DECLARATORIA

Declaro que la investigación es absolutamente original, autentica, personal, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes. Las ideas, doctrinas resultados y conclusiones a los que he llegado de mi absoluta responsabilidad.

_____

Diego Xavier Ipiales Trujillo

CC. 1720995743

CESIÓN DE DERECHOS

Yo, Ipiales Trujillo Diego Xavier, alumno de la Escuela de Análisis de Sistemas, libre y voluntariamente cedo los derechos de autor y de mi investigación en favor del Instituto Tecnológico Superior "Cordillera".

_____

Ipiales Trujillo Diego Xavier

CC. 1720995743

AGRADECIMIENTO

Le agradezco en primer lugar a Dios, que me ha protegido y guiado durante toda mi vida.

A mis padres porque sin ellos no hubiese logrado esta importante meta.

A mi familia por apoyarme siempre en los momentos buenos y malos.

Agradezco al Instituto Tecnológico Superior "CORDILLERA" por ser mi casa de estudio y permitirme formarme como profesional.

A mis profesores quienes tuvieron la tarea de impartir tantos conocimientos en mí, guiándome en mi aprendizaje, les estaré eternamente agradecido.

A mis amigos que estuvieron presente brindándome apoyo incondicional, motivación y fuerzas para no decaer.

A mi tutor y lector: Ing. Patricia Garzón e Ing. Jaime Besantes quienes fueron guías para la elaboración de este trabajo que hoy les presento.

DEDICATORIA

Este logro se lo dedico a mis padres,

que desde niño me supieron llenar de amor y fortaleza

para ser la persona que soy hoy en día.

A mi hermano que siempre estuvo brindándome

su apoyo incondicional

y a todos mis amigos

que me supieron dar su verdadera amistad

para superarme incluso en los momento de mas tensión

INDICE GENERAL

Contenido

INDICE DE TABLAS

INDICE DE FIGURAS

RESUMEN EJECUTIVO

En la Empresa de seguridad Privada ¨LOMBEYDA CONTROL Y SEGURIDAD¨.

Las actividades que realiza como es el control de eventos públicos se realizan de forma manual, trayendo como consecuencia la pérdida de tiempo y falta de organización al momento de buscar información. Es por esta razón que se planteó el diseño de un sistema de información que permita obtener resultados favorables facilitando la ejecución de las actividades cotidianas de dicha empresa.

La presente Investigación tiene como finalidad el desarrollo de un sistema El cual es se encarga de distribuir automáticamente el personal de seguridad y que cuenta con las funcionalidades de registro de la información, carga y actualización de datos, validación de datos, generación de reportes, búsqueda de personal y administración del sistema. La realización de la misma se llevó a cabo con la metodología apoyado en las herramientas gráficas de UML técnica del Lenguaje de Modelado Unificado mostrando mediante sus diagramas, cómo será el flujo de la información en el nuevo diseño, que a su vez servirá para la posterior aplicación de un software cumpliendo así con la fase de implantación del UML.

El desarrollo de este sistema está enfocado principalmente a la reducción de los tiempos de manejo de la información, de los riesgos de perdida de información y de la generación de reportes de gestión con mayor rapidez para la toma de decisiones gerenciales efectivas, con mínimos porcentajes de error.

ABSTRACT

In the private security company ¨LOMBEYDA CONTROL Y SEGURIDAD ¨. The activities that it make like control of public event it is performed manually, consequently resulting in loss of time and lack of organization when it search information. It is for this reason that raised the design of an information system to obtain favorable results facilitating the implementation of the daily activities of the company.

This research is aimed at developing a system Which is it automatically distributes security personnel and has the features of recording information , data loading and updating , data validation , reporting , search staff and administration of the system. The realization of the same was carried out using the methodology supported UML graphical tools technique Unified Modeling Language by their diagrams showing , how will the flow of information in the new design , which in turn serve for subsequent application software thus fulfilling the implementation phase of UML .

The development of this system is mainly focused on reducing the time information management, risk of loss of data and generation of management reports more quickly for effective management decision making with minimal percentages mistake

Capítulo I: Antecedentes

1.01 Contexto

Día a día la seguridad privada se va haciendo importante en nuestro núcleo social, ya que no tan solo los oficiales de seguridad se encargan de proteger el patrimonio de las empresas, sino que también se encargan de salvaguardar la integridad de las personas.

Las empresas de seguridad deben conocer y comprender determinadas funciones como son:

- Ejercer la vigilancia y protección de bienes muebles e inmuebles, así como la protección de las personas que puedan encontrarse en los mismos.
- Efectuar controles de identidad en el acceso o en el interior de inmuebles determinados, sin que en ningún caso puedan retener la documentación personal
- Evitar la comisión de actos delictivos o infracciones en relación con el objeto de su protección
- Poner inmediatamente a disposición de los miembros de las Fuerzas y Cuerpos de Seguridad a los delincuentes en relación con el objeto de su protección, así como los instrumentos, efectos y pruebas de los delitos, no pudiendo proceder al interrogatorio de aquéllos

Es por esto, que este enfoque va dirigido hacia todas aquellas personas involucradas en los procesos de seguridad pero en especial a los desarrolladores y responsables de

las aglomeraciones de público que buscan alrededor de la seguridad en espectáculos públicos, formular herramientas técnicas y conceptuales que permitan intervenir sobre los riesgos a los que estamos expuestos y la mejor manera de superarlos, además de satisfacer las necesidades en cuestiones de seguridad para eventos. El compromiso es brindar un excelente servicio hacia el público que va a presenciar un espectáculo.

En este sentido es necesario la creación un plan de organización estratégica del personal. Esta estrategia debe ser desarrollada por los organizadores y responsables de estas actividades para identificar, prevenir y mitigar los riesgos, además de atender adecuadamente una emergencia.

La Empresa "LOMBEYDA CONTROL Y SEGURIDAD" es una empresa que se encarga r de brindar seguridad privada y tiene como objetivo la protección y seguridad de eventos deportivos y eventos públicos. Pero el desconocimiento de la organización del personal de seguridad en eventos públicos ha venido trayendo una problemática durante muchos años en los diferentes escenarios públicos del Distrito Metropolitano de Quito.

1.02 **Justificación**

En el Distrito Metropolitano de Quito se realizan diferentes eventos de asistencia de público masivo, en los cuales las diferentes empresas de seguridad privada brindan sus servicios.

La organización del personal en general para la afluencia de público implica describir la actividad, la población y las instalaciones en las que se desarrolla el evento. Se debe tener en cuenta el número de personas esperadas (aforo), el número de personas fijas en el lugar, el rango de edad, personas en condición de discapacidad.

Cuando se cuente con la totalidad de la información de la actividad se debe diligenciar el formato de presentación del plan. Para determinar el aforo esperado se deben tener en cuenta la capacidad de ocupación y la capacidad de evacuación del lugar donde se desarrollará la actividad.

Una vez definidos los valores para cada una de las capacidades, se debe tomar el menor de éstos valores como el aforo esperado para la aglomeración.

Por todo lo mencionado anteriormente tener una buena organización en una empresa de seguridad contando con una herramienta tecnológica que permita organizar de

una manera técnica el personal de seguridad que se asigna al espectáculo público es una parte fundamental para que se realicen todas las actividades con éxito.

1.03. **Definición del problema central**

Quito es una ciudad donde con mucha frecuencia se realizan espectáculos públicos. En la actualidad, las empresas de seguridad privada son las encargadas de brindar servicio de seguridad a dichos espectáculos públicos que se realizan con fines de lucro. Sin embargo los resultados esperados no han cumplido con las expectativas creadas con esta actividad incorporada, desencadena en la problemática de una inadecuada organización del personal que brinda seguridad en los espectáculos públicos. Entre las principales causas se identifican: El desconocimiento técnico sobre la distribución del personal en las diferentes localidades de los escenarios públicos, irregularidad entre el personal asignado poco capacitado en normas de seguridad, escaso apoyo por parte de la policía nacional y metropolitana, no se cuenta con un plan de organización previa al evento.

Las causas anteriores provocan: desacuerdos entre jefes y supervisores del evento, insatisfacción del público que asiste debido al caos que genera la aglomeración en los accesos al escenario donde se desarrolla el evento, el momento que se presenta un incidente el personal de seguridad no actúa de manera eficaz, se suscitan robos,

agresiones que pueden desencadenar en conflictos graves entre los asistentes, pérdida de contratos.

Es por eso la evidente necesidad de diseñar un modelo de organización técnicamente estructurado donde se contemplen sistémicamente la distribución del personal que llevará a una mayor acción en el momento que se vea algún imprevisto ya sea antes, durante o después del evento.

1.04 **Análisis de fuerzas**

Tabla 1.

Análisis de fuerzas t

ANALISIS DE FUERZAS T					
Situación Empeorada	**Situación Actual**				**Situación Mejorada**
Accidentes catastróficos en el eventos	inadecuada organización de personal de seguridad en eventos públicos del D.M.Q.				Terminar con éxito y tranquilidad los eventos públicos
Fuerzas Impulsadoras	**I**	**PC**	**I**	**PC**	**Fuerzas Bloqueadoras**
Actuacion acertada del prsonal	1	3	3	4	Desconocimiento tecnico de normas de seguridad
Coordinación entre el empresario y jefe de seguridad	2	5	3	5	Inadecuada coordinacion entre jefes y supervisores encargados del eventos
Buen uso de herramientas de seguridad	2	5	4	5	Improvisación del personal asignado de seguridad

Análisis de la matriz de fuerzas T. Luego de realizar una observación de las fuerzas bloqueadoras, se ha determinado las más relevantes tales como; improvisación del personal, inadecuada coordinación entre los superiores y desconocimiento de las normas; impiden a que la situación actual de la empresa mejore, por lo que es necesario destacar las fuerzas impulsadoras tales como: buen uso de herramientas, coordinación y la actuación acertada del personal para corregir los problemas encontrados en dicha situación, las cuales comprueban que el proyecto a desarrollarse es viable.

Capítulo II: Involucrados

2.01 Análisis Involucrados

- Jefe de empresa
- Empresarios
- Organismos de apoyo
- Publico
- Ejecutor del proyecto
- Asamblea

2.01.01. Involucrados.

Dentro de los involucrados encontramos a dos tipos, los cuales son los directos e indirectos. Los directos se encuentran interactuando de forma continua al problema presentado.

En este primer grupo encontramos el gerente de la empresa de seguridad, el mismo que está a cargo de los eventos a realizarse, esta empresa no tiene otros departamentos, por tanto no existe necesidad de funcionarios en algunas áreas.

Encontramos también al empresario que es el que realiza un evento, éstos son los encargados de contratar la seguridad para un evento.

El público asistente, son los beneficiarios de las acciones que se toman en el momento de la realización de un evento.

Los organismos de apoyo quienes son los encargados de evitar incidentes dentro y fuera del escenario donde se está llevando a cabo un programa

(Ver anexo A.01)

Tabla 2.

Análisis de involucrados

Actores involucrados	Intereses sobre el problema central	Problemas percibidos	Recursos mandatos y capacidades	Intereses sobre el proyecto
Asamblea	Dictar leyes que regulen de forma adecuada la seg. en eventos públicos	Inadecuada organización en la seg. de eventos públicos	Recursos humanos Plan nacional de seguridad ciudadana	Mejorar los niveles de seg. en el D.M.Q
Empresa de seguridad	Brindar un buen servicio de seguridad	desorganización		realizar una organización previa
Organismos de apoyo	Evitar incidentes en el D.M.Q			colaborar con la seguridad hacia el publico
Publico	mejor trato	mucho tiempo de espera para el ingreso		ingresos con tranquilidad
Empresarios	Realización de un eventos con éxito	discrepancias con la planificación de seguridad	leyes de eventos	terminar el evento sin ningún problema
Ejecutor del proyecto	Realizar un plan organizacional del personal de seguridad	inadecuada organización del personal de seguridad		mejorar la planificación del personal de seguridad

Nota: D.M.Q: Distrito Metropolitano de Quito

Capitulo III: Problemas y objetivos

3.01.- Análisis del árbol de Problemas

El árbol de problemas toma en cuenta las situaciones empeoradas que presentan en el desarrollo del presente proyecto y se presenta en la situación actual de la empresa para una pronta solución.

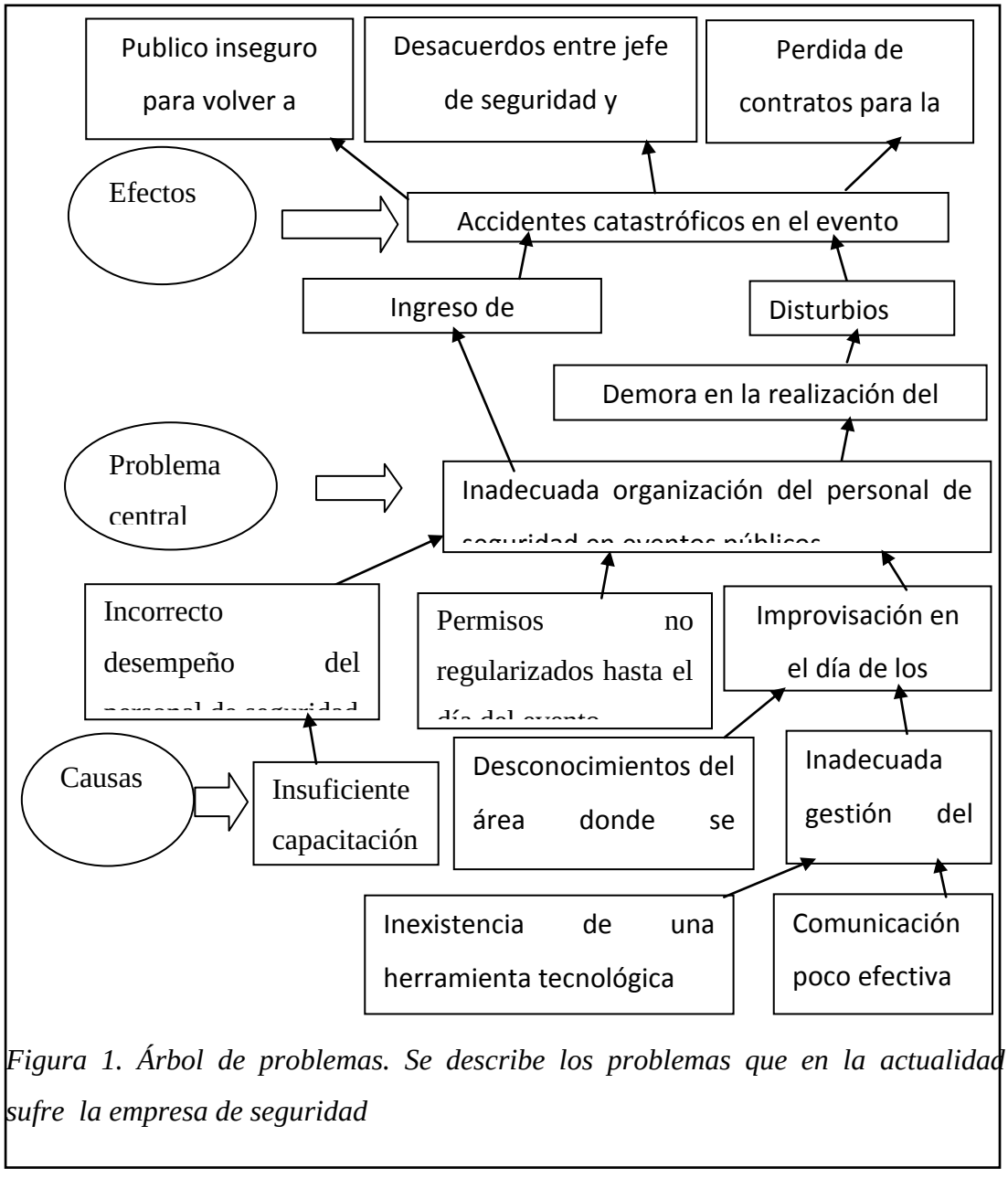


Figura 1. Árbol de problemas. Se describe los problemas que en la actualidad sufre la empresa de seguridad

Conclusiones

La inadecuada organización de personal y la insuficiente protección policial provoca el ingreso de antisociales así como de ventas ambulantes, que la mayoría de estas personas son las que causan los robos.

Esta inseguridad provoca que el público que asiste a un evento no se encuentre en conformidad durante el desarrollo de dichos eventos.

3.03. Árbol de Objetivos

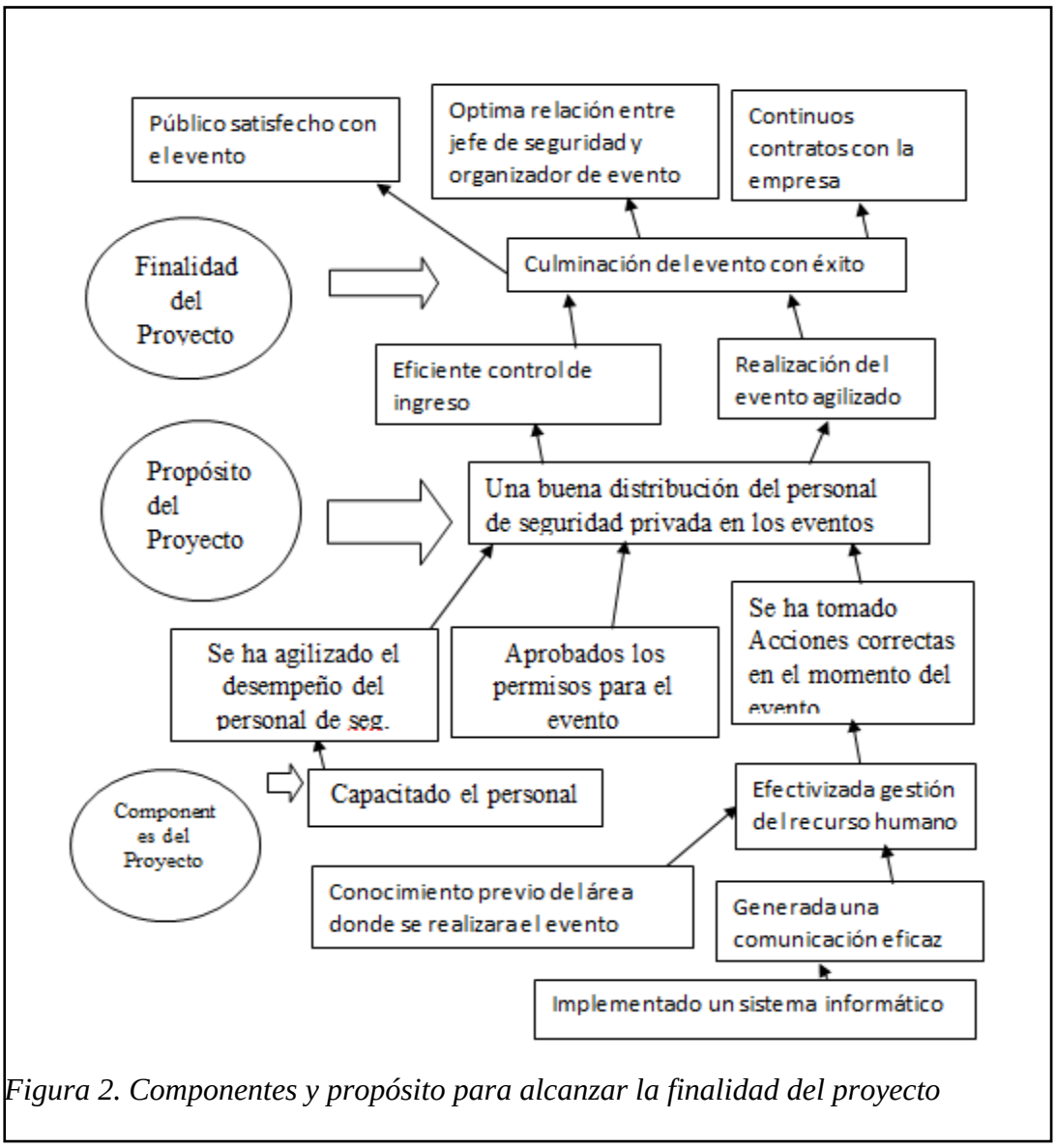


Figura 2. Componentes y propósito para alcanzar la finalidad del proyecto

Análisis del árbol de objetivos.-

La eficiente organización del personal de seguridad privada ha evitado el ingreso de antisociales logrando tener una plena seguridad en el evento, logrando la tranquilidad de los asistentes.

Se han evitado inconvenientes durante la realización del evento debido a que el personal se encuentra capacitado para tomar correctas acciones sin que afecte al público.

Capitulo IV: Análisis de alternativas

4.01 Análisis de Alternativas

En el análisis de alternativas tomamos en cuenta los objetivos que cumpliremos para finalizar con éxito el proyecto, que se ampara por medio de indicadores que demuestran porcentaje de suma importancia los mismos que interpretan la incidencia de los impactos que genera los propósitos de metas, tiene como finalidad categorizar las propuestas, para posteriormente dar una comparación sobre cuanto los objetivos están involucrados en el proyecto.

Tabla 3.
Matriz de análisis de alternativas

MATRIZ DE ANALISIS DE ALTERNATIVAS

OBJETIVOS	Impacto sobre el propósito	Factibilidad técnica	Factibilidad financier	Factibilidad social	Factibilidad política	Total	Categorías
Analizar el área designada a realizar el evento.	3	3	1	3	3	13	ALTA
Distribuir el personal asignado para cada una de las áreas del evento.	3	3	1	3	3	13	ALTA
Toma de decisiones correctas en el momento del evento.	3	3	2	3	1	12	ALTA
Culminar con éxito el evento	3	3	1	3	3	13	ALTA
TOTAL	12	12	5	12	10	51	

El Analizar el área designada para el evento se clasifica en categoría alta ya que en el negocio es muy importante que cada supervisor tenga conocimiento del área la cual estará designada a su cargo.

La distribución del personal en cada área del evento ayudara a tener el personal adecuado y equitativo en cada área dividida por supervisor, por lo tanto se clasifica en categoría alta.

La toma de decisiones correctas en el momento del evento, se realiza por cualquier altercado que suceda en el evento, se clasifica en categoría alta.

El culminar el evento con éxito se clasifica en categoría alta, ya que si se llega a cumplir será de beneficio y subirá el prestigio de la empresa lo que ayudara en futuros eventos.

4.01.02 Análisis del Impacto de los Objetivos.

Se realiza un análisis de los objetivos, se determina la factibilidad de lograrse, la relevancia, la sostenibilidad y se define la categoría de impacto que refleja cada objetivo en el proyecto como se muestra en la siguiente tabla

Tabla 4. Análisis de impacto

OBJETIVOS	Factibilidad de Lograse	Impacto Ambiental	Relevancia	Sostenibilidad	TOTAL
	1.Los beneficios son mayores que los costos 2.Se cuenta con financiamiento propio 3.Es aceptable y conveniente para la empresa Lombeyda Control y Seguridad 4.Se conoce tecnología adecuada para su realización 5. Se cuenta con soporte y seguimiento por parte de la empresa.	1. Contribuye a proteger el entorno físico. 2. Mejora el entorno social. 3. Mejora el entorno cultural. 4. Protege el uso de los recursos. 5.Favorece la educación ambiental	1.Responde a las expectativas de los beneficiarios 2.Es una prioridad sentida por los beneficiarios 3.Beneficia a grupos de mayor carencia y vulnerabilidad 4.Los beneficios son deseados por los beneficiarios	1.Fortalece la participación de los beneficiarios y población local 2.Fortalece la Organización local 3.La población está en posibilidades de aportar medios 4.Se puede conseguir financiamiento a futuro	
Total	20 Puntos	**20** puntos	**14**puntos	13 puntos	67

4.01.02.01 Conclusiones.

El análisis de impacto si influye, ya sea tanto en el impacto ambiental, como en el impacto social.

En el impacto ambiental genera el ahorro de papel, ya que en la actualidad toda la logística se realiza en papeles y es una contaminación en el ecosistema.

En el impacto social genera la tranquilidad hacia el público asistente a un evento y no surgirían anomalías como las que afectan hasta el momento.

4.03 Diagrama de Estrategias

En el diagrama de estrategias toma en cuenta el árbol de problemas y objetivos para mostrar las actividades que se van a realizar dentro de la solución al problema como la finalidad, propósito y componentes del mismo.

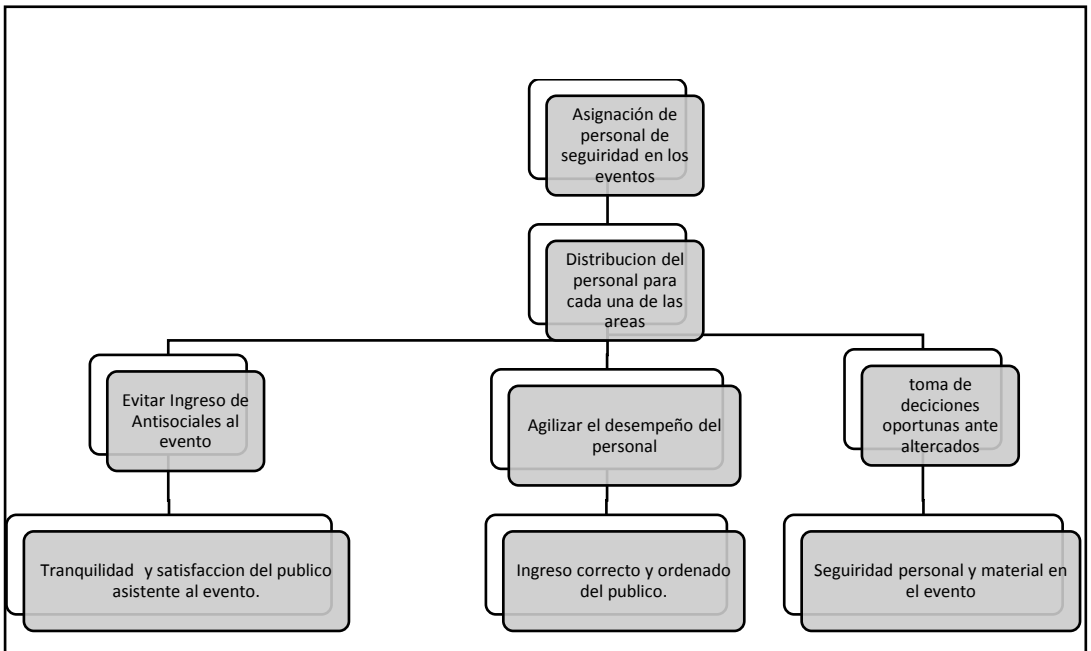


Figura 3. Esquema de estrategias para lograr la finalidad principal.

En el diagrama de estrategias tenemos como finalidad del proyecto incrementar el registro y control de los clientes, evitar las pérdidas de información verificando que nuestro propósito es alcanzar la automatización de los procesos para ello contamos con los componentes de actividades que en sí son programas que se utilizan como herramientas tecnológicas que lograrán la solución a la escases que existe en la automatización de los procesos de los eventos, éstas actividades a realizar van acorde a los programas planteados para evitar que exista redundancia de procesos.

4.04 Matriz de Marco Lógico

En el siguiente cuadro denotamos la explicación al cuadro anterior, demostrando y haciendo un breve análisis con los indicadores que muestran el estado inicial del problema y como se ha ido desarrollando esta actividad hasta en la actualidad con su respectiva explicación de cómo se ha podido verificar y los medios utilizados para la obtención de la información proporcionada, y adicional se propuso los supuestos del proyecto demostrando que situaciones se podrían presentar dentro del desarrollo del mismo.

Nota:(ver anexo 4)

Un análisis de indicadores que van desde los años anteriores hasta la fecha en curso, ya los inconvenientes presentados para demostrar que en los meses venideros tendremos un mejoramiento en todos los procesos, con el propósito de alcanzar las metas para demostrar la finalidad del proyecto, el cual nos permitirá la automatización solicitada por el jefe de la empresa y los actores directos e indirectos.

Capitulo V. Desarrollo del sistema

5.01 justificación técnica

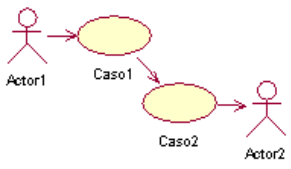
El Proyecto a desarrollar, se realiza por la necesidad que tiene la empresa de seguridad en eventos "Lombeyda Control y Seguridad", ya que no cuenta con un control de distribución del personal destinado a cada evento, optimizando así los servicios y recursos que presta la empresa.

El sistema realiza la asignación del personal de una forma automatizada, consiguiendo así que el personal se divida parcialmente en cada una de las áreas y localidades del recinto donde se realiza el evento, para ello ingresaremos en el sistema el número de áreas del evento y personas necesarias para el mismo.

5.02 Análisis y diseño

5.02.01. Diagrama de Casos de Uso

Un Diagrama de Casos de Uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. Los elementos que pueden aparecer en un Diagrama de Casos de Uso son: actores, casos de uso y sus relaciones.

Casos de Uso		Muestra un conjunto de casos de uso, los actores implicados y sus relaciones. Son diagramas fundamentales en el modelado y organización del sistema.

5.02.02. Casos de Uso

Un caso de uso es una descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica. Expresa una unidad coherente de funcionalidad, y se representa en el Diagrama de Casos de Uso mediante una elipse con el nombre del caso de uso en su interior. El nombre del caso de uso debe reflejar la tarea específica que el actor desea llevar a cabo usando el sistema.

5.02.03. Actores

Un actor es una entidad externa al sistema que realiza algún tipo de interacción con el mismo. Se representa mediante una figura humana dibujada con líneas. Esta representación sirve tanto para actores que son personas como para otro tipo de actores (otros sistemas, sensores, etc.).

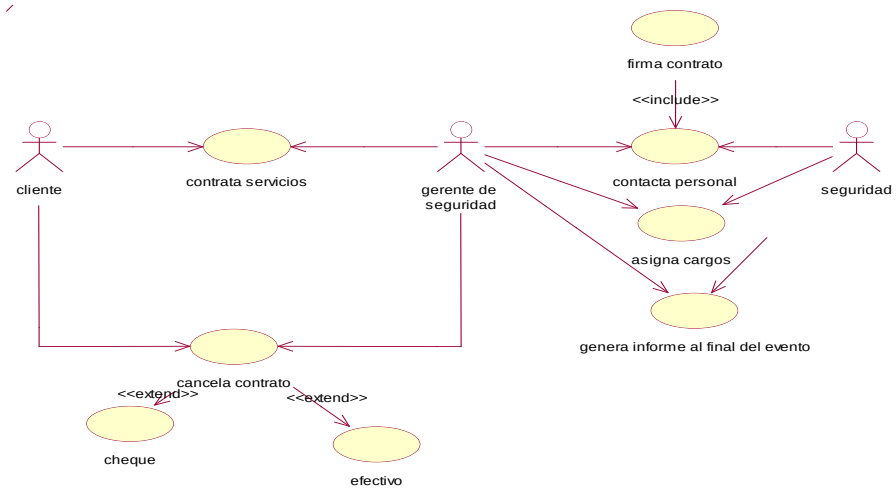


***Figura 4**. Diagrama de caso de uso general*

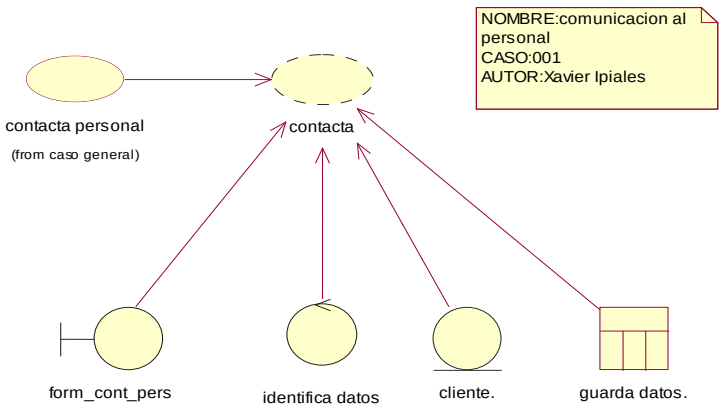


Figura 5. D.C.U: registro del personal de seguridad

Tabla 5.

Registro de datos

Nombre	Recepción de datos id: cu:001
Actores:	administrador - empleados
Descripción	El caso de usos comienza cuando el empleado envía sus datos para el registro
Flujo de eventos	1.-. se sube el archivo o digita los datos 2.-. Se asigna cargo 3.-. datos guardados en la base de datos
Post condiciones	La documentación es enviada por el empleado al administrador

Nota: proceso para el ingreso de datos

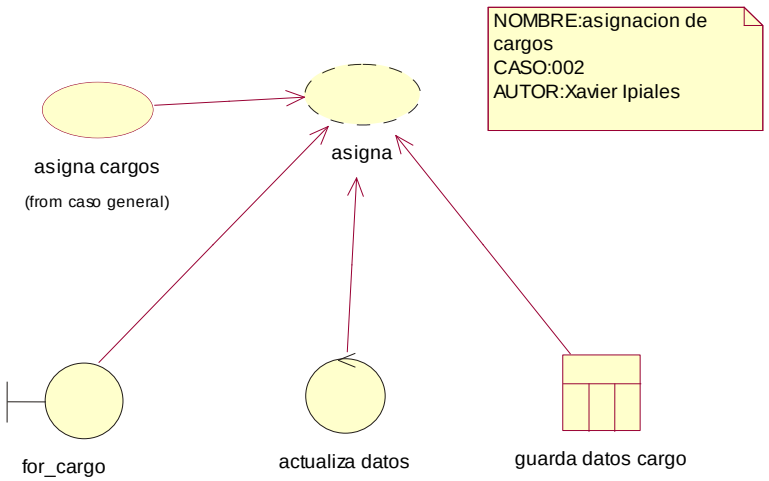


Figura 6. D.C.U: Asignación de cargos

Tabla 6.

Asignación de cargo de los empleados

Nombre	Recepción de datos id: cu:002
Actores:	administrador - supervisor
Descripción	El administrador registra el cargo o el área del supervisor 1.-. busca los datos del empleado 2.-. Se asigna área y cargo 3.-. datos guardados en la base de datos
Flujo de eventos	
Post condiciones	El administrador es el único que puede realizar esta acción

Nota: asignación de cargo para los supervisores

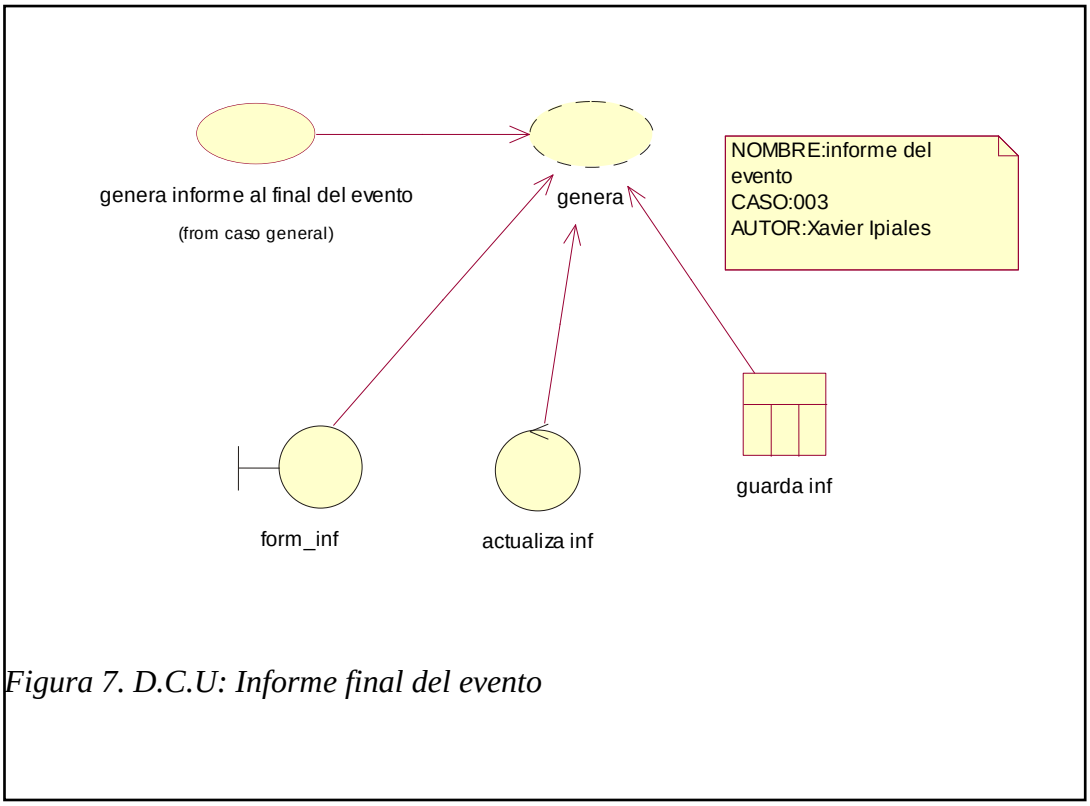


Figura 7. D.C.U: Informe final del evento

Tabla 7.

Reportes del evento ya finalizado

Nombre	Recepción de datos id: cu:003
Actores:	administrador - supervisores
Descripción	El administrador y los supervisores entregan reporte del evento para registrar
Flujo de eventos	1.-. busca los datos del evento 2.-. Se digita el detalle del evento 3.-. se actualiza en la base de datos
Post condiciones	El administrador es el único que puede realizar esta acción

Nota: se ingresa los reportes después de finalizar el evento

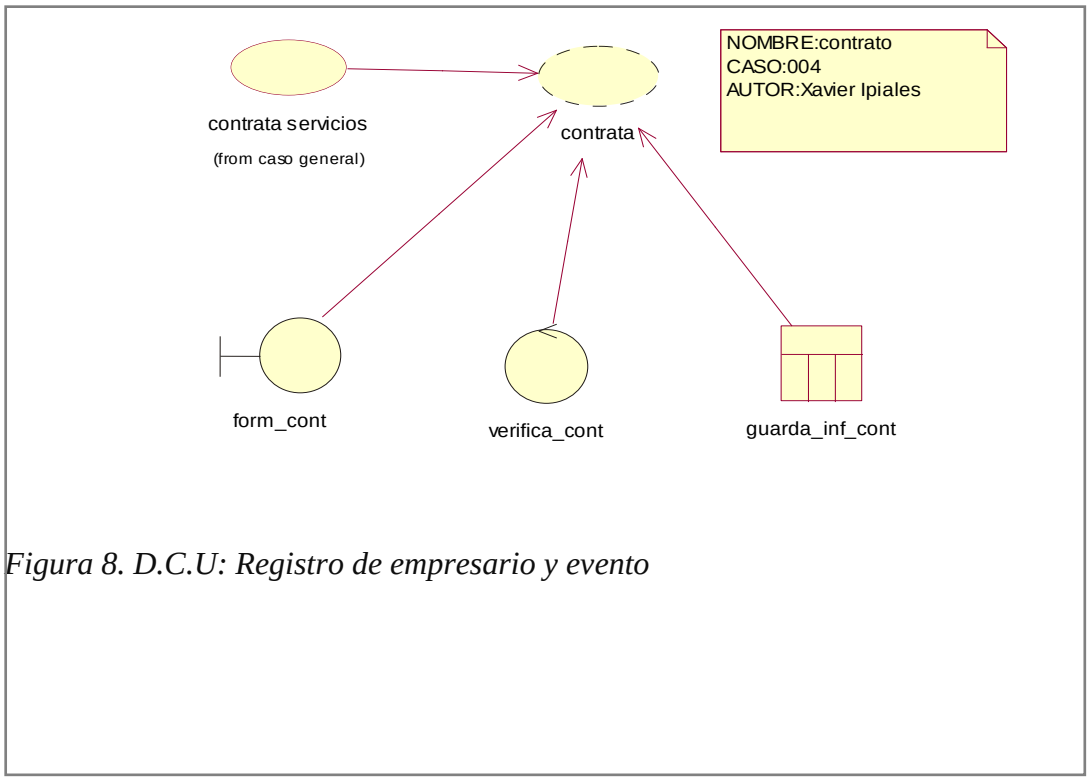


Figura 8. D.C.U: Registro de empresario y evento

Tabla 8.

Registro de datos del empresario y evento

Nombre	Recepción de datos id: cu:004
Actores:	empresario - administrador (gerente de la empresa)
Descripción	El empresario contrata los servicios de la empresa
Flujo de eventos	1.-. busca datos del empresario o ingresa datos del nuevo empresario 2.-. ingresa datos del evento 3.-. guarda datos en la base de datos
Post condiciones	El administrador es el único que puede realizar esta acción

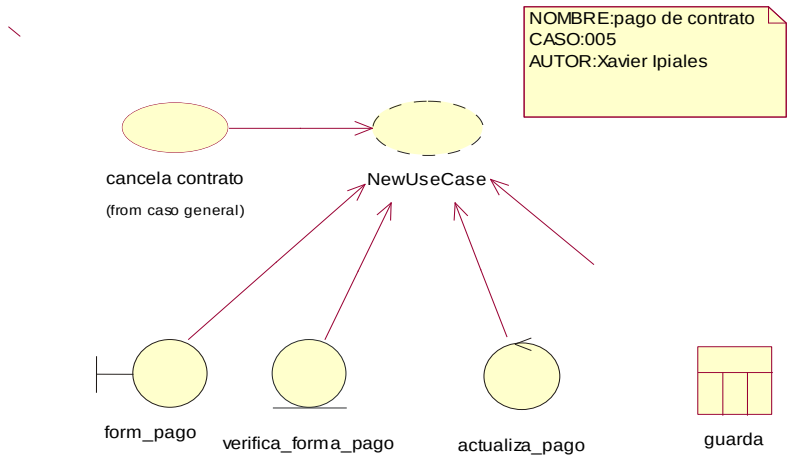


Figura 9. D.C.U: detalle de la forma de pago

Tabla 9.

Detalle de forma de pago

Nombre	Recepción de datos id: cu:004
Actores:	empresario - administrador (gerente de la empresa)
Descripción	El empresario realiza los pagos del contrato
Flujo de eventos	1.-. busca datos del evento 2.-.detalla la forma de pago (cuantas partes) 3.-. finaliza el evento 4.-. actualiza el detalle del pago del evento 3.-. guarda datos en la base de datos
Post condiciones	El administrador es el único que puede realizar esta acción

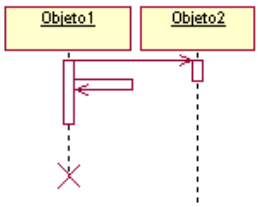
Nota: ingreso de la forma de pago del evento

5.02.02 Diagrama de Secuencia

Un diagrama de Secuencia muestra una interacción ordenada según la secuencia temporal de eventos. En particular, muestra los objetos participantes en la interacción y los mensajes que intercambian ordenados según su secuencia en el tiempo.

El eje vertical representa el tiempo, y en el eje horizontal se colocan los objetos y actores participantes en la interacción, sin un orden prefijado. Cada objeto o actor tiene una línea vertical, y los mensajes se representan mediante flechas entre los distintos objetos. El tiempo fluye de arriba abajo.

Se pueden colocar etiquetas (como restricciones de tiempo, descripciones de acciones, etc.) en el margen izquierdo o junto a las transiciones o activaciones a las que se refieren.

Secuencia		Son diagramas de interacción, muestran un conjunto de objetos y sus relaciones, así como los mensajes que se intercambian entre ellos. Cubren la vista dinámica del sistema. El diagrama de secuencia resalta la ordenación temporal de los mensajes,

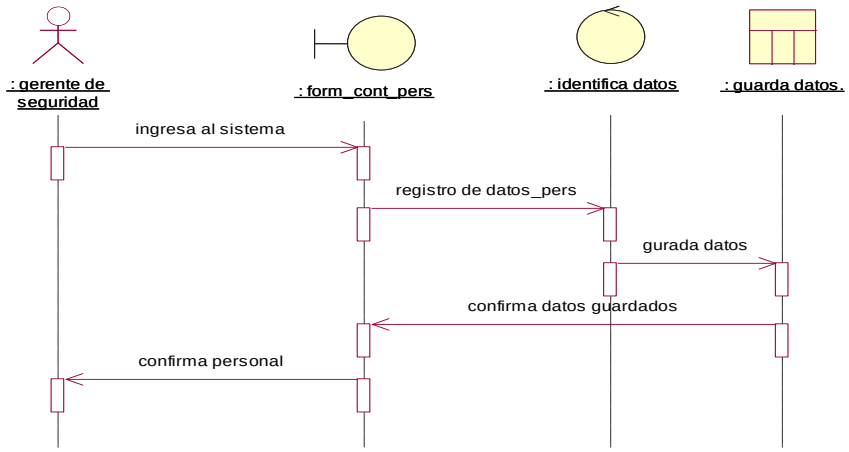


Figura10. D.S: Registro de datos del empleado

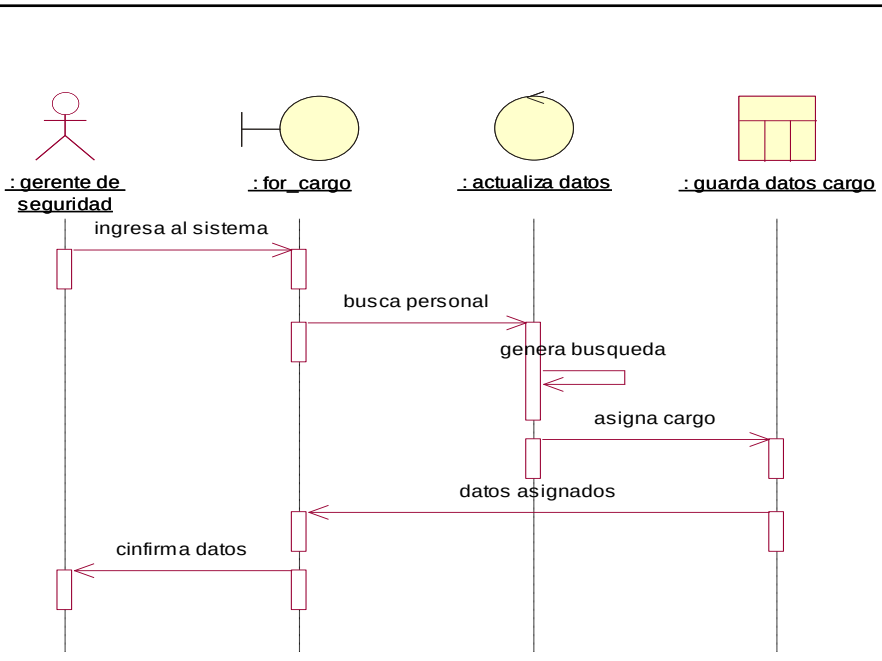


Figura 11. D.S: Asignación de cargos de los empleados

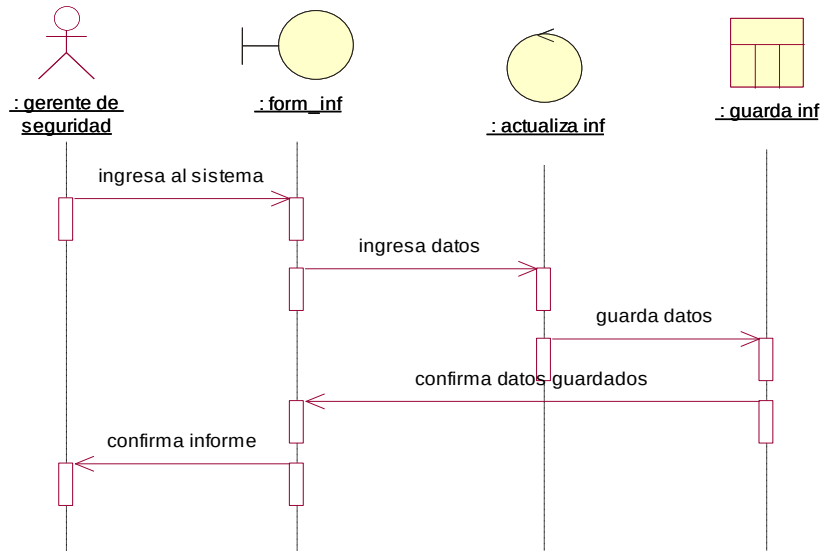


Figura 12. D.S: Informe final del evento

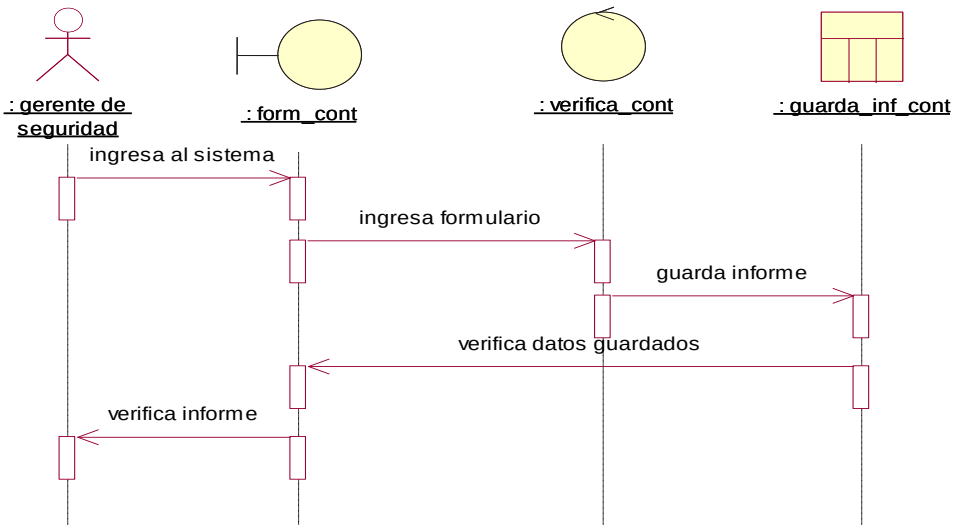


Figura 13. D.S: Registro de contrato, empresario y evento

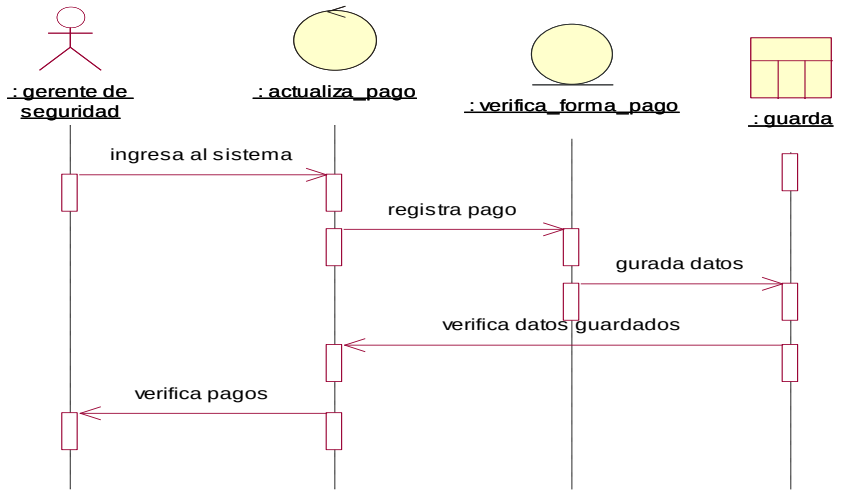
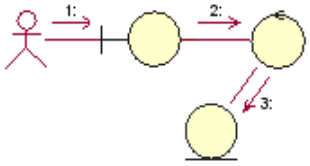


Figura 14. D.S: Informe del pago del evento

5.02.03 Diagrama de Colaboración

Un Diagrama de Colaboración muestra una interacción organizada basándose en los objetos que toman parte de la misma, y los enlaces entre ellos (en cuanto a la interacción se refiere). A diferencia de los Diagramas de Secuencia, los Diagramas de Colaboración muestran las relaciones entre los roles de los objetos. La secuencia de los mensajes y los flujos de ejecución concurrentes deben determinarse explícitamente mediante números de secuencia.

Colaboración		el diagrama de colaboración resalta la organización estructural de los objetos

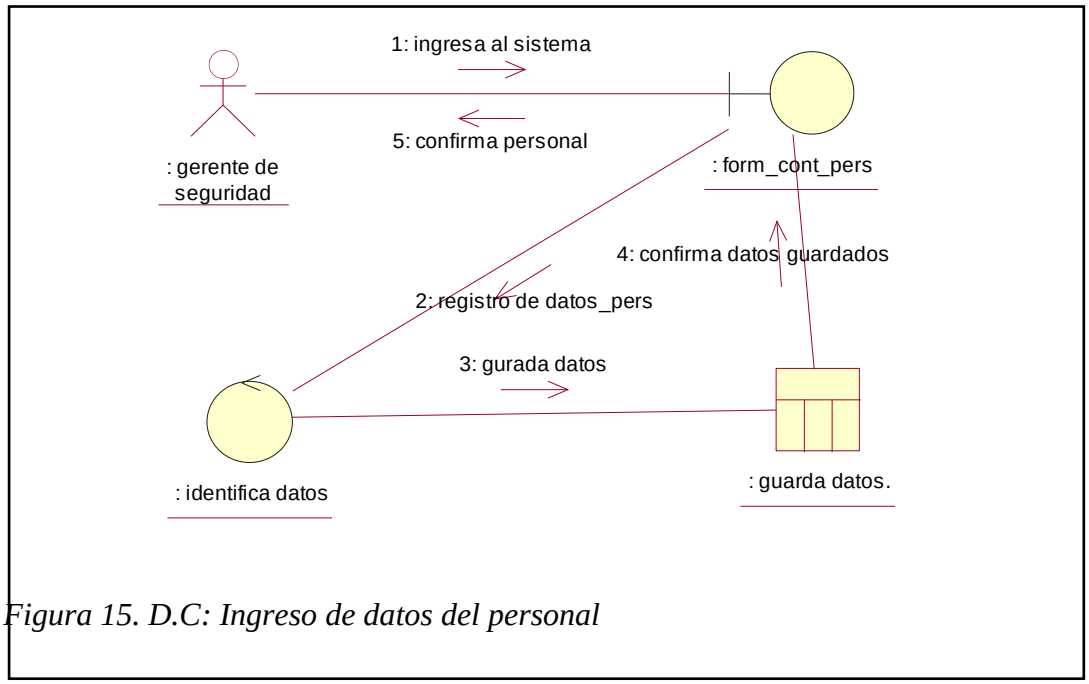


Figura 15. D.C: Ingreso de datos del personal

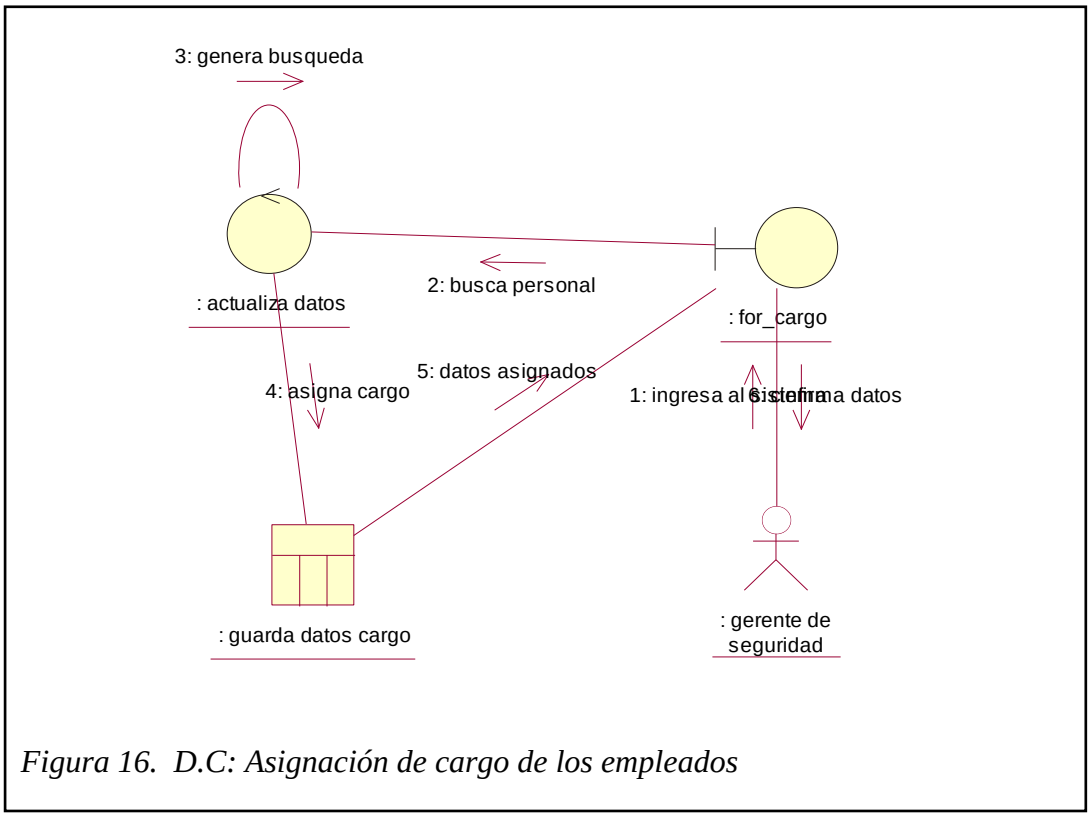


Figura 16. D.C: Asignación de cargo de los empleados

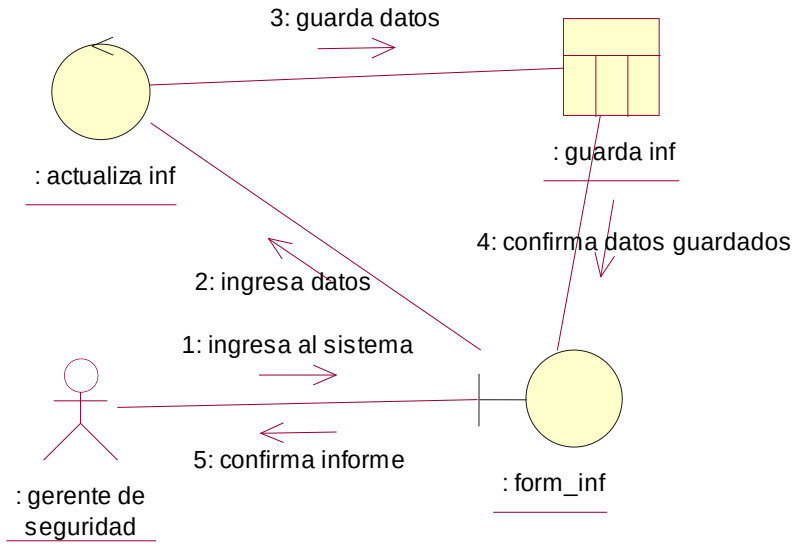


Figura 17. D.C: Detalle final para el registro del evento

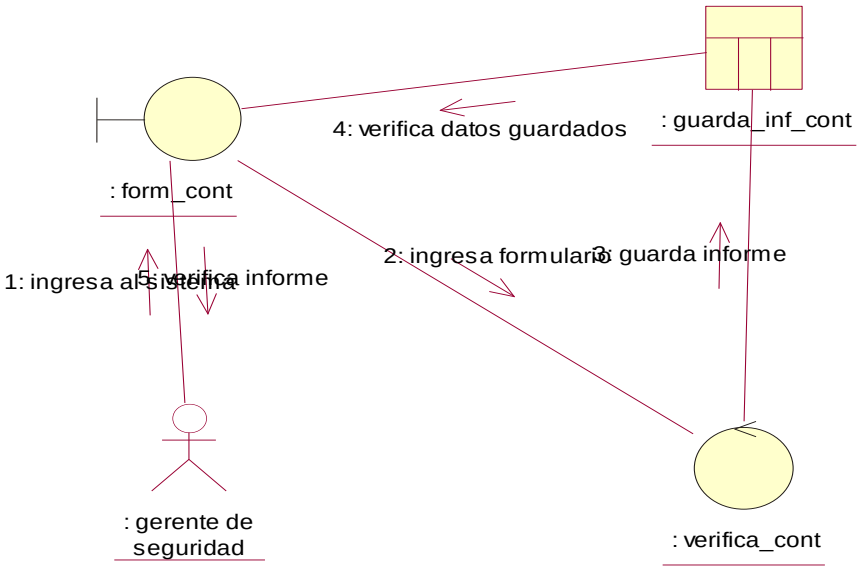


Figura 18. D.C: elementos para ingreso de datos del empresario y evento

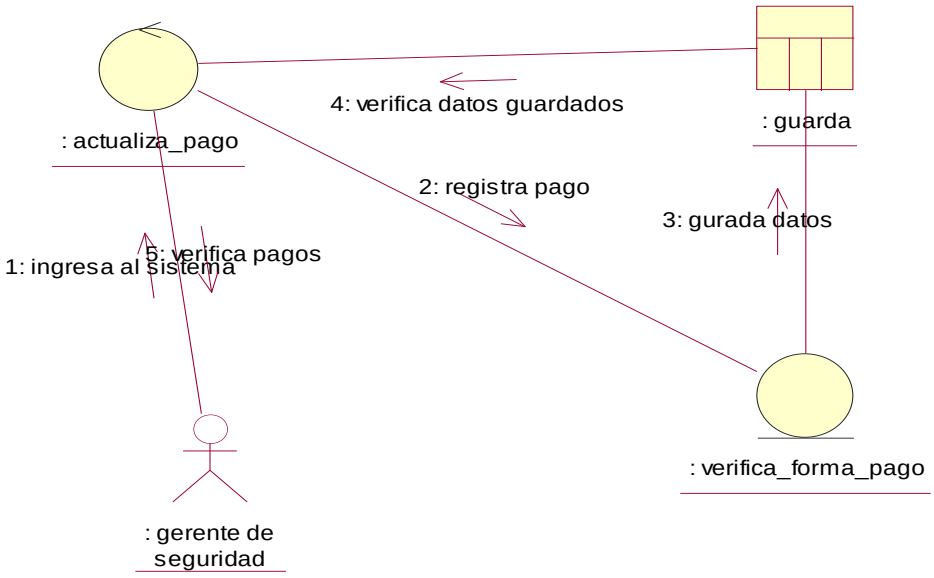


Figura 19. D.C: elementos que se utilizan para el detalle de pago del evento

5.02.04Diagramas de Componentes

Un diagrama de componentes muestra la organización y las dependencias entre un conjunto de componentes.

Para todo sistema se han de construir una serie de diagramas que modelan tanto la parte estática (diagrama de clases), como dinámica (diagramas de secuencia, colaboración, estados y de actividades), pero llegado el momento todo esto se debe materializar en un sistema implementado que utilizará partes ya implementadas de otros sistemas, todo esto es lo que pretendemos modelar con los diagramas de componentes.

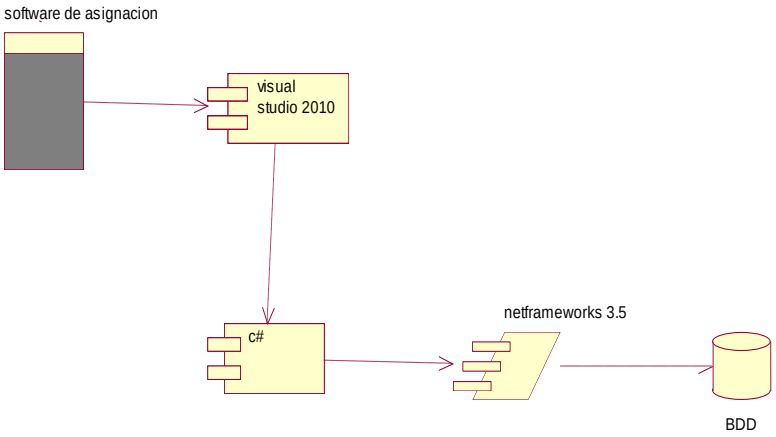


Figura 20. El diagrama de componente Muestra la organización y dependencias de un conjunto de componentes.

5.02.05 Diagrama físico

5.02.05.01. Componentes

Los componentes pertenecen al mundo físico, es decir; representan un bloque de construcción al modelar aspectos físicos de un sistema.

Una característica básica de un componente es que:

"debe definir una abstracción precisa con una interfaz bien definida, y permitiendo reemplazar fácilmente los componentes más viejos con otros más nuevos y compatible

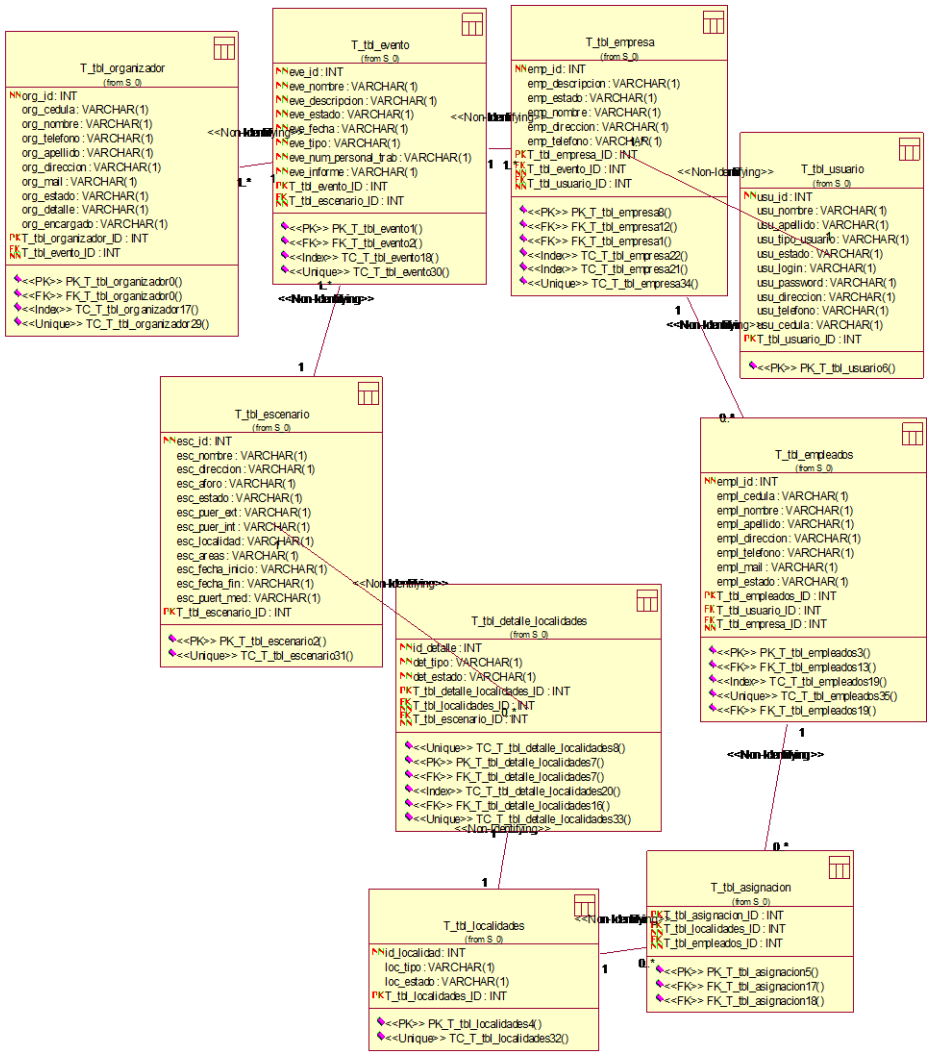


Figura 21. Diagrama físico.- Este diagrama tiene como objetivo conseguir una instrumentación, lo más eficiente posible, del esquema lógico, donde describe los componentes, de hardware y de software, que se desplegarán en el ambiente seleccionado. En este caso, empleando el gestor de la DDBB, se implementan las tablas de las DDBB con sus características, organización y estructuras de almacenamiento interno.

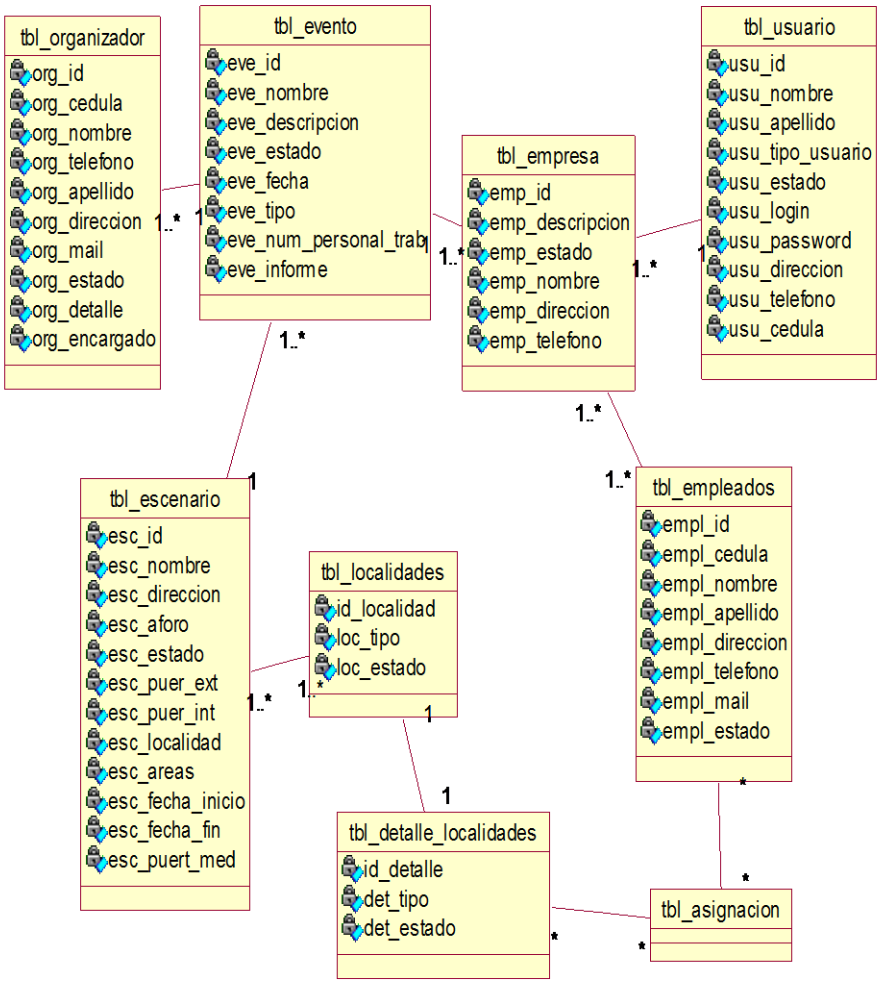


Figura 22. Diagrama Lógico.-El diagrama lógico de la base de datos, es una descripción de la estructura de la base de datos que puede procesar un SGBD. Al mismo tiempo adaptarlo al modelo de datos que se va a utilizar. Transformando las entidades y relaciones en tablas.

5.03. Desarrollo

5.03.01 arquitectura de software

Arquitectura en 3 capas

La arquitectura en 3 capas, es la más común en sistemas de información que además de tener una interfaz de usuario contemplan la persistencia de los datos.

Una descripción de los tres niveles sería la siguiente:

Nivel 1: Presentación ventanas, colores, texto de ingreso, botones, informes

Nivel 2: Lógica de la Aplicación – conexión a la base de datos, manejo del proceso transaccional y todo aquello que tenga que ver con las reglas del negocio

Nivel 3: Almacenamiento – almacenamiento de información, procedimientos, funciones, triggers, etc

5.03.01.01. Arquitectura de tres niveles orientadas a objetos

a) *Descomposición del nivel de lógica de la aplicación*

En el diseño orientado a objetos, el nivel de lógica de la aplicación se descompone en sub- niveles que son los siguientes:

Objetos del Dominio: son clases que representan objetos del dominio. Por ejemplo en un problema de ventas, una "Venta" sería un objeto del dominio.

Servicios: se hace referencia a funciones de interacción con la base de datos, informes, comunicaciones, seguridad, etc.

5.04 estándares de programación

.5.04.01 Estándares de Programación.

Dentro del desarrollo de la programación de va manejar distintos tipos de objetos los cuales se los manejara con la siguiente nomenclatura que se detalla a continuación. La siguiente tabla muestra los estándares de programación aplicados.

Tabla 10.

Estándares

Tipo de Objeto	Nomenclatura
TextBox	Txt_ **Ejemplo:** Txt_Nom
Button	Btn_ **Ejemplo:** Btn_Gua
CheckBox	Chk_ **Ejemplo:** Chk_Sel
DropDownList	Ddl_ **Ejemplo:** Ddl_Cri
FileUpload	Fup_ **Ejemplo:** Fup_CargarArchivos
Image	Img_ **Ejemplo:** Img_Gua
Label	Lbl_ **Ejemplo:** Lbl_Me
DataGrid	DGR_ **Ejemplo:** DGR_Usu
UpdatePanel	Udp_ **Ejemplo:** Udp_Usu

Nota: Estándares que se utilizan en el lenguaje de programación para realizar el sistema.

5.04.02Nombres de Clases.

El nombre de la clase debe ser significativo.

Tabla 11.

Nombre de las clases

Nombre	**Nomenclatura**
Métodos	Class Métodos
UsuarioInfo	Class UsuarioInfo

Nota: se describe la forma de nombrar a cada clase

5.04.03Nombres de las Funciones y Procedimientos.

El nombre de la funciones debe de ser descriptivo a la acción que raleza, debe contener tipo de acceso, dato de retorno, nombre de la función, y argumentos si los necesitase.

Tabla 12.
Nombres de Funciones y Procedimientos

Nombre	**Nomenclatura**
Obtener lista de usuarios	Public List<UsaurioInfo> ObtenerUsuarioLista(){}
Obtener la información de usuario por código.	Public UsuarioInfo ObtenerUsuarioInfoXId(int id_Usuario){}

5.04.04 Nombres de Variables.

Las variables dependiendo su alcance, se dividen en variables de Sesión, y variables de Aplicación. Sus nombres deber de estas con la siguiente nomenclatura tipo de dato y nombre de la variable.

Tabla 13.

Nombres de Variables

Nombre	**Nomenclatura**
Variable de sesión rol	Session.add("sesión_Rol", tipo de dato);
Variable de aplicación	Int código;

5.05. Estándares de Base de Datos.

Los objetos de base de datos deberán estar debidamente documentados con las siguientes propiedades.

Descripción: debe contener la descripción del uso o la funcionalidad el objeto y que no sea una simple repetición del nombre.

5.05.01 Tipos de Datos.

La descripción de los tipos de datos a utilizar dentro del diseño de la base de datos debe ser clara y precisa.

Tabla 14

Tipos de Datos

Tipo de Dato	Cuando se debe utilizar
VARCHAR	Para campos de texto de tamaño variable de hasta 5000 caracteres. Siempre se utilizará VACHAR2 en lugar de VARCHAR.
CHAR	Para campos de texto de tamaño fijo, por ejemplo para el uso de estados (SI/NO), (A/I).
NUMBER	Para campos numéricos. Siempre se especificará la precisión, por ejemplo para un número de 5 cifras enteras y 2 decimales será: NUMBER (7,2).
DATE	Para campos de fecha y de fecha y hora.
CLOB	Para campos de texto de más de 5000 caracteres.
BLOB	Para almacenamiento de archivos binarios, por ejemplo imágenes, archivos pdf, Word, etc.
XMLTYPE	Para campos en formato XML sobre los cuales se ejecutarán consultas con XPATH desde la Base de Datos. Si no se requerirá utilizar el motor de XML de la Base de Datos, es preferible utilizar un campo CLOB

Para los nombres de las tablas de rompimiento se considerará el nombre de las tablas involucradas en el rompimiento tomando las tres primeras letras de cada

tabla o sí llegara a coincidir las tres primeras letras se toman cuatro de cata tabla de rompimiento.

El nombre de la tabla se escribirá en minúsculas ya que el gestor de base de datos MySQL al momento de recuperar el script los escribe así, y se escribirá en singular con algunas excepciones que pueden presentarse para una mejor descripción de la tabla.

5.06. Interfaces de software.

Para interfaz de software se instalara el aplicativo Visual Studio 2010 y el gestor de base datos MySql 5.5. Microsoft information enterprice edition Para poder administrar las tablas y toda la información del sistema.

5.07 casos de pruebas

Caso de Prueba 1. Ingreso al sistema
Actor: administrador
Condiciones: El administrador debe estar registrado
Propósito: Ingresar a la pantalla principal del sistema
Escenario: Ingreso al sistema

Tabla 15**.**
Caso de prueba 1

Sección	Actividad	Clase de equivalencia	Resultado
➢ **Login**	➢ Ingresa el nombre de usuario ➢ Ingresa la clave	➢ El usuario debe estar registrado ➢ El usuario solo puede digitar la clave 3 veces	➢ El usuario ingresa a la pantalla principal del sistema

Nota: ingreso al sistema

Caso de Prueba 2. Registro del personal de seguridad

Actor: Empleado

Condiciones: El administrador debe estar logueado en el sistema

Propósito: Ingresar los datos del empleado al sistema

Escenario: Registro_empleado

Tabla 16.

Caso de prueba 2

Sección	Actividad	Clase de equivalencia	Resultado
1	➢ Ingreso de datos Personales de los empleados	➢ La cedula tiene que ser validada	➢ Los datos de los empleados son correctos y se cargan en la base de datos
2	➢ Cedula repetida	➢ Error al cargar nuevo empleado	➢ Mensaje de error "cedula ya ingresada" ➢ El dato no se registra en la base de datos
3	➢ Importar archivo Excel	➢ Examinar el archivo a importar	➢ Subir lista de datos de los empleados
4	➢ Guardar archivo	➢ Registro guardado con éxito	➢ Mensaje "registro guardado" ➢ Los datos son ingresados en la base de datos

Nota: ingreso de datos de las personas para trabajar en un evento

Caso de Prueba 3. Registro de datos del empresario y del evento

Actor: Empresario

Condiciones: El administrador debe estar logueado en el sistema

Propósito: Ingresar los datos del Empresario y evento al sistema

Escenario: Registro_empresario y Registro_evento

Tabla 17.

Caso de prueba 3

Sección	Actividad	Clase de equivalencia	Resultado
1	➢ Ingreso de datos Personales del nuevo empresario	➢ La cedula tiene que ser validada	➢ Los datos de los empleados son correctos y se cargan en la base de datos
2	➢ Cedula repetida	➢ Error al cargar nuevo empleado	➢ Mensaje de error "cedula ya ingresada" ➢ El dato no se registra en la base de datos
3	➢ Guardar cliente	➢ Registro guardado	➢ Los datos se guardan en la base de datos
4	➢ Ingreso de datos del evento del cliente	➢ selección del empresario para el registro	➢ verificación de datos
5	➢ escenario y fecha de evento	➢ ingrese fecha del evento y seleccione el escenario	➢ verificación los datos
5	➢ guardar evento	➢ registro guardado	➢ los datos del evento se guardan en la base de datos
6	➢ datos incorrectos	➢ error al guardar datos	➢ mensaje de error "fecha o escenario ya registrado" ➢ verificar la fecha o el escenario que no tenga otro evento para esa fecha en el mismo escenario.

5.08. Pruebas de modulo

5.08.01. Pruebas de Interfaz de Usuario.

Esta prueba se realiza inicialmente verificando facilidad con la que el usuario se desenvuelve en realizar las distintas operaciones en el sistema, y se obtuvo que el tiempo de respuesta es óptimo en el desenvolvimiento en el registro de información así como la entrega de resultados específicamente en la verificación de datos, ya esa interfaz es la que maneja mayor obtención de información de la base de datos.

En este formulario verificamos que el salto de las cajas de texto se las realice con tab, igualmente se cumple con la verificación de los estándares GUI que implica color de fondo de las cajas de texto, color de texto de la recuperación de datos simetría en la distribución de cajas recuperadoras de información, diseño de ubicación de objetos de interfaz de usuario (text box, check box etc).

De igual manera se verifica el estándar de comunicación que existe entre los datos estableciendo tiempos de respuesta en la recuperación de información desde la base de datos hacia la aplicación y desde el aplicativo hacia la base de datos.

Por otro lado se determinó una prueba de ejecución del manejo de los iconos inicialmente que estén acorde con la información solicitada; en este punto se verifico ventanas y mensajes de alerta, ventanas y mensajes de información de la acción que se ha ejecutado, ventana y mensajes de captura de errores con el manejo de excepciones.

5.08.02. Pruebas de Desempeño.

Con la finalidad de poder ejecutar este tipo de pruebas y que las mismas tienen incidencia con la arquitectura montada, fue necesario estructurar el sistema, en un ambiente distribuido que claramente se pueda evidenciar el servidor de datos, servidor de aplicaciones. La primera prueba realizada fue la verificación del tiempo de respuesta del cliente hacia el servidor de aplicaciones y posteriormente del servidor de aplicaciones al servidor de datos, el tiempo de respuesta final se lo determina con la suma total de tiempo realizado desde la petición de información hasta la recepción de la misma y esto divido por dos.

Se considera que el servidor debe estar separado para poder realizar las pruebas, de esta manea podremos verificar el tiempo de respuesta de acceso remoto a la aplicación.

Otra de las pruebas fue la verificación del tiempo de respuesta en las transacciones en la que tiempo se demora.

Continuando con las pruebas de desempeño tenemos que verificar tiempos validos de respuesta para recibir y enviar información mediante el XML igualmente el tiempo de respuesta que se demora en generar este tipo de información.

5.08.03 Pruebas de Carga.

Para realizar esta prueba es necesario ejecutarla sobre una sola tabla la misma que es la que tiene más información dentro del en el proyecto la tabla "Tbl_empleados" con un soporte mínimo de 50 personas. Se medirá la persistencia de las tablas y de la programación para así obtener la persistencia de la transaccionalidad del sistema

5.08.04. Prueba de Estrés.

Esta prueba la realizamos con por lo menos 100 sesiones abiertas, en cada consulta se procedió a obtención de la información de los procesos que interviene en los registros. Para esta prueba se utilizó bucles repetitivos que simulaba que el ingreso masivo de información hacía una sola tabla.

5.08.05. Prueba de Volumen.

Esta prueba se la realiza después de un año de ejecución. Para poder determinar la cantidad e información con la que está trabajando es sistema. Para así poder determinar si el sistema puede trabajar con gran cantidad de información.

5.08.06. Prueba de Seguridad.

La interrupción abarca un amplio rango de actividades se coloca interrupción en los programas para comprobar que su desempeño sea idóneo igualmente se colocó interrupción en el código para verificar sus errores; en los accesos alas base de datos

se colocaron banderas para verificar el acceso a las mismas; todo el código tiene un acto grado de manejo de excepciones para la captura de errores y de rendimiento de la aplicación.

5.09. Diseño de Casos de Prueba

Se debe diseñar pruebas que puedan generar el mayor número de errores posibles con una cantidad mínima de tiempo.

5.09.01. Pruebas de caja Blanca.

Las pruebas de caja blanca se las realiza minuciosamente de los detalles procedimentales del sistema. Introduciendo pequeños bloques de código o condiciones que luego serán ejecutadas. Esto no garantiza un programa 100% funcional ya que existen un número indefinido de caminos lógicos los cuales no se pueden llegar a controlar.

5.09.02 Pruebas de Cajas Negras de Unidad de Sistema de Integración.

Estas pruebas se las realiza sobre la interfaz del software, se pretende demostrar que las funciones del software son operativas el ingreso de información es correcto así como la salida se mantiene la integridad de la información.

5.09.03. Pruebas de Unidad.

Verifica si el diseño fue realizado acorde a las relaciones existentes entre las tablas de la base datos. Enfocándose en la información que almacena la tabla de control de procesos de producción tanto ingresos como salidas.

Se concentra en la unidad más pequeña de diseño del software. Así como en lo más fundamental que es la lógica del negocio en que el envío y recepción de datos cumplan con los requerimientos y cálculos del negocio.

Esta prueba se realizó y el sistema mantuvo la consistencia e integridad de la información.

5.09.04. Pruebas de Integración.

Lo principal de esta prueba el verificar el correcto ensamblaje entre los distintos componentes una vez que hayan sido probados unitariamente con el fin de probar que interactúan correctamente a través de sus interfaces, tanto internas como externas.

5.09.05. Pruebas de Validación.

El objetivo principal de las prueba de validación es obtener información útil para la validación de la implementación. Una vez que el sistema ha cumplido con la

verificación de errores por lo tanto está libre de errores en tiempos de ejecución, lo que significa que está libre de errores lógicos.

El módulo de seguridad es donde se realiza la prueba de validación pues controla el acceso al sistema validando y verificando que los datos ingresados sean los correctos.

Se verifica que existan mensajes acorde a las acciones erróneas que se realizan en el ingreso de datos del sistema. La validación del software se la realiza con pruebas de verificación que demuestren que es 100% funcional.

Capítulo VI Recursos, presupuesto y cronograma

6.01 Recursos

Tabla 18.

Recursos materiales

TIPO	RECURSO
Recurso Intangible	Computadora
Recurso Intangible	Impresora
Recurso Renovable	Hojas
Recurso Intangible	Impresiones
Recurso Tangible	Transporte
Recurso Tangible	Alimentación
Recurso Didáctico	Internet
Recursos Humanos	Experiencias
Recursos Humanos	Conocimientos
Recursos Humanos	Entrevistas
Recursos Económicos	Proyecto

Nota: se detalla todos instrumentos utilizados para el desarrollo

6.02 Presupuesto

Tabla 19.
Presupuesto

RECURSO	VALOR UNITARIO	VALOR TOTAL
Proyecto (Seminario,Tutorías)		720.00
Impresiones B/N	0,10	36.00
Impresiones Color	0,30	65.00
Internet		100.00
Transporte	0,25	30.00
Alimentación	2,50	60.00
Otros Gastos		100.00
Anillado y Empastado		40.00
VALOR TOTAL		1151.00

Nota: se describe todo el costo del desarrollo del sistema

6.03 Cronograma

(ver anexo A.02)

Capitulo VII: Conclusiones y recomendaciones

7.01 Conclusiones

I.- El análisis del sistema actual a través de los diagramas de casos de uso y de objetos, facilitó la determinación de los requerimientos ya que a través de ellos se realizó la identificación, definición y recolección de los verdaderos requisitos necesarios para el desarrollo de la aplicación, que se centran en el usuario y sus necesidades.

II.- Para el diseño del sistema se utilizó como herramienta UML, que a través de sus diagramas se pudo crear la estructura del software del sistema. Con los modelos de caso de uso se representó la forma de cómo el usuario interactúa con el sistema y todas las operaciones que el usuario necesita que éste haga. Así como también, los de análisis y colaboración muestran las relaciones entre los roles de los objetos y, los de diseño que ayudaron a determinar los atributos, método y relaciones entre los objetos que operan en el sistema.

III.- Para la segunda fase se logró establecer de forma inmediata una arquitectura del sistema sólida y lista para ser puesta en práctica debido a que los diagramas que se realizaron permitieron concretar los flujos y vías con la se trabajaran los datos para generar una información veraz al usuario. Por lo anteriormente descrito y junto a los diagramas de casos de uso implementados se llevó a cabo la fase de elaboración.

IV.- El adecuado diseño del modelo de la base de datos resultó fundamental en el desarrollo del software, ya que el mismo sirvió de base para erigir la arquitectura del sistema.

V.- Las pruebas de integración permitieron un análisis más detallado a las funcionalidades del sistema y su correcto funcionamiento.

7.02 Recomendaciones

I.- Realizar la debida promoción del sistema, para la participación total de los empleados y obtener de esta forma una información siempre actualizada.

II.- Realizar el mantenimiento preventivo con el fin de mantenerse lo más actualizado posible.

III.- Realizar pruebas con los usuarios finales del sistema que permitan determinar si la totalidad de sus requerimientos fueron alcanzados, en caso contrario estudiar la culminación de los requerimientos que carece el sistema.

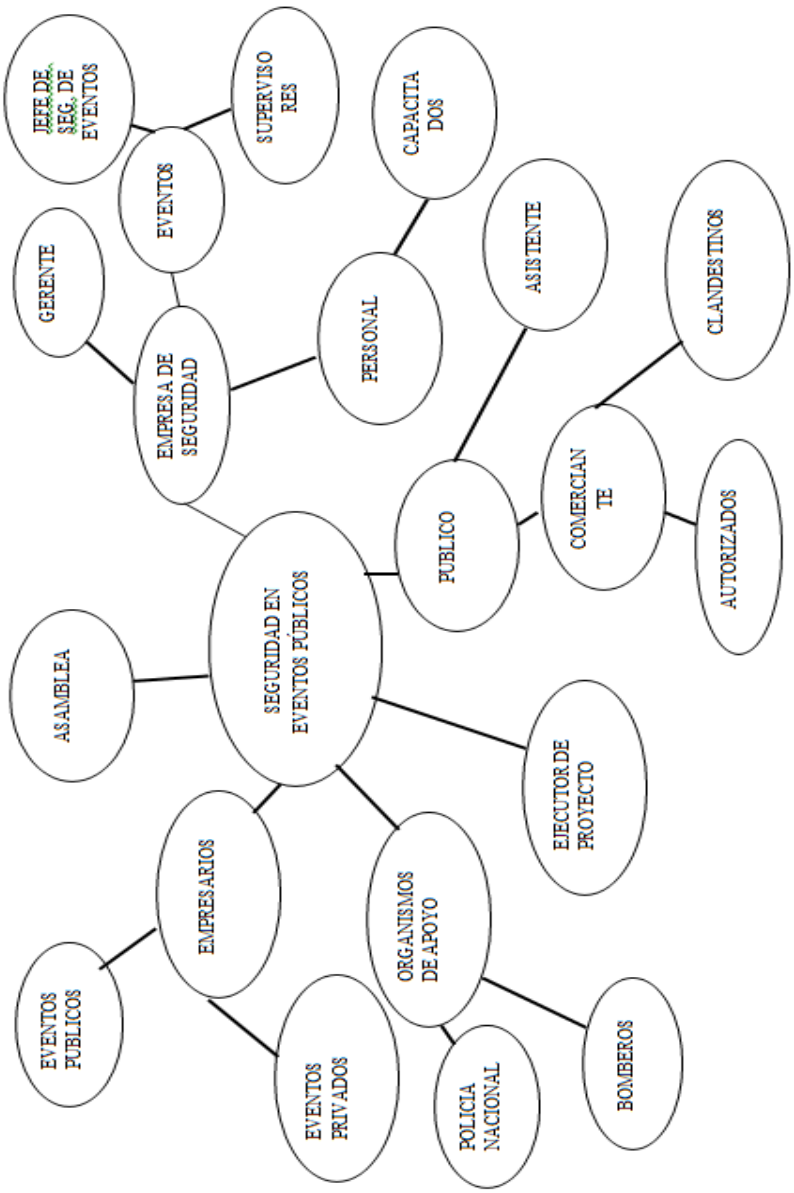
IV.- Implementar políticas de seguridad para garantizar el resguardo de los datos.

V.- Crear un manual claro y sencillo del manejo del sistema en función de las actividades operacionales del Departamento de servicios que sirva como soporte y ayuda a los usuarios.

VI.- Debido a que es el primer sistema de información con que el personal va a trabajar en la empresa, es necesario hacer un curso de capacitación para el logro del manejo eficiente del mismo.

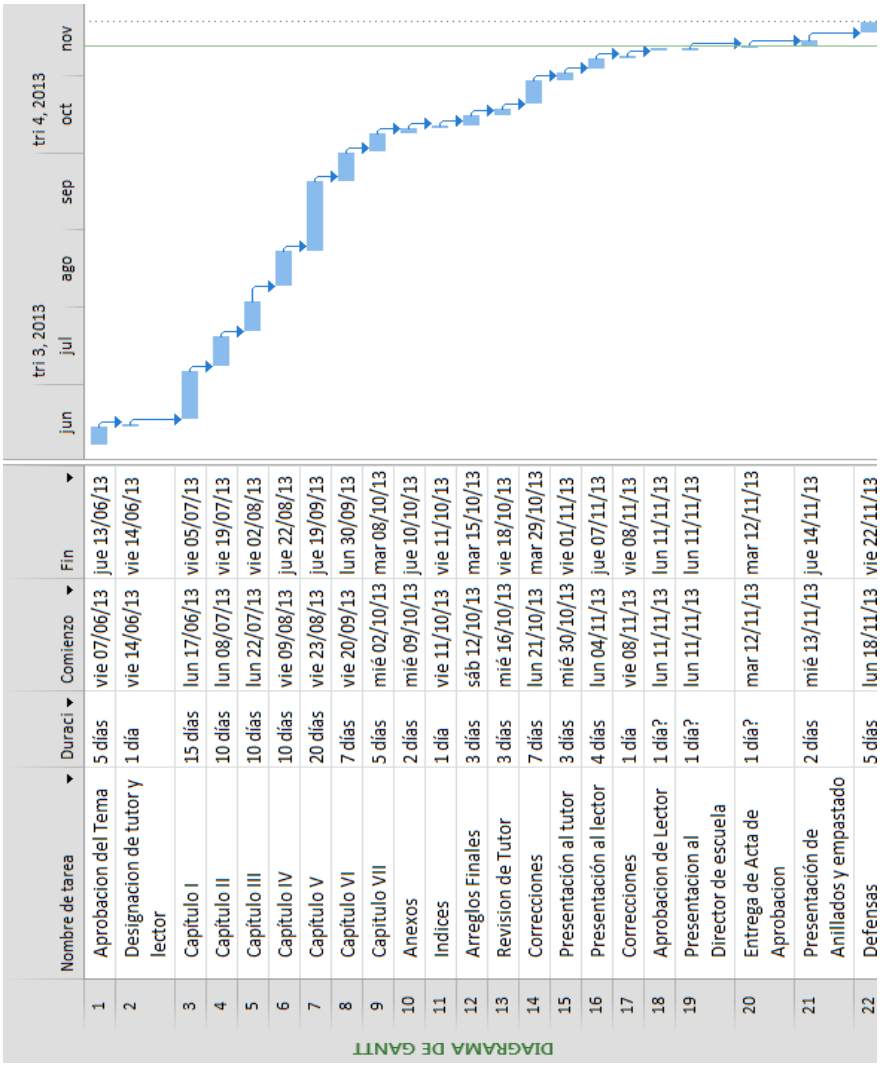
TABLAS Y FIGURAS

A.01 mapa de involucrados



Análisis de involucrados. Se describe todos lo personajes que actúan en el problema central

A.02 Cronograma



	Nombre de tarea	Duraci	Comienzo	Fin
1	Aprobacion del Tema	5 días	vie 07/06/13	jue 13/06/13
2	Designacion de tutor y lector	1 día	vie 14/06/13	vie 14/06/13
3	Capitulo I	15 días	lun 17/06/13	vie 05/07/13
4	Capitulo II	10 días	lun 08/07/13	vie 19/07/13
5	Capitulo III	10 días	lun 22/07/13	vie 02/08/13
6	Capitulo IV	10 días	vie 09/08/13	jue 22/08/13
7	Capitulo V	20 días	vie 23/08/13	jue 19/09/13
8	Capitulo VI	7 días	vie 20/09/13	lun 30/09/13
9	Capitulo VII	5 días	mié 02/10/13	mar 08/10/13
10	Anexos	2 días	mié 09/10/13	jue 10/10/13
11	Indices	1 día	vie 11/10/13	vie 11/10/13
12	Arreglos Finales	3 días	sáb 12/10/13	mar 15/10/13
13	Revision de Tutor	3 días	mié 16/10/13	vie 18/10/13
14	Correcciones	7 días	lun 21/10/13	mar 29/10/13
15	Presentación al tutor	3 días	mié 30/10/13	vie 01/11/13
16	Presentación al lector	4 días	lun 04/11/13	jue 07/11/13
17	Correcciones	1 día	vie 08/11/13	vie 08/11/13
18	Aprobacion de Lector	1 día?	lun 11/11/13	lun 11/11/13
19	Presentacion al Director de escuela	1 día?	lun 11/11/13	lun 11/11/13
20	Entrega de Acta de Aprobacion	1 día?	mar 12/11/13	mar 12/11/13
21	Presentación de Anillados y empastado	2 días	mié 13/11/13	jue 14/11/13
22	Defensas	5 días	lun 18/11/13	vie 22/11/13

Cronograma de actividades del desarrollo del proyecto.

A.03 Diccionario de datos

3.01 Introducción

En el siguiente diccionario de datos, contiene las características lógicas y puntuales de los datos, aquí se encuentra la lista de todos los elementos que forman parte del flujo de datos de todo el sistema.

Tabla 20

Registro Usuario

Tabla	Campo	Tipo dato	Longitud	Llave primaria	Descripción	Tabla foránea
Tbl_Usuario	Usu_Codigo	Int	10	PK	Código del usuario	
	Usu_Cedula	Varchar	50		Cedula del usuario	
	Usu_Nombre	Varchar	50		Nombre del usuario	
	Usu_Apellido	Varchar	50		Apellido del usuario	
	Usu_Direccion	Varchar	50		Dirección del usuario	
	Usu_Telefono	Varchar	50		Teléfono del usuario	
	Usu_Email	Varchar	50		Email del usuario	
	Usu_DirFoto	Varchar	50		Foto del usuario	
	Usu_Estado	Char	10		Estado del usuario	
	Usu_Usuario	Varchar	50		Usuario del usuario	
	Usu_Clave	Varchar	50		Clave del usuario	
	TipUsu_Codigo	Int	10	FK	Código del tipo de usuario	Tbl_Tipo_Usuario

Nota: datos de la tabla de usuario

Tabla21

Registro Empresario

Tabla	Campo	Tipo dato	Longitud	Llave primaria	Descripción	Tabla foránea
Tbl_organizador	org_id	Int	10	PK	Código del empresario	
	Org_cedula	Varchar	12		Cedula del usuario	
	Org_Nombre	Varchar	50		Nombre del usuario	
	Org_telefono	Varchar	50		telefono del usuario	
	Org_apellido	Varchar	50		apellido del usuario	
	Org_direccion	Varchar	50		direccion del usuario	
	Org_Email	Varchar	20		Email del empresario	
	Org_estado	Char	1		estado del usuario	
	Org_detalle	Varchar	100		detalle del usuario	
	Org_encargado	Varchar	50		Segundo responsible	

Nota: registro de datos del empresario

Tabla 22

Registro Empresa

Tabla	Campo	Tipo dato	Longitud	Llave primaria	Descripción	Tabla foránea
Tbl_empresa	Emp_id	Int		PK	Código de la empresa	
	Emp_descripcion	Varchar	50		descripcion de la empresa	
	Emp_estado	Varchar	1		Estado de la empresa	
	Emp_nombre	Varchar	50		nombre de la empresa	
	Emp_direccion	Varchar	50		Direccion de la empresa	
	Emp_telefono	Varchar	50		Telefono de la empresa	

Nota: registro de datos de la empresa de seguridad

Tabla 23

Registro evento

Tabla	Campo	Tipo dato	Longitud	Llave primaria	Descripción	Tabla foránea
Tbl_evento	eve_id	Int	1	PK	Código del empresario	
	eve_nombre	Varchar	50		Cedula del usuario	
	eve_descripcion	Varchar	50		Nombre del usuario	
	eve_estado	Varchar	1		telefono del usuario	
	eve_fecha	Varchar	50		apellido del usuario	
	eve_tipo	Varchar	50		direccion del usuario	
	eve_num_per_trab	Varchar	20		Email del empresario	
	eve_informe	Char	100		estado del usuario	
	Org_id	Int	1	FK	detalle del usuario	Tbl_organizador
	Emp_id	Int	1	FK	Segundo responsible	Tbl_empresa

Nota: registro de datos del evento

Tabla 24

Registro Empleados

Tabla	Campo	Tipo dato	Longitud	Llave primaria	Descripción	Tabla foránea
Tbl_empleados	Empl_id	Int	1	PK	Código de empresario	
	Empl_cedula	Varchar	50		Cedula del usuario	
	Empl_nombre	Varchar	50		Nombre del usuario	
	Empl_apellido	Varchar	1		telefono del usuario	
	Empl_direccion	Varchar	50		apellido del usuario	
	Empl_telefono	Varchar	50		direccion del usuario	
	Empl_mail	Varchar	20		Email del empresario	
	Empl_estado	Char	100		estado del usuario	
	Emp_id	Int	1	FK	Segundo responsible	Tbl_empresa

Nota:registro de datos de los empleados de seguridad

Tabla 25

Registro Escenario

Tabla	Campo	Tipo dato	Longitud	Llave primaria	Descripción	Tabla foránea
Tbl_escenario	Esc_id	Int	1	PK	Código del scenario	
	Esc_nombre	Varchar	50		Nombre del escenario	
	Esc_direccion	Varchar	50		Direccion del escenari	
	Esc_aforo	Varchar	50		Aforo del scenario	
	Esc_estado	Varchar	1		Estado del scenario	
	Esc_puer_ext	Varchar	50		Numero de puerta externas del escenario	
	Esc_puer_int	Varchar	50		Numero de puerta internas del escenario	
	Esc_localidad	Varchar	100		Numero de localidade del escenario	
	Esc_areas	Varchar	50		Número de áreas de escenario	
	Esc_fecha_inicio	Datetime			Fecha de inicio	
	Esc_fecha fin	Datetime			Fecha final	
	Esc_puert_med				Numero de puerta intermedias de escenario	
	Eve_id	Int	1	FK	Codigo del evento	Tbl_evento
	Emp_id	Int	1	FK	Codigo de la empresa	Tbl-empresa

Nota: registro de datos del escenario

Tabla 26

Registro de localidades

Tabla	Campo	Tipo dato	Longitud	Llave primaria	Descripción	Tabla foránea
Tbl_localidades	Id_localidad	Int	1	PK	Código de localidad	
	Esc_id	Int	1	FK	Codigo del scenario	Tbl_escenari
	Loc_nombre	Varchar	5		Nombre de localidad	
	Loc_estado	Char	1		Estado de localidad	

Nota:registro de datos de localidades

Tabla 27

Detalle de localidades

Tabla	Campo	Tipo dato	Longitud	Llave primaria	Descripción	Tabla foránea
Tbl_detalle_localidades	Id_detalle	Int	1	PK	Código de localidad	
	Id_localidad	Int	1	FK	Codigo del scenario	Tbl_localidades
	det_nombre	Varchar	50		Numero de personas en las localidades	
	det_estado	Char	1		Estado de detalle	

Nota: registro de la cantidad de personal para las diferentes localidades

A.04 Matriz de Marco lógico

RESUMEN NARRITIVO	INDICADORES	MEDIOS DE VERIFICACIÓN	SUPUESTOS
FIN DEL PROYECTO • Satisfacción del público asistente • Aumento de contratos para la empresa.	• El incremento de contratos aumenta en un 20% • Las pérdidas de información se reducen en un 90%	• Las encuestas realizadas a empresas de seguridad quienes realizan los informes determina, que los procesos son realizados de forma manual.	• El administrador y el personal de seguridad asistente, no tendrán la dificultad en la generación de informes y en el registro de eventos.
PROPÓSITO DEL PROYECTO • Adecuada distribución sistematizada del personal de seguridad en las diferentes áreas y localidades del escenario	• Mejora la logística de organización • Se reduce tiempo en la búsqueda de información sobre algún imprevisto que se haya suscitado en el desarrollo de un evento	• Las encuestas realizadas a los administradores sobre el uso de la herramienta tecnológica, han demostrado que tienen bajo conocimiento sobre el manejo de las aplicaciones, por lo tanto será necesario realizar un curso de capacitación sobre el uso y utilidad del software.	• Se logra optimizar el proceso de gestión humana y los registros de los diferentes empresarios así mismo como los diferentes eventos realizados o para realizarse
COMPONENTES DEL PROYECTO • Mejor relación entre jefe de seguridad y	• Los procesos de contratos desde el 2008 se han realizado manualmente hasta la actualidad, a partir desde el mes de noviembre de 2013 se optimizará	• El análisis realizado a la situación actual, ha permitido crear nuevas estrategias, el cual optimice los procesos que realiza de forma eficiente y adecuada con la finalidad de brindar un buen servicio al público asistente.	• Los administradores y el personal asistente optaron con nuevos conocimientos y están dispuestos a utilizar la tecnología mediante la cual van a realizar sus labores con el afán de brindar comodidad a todos los

organizador • dispone de la tecnología adecuada para registrar la información de los pacientes. • Los itinerarios han sido optimizados	automáticamente la gestión de contratos. • La asignación del personal de seguridad se la estado realizando en forma manual hasta en la fecha actual, a partir del mes de noviembre de 2013 se realizara sistematizadamente la asignación.		empresarios. • Gerente conforme con interacción que se realiza en la utilización del software.
ACTIVIDADES DEL PROYECTO • Implementar un sistema que mejore el tiempo de respuesta a los organizadores • Agilizar la gestión humana • Optimizar tiempo • Generar una bitácora sobre los evento realizados		• Las necesidades que requiere el desarrollador del proyecto. los requerimientos se basan en entorno y en función mediante el cual el proyecto se encuentre en la fase de desarrollo.	• El manteamiento y las configuraciones de la herramienta tecnológica a utilizarse será respaldada por el desarrollador del proyecto.

A.05 Manual técnico

En el manual técnico se detalla parte de la estructura de la programación realizada. Con la finalidad que el departamento técnico pueda comprender la lógica de programación empleada. Además de cómo fueron estructurados los datos para la creación de las tablas de la base de datos.

5.02 Objetivo del Manual.

Proporcionar un resumen de la composición técnica de cada módulo y pantalla, así como los detalles técnicos de manera clara mediante la descripción y gráficos del sistema, para que los usuarios técnicos tengan una mayor visión de la estructuración y funcionamiento del sistema.

5.03 Capa de Datos

5.03.01 Clase conectar.

```
private void general_Load(object sender, EventArgs e)
      {
         base_datos.OpenDB();
         if (var.R == true)
         {
            MessageBox.Show("Conectado a SQL", "");
         }
         else
         {
            var.Mensaje = "No se ha establecido la \n";
            var.Mensaje = var.Mensaje + "conección con la base \n";
            var.Mensaje = var.Mensaje + "de datos \n";
            MessageBox.Show(var.Mensaje, "Conexion", MessageBoxButtons.OK, MessageBoxIcon.Error);
         }
      }
```

5.03.02 Clase registro de usuario.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace eventos
{
   public partial class registro_usuario : Form
   {
      public registro_usuario()
      {
         InitializeComponent();
      }

      private void txt_desc_TextChanged(object sender, EventArgs e)
      {

      }
      private void registro_usuario_Load(object sender, EventArgs e)
      {
         Cargausuario();
         UltimoReg();

         datosemp();
      }

      private void Cargausuario()
      {
         {
            SqlCommand sp_Seleusuario = new SqlCommand("sp_Selectusuario", var.Conectar);
//especificamos nombre del Procedimiento Almacenado
            sp_Seleusuario.CommandType = CommandType.StoredProcedure;
            DataTable dt = new DataTable();
            SqlDataAdapter da = new SqlDataAdapter(sp_Seleusuario);
            da.Fill(dt);
            dgr_usu.DataSource = dt;
         }
      }

      private void dgr_usu_CellClick(object sender, DataGridViewCellEventArgs e)
      {
         SqlCommand Consulta = new SqlCommand("SELECT * FROM tbl_usuario WHERE
usu_nombre='" + dgr_usu.CurrentCell.Value + "';", var.Conectar);
         SqlDataReader dr = Consulta.ExecuteReader();
         if (dr.HasRows) //indica si obtiene una o varias filas
         {
            if (dr.Read())
            {
               Cancelar();
               this.txt_id.Text = dr[0].ToString();
              // this.txt_id.Text = dr[1].ToString();
               this.txt_nom.Text = dr[2].ToString();
               this.txt_ape.Text = dr[3].ToString();
               this.txt_tip.Text = dr[4].ToString();
```

```
                this.txt_est.Text = dr[5].ToString();
                this.txt_log.Text = dr[6].ToString();
                this.txt_pas.Text = dr[7].ToString();
                this.txt_dir.Text = dr[8].ToString();
                this.txt_tel.Text = dr[9].ToString();
                this.txt_ced.Text = dr[10].ToString();

            }
        }
        else
        {
            MessageBox.Show("No Existe Datos para Leer", "usuario", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            txt_id.Focus();
        }
        dr.Close();
    }



    private void Cancelar()
    {
        txt_id.Text = "";
       // txt_emp.Text = "";
        txt_nom.Text = "";
        txt_ape.Text = "";
        txt_tip.Text = "";
        txt_est.Text = "";
        txt_log.Text = "";
        txt_pas.Text = "";
        txt_dir.Text = "";
        txt_tel.Text = "";
        txt_ced.Text = "";
        txt_id.Focus();
    }

    private void lbl_registro_Click(object sender, EventArgs e)
    {

    }

    private void btn_guardar_Click(object sender, EventArgs e)
    {
        if (txt_nom.Text == string.Empty || txt_ape.Text == string.Empty || txt_log.Text == string.Empty ||
txt_tel.Text == string.Empty || txt_ced.Text == string.Empty || txt_pas.Text == string.Empty)
        {
            MessageBox.Show("Verifique el ingreso de datos", "", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            txt_nom.Focus();
        }
        else
        {
            GrabarRegistro(txt_id.Text,txt_codemp.Text,txt_nom.Text, txt_ape.Text, txt_tip.Text,
txt_est.Text, txt_log.Text,txt_pas.Text,txt_dir.Text,txt_tel.Text,txt_ced.Text);
            MessageBox.Show("<< Registro grabado >>", "usuario", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            Cargausuario();
            Cancelar();
        }
    }

    private bool GrabarRegistro(String Cod, string Emp, string Nom, string Ape, string Tip, string Est,
string Log, string Pas, String Dir, string Tel, string Ced)
```

```
        {
            //Instruccion SQL
            String CadenaSQL = "INSERT INTO tbl_usuario
(usu_id,emp_id,usu_nombre,usu_apellido,usu_tipo_usuario,usu_estado,usu_login,usu_password,usu_direc
cion,usu_telefono,usu_cedula) ";
            CadenaSQL = CadenaSQL + "VALUES ('" + Cod + "','" + Emp + "','" + Nom + "','" + Ape + "','" +
Tip + "','" + Est + "','" + Log + "','" + Pas + "','" + Dir + "','" + Tel + "','" + Ced + "')";

            // Crear comando
            SqlCommand guardar = var.Conectar.CreateCommand();
            guardar.CommandText = CadenaSQL;

            //Ejecutar SQL Insercion de registros
            guardar.ExecuteNonQuery();
            Cargausuario();
            return true;
        }

        private void txt_buecar_TextChanged(object sender, EventArgs e)
        {

        }

        private void btn_eliminar_Click(object sender, EventArgs e)
        {// Crear comando
            if (txt_id.Text != string.Empty)
            {
                if (MessageBox.Show("Desea Eliminar este Registro?", "Eliminar Datos",
MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
                {
                    SqlCommand Elimina = var.Conectar.CreateCommand();
                    Elimina.CommandText = "DELETE FROM tbl_usuario WHERE usu_id = '" +
Convert.ToInt32(txt_id.Text) + "'";

                    //Ejecutar SQL Eliminacion de registros
                    Elimina.ExecuteNonQuery();
                    MessageBox.Show("Registro Eliminado ...", "Eliminar Datos", MessageBoxButtons.OK,
MessageBoxIcon.Information);
                    Cargausuario();
                    Cancelar();
                    UltimoReg();
                }
            }
        }

        private void UltimoReg()
        {
            SqlCommand Consulta = new SqlCommand("SELECT  TOP 1 usu_id  FROM tbl_usuario ORDER
BY usu_id DESC;", var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    txt_id.Text = Convert.ToString(int.Parse(dr[0].ToString()) + 1);
                }
            }
            else
            {
                txt_id.Text = "1";
            }
            dr.Close();
```

```
    }

    private void btn_actualizar_Click(object sender, EventArgs e)
    {
      if (txt_id.Text != string.Empty)
      {
        // Crear comando
        if (MessageBox.Show("Desea modificar este Registro?", "Actualizar Datos",
MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
        {
          SqlCommand Actualiza = var.Conectar.CreateCommand();
          Actualiza.CommandText = "UPDATE tbl_usuario " +
                                  "SET usu_nombre='" + txt_nom.Text + "',usu_apellido='" + txt_ape.Text +
"',usu_tipo_usuario='" + txt_tip.Text + "',usu_estado='" + txt_est.Text + "',usu_login='" + txt_log.Text +
"',usu_password='" + txt_pas.Text + "',usu_direccion='" + txt_dir.Text + "',usu_telefono='" +txt_tel.Text+
"',usu_cedula='" +txt_ced.Text +"' " +
                                  "WHERE (usu_id='" + txt_id.Text + "')";

          //Ejecutar SQL Actualización de registros
          Actualiza.ExecuteNonQuery();
          MessageBox.Show("Registro Actualizado ...", "Actualizar Datos", MessageBoxButtons.OK,
MessageBoxIcon.Information);
          Cargausuario();
          Cancelar();
        }
      }
    }

    private void btn_cancelar_Click(object sender, EventArgs e)
    {
      Cancelar();
      btn_guardar.Enabled = false;
      //btnActualizar.Enabled = false;
      //btnEliminar.Enabled = false;
      //UltimoReg();
      Close();
    }


    private void datosemp()
    {
      SqlCommand spempresa = new SqlCommand("sp_Selectempresa", var.Conectar);
      spempresa.CommandType = CommandType.StoredProcedure;
      DataTable at = new DataTable();
      SqlDataAdapter da = new SqlDataAdapter(spempresa);
      da.Fill(at);
      txt_emp.DataSource = at;
      txt_emp.DisplayMember = "emp_nombre";
    }

    private void txt_emp_Validating_1(object sender, CancelEventArgs e)
    {
      if (txt_emp.Text.Length == 0)
      {
        error.SetError(txt_emp, "seleccione ");
      }
      else
      {
        error.SetError(txt_emp, string.Empty);
      }

    }
```

```
        private void txt_emp_SelectedValueChanged_1(object sender, EventArgs e)
        {
            SqlCommand emp_codigo = new SqlCommand("Select emp_id From tbl_empresa where
emp_nombre= '" + txt_emp.Text + "';", var.Conectar);
            SqlDataReader dr = emp_codigo.ExecuteReader();
            if (dr.HasRows)
            {
                if (dr.Read())
                {
                    txt_codemp.Text = dr[0].ToString();

                }
            }
            dr.Close();

        }




        private void buscar2()
        {

            SqlCommand buscar_usu2 = new SqlCommand("buscar_usu2", var.Conectar);    //especificamos
nombre del Procedimiento Almacenado
            buscar_usu2.CommandType = CommandType.StoredProcedure;
            DataTable dt = new DataTable();
            SqlDataAdapter da = new SqlDataAdapter(buscar_usu2);
            buscar_usu2.Parameters.AddWithValue("@nom", txt_buscar.Text);
            da.Fill(dt);
            dgr_usu.DataSource = dt;

        }




        private void btn_bus_Click(object sender, EventArgs e)
        {
            SqlCommand buscar_usu2 = new SqlCommand("buscar_usu2", var.Conectar);    //especificamos
nombre del Procedimiento Almacenado
            buscar_usu2.CommandType = CommandType.StoredProcedure;
            DataTable dt = new DataTable();
            SqlDataAdapter da = new SqlDataAdapter(buscar_usu2);
            buscar_usu2.Parameters.AddWithValue("@nom", txt_buscar.Text);
            da.Fill(dt);
            dgr_usu.DataSource = dt;

        }


    }

}
```

5.03.03 Clase registro de empresario.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace eventos
{
    public partial class registro_empresario : Form
    {
        public registro_empresario()
        {
            InitializeComponent();
        }

        private void txt_desc_TextChanged(object sender, EventArgs e)
        {

        }

        private void registro_empresario_Load(object sender, EventArgs e)
        {
            Cargaempresario();
            UltimoReg();
        }

        private void Cargaempresario()
        {
            {
                SqlCommand sp_Seleempresario = new
SqlCommand("sp_Selectorganizador", var.Conectar);    //especificamos nombre
del Procedimiento Almacenado
                sp_Seleempresario.CommandType = CommandType.StoredProcedure;
                DataTable dt = new DataTable();
                SqlDataAdapter da = new SqlDataAdapter(sp_Seleempresario);
                da.Fill(dt);
                dgr_emp.DataSource = dt;
            }
        }

        private void dgr_emp_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
            SqlCommand Consulta = new SqlCommand("SELECT * FROM
tbl_organizador WHERE org_nombre='" + dgr_emp.CurrentCell.Value + "';",
var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    Cancelar();
                    this.txt_id.Text = dr[0].ToString();
                    this.txt_ced.Text = dr[1].ToString();
```

```
                    this.txt_nom.Text = dr[2].ToString();
                    this.txt_tel.Text = dr[3].ToString();
                    this.txt_ape.Text = dr[4].ToString();
                    this.txt_dir.Text = dr[5].ToString();
                    this.txt_cor.Text = dr[6].ToString();
                    this.txt_est.Text = dr[7].ToString();
                    this.txt_det.Text = dr[8].ToString();


                }
            }
            else
            {
                MessageBox.Show("No Existe Datos para Leer", "usuario",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_id.Focus();
            }
            dr.Close();
        }

        private void Cancelar()
        {
            txt_id.Text = "";
            txt_ced.Text = "";
            txt_nom.Text = "";
            txt_tel.Text = "";
            txt_ape.Text = "";
            txt_dir.Text = "";
            txt_cor.Text = "";
            txt_est.Text = "";
            txt_det.Text = "";
            txt_id.Focus();
        }

        private void lbl_registro_Click(object sender, EventArgs e)
        {

        }

        private void lbl_buscar_Click(object sender, EventArgs e)
        {

        }

        private void btn_guardar_Click(object sender, EventArgs e)
        {
            if (txt_nom.Text == string.Empty || txt_ape.Text == string.Empty
|| txt_dir.Text == string.Empty || txt_tel.Text == string.Empty ||
txt_ced.Text == string.Empty || txt_det.Text == string.Empty)
            {
                MessageBox.Show("Verifique el ingreso de datos", "",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_nom.Focus();
            }
            else
            {
                GrabarRegistro(txt_id.Text,txt_ced.Text,txt_nom.Text,
txt_tel.Text, txt_ape.Text, txt_dir.Text,
txt_cor.Text,txt_est.Text,txt_det.Text);
                MessageBox.Show("<< Registro grabado >>", "usuario",
MessageBoxButtons.OK, MessageBoxIcon.Information);
```

```
                Cargaempresario();
                Cancelar();
            }
        }

        private bool GrabarRegistro(String Cod, string Ced, string Nom, string
Tel, string Ape, string Dir, string Cor, string Est, String Det)
        {
            //Instruccion SQL
            String CadenaSQL = "INSERT INTO tbl_organizador
(org_id,org_cedula,org_nombre,org_telefono,org_apellido,org_direccion,org_mail
,org_estado,org_detalle) ";
            CadenaSQL = CadenaSQL + "VALUES ('" + Cod + "','" + Ced + "','" +
Nom+ "','" + Tel + "','" + Ape + "','" + Dir + "','" + Cor + "','" + Est +
"','" + Det + "')";

            // Crear comando
            SqlCommand guardar = var.Conectar.CreateCommand();
            guardar.CommandText = CadenaSQL;

            //Ejecutar SQL Insercion de registros
            guardar.ExecuteNonQuery();
            Cargaempresario();
            return true;
        }

        private void btn_eliminar_Click(object sender, EventArgs e)
        {
            // Crear comando
            if (txt_id.Text != string.Empty)
            {
                if (MessageBox.Show("Desea Eliminar este Registro?", "Eliminar
Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
                {
                    SqlCommand Elimina = var.Conectar.CreateCommand();
                    Elimina.CommandText = "DELETE FROM tbl_organizador WHERE
org_id = '" + Convert.ToInt32(txt_id.Text) + "'";

                    //Ejecutar SQL Eliminacion de registros
                    Elimina.ExecuteNonQuery();
                    MessageBox.Show("Registro Eliminado ...", "Eliminar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargaempresario();
                    Cancelar();
                    UltimoReg();
                }
            }
        }

        private void UltimoReg()
        {
            SqlCommand Consulta = new SqlCommand("SELECT  TOP 1 org_id  FROM
tbl_organizador ORDER BY org_id DESC;", var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    txt_id.Text = Convert.ToString(int.Parse(dr[0].ToString())
+ 1);
                }
```

```
            }
            else
            {
                txt_id.Text = "1";
            }
            dr.Close();

        }

        private void btn_actualizar_Click(object sender, EventArgs e)
        {
            if (txt_id.Text != string.Empty)
            {
                // Crear comando
                if (MessageBox.Show("Desea modificar este Registro?",
"Actualizar Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) ==
DialogResult.Yes)
                {
                    SqlCommand Actualiza = var.Conectar.CreateCommand();
                    Actualiza.CommandText = "UPDATE tbl_organizador " +
                                            "SET org_cedula='" + txt_ced.Text
+ "',org_nombre='" + txt_nom.Text + "',org_telefono='" + txt_tel.Text +
"',org_apellido='" + txt_ape.Text + "',org_direccion='" + txt_dir.Text +
"',org_mail='" + txt_cor.Text + "',org_estado='" + txt_est.Text +
"',org_detalle='" + txt_det.Text + "' " +
                                            "WHERE (org_id='" + txt_id.Text +
"')";

                    //Ejecutar SQL Actualización de registros
                    Actualiza.ExecuteNonQuery();
                    MessageBox.Show("Registro Actualizado ...", "Actualizar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargaempresario();
                    Cancelar();
                }
            }
        }

        private void btn_cancelar_Click(object sender, EventArgs e)
        {
            Cancelar();
            btn_guardar.Enabled = false;
            //btnActualizar.Enabled = false;
            //btnEliminar.Enabled = false;
            //UltimoReg();
            Close();

        }

    }
}
```

5.03.04 Clase registro de evento.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
```

```
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace eventos
{
    public partial class registro_evento : Form
    {
        public registro_evento()
        {
            InitializeComponent();
        }
        private void txt_desc_TextChanged(object sender, EventArgs e)
        {

        }


        private void registro_evento_Load(object sender, EventArgs e)
        {
            Cargaevento();
            datosorg();
            UltimoReg();
            datosemp();
        }

        private void Cargaevento()
        {
            {
                SqlCommand sp_Seleevento = new SqlCommand("sp_Selectevento",
var.Conectar);    //especificamos nombre del Procedimiento Almacenado
                sp_Seleevento.CommandType = CommandType.StoredProcedure;
                DataTable dt = new DataTable();
                SqlDataAdapter da = new SqlDataAdapter(sp_Seleevento);
                da.Fill(dt);
                dgr_evento.DataSource = dt;
            }
        }


        private void dgr_evento_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
            SqlCommand Consulta = new SqlCommand("SELECT * FROM tbl_evento
WHERE eve_nombre='" + dgr_evento.CurrentCell.Value + "';", var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    Cancelar();
                    this.txt_id.Text = dr[0].ToString();
                    this.txt_codorg.Text = dr[1].ToString();
                    this.txt_nom.Text = dr[2].ToString();
                    this.txt_des.Text = dr[3].ToString();
                    this.txt_est.Text = dr[4].ToString();
                    this.txt_fec.Text = dr[5].ToString();
                    this.txt_tip.Text = dr[6].ToString();
                    this.txt_num.Text = dr[7].ToString();
                    this.txt_emp.Text = dr[8].ToString();
                    this.txt_inf.Text = dr[9].ToString();
```

```
                    this.txt_enc.Text = dr[10].ToString();
                    //this.txt_.Text = dr[10].ToString();

                }
            }
            else
            {
                MessageBox.Show("No Existe Datos para Leer", "evento",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_id.Focus();
            }
            dr.Close();
        }



        private void Cancelar()
        {
            txt_id.Text = "";
            txt_codorg.Text = "";
            txt_nom.Text = "";
            txt_des.Text = "";
            txt_est.Text = "";
            txt_fec.Text = "";
            txt_tip.Text = "";
            txt_num.Text = "";
            txt_emp.Text = "";
            txt_inf.Text = "";
            txt_enc.Text = "";
            txt_id.Focus();
        }

        private void txt_fec_Enter(object sender, EventArgs e)
        {

        }

        private void btn_guardar_Click(object sender, EventArgs e)
        {
            if (txt_nom.Text == string.Empty || txt_des.Text == string.Empty
|| txt_tip.Text == string.Empty || txt_enc.Text == string.Empty)
            {
                MessageBox.Show("Verifique el ingreso de datos", "",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_nom.Focus();
            }
            else
            {

GrabarRegistro(txt_id.Text,txt_codorg.Text,txt_nom.Text,txt_des.Text,
txt_est.Text, txt_fec.Text,  txt_tip.Text, txt_num.Text,
txt_emp.Text,txt_inf.Text,txt_enc.Text);
                MessageBox.Show("<< Registro grabado >>", "evento",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                Cargaevento();
                Cancelar();
            }
        }
```

```
        private bool GrabarRegistro(String Cod, string Org, string Nom, string
Des, string Est, string Fec, string Tip, string Num, String Emp,string
Inf,string Enc)
        {
            //Instruccion SQL
            String CadenaSQL = "INSERT INTO tbl_evento
(eve_id,org_id,eve_nombre,eve_descripcion,eve_estado,eve_fecha,
eve_tipo,eve_num_personal_trab,emp_id,eve_informe,eve_org_enc) ";
            CadenaSQL = CadenaSQL + "VALUES ('" + Cod + "','" + Org + "','" +
Nom + "','" + Des + "','" + Est + "', '" + Fec + "','" + Tip + "','" + Num +
"','" + Emp + "','" + Inf + "','" +Enc + "' )";

            // Crear comando
            SqlCommand guardar = var.Conectar.CreateCommand();
            guardar.CommandText = CadenaSQL;

            //Ejecutar SQL Insercion de registros
            guardar.ExecuteNonQuery();
            Cargaevento();
            return true;
        }

        private void btn_eliminar_Click(object sender, EventArgs e)
        {
            // Crear comando
            if (txt_id.Text != string.Empty)
            {
                if (MessageBox.Show("Desea Eliminar este Registro?", "Eliminar
Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
                {
                    SqlCommand Elimina = var.Conectar.CreateCommand();
                    Elimina.CommandText = "DELETE FROM tbl_evento WHERE eve_id
= '" + Convert.ToInt32(txt_id.Text) + "'";

                    //Ejecutar SQL Eliminacion de registros
                    Elimina.ExecuteNonQuery();
                    MessageBox.Show("Registro Eliminado ...", "Eliminar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargaevento();
                    Cancelar();
                    UltimoReg();
                }
            }
        }

        private void UltimoReg()
        {
            SqlCommand Consulta = new SqlCommand("SELECT  TOP 1 eve_id  FROM
tbl_evento ORDER BY eve_id DESC;", var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    txt_id.Text = Convert.ToString(int.Parse(dr[0].ToString())
+ 1);
                }
            }
            else
            {
                txt_id.Text = "1";
```

```
            }
            dr.Close();
        }

        private void btn_actualizar_Click(object sender, EventArgs e)
        {
            if (txt_id.Text != string.Empty)
            {
                // Crear comando
                if (MessageBox.Show("Desea modificar este Registro?",
"Actualizar Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) ==
DialogResult.Yes)
                {
                    SqlCommand Actualiza = var.Conectar.CreateCommand();
                    Actualiza.CommandText = "UPDATE tbl_evento " +
                                            "SET eve_nombre='" + txt_nom.Text
+ "',eve_descripcion='" + txt_des.Text + "',eve_estado='" + txt_est.Text +
"',eve_fecha='" + txt_fec.Text + "',eve_tipo='" + txt_tip.Text +
"',eve_num_personal_trab='" + txt_num.Text + "',eve_informe='" + txt_inf.Text
+ "',eve_org_enc='" + txt_enc.Text + "' " +
                                            "WHERE (eve_id='" + txt_id.Text +
"')";

                    //Ejecutar SQL Actualización de registros
                    Actualiza.ExecuteNonQuery();
                    MessageBox.Show("Registro Actualizado ...", "Actualizar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargaevento();
                    Cancelar();
                }
            }
        }

        private void btn_cancelar_Click(object sender, EventArgs e)
        {
            Cancelar();
            btn_guardar.Enabled = false;
            //btnActualizar.Enabled = false;
            //btnEliminar.Enabled = false;
            //UltimoReg();
            Close();
        }

        private void datosorg()
        {
            SqlCommand spempresa = new SqlCommand("sp_Selectempresa",
var.Conectar);
            spempresa.CommandType = CommandType.StoredProcedure;
            DataTable at = new DataTable();
            SqlDataAdapter da = new SqlDataAdapter(spempresa);
            da.Fill(at);
            txt_emp.DataSource = at;
            txt_emp.DisplayMember = "emp_nombre";
        }

        private void txt_emp_Validating(object sender, CancelEventArgs e)
        {
            if (txt_emp.Text.Length == 0)
            {
                error.SetError(txt_emp, "seleccione ");
            }
```

```
            else
            {
                error.SetError(txt_emp, string.Empty);
            }

        }

        private void txt_emp_SelectedValueChanged(object sender, EventArgs e)
        {
            SqlCommand emp_codigo = new SqlCommand("Select emp_id From
tbl_empresa where emp_nombre= '" + txt_emp.Text + "';", var.Conectar);
            SqlDataReader dr = emp_codigo.ExecuteReader();
            if (dr.HasRows)
            {
                if (dr.Read())
                {
                    txt_codemp.Text = dr[0].ToString();

                }
            }
            dr.Close();


        }

        private void pictureBox1_Click(object sender, EventArgs e)
        {

        }

        private void datosemp()
        {
            SqlCommand spempresa = new SqlCommand("sp_Selectorganizador",
var.Conectar);
            spempresa.CommandType = CommandType.StoredProcedure;
            DataTable at = new DataTable();
            SqlDataAdapter da = new SqlDataAdapter(spempresa);
            da.Fill(at);
            txt_org.DataSource = at;
            txt_org.DisplayMember = "org_nombre";

        }
        private void txt_org_SelectedValueChanged(object sender, EventArgs e)
        {
             SqlCommand org_codigo = new SqlCommand("Select org_id From
tbl_organizador where org_nombre= '" + txt_org.Text + "';", var.Conectar);
            SqlDataReader dr = org_codigo.ExecuteReader();
            if (dr.HasRows)
            {
                if (dr.Read())
                {
                    txt_codorg.Text = dr[0].ToString();

                }
            }
            dr.Close();

        }

        private void txt_org_Validating(object sender, CancelEventArgs e)
        {
```

```
            if (txt_org.Text.Length == 0)
            {
                error.SetError(txt_org, "seleccione ");
            }
            else
            {
                error.SetError(txt_org, string.Empty);
            }
        }

        private void btn_bus_Click(object sender, EventArgs e)
        {
            SqlCommand buscar_eve = new SqlCommand("buscar_eve",
var.Conectar);    //especificamos nombre del Procedimiento Almacenado
            buscar_eve.CommandType = CommandType.StoredProcedure;
            DataTable dt = new DataTable();
            SqlDataAdapter da = new SqlDataAdapter(buscar_eve);
            buscar_eve.Parameters.AddWithValue("@nom", txt_buscar.Text);
            da.Fill(dt);
            dgr_evento.DataSource = dt;

        }



    }
}
```

5.03.05 Clase registro de empleados.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace eventos
{
    public partial class registro_empleado : Form
    {
        public registro_empleado()
        {
            InitializeComponent();
        }
        private void registro_empleado_Load(object sender, EventArgs e)
        {
            Cargaempleado();
            UltimoReg();
        }
        private void Cargaempleado()
        {
            {
                SqlCommand sp_Seleusuario = new
SqlCommand("sp_Selectempleados", var.Conectar);    //especificamos nombre del
Procedimiento Almacenado
```

```
                sp_Seleusuario.CommandType = CommandType.StoredProcedure;
                DataTable dt = new DataTable();
                SqlDataAdapter da = new SqlDataAdapter(sp_Seleusuario);
                da.Fill(dt);
                dgr_empl.DataSource = dt;
            }
        }



        private void btn_guardar_Click(object sender, EventArgs e)
        {
            if (txt_nom.Text == string.Empty || txt_ape.Text == string.Empty
|| txt_dir.Text == string.Empty || txt_tel.Text == string.Empty ||
txt_ced.Text == string.Empty || txt_tel.Text == string.Empty)
            {
                MessageBox.Show("Verifique el ingreso de datos", "",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_nom.Focus();
            }
            else
            {
                GrabarRegistro(txt_id.Text,txt_emp.Text,txt_ced.Text,
txt_nom.Text, txt_ape.Text, txt_dir.Text,
txt_tel.Text,txt_cor.Text,txt_est.Text,txt_car.Text);
                MessageBox.Show("<< Registro grabado >>", "usuario",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                Cargaempleado();
                Cancelar();
            }
        }

        private bool GrabarRegistro(String Cod, string Emp, string Ced, string
Nom, string Ape, string Dir, string Tel, string Cor, String Est, string Car)
        {
            //Instruccion SQL
            String CadenaSQL = "INSERT INTO tbl_empleados
(empl_id,emp_id,empl_cedula,empl_nombre,empl_apellido,empl_direccion,empl_tele
fono,empl_mail,empl_estado,empl_cargo) ";
            CadenaSQL = CadenaSQL + "VALUES ('" + Cod + "','" + Emp + "','" +
Ced + "','" + Nom + "','" + Ape + "','" + Dir + "','" + Tel + "','" + Cor +
"','" + Est + "','" + Car + "')";

            // Crear comando
            SqlCommand guardar = var.Conectar.CreateCommand();
            guardar.CommandText = CadenaSQL;

            //Ejecutar SQL Insercion de registros
            guardar.ExecuteNonQuery();
            Cargaempleado();
            return true;
        }

        private void dgr_empl_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
              SqlCommand Consulta = new SqlCommand("SELECT * FROM
tbl_empleados WHERE empl_nombre='" + dgr_empl.CurrentCell.Value + "';",
var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
```

```
            {
                if (dr.Read())
                {
                    Cancelar();
                    this.txt_id.Text = dr[0].ToString();
                    this.txt_emp.Text = dr[1].ToString();
                    this.txt_ced.Text = dr[2].ToString();
                    this.txt_nom.Text = dr[3].ToString();
                    this.txt_ape.Text = dr[4].ToString();
                    this.txt_dir.Text = dr[5].ToString();
                    this.txt_tel.Text = dr[6].ToString();
                    this.txt_cor.Text = dr[7].ToString();
                    this.txt_est.Text = dr[8].ToString();
                    this.txt_car.Text = dr[9].ToString();


                }
            }
            else
            {
                MessageBox.Show("No Existe Datos para Leer", "usuario",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_id.Focus();
            }
            dr.Close();
        }



        private void Cancelar()
        {
            txt_id.Text = "";
            txt_emp.Text = "";
            txt_ced.Text = "";
            txt_nom.Text = "";
            txt_ape.Text = "";
            txt_dir.Text = "";
            txt_tel.Text = "";
            txt_cor.Text = "";
            txt_est.Text = "";
            txt_car.Text = "";

            txt_id.Focus();
        }

        private void btn_eliminar_Click(object sender, EventArgs e)
        {
            // Crear comando
            if (txt_id.Text != string.Empty)
            {
                if (MessageBox.Show("Desea Eliminar este Registro?", "Eliminar
Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
                {
                    SqlCommand Elimina = var.Conectar.CreateCommand();
                    Elimina.CommandText = "DELETE FROM tbl_empleados WHERE
empl_id = '" + Convert.ToInt32(txt_id.Text) + "'";

                    //Ejecutar SQL Eliminacion de registros
                    Elimina.ExecuteNonQuery();
                    MessageBox.Show("Registro Eliminado ...", "Eliminar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

```
                    Cargaempleado();
                    Cancelar();
                    UltimoReg();
                }
            }
        }

        private void UltimoReg()
        {
            SqlCommand Consulta = new SqlCommand("SELECT  TOP 1 empl_id  FROM
tbl_empleados ORDER BY empl_id DESC;", var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    txt_id.Text = Convert.ToString(int.Parse(dr[0].ToString())
+ 1);
                }
            }
            else
            {
                txt_id.Text = "1";
            }
            dr.Close();
        }


        private void btn_actualizar_Click(object sender, EventArgs e)
        {
            if (txt_id.Text != string.Empty)
            {
                // Crear comando
                if (MessageBox.Show("Desea modificar este Registro?",
"Actualizar Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) ==
DialogResult.Yes)
                {
                    SqlCommand Actualiza = var.Conectar.CreateCommand();
                    Actualiza.CommandText = "UPDATE tbl_empleados " +
                                            "SET empl_nombre='" + txt_nom.Text
+ "',empl_cedula='" + txt_ced.Text  + "',empl_apellido='" + txt_ape.Text +
"',empl_direccion='" + txt_dir.Text + "',empl_telefono='" + txt_tel.Text +
"',empl_mail='" + txt_cor.Text + "',empl_estado='" + txt_est.Text +
"',empl_cargo='" + txt_car.Text + "' " +
                                            "WHERE (empl_id='" + txt_id.Text +
"')";

                    //Ejecutar SQL Actualización de registros
                    Actualiza.ExecuteNonQuery();
                    MessageBox.Show("Registro Actualizado ...", "Actualizar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargaempleado();
                    Cancelar();
                }
            }
        }

        private void btn_cancelar_Click(object sender, EventArgs e)
        {
            Cancelar();
            btn_guardar.Enabled = false;
```

```
            //btnActualizar.Enabled = false;
            //btnEliminar.Enabled = false;
            //UltimoReg();
            Close();
        }



        }
    }
```

5.03.06Clase registro de escenario

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace eventos
{
    public partial class registro_esc_final : Form
    {
        public registro_esc_final()
        {
            InitializeComponent();
        }

        private void tabPage1_Click(object sender, EventArgs e)
        {

        }

        private void registro_esc_final_Load(object sender, EventArgs e)
        {
            //Cargaescenario();
            //UltimoReg();
          //Cargalocalidad();
        }
        private void Cargaescenario()
        {
            {
                SqlCommand sp_Seleescenario = new
SqlCommand("sp_Selectescenario", var.Conectar);    //especificamos nombre del
Procedimiento Almacenado
                sp_Seleescenario.CommandType = CommandType.StoredProcedure;
                DataTable dt = new DataTable();
                SqlDataAdapter da = new SqlDataAdapter(sp_Seleescenario);
                da.Fill(dt);
                dgr_esc.DataSource = dt;
            }
        }

        private void dgr_esc_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
```

```
            SqlCommand Consulta = new SqlCommand("SELECT * FROM tbl_escenario
WHERE esc_nombre='" + dgr_esc.CurrentCell.Value + "';", var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    Cancelar();
                    this.txt_id.Text = dr[0].ToString();
                    this.txt_nombre.Text = dr[1].ToString();
                    this.txt_dir.Text = dr[2].ToString();
                    this.txt_afo.Text = dr[3].ToString();
                    this.txt_est.Text = dr[4].ToString();
                    this.txt_acc_ext.Text = dr[5].ToString();
                    this.txt_acc_int.Text = dr[6].ToString();
                    this.txt_fec_ini.Text = dr[7].ToString();
                    this.txt_fec_fin.Text = dr[8].ToString();
                    this.txt_acc_med.Text = dr[9].ToString();
                    this.txt_eve.Text = dr[10].ToString();
                    this.txt_emp.Text = dr[11].ToString();

                }
            }
            else
            {
                MessageBox.Show("No Existe Datos para Leer", "usuario",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_id.Focus();
            }
            dr.Close();
        }



        private void Cancelar()
        {
            txt_id.Text = "";
            txt_nombre.Text = "";
            txt_dir.Text = "";
            txt_afo.Text = "";
            txt_est.Text = "";
            txt_acc_ext.Text = "";
            txt_acc_int.Text = "";
            txt_fec_ini.Text = "";
            txt_fec_fin.Text = "";
            txt_acc_med.Text = "";
            txt_eve.Text = "";
            txt_emp.Text = "";
            txt_id.Focus();
        }

        private void btn_guardar_Click(object sender, EventArgs e)
        {
            if (txt_nombre.Text == string.Empty|| txt_afo.Text == string.Empty
|| txt_est.Text == string.Empty || txt_fec_ini.Text == string.Empty ||
txt_fec_fin.Text == string.Empty)
            {
                MessageBox.Show("Verifique el ingreso de datos", "",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_nombre.Focus();
            }
```

```
            else
            {
                GrabarRegistro(txt_id.Text,txt_nombre.Text,txt_dir.Text,
txt_afo.Text, txt_est.Text, txt_acc_ext.Text,
txt_acc_int.Text,txt_fec_ini.Text,txt_fec_fin.Text,txt_acc_med.Text,txt_eve.Te
xt,txt_emp.Text);
                MessageBox.Show("<< Registro grabado >>", "usuario",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                Cargaescenario();
                Cancelar();
            }
        }

        private bool GrabarRegistro(String Cod, string Nom, string Dir, string
Afo, string Est, string ACe, string ACi, string Fin, String Ffi, string ACm,
string Eve, string Emp)
        {
            //Instruccion SQL
            String CadenaSQL = "INSERT INTO tbl_escenario
(esc_id,esc_nombre,esc_direccion,esc_aforo,esc_estado,esc_puer_ext,esc_puer_in
t,esc_fecha_inicio,esc_fecha_fin,esc_puert_med,eve_id,emp_id) ";
            CadenaSQL = CadenaSQL + "VALUES ('" + Cod + "','" + Nom + "','" +
Dir + "','" + Afo + "','" + Est + "','" + ACe + "','" + ACi + "','" + Fin +
"','" + Ffi + "','" + ACm + "','" + Eve + "','" + Emp + "')";

            // Crear comando
            SqlCommand guardar = var.Conectar.CreateCommand();
            guardar.CommandText = CadenaSQL;

            //Ejecutar SQL Insercion de registros
            guardar.ExecuteNonQuery();
            Cargaescenario();
            return true;
        }

        private void btn_eliminar_Click(object sender, EventArgs e)
        {
            if (txt_id.Text != string.Empty)
            {
                if (MessageBox.Show("Desea Eliminar este Registro?", "Eliminar
Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
                {
                    SqlCommand Elimina = var.Conectar.CreateCommand();
                    Elimina.CommandText = "DELETE FROM tbl_escenario WHERE
esc_id = '" + Convert.ToInt32(txt_id.Text) + "'";

                    //Ejecutar SQL Eliminacion de registros
                    Elimina.ExecuteNonQuery();
                    MessageBox.Show("Registro Eliminado ...", "Eliminar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargaescenario();
                    Cancelar();
                    UltimoReg();
                }
            }
        }

        private void UltimoReg()
        {
            SqlCommand Consulta = new SqlCommand("SELECT  TOP 1 esc_id  FROM
tbl_escenario ORDER BY esc_id DESC;", var.Conectar);
```

```
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    txt_id.Text = Convert.ToString(int.Parse(dr[0].ToString())
+ 1);
                }
            }
            else
            {
                txt_id.Text = "1";
            }
            dr.Close();

        }

        private void btn_actualizar_Click(object sender, EventArgs e)
        {
            if (txt_id.Text != string.Empty)
            {
                // Crear comando
                if (MessageBox.Show("Desea modificar este Registro?",
"Actualizar Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) ==
DialogResult.Yes)
                {
                    SqlCommand Actualiza = var.Conectar.CreateCommand();
                    Actualiza.CommandText = "UPDATE tbl_escenario " +
                                            "SET esc_nombre='" +
txt_nombre.Text + "',esc_direccion='" + txt_dir.Text + "',esc_aforo='" +
txt_afo.Text + "',esc_estado='" + txt_est.Text + "',esc_puer_ext='" +
txt_acc_ext.Text + "',esc_puer_int='" + txt_acc_int.Text +
"',esc_fecha_inicio='" + txt_fec_ini.Text + "',esc_fecha_fin='" +
txt_fec_fin.Text + "',esc_puert_med='" + txt_acc_med.Text + "' " +
                                            "WHERE (esc_id='" + txt_id.Text +
"')";

                    //Ejecutar SQL Actualización de registros
                    Actualiza.ExecuteNonQuery();
                    MessageBox.Show("Registro Actualizado ...", "Actualizar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargaescenario();
                    Cancelar();
                }
            }
        }

        private void btn_cancela_Click(object sender, EventArgs e)
        {
            Cancelar();
            btn_guardar.Enabled = false;
            //btnActualizar.Enabled = false;
            //btnEliminar.Enabled = false;
            //UltimoReg();
            Close();
        }

        private void tabPage6_Click(object sender, EventArgs e)
        {
```

```
        }

//*************************************************************************
**************************************

//***********************LOCALIDADES*****************************************
************************************************

//*************************************************************************
**************************************

        private void Cargalocalidad()
        {
            {
                SqlCommand sp_Selelocalidad = new
SqlCommand("sp_Selectlocalidades", var.Conectar);    //especificamos nombre
del Procedimiento Almacenado
                sp_Selelocalidad.CommandType = CommandType.StoredProcedure;
                DataTable dt = new DataTable();
                SqlDataAdapter da = new SqlDataAdapter(sp_Selelocalidad);
                da.Fill(dt);
                dgr_loc.DataSource = dt;
            }
        }

        private void dgr_loc_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
            SqlCommand Consulta = new SqlCommand("SELECT * FROM
tbl_localidades WHERE loc_nombre='" + dgr_loc.CurrentCell.Value + "';",
var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    Cancelar_loc();
                    this.txt_id_loc.Text = dr[0].ToString();
                    this.txt_id.Text = dr[1].ToString();
                    this.txt_nom_loc.Text = dr[2].ToString();
                    this.txt_est_loc.Text = dr[3].ToString();

                }
            }
            else
            {
                MessageBox.Show("No Existe Datos para Leer", "usuario",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_id_loc.Focus();
            }
            dr.Close();
        }



        private void Cancelar_loc()
        {
            txt_id_loc.Text = "";
            txt_id.Text = "";
            txt_nom_loc.Text = "";
```

```
            txt_est_loc.Text = "";
            txt_id_loc.Focus();
        }

        private void btn_guardar_loc_Click(object sender, EventArgs e)
        {
            if (txt_nom_loc.Text == string.Empty|| txt_est_loc.Text ==
string.Empty )
            {
                MessageBox.Show("Verifique el ingreso de datos", "",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_nom_loc.Focus();
            }
            else
            {

GrabarRegistro(txt_id_loc.Text,txt_id.Text,txt_nom_loc.Text,txt_est_loc.Text);
                MessageBox.Show("<< Registro grabado >>", "localidad",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                Cargalocalidad();
                Cancelar_loc();
            }
        }

        private bool GrabarRegistro(String id_loc, string Cod, string nom_loc,
string est_loc)
        {
            //Instruccion SQL
            String CadenaSQL = "INSERT INTO tbl_localidades
(id_localidad,esc_id,loc_nombre,loc_estado) ";
            CadenaSQL = CadenaSQL + "VALUES ('" + id_loc + "','" + Cod+ "','"
+ nom_loc + "','" + est_loc + "')";

            // Crear comando
            SqlCommand guardar_loc = var.Conectar.CreateCommand();
            guardar_loc.CommandText = CadenaSQL;

            //Ejecutar SQL Insercion de registros
            guardar_loc.ExecuteNonQuery();
            Cargalocalidad();
            return true;
        }

        private void btn_eliminar_loc_Click(object sender, EventArgs e)
        {

            if (txt_id_loc.Text != string.Empty)
            {
                if (MessageBox.Show("Desea Eliminar este Registro?", "Eliminar
Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
                {
                    SqlCommand Elimina = var.Conectar.CreateCommand();
                    Elimina.CommandText = "DELETE FROM tbl_localides WHERE
id_localidad = '" + Convert.ToInt32(txt_id.Text) + "'";

                    //Ejecutar SQL Eliminacion de registros
                    Elimina.ExecuteNonQuery();
                    MessageBox.Show("Registro Eliminado ...", "Eliminar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargalocalidad();
                    Cancelar_loc();
```

```
                    UltimoReg_loc();
                }
            }
        }

        private void UltimoReg_loc()
        {
            SqlCommand Consulta = new SqlCommand("SELECT  TOP 1 id_localidad
FROM tbl_localidades ORDER BY id_localidad DESC;", var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    txt_id_loc.Text =
Convert.ToString(int.Parse(dr[0].ToString()) + 1);
                }
            }
            else
            {
                txt_id_loc.Text = "1";
            }
            dr.Close();

        }

        private void btn_actualiza_loc_Click(object sender, EventArgs e)
        {

            if (txt_id_loc.Text != string.Empty)
            {
                // Crear comando
                if (MessageBox.Show("Desea modificar este Registro?",
"Actualizar Datos", MessageBoxButtons.YesNo, MessageBoxIcon.Question) ==
DialogResult.Yes)
                {
                    SqlCommand Actualiza_Loc = var.Conectar.CreateCommand();
                    Actualiza_Loc.CommandText = "UPDATE tbl_localidades " +
                                                "SET loc_nombre='" +
txt_nom_loc.Text + "',loc_estado='" + txt_est_loc.Text + "' " +
                                                "WHERE (esc_id='" +
txt_id_loc.Text + "')";

                    //Ejecutar SQL Actualización de registros
                    Actualiza_Loc.ExecuteNonQuery();
                    MessageBox.Show("Registro Actualizado ...", "Actualizar
Datos", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    Cargalocalidad();
                    Cancelar_loc();
                }
            }
        }



        private void btn_cancela_loc_Click(object sender, EventArgs e)
        {
            Cancelar_loc();
            btn_guardar_loc.Enabled = false;
            //btnActualizar.Enabled = false;
            //btnEliminar.Enabled = false;
```

```
            //UltimoReg();
            Close();
        }


        private void tab_sup_Selecting(object sender,
TabControlCancelEventArgs e)
        {
            int a;
            a = tab_sup.SelectedIndex;
            if (a == 0)
            {
                Cargaescenario();
                UltimoReg();
            }
            if (a == 1)

            {
                Cargalocalidad();
                UltimoReg_loc();
            }
            if (a == 2)
            {

            }



        }


//*************************************************************************
**************************************
        //**************************NUMERO DE PERSONAS POR
LOCALIAD*************************************************************************
*****************

//*************************************************************************
**************************************


        private void Cargadetalle()
        {
            {
                SqlCommand sp_Seledetalle = new
SqlCommand("sp_Selectdetalle_loc", var.Conectar);    //especificamos nombre
del Procedimiento Almacenado
                sp_Seledetalle.CommandType = CommandType.StoredProcedure;
                DataTable dt = new DataTable();
                SqlDataAdapter da = new SqlDataAdapter(sp_Seledetalle);
                da.Fill(dt);
                dgr_det.DataSource = dt;
            }


        }

        private void drg_det_CellClick(object sender,
DataGridViewCellEventArgs e)
        {
```

```
            SqlCommand Consulta = new SqlCommand("SELECT * FROM
tbl_detalle_localidades WHERE det_nombre='" + dgr_loc.CurrentCell.Value +
"';", var.Conectar);
            SqlDataReader dr = Consulta.ExecuteReader();
            if (dr.HasRows) //indica si obtiene una o varias filas
            {
                if (dr.Read())
                {
                    Cancelar_det();
                    this.txt_id_loc.Text = dr[0].ToString();
                    this.txt_id.Text = dr[1].ToString();
                    this.txt_nom_loc.Text = dr[2].ToString();
                    this.txt_est_loc.Text = dr[3].ToString();

                }
            }
            else
            {
                MessageBox.Show("No Existe Datos para Leer", "usuario",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                txt_id_loc.Focus();
            }
            dr.Close();
        }



        private void Cancelar_det()
        {
            txt_id_loc.Text = "";
            txt_id.Text = "";
            txt_nom_loc.Text = "";
            txt_est_loc.Text = "";
            txt_id_loc.Focus();
        }
        }

    }
```

5.04 lógica de negocio

5.04.01 Ingreso de datos.

```
USE [bd_seg_final]
GO
```

```
/****** Object:  StoredProcedure [dbo].[sp_Insert_organizador]
Script Date: 10/31/2013 11:18:02 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[sp_Insert_organizador]
@cod int,
@ced varchar(12),
@nom varchar(50),
@tel varchar(50),
@ape varchar(50),
@dir varchar(50),
@cor varchar(50),
@est varchar(50),
@det varchar(100),
@Existe int output,
@Graba int output
AS

if(@cod is null)
      begin
            print 'debe ingresar codigo de empresario'
            set @Existe=1
            return 1
      end
else
      set @Existe=0

if exists(select 1 from tbl_organizador as e where  e.org_id=@cod)
begin
      print 'codigo de empresario ya registrado'
      set @Graba=1
      return 2
end
else
      set @Graba=0
BEGIN
      insert into
tbl_organizador(org_id,org_cedula,org_nombre,org_telefono,org_apelli
do,org_direccion,org_mail,org_estado,org_detalle)
      values(@cod,@ced,@nom,@tel,@ape,@dir,@cor,@est,@det)
      set @Graba=1
      return 0
END
```

5.04.02 Actualizar datos.

```
USE [bd_seg_final]
GO
/****** Object:  StoredProcedure [dbo].[sp_Updateorganizador]
Script Date: 10/31/2013 11:30:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[sp_Updateorganizador]
@cod int,
@ced varchar(12),
```

```
@nom varchar(50),
@tel varchar(50),
@ape varchar(50),
@dir varchar(50),
@cor varchar(20),
@est char(1),
@det varchar(100),
@Existe int output,
@Update int output
AS
if(@cod is null)
      begin
            print 'ingrese codigo de empresario'
            set @Existe=1
            return 1
      end
else
      set @Existe=0
if not exists(select 1 from tbl_organizador as e where
e.org_id=@cod)
begin
      print 'codigo de empresario no registrado'
      set @Existe=0
      return 2
end
else
      set @Existe=1
BEGIN
      UPDATE tbl_organizador SET
      org_id=@cod,
      org_cedula=@ced,
      org_nombre=@nom,
      org_telefono=@tel,
      org_apellido=@ape,
      org_direccion=@dir,
      org_mail=@cor,
      org_estado=@est,
      org_detalle=@det
      WHERE org_id=@cod
      set @Update=1
      return 0
END
```

5.04.03 Borrar datos.

```
USE [bd_seg_final]
GO
/****** Object:  StoredProcedure [dbo].[sp_Deleteorganizador]
Script Date: 10/31/2013 11:31:19 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[sp_Deleteorganizador]
@nom varchar (12),
@Elimina int output
AS
```

```
if not exists(select 1 from tbl_organizador as e where
e.org_nombre=@nom)
begin
      print 'usuario no registrado'
      set @Elimina=0
      return 1
end
BEGIN
      DELETE FROM tbl_organizador
      WHERE (org_nombre=@nom)
      set @Elimina=1
      return 0
END
```

5.04.04 Seleccionar datos.

```
USE [bd_seg_final]
GO
/****** Object:  StoredProcedure [dbo].[sp_Selectorganizador]
Script Date: 10/31/2013 11:32:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[sp_Selectorganizador]
AS
BEGIN
      SELECT * FROM tbl_organizador
      return
END
```

5.04.05 Buscar datos.

```
USE [bd_seg_final]
GO
/****** Object:  StoredProcedure [dbo].[buscar_org]    Script Date:
10/31/2013 11:33:23 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[buscar_org]
@nom varchar (50)

AS

BEGIN
      select * FROM tbl_organizador
      WHERE (org_nombre like '%' + @nom + '%') or (org_cedula like
'%'+@nom+'%')
      return
END
```

A.06 Manual usuario administrador.

En este manual detalla como es el funcionamiento de cada uno de los módulos del sistema, realizándose una explicación de la acción que debe realizar para el correcto funcionamiento del sistema.

6.01 Objetivo del manual

El objetivo principal es la correcta manipulación del sistema. Proporcionando información clara y detallada de los distintos botones y formularios que realizan alguna acción dentro del sistema, de esta manera el usuario administrador podrá manipular el sistema de forma correcta.

6.02Ingreso al Sistema

En esta pantalla el usuario debe digitar el Nombre de Usuario y Clave y presiona sobre el botón aceptar tal como se muestra en la figura siguiente, los datos que se ingresan se los proporciona al momento de registrarse, si usted no está registrado tiene que hablar con el administrador para que lo registre, ya que este proceso solo lo puede realizar el administrador.

Figura23. Login del sistema

6.03Pantalla principal

En esta pantalla puede seleccionar las diferentes pestañas que sirve para los diferentes registros

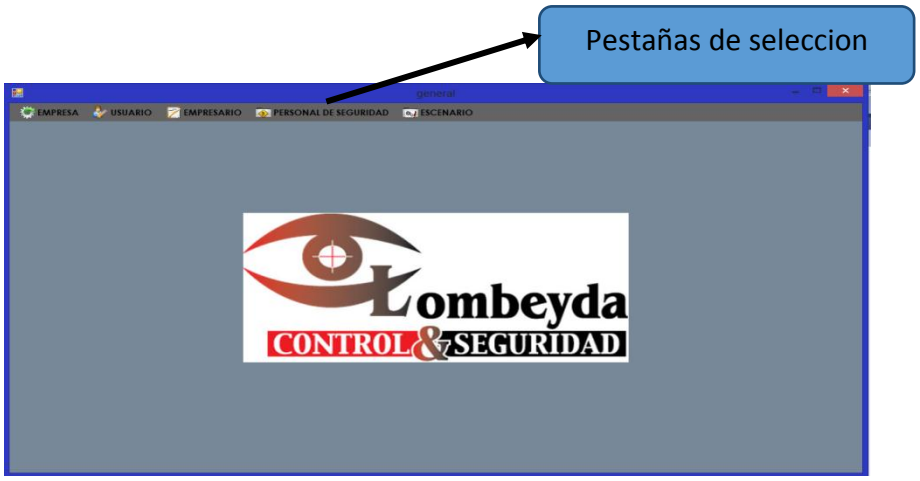


Figura 24. Pantalla principal del sistema

6.04 Registro de nuevo administrador

En esta ventana se ingresan los datos solo de las personas que van a trabajar con el sistema como lo son el administrador general o los supervisores, no existen mas tipos de usuarios.

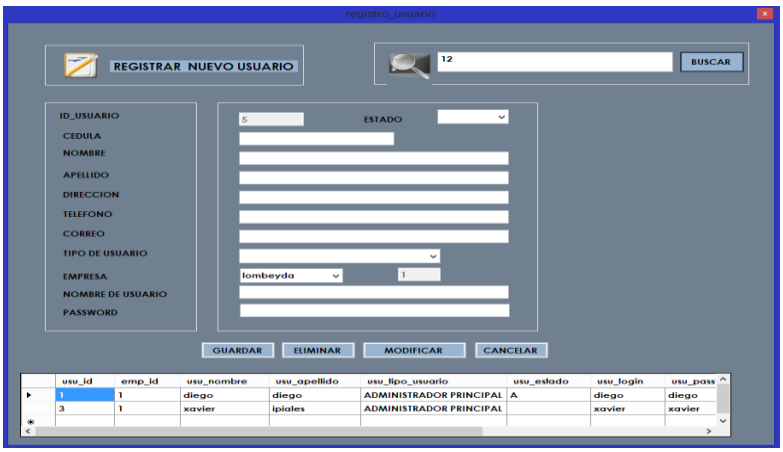


Figura 25. Registro de usuario

6.05 Registro de nuevo empresario.

En esta ventana se ingresan los datos, modifican, borran los datos de los empresarios que contratan los servicios de la empresa, para poder modificar o borrar los datos el usuario tiene que dar clic en el nombre del usuario ya registrado, caso contrario no podrá abrir el dato guardado.

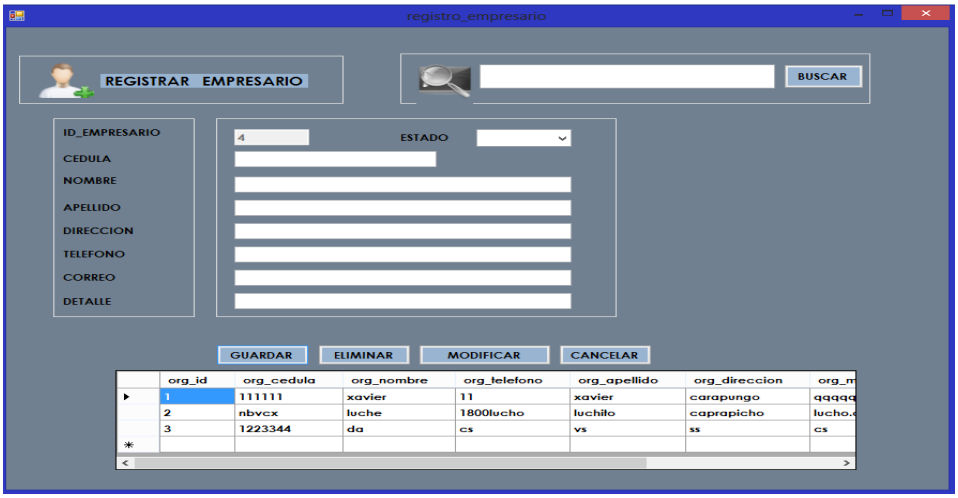


Figura 26. Registro de la empresa

6.06 Registro de evento

En esta ventana se realiza el registro de datos del evento a realizarse, para esto debe tener registrado el empresario que realiza el evento caso contrario no podrá registrarlo.

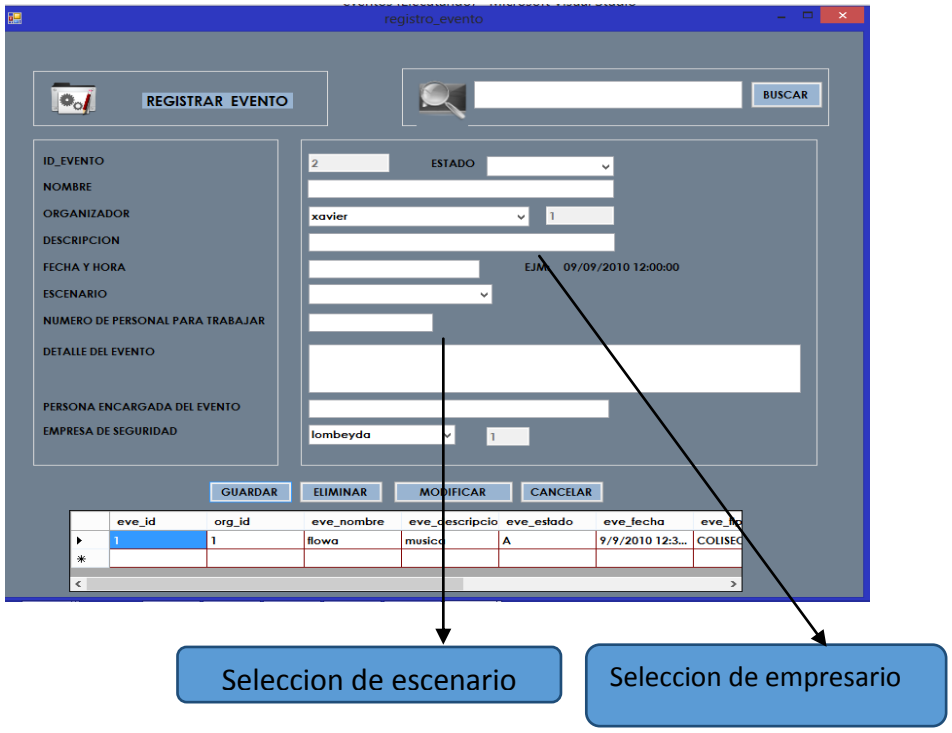


Figura 27. Registro del evento

6.07 Registro de empleados para seguridad privada para el evento

En esta ventana el administrador puede ingresar manualmente los datos de los empleados para los eventos.

Puede también el administrador importar un archivo para guardar en la base de datos ya que se trabaja con muchos empleados.

El administrador puede asignar diferentes cargos a los empleados, como puede ser el de supervisor de las diferentes áreas o también como persona encargada de las llaves de las diferentes localidades.

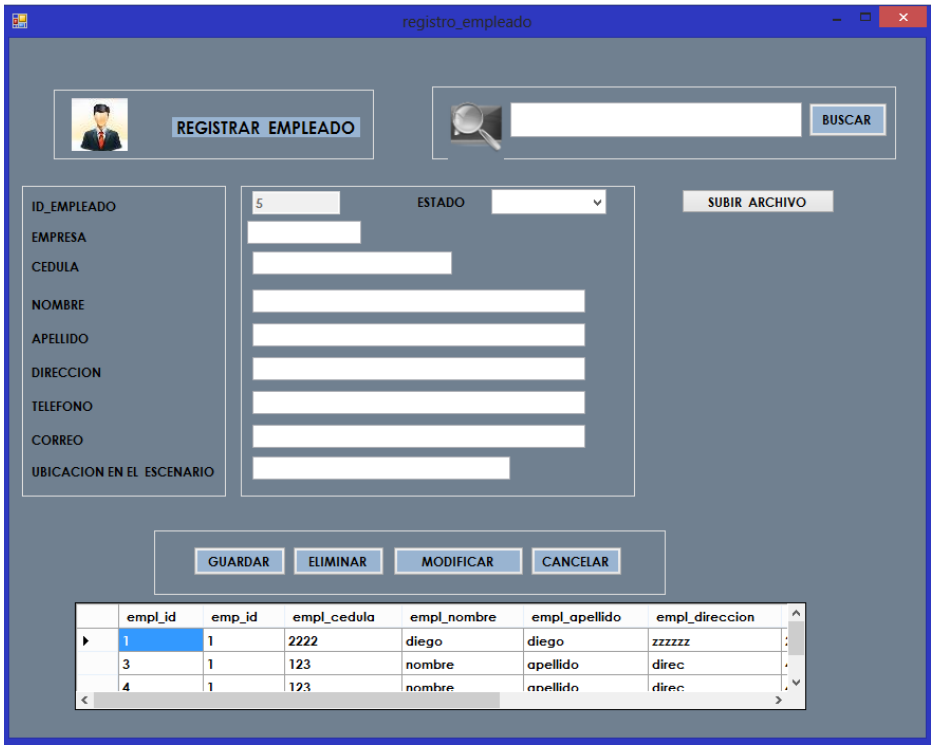


Figura 28. Registro de empleados

6.08 Registro y detalles de los escenarios para los eventos

En esta ventana se realiza el registro de los escenarios, así mismo tiene las pestañas para poder ingresar los detalles de lo que se refiere a localidades, como por ejemplo que pertas, acceso o localidades se van a habilitar según sea el evento que se va a realizar.

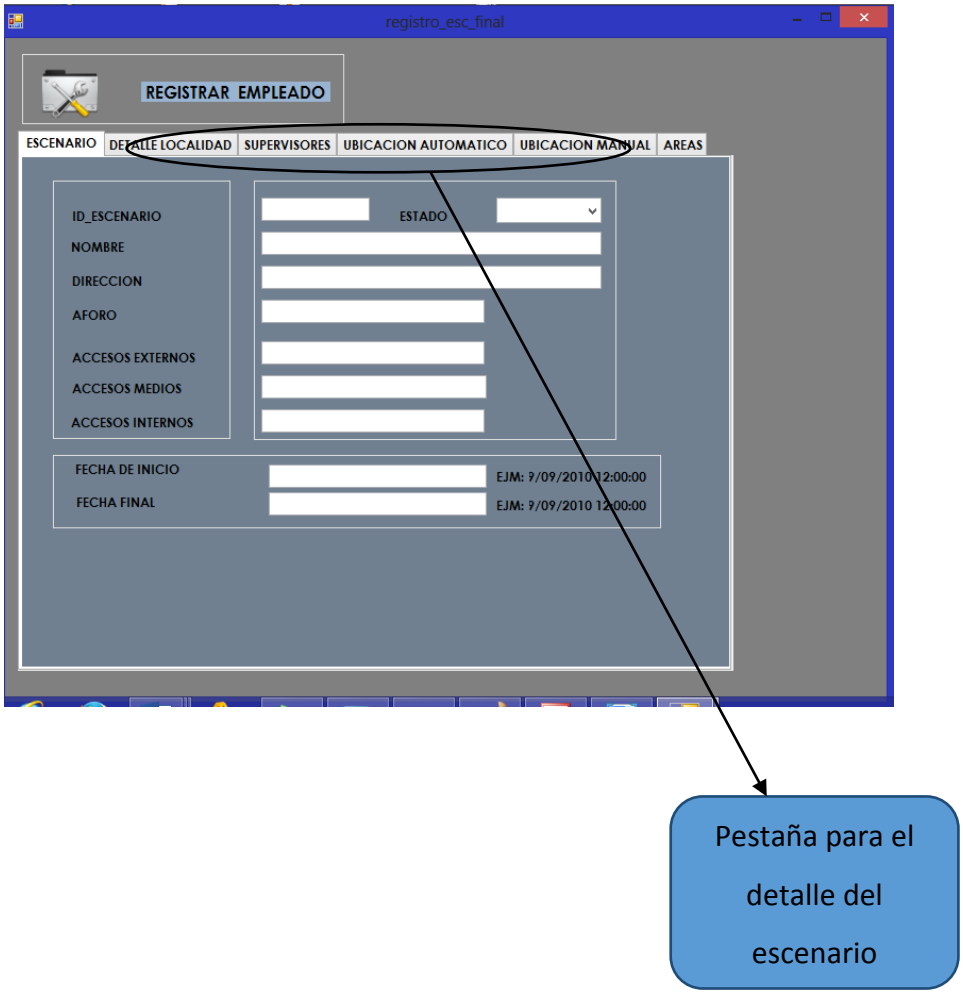


Figura 29.registro y detalle del evento y escenario

Referencias

*www7.**quito**.gob.ec/.../ORDM-284%20-%20ESPECTACULOS%20PUB...*

*www7.**quito**.gob.ec/.../ORDM-284%20-%20ESPECTACULOS%20PUB...*

*https://sites.google.com/a/udo.edu.ve/adsi/**tesis-pdf***

*ri.biblioteca.udo.edu.ve/.../**Tesis**.DISEÑO%20DE%20UN%20**SISTEMA**...*

*www.marsoft.cl/**manuales**/**Manual**%20WinEvaGropup%20rev.2.**pdf***

www.contratos.gov.co/.../DA_PROCESO_07-1-24838_118001000_324...