra la ventana para ingresar los datos correspondientes.

Acceso al Sistema

Preinscripción de alumnos Inscripción de Alumnos Registro de ponentes Activar Paquete Registro de actividades Asistencia Alumnos Asistencia Ponente Generar constancia

### Activar Paquete

Folio  Costo

Temas de ponencia

Temas de taller

RFC Administrador

Imagen 99. Ventana de activar paquete

## A.6 Registro de actividades

En el botón de registro de actividades muestra la ventana para registrar actividades. Para dar de alta actividades deben estar activados los paquetes para seleccionar el folio del paquete al que pertenecerá cada actividad. Se ingresan el horario y fecha en que se dará la exposición o taller, también se ingresa el RFC del ponente que lo imparte y el tema.

Acceso al Sistema

Preinscripción de alumnos Inscripcion de Alumnos Registro de ponentes Activar Paquete Registro de actividades Asistencia Alumnos Asistencia Ponente Generar constancia

### Registro de actividades

Tema: Beneficios de la robotica

Hora de inicio: 12:00:00

RFC Ponente: PEPJ891209

Tipo: Ponencia

Paquete: 1

Fecha de ponencia: 25/02/16 2016/2/6

Lugar: Auditorio 1

RFC Administrador: CRML920414

Salir Limpiar

Registrar

Imagen 100. Ventana de registro de actividades

## A.7 Asistencia de alumnos

El botón de registro de asistencia muestra una ventana para realizar una consulta. En ésta se muestran los datos del alumno. La consulta se realiza al escanear el gafete con el código de barras.

No_cuenta	Nombre	ApellidoPaterno	ApellidoMaterno	Licenciatura	Semestre
0714575	Adrián	Morales	Cruz	ICO	9

*Imagen 101. Ventana de asistencia de alumnos*

Después de realizar la consulta, se ingresan la fecha y hora de asistencia para registrarlos.

## A.8 Asistencia de ponente

El botón de registro de asistencia del ponente funciona igual al de asistencia de alumnos, con excepción de ingresar la hora, pues solo nos interesa saber si cumplió con su exposición. Para consultar los datos del ponente se ingresa su RFC.

Acceso al Sistema

Preinscripción de alumnos Inscripción de Alumnos Registro de ponentes Activar Paquete Registro de actividades Asistencia Alumnos Asistencia Ponente Generar constancia

Ingrese RFC del ponente

COMI870118 Consultar

RFC	Nombre	ApellidoPaterno	ApellidoMaterno	Gradoacademico	TemaExposicion
COMI870118	Ivan	Contreras	Maldonado	Ingeniero en sistemas c...	Beneficios de la robotica

RFC Ponente COMI870118

25/02/16 2016/2/26

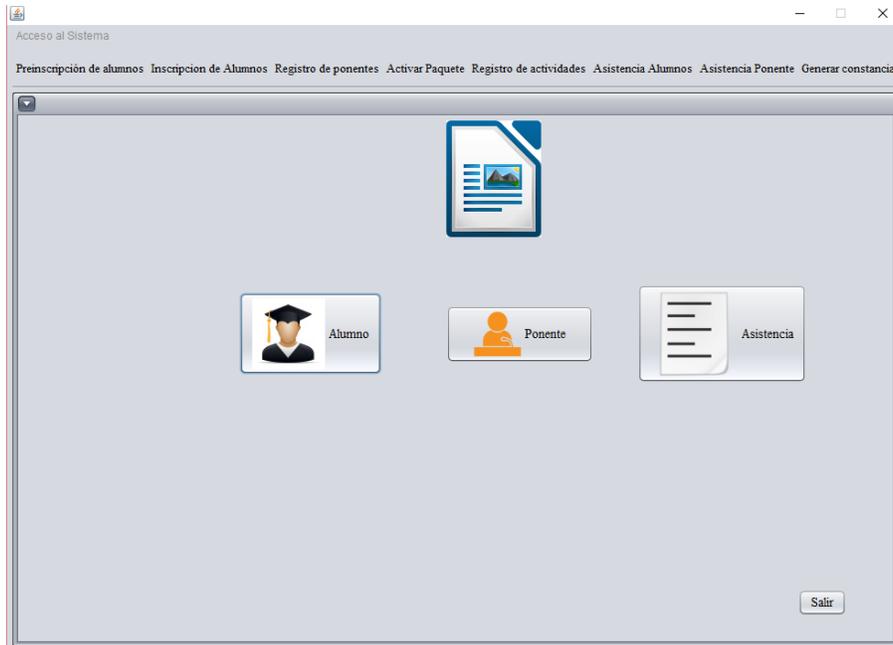
RFC Administrador CRML920414

Salir Limpiar Guardar

Imagen 102. Ventana de asistencia de ponente

## A.9 Generar constancia

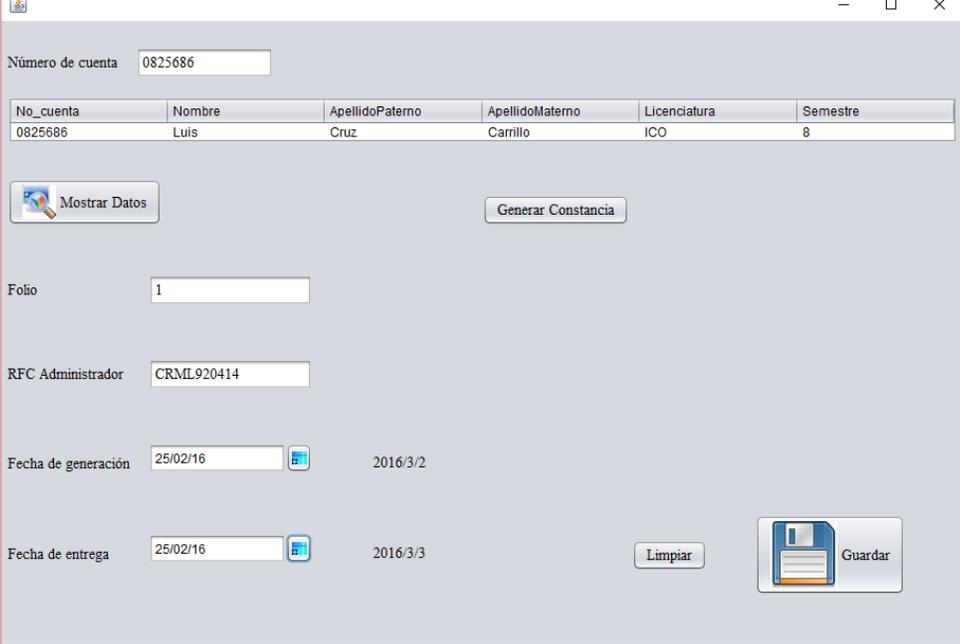
El botón para generar constancias muestra tres botones, uno para generar una constancia para los alumnos que asistieron a los eventos, otro para crear un agradecimiento a los ponentes, y finalmente uno para generar listas de asistencia.



*Imagen 103. Ventana de generar constancias*

## A.9.1 Constancia de asistencia de alumno

Para generar la constancia de asistencia del alumno se abre una interfaz para realizar una consulta. Si la consulta existe se habilita el botón “Generar constancia” Se ingresan el folio de constancia, fecha en que se genera y fecha de entrega.



Número de cuenta

No_cuenta	Nombre	ApellidoPaterno	ApellidoMaterno	Licenciatura	Semestre
0825686	Luis	Cruz	Carrillo	ICO	8

Folio

RFC Administrador

Fecha de generación  2016/3/2

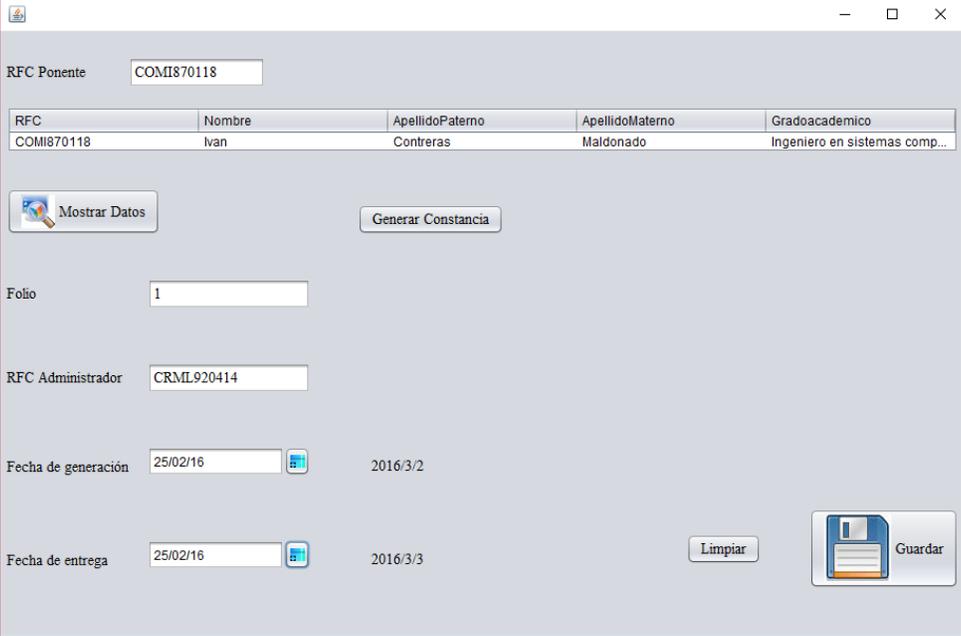
Fecha de entrega  2016/3/3

Imagen 104. Ventana de generar constancia de alumno

Al realizar la consulta, y en caso de que exista, se habilita el botón de generar constancia.

## A.9.2 Agradecimiento ponente

Para el agradecimiento del ponente se abre una interfaz similar. La consulta se realiza con el RFC del ponente.



RFC	Nombre	ApellidoPaterno	ApellidoMaterno	Gradoacademico
COMI870118	Ivan	Contreras	Maldonado	Ingeniero en sistemas comp...

Mostrar Datos      Generar Constancia

Folio: 1

RFC Administrador: CRML920414

Fecha de generación: 25/02/16      2016/3/2

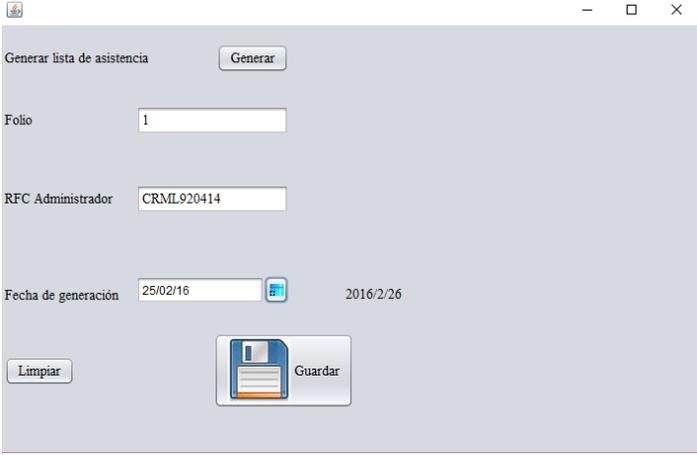
Fecha de entrega: 25/02/16      2016/3/3

Limpiar      Guardar

Imagen 105. Ventana de agradecimiento de ponente

### A.9.3 Lista de asistencia

Para la lista de asistencia se abre una interfaz para confirmar que se va a generar una lista de asistencia e ingresar folio y fecha en que se genera.



Generar lista de asistencia

Folio

RFC Administrador

Fecha de generación   2016/2/26

*Imagen 106. Ventana de lista de asistencia*