Universidad Nacional Autónoma de México

Facultad de Estudios Superiores Aragón

Sistema de apoyo para la captura y administración de la información del Programa del Servicio Social UNAM-Peraj

Tesis

Para obtener el título de:

Ingeniero en Computación

Presentan:

Heriberto Galdamez Torija

David Tirado Vázquez

Director de Tesis

Ing. José González Bedolla

San Juan de Aragón, Estado de México,

Septiembre de 2008

AGRADECIMIENTOS

A MIS PADRES

Por haberme dado la vida, creer en mí sin condiciones y darme la oportunidad de concluir una licenciatura, por haberme guiado y apoyado en todo momento, enseñarme a trabajar y lograr mis objetivos. Por su cariño y compresión. Tuve la fortuna de tenerlos como padres, este trabajo es para ustedes. GRACIAS!!!!

A MI FAMILIA.

A mis hermanos, Ricardo y Julio, a mi prima Raquel y a mis abuelitas que se que me están apoyando desde un lugar muy especial, en fin a toda mi familia, muchas gracias.

A MIS AMIGOS

Por confiar en mí y darme ánimos para lograr lo que hoy es una realidad, por su confianza y lealtad que me otorgaron, su valiosa amistad fue un pilar para poder realizar este trabajo, a Alejandra, marko, Víctor, Ernesto, Xavier, Rox, a todos ellos muchas gracias.

A MIS AMIGOS Y COMPAÑEROS DE LA FACULTAD.

Por brindarme su amistad durante mi estancia en la facultad, por el apoyo académico y mi formación como ser humano y persona, por haber compartido momentos muy singulares y especiales, a Luciano, Ramón, Sergio, Osvaldo, que convivieron conmigo gran parte de los años maravilloso en la facultad, a Jesús Omar, Víctor David, Felipe, los cuales fueron un aliento para el estudio y hacer muchos momentos divertidos. A todo ellos muchas Gracias.

A MIS AMIGAS Y COMPAÑERAS DE LA FACULTAD.

Por estar conmigo en las buenas y en las malas, por enriquecer mi vida con su cariño y alegría, por su paciencia y tolerancia conmigo, Ana, Wendy, Laura, y Ángela. En especial A ti Nuevamente Agradeciendo EL Incomparable tiempo que pasamos juntoZ muchAs Gracias.

A MI ASESOR.

Agradezco con mucho cariño y respeto al Ingeniero José González Bedolla , por su guía y compresión durante cuatro años, por sus consejos y enseñanzas, culminando mi estancia bajo su tutela con este trabajo. Muchas Gracias.

A LA UNIVERSIDAD

Por la formación que recibí durante casi ocho años, 3 de bachillerato y 5 de la carrera, por permitirme estudiar y realizarme como persona en la máxima casa de estudios. Agradeciendo también a PRONABES, por haberme proporcionado un apoyo y poder exigirme al máximo.

AGRADECIMIENTOS

Este trabajo de tesis quiero dedicarlo en especial a mis padres, ellos quienes me han dado todo en la vida, y de quienes he aprendido los valores del amor, el respeto y la libertad en la familia, ellos quienes han sido un pilar muy importante para poder concluir esta etapa, porque se el gran esfuerzo y sacrificio que hicieron para darme este tesoro. Gracias mamá y papá porque por ustedes soy lo que soy.

Quiero también dejar una dedicatoria hacia mi abuelo y su memoria porque aun recuerdo aquel día en que me animo y me contó el tiempo que debía luchar sin mirar atrás para llegar a este momento y ese momento ha llegado y también es parte tuya abuelo.

Agradezco a todos mis hermanos porque siempre he tenido de ellos apoyo en todos los sentidos pero sobre todo amor y palabras de aliento en los momentos difíciles por los que he atravesado y ustedes saben hermanos que cada uno tiene una parte muy especial en mi corazón. Arturo, Beatriz, Laura, Alma y Maribel. Gracias.

Agradezco muy del fondo de mis ser y mi corazón a mis dos grandes tesoros por los que debo velar y vivir por que así lo quiero, mi familia, tu Elizabeth a quien con orgullo digo mi esposa y quien se ha preocupado por mí desde el primer momento de nuestro noviazgo y a quien amo mas que a nada en el mundo, y mas ahora que me regalas la dicha de ser padre y me das una nueva esperanza de vida y una hermosa ilusión nuestra hija.

Agradezco a la UNAM con todo lo que hay en ella, por que me dio la oportunidad de formarme como ingeniero, a los profesores por permitirme tomar sus conocimientos y por enseñarme el valor de la educación, en especial a nuestro asesor el Ingeniero José González Bedolla por su apoyo y dedicación a este proyecto, a mis amigos y todos aquellos que formaron parte de mi vida universitaria, a todos ustedes Gracias.

TABLA DE CONTENIDO

INTRODUCCIÓN

Las comunicaciones y el manejo del la información han tomado gran importancia en cualquier institución, el disponer de información confiable en el momento oportuno, constituye un elemento esencial para garantizar la administración de los recursos de la misma, así como , mejorar la calidad de los servicios que presta y adecuarse constantemente al entorno que lo rodea. Por lo que se requiere que se planifiquen, desarrollen y mantengan sistemas que realicen una administración adecuada de la información. En la actualidad, las aplicaciones web poseen características que las diferencian de los sistemas tradicionales. Estas características se deben al tamaño y complejidad de las aplicaciones, dentro de las cuales podemos resaltar la facilidad para actualizar y mantener las aplicaciones sin distribuir e instalar software .

La realización de la presente tesis titulada "**SISTEMA DE APOYO PARA LA CAPTURA Y ADMINISTRACION DEL PROGRAMA DEL SERVICIO SOCIAL TUTORIAL UNAM-PERAJ**", tiene como objetivo principal, como su nombre lo indica, agilizar y automatizar la captura y administración de la información del programa UNAM-PERAJ, permitiendo a los aspirantes a participar dentro del programa de servicio social realizar su solicitud mediante una pagina Web vía Internet, evitando tiempos de espera y proporcionando su información de una manera eficaz y segura, también permite a las administradores de este programa de servicio social, consultar la información de cada uno de los aspirantes, generar reportes en formato Excel , generar gafetes en formato Pdf, poder generar cartas de registro y termino de servicio social de los participantes dentro del programa, entre otras funcionalidades. Para lograr el objetivo del presente proyecto se ha estructurado su desarrollo en 4 capítulos, los cuales tratan lo siguiente:

En el Capitulo Uno describiremos el programa de Servicio social UNAM-PERAJ, sus antecedentes, propósitos y objetivos, así como los requisitos que piden para poder entrar en el programa y dar a conocer las etapas que conforman el desarrollo del programa, con el fin de proporcionar un contexto general en donde se desarrollará la aplicación y realizar un análisis previo de todo lo que necesitamos saber para realizar el sistema.

En el Capitulo Dos describiremos las bases teóricas con las cuales podremos realizar una investigación de las herramientas de desarrollo que se utilizan para construir un sistema vía Web, así como los lenguajes de programación y bases de datos, realizando un análisis de cada uno de ellos, una breve recopilación de cada una de las tecnologías investigadas, sus características, ventajas y desventajas de cada una de ellas, para poder dar una conclusión final de cuales de ellas elegiremos y el motivo por el cual fueron seleccionadas.

En el Capitulo 3 se describe cada uno de los puntos de la problemática actual del programa, identificando los requerimientos en cada una de las etapas del programa del servicio social analizando la forma de como se llevan a cabo dichos procesos actualmente, así como con entrevistas con las personas encargadas de la

administración del programa de servicio social, para plantear una solución viable, confiable y con la calidad para poder responder a cada necesidad de acuerdo con los recursos con los que cuentan la institución.

Por ultimo en el Capitulo 4 trata el diseño, desarrollo e implementación del sistema propuesto, aplicando la metodología Yourdon para el desarrollo de sistemas, utilizando herramientas graficas como lo son los diagramas de contexto y los diagramas de flujo, para la base de datos utilizaremos el diagrama entidad relación y el diccionario de datos, así como el diseño del entorno visual con el cual van interactuar los diferentes tipos de usuarios , realizando distintas pruebas de concepto para cada uno de los módulos desarrollados.

CAPITULO I. PROGRAMA DE SERVICIO SOCIAL "ADOPTA UN AMIGO" UNAM-PERAJ

El presente capitulo describe la operación del Programa de servicio social tutorial UNAM-PERAJ, desarrollado a partir de 2003 en la Universidad Nacional Autónoma de México, a cargo de la Dirección General de Orientación y Servicios Educativos, con el propósito de apoyar el desarrollo social de alumnos de educación básica de escuelas públicas, a través de la tutoría de jóvenes universitarios prestadores de servicio social

Se pretende ofrecer una guía del programa en instituciones de educación superior, con base en el modelo desarrollado, el cual se retoma la experiencia realizada en Israel a iniciativa de la Asociación de Amigos del Instituto Weizmann de Ciencias, y se desarrolla a partir de dos estrategias fundamentales: la atención a niños de escuelas públicas y la participación de jóvenes universitarios.

1.1 ANTECEDENTES

El Programa PERAJ "Adopta un amig@" tuvo su origen en Israel en 1974 a iniciativa del Instituto Weizman de Ciencias, y se conformó como un programa institucional en el que los jóvenes universitarios adquieren el compromiso de ser tutores de niños de educación básica que lo requieran.

En Israel el Programa PERAJ es apoyado y coordinado por el gobierno a través del Ministerio de Educación, en donde los jóvenes universitarios que cuentan con el apoyo de una beca, adquieren la responsabilidad de ser tutores de al menos un niño durante un ciclo escolar. Entre 1974 y 2004, el número de tutores se incrementó de 8 a 26 mil estudiantes, con lo que se logró la integración de las instituciones de educación superior, y se generó un impacto académico y social trascendental para el desarrollo de Israel.

El modelo tutorial PERAJ se lleva a cabo en alrededor de 15 países en todo el mundo: Alemania, Hungría, Irlanda del Norte, Islandia**,** Reino Unido, Suecia, Autoridad Palestina, Brasil, Filipinas, Singapur, Australia, Nueva Zelanda, Chile y México.

La experiencia más cercana a México es el Programa "Adopta un Hermano" que se organiza en Chile desde 2001. En ambos países, el modelo de tutorías ha logrado en los menores la disminución de las tasas de deserción escolar y de participación en actividades delictivas, favoreciendo un desarrollo positivo de los niñ@s en su ámbito escolar, personal, familiar y social.

La propuesta de implantar este programa en la Universidad Nacional Autónoma de México fue presentada en 2003 por el Dr. Armando Jinich Ripstein, de la Asociación de Amigos del Instituto Weizmann de Ciencias, y por la Mtra. Ana Luz Trejo Lerdo, de Excelencia Educativa, A. C., al Dr. Juan Ramón de la Fuente, Rector de la UNAM, quien designó para su implantación a la Dirección General de Orientación y Servicios Educativos (DGOSE).

Con fundamento en la larga tradición de compromiso y vinculación que tiene la UNAM con los problemas sociales de su entorno, mediante la prestación del servicio social de los estudiantes, la DGOSE instrumentó el Peraj como un programa de servicio social, por ser esta práctica el instrumento natural por medio del cual los estudiantes entran en contacto directo con las necesidades de la población y ponen a su disposición sus conocimientos, habilidades y

destrezas para la solución de problemas comunitarios, además de ampliar su etapa formativa, al fomentar en ellos una conciencia cívica, de servicio y retribución a la sociedad.

Como dependencia que tiene a su cargo la coordinación del servicio social universitario, la DGOSE cuenta con una amplia experiencia en la generación de modelos de programas disciplinarios y multidisciplinarios de servicio social, con la participación de las facultades y escuelas, que inciden en la búsqueda de alternativas de solución de problemas de la comunidad.

A partir de esta experiencia, la DGOSE diseñó y construyó un modelo de programa de servicio social retomando la filosofía original del Peraj en Israel, y lo adecuó al contexto de nuestro país, de manera que el programa se establece y consolida como una opción de servicio social para los universitarios, en particular para los becarios del Programa Nacional de Becas para la Educación Superior (PRONABES-UNAM), quienes tienen el compromiso de realizar el servicio social en programas comunitarios de impacto social.

Los prestadores de servicio social tienen la oportunidad de apoyar a la comunidad con un programa educativo de tutoría, a través de una estrategia de integración social y afectiva entre los tutores y los amig@s, niños de entre 8 y 12 años de edad, de escuelas de nivel básico, cercanas a los campus universitarios.

En este marco, el Peraj representa una propuesta innovadora con características específicas, orientado a generar nuevas oportunidades y expectativas para los niños que participan en el Programa, quienes fortalecen su autoestima y sociabilidad, mejoran sus hábitos de estudio y amplían sus conocimientos y cultura general. Asimismo, complementa la formación integral de los jóvenes universitarios, al involucrarlos de manera real y efectiva en la atención a población vulnerable y en riesgo social.

Con estas experiencias, el modelo de participación desarrollado por la UNAM se presenta como un Programa que además de poner énfasis en el compromiso social de la Universidad con la comunidad, permite a los estudiantes realizar su servicio social en un contexto universitario.

1.2 PERAJ EN MÉXICO

El Programa PERAJ fue presentado en la Universidad Nacional Autónoma de México por la Asociación de Amigos del Instituto Weizmann de Ciencias. Para su implantación en México, la UNAM diseñó y construyó un modelo retomando la filosofía del Programa PERAJ en Israel y adecuándolo al contexto de nuestro país, con base en dos estrategias fundamentales: la atención a niños de escuelas públicas y la participación de jóvenes universitarios.

La operación del Programa PERAJ al interior de la UNAM es responsabilidad de la Dirección General de Orientación y Servicios Educativos de la Secretaría de Servicios a la Comunidad, que tiene entre sus funciones la coordinación del servicio social universitario.

El modelo de participación desarrollado por la UNAM se implementa como un programa que fortalece el compromiso de nuestra Universidad con la sociedad, al permitir a los estudiantes realizar su servicio social en un contexto universitario.

Los jóvenes universitarios se integran en equipos de trabajo multidisciplinarios donde ponen en práctica sus conocimientos y desarrollan actividades acordes con su perfil profesional, compartiendo sus cualidades humanísticas, involucrándose en un proceso educativo, afectivo y social que apoya a los niñ@s participantes en el Programa, en mejorar su desempeño

educativo y lograr el establecimiento de un vínculo de amistad que sirva como soporte en diversas situaciones de carácter individual, familiar y de interacción con su ámbito social.

1.3 Objetivos del Programa de Servicio Social UNAM-Peraj

1.3.1 Propósito

El Programa UNAM PERAJ Adopta un amig@" es un programa de servicio social que vincula a jóvenes universitarios que fungen como tutores de niños de entre 8 y 12 años de edad, de escuelas públicas cercanas al campus universitario, durante un ciclo escolar. Al tiempo de establecer una relación significativa, el Programa se propone fortalecer:

- El desarrollo social, psicológico y educativo del menor mediante estrategias pedagógicas, y
- La formación profesional y humana del joven universitario y su compromiso de retribución social, a través de la organización de actividades lúdico-recreativas.

1.3.2 Objetivos

- Contribuir a elevar la calidad del proceso formativo en el ámbito de la construcción de valores, actitudes y hábitos positivos y a la promoción del desarrollo de habilidades intelectuales en los estudiantes, mediante la utilización de estrategias de atención personalizada que complementen las actividades regulares recibidas en su salón de clase con su profesor.
- Prevenir y contribuir al abatimiento de la deserción en niveles educativos posteriores.
- Contribuir al mejoramiento de las circunstancias o condiciones del aprendizaje de los alumnos a través del apoyo académico que se brinda.

1.4 Organización

La operación del programa de servicio social tutorial UNAM-Peraj. "Adopta un Amig@" descansa en la estructura administrativa de la Dirección General de Orientación y Servicios Educativos, instancia adscrita a la Secretaría de Servicios a la Comunidad, que tiene entre sus funciones la responsabilidad de coordinar el servicio social universitario.

Con el objeto de implementar el funcionamiento del Peraj, la DGOSE formuló la propuesta de programa de servicio social multidisciplinario y lo sometió a la aprobación de todas las facultades y escuelas, con el fin de que participen jóvenes de las diferentes licenciaturas que ofrece la UNAM.

En el ciclo escolar 2002-2003 se realizó una prueba piloto, con base en resultados, se definió la capacidad de atención en el campus Ciudad Universitaria en 100 amigos, acompañados por 100 tutores, prestadores de servicio social.

El grupo de tutores es supervisado (asesorado) por cuatro coordinadores, quienes en ciclos anteriores fungieron como tutores, y cada uno se encarga de apoyar la función de 25 tutores. La actividad, tanto de coordinadores como de tutores, es dirigida por la Subdirección de Servicio Social y Vinculación Laboral de la DGOSE, a través del Departamento de Programas

Multidisciplinarios de Servicio Social, el cual, conjuntamente con tutores y coordinadores, se encarga de elaborar el programa de actividades de todo el ciclo escolar, con base en dos líneas estratégicas: Apoyo al desempeño académico de los niños, y apoyo a su formación integral, a través de actividades culturales y lúdico recreativas.

Esta forma de operación permite aprovechar los recursos humanos y materiales de la dependencia, además de concentrar el trabajo de los prestadores de servicio social, de manera que no se destina presupuesto específico para el desarrollo del Peraj.

Las instalaciones físicas de la DGOSE son la sede principal del programa, donde se reúne a los tutores y amigos dos veces por semana, por espacio de tres horas, en un aula acondicionada para tal efecto; esta sala cuenta con herramientas de apoyo al aprendizaje, como son equipos de cómputo, Programa Enciclomedia, biblioteca y materiales didácticos.

Asimismo, el programa de actividades contempla visitas guiadas a los recintos culturales y deportivos de la UNAM, como son el Centro Cultural Universitario, el Museo Universum, la Biblioteca Central, el Estadio Olímpico y las instalaciones deportivas de Ciudad Universitaria, entre otros.

1.5 Etapas de desarrollo

1.5.1 Promoción y difusión.

Cada año al inicio del ciclo escolar se invita a los prestadores de servicio social participar en este programa, a través de una Convocatoria que se difunde ampliamente a través de Gaceta UNAM y de las gacetas de facultades y escuelas, además de que se realizan presentaciones a los responsables de servicio social de las entidades académicas y a los alumnos becarios que están próximos a realizar su servicio social.

1.5.2 Selección de tutores.

El universitario interesado en participar en el Programa requiere de un proceso de selección, a través del cual es posible conocer características de su personalidad y a partir de ello evaluar si estos rasgos son acordes con el perfil que se requiere. La importancia de este proceso radica en la necesidad de incluir en el Programa únicamente a aquellos tutores que favorezcan el desarrollo integral de los niños y descartar estudiantes con rasgos de personalidad que puedan interferir en su desarrollo. La selección de los tutores se lleva a cabo utilizando los instrumentos que a continuación se describen:

- Entrevista con el responsable del programa, cuyo propósito es detectar aptitudes y actitudes que determinen si el estudiante cubre el perfil requerido.
- Aplicación del Inventario Multifásico de la Personalidad MMPI–Español, versión corta. El MMPI es uno de los instrumentos más utilizados para la evaluación de la personalidad, ya que permite que el estudiante defina sus propias características y la imagen que tiene de sí mismo. A través de su aplicación es posible detectar tanto elementos negativos que puedan existir en su personalidad, como sus recursos y aspectos positivos que favorezcan su participación en el Programa Peraj.

Una vez que el candidato fue entrevistado y respondió el MMPI, se procede a la calificación del Inventario, en el cual se evalúa lo siguiente:

- Autoestima
- Depresión
- Necesidades afectivas
- Manejo de impulsos agresivos
- Nivel de ansiedad
- Habilidades sociales
- Nivel de energía

La evaluación de estos indicadores se realiza considerando tanto la media estadística como la distribución del perfil de cada persona. Cuando algunos de los rasgos mencionados se presentan fuera de la normalidad estadística, se solicita una nueva entrevista al estudiante, con el propósito de explorar más a fondo los datos que presentan los resultados del instrumento. Posteriormente con base a la información obtenida en la entrevista y de común acuerdo con los responsables del programa, se decide la aceptación o no del candidato.

1.5.3 Selección de coordinadores.

La elección de los coordinadores se basa en su desempeño como tutor en el periodo anterior del programa, evaluando la calidad de su trabajo, su compromiso con el programa y que conozca a fondo las actividades que se realizan en el.

1.5.4 Selección Instituciones de Educación Básica.

La elección de las escuelas primarias que participan en el Programa la realiza la UNAM tomando en cuenta la evaluación de la Secretaría de Educación Pública, en el sentido de que estén consideradas como escuelas de calidad, además de que estén ubicadas físicamente en el entorno inmediato del campus universitario, con el fin de facilitar el acceso de los amig@s. Una vez seleccionadas las escuelas primarias, se realizan presentaciones del Programa a los directivos y profesores para invitarlos a participar. Se sugiere invitar a un máximo de cinco escuelas.

1.5.5 Selección de Amig@s

La escuela primaria realiza la selección de los amig@s aspirantes a recibir la tutoría, de acuerdo con los criterios establecidos en el Programa.

Presentación del programa a padres de familia. La IES que operará el Programa en coordinación con las escuelas elegidas hará la presentación del modelo a los padres de familia de los amig@s aspirantes a participar en el programa de tutorías. Una vez aceptado el Programa por los padres de familia, se sugiere suscribir una carta compromiso

Una vez seleccionados los niños, se procede a la aplicación de una batería psicológica que integra las siguientes pruebas:

- Prueba de autoestima para niños
- Prueba de depresión para niños
- Test de la Figura Humana
- Test de la Familia

Los resultados de estos instrumentos permiten obtener las mismas características que arroja la evaluación de los tutores a través del MMPI:

- Autoestima
- Depresión
- Necesidades afectivas
- Manejo de impulsos agresivos
- Nivel de ansiedad
- Habilidades sociales
- Nivel de energía

De acuerdo con el análisis de los resultados de los instrumentos aplicados y considerando la afinidad que existe entre las características de personalidad de tutores y amigos, se lleva a cabo la asignación de tutores a los amig@s. Se pretende que logren identificarse y como consecuencia de esta identificación exista una buena relación a lo largo del ciclo escolar, que permita fortalecer el desarrollo tanto de los amig@s como de los tutores.

En esta etapa el programa de tutorías es presentado a los padres de los amig@s aspirantes a participar, quienes al término de la sesión suscriben una carta compromiso.

1.5.6 Inducción.

Una vez que se ha integrado el grupo de prestadores de servicio social que fungirán como tutores y coordinadores, se realiza la semana de inducción de tutores, que incluye, además de la aplicación de instrumentos, la presentación del programa, la impartición del taller para la formación de tutores; y se abordan temas sobre niñez y adolescencia, trabajo en equipo, uso de materiales didácticos, entre otros aspectos.

1.5.7 Programa de Actividades.

Para cada ciclo escolar, los responsables de la Dirección General de Orientación y Servicios Educativos, conjuntamente con los coordinadores, diseñan un programa de trabajo basado en dos líneas estratégicas: Apoyo al desempeño académico y apoyo a la formación integral, a través de actividades lúdico recreativas.

El programa contiene objetivos, metas, y calendarización de actividades, buscando abarcar los tres ejes temáticos principales:

- Academia e Investigación
- Cultura
- Deporte y recreación

Estos ejes temáticos se desarrollan a lo largo del ciclo escolar, definiendo rubros específicos por mes: Introducción a la Universidad; Cultura y tradiciones; Protección civil; Ciencias y computación; Actividades culturales; autoestima; Actividades deportivas; Hábitos de estudio; Autocuidado de la salud; Derechos de los niños; Actividades cívicas; Medio ambiente, etcétera.

La programación mensual facilita la gestión de apoyos que se requieren de las diferentes entidades universitarias y externas para realizar las actividades que se realizan fuera de la sede del programa, como son las visitas guiadas a recintos académicos, culturales y deportivos.

La Dirección General de Orientación y Servicios Educativos se encarga de los aspectos administrativos del programa, que incluyen vigilar el cumplimiento de los requisitos establecidos por cada escuela y facultad para la prestación y liberación del servicio social, el otorgamiento de becas a coordinadores y tutores y la gestión de apoyos.

1.5.8 Evaluación y seguimiento.

El sistema de evaluación del Peraj tiene como reto responder a las múltiples dimensiones que lo conforman, siempre guiado por el objetivo básico de identificar y analizar los indicadores que permiten evaluar el funcionamiento del Programa, los escenarios de crecimiento y prospectiva, y en su caso corregir procesos.

La evaluación abarca las siguientes dimensiones:

- Para los tutores: Informe y reuniones semanales e informe de servicio social.
- Para los tutores coordinadores: Elaboración de las cartas descriptivas por actividad e informe mensual de actividades.
- Para los niñ@s: Aplicación de instrumento de integración
- Para las instancias coordinadoras: Informe inicial, intermedio y final.
- Para los padres de familia: Reunión trimestral de información.
- Para las autoridades escolares: Reunión trimestral de información.

1.5.8.1 Instrumentos de evaluación.

Se utilizan diversos instrumentos orientados temáticamente a diferentes aspectos, así como a los diversos tipos de actores.

Instrumentos de evaluación psicológica. Se aplican tanto a los tutores como a los niños al inicio y al finalizar el ciclo escolar, con el propósito de conocer los cambios que se presentaron en ellos después de haber vivido, en el caso de los niños, la experiencia de tener un tutor y en el caso de los tutores, la experiencia de apoyar y acompañar a un niño a lo largo de todo un ciclo escolar.

1.6 Estructura

Instancias Participantes

- Instituciones de Educación Superior del país
- Instituciones de Educación Básica del país
- Universidad Nacional Autónoma de México.
    - Secretaría de Servicios a la Comunidad.
    - Dirección Generar de Orientación y Servicios Educativos.
    - Escuelas y Facultades, Centros, Institutos, Dependencias Universitarias.
    - Asociación Mexicana de Amigos del Instituto Weizmann de Ciencias

Coordinación General

- Dirección General de Orientación y Servicios Educativos.

Dra. María Elisa Celis Barragán.

- Asociación Mexicana de Amigos del Instituto Weizmann de Ciencias Armando Jinich Ripstein
- Excelencia Educativa A. C. Ana Luz Trejo Lerdo

Coordinación Operativa.

- Ing. Ana de Gortari Pedroza, DGOSE-UNAM
- Lic. Claudia Navarrete García, DGOSE-UNAM
- Lic. Víctor Cruz Celis, DGOSE-UNAM
- Universitarios que fungen como líderes coordinadores y apoyos logísticos

Sede

- Infraestructura y recintos universitarios, UNAM

Recursos Humanos

- Coordinadores del Programa
- Coordinadores de tutores
- Tutores
- Amig@s
- Investigadores
- Académicos
- Funcionarios

Recursos Materiales

- Recursos Financieros (Becas, Gasto operativo)
- Transporte para los niñ@s
- Instalaciones (COE-DGOSE)
- Equipo de cómputo
- Programa Enciclomedia
- Material didáctico
- Biblioteca
- Papelería.

CAPITULO II. FUNDAMENTOS TEÓRICOS.

2.1 CONCEPTO DE BASE DE DATOS.

Primero se hablaba de archivos y conjuntos de datos, ahora se habla de grandes bases; algunos autores afirman que las bases de datos son una recopilación de los grandes bancos de datos, pero para definir correctamente se muestran a continuación algunos textos:

"Una base de datos es como un archivo de datos interrelacionados, recolectados, que satisfacen las necesidades de información de una comunidad determinada de usuarios. Cada unidad de información almacenada en una base de datos está compuesta por datos elementales, cada uno de los cuales representa características particulares de la entidad que se describe. (Gil Rivera, María del Carmen, "Las bases de datos, importancia y aplicación en la educación")

"Una base de datos es una colección de datos estructurados según un modelo que refleje las relaciones y restricciones existentes en el mundo real. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de éstas y, su definición y descripción han de ser únicas estando almacenadas junto a los mismos. Por último, los tratamientos que sufran estos datos tendrán que conservar la integridad y seguridad de éstos." (Mota Herranz, Laura, "Bases de datos relacionales, teoría y diseño").

"Una base de datos es una colección de datos correspondientes a las diferentes perspectivas de un sistema de información (de una empresa o institución), existentes en algún soporte de tipo físico (normalmente de acceso directo), agrupados en una organización integrada y centralizada en la que figuran no sólo los datos en sí, sino también las relaciones existentes entre ellos, y de forma que se minimiza la redundancia y se maximiza la independencia de los datos de las aplicaciones que los requieren." (Guilera Aguilera, Llorenc, "Introducción a la informática").

*"Una base de datos es una colección de archivos interrelacionados creados con un DBMS (Sistema Manejador de una Base de datos). El contenido de una base se obtiene combinando datos de todas las diferentes fuentes en una organización, de tal manera que los datos estén disponibles para todos los usuarios, y los datos redundantes puedan eliminarse, o al menos minimizarse." (*Alice Y.,H. Tsai, "Sistema de base de datos, Administración y uso"*).*

Por lo anterior podemos decir que las bases de datos son un conjunto de información relacionada no redundante, que está organizada, sistematizada y debe encauzarse en un propósito específico en beneficio de una comunidad. Del mismo modo tiene que cumplir con los objetivos de independencia entre las aplicaciones y las estructuras de datos, así como la integridad de los mismos y la seguridad de éstos ante los múltiples usuarios que la utilicen.

2.2 FUNDAMENTOS DE BASE DE DATOS RELACIONALES.

Es importante señalar que en el desarrollo de las bases de datos de datos el modelo más utilizado por las ventajas que ofrece sobre los demás (jeràrquicas,de red,documentales, etc.) es el modelo relacional. Los sistemas y bases de datos se han convertido en la vida cotidiana

de la sociedad en algo imprescindible. Todos los días la mayoría de la gente realiza actividades que dependen de la interacción con una base de datos.

El uso de sistemas de información por parte de las organizaciones requieren almacenar grandes cantidades de información, ya sea para el uso mismo del sistema, para generar resultados o para compartir información con otros sistemas.

Algunas ventajas de las bases de datos son:

• *Globalización de la información.*

• *Eliminación de información inconsistente.*

• *Permite compartir información.*

• *Permite mantener la integridad en la información.*

• *Independencia de datos.*

• *Coherencia de resultados.*

A continuación se definen conceptos básicos sobre las bases de datos.

• **Sistema de Gestión de Base de Datos**

Un Sistema de Gestión de Base de Datos (SGBD) es un conjunto de programas (sistema de software) de propósito general que facilita el proceso de definición, construcción y manipulación de la base de datos para usos diversos. Estos procesos se definen como:

- **Definición:** Para especificar tipos de datos, estructuras de datos y restricciones de los datos.
- **Construcción:** Para guardar los datos en un dispositivo de almacenamiento controlado por el SGBD.
- **Manipulación:** Para poder consultar y actualizar información.

• **Sistema de Base de Datos**

Es el conjunto formado por una serie de programas que interactúan con el SGBD, por el propio SGBD y la base de datos. Estos sistemas deben mantener la seguridad de la información ante caídas del sistema y ante la posibilidad de accesos concurrentes.

• **Sistemas de Información**

Un sistema de información es un conjunto de elementos que interactúan entre si con el fin de apoyar las actividades de una empresa o negocio. Un sistema de información realiza cuatro actividades básicas: entrada, almacenamiento, procesamiento y salida de información.

Existen tres tipos de sistemas de información:

- **Transaccionales:** Tienen como objetivo lograr la automatización de procesos operativos dentro de una organización. Su función principal consiste en procesar transacciones.
- **Apoyo a las decisiones:** Estos sistemas apoyan el proceso de toma decisiones.
- **Estratégicos:** Se desarrollan con el fin de lograr ventajas competitivas, a través del uso de la tecnología de la información.

• Abstracción de datos

Un sistema de base de datos es un conjunto de archivos de datos interrelacionados junto con una serie de programas que permiten acceder y modificar los datos.

Un objetivo importante de los sistemas de Bases de Datos es proporcionar a los usuarios una visión abstracta de los datos, es decir, el sistema debe ocultar al usuario los detalles sobre como se acceden y se manipulan los datos. Sin embargo para que el sistema sea eficiente, tiene que disponer de estructuras de datos bien diseñadas que son las que se guardan en la base de datos.

Esto da origen a la creación de niveles de abstracción, que muestren distintas visiones de la complejidad de la representación de la información, es decir, oculten detalles de almacenamiento. Una de las arquitecturas más comunes es la ANSI/SPARC la cuál permite ver una base de datos a tres niveles de abstracción:

Nivel físico: Nivel de abstracción mas bajo, y describe como se almacenan realmente los datos.

Nivel lógico: Nivel que describe que información se almacena en la base de datos, y como está relacionada dicha información. La definición de estructuras de datos a nivel lógico o conceptual puede suponer la creación de varias estructuras complejas a nivel físico (creación de archivos indexados por varios campos).

Nivel de visión: Nivel de abstracción mas alto, y en el que solo se describen partes de la base de datos, ya que no todos los usuarios pueden acceder a la misma parte de la base de datos. Para facilitar la interacción del usuario con el sistema, se definen varios niveles de visión, de forma que cada uno represente lo que cada usuario o grupo de usuarios necesita.

• Esquemas e instancias

La información de la base de datos cambia continuamente a medida que se va modificando la información que contiene. Al conjunto de datos de la base de datos en un momento determinado se le denomina instancia de la base de datos y cambia continuamente.

• Independencia de datos

Es la capacidad de modificar un esquema de un nivel sin afectar a los esquemas de nivel superior. Hay dos niveles de independencia de datos:

Física de datos: Es la capacidad de modificar el esquema físico sin necesidad de modificar los programas de aplicación. (pej: modificar el tamaño de un campo o modificar Índices).

Lógica de datos: Es la capacidad de modificar el esquema conceptual sin necesidad de modificar los programas de aplicación.

La independencia lógica es más difícil de conseguir, ya que los programas de aplicación suelen ser dependientes de la estructura lógica de los datos que acceden. No obstante, es posible realizar cambios mediante la definición de esquemas externos, de forma que en lugar de

eliminar algunas partes del esquema conceptual, se podría definir un esquema externo a partir del esquema conceptual, sin afectar a los esquemas y aplicaciones existentes.

2.2.1. Modelo Entidad - Relacion.

El modelo de datos entidad-relación (E-R) se basa en una percepción del mundo real que consiste en coleccionar objetos básicos denominados entidades y relaciones entre estos objetos.

Una entidad es un objeto que se diferencia de otros objetos mediante una serie de atributos. Una relación es una asociación entre varias entidades. Además de las entidades y relaciones, el modelo E-R representa ciertas restricciones sobre el contenido de la base de datos. Una restricción importante es la cardinalidad de asignación, que expresa el número de entidades a las que puede asociarse una entidad a través de un conjunto de relaciones.

Cardinalidad

La cardinalidad expresa cuántos elementos del conjunto de entidades de un extremo de la relación (o tabla) están relacionadas con cuántas entidades del conjunto del otro extremo. Pueden ser “uno a uno'', “uno a varios'' o “varios a varios''.

Entre dos relaciones, las posibles cardinalidades son:

• **Uno a Uno:** Cada entidad de A puede estar asociada con no más de una entidad B y viceversa.

• **Uno a Muchos:** Cada entidad de A puede estar asociada a cualquier número de entidades de B y cada entidad de B puede estar relacionada con no más de una entidad de A.

• **Muchos a Muchos:** Una entidad en A está asociada a cualquier número de entidades de en B y viceversa.

Hay varias maneras de mostrar las cardinalidades en el diagrama. Una de ellas es poner etiquetas en las líneas que unen las relaciones con las entidades. La etiqueta consiste de un mínimo y un máximo, cada uno de los cuales contiene un cero, un uno o una letra *n* (“varios''). Si la cardinalidad es exactamente uno, se pone sólo el uno. En el caso de una relación *varios a varios*, lo usual es poner una *m* en un extremo y una *n* en el otro.

El modelo relacional representa los datos y las relaciones entre los datos mediante un conjunto de tablas, donde cada una de las tablas tiene una serie de columnas con nombres únicos.

2.2.2. Base De Datos Relacionales.

El modelo relacional representa una base de datos como una colección de relaciones, es decir, un conjunto de tablas formado por filas y columnas. Cada fila representa un conjunto de datos relacionados entre sí. Estos valores pueden referirse a un conjunto de hechos que describen a una entidad o bien un vinculo entre entidades.

En terminología del modelo relacional, una fila se denomina tupla, una columna es un atributo y una tabla es una relación. Podemos decir que un dominio es un conjunto de valores indivisibles. Puede especificarse con el tipo de datos al que pertenecen los valores, y también con un nombre significativo que ayude a interpretarlos, así como información adicional.

En términos generales una base de datos relacional se define como un conjunto de esquemas de relaciones y un conjunto de restricciones de integridad.

Llave primaria y llave foránea

La llave primaria, es un conjunto de atributos que permiten identificar unívocamente una tupla en una relación. Naturalmente, en una relación puede haber más combinaciones de atributos que permitan identificar unívocamente una tupla ("llaves candidatas"), pero entre éstas se elegirá una sola para utilizar como llave primaria. Los atributos de la llave primaria no pueden asumir el valor nulo que significa un valor no determinado, en tanto que ya no permitirían identificar una tupla concreta en una relación.

La llave foránea o externa es una combinación de atributos de una relación que son, a su vez, una llave primaria para otra relación. Una característica fundamental de los valores presentes en una llave externa es que, a no ser que no sean null, tienen que corresponder a valores existentes en la llave primaria de la relación a la que se refieren.

Integridad de entidades, integridad referencial y claves externas

• **Restricción de integridad de entidades**

Establece que ningún valor de clave primaria puede ser nulo. Esto se debe a que el valor de la clave primaria sirve para identificar las tuplas de una relación, y si la clave primaria pudiera tener valores nulos, no podríamos identificar algunas tuplas.

• **Restricción de integridad referencial**

Se especifica entre dos relaciones, y establece que en una tupla de una relación que haga referencia a otra relación, deberá referirse a una tupla existente en dicha relación.

Operaciones de actualización y tratamiento de la violación de restricciones

Las operaciones del modelo relacional pueden clasificarse en recuperaciones y actualizaciones. Estas operaciones son tres:

• **Insertar:** Insertar una o más tuplas nuevas en una relación.

• **Eliminar:** Elimina tuplas en una relación.

• **Actualizar:** Modificar los valores de algunos atributos de tuplas existentes.

2.2.3. Lenguajes De Consulta.

Un lenguaje de consulta es un lenguaje en el que el usuario puede solicitar información de las bases de datos. Estos lenguajes de consulta suelen ser de más alto nivel que los lenguajes de programación estándar, y pueden clasificarse en procedurales y no procedurales.

• **Lenguaje procedural:** el usuario da instrucciones al sistema para que realice una serie de operaciones sobre la base de datos con el fin de obtener el resultado deseado.

• **Lenguaje no procedural**: el usuario describe la información deseada sin dar un procedimiento concreto sobre como obtener la información.

A continuación se describirán dos lenguajes de consulta formales (no comerciales), como son el álgebra relacional, que es un lenguaje de consulta procedural, y el calculó relacional, un lenguaje de consulta procedural.

2.2.3.1. ÁLGEBRA RELACIONAL.

El álgebra relacional es un lenguaje de consulta procedural. Consta de un conjunto de operaciones que toman una o dos relaciones como entrada y producen una nueva relación como salida.

Las operaciones básicas del álgebra relacional son:

• **Selección (Select):** Esta operación selecciona tuplas que satisfacen un predicado determinado.

• **Proyección (Project):** Es una operación unaria que devuelve su relación argumento con ciertas columnas omitidas.

• **Unión:** La operación unión se limita a unir resultados de relaciones, y funciona de la misma forma que la unión de conjuntos, con la observación de que las relaciones que se unan han de tener los mismos atributos, y que se eliminan la tuplas repetidas.

• **Diferencia de conjuntos (Difference):** La operación nos permite encontrar las tuplas que estén en una relación pero no estén en otra.

• **Producto cartesiano (Product):** Esta operación permite combinar varias relaciones. El resultado de la operación producto cartesiano es una nueva relación que tiene tantas columnas como la suma del número de columnas de cada relación, y las tuplas de esta nueva relación se obtiene mediante la combinación de todas las tuplas de las relaciones que participan en la operación de producto cartesiano.

Otras operaciones del álgebra relacional

• **Intersección de conjuntos:** Se trata de una operación adicional del álgebra de conjuntos, puesto que la operación se puede expresar como dos diferencias de conjuntos.

• **Producto Theta:** Es una operación para simplificar las consultas que utilizan la operación de producto cartesiano, y que permiten especificar la condición de selección de registros como subÍndice de dicha operación.

• **Producto Natural:** Permite simplificar las consultas que utilizan la operación de producto cartesiano o producto theta, y que permite obviar la especificación de condición de selección de registros como subÍndice de dicha operación.

Existen algunas extensiones del producto natural y de la unión, denominadas como uniones externas (outer joins).

• **Unión Externa izquierda (Left join):** Esta operación conserva todas las tuplas de la primera relación (la de la izquierda), de forma que si no se encuentran tuplas coincidentes, se rellenan con valores nulos.

• **Unión externa:** Une dos relaciones que no son compatibles, es decir, no se le puede aplicar el operador de unión, ya que solo algunos de los atributos son compatibles con la unión.

2.2.4. Normalización.

La normalización es un método de diseño de bases de datos relacionales basado en una serie de fundamentos teóricos y con una serie de pasos establecidos: Este método parte del conjunto de todos los atributos o campos a almacenar en la base de datos y obtiene un conjunto de tablas que almacena toda la información (datos y relaciones) de la base de datos.

Existen tres formas normales básicas para el diseño de bases de datos relacionales; se basan en el estudio de las dependencias funcionales existentes entre los atributos de una relación. Los esquemas de relación que no cumplan ciertos criterios, conocidos como las pruebas de las formas normales, deberán ser descompuestos en esquemas de relación más pequeños que satisfagan dichos criterios, obteniendo al final un conjunto de tablas que minimizan la redundancia.

En el proceso de normalización se debe de cumplir que al aplicar una forma normal determinada, se produce descomposición sin pérdida y se conservan las dependencias; es decir, las tablas que resulten de la descomposición de una tabla que no cumple los criterios de una forma normal dada, conservan las dependencias funcionales existentes en la relación original, además de que si se realiza el producto natural de las nuevas tablas obtenidas se obtiene exactamente la tabla original.

• **Primera Forma Normal:** Una relación no debe contener atributos no atómicos o multivaluados ni debe contener relaciones anidadas. Si una relación no cumple con esta condición, es decir, no esta en 1FN, se crean nuevas relaciones con los atributos multivaluados o relaciones anidadas.

• **Segunda Forma Normal:** Una relación que tiene una clave primaria compuesta no debe contener atributos no primos que dependan funcionalmente de parte de la clave. Por lo tanto, todas las relaciones con claves simples están en 2FN. Si una relación no esta en 2FN se crea una nueva relación con cada clave parcial y el conjunto de atributos que dependen funcionalmente de ella. Hay que tener en cuenta que debe seguir existiendo una relación con la clave primaria original y los atributos que dependen totalmente de ella.

• **Tercera Forma Normal:** Una relación no debe tener atributos no primos que dependan funcionalmente de atributos no primos, es decir, no deberían existir dependencias funcionales transitivas por parte de los atributos no primos. Si una relación no está en 3FN se crea una nueva relación con los atributos no primos que determinan funcionalmente los otros atributos no primos.

2.2.5 Diccionario De Datos.

Un Sistema de gestión de base de datos debe proporcionar un catálogo en el que se almacenen las descripciones de los datos y que sea accesible por los usuarios. Este catálogo es lo que se denomina diccionario de datos y contiene información que describe los datos de la base de datos (metadatos)

Normalmente, un diccionario de datos almacena:

- Nombre, tipo y tamaño de los datos.
- Nombre de las relaciones entre los datos.
- Restricciones de integridad sobre los datos.
- Nombre de los usuarios autorizados a acceder a la base de datos.
- Esquemas externos, conceptuales e internos, y correspondencia entre los esquemas.
- Estadísticas de utilización, tales como la frecuencia de las transacciones y el número de accesos realizados a los objetos de la base de datos.

Algunos de los beneficios que reporta el diccionario de datos son los siguientes:

- La información sobre los datos se puede almacenar de un modo centralizado. Esto ayuda a mantener el control sobre de ellos.
- El significado de los datos se puede definir, lo que ayudará a los usuarios a entender el propósito de los mismos.
- La comunicación se simplifica, ya que se almacena el significado exacto. El diccionario de datos también puede identificar al usuario o usuarios que poseen los datos o que los acceden.
- Las redundancias y las inconsistencias se pueden identificar más fácilmente ya que los datos están centralizados.
- Se puede tener un historial de los cambios realizados sobre la base de datos.
- El impacto que puede producir un cambio se puede determinar antes de que sea implementado, ya que el diccionario de datos mantiene información sobre cada tipo de dato, todas sus relaciones y todos sus usuarios.
- Se puede hacer respetar la seguridad.
- Se puede garantizar la integridad.
- Se puede proporcionar información para auditorias.

2.2.6 Arquitectura Cliente /Servidor.

La arquitectura cliente/servidor es un modelo para el desarrollo de sistemas de información, en el que las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos. Se denomina cliente al proceso que inicia el diálogo o solicita los recursos y servidor, al proceso que responde a las solicitudes.

Los principales componentes del esquema cliente/servidor son entonces los clientes, los servidores y la infraestructura de comunicaciones.

En este modelo, las aplicaciones se dividen de forma que el servidor contiene la parte que debe ser compartida por varios usuarios, y en el cliente permanece sólo lo particular de cada usuario.

Los clientes interactúan con el usuario, usualmente en forma gráfica. Frecuentemente se comunican con procesos auxiliares que se encargan de establecer conexión con el servidor,

enviar el pedido, recibir la respuesta, manejar las fallas y realizar actividades de sincronización y de seguridad.

Los clientes realizan generalmente funciones como:

- Manejo de la interface del usuario.
- Captura y validación de los datos de entrada.
- Generación de consultas e informes sobre las bases de datos.

Por su parte los servidores realizan, entre otras, las siguientes funciones:

- Gestión de periféricos compartidos.
- Control de accesos concurrentes a bases de datos compartidas.
- Enlaces de comunicaciones con otras redes de área local o extensa.
- Siempre que un cliente requiere un servicio lo solicita al servidor correspondiente y éste, le responde proporcionándolo. Normalmente, pero no necesariamente, el cliente y el servidor están ubicados en distintas computadoras. Los clientes se suelen situar en computadoras personales y los servidores en computadoras y/o estaciones de trabajo departamentales o de grupo.

Para que los clientes y los servidores puedan comunicarse se requiere una infraestructura de comunicaciones, que proporciona los mecanismos básicos de direccionamiento y transporte.

La mayoría de los sistemas cliente/servidor actuales, se basan en redes locales y por lo tanto utilizan protocolos no orientados a conexión, lo que implica que las aplicaciones deben hacer las verificaciones. La red debe tener características adecuadas de desempeño, confiabilidad, transparencia y administración. Entre las principales características de la arquitectura cliente / servidor, se pueden destacar las siguientes:

- El servidor presenta a todos sus clientes una interface única y bien definida.
- El cliente no necesita conocer la lógica del servidor, sólo su interface externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente.

En el sistema cliente /servidor de bases de datos, el cliente (que generalmente es otro servidor, por ejemplo, un servidor web) envía mensajes que son representados en solicitudes SQL hacia el servidor de bases de datos. Los resultados de cada orden de SQL son devueltos al cliente.

El SGDB se encarga de recolectar los datos desde su base de datos, no envía los registros completos, teniéndose un uso mucho más eficiente de la capacidad de procesamiento distribuida. Es usual que se generen aplicaciones en el cliente y en el servidor.

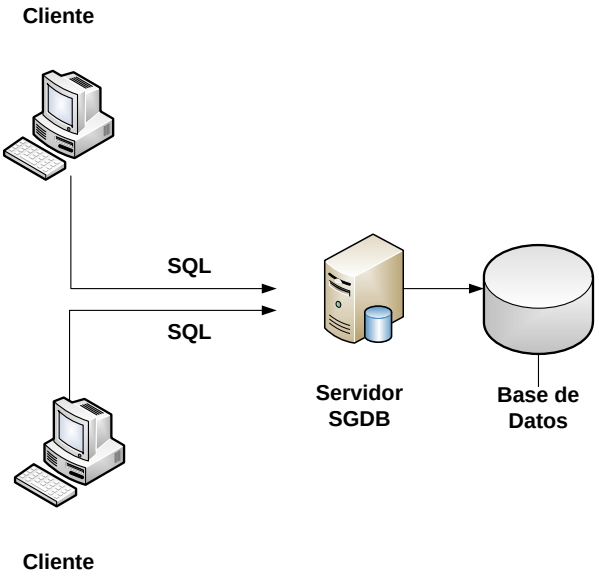

Figura 2.2.6.1 Arquitectura Cliente-Servidor

2.2.7 POSTGRESQL

PostgreSQL es un DBMS Objeto-Relacional (ORDBMS) que ha sido desarrollado de varias formas desde 1977. Comenzó como un proyecto denominado Ingres en la Universidad Berkeley de California. Ingres fue más tarde desarrollado comercialmente por la Relational Technologies/Ingres Corporation.

En 1986 otro equipo dirigido por Michael Stonebraker de Berkeley continuó el desarrollo del código de Ingres para crear un sistema de bases de datos objeto-relacionales llamado Postgres. En 1996, debido a un nuevo esfuerzo de código abierto y a la incrementada funcionalidad del software, Postgres fue renombrado a PostgreSQL, tras un breve periodo como Postgres95.

El proyecto PostgreSQL sigue actualmente un activo proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuidores de código abierto.

PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. PostgreSQL proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales tales como DB2 u Oracle.

A continuación se observa una breve lista de algunas de esas características, a partir de PostgreSQL 7.1.x.

2.2.7.1 CARACTERÍSTICAS Y VENTAJAS

DBMS Objeto-Relacional

PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL

declarativas, control de concurrencia multi-versión, soporte multi-usuario, transacciones, optimización de consultas, herencia y arrays.

Altamente Extensible

PostgreSQL soporta operadores, funciones métodos de acceso y tipos de datos definidos por el usuario.

Soporte SQL Comprensivo

PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.

Integridad Referencial

PostgreSQL soporta integridad referencial, que es utilizada para garantizar la validez de los datos de la base de datos.

API Flexible

La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++ y Pike.

Lenguajes Procedurales

PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido.

MVCC

MVCC, o Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control), es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. Cuando se usa algún SGDB con capacidades SQL, tal como MySQL o Access, hay ocasiones en que una lectura tiene que esperar para acceder a la información de la base de datos. La espera está provocada por usuarios que están escribiendo en la base de datos.

Resumiendo, el lector está bloqueado por los escritores que están actualizando registros. Mediante el uso de MVCC, PostgreSQL evita este problema por completo. MVCC está considerado mejor que el bloqueo a nivel de fila porque un lector nunca es bloqueado por un escritor.

En su lugar, PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos. PostgreSQL es capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles.

Cliente/Servidor

PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Ésta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

Write Ahead Logging (WAL)

La característica de PostgreSQL conocida como *Write Ahead Logging* incrementa la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la

base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del que podremos restaurar la base de datos.

Esto puede ser enormemente beneficioso en el caso de caída, ya que cualesquiera cambios que no fueron escritos en la base de datos pueden ser recuperados usando el dato que fue previamente registrado. Una vez el sistema ha quedado restaurado, un usuario puede continuar trabajando desde el punto en que lo dejó cuando cayó la base de datos.

Una lista breve de características técnicas que PostgreSQL ofrece:

- Cumple completamente con ACID
- Cumple con ANSI SQL
- Integridad referencial
- Replicación (soluciones comerciales y no comerciales) que permiten la duplicación de bases de datos maestras en múltiples sitios de replica
- Interfaces nativas para ODBC, JDBC, C, C++, PHP, Perl, TCL, ECPG, Python y Ruby
- Reglas
- Vistas
- Triggers
- Unicode
- Secuencias
- Herencia
- Outer Joins
- Sub-selects
- Una API abierta
- Procedimientos almacenados
- Soporte nativo SSL
- Lenguajes procedurales
- Respaldo en cliente
- Bloqueo a nivel mejor-que-fila
- Índices parciales y funcionales
- Autentificación Kerberos nativa
- Soporte para consultas con UNION, UNION ALL y EXCEPT
- Extensiones para SHA1, MD5, XML y otras funcionalidades
- Herramientas para generar SQL portable para compartir con otros sistemas compatibles con SQL
- Sistema de tipos de datos extensible para proveer tipos de datos definidos por el usuario, y rápido desarrollo de nuevos tipos

- Funciones de compatibilidad para ayudar en la transición desde otros sistemas menos compatibles con SQL

Ventajas

Instalación ilimitada. Es frecuente que las bases de datos comerciales sean instaladas en más servidores de lo que permite la licencia. Algunos proveedores comerciales consideran a esto la principal fuente de incumplimiento de licencia. Con PostgreSQL, nadie puede demandarlo por violar acuerdos de licencia, puesto que no hay costo asociado a la licencia del software.

Esto tiene varias ventajas adicionales:

- Modelos de negocios más rentables con instalaciones a gran escala.
- No existe la posibilidad de ser auditado para verificar cumplimiento de licencia en ningún momento.
- Flexibilidad para hacer investigación y desarrollo sin necesidad de incurrir en costos adicionales de licenciamiento.

Mejor soporte que los proveedores comerciales. Tiene una importante comunidad de profesionales y entusiastas de PostgreSQL de los que se puede obtener beneficios y contribuir.

Estabilidad y confiabilidad. En contraste a muchos sistemas de bases de datos comerciales, es extremadamente común que compañías reporten que PostgreSQL nunca ha presentado caídas en varios años de operación de alta actividad. Ni una sola vez. Simplemente funciona.

Extensible. El código fuente está disponible para todos sin costo. Si un equipo de trabajo necesita extender o personalizar PostgreSQL de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales. Esto es complementado por la comunidad de profesionales y entusiastas de PostgreSQL alrededor del mundo que también extienden PostgreSQL todos los días.

Multiplataforma. PostgreSQL está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y una versión nativa de Windows.

Diseñado para ambientes de alto volumen. PostgreSQL usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mejor respuesta en ambientes de grandes volúmenes. Los principales proveedores de sistemas de bases de datos comerciales usan también esta tecnología, por las mismas razones.

Herramientas gráficas de diseño y administración de bases de datos. Existen varias herramientas gráficas de alta calidad para administrar las bases de datos (pgAdmin, pgAcces) y para hacer diseño de bases de datos (Tora, Data Architect).

2.3 Lenguaje de Programación PHP.

Hoy en día, Internet es una herramienta esencial para todos tanto en el ámbito tecnológico y económico como en el cultural. Cualquier persona con una computadora y una conexión a Internet puede publicar texto, imágenes y multimedia por un costo relativamente bajo.

La infraestructura que compone Internet esta cimentada en el uso de software y hardware el cual se encuentra en continua evolución.

Productos de diferentes compañías y organizaciones trabajan en conjunto, no perfectamente, pero con un éxito muy notable basados en el cumplimiento de estándares técnicos públicos. De este contexto, surgieron los lenguajes m*iddleware*, que representan el enlace entre páginas web ordinarias interpretadas y desplegadas en el navegador de los visitantes de un sitio (llamados clientes) y los sistemas operativos y de bases de datos residentes en un equipo (llamado servidor). Esta "traducción" sobre demanda es comúnmente conocida como interpretación y la lleva a cabo un software servidor web.

2.3.1 Características y Ventajas

PHP es un lenguaje middleware cuyos programas se llaman scripts incrustados dentro del HTML. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. El término scripts se refiere a archivos en lenguajes que son almacenados de manera casi idéntica a cuando son escritos y que son traducidos en el momento de su ejecución.

Los principales lenguajes middleware competidores de PHP, son Perl, Microsoft Active Server Pages (ASP), Java Server Pages (JSP) y Macromedia Cold Fusion.

En comparación con estos productos, PHP ofrece muchas ventajas, incluyendo las siguientes:

Excelente desempeño. PHP es muy eficiente. Mediante la utilización de un servidor pequeño, es posible atender millones de solicitudes por día.

Facilidad en el aprendizaje y uso. La sintaxis esta basada en el lenguaje C y Perl principalmente. El conocimiento de estos lenguajes u otros como C++ o Java permite obtener resultados productivos con PHP en muy corto tiempo.

Portabilidad. PHP esta disponible para una gran variedad de sistemas operativos. PHP trabaja en versiones de sistemas Unix tanto comerciales (IRIX, Tru64, Solaris, etc.) como en FreeBSD o Linux en sus distintas distribuciones. El código escrito en PHP trabaja en general, con un mínimo de modificaciones en plataformas de Microsoft Windows.

El código fuente esta disponible. A diferencia de productos comerciales cerrados, en PHP se tiene acceso al código fuente. Si el desarrollo de algún proyecto requiere la adición o modificación de la funcionalidad proporcionada por PHP, no existe restricción para ello.

Interfaces para una gran variedad de sistemas de bases de datos. PHP proporciona conectividad nativa con una gran cantidad de manejadores de Bases de Datos. Además de Mysql, es posible conectarse directamente con PostgreSQL, mSQL, Oracle, dbm, filePro, Hyperwave, Informix, Interbase, Sybase, MS SQL Server, entre otros. Mediante ODBC (Open Database Conectivity) es posible conectarse con cualquier base de datos que cuente con un manejador para este protocolo.

Manejo de sesiones. PHP permite el manejo de sesiones persistentes en el servidor. Esta funcionalidad permite dar seguimiento y soporte a las actividades de un usuario autentificado en una sesión de su navegador.

2.3.2 Breve referencia al Lenguaje

Variables en PHP.

Para definir una variable en PHP, la sintaxis general es la siguiente:

$NOMBRE-VARIABLE

Ejemplo Variables numéricas.

Enteros $enteros=2007

Reales $real=3.1416

Ejemplo Variables alfanuméricas:

$cadena=”HOLA MUNDO”

Arreglos (arrays).

Son colecciones de variables bajo un mismo nombre de variable, lo que las identifica es un índice.

$carro[1]="vocho";

$carro[2]="atos";

$carro[3]="ibiza";

$carro[4]="stratus";

Variables de sistema en PHP

Este tipo de variables informan sobre el servidor y sobre el cliente. La información de estas variables es atribuida por el servidor y no es posible modificar sus valores directamente mediante un script. Estas son algunas de estas variables y la información que nos aportan:

Variable	Descripción
$HTTP_USER_AGENT	Informa principalmente sobre el sistema operativo, tipo y versión del navegador del cliente.
$HTTP_ACCEPT_LANGUAGE	Devuelve la o las abreviaciones de la lengua considerada como principal por el navegador.
$HTTP_REFERER	Indica la URL desde la cual el cliente tuvo acceso a la pagina
$PHP_SELF	Regresa una cadena con la URL del script que está siendo ejecutado.
$HTTP_GET_VARS	Es un arreglo que almacena los nombres y contenidos de las variables enviadas al script por URL o por el método GET de los formularios
$HTTP_POST_VARS	Es un arreglo que almacena los nombres y contenidos de las variables enviadas al script por el método POST de los formularios
$HTTP_COOKIES_VARS	Es un arreglo que almacena los nombres y contenidos de las cookies
$PHP_AUTH_USER	Almacena la variable *usuario* cuando se efectúa la entrada a páginas de acceso restringido
$PHP_AUTH_PW	Almacena la variable *password* cuando se efectúa la entrada a páginas de acceso restringido.
$REMOTE_ADDR	Muestra la dirección IP del visitante.
$DOCUMENT_ROOT	Devuelve el *path* físico en el que se encuentra alojada la página en el servidor.

Tabla 2.3.2.1 Variables de Sistema de PHP

Variables superglobales.

Estas variables hacen referencia a las mismas que se acceden por medio de los arrays del tipo *$HTTP_*_VARS*. A partir de PHP 5.0.0 se pueden desactivar con la directiva register_long_arrays.

Estas son algunas de estas variables y la información que aportan:

Variable	Descripción
$GLOBALS	Contiene una referencia a cada variable disponible en el script.
$_SERVER	Son variables definidas por el servidor Web ó directamente relacionadas con el entorno en donde el script se esta ejecutando
$_GET	Variables proporcionadas al script por medio de HTTP GET.
$_POST	Variables proporcionadas al script por medio de HTTP POST
$_COOKIE	Variables proporcionadas al script por medio de http cookies
$_FILES	Variables proporcionadas al script por medio de la subida de archivos vía HTTP.
$_ENV	Variables proporcionadas al script por medio del entorno.
$_REQUEST	Variables proporcionadas al script por medio de cualquier mecanismo de entrada del usuario y por lo tanto no se puede confiar en ellas.
$_SESSION	Variables registradas en la sesión del script.

Tabla 2.3.2.2 Variables SuperGlobales de PHP

Manejo de arreglos con PHP

Un arreglo es un conjunto de elementos bajo un mismo nombre de variable, cada elemento está catalogado por medio de una clave y deberá tener un valor.

Existen varias formas de declarar arreglos, las cuales se mencionan a continuación.

SINTAXIS:

```
$NOMBRE_ARREGLO[CLAVE1] = VALOR1;

$NOMBRE_ARREGLO[CLAVE2] = VALOR2;

$NOMBRE_ARREGLO[CLAVEN] = VALORN;
```

Otra forma de definir un arreglo es la siguiente:

```
$NOMBRE_ARREGLO = ARRAY("CLAVE1"=> VALOR1,"CLAVE2" => VALOR2" …);
```

Es posible anidar arreglos, es decir, se puede tener un arreglo que tenga como elementos a otros arreglos.

```
$MONEDA["MEXICO"] = "PESO";

$MONEDA["FRANCIA"] = "FRANCO";

$MONEDA = ARRAY("MEXICO"=> "PESO","FRANCIA" => FRANCO");

$NOMBRE_ARREGLO = ARRAY("CLAVE1"=> VALOR1,"CLAVE2" => VALOR2" …);
```

Funciones básicas para el manejo de arreglos

Existen muchas funciones para el manejo de arreglos, a continuación se describen las funciones básicas.

Función	Descripción
array_values (arreglo)	Lista los valores contenidos en *arreglo*
asort(arreglo) y **arsort(arreglo)**	Ordena por orden alfabético directo o inverso en función de los valores
count(arreglo)	Regresa el número de elementos de nuestro arreglo.
ksort(arreglo) y **krsort(arreglo)**	Ordena por orden alfabético directo o inverso en función de las claves.
list ($variable1,** **$variable2…) = arreglo	Asigna cada variable a cada uno de los valores del array.
next(arreglo), prev(arreglo), **reset(arreglo)** y **end(arreglo)**	Permiten moverse dentro del arreglo con un puntero hacia delante, atrás, al principio y al final.
each(arreglo)	Da el valor y la clave del elemento en el que se encuentra y mueve al puntero al siguiente elemento.
array_slice( )	Se utiliza cuando se quieren recortar algunos elementos del arreglo, sabiendo los índices de las casillas que se desean conservar. Recibe tres parámetros: el *arreglo*, el *índice del primer elemento* y el *número de elementos a tomar*, siendo este último parámetro opcional.
array_shift( )	Esta función extrae el primer elemento del arreglo y lo devuelve. Además, acorta la longitud del arreglo eliminando el elemento que estaba en la primera casilla. Solo recibe como parámetro el nombre del arreglo.

Tabla 2.3.2.3 Funciones Básicas para el manejo de arreglos

Cadenas

Para asignar a una variable un contenido de tipo cadena, se debe escribir entre comillas. Si se quiere ver en pantalla el valor de una variable o bien un mensaje cualquiera se puede utilizar la instrucción *echo*.

Sintaxis:

$CADENA = "ESTA ES LA INFORMACIÓN DE MI VARIABLE"

Concatenación de cadenas

Para concatenar varias cadenas, simplemente se debe poner un "." entre ellas.

Ejemplo:

<?

$VAR1="HOLA";

$VAR2= "MUNDO";

ECHO VAR1." ".VAR2;

?>

LA SALIDA ES : HOLA MUNDO

Funciones básicas para el manejo de cadenas

A continuación se describen algunas funciones básicas para el manejo de cadenas:

Función	Descripción
strcmp(cadena1,cadena2)	Esta función compara cadenas. Devuelve < 0 si *cadena1* es menor que *cadena2*; >0 si *cadena1* es mayor que *cadena2* y 0 si son iguales.
strlen(cadena)	Esta función regresa la longitud de la cadena indicada
strtolower(cadena)	Pasa a minúsculas una cadena.
strtoupper(cadena)	Devuelve la *cadena* con todas sus letras en mayúsculas.
substr(cadena, comienzo, longitud)	Devuelve la porción de *cadena* especificada por los parámetros *comienzo* y *longitud*
trim(cadena)	Elimina espacios en blanco (u otros caracteres) del principio y final de una cadena.

Tabla 2.3.2.4 Funciones básicas para el manejo de cadenas

Funciones

Una función es un conjunto de instrucciones que explotan ciertas variables y realiza una tarea específica. Es posible crear nuestras propias funciones de la siguiente forma:

Sintaxis:

```
FUNCTION NOMBRE_FUNCION ($ARG_1, $ARG_2, ..., $ARG_N)
{
BLOQUE DE INSTRUCCIONES;
//VALOR QUE REGRESA LA FUNCIÓN
RETURN $RETVAL;
}
```

Operadores

Los operadores permiten crear, modificar y comparar variables dentro de un programa. A continuación se describen los diferentes tipos de operadores que se pueden utilizar en PHP.

Operadores aritméticos

Permiten realizar operaciones numéricas con nuestras variables:

+ Suma

- Resta

***** Multiplicación

/ División

% Módulo (Devuelve el residuo de la división)

Operadores de comparación

Se utilizan para comparar dos variables y verificar si cumple o no la propiedad del operador.

== Igualdad

!= Desigualdad

< Menor que

> Mayor que

<= Menor igual que

>= Mayor igual que

Operadores lógicos

Se usan en combinación con los operadores de comparación cuando la expresión de la condición lo requiere.

&& AND

|| OR

! NOT

Operadores de incremento

Sirven para aumentar o disminuir de una unidad el valor de una variable

++$variable Aumenta en 1 el valor de $variable

--$variable Reduce en 1 el valor de $variable

Operadores combinados

Una forma habitual de modificar el valor de las variables es mediante los operadores combinados:

$VARIABLE += 10 SUMA 10 A $VARIABLE

$VARIABLE -= 10 RESTA 10 A $VARIABLE

$VARIABLE.= "AÑADO" //CONCATENA LAS CADENAS $VARIABLE Y "AÑADO"

Sentencia IF-ELSE

En esta estructura de control primero se evalúa la condición, si la condición se cumple se ejecuta el *bloque de instrucciones 1,* si no, se ejecutará el *bloque de instrucciones 2.*

SINTAXIS:

IF (CONDICIÓN) { BLOQUE DE INSTRUCCIONES 1; } ELSE

{ BLOQUE DE INSTRUCCIONES 2; }

Ejemplo:

<?

$X=55;

$Y=26;

IF($X > $Y) { ECHO "X ES MAYOR QUE Y"; } ELSE

{ ECHO "X NO ES MAYOR QUE Y"; } ?>

SALIDA: X ES MAYOR QUE Y

Sentencia WHILE

Esta estructura permite repetir instrucciones mientras se cumpla una condición.

Sintaxis: *while* (condición)

{ BLOQUE DE INSTRUCCIONES A REPETIR; }

Ejemplo:

<?

$CONTADOR=0;

WHILE ($CONTADOR<5)

{ ECHO $CONTADOR; $CONTADOR++; } ?>

SALIDA: 0 1 2 3 4

Sentencia FOR

La estructura *for* funciona de la siguiente manera: La primera expresión *expr1* se evalúa incondicionalmente una vez al principio del ciclo. Al comienzo de cada iteración, se evalúa *expr2*. Si se cumple la condición, el ciclo continúa y se ejecuta el *bloque de instrucciones*. Si no se cumple, la ejecución del ciclo finaliza. Al final de cada iteración, se evalúa *expr3*.

Sintaxis:

```
FOR (EXPR1; EXPR2; EXPR3)
{ BLOQUE DE INSTRUCCIONES; }
```

Ejemplo:

```
<?
$CARRO[0]="INICIO";
$CARRO[1]="VOCHO";
$CARRO[2]="ATOS";
$CARRO[3]="IBIZA";
$CARRO[4]="STRATUS";
FOR($I=0;$I<5;$I++)
{ ECHO $CARRO[$I]. " "; }
?>
```

SALIDA : INICIO VOCHO ATOS IBIZA STRATUS

Sentencia FOREACH

Esta estructura repetitiva permite recorrer fácilmente los valores de un arreglo.

Sintaxis:

```
FOREACH ($ARREGLO AS $CLAVE=>$VALOR)
{ BLOQUE DE INSTRUCCIONES; }
```

Ejemplo:

```
<?
$MONEDA = ARRAY("PAIS"=>"MONEDA","MEXICO"=> "PESO","FRANCIA" =>
"FRANCO","USA"=>"DOLAR");
FOREACH ($MONEDA AS $CLAVE=>$VALOR)
{ ECHO $CLAVE ." => ". $VALOR . "<BR />"; } ?>
```

SALIDA:

PAIS => MONEDA
MEXICO => PESO
FRANCIA => FRANCO
USA => DÓLAR

Paso de variables por la URL

Para pasar las variables de una página a otra se puede hacer introduciendo dicha variable dentro del enlace hipertexto de la página destino.

Sintaxis:

<A HREF="DESTINO.PHP?VARIABLE1=VALOR1&VARIABLE2=VALOR2&...">

MI ENLACE

</A>

Ejemplo:

SALUDO.HTML

<HTML> <HEAD> <TITLE>SALUDO</TITLE> </HEAD>

<BODY> <A HREF="SALUDO.PHP?SALUDO=HOLA&TEXTO=MUNDO">PASAR SALUDO</A> </BODY>

</HTML>

SALUDO.PHP

<? ECHO "EL SALUDO ES $SALUDO $TEXTO
\N"; ?>

En la primera página nos generará un enlace al archivo saludo.php y mandará dos variables ($saludo y $texto) y el valor de estas se mostrará

SALIDA: EL SALUDO ES HOLA MUNDO

Procesar variables de Formularios

Si se quieren procesar las variables provenientes de un formulario, PHP permite hacerlo de una manera sencilla. Primero se presentará en una página el formulario clásico a llenar y estas variables las procesa un segundo archivo.

Ejemplo:

FORMULARIO.HTML

<HTML><HEAD><TITLE>FORMULARIO.HTML</TITLE></HEAD>

<BODY>

<FORM METHOD="POST" ACTION="DESTINOFORM.PHP"> NOMBRE

<INPUT TYPE="TEXT" NAME="NOMBRE">
 APELLIDOS

<INPUT TYPE="TEXT" NAME="APELLIDOS">

<INPUT TYPE="SUBMIT" VALUE="ENVIAR" /></FORM></BODY></HTML>

DESTINOFORM.PHP

```
<?
ECHO "VARIABLE \$NOMBRE: $NOMBRE <BR>\N";
ECHO "VARIABLE \$APELLIDOS: $APELLIDOS <BR>\N"
?>
```

El formulario tendrá dos cajas de texto, *nombre* y *apellidos,* que también serán los nombres de las variables a procesar en el script *destinoform.php.* El script *destino.php* simplemente imprimirá el contenido de estas dos variables.

Formulario.html

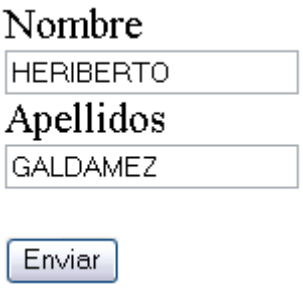

Figura 2.3.2.1 Formulario HTML

SALIDA:

VARIABLE $NOMBRE: HERIBERTO
VARIABLE $APELLIDOS: GALDAMEZ

Sesiones

Una sesión resulta muy práctica en los casos en los que se quiere conservar una variable en varios scripts diferentes y distantes unos de otros.

Características de las variables de sesión:

- Estas variables residen en el servidor
- Son específicas de un solo usuario definido por un identificador.
- Pueden ser utilizadas en la globalidad de nuestras páginas.

Para iniciar una sesión se puede hacer de dos formas distintas:

1. Declarar la apertura de sesión por medio de la función **session_start()**.

Esta función crea una nueva sesión para un nuevo visitante o bien recupera la que está siendo llevada a cabo.

2. Declarar una variable de sesión por medio de la función *session_register('variable')*. Esta función, además de crear o recuperar la sesión para la página en la que se incluye también sirve para introducir una nueva variable de tipo sesión.

Las sesiones deben ser iniciadas al principio del script. Antes de abrir cualquier etiqueta o de imprimir cualquier cosa. En caso contrario recibiremos un error.

Funciones importantes para la gestión de sesiones

Función	**Descripción**
session_id( )	Nos devuelve el identificador de la sesión.
session_destroy( )	Da por abandonada la sesión eliminando variables e identificador.
session_start()	crea una sesión
session_unregister('variable')	Abandona una variable sesión.

Tabla 2.3.2.5 Funciones para la gestión de sesiones.

2.3.3 Interacción PHP – PostgreSQL

El sistema de acceso y manipulación de bases de datos desde PHP es similar al de otros lenguajes de script: establece la conexión con la base de datos, ejecuta las sentencias de consulta o modificación y finalmente cierra la conexión.

A continuación se muestran algunas instrucciones utilizadas y su descripción.

Conexión con bases de datos PostgreSQL

pg_connect(). Establece la conexión a una base de datos de PostgreSQL. Cada uno de los argumentos debe ser una cadena entrecomillada.

La conexión al servidor se cerrará en cuanto la ejecución del script acabe, a menos que se cierre antes con la función pg_close.

SINTAXIS CLASICA:

PG_CONNECT (STRING SERVIDOR, STRING PUERTO, STRING BASEDEDATOS,STRING USUARIO,

STRING PASSWORD)

Pg_close. Cierra la conexión a la base de datos de PostgreSQL asociada al identificador de conexión especificado. Si no se especifica un identificador de conexión, se asume la última conexión abierta.

SINTAXIS

PG_CLOSE(INT IDCONEXIÓN)

Pg_exec. Devuelve un índice de resultado si se pudo ejecutar la consulta, o **false** en caso de fallo o si *connection* no es un índice de conexión válido. Envía una sentencia SQL a la base de datos PostgreSQL especificada por el índice de conexión,

connection debe ser un índice válido devuelto por pg_Connect(). El valor de devuelto por esta función es un índice para ser usado al acceder a los resultados de la consulta desde otras funciones PostgreSQL.

SINTAXIS:

PG_EXEC (INT IDCONEXIÓN, STRING CONSULTA)

pg_fetch_array. Devuelve un arreglo que se corresponde con la fila obtenida, o **false** si no hay más filas. El primer parámetro es un contenedor de resultados, este puede ser el resultado de la función pg_exec, el parámetro de numerofila es el número de fila dentro del contenedor de resultados, El tercer parámetro opcional tiporesultado en pg_fetch_array() es una constante y puede tomar cualquiera de los siguientes valores: PGSQL_ASSOC, PGSQL_NUM, y PGSQL_BOTH.

SINTAXIS:

PG_FETCH_ARRAY (INT RESULTADO, INT NUMEROFILA [, INT TIPORESULTADO])

pg_fetch_object() es parecida a la función pg_fetch_array(), con una diferencia , se devuelve un objeto, en vez de un arreglo. Indirectamente, eso significa que solo puedes acceder a los datos por medio de su nombre de campo, y no a través de sus posiciones, nos regresa un objeto cuyas propiedades se corresponden con los campos de la fila obtenida, o **false** si no hay más filas. La sintaxis es la misma que la función pg_fetch_array

SINTAXIS:

PG_FETCH_OBJECT(INT RESULTADO, INT NUMEROFILA [, INT TIPORESULTADO])

pg_num_rows. Devuelve el número de filas en un resultado PostgreSQL. El parámetro es un identificador de resultado PostgreSQL válido devuelto por pg_Exec(). En caso de error se devuelve -1.

SINTAXIS:

PG_NUMROWS (INT RESULTADO)

2.4 SERVIDOR APACHE

El **SERVIDOR APACHE** es el servidor más popular actualmente, su nombre Apache es **"A PAtCHy Server",** desde su origen a evolucionado hasta convertirse en uno de los mejores servidores en términos de eficiencia, funcionalidad y velocidad. Apache ha demostrado ser más rápido que muchos otros servidores libres, considerándole un servidor de código fuente abierto que ha llegado a competir de cerca con los mejores servidores comerciales como son: Microsoft y Netscape, como plataforma de servidores Web.

2.4.1 CARACTERÍSTICAS Y VENTAJAS.

La historia de Apache se remonta a febrero de 1995, donde empieza el proyecto del grupo Apache, el cual esta basado en el servidor Apache httpd de la aplicación original de NCSA. El desarrollo de esta aplicación original se estancó por algún tiempo tras la marcha de Rob McCool por lo que varios webmaster siguieron creando sus parches para sus servidores web hasta que se contactaron vía correo electrónico para seguir en conjunto el mantenimiento del servidor web, fue ahí cuando formaron el grupo Apache.

Fueron Brian Behlendorf y Cliff Skolnick quienes a través de una lista de correo coordinaron el trabajo y lograron establecer un espacio compartido de libre acceso para los desarrolladores.

La primera versión (0.6.2) de Apache que fue distribuida al público se estrenó en abril de 1995. La versión 1.0 se estrenó el 1 de diciembre de 1995. Aquella primera versión y sus sucesivas evoluciones y mejoras alcanzaron una gran implantación como software de servidor inicialmente solo para sistemas operativos UNIX y fruto de esa evolución es la versión para Windows .

Las características más importantes son:

• Apache trabaja con gran cantidad de Perl, **PHP** y otros lenguajes de script.

• Perl destaca en el mundo del script y Apache hace lo mismo con Perl, tanto con soporte CGI como con soporte mod perl.

• Funciona sobre muchas plataformas (como son: Unix, Linux, Vms, Win32,OS2)

• Módulos cargados dinámicamente.

• CGI, Perl (ejemplo: formularios, diccionarios en línea, entre otros)

• **PHP**5 + Bases de datos

• SSL: transacciones seguras

• Soporte para host virtuales

• Alto desempeño

Ventajas

Debido a su facilidad de configuración, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. Algunas ventajas de Apache son:

- Corre en una gran cantidad de Sistemas Operativos, lo que lo hace prácticamente universal.
- Apache es una tecnología gratuita de código fuente abierto. El hecho de ser gratuita es importante pero no tanto como que se trate de código fuente abierto.
- Esto le da una transparencia a este software de manera que si queremos ver que es lo que estamos instalando como servidor , lo podemos saber, sin ningún secreto, sin ninguna puerta trasera.
- Apache es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor Web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este, y están ahí para que los instalemos cuando los necesitemos. Otra cosa importante es que cualquiera que posea una experiencia decente en la programación de C o Perl puede escribir un modulo para realizar una función determinada.
- Apache trabaja con gran cantidad de lenguajes de script como Perl, PHP y otros. Perl destaca en el mundo del script y Apache utiliza su parte del pastel de Perl tanto con soporte CGI como con soporte mod perl. También trabaja con Java y páginas jsp. Teniendo todo el soporte que se necesita para tener páginas dinámicas.
- Apache te permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.
- Tiene una gran capacidad de configuración en la creación y gestión de logs. Apache permite la creación de ficheros de log a medida del administrador, de este modo puedes tener un mayor control sobre lo que sucede en tu servidor.
- Se pueden extender las características de Apache hasta donde nuestra imaginación y conocimientos lleguen.

CAPITULO III. PLANTEAMIENTO DEL PROBLEMA Y PROPUESTA DE SOLUCIÓN.

En la mayoría de los programas de servicio social, se necesita realizar varios trámites administrativos por parte de los prestadores así como de la institución o dependencia que los solicita, lo que genera un gran número de documentos entre ellos formas de registros, listas, gafetes, etc. y toda su información en particular. Aunque en cada instituto o dependencia existen formas impresas para manejar algunos aspectos muy específicos del control administrativo de servicio social como la selección de aspirantes o la asignación de funciones, muchos trámites y procedimientos de este control se realizan de forma manual. Esto genera que una misma información sea capturada dos o más veces, dando lugar a errores y duplicidades.

3.1 PROBLEMÁTICA ACTUAL.

En el programa de Servicio Social UNAM-PERAJ se han identificado etapas muy importantes y se han observado tareas que se enfocan directamente al control administrativo. Las etapas que se han distinguido durante el análisis son:

- Inicio de Servicio Social
- Realización de Servicio Social
- Fin de Servicio Social

En la tabla 3.1.1 se muestran las actividades más significativas de cada etapa y las consecuencias de llevar un control manual dentro de la administración del Servicio Social.

Inicio de Servicio Social

Administrativo	**Prestador**
1. Publicar Convocatoria	
	2. Recoger formato de inscripción
	3. Entregar formato de inscripción
4. Seleccionar aspirantes	
5. Asignar Tareas	
6. Publicar Seleccionados	
	7. Realizar la petición de la carta de aceptación en caso de ser seleccionado
8. Entregar Carta de Registro	
	9. Recoger Carta de Registro

Tabla 3.1.1 Inicio de Servicio Social

A continuación se describe brevemente cada uno de los bloques y las observaciones del análisis realizado.

1. El departamento de control administrativo publicará el periodo de inscripción y selección para el programa de Servicio Social UNAM-PERAJ.
   - El aspirante debe consultar fechas y requisitos.
   - No todos los posibles aspirantes pueden enterarse sin consultar la gaceta (a veces insuficientes) o los carteles publicados.
   - La convocatoria no tiene el suficiente impacto.

2. El aspirante recoge su formato de inscripción.
   - Los alumnos no alcanzan a recoger la papelería en las fechas señaladas.
   - Las filas para recoger la papelería pueden ser largas ciertos días del periodo de inscripción.

3. El alumno deberá entregar los formatos de inscripción al departamento de control administrativo para su debida inscripción.

- . El personal de control administrativo archiva de manera manual la documentación por aspirante, lo cual provoca tiempos de atención muy grandes y por lo tanto largas filas de espera.
- El aspirante llena de manera incorrecta el formato de inscripción, provocando entregar nueva papelería y un retraso en la inscripción.

4. El departamento de control administrativo deberá realizar una selección de aspirantes de acuerdo a un perfil preestablecido.
   - Realizar de manera manual la búsqueda de formatos de registro, sin ningún tipo de filtro.
   - El aspirante no tiene el perfil indicado para prestar el servicio social.

5. El departamento definirá las funciones de acuerdo a un perfil dentro del programa de servicio social.
   - Se realizan de manera manual la asignación de funciones.
   - Puede haber duplicación de funciones en un solo aspirante

6. Publicar Aspirantes Seleccionados
   - El aspirante seleccionado no puede enterarse de su selección

7. El aspirante deberá de pedir su carta de aceptación al control administrativo en caso de ser seleccionado.
   - El aspirante proporciona datos incorrectos para realizar su carta de aceptación

8. Entregar Carta de aceptación por parte del departamento de control administrativo
   - Buscar manualmente el formato de registro del aspirante aceptado.
   - El aspirante lleno de manera incorrecta su formato de registro.

9. Recoger carta de aceptación.
   - La carta de aceptación tiene datos incorrectos

Una vez que ha concluido el periodo de inicio de servicio social, control administrativo entra en la etapa de la realización del Servicio Social. Durante está etapa la demanda hacia el departamento disminuye considerablemente, sin embargo, los procesos son repetitivos y la

falta de comunicación y mejora continua puede provocar que un proceso sea tedioso y poco ágil.

Se muestra un ejemplo en la tabla 3.1.2.

Realización del Servicio Social	
Administrativo	**Prestador**
2. Entrega de apoyo Económico (1 x mes)	
	4. Recoger apoyo Económico (1 x mes)

Tabla 3.1.2 Realización del Servicio Social

1. La entrega del apoyo económico será de manera personal y en forma de cheque.
    - La búsqueda para integrar la nómina será de manera manual

2. El prestador del servicio social recogerá el apoyo económico una vez al mes durante la realización del servicio social
    - Cualquier dato incorrecto en el cheque se tendrá que corregir de la misma forma, pero ocurrirá un contratiempo, ya que se debe informar al banco que lo expide y hacer un nuevo cheque.

Como podemos observar, los datos proporcionados por el aspirante en la primera etapa son muy importantes, ya que con estos se generarán documentos necesarios que pueda concluir su registro y siga participando en el programa.

La etapa final se refiere al fin de servicio social; durante esta etapa control administrativo debe elaborar los diferentes documentos oficiales que acrediten el fin de servicio social para los prestadores. Un ejemplo es la tabla 3.1.3

Fin del Servicio Social	
Administrativo	**Prestador**

1. Entrega de carta Termino del Servicio Social

2. Recoger Carta de Termino del servicio Social

Tabla 3.1.3 Fin del Servicio Social

Como se observa en el análisis, el departamento de control administrativo del servicio social UNAM-PERAJ no cuenta con sistemas automatizados para llevar a cabo sus tareas de gestión y control, debido a que la mayoría de los procesos de información son elaborados manualmente y como consecuencia:

- Las tareas administrativas se vuelven lentas.
- Los reportes solicitados no se elaboraran en forma expedita.
- Existe la posibilidad de duplicidad de información.
- Usualmente se cometen errores por parte del alumno o la institución educativa.

Es de vital importancia, modificar procesos e implementar un sistema que los automatice de tal manera que el tiempo disminuya considerablemente, se eviten los errores de duplicidad de información, traspapeleo de expedientes, y la información sea consistente con el fin de apoyar al área administrativa en las diferentes actividades, por ejemplo:

- Evitar el manejo de papelería.
- Hacer eficiente la consulta de información y generación de reportes de las diferentes etapas.
- Desarrollar un ambiente de trabajo moderno, amigable y aprovechando las bondades de la tecnología actual.

3.2 Identificación De Requerimientos Generales Y Particulares

Un requerimiento es un conjunto de propiedades o restricciones definidas con precisión, que un sistema de software debe satisfacer.

Los requerimientos son la pieza fundamental en un proyecto de desarrollo de software, ya que es la base para:

- Planear el proyecto y los recursos que se usaran en el.
- Especificar el tipo de pruebas que se habrán de realizar al sistema.
- Planear la estrategia de prueba a la que habrá de ser sometido el sistema.
- Son el fundamento del ciclo de vida del proyecto

Requerimientos Generales

Los requerimientos generales para este sistema son:

- Se espera tener desarrollada una aplicación que proporcione a las instituciones escolares, herramientas para apoyar a la administración del programa de servicio social UNAM-PERAJ,
- En todas las áreas se deberá tener la facilidad y claridad de acceder a registros para su actualización, en caso de proceder, de acuerdo a la naturaleza del campo y facultades del operador.
- El Sistema deberá ser desarrollado con herramientas de distribución gratuita (PHP y POSTGRESQL en nuestro caso) que posean la más alta tecnología en cuanto al manejo y presentación de la información; cabe aclarar que el uso de este software es requerido por la institución.
- El Sistema deberá ser operado a través de cualquier navegador comercial desde el cual el usuario pueda acceder desde su casa, escuela o café Internet, para realizar cada una de las actividades que el Sistema propone.
- A las peticiones de registro de amigos se les asignara un identificador único, de manera que se pueda tener un control de operaciones.
- El acceso a registro de amigos, tutores, coordinadores, e integrantes del apoyo logístico se realizara mediante nombres de usuarios y contraseñas preestablecidas.

- Los gafetes serán generados como documentos PDF en versiones imprimibles.
- Se generará documentos en hojas de calculo (Excel) con la información Personal de los participantes en el programa.
- Se generarán las cartas de registro y término de servicio social
- Esta base de datos sólo será administrada por los usuarios autorizados.

- Ningún empleado o alumno podrá acceder a ella.
- La información que contenga debe ser verdadera siempre actualizada y la necesaria para realizar los procesos que dependen de ella.
- Debe permitir ser usada en otras plataformas.

Requerimientos Particulares.

Los requerimientos particulares para el sistema son:

- Captura y actualización de información de amigos

  La clave para ingresar al sistema será preestablecida

- Captura y actualización de información de tutores, coordinadores y apoyo logístico

  Deberán ser alumnos o ex alumnos de la UNAM con un número de cuenta de 9 dígitos

- Generar e imprimir gafetes según la función en el programa

  Los gafetes se generaran en formato PDF, por lo cual se deberá contar con algún programa que permita visualizar archivos PDF.

- Generar reportes de la información de los participantes en el programa según su función en el sistema.

  Los reportes se generaran en formato xls (hoja de calculo). Se deberá contar con algún programa que permita visualizar archivos xls

- Asignación de tutorías para cada amigo

  Registrar un tutor por amigo.

  El tutor debe estar registrado previamente en el sistema.

  El perfil del tutor debe ser el adecuado para poder desarrollar su función

- Generar las cartas de registro y término de servicio social de prestadores del servicio social.
- Los documentos se generaran en formato doc o rtf

  Se deberá contar con algún procesador de textos que permita visualizar estos archivos.

3.3 Recopilación Y Análisis De La Información.

Para lograr que se cumplan los objetivos con el desarrollo del sistema, tenemos que recopilar información sobre los requisitos del mismo. De hecho, entre más información logremos reunir, tendremos mayor posibilidad de éxito en lograr los objetivos propuestos.

La razón principal por la cual es necesario reunir información acerca de cómo maneja actualmente el departamento administrativo su información es entender correctamente el comportamiento y las necesidades del manejo de la información.

Los usuarios o personas que desempeñan un papel determinante en el manejo de la información son básicamente personal administrativo que hacen uso directo de la información y pueden ser considerados como los principales usuarios, por ser capaces de poder suministrar u obtener información necesaria para ser manipulada.

Actualmente los amigos y tutores hacen uso de los siguientes formatos para el manejo de su información:

FICHA DE IDENTIFICACION DEL AMIG@

PROGRAMA DE SERVICIO SOCIAL TUTORIAL UNAM-PERAJ

Adopta un amig@

Nombre completo			
Edad			
Nombre del Padre			
Nombre de la Madre			
Dirección Calle y Número			
Colonia		C.P.	
Delegación		Entidad federativa	
Teléfono			
Fecha de Nacimiento		Sexo	

Datos personales

En caso de emergencia avisar

Nombre completo	
Teléfono	
Parentesco	

Información académica

Nombre de la escuela			
Grado		Grupo	

Promedio del periodo anterior	
Nombre del Profesor	
Materia que más te gusta	
¿Por qué?	
Materia que menos te gusta	
¿Por qué?	

Información personal

¿Cómo es la relación con tu familia? (Marca con una "X")

Muy Buena	
Buena	
Regular	
Mala	
Muy Mala	

Con qué miembro de tu familia te llevas mejor? (Marca con una "X")

Papá			
Mamá			
Papá y Mamá			
Hermanos (as)			
Otro		¿Cuál?	

¿Con qué miembro de tu familia no te llevas bien? (Marca con una "X")

Papá			
Mamá			
Papá y Mamá			
Hermanos (as)			
Otro		Cuál	

Gustos y preferencias

¿Qué te gustaría ser de grande?	
¿Qué te gusta hacer en tu tiempo libre?	

¿Te gustaría hacer algo diferente?	SI	
	NO	
¿Qué?		

Figura 3.3.1 Ficha para el registro de un amig@

FICHA DE IDENTIFICACION DEL TUTOR

PROGRAMA DE SERVICIO SOCIAL TUTORIAL UNAM-PERAJ

Adopta un amig@

Datos personales

Numero de cuenta			
Nombre Completo			
RFC			
CURP			
Dirección Calle y Número			
Colonia		C.P.	
Delegación		Entidad federativa	
Teléfono			
Fecha de Nacimiento		Sexo	
¿Eres Pronabes?	SI		
	NO		

Información académica

Facultad o Escuela	

Área académica	
Carrera	
Porcentaje de créditos	
Promedio	
Semestre	

Figura 3.3.2 Ficha para el registro de un alumno

3.4 Identificación Del Problema.

La mayor necesidad a cubrir con el nuevo Sistema, es la automatización de los procesos llevados a cabo para el registro de amig@s, tutores, coordinadores e integrantes del apoyo logístico involucrando de esta manera a los prestadores del servicio social y amig@s en el registro de sus datos en el programa y no depender solo del personal encargado.

3.4.1 Proceso de inscripción de un amig@.

Actualmente para el proceso de inscripción de un amig@ se realizan los siguientes pasos:

- Recoger en DGOSE el formato de inscripción.
- Llenar a mano el formato de inscripción.
- Entregar el formato de inscripción y Entregar documentación correspondiente en DGOSE
- Generar gafete con sus datos personales.

Como podemos observar, el proceso de registro de un amig@ puede ser sencillo, sin embargo, la forma de registro contiene campos personales que el amig@ puede desconocer, además de necesitar la orientación de alguno de sus padres.

Este proceso no es eficiente por lo que se pretende sea realizado de la siguiente manera:

- Llenar formulario por medio de Internet
- Consultar documentación por medio de Internet
- Recoger gafete el mismo día que el amig@ entregue su documentación en DGOSE.

El amig@ podrá ingresar al sistema vía Web con la dirección URL proporcionada en DGOSE y llenará un formulario con sus datos personales, de esta manera podrá contar con el apoyo de sus padres o maestros

De esta manera se pretende reducir tiempos y errores en el llenado de los formularios de inscripción manualmente y se pueda contar con una información correcta.

El sistema asignara un identificador único a cada amig@ para poder localizarlo de una manera mas rápida.

3.4.2 Proceso de inscripción de un tutor, coordinador y apoyo logístico.

El proceso de inscripción de un tutor, coordinador o apoyo logístico es muy similar al de un amig@, son prácticamente los mismos pasos:

- Recoger en DGOSE el formato de inscripción.
- Llenar a mano el formato de inscripción.
- Entregar el formato de inscripción y Entregar documentación correspondiente en DGOSE
- Generar carta de aceptación y de término de servicio social
- Generar gafete con sus datos personales.

El proceso de registro se puede mejorarse de igual manera mediante el sistema siguiendo los mismos pasos que un amig@

- Llenar formulario por medio de Internet
- Consultar documentación por medio de Internet
- Recoger las cartas de registro y término de servicio social
- Recoger gafete

El tutor, coordinador o apoyo logístico podrá ingresar al sistema vía Web con la dirección URL proporcionada en DGOSE y llenará un formulario con sus datos personales y con esto también se pretende reducir tiempos en el llenado de los formularios de inscripción manualmente y se pueda contar con una información correcta.

En este caso, el identificador para cada alumno o exalumno que se inscriba como tutor, coordinador o apoyo logístico será su número de cuenta, así que nadie podrá desempeñar dos funciones al mismo tiempo dentro del programa

3.4.3 Asignación de tutorías.

La asignación de un tutor a un amig@ la realiza el personal administrativo una vez que haya terminado el periodo de inscripción de tutores y amig@s.

Este proceso se hacia manualmente de la siguiente manera:

- Se contabilizaban los amig@s registrados en el programa
- Se contabilizaban los tutores registrados en el programa
- Se realizaba la asignación de manera aleatoria

Cuando se hace una asignación de este manera, existe la opción de que se puede repetir un amig@ o un tutor en una asignación

Para que éste proceso sea eficiente, debe realizarse de la siguiente manera:

- Elegir un tutor por carrera o por escuela /facultad
- Elegir un amig@ por escuela
- Realizar la asignación

Esto será realizado por el personal administrativo de la institución ingresando al sistema, con la certeza de que solo se haga una asignación en cada tutoría, con la opción de realizar cambios de una manera efectiva.

3.4.4 Administración en general de la información

Por medio del sistema, el personal administrativo de la institución tendrá acceso a la información de cada persona que se haya registrado en el programa, realizando búsquedas sencillas y eficientes, mediante diferentes parámetros y teniendo la opción de poder guardar modificaciones en cada registro, podrá descargar diferentes tipos de listados con la información personal en formato XLS para apoyar la elaboración de diversos documentos, por ejemplo, las cartas de registro de servicio social o los apoyos económicos, podrá generar gafetes con los datos personales de cada participante en formato PDF además de las cartas de registro y término de servicio social

3.5 Propuesta de solución

La necesidad de automatizar ciertos procesos provoca cambios en diversos campos, tecnológicos, administrativos, sociales, entre otros. Estos cambios tienen repercusión sobre la organización de la institución y están motivados por las necesidades básicas de la misma

Es así como surge el requerimiento de utilizar nuevas tecnologías, destinadas a mejorar la administración de nuestra institución. Para tal motivo se desarrollará un sistema capaz de llevar de una manera más eficiente el control administrativo del programa del servicio social. Las opciones tecnológicas que se pueden elegir para implementar este sistema son diversas, sin embargo, debido a que este debe ser accesible desde cualquier lugar y de una manera segura, se desarrollará para que sea accesible vía Web, esto debido a que la Web puede ofrecernos una gran cantidad de recursos que podemos emplear para conseguir los objetivos planteados.

Para establecer un sistema vía Web se requiere del siguiente software:

- Un servidor Web.
- Un manejador de base de datos (DBMS).
- Un lenguaje de programación, del lado del servidor y para ejecutarse del lado del cliente.
- Un navegador de Internet.

Un servidor Web es un programa que implementa el protocolo HTTP (hypertext transfer protocol). Este protocolo está diseñado para transferir lo que llamamos hipertextos, páginas Web o páginas HTML (hypertext markup language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonidos. Un servidor Web se encarga de mantenerse a la espera de peticiones HTTP llevada a cabo por un cliente HTTP que conocemos como navegador. El navegador realiza una petición al servidor y éste le responde con el contenido que el cliente solicita.

El sistema manejador de bases de datos (SGDB) es la porción más importante del software de un sistema de base de datos. Un SGDB es una colección de numerosas rutinas de software interrelacionadas, cada una de las cuales es responsable de alguna tarea específica

Las funciones principales de un DBMS son:

- Crear y organizar la base de datos.
- Establecer y mantener las trayectorias de acceso a la base de datos de tal forma que se pueda tener un acceso a los datos rápidamente.
- Manejar los datos de acuerdo a las peticiones de los usuarios.
- Registrar el uso de las bases de datos.
- Interacción con el manejador de archivos. Esto a través de las sentencias en DML, al comando del sistema de archivos. Así el Manejador de base de datos es el responsable del verdadero almacenamiento de los datos.
- Respaldo y recuperación. Consiste en contar con mecanismos implantados que permitan la recuperación fácilmente de los datos en caso de ocurrir fallas en el sistema de base de datos.
- Control de concurrencia. Consiste en controlar la interacción entre los usuarios concurrentes para no afectar la inconsistencia de los datos.
- Seguridad e integridad. Consiste en contar con mecanismos que permitan el control de la consistencia de los datos evitando que estos se vean perjudicados por cambios no autorizados o previstos.

Se requiere de un lenguaje de programación que nos permita crear páginas dinámicas que sean reconocidas, interpretadas y ejecutadas por el propio servidor. Las páginas dinámicas del servidor se suelen escribir en el mismo archivo HTML, mezclado con el código HTML. Cuando una página es solicitada por parte de un cliente, el servidor ejecuta los scripts y se genera una página resultado, que solamente contiene código HTML. Este resultado final es el que se envía al cliente y puede ser interpretado sin lugar a errores ni incompatibilidades, puesto que sólo contiene HTML. Para escribir páginas dinámicas de servidor existen varios lenguajes: Common Gateway Interface

(CGI) comúnmente escritos en Perl, Active Server Pages (ASP), Hipertext Preprocesor (PHP), y Java Server Pages (JSP).

Las ventajas de este tipo de programación son que el cliente no puede ver los scripts, ya que se ejecutan y transforman en HTML antes de enviarlos. Además son independientes del navegador del usuario, ya que el código que reciben es HTML fácilmente interpretable.

Otro lenguaje que se requiere es uno que se ejecute en el cliente (navegador de Internet), para ello se utilizan dos lenguajes de programación principalmente: Javascript y Visual Basic Script (VBScript). Las páginas del cliente son muy dependientes del sistema donde se están ejecutando y esa es su principal desventaja, ya que cada navegador tiene sus propias características, incluso cada versión, y lo que puede funcionar en un navegador puede no funcionar en otro. Como ventaja se puede decir que estas páginas descargan al servidor algunos trabajos, ofrecen respuestas inmediatas a las acciones del usuario y permiten la utilización de algunos recursos de la máquina local.

Para poder seleccionar un conjunto de productos para desarrollar este sistema debemos considerar algunos aspectos como: seguridad, costo, compatibilidad y rendimiento. Tomando como punto de referencia el costo, se pretende que el sistema no resulte muy costoso en cuanto a la adquisición de licencias por el uso de productos para su implementación, motivo por el cual se usara software libre (open source).

Para la base de datos y lenguaje de programación, se debe tener en cuenta la compatibilidad y rendimiento entre estas herramientas, además de que deben ser independientes de la plataforma donde se implementen.

A continuación se comparan algunas tecnologías con el fin de encontrar la mejor conjunción entre el servidor Web, la base de datos y los lenguajes de programación.

Como manejadores de base de datos, tenemos dos opciones muy populares: MySQL y PostgreSQL, a continuación se hará una breve comparación:

Ventajas de PostgreSQL ...

Las características positivas que posee este gestor son:

1. Posee una gran escalabilidad. Es capaz de ajustarse al número de computadoras y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta (algunos usuarios que han utilizado comenta que puede llegar a soportar el triple de carga de lo que soporta MySQL).
2. Implementa el uso de rollback's, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz, y ofreciendo soluciones en campos en las que MySQL no podría.
3. Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle.

Desventajas

Por contra, los mayores inconvenientes que se pueden encontrar a este gestor son:

1. Consume mas recursos que MYSQL.
2. Tiene un límite de 8K por fila, aunque se puede aumentar a 32K, con una disminución considerable del rendimiento.
3. Es de 2 a 3 veces más lento que MySQL.

Ventajas de MySQL ...

La mayoría de gente usa este gestor en Internet, veamos las razones:

1. Sin lugar a duda, lo mejor de MySQL es su velocidad a la hora de realizar las operaciones, lo que le hace uno de los gestores que ofrecen mayor rendimiento.
2. Su bajo consumo lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema.
3. Las utilidades de administración de este gestor son envidiables para muchos de los gestores comerciales existentes, debido a su gran facilidad de configuración e instalación.
4. Tiene una probabilidad muy reducida de corromper los datos, incluso en los casos en los que los errores no se produzcan en el propio gestor, sino en el sistema en el que está.
5. El conjunto de aplicaciones Apache-PHP-MySQL es uno de los más utilizados en Internet en servicios de foro de buscadores de aplicaciones.

Desventajas.

Debido a esta mayor aceptación en Internet, gran parte de los inconvenientes se exponen a continuación, han sido extraídos de comparativas con otras bases de datos:

1. Carece de soporte para transacciones, rollback's y subconsultas.
2. El hecho de que no maneje la integridad referencial, hace de este gestor una solución pobre para muchos campos de aplicación, sobre todo para aquellos programadores que provienen de otros gestores que sí que poseen esta característica.
3. No es viable para su uso con grandes bases de datos, a las que se acceda continuamente, ya que no implementa una buena escalabilidad.

Debido a las ventajas que nos ofrece PostgreSQL en cuanto a integridad referencial uso de funciones, es el elegido para ser nuestro SGDB y de esta manera poder hacer nuestro sistema confiable y efectivo.

En cuanto a los lenguajes de programación del lado del servidor, las 3 opciones mas viables son : PHP, ASP, JSP. A continuación se comparan estos lenguajes

	LENGUAJES		
Característica	**PHP**	**ASP**	**JSP**
Plataforma	Independiente de Plataforma y servidor (Unix,Windows, Linux)	Se trata de un sistema propietario que es usado nativamente sólo por Microsoft Internet Information Server (IIS). Esto limita su disponibilidad a servidores basados en Win32.	Independiente de Plataforma y servidor. (Unix,Windows, Linux)
Seguridad	Instalado sobre servidores Unix o Linux, es más veloz y seguro. Además, PHP permite configurar el servidor de modo que se permita o rechacen diferentes usos, lo que puede hacer al lenguaje seguro dependiendo de las necesidades	Debido que solo funciona bajo Windows, es mas susceptible a sufrir caídas, e infecciones de virus	Las páginas JSP son compiladas en Servlets por lo que actúan como una puerta a todos los servicios Java de Servidor y librerías Java para aplicaciones http. Ayudando a proteger el sistema contra las caídas
Rendimiento	En el caso de estar montado sobre un servidor Unix o Linux, es más rápido dado que se ejecuta en un único espacio de memoria	Son demasiadas las comunicaciones entre componentes COM que se realizan entre todas las tecnologías implicadas en una página ASP. Consumiendo así mas recursos	Los componentes JSP son reusables en distintas plataformas (UNIX, Windows), también ayuda en el manejo de la memoria protegiendo contra fallos
Soporte	Recibe contribuciones de diversos desarrolladores	ASP es específica de Microsoft que desarrolla sus procesos internamente.	El API JSP se beneficia de la extendida comunidad JAVA existente.

Figura 3.4.5.1 Tabla comparativa de Lenguajes de programación Web

Una vez comparado los manejadores de base de datos, observamos que POSTGRESQL resulta la mejor opción ya que no consume demasiados recursos, es robusto, seguro y tiene un mejor rendimiento.

En cuanto a los lenguajes de programación observamos que PHP y JSP son buenas opciones ya que ASP, solo trabaja en Windows, motivo por el cual queda descartado, por otro lado considerando que POSTGRESQL tiene una buena interacción con PHP y este es un lenguaje fácil de aprender, además de desarrollar aplicaciones de una manera rápida, PHP resulta la mejor opción.

PHP puede hacer cualquier cosa que se pueda hacer con un script CGI, como procesar la información de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies. PHP soporta la mayoría de servidores web de hoy en día, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape e iPlanet, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd y muchos otros. PHP tiene módulos disponibles para la mayoría de los servidores, para aquellos otros que soporten el estándar CGI, PHP puede usarse como procesador CGI.

Con PHP no se encuentra limitado a resultados en HTML. Entre las habilidades de PHP se incluyen: creación de imágenes, archivos PDF y películas Flash (usando libswf y Ming) sobre la marcha. También puede presentar otros resultados, como XHTM y archivos XML. PHP puede autogenerar estos archivos y almacenarlos en el sistema de archivos en vez de presentarlos en la pantalla

Quizás la característica más potente y destacable de PHP es su soporte para una gran cantidad de bases de datos. Escribir un interfaz vía Web para una base de datos es una tarea simple con PHP. PHP también cuenta con soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros. También se pueden crear sockets puros. PHP soporta WDDX para el intercambio de datos entre lenguajes de programación en web. Y hablando de interconexión, PHP puede utilizar objetos Java de forma transparente como objetos PHP Y la extensión de CORBA puede ser utilizada para acceder a objetos remotos.

En cuanto al lenguaje que se ejecuta del lado del cliente debemos considerar cual es más compatible con los navegadores de Internet mas utilizados. Debido a que Visual Basic Script (VBScript) solo es compatible con Internet Explorer queda descartado ya que reduce las opciones en cuanto al funcionamiento de la aplicación, pues forzaría a los usuarios a utilizar un navegador en especifico, mientras que Javascript es compatible con mas navegadores, como son Internet Explorer, Mozilla Firefox, Netscape, Opera, entre otros, es por ello que se utilizara Javascript con lenguaje para ejecutarse del lado del cliente.

Y por ultimo, para elegir el servidor web, lo haremos en función de nuestro lenguaje de programación elegido, APACHE es considerado uno de los mejores servidores web en lo que se refiere a software libre, además de tener una gran compatibilidad con PHP. Es flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos.

Entres sus características destacan:

- Mutiplataforma.
- Es un servidor Web conforme al protocolo http/1.1.
- Modular: Puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la API de programación de modulos, para el desarrollo de módulos específicos.
- Basado en hebras o hilos en la versión 2.0.

- Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.
- Se desarrolla en forma abierta.
- Extensible: Se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor.

CAPITULO IV. DESARROLLO E IMPLANTACIÓN DEL SISTEMA

Una vez que hemos definido y analizado los requerimientos del sistema utilizaremos la metodología de YOURDON que determina los elementos internos y externos que intervienen en el sistema, es decir los elementos que el sistema recibe del exterior y la respuesta que se genere a través del mismo.

La metodología YOURDON incluye una serie de eventos, los cuales son una descripción de los diferentes sucesos o estímulos que ocurren en el mundo exterior y a los cuales debe de dar una respuesta el sistema diseñado.

Se puede ejemplificar esta metodología analizando un evento dentro del departamento de servicio social el cual tiene una serie de sucesos o estímulos y que reciben del exterior una serie de datos que finalmente se traduce en información.

4.1 DIAGRAMAS DE CONTEXTO

Un sistema puede ser representado gráficamente de diferentes formas, los diversos modelos gráficos muestran las fronteras del sistema y la información usada dentro del sistema

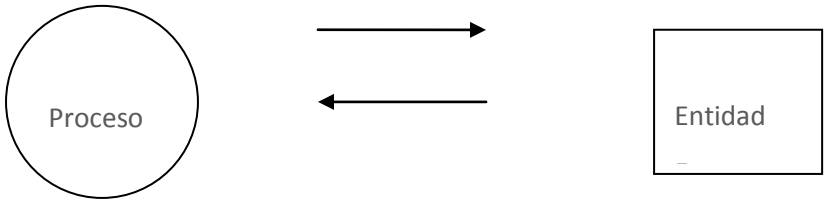

Figura 4.1.1 Componentes básicos de un diagrama de contexto

Estos componentes se pueden definir como:

- Proceso.- Un proceso significa que se realizan algunas acciones o grupo de acciones
- Entidad externa.- Una entidad externa puede ser una persona, grupo o departamento o cualquier sistema que recibe u origina información o datos pero que no es parte del sistema.
- Flechas.- Un flujo de datos muestra que es una información desde o hacia un proceso y conecta entidades.

El diagrama de contexto nos permitirá identificar a las personas o entidades que se comunican con el sistema y recibirá datos del medio ambiente que serán sus entradas necesarias para producir datos que el mundo exterior espera e identificará como salida.

El sistema debe reconocer cada flujo de entrada para identificar que ha ocurrido un evento y cada evento debe generar salidas inmediatas como respuesta o bien

almacenar datos, que posteriormente se convertirán en salida o un indicador de un cambio de estado dentro del sistema

En la figura 4.1.2 el diagrama de contexto muestra cómo se conectará nuestro sistema con cada una de las áreas que lo rodea.

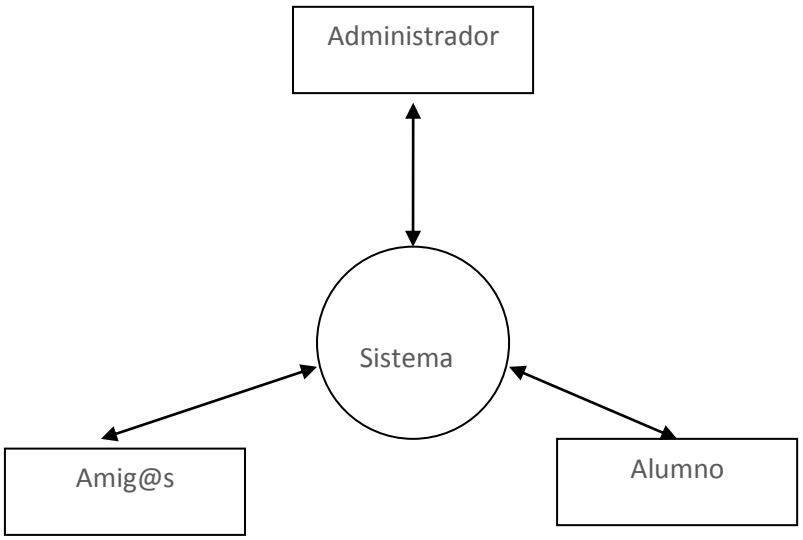

Figura 4.1.2

En la figura anterior se muestra un panorama general de las diferentes entidades que intervienen en el sistema, así como el flujo de información entre ellos, lo cual significa que el sistema de control escolar recibirá datos de las entidades, los procesará y emitirá resultados que involucren a otras entidades o a la misma que generó la información.

Se listan algunos de los eventos considerados para el desarrollo del sistema.

- El Alumno solicita trámite de inscripción.
- El Amig@ solicita trámite de inscripción.
- El administrador del sistema tiene control sobre la información capturada en el sistema.

La siguiente figura (4.1.3) muestra el primer nivel del diagrama de contexto. A través de este nivel podemos visualizar los diferentes módulos que conforman el sistema y se aprecia cada modulo atiende una situación existente y tiene una comunicación con entidades bien determinadas.

Como se puede observar, cada modulo se ejecutan diferentes procesos y procedimientos los cuales se convertirán en subfunciones que darán origen a un diagrama de nivel inferior.

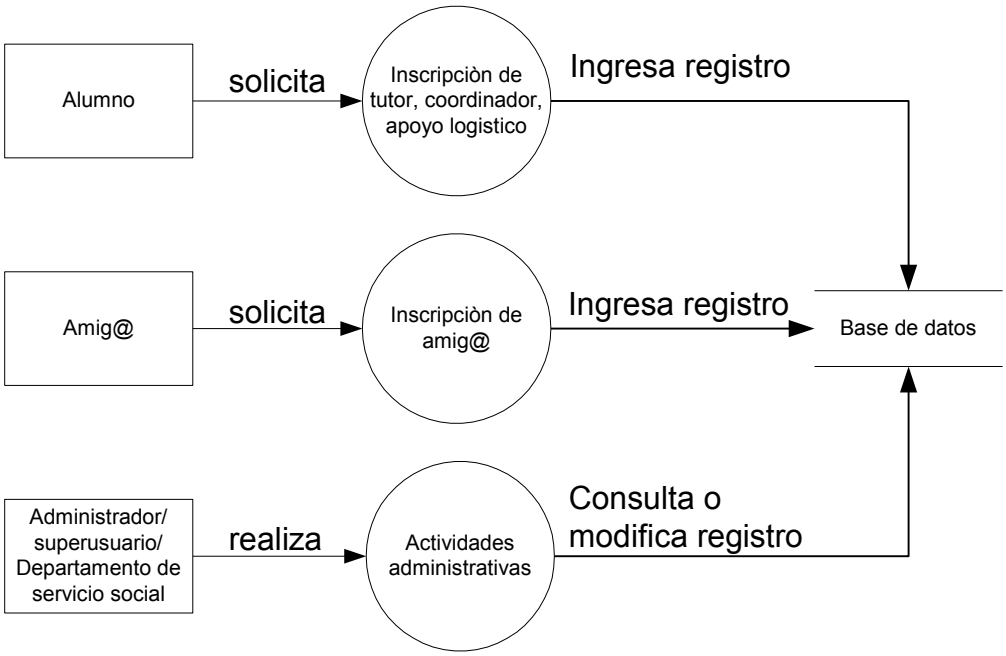

Figura 4.1.3

En la figura 4.1.4 se muestra el proceso para que un alumno pueda solicitar su inscripción como tutor, coordinador o apoyo logístico.

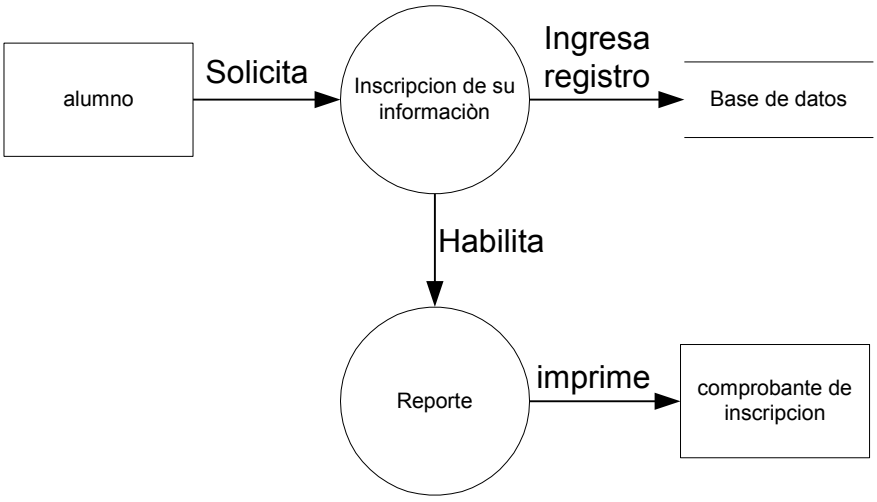

Figura 4.1.4

Para que un alumno se pueda inscribir en el programa de servicio social, debe de realizar los siguientes pasos:

- Ingresar al sistema vía web
- Registrar sus datos.
- Envía su registro a la base de datos.
- Genera el reporte correspondiente.

En la siguiente figura (4.1.5) se muestra el proceso para que un amig@ pueda solicitar su inscripción al programa de servicio social.

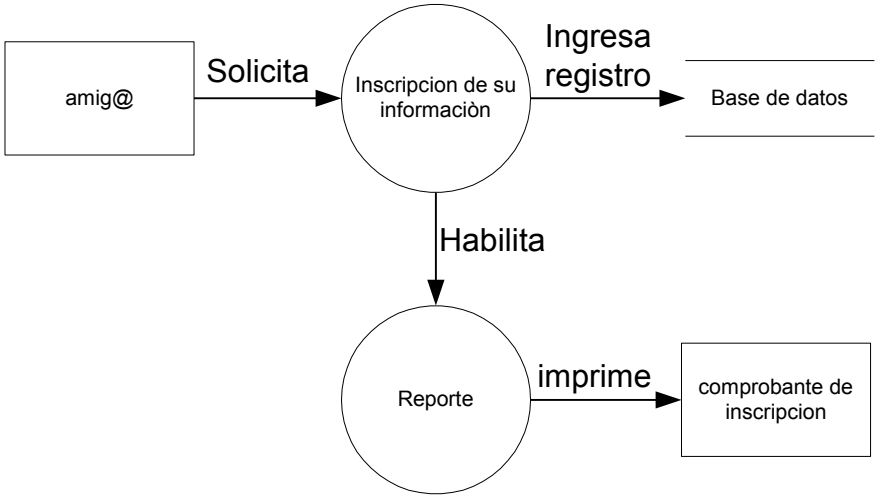

Figura 4.1.5

Para que un amig@ se pueda inscribir en el programa de servicio social, debe de realizar los siguientes pasos:

- Ingresar al sistema vía web
- Registrar sus datos.
- Envía su registro a la base de datos.
- Genera el reporte correspondiente.

Cuando los tutores, coordinadores, apoyo logístico, amig@s ya fueron capturados, se procede al registro de tutorías como se muestra en la figura 4.1.6

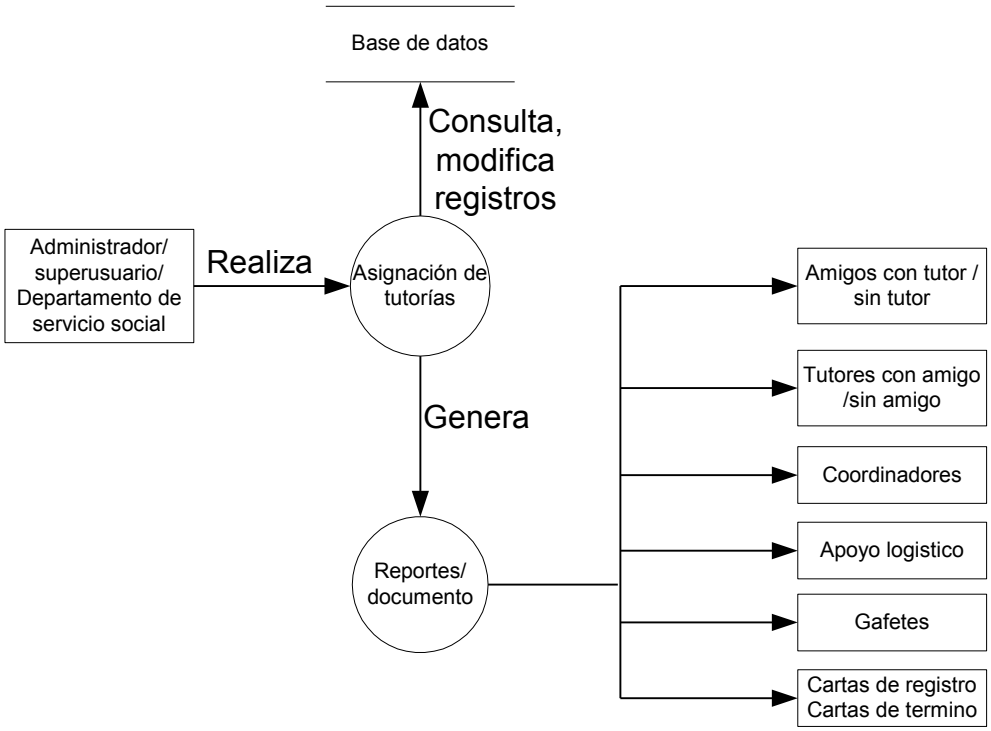

Figura 4.1.6

4.2 DIAGRAMA DE FLUJO.

Diagrama General

Partiendo de la pantalla principal (inicio de sesión) en las siguientes figuras se representa gráficamente de forma general los procesos que lleva acabo el sistema para el registro dependiendo del tipo de usuario:

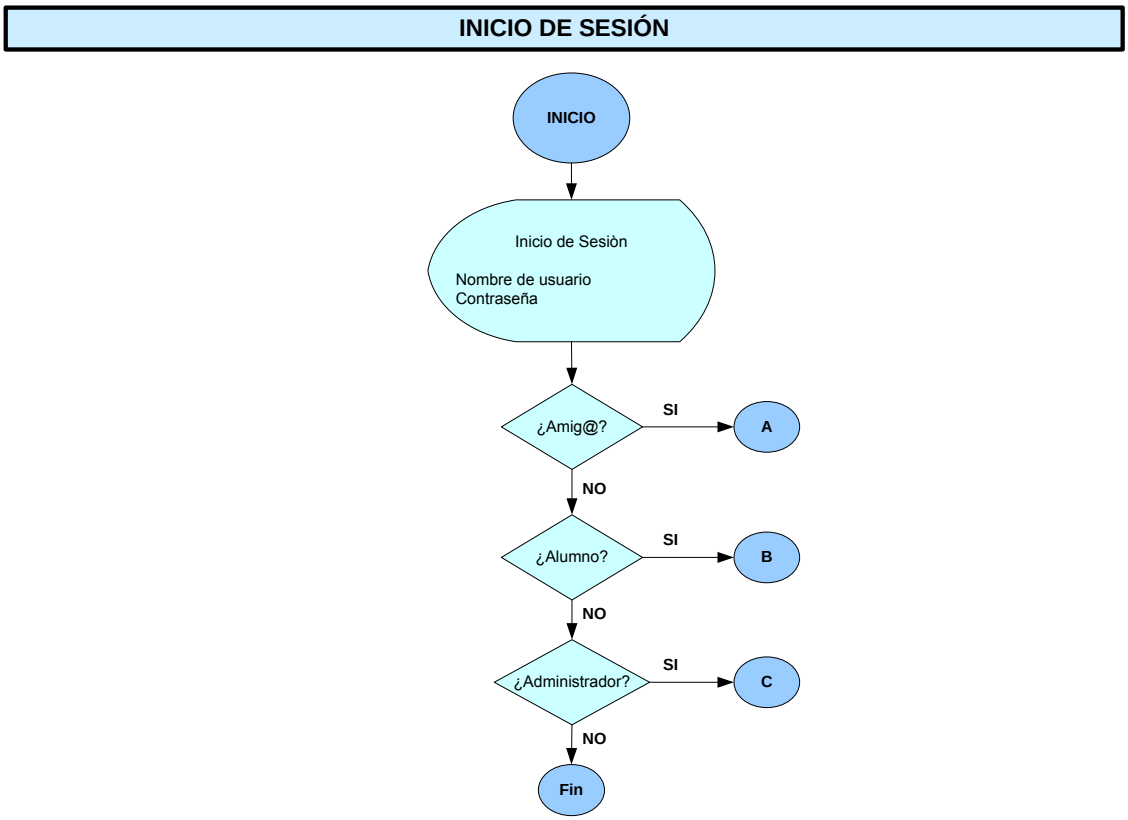

Figura 4.2.1 Inicio de sesión

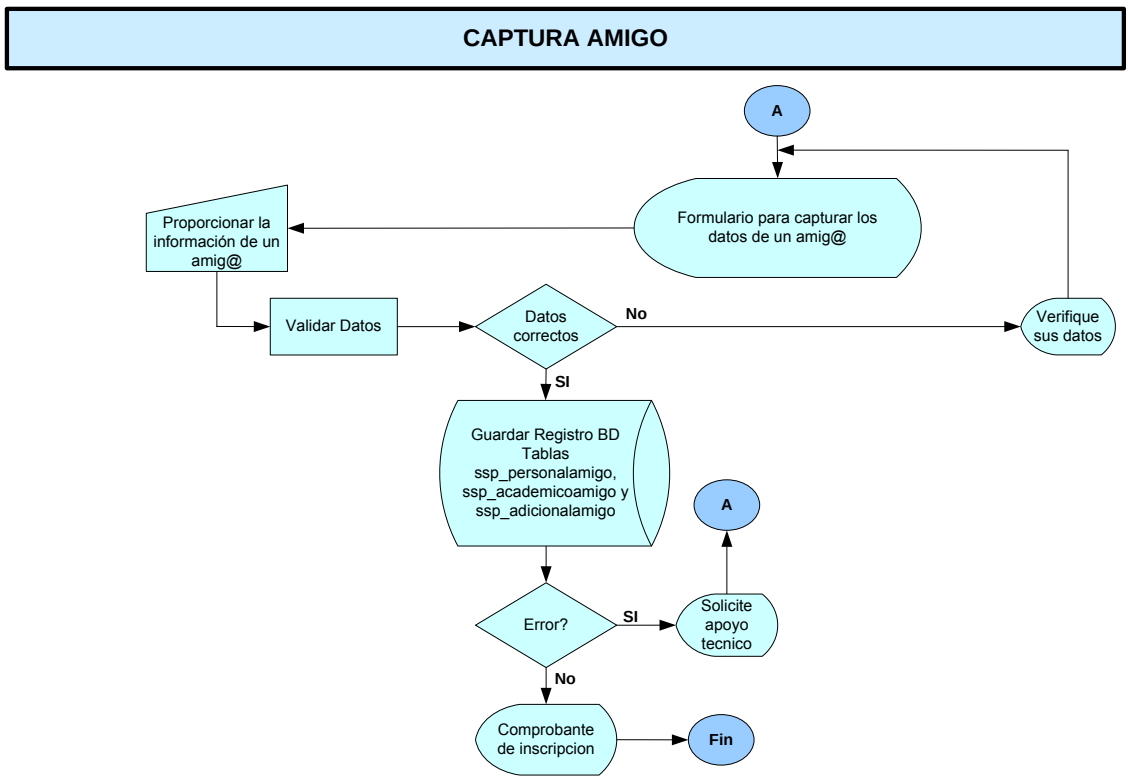

Figura 4.2.2 Captura amig@

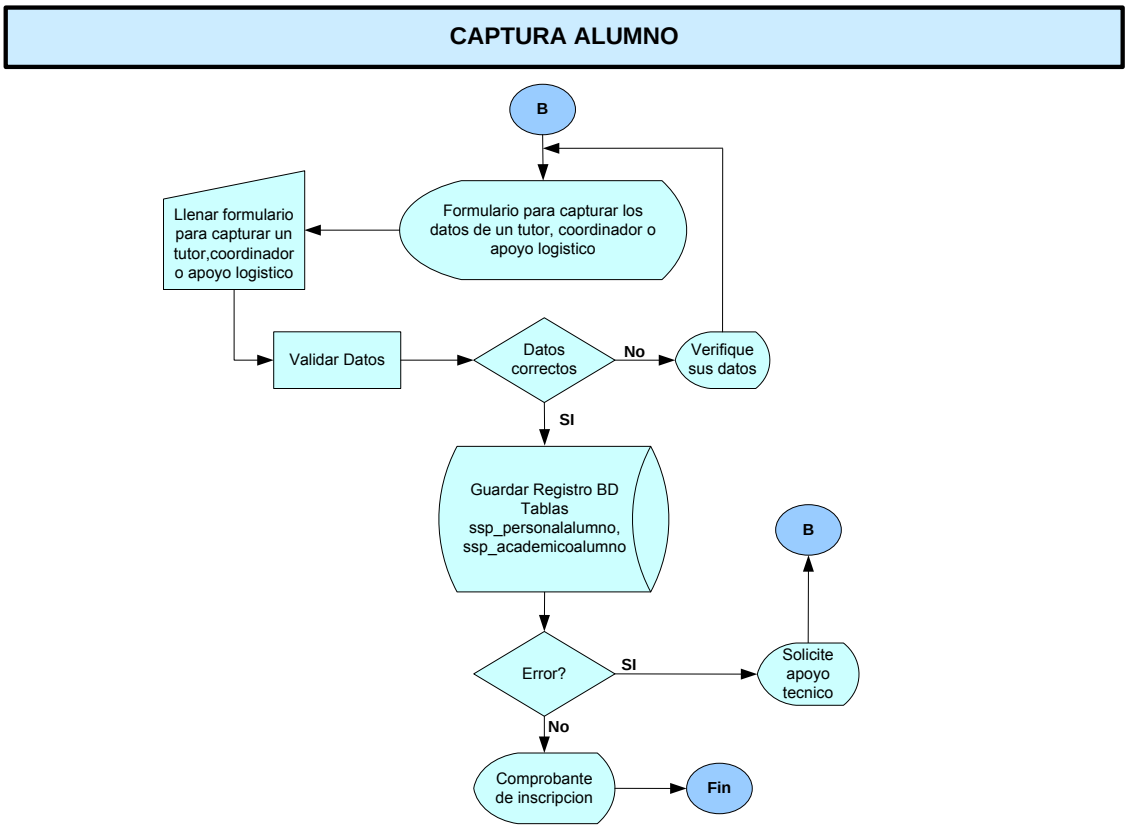

Figura 4.2.3 Captura alumno

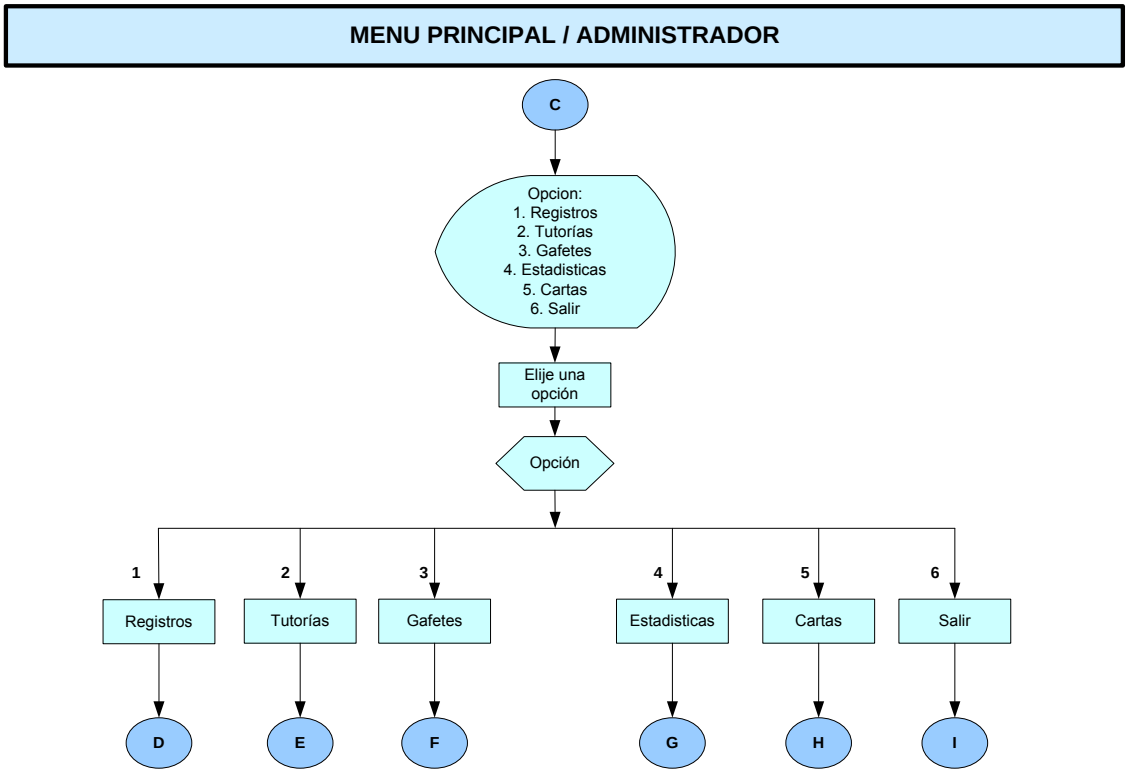

Figura 4.2.4 Menú Principal

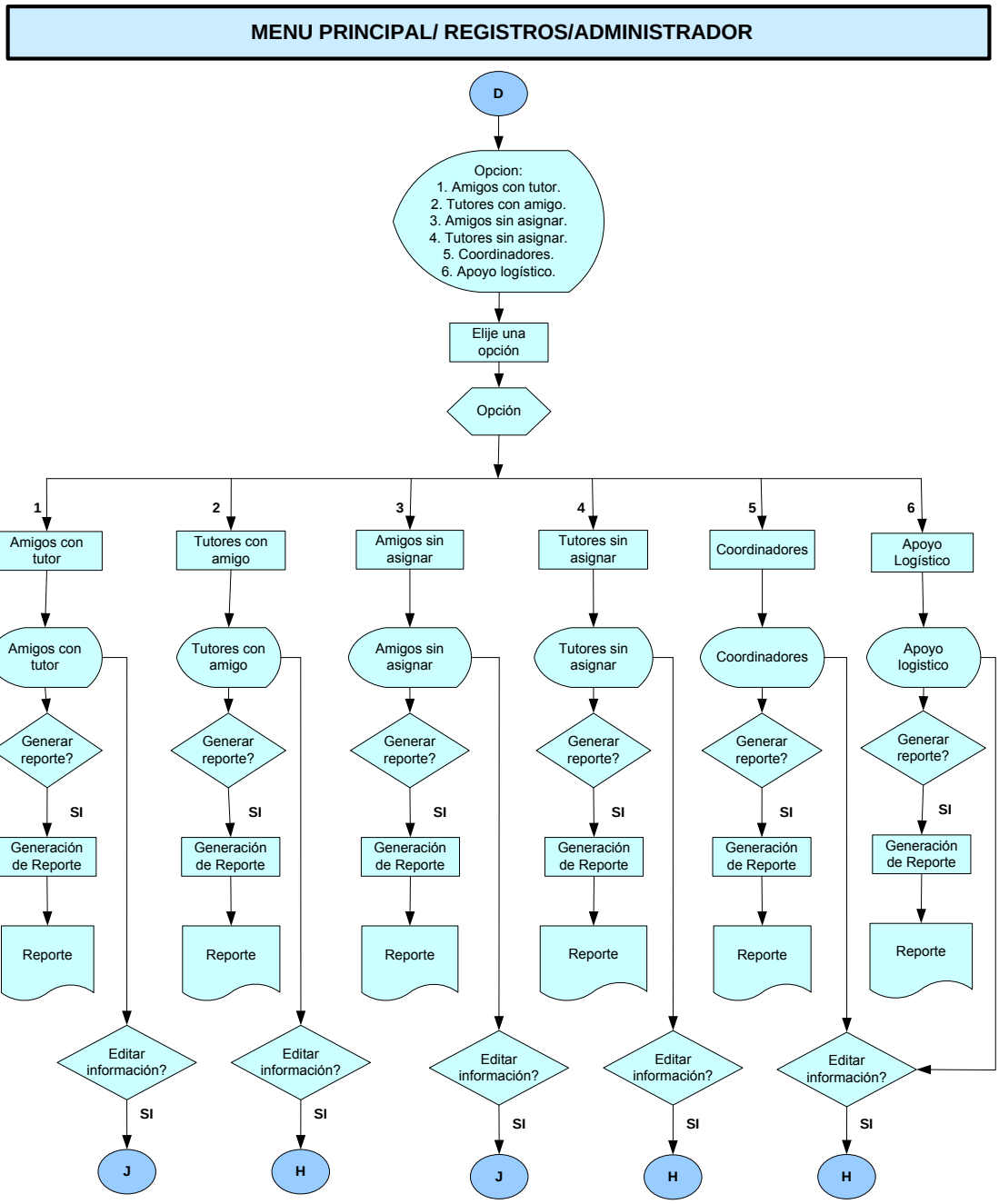

Figura 4.2.5 Registros.

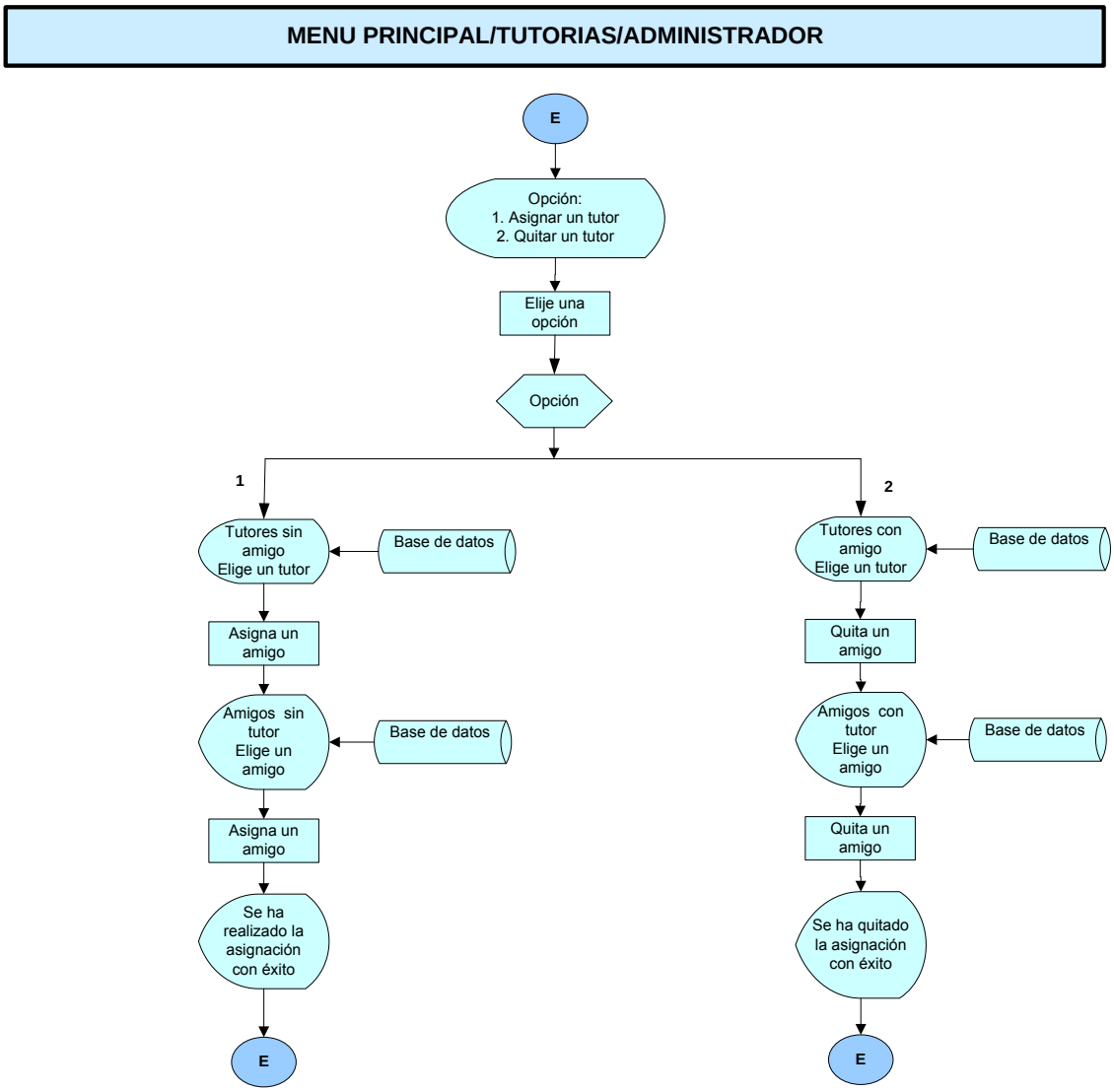

Figura 4.2.6 Tutorías.

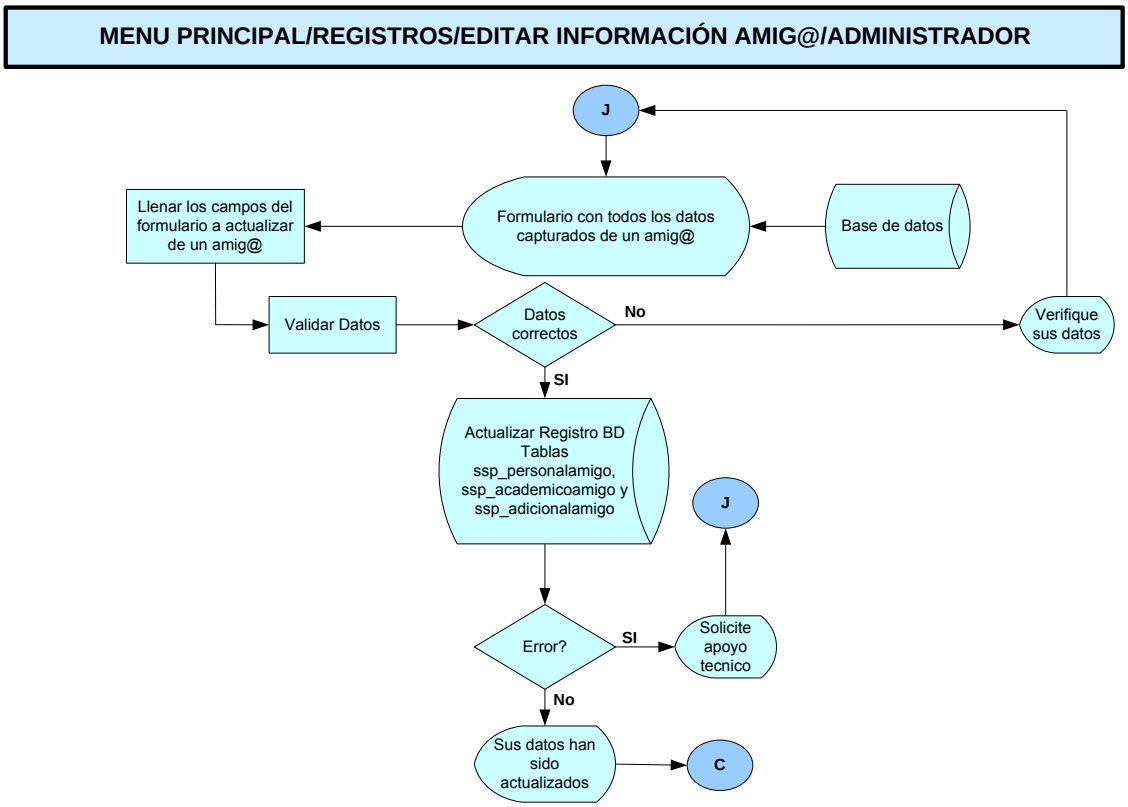

Figura 4.2.7 Editar Información de un amig@

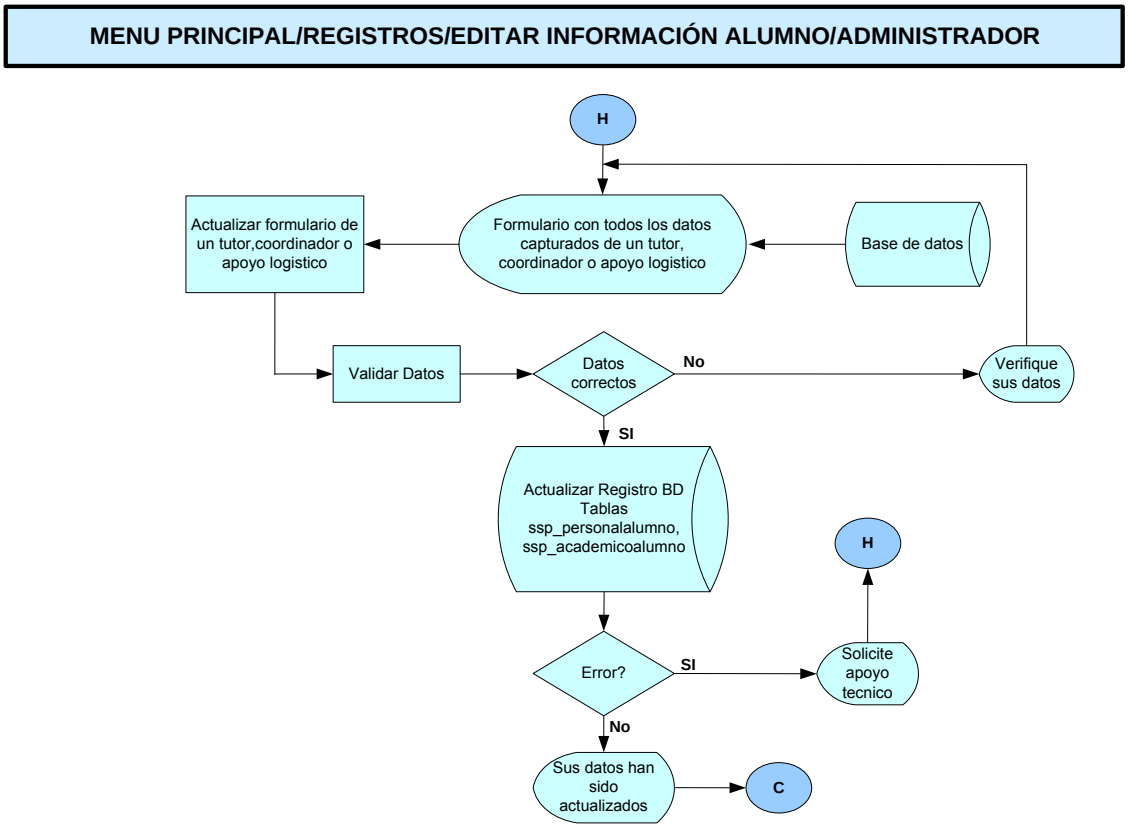

Figura 4.2.8 Editar Información de un alumno

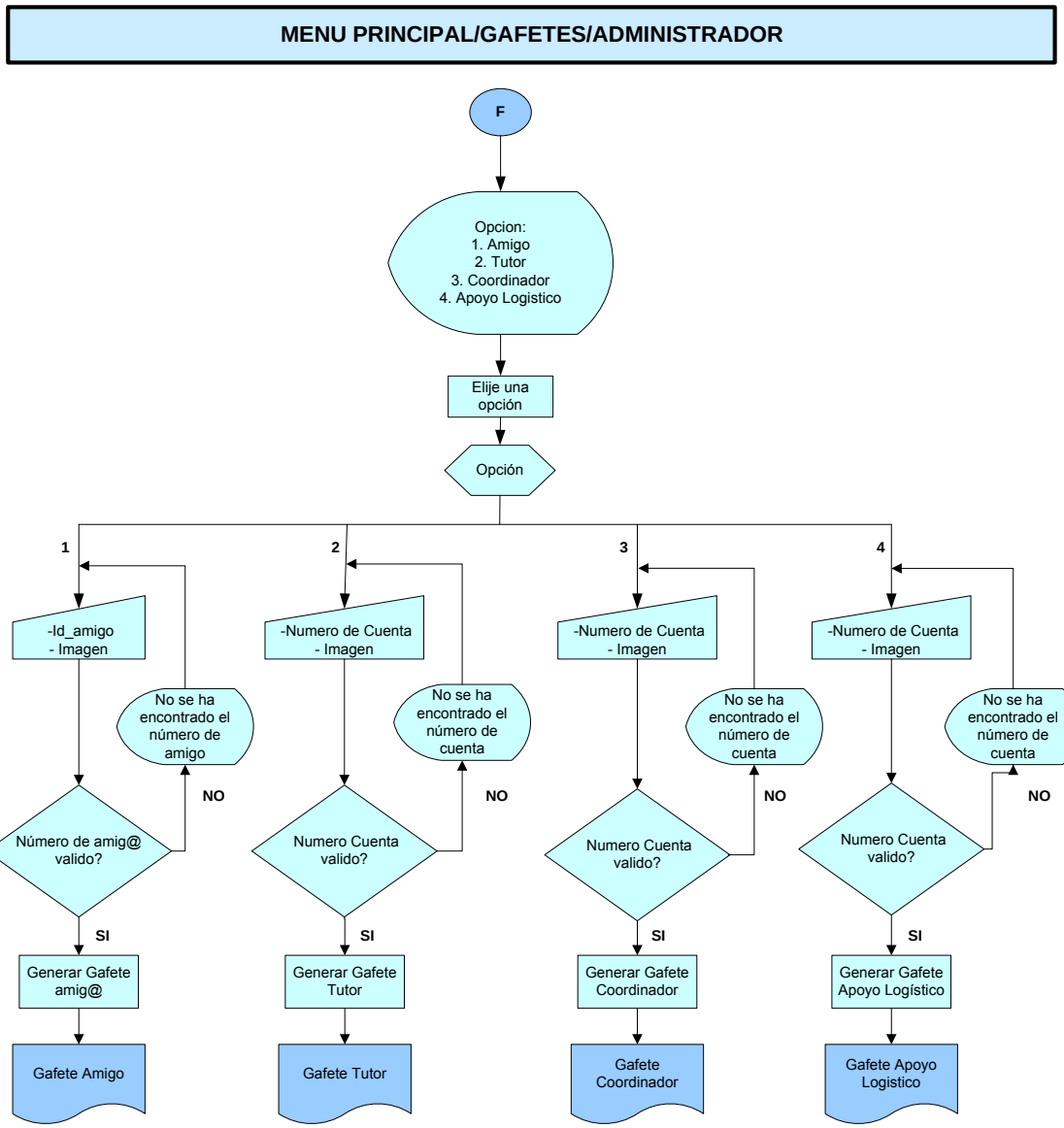

Figura 4.2.9 Generación de Gafetes.

MENU PRINCIPAL/ESTADISTICA/ADMINISTRADOR

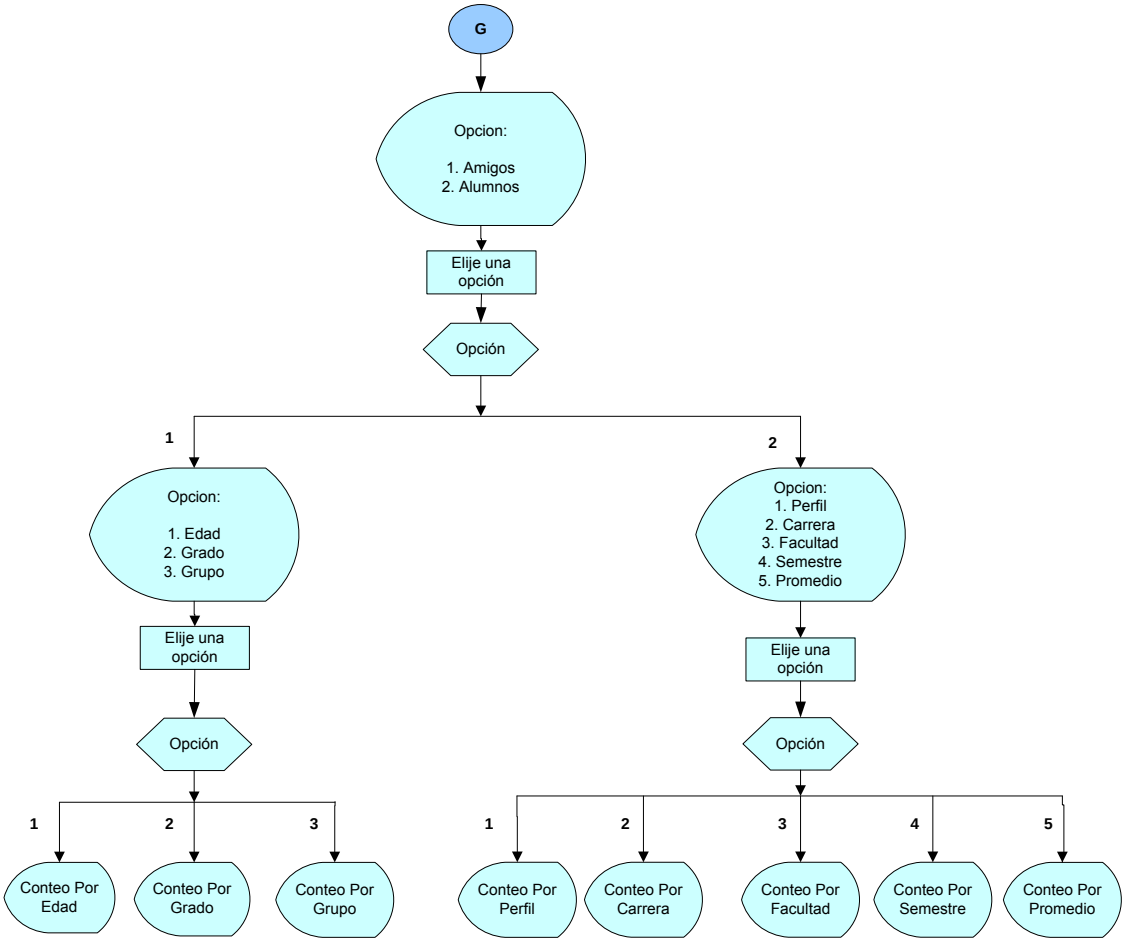

Figura 4.2.10 Estadísticas.

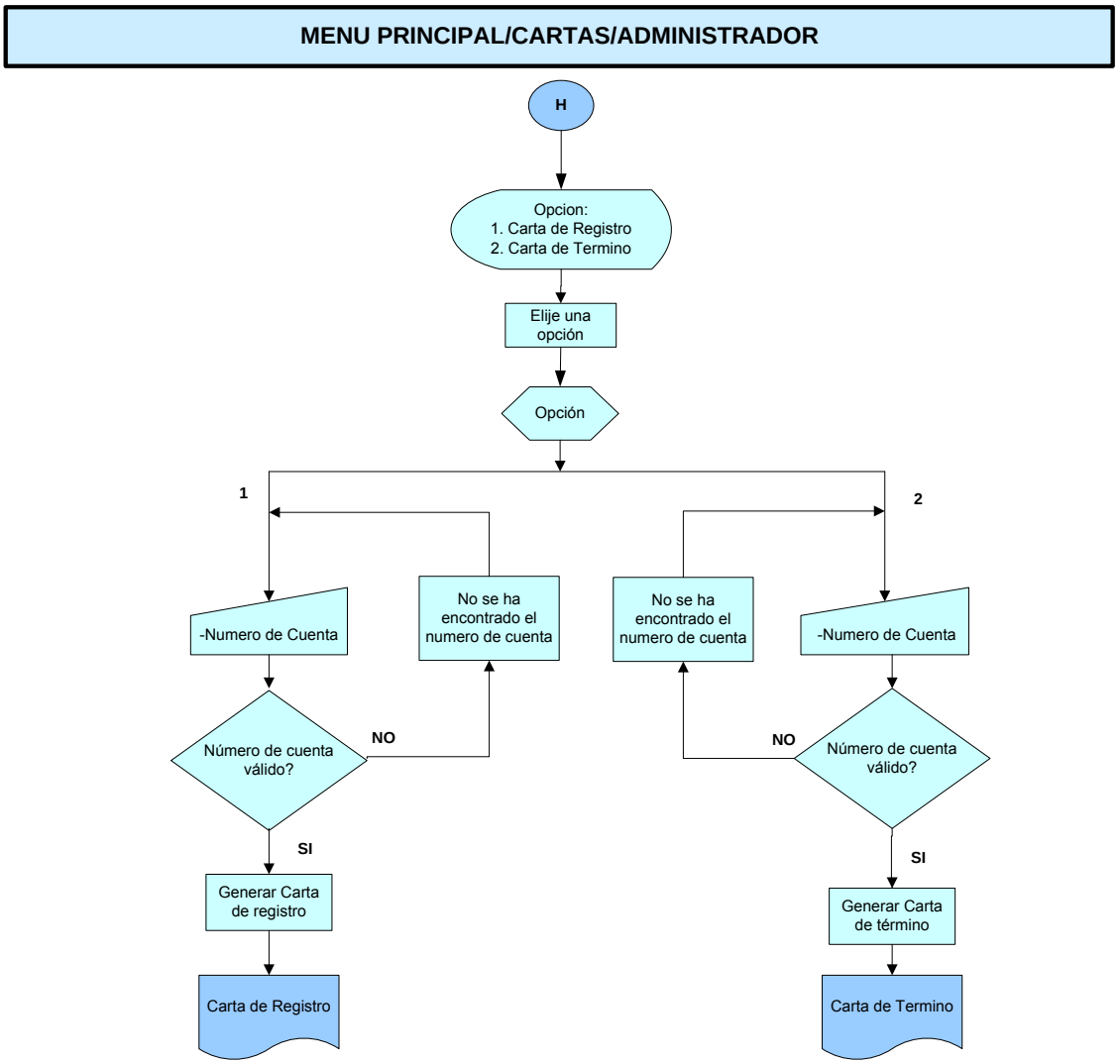

Figura 4.2.11 Generación de cartas de registro y término

4.3 DICCIONARIO DE DATOS

El diccionario de datos es una forma de documentación para el diseñador de base de datos. Su utilidad básica es describir las siguientes funciones:

- Explicar el propósito de la base de datos
- Proveer una detallada descripción de cada tabla dentro de la base de datos.
- Documentar la estructura interna de cada tabla.
- Describir reglas de cómo pueden ser tratados los valores no nulos, valores únicos.

En la tabla 4.3.1 se muestra las diferentes tablas que se utilizan en la base de datos del sistema y un breve comentario acerca de ellas:

Tablas	Comentario
ss_areas	Tabla que contiene las diferentes áreas académicas
ss_carreras	Tabla que contiene las diferentes carreras
ss_facultades	Tabla que contiene los diferentes tipos de escuelas o facultades
ss_usuario	Tabla que contiene los tipos de usuario
ssp_academicoalumno	Tabla que contiene la información académica de los alumnos
ssp_academicoamigo	Tabla que contiene la información académica de los amigos
ssp_adicionalamigo	Tabla que contiene los datos adicionales del amigo, como gustos , preferencias, relaciones familiares
ssp_personalalumno	Tabla que contiene la información personal del alumno
ssp_personalamigo	Tabla que contiene la información personal del amigo
ssp_tutorias	Tabla que contiene la información de las asignaciones amig@-tutor

Tabla 4.3.1 Tablas de la base de datos

A continuación se describen cada una de ellas.

Administración de Áreas Académicas.

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
idarea	integer	Si	Si	0	Identificador para cada tipo de área
nombre	character varying(70)	Si	No	0	Nombre del área académica

Tabla 4.3.2 ss_areas

Administración de carreras

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
idcarrera	integer	Si	Si	0	Identificador para cada carrera
nombre	character varying(200)	Si	No	0	Nombre de cada carrera
idarea	integer	Si	No	0	Identificador del área académica

Tabla 4.3.3 ss_carreras

Administración de Escuelas o Facultades

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
idfac	integer	Si	Si	0	Identificador para cada escuela o facultad
nombre	character varying(80)	Si	No	0	Nombre de la escuela o facultad

Tabla 4.3.4 ss_facultades

Administración de los usuarios del sistema

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
id_usuario	integer	Si	Si	serie	identificador único para cada usuario
nombre_usuario	character varying(100)	Si	No	0	nombre del usuario
nombre_completo	character varying(100)	Si	No	0	Nombre completo de usuario
active	boolean	Si	No	false	Activación de los usuarios
contrasenia	character varying(100)	Si	No	0	Contraseña codificada

Tabla 4.3.5 ss_usuario

Información personal del alumno

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
nombre	character varying(80)	Si	No	0	Nombre del alumno
ap_paterno	character varying(80)	Si	No	0	Apellido Paterno
ap_materno	character varying(80)	Si	No	0	Apellido materno
calle	character varying(80)	Si	No	0	Calle del domicilio del alumno
no_int	character varying(80)	No	No		Numero interior del domicilio del alumno
no_ext	character varying(80)	Si	No	0	Numero exterior del domicilio del alumno
colonia	character varying(80)	Si	No	0	Colonia del domicilio del alumno
cp	character varying(30)	Si	No	0	Codigo postal del domicilio del alumno
delegacion	character varying(80)	Si	No	0	Delegacion del domicilio del alumno
entidad	character varying(80)	Si	No	0	Entidad federativa del domicilio del alumno
tel_casa	character varying(40)	Si	No	0	Numero telefonico del alumno
rfc	character varying(40)	Si	No	0	Registro Federal de contribuyentes del alumno
curp	character varying(40)	No	No		curp del alumno
sexo	character varying(1)	Si	No	0	Identificador del sexo del alumno
email	character varying(60)	Si	No	0	Correo electronico del alumno
fecha_nacimiento	character varying(20)	Si	No	0	Fecha de nacimiento del alumno
edad	character varying(2)	Si	No	0	Edad del alumno
numcuenta	character varying(9)	Si	Si	0	Numero de cuenta de la UNAM del alumno
tel_cel	character varying(40)	No	No		Telefono Celular del alumno
perfil	smallint	Si	No	0	Identificador del perfil del alumno
num_asignacion	integer	No	No	0	Identificador del asignacion

Tabla 4.3.6 ssp_personalalumno

Información académica del alumno.

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
numcuenta	character varying(9)	Si	Si	0	Numero de cuenta del alumno
facultad	character varying(30)	Si	No	0	identificador de la escuela o facultad del alumno
carrera	character varying(30)	Si	No	0	Identificador de la carrera del alumno
creditos	character varying(20)	Si	No	0	Identificador del porcentaje de creditos del alumno
promedio	character varying(20)	Si	No	0	Identificador del promedio del alumno
semestre	character varying(20)	Si	No	0	Identificador del semestre del alumno
area	character varying(10)	Si	No	0	Identificador del area academica
beca	character varying(40)	Si	No	0	Identificador de alumno becado

Tabla 4.3.7 ssp_academicoalumno

Información Personal del amig@

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
nombre	character varying(80)	SI	No	0	Nombre del amig@
ap_paterno	character varying(80)	SI	No	0	Apeliido paterno del amig@
ap_materno	character varying(80)	SI	No	0	Apellido materno del amig@
calle	character varying(80)	SI	No	0	calle de la direccion del amig@
no_int	character varying(80)	No	No	0	numero interior de la direccion del amig@
no_ext	character varying(80)	SI	No	0	Numero exterior de la direccion del amig@
colonia	character varying(80)	SI	No	0	Colonia de la direccion del amig@
cp	character varying(30)	SI	No	0	Codigo postal de la direccion del amig@
delegacion	character varying(80)	SI	No	0	Delegacion de la direccion del amig@
entidad	character varying(80)	SI	No	0	Entidad federativa de la direccion del amig@
tel_casa	character varying(40)	SI	No	0	Telefono de casa del amig@
sexo	character varying(1)	SI	No	0	Sexo del amig@
fecha_nacimiento	character varying(20)	SI	No	0	Fecha de nacimiento del amig@
edad	character varying(2)	SI	No	0	Edad del amig@
idamigo	integer	Si	Si	serie	Identificador del amig@
num_asignacion	integer	No	No	serie	Identificador de asignación

Tabla 4.3.8 ssp_personalamig@

Información Académica del amig@

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
idamigo	integer	Si	Si		Identificador de amigo
escuela	character varying(80)	Si	No		Escuela del amig@
grado	character varying(80)	Si	No		Grado escolar del amig@
grupo	character varying(80)	Si	No		Grupo escolar del amig@
promedio	character varying(80)	Si	No		Promedio escolar del amig@
nom_prof	character varying(80)	Si	No		Nombre del profesor encargado del amig@
mat_pref	character varying(80)	Si	No		Materia preferida del amig@
mat_pref_razon	character varying(80)	Si	No		Razon de la materia preferida
mat_nopref	character varying(80)	Si	No		Materia no preferida del amigo
mat_nopref_razon	character varying(80)	Si	No		Razon de la materia no preferida

Tabla 4.3.9 ssp_academicoamig@

Información Adicional del amig@

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
idamigo	integer	Si	Si		Identificador del amig@
nombrepapa	character varying(80)	Si	No		Nombre completo del padre del amig@
nombremama	character varying(80)	Si	No		Nombre completo de la madre del amig@
nombre_ref	character varying(80)	Si	No		Nombre de una persona conocida del amig@
tel_ref	character varying(80)	Si	No		Telefono de referencia del amig@
parentesco	character varying(80)	Si	No		Parentesco con la persona de referencia del amig@
rel_fam	character varying(80)	Si	No		Identificador de las relaciones familiares del amig@
fam1	character varying(80)	Si	No		Identificador del familiar con una buena relación con el amig@
fam2	character varying(80)	Si	No		Identificador de la persona con una mala relación con el amig@
nomfam1	character varying(80)	Si	No		Nombre de la persona con una buena relación con el amig@ en caso de no estar elegible
nomfam2	character varying(80)	Si	No		Nombre de la persona con una mala relación con el amig@ en caso de no estar elegible
profesion	character varying(80)	Si	No		Nombre de la aspiración profesional del amig@
tiempo_libre	character varying(80)	Si	No		Actividad (es) en el tiempo libre del amig@
aspiracion	character varying(80)	Si	No		Respuesta a la pregunta de aspiraciones
nomaspiracion	character varying(80)	No	No		Respuesta de aspiraciones en caso de ser positiva la de aspiracion

Tabla 4.3.10 ssp_adicionalamig@

Nombre	Tipo de Dato	No Nulo	Clave Primaria	Defecto	Comentario
numcuenta	character varying(9)	Si	Si	0	identificador del tutor
Id_amigo	Integer(4)	Si	si	0	Identificador del amig@
num_asginacion	Integar(4)	Si	No	serie	Numero de asignacion

Tabla 4.3.11 ssp_tutorias

4.4 Diagrama entidad Relación

El sistema utiliza una base de datos con una estructura definida, ya que cada entidad puede ser distinguida una de otra según sus propiedades, todas las entidades cuentan con atributos que describen las propiedades de cada una de éstas.

Dentro de la estructura de la base de datos existen relaciones, las cuales describen las acciones que tiene una entidad sobre otra. Un ejemplo de las relaciones de la base de datos es:

- Un alumno pertenece o perteneció a una escuela
- Un alumno estudia o estudio una carrera

Pare establecer una relación entre dos entidades se cuenta con una llave primaria en una de ellas, que es un campo dentro de la tabla correspondiente que servirá como llave de acceso a la tupla, por lo que no deberá ser nulo y que existe en la otra entidad como llave foránea, estableciendo de está forma, la asociación entre ellas, como se muestra en la siguiente figura (4.4.1):

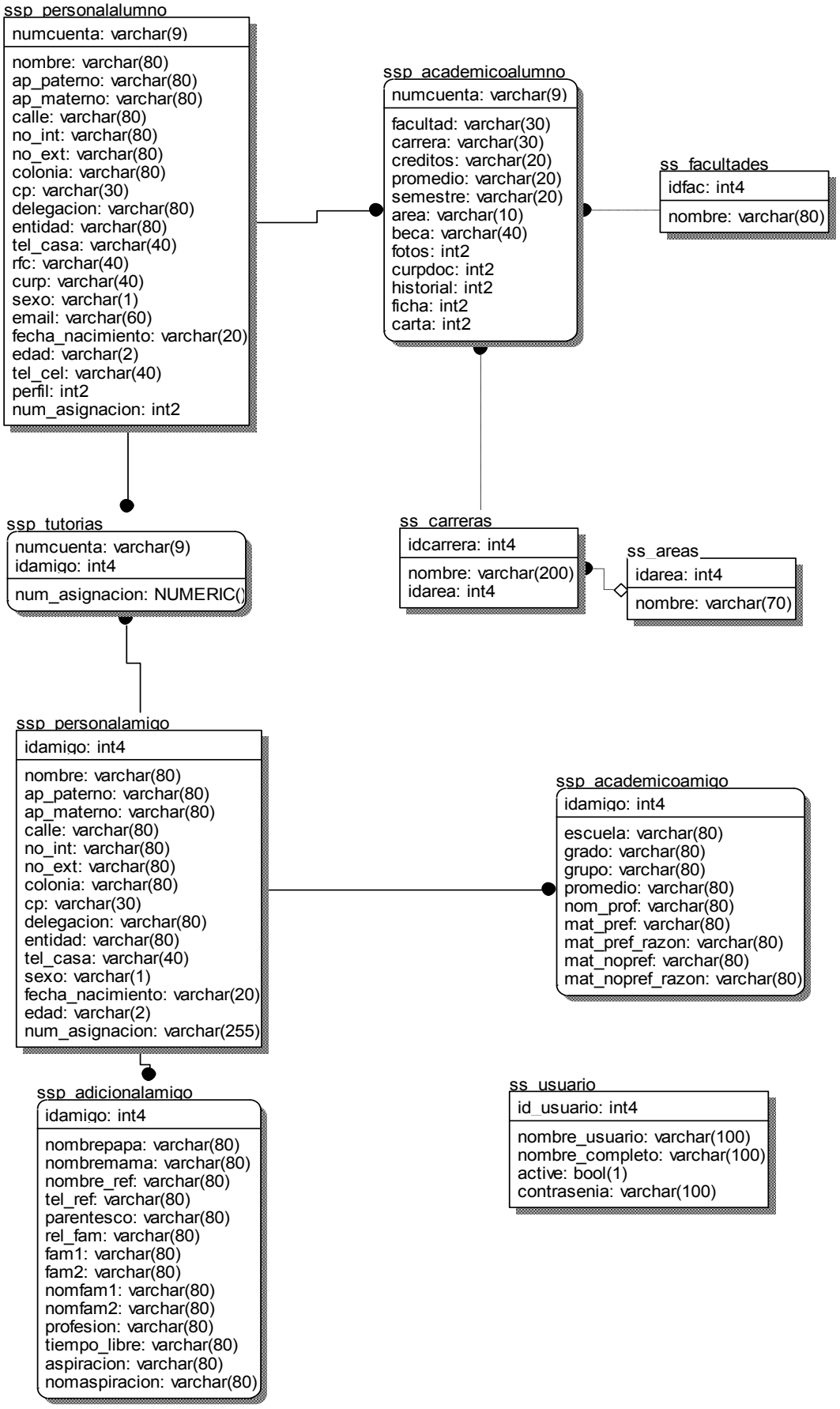

Figura 4.4.1 Diagrama entidad relación

4.5 Diseño y Generación de Back-End.

Para el sistema utilizamos como Back-end el manejador de base de datos postgresSQL, el cual podemos manejar de manera “comando” o de manera “gráfica” con una herramienta gráfica llamada Pgadmin III que se instala con PostgreSQL en la versión para Windows.

Para crear nuestra base de datos lo haremos de manera grafica, aunque tambien podemos realizarlo en modo “comando” y ejecutar el script que viene en el anexo V. Para poder crear una base de datos en modo comando vamos a la carpeta BIN de nuestra carpeta de instalación, ejecutamos la sentencia:

```
...BIN>PSQL –U
```

Y se nos pedirá el nombre de usuario y password para poder conectarnos, (que son los que establecimos durante la instalación). Enseguida crearemos la base de datos PERAJ con la sentencia:

```
POSTGRES=# CREATE DATABASE PERAJ;
```

Ya con la base de datos creada, nos conectamos a ella para crear las tablas con la sentencia

```
POSTGRES=# \C PERAJ;
```

De esta manera todas las tablas que crearemos serán dentro de la base de datos. Para crear las tablas solo tenemos que ejecutar el script de cada tabla que viene en el anexo V.

Para el modo grafico buscamos el acceso directo a Pgadmin III y nos identificamos con el nombre de usuario y password que proporcionamos en la instalación, debajo del nombre de nuestro servidor se encuentra la opción “Base de datos” hacemos clic derecho en ella y elegiremos “Nueva base de datos..” y le daremos el nombre PERAJ con las opciones predeterminadas; desplegaremos las opciones de la opción de “Base de datos” y aparecerá la que acabamos de crear , damos doble clic sobre ella para conectarnos, en la barra de tareas presionaremos la opción “ejecutar sentencias sql arbitrarias” y en la ventana pegaremos script del anexo V y lo ejecutaremos presionando la tecla F5

Cabe señalar que la CREACIÓN de la base de datos y las tablas se realizó con una cuenta de administrador de postgreSQL.

4.6 Diseño y Generación del Front-End

El front-end es la interfaz gráfica del sistema, es decir, el conjunto de ventanas que el usuario final utiliza para capturar y administrar la información. El lenguaje utilizado para recolectar y enviar información a través de la web es PHP, y este lenguaje esta muy relacionado con el lenguaje de Hypertextos HTML, tanto es así, que el código PHP aparece normalmente insertado dentro de un documento HTML. El documento PHP, una vez interpretado correctamente en el servidor genera una página HTML que será enviado al cliente. Los fundamentos del lenguaje consultamos en el anexo III. Podemos observar la implementación de los elementos de HTML en el sistema, un ejemplo se muestra en la figura 4.6.1:

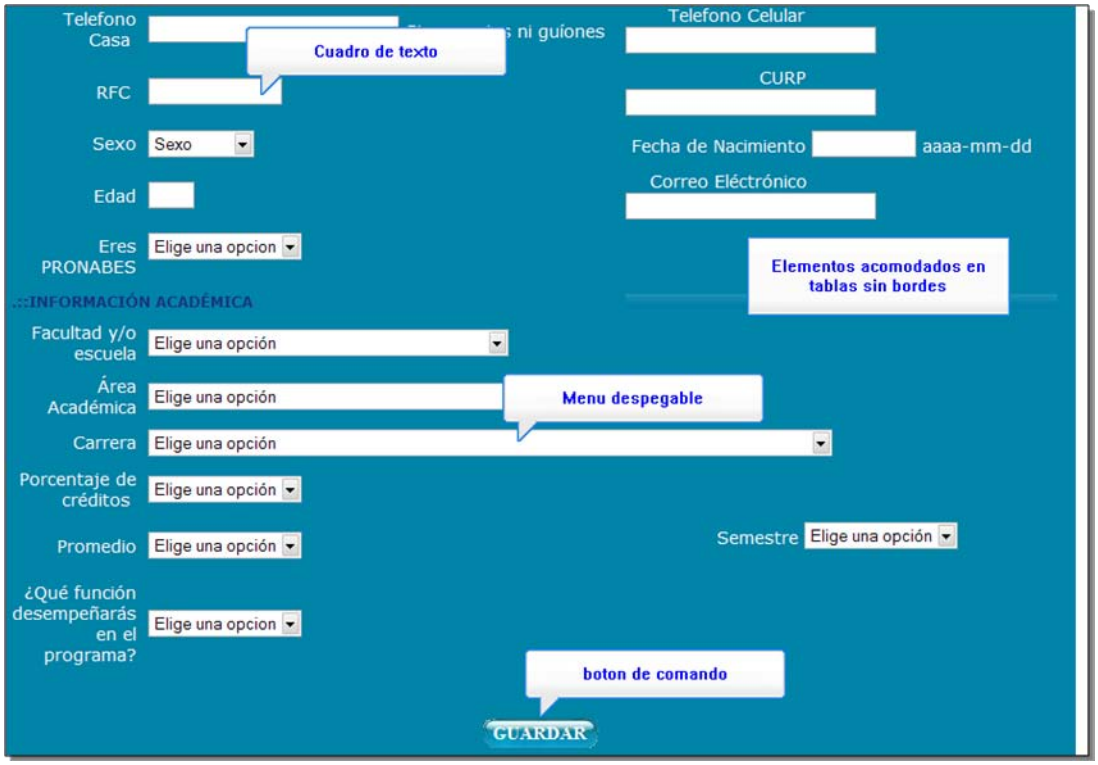

Figura 4.6.1

4.7 PRUEBAS E IMPLANTACIÓN DEL SISTEMA.

En esta fase, vamos a detectar posibles fallas en el funcionamiento de nuestro sistema para poder corregirlas y entren en operación.

En general, el objetivo del sistema es automatizar los procesos para el registro de amig@s, tutores, coordinadores e integrantes de apoyo logístico que anteriormente se desarrollaban de manera manual, por lo que traerá como consecuencia un mejor desempeño admnistrativo de dichos procesos en el plantel ya que se espera disminuir los tiempos de procesamiento de la información.

Prueba de caja negra. Se enfocan en la respuesta esperada de un módulo, intentando encontrar casos en que el módulo no cumple con su especificación. Por este motivo se denominan pruebas funcionales, y el probador se limita a suministrarle datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo el módulo en su interior.

Las pruebas de caja negra van dirigidas a aquellos módulos que van hacer interfaz con el usuario (en sentido general, teclado, pantalla, archivos, etc.), se apoyan en la especificación de requisitos del modulo.

Las pruebas de caja negra se realizaron en los siguientes modulos:

Módulo de Cartas. Al ingresar a éste modulo de la figura 4.7.1 siempre aparecerá en el panel de subtareas (parte izquierda de nuestro entorno de trabajo) las opciones que tenemos para poder generar diferentes tipos de cartas, permitiendo elegir una opción y poder guardar los archivos que se generen.

Figura 4.7.1

Después de elegir una tarea del panel de subtareas se mostrará en la parte superior en que parte del módulo nos encontramos (en este caso, tipo de carta) un campo de texto en el cual debemos de introducir el número de cuenta del alumno y un botón para generar el documento. Como se muestra en la Figura 4.7.2

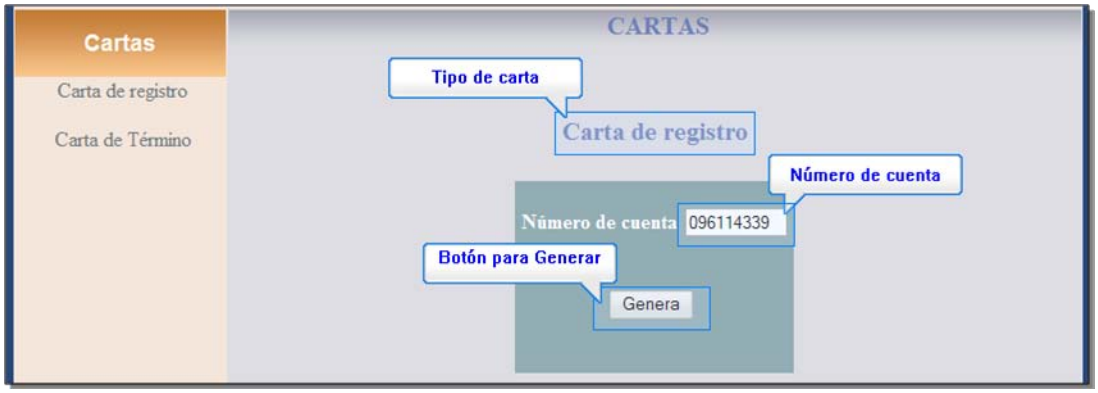

Figura 4.7.2

Pruebas de caja blanca. Se basan en la lógica del programa, no en la especificación. Se realizan utilizando el conocimiento del funcionamiento interno del código por lo que solo se pueden realizar por programadores.

Entre las pruebas de caja blanca se encuentran las pruebas de instrucciones, pruebas de decisiones que se refieren a que en el código del programa, las sentencias de bifurcación condicionales deben tener una coherencia y una cobertura de ramas al 100% como una situación deseable, pero habitualmente es un objetivo excesivamente costoso de alcanzar en su plenitud. También se encuentran las pruebas de cubrimiento y las pruebas de ciclos. En la figura 4.7.3 haremos una prueba de decisión. En la cual validamos a un usuario e inicia una sesion, si el usuario no es valido, se hace una redirección hacia la pagina de inicio de sesión

```
<?
defined( 'VALID_VARIABLE' ) or die( 'El acceso es restringido ' );


session_start();


//COMPRUEBA QUE EL USUARIO ESTA AUTENTIFICADO
if($_SESSION["Auth"]!="Yes")
{
  header("Location: ./Login.php");
}
```

Figura 4.7.3 Prueba de caja blanca

Prueba de integración. Las pruebas de integración se llevan a cabo durante la

construcción del sistema, involucran a un número creciente de módulos y terminan probando el sistema como conjunto.

Estas pruebas se pueden plantear desde un punto de vista estructural o funcional. Las pruebas estructurales de integración son similares a las pruebas de caja blanca; pero en lugar de referirnos a sentencias del lenguaje, nos referiremos a llamadas entre módulos.

Las pruebas funcionales de integración son similares a las pruebas de caja negra. Aquí trataremos de encontrar fallos en la respuesta de un módulo cuando su operación depende de los servicios prestados por otro(s) módulo(s).

Las pruebas finales de integración cubren todo el sistema y pretenden cubrir plenamente la especificación de requisitos del usuario. Ejemplo: En la figura 4.7.4 se muestra la integración entre módulos del sistema, aquí el módulo para el registro de alumnos una vez que contiene los datos del tutor, coordinador o apoyo logístico a registrar, envía dichos datos en correcto formato al módulo guardar información como lo muestra la figura 4.7.5 el cual para que cumpla sus funciones deberá estar relacionado con el primer módulo presentado y viceversa ya que una vez guardada el nuevo tutor, coordinador o apoyo logístico y el administrador puede entrar al módulo de consulta de registros para ver a los alumnos ya registrados incluyendo el recién ingresado.

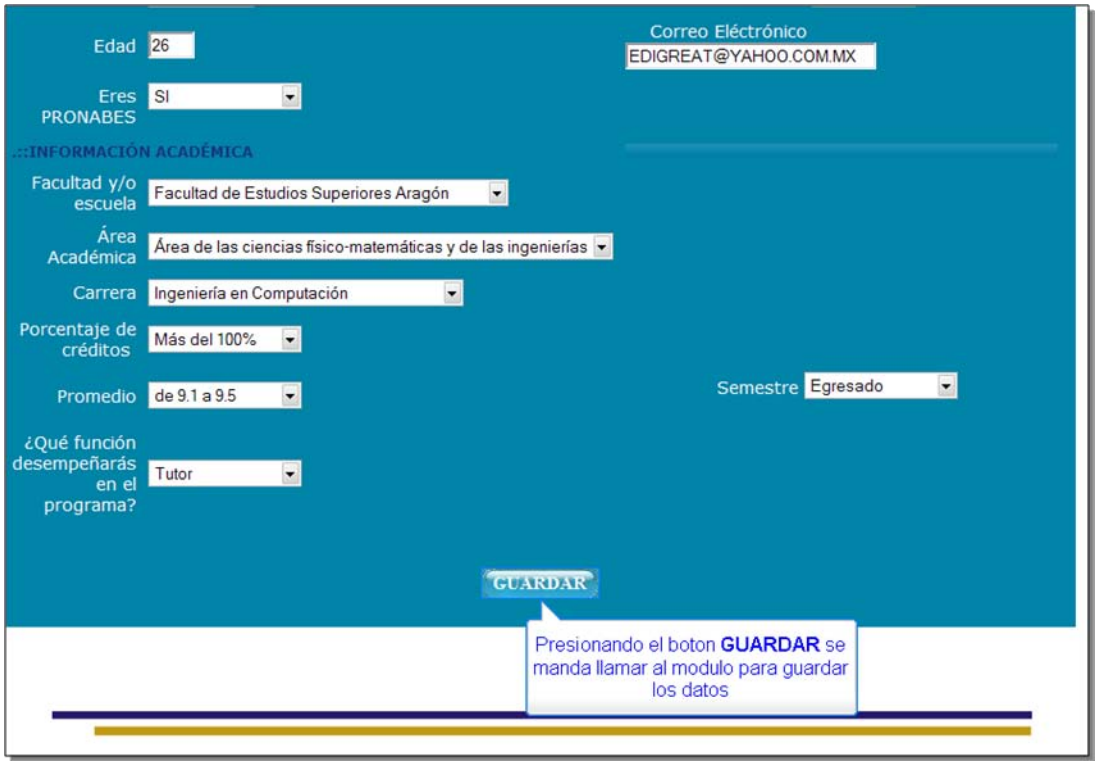

Figura 4.7.4 Guardando la información

Figura 4.7.5 Modulo para Guardar la información

Prueba de regresión. Es volver a ejecutar un subconjunto de pruebas que se han llevado a cabo anteriormente para asegurarse de que los cambios no han propagado efectos colaterales no deseados.

Prueba de validación. La validación proporciona una seguridad final de que el software satisface todos los requisitos funcionales de comportamiento y rendimiento. Durante la validación se usan exclusivamente pruebas de caja negra ya mencionada en éste capítulo.

Prueba de resistencia (Stress). Estas pruebas se ejecutan en el sistema de forma que demande recursos en cantidad, frecuencia o volúmenes anormales, es decir, están diseñadas para enfrentar a los programas con situaciones poco comunes. Ejemplo: Retomando la ventana de validación del registro de un alumno se tiene que cuando se ingresa dicha información, el usuario podría caer en el error de ingresar caracteres no válidos, o no poner el número de caracteres permitidos, debido a lo anterior el sistema esta diseñado para no tener una ruptura de la ejecución normal del mismo debido a la detección y manejo de posibles errores durante su operación. En la figura 4.7.6 se muestra un mensaje de alerta, para notificar al usuario que existen errores en su captura de de datos. Los campos que contienen errores se "iluminan" con color amarillo (figura 4.7.7) y se le pide al usuario que consulte la parte final de la página.

Figura 4.7.6 Mensaje de alerta de error

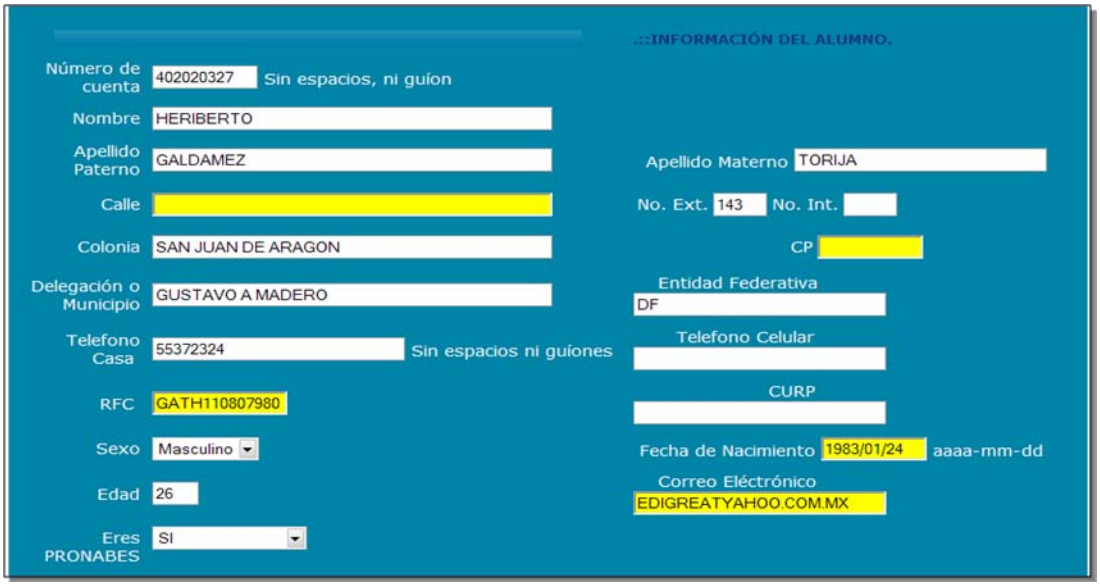

Figura 4.7.7 Campos que contienen errores.

En las figuras 4.7.6, 4.7.7 y 4.7.8 se muestra la capacidad del programa para manejar una situación anormal provocada por intentar guardar información errónea, que puede contener caracteres no válidos, campos vacíos o campos con un formato no valido, mostrando los mensajes de error con la posible causa.

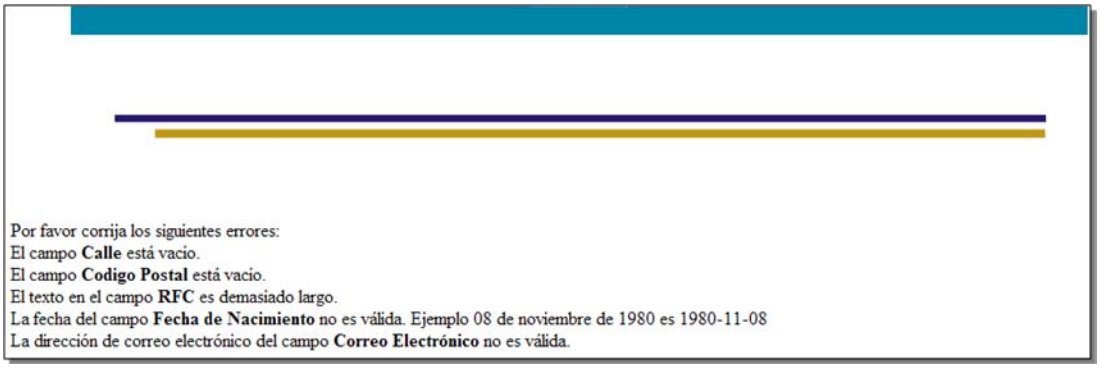

Figura 4.7.8 Causa de los posibles errores al final de la página

CONCLUSIONES.

El desarrollo de la presente tesis refleja la recopilación y aplicación de los conocimientos que hemos adquirido durante nuestra vida académica, especialmente de los años de estudio dentro de la Facultad Estudios Superiores Aragón. La formación que obtuvimos nos permite enfrentar cualquier tipo de problema y resolverlo de la forma más adecuada. La información que recibimos a lo largo de toda la carrera es bastante y a pesar de que en algunas ocasiones es muy general. Esta información son cimientos sólidos que nos sirven como base para enfrentar algún problema en nuestra vida personal y laboral. Sin embargo, el presente trabajo no lo vemos como el final o la conclusión de nuestra formación sino como el inicio de nuestro desarrollo profesional y con ello resolver cualquier tipo de problema que sea factible de ser solucionado por medios computacionales.

En la actualidad las tecnologías de la información son una herramienta que facilita las tareas de cualquier usuario, El sistema de apoyo para la captura y administración del programa del servicio social tutorial unam-peraj permitirá a los usuarios internos y externos de la Institución educativa convivir con una nueva tecnología comúnmente llamada WEB y en consecuencia derivar actividades en beneficio de la comunidad escolar que se encuentra en general.

El sistema de apoyo para la captura y administración del programa del servicio social tutorial unam-peraj permite a los usuarios desempeñar sus funciones en forma ágil y segura, es decir, se reduce el margen de error en los diferentes procesos. Finalmente, el desarrollo del Sistema refleja la formación que la Facultad de Estudios Superiores Aragón a través de los planes de estudio y educación continua ofrece a los universitarios para competir profesionalmente en el campo laboral a través de la formación recibida y ante una sociedad que requiere soluciones integrales y de calidad. El sistema diseñado ofrece una solución al exceso de información tangible (papel) que se maneja en el departamento de servicio social.

Esta solución puede ser ejecutada por varios usuarios con diferentes perfiles al mismo tiempo y realizar el mismo tramite varias veces simultáneamente, con lo cual se cumple el objetivo de evitar la perdida de tiempo.

Los diferentes departamentos dentro de una institución, no solo el de Servicio Social, pueden tener una herramienta para automatizar las tareas de gestión de consultas y captura de registros, ahorrando horas hombre para la institución. El éxito de muchas instituciones educativas se basa en el uso eficiente de su información, es decir, la capacidad que tenga esta para recibir y transmitir información, así como el manejo rápido y adecuado de la misma. El sistema desarrollado en esta tesis cumple con lo anterior dado que al disminuir los recursos humanos y materiales que consumía el proceso anteriormente al hacerlo de forma manual, disminuyo la cantidad de errores en el manejo de la información y actualmente se fluye con mayor rapidez. Se busco que el sistema fuera amigable y flexible para el usuario final para garantizar la aceptación del mismo.

GLOSARIO

INTRODUCCIÓN

Este documento recoge todos y cada uno de los términos manejados a lo largo de todo el proyecto de desarrollo del sistema. Se trata de un diccionario informal de datos y definiciones de la nomenclatura que se maneja, de tal modo que se crea un estándar para todo el proyecto.

PROPÓSITO

El propósito de este glosario es definir con exactitud y sin ambigüedad la terminología manejada en el proyecto de desarrollo del sistema de apoyo para la captura y administración de la información del programa de servicio social "UNAM-Peraj". También sirve como guía de consulta para la clarificación de los puntos conflictivos o poco esclarecedores del proyecto.

ALCANCE

El alcance del presente documento se extiende a todos los subsistemas definidos para el sistema. De tal modo que la terminología empleada, se refleja con claridad en este documento.

ORGANIZACIÓN DEL GLOSARIO

El presente documento está organizado por definiciones de términos ordenados de forma ascendente según la ordenación alfabética tradicional del español.

DEFINICIONES

A continuación se presentan todos los términos manejados a lo largo de todo el proyecto de desarrollo del sistema.

Alumno: Cualquier alumno o exalumno de la UNAM con los requisitos suficientes para realizar su servicio social

Amig@. Niños de entre 8 y 12 años de edad, de escuelas de nivel básico, de escuelas públicas cercanas al campus universitario.

ANSI. (American Nacional Standard Institute, instituto Nacional Americano de Estándares). Se trata de una organización norteamericana que se encarga de la formulación de normas en diversos sectores técnicos. En Windows es el juego de códigos empleado para definir los caracteres que se introducen en los documentos.

API. (Aplication Programming Interface). Desde un lenguaje de alto nivel el API es un conjunto de procedimientos y funciones que se ofrecen al programador para realizar una serie de acciones.

Aplicación. Es el problema o conjunto de problemas para los que se diseña una solución mediante computadora. Ejemplos son los procesadores de texto, las bases de datos, las hojas de cálculo. En Windows se emplea este término indistintamente con el de programa.

Archivo. Es un conjunto de datos relacionados de manera lógica, como puede ser el conjunto de los nombres, direcciones y teléfonos de los empleados de una empresa determinada.

Backup. Copia de seguridad. Se hace para prevenir una posible perdida de la información.

Cabecera. Encabezamiento de un impreso o documento. También se aplica a la información preliminar incluida al comienzo de un bloque de datos relativa al bloque siguiente. En comunicaciones es un bloque de caracteres que indica las características del mensaje.

Capas lógicas. Es el diseño lógico o conceptual para especificar la estructura de la aplicación, tales capas incluyen el orden de procesamiento, mantenimiento y seguimiento, a diferencia del diseño físico que especifica exactamente donde se encontrarán las piezas de la aplicación (como discos, ejecutables, cable de red y computadoras).

CGI (Common Gateway Interface). Es un protocolo genérico que permite extender las capacidades de HTTP, su función principal es la de añadir mayor interacción a los documentos web que por medio del HTML se presentan de forma estática para aportar un contenido dinámico.

Cliente/Servidor. Es un sistema en el que una máquina llamada cliente solicita a una segunda máquina llamada servidor que ejecute una tarea específica. El cliente suele ser una computadora personal común y el servidor es, por lo general, una máquina anfitriona la cual procesa las peticiones de los clientes.

Compilador. Es un programa que toma los datos de un programa escrito en lenguaje de alto nivel y da como resultado el programa escrito en lenguaje máquina. En particular, el lenguaje máquina creado por el compilador Java es conocido como Byte Code (código de bytes).

Configurar. Desde el punto de vista de software, se refiere a establecer, desde un programa especial, las características de un dispositivo periférico; desde el punto de vista de hardware, consiste en personalizar físicamente dichas características para habilitar su funcionamiento.

Coordinador. Jóvenes que en ciclos anteriores fungieron como tutores, y cada uno se encarga de apoyar la función de 25 tutores.

Dato. Es un termino genérico empleado para designar números, letras u otros caracteres existentes en una computadora o en su memoria y sobre los cuales actúan los programas.

DBA. (Database administrator). Administrador de la base de datos. Persona que diseña y mantiene la base de datos.

DBMS (Database Management System). Sistema Manejador de Bases de Datos.

DDI (Data Definition Language). Lenguaje de Definición de Datos.

DML (Data Manipulation Language) Lenguaje de Manipulación de Datos.

FTP (File Transfer Protocol). Protocolo de Transferencia de Archivos. Uno de los protocolos de transferencia de ficheros mas usado en internet.

Hipertexto. Programa de generación de documentos con un sistema de acceso que puede jerarquizar el mismo usuario que crea el documento. Los documentos creados con un programa de este tipo han sido habitualmente orientados a su utilización en multimedia, debido a su asombrosa versatilidad.

Host. Anfitrión, computador conectado a internet. Computador en general.

HTML (HyperText Marckup Language). Lenguaje de Marcas de Hipertexto. Lenguaje para elaborar paginas web. Fue desarrollado en el CERN (conseil Europeen pour la Recherche Nucleaire, Consejo Europeo para la Investigación Nuclear).

HTTP (HyperText Transfer Protocol). Protocolo de Transferencias de Hipertexto. Protocolo usado pro la red WWW, Define como se les da formato a los mensajes, como son trasmitidos y que acciones deben tomar los servidores Web y los navegadores en respuesta a varios comandos.

Intérprete. Es un programa que efectúa la traducción y ejecución simultáneamente para cada una de las sentencias de la aplicación. Es el que verifica cada línea del programa cuando se escribe.

IP (Internet Protocol) Protocolo de Internet. Es un protocolo de bajo nivel para redes que describe la manera como el usuario puede comunicarse con los miembros internet. Bajo este se agrupan los protocolos de internet. También se refiere a las direcciones de red internet.

Java. Lenguaje de programación orientado a objetos. Usado en www para la telecarga y telejecucion de programas en el computador cliente. Desarrollado por jun Microsystems, con el propósito de mejorar las capacidades de las paginas de web.

LAN (Local Area Network). Red de Área Local. Es un sistema de comunicación de alta velocidad de transmisión. Estos sistemas entran diseñados para permitir la comunicación y transmisión de datos entre estaciones de trabajo inteligentes, comúnmente conocidas como Computadoras Personales. Todas las PCs, conectadas a una red local, pueden enviar y recibir información. Cubre distancias cortas, se limita a una planta o un edificio

Lenguaje de alto nivel. Es el lenguaje de programación que utiliza una terminología fácilmente comprensible, esto es, se aproxima más al lenguaje humano.

Lenguaje de Programación. Son instrucciones creadas para la comunicación con dispositivos electrónicos por ejemplo con computadoras, con el objetivo de que realicen alguna acción y arrojen un resultado.

Manejador de base de datos. DBMS por sus siglas en inglés Data Base Management System. Es un sistema que permite crear, extraer, almacenar y manipular la información de una base de datos.

Máquina virtual Java. Es un programa que permite que se ejecute una aplicación Java en cualquier computadora que la tenga instalada. Esta es una de las características que hace a Java un lenguaje independiente de plataforma.

Página dinámica. Es una página Web cuyo contenido se genera a partir de una petición en la página o en un formulario, por ejemplo mostrar o actualizar el contenido de una base de datos.

Página estática. Es una página Web cuyo contenido se encuentra incluido en un archivo HTML, lo que significa que su contenido no cambia.

Pagina Web. También conocida como página de Internet es un documento electrónico que contiene información con texto, imágenes, audio, video y otros elementos a los que uno puede acceder a través de enlaces sobre un tema en particular y es cargado en Internet para ser accedido por cualquier persona que se conecte a esta red mundial y cuente con los permisos apropiados. La página Web puede ser dinámica o estática.

Query. Una consulta query se define como una expresión lógica sobre los objetos y relaciones definidos en el esquema conceptual: el resultado es la identificación de un subconjunto lógico de la base de datos.

RDBMS (Relational Database Management System). Sistema Manejador de Bases de Datos Relacionales).

Reglas del Negocio. Son los métodos creados con el propósito de regular alguna acción del usuario y son definidas en base a las políticas de la compañía. Por ejemplo, en una aplicación bancaria una regla del negocio podría ser que el cliente no debe retirar por taquilla más de $100,000 pesos y en caso de una petición de este tipo se genere un error.

Servidor Web. Es un programa que escucha las peticiones HTTP que le llegan. Dependiendo del tipo de la petición el servidor Web buscará una página Web o bien ejecutará un programa en el servidor. De cualquier modo, siempre devolverá algún tipo de resultado al cliente o navegador que realizó la petición.

SO (Sistema Operativo). Es el programa que administra todos los recursos de la computadora, esto es, define qué aplicaciones y en qué orden serán ejecutadas, maneja la memoria del sistema, los dispositivos de entrada, de salida y envía mensajes de error o información necesaria para el trabajo estable.

TCP/IP (Transmisión Control Protocol / Internet Protocol). El término describe dos mecanismos de software empleados para posibilitar la múltiple comunicación entre computadores de manera libre de error. TCP/IP es el lenguaje común del internet, el que permite que diferentes tipos de computadoras utilicen la red y comuniquen unas con otras, indiferentemente de la plataforma o sistema operativo que usen.

Tutores. Jóvenes universitarios prestadores de Servicio Social que guiaran durante un ciclo escolar a los amig@s, apoyándolos en sus labores académicas y culturales

URL (Universal Resource Locator). Nombre genérico de la dirección en Internet. Indica al usuario donde localizar un archivo HTML determinado, en la web.

WAN (Wide Area Network) Red de Área Global. Es una red de computadoras heterogénea sin limitación de distancia en la que sus componentes pueden estar conectados de muy diversos modos, no solamente mediante cables.

Web. Site, sitio en el World Wide Web. Conjunto de paginas web que forman una unidad de presentación, como una revista o libro. Un sitio esta formado por una colección de páginas web.

WWW (World Wide Web). Servidor de información, desarrollado en el CERN (Laboratorio Europeo de Física de Partículas), buscando construir un sistema distribuido hipermedia e hipertexto. También llamado web y www. Existen gran cantidad de clientes www para diferentes plataformas.

ANEXOS

ANEXO I. MANUAL DE USUARIO

CONSIDERACIONES

- El expediente digital sustituirá al de papel, por tal motivo, los datos deben ser capturados con el mayor cuidado.
- Con el apoyo de los documentos originales, los datos deberá ser revisados primero por el usuario y después por el departamento de servicio social.
- La información solicitada será de acceso exclusivo del departamento de servicio social.
- En cualquier computadora que tenga acceso a Internet el usuario podrá teclear la direcciòn de la página y seguir las instrucciones.

INICIO DE SESIÓN

Figura 1 Inicio de Sesión.

En la figura 1 se muestra la página principal, en ella Introduce el nombre de usuario y contraseña correspondientes, conforme a esto mostrará en la siguiente página las operaciones para los diferentes usuarios. Presiona el botón “Entrar al sistema”

a) CAPTURA AMIGO

El registro de un amig@ se divide en 5 partes, las cuales contiene diferentes campos y a continuación describiremos cada una de ellas.

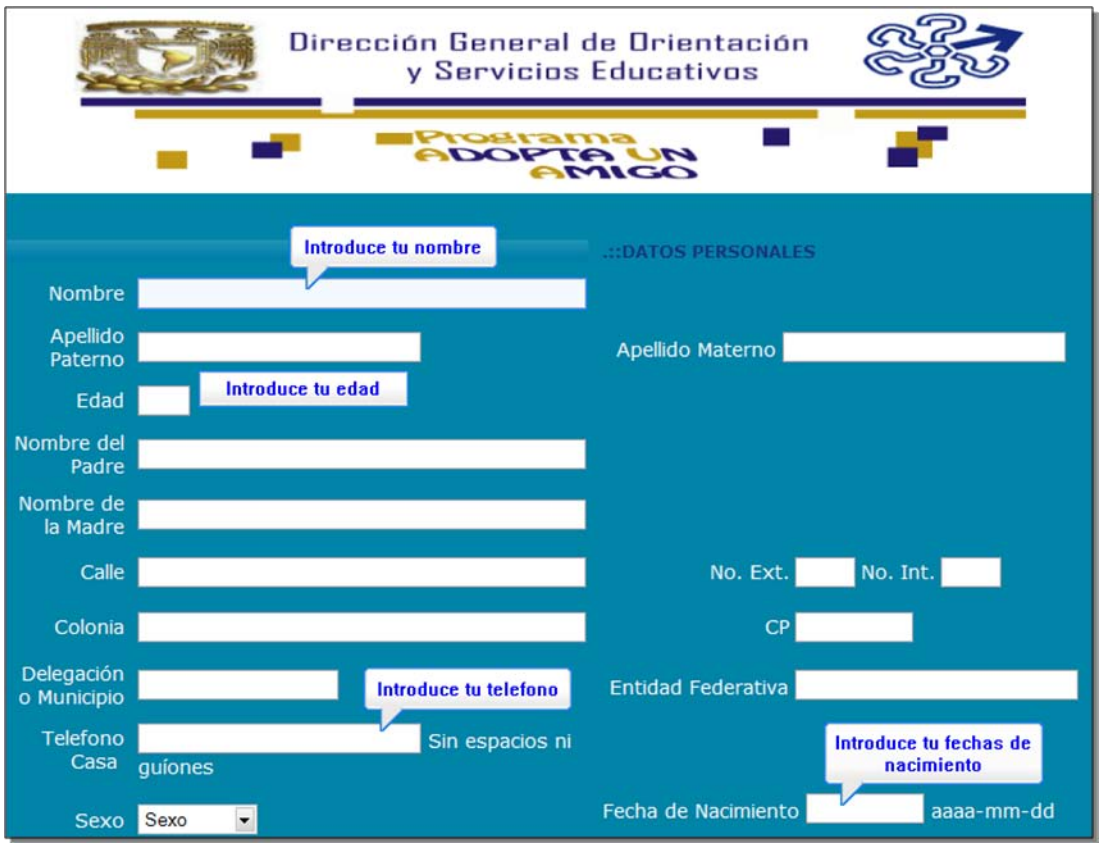

Figura 2. Datos personales de un amigo.

En la primera parte (figura 2) es dedicada a los datos personales del amigo, la mayorìa de los campos se puede introducir texto, exceptuando algunos de ellos, el campo edad solo admites dos numeros enteros, el campo de telefono casa solo admite números sin guíones ni espacios en blanco, el campo de fecha de nacimiento tiene un formato de aaaa-mm-dd por ejemplo, 1980-11-08 y en campo sexo solo elige la opción correspondiente.

En la siguiente sección es para obtener los datos de una persona con la cual podamos comunicarnos en caso de alguna emergencia, en este caso los campos son texto, exceptuando el campo del telefono.(Figura 3).

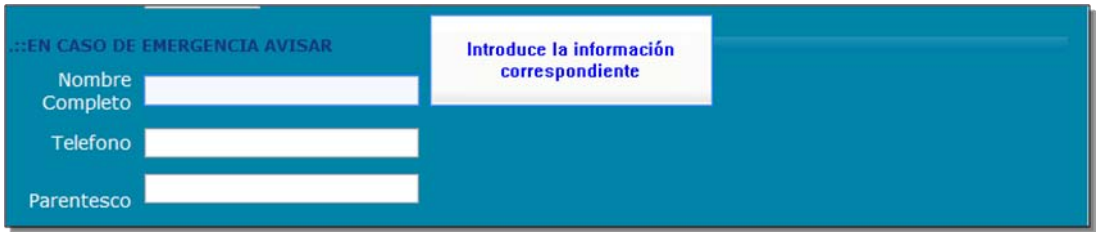

Figura 3. Persona de referencia.

En la sección acádemica se busca obtener un perfil academico estimado acerca del amig@,los campos son preguntas abiertas que se pueden responder de una manera especifica, exceptuando el campo del promedio, aquí solo se elige la opción correspondiente.(Figura 4)

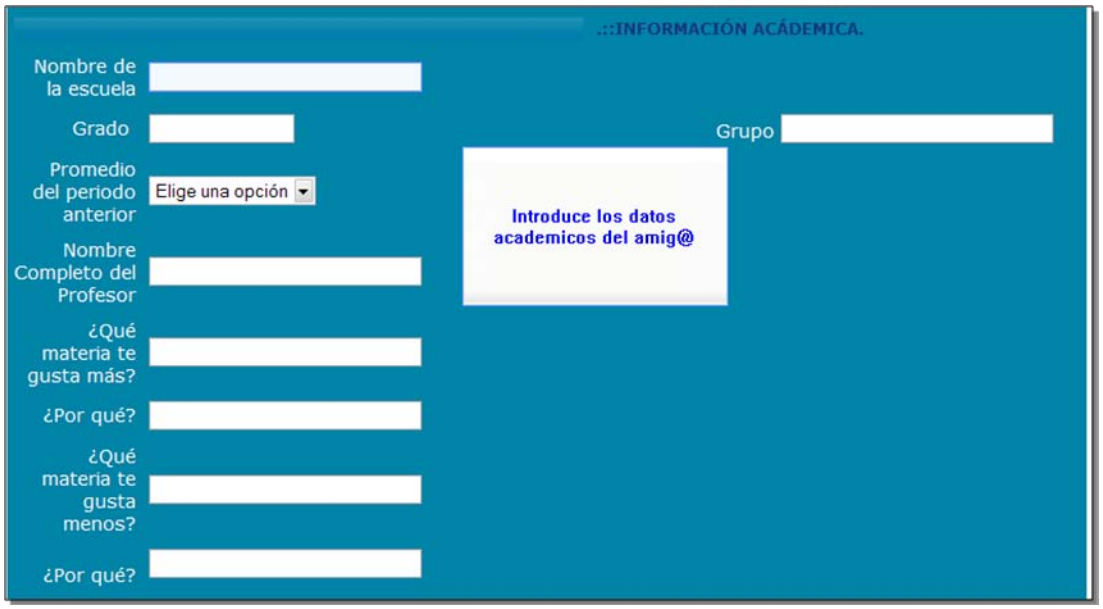

Figura 4. Datos Acádemicos del alumno

En las siguientes dos secciones se tiene por objetivo conocer la relación con la familia y un perfil aspiracional del amig@, en las preguntas de ¿Con que miembro de tu familia te llevas mejor? y ¿Con que miembro de la familiar NO te llevas bien? Se despliegan varias opciones, en caso de que no aparezca la deseada se elige la opciòn “Otro” y aparecerá un campo de texto en el cuál podrás ponerla. También en la pregunta ¿Te gustaría hacer algo diferente? En caso de elegir la respuesta “SI” aparecerá un campo de texto que complementará la respuesta a esta pregunta. (Figura 5)

Cuando se haya finalizado el registro del amig@, presionaremos el boton “Guardar”. Y aparecerá la siguiente pantalla mostrando el comprobante de inscripción, como se muestra en la figura 6

Figura 5. Datos Adicionales del Amig@

Figura 6. Comprobante de inscripción del amigo

En la cual se puede observar algunos datos personales del amigo y se mostrará el folio de registro que se le asigno para identificarlo de manera única en el sistema. Aquí ha finalizado el registro del amig@.

B) Captura Tutor, Coordinador o Apoyo Logistico.

A continuación se describirá la forma para el registro de un tutor, coordinador o apoyo logistico en el sistema, es importante conocer la función que el prestador del servicio social realizará dentro del programa antes de su registro, ya que aquí deberá confirmar dicha función. También es muy importante su número de cuenta, ya que con este se podrá identificar de manera única en el sistema. El registro consta de dos partes que a continuación describiremos y podemos observar en la figura 7.

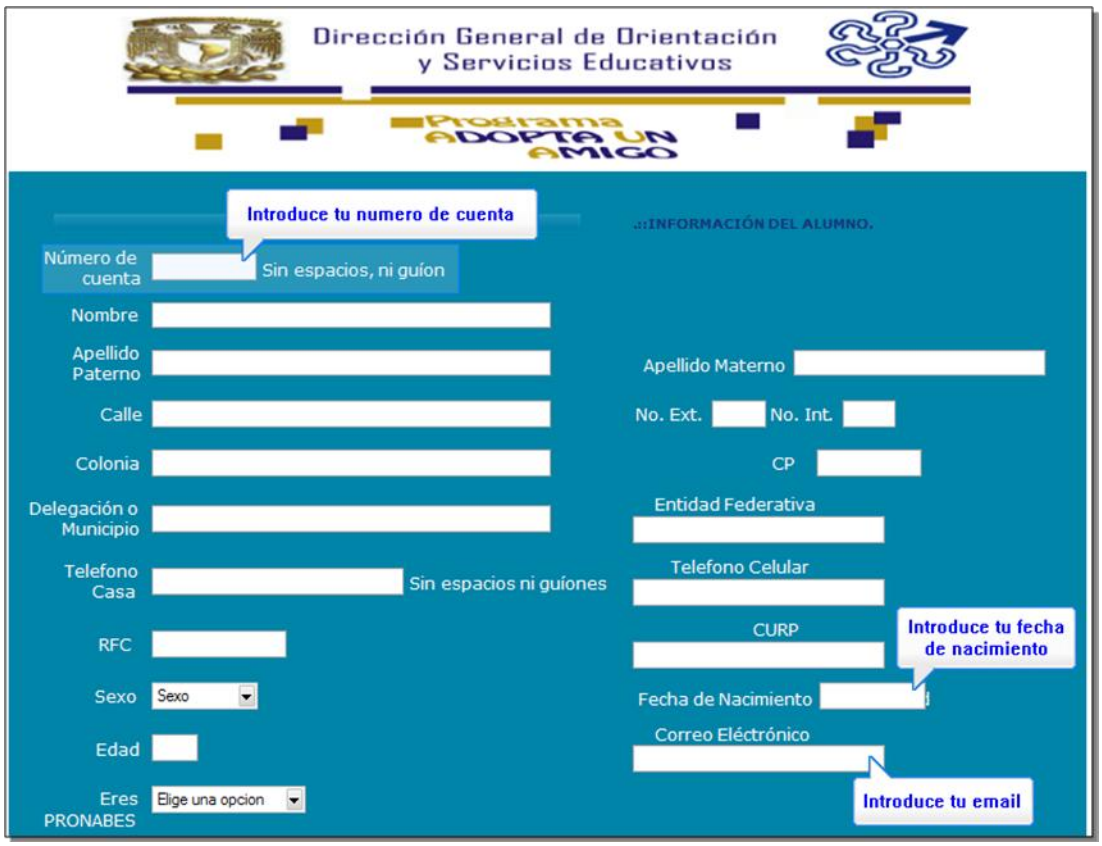

Figura 7. Datos Personales del alumno.

En la primer parte es acerca de la información personal del alumno la mayoriá de los campos son de texto, exceptuando los siguientes:

- Campo de número de cuenta. Deberà de ser de 9 digitos sin espacios ni guiones en caso de que sea una generaciòn anterior a la 2000 deberá anteponer un 0 (cero).
- El campo edad solo acepta dos numeros enteros.
- El campo telefono solo acepta numeros sin espacios ni guiones.
- El campo de la Fecha de nacimiento tiene un formato aaaa-mm-dd, ejemplo (1980-11-08).
- El campo de correo eléctronico deberá cumplir con las caracteristicas tipicas,por ejemplo (usuario@servidor.com).

En esta parte, es importante el número de cuenta, por que es un identificador único, así que ningún prestador de servicio social podrá desempeñar dos funciones dentro del programa al mismo tiempo. El campo de correo eléctronico también es importante, ya que por medio de este llegarán noticias, actividades o anuncios acerca del programa. El campo de becas PRONABES también es muy importante, ya que con tu participación en este programa podrás cubrir los requisitos y los compromisos que realizaste al integrarte a este programa de becas.

En la siguiente sección elige las opciones correspondientes en cada uno de los campos, es importante que se conozca la función que se desempañará dentro del programa previo al registro, presiona el boton “Guardar” cuando haya terminado de llenar todos los campos. (Figura 8).

Figura 8. Datos academicos del alumno.

A continuación se mostrará en la figura 9 una pagina con el comprobante de inscripción de un tutor, coordinador o apoyo logistico en el sistema.

Figura 9. Comprobante de inscripción.

En el cual se podrá observar algunos datos personales, como el número de cuenta o el perfil elegido dentro del programa. Con este paso se finaliza el registro.

OBSERVACIONES SOBRE EL REGISTRO.

- La mayoría de los campos son obligatorios
- Algunos campos deben de cumplir con un formato definido, por ejemplo, la fecha de nacimiento.
- En caso de existir un error, revisar la parte final de la pagina para consultar la posible causa, además de verificar los campos iluminados con amarillo , que son los que no cumplen con algún requisito.

ADMINISTRACIÓN DE LA INFORMACIÓN

Cuando entramos en esta parte del sistema, lo primero que veremos es la siguiente pantalla, que muestra en la figura 10

Figura 10. Menu de inicio.

Donde encontramos el nombre del usuario, la barra de Tareas, el panel de resultados y un boton de Salir cuando se desea abandodar el sistema. En la barra de Tareas encontramos las diferentes opciones que podemos realizar para administrar la información capturada en el sistema. Estas tareas son:

- Registros. En esta parte podremos consultar la información de los amig@s, tutores, coordinadores o integrantes del apoyo logistico.
- Tutorias. En esta sección haremos las asignación de un tutor a un amig@.
- Gafete. Los gafetes para identificar a cada participante se generarán en esta parte.
- Estadisticas. Consultaremos diferentes tipos de estadisticas para conocer el estado del sistema en cuanto a información capturada se refiere.
- Cartas. Las cartas de registro y termino de Servicio Social se podrán generar en esta secciòn

En cada una de las tareas, al presionar sobre ellas, aparecerá un nuevo menú de subtareas, al lado izquierdo de nuestro panel de resultados, que nos mostrarán las diferentes opciones que podemos realizar en la tarea elegida.

A continuaciòn se describirán cada una de las tareas que podemos realizar en el sistema

A) Registros

En “Registros” podremos consultar la información capturada en el sistema, ya sean amig@s, tutores, coordinadores o apoyo logistico, el panel de tareas nos otorga filtros para seleccionar que tipo de personas podemos consultar las opciones que tenemos son:

- Amigos con tutor. Se desplegarán los amigos que ya han sido asignados a un tutor.
- Tutores con amigo. Se desplegarán los tutores que ya han sido asignados a un amig@.
- Amigos sin asignar. Amig@s que no tienen tutor asignado.

- Tutores sin asignar. Tutores que no tienen asignado a un amig@.
- Coordinadores. Coordinadores inscritos en el programa de servicio social.
- Apoyo logistico. Integrantes del apoyo logistico dentro del programa de servicio social.

Figura 11. Consulta de registros

En cada opción la información desplegada en el panel de resultados es semejante, se tiene el titulo de la subtarea, el número de personas o registros, un boton para generar un reporte con formato XLS (deberá contar con un programa para poder visualizar el contenido de estos archivos) que contiene toda la información de las personas que corresponden a la subtarea elegida.

Enseguida viene una tabla con la información básica de las personas que integran esta subtarea, esta tabla contiene diferentes campos, por ejemplo, la imagen corresponden a los tutores que no han sido asignados a un amig@, estos campos son el número, número de cuenta, nombre completo, escuela o facultad, carrera y acciones. Estos datos cambiarán según la subtarea elegida, por ejemplo, los tutores que ya han sido asignados contendrá el nombre del amig@ que esta bajo su tutela.

El campo acciones contiene un boton “Ver información” que cuando presionemos mostrará un formulario igual al de registro dentro del panel de resultados, ya sea de amigo o de tutor, el cual contendrá la información de la persona elegida, si se necesita hacer un cambio , se llena el campo correctamente (Ver CAPTURA DE DATOS) y se presiona el boton “Guardar”.

Es muy importante que la información guardada en el sistema sea la correcta, ya que con ella podremos generar diferentes documentos de importancia, por ejemplo las cartas de registro de servicio social.

b) TUTORÍAS.

En esta parte podremos realizar o quitar una asignación tutor-amig@, en el panel de subtareas tenemos dos opciones, asignar un tutor a un amigo y quitar una asignación de tutor, para la primera opción, primero elegiremos al tutor deseado, es se podrá por buscar por facultad o carrera o por ambas, cuando se elija la facultad o carrera deseada, en la parte de abajo aparecerán todas las personas que cubran con este requisito y no tengan asignado a un amig@, cuando encontremos al tutor indicado, presionamos el boton "Asignar Amigo" (Figura 11).

Figura 12. Elegir un tutor para una asignarlo a un amigo.

Enseguida aparecerán una lista de amig@s que no han sido asignados, se busca al amig@ deseado, (la lista se encuentra ordenada alfabeticamente por nombre) y presionamos el boton "Asignar Amigo", con esto habremos hecho la asignación(Figura 13).

Figura 13. Elegir un amig@ para asignarlo a una tutoria

Para Quitar la asignación de un tutor a un amig@ son los mismos pasos en la segunda opción del panel de subtareas, primero elegiremos al tutor que deseamos retirar de la tutoría (previamente ya asignado) y presionamos el boton “Quitar asignaciòn” y enseguida aparecerá el amigo que se le fue asignado, presionamos el boton “Quitar amigo” y con esto habremos eliminado la asignación.

c) GAFETES

En esta opción se podrán generar diferentes tipos de gafetes según el papel desempeñado en el programa, en el panel de subtareas podremos elegir tenemos diferentes opciones, para tutores, coordinadores o apoyo logistico necesitarán el número de cuenta con el cual se registrarón en el programa, y para los amigos su folio (este puede consultar en la sección de registros). Si contamos con una foto tamaño infantil en formato jpeg o png, presionaremos el boton examinar y le daremos la ruta para que la foto salga en el gafete, si no, se deja en blanco este campo, para generar el gafete se presiona el boton enviar.

Figura 14. Gafetes

El gafete se generará en formato pdf (se necesitará un programa para poder visualizar este tipo de archivos) el cual contendrá en la parte de enfrente su foto (o un cuadro en blanco para

que despues la pueda pegar), su nombre y su funciòn, en la parte de atrás información personal, en el caso de tutores, coordinadores y apoyo logistico, la carrera, escuela o facultad, domicilio y telefono, en el caso de amig@ escuela, domiclio y telefono.

Para poder diferenciar los tipos de gafetes cada uno tiene un color caracterísctico en el campo del nombre, para amig@s y turores es de color verde y para coordinadores y apoyo logistico es de color rojo, un ejemplo de un gafete de un tutor se muestra en la figura 15:

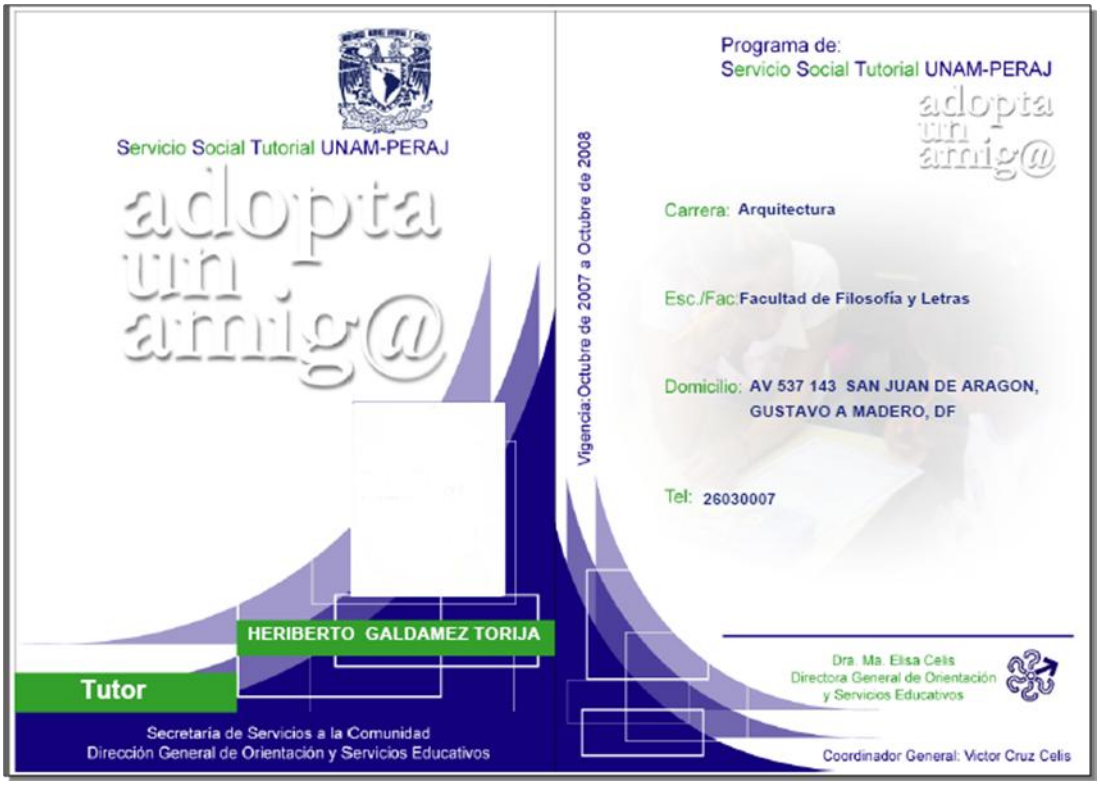

Figura 15. Gafete de un Tutor.

d) ESTADISTICA

En esta sección podremos conocer la información capturada en el sistema , en el panel de subtareas elegiremos una opción , amigos o alumnos (esta última incluye tutores, coordinadores, y apoyo logistico) y enseguida se podrá elegir diferentes opciones para realizar diferentes conteos, por ejemplo, cuantos coordinadores, tutores y personas de apoyo logistico hay registrados en el sistema(Figura 16).

Figura 16. Estadisticas

e) CARTAS

Y la última tarea es generar cartas de registro y termino de servicio social de los alumnos o exalumnos registrados en el programa, del panel de subtareas podremos elegir que tipo de carta deseamos generar, carta de registro (o de aceptación) o carta de termino, después proporcionaremos el número de cuenta del alumno y presionaremos el boton "Genera", enseguida aparacerá un enlace en la parte de abajo, lo presionaremos y obtendremos el documento, este se generará con la información guardada en el sistema, y se generarán según la escuela de donde proviene el alumno, y en algunos casos, también la carrera, ya que existen escuelas o facultades que tienen diferentes formatos para sus diferentes carreras. El formato de los documentos es RTF y se deberá contar con un programa que permita visualizar el contenido de este tipo de archivos.

Figura 17. Cartas de registro y Termino de servicio social

A continuación se muestra un ejemplo en la figura 18 de una carta de aceptación de la Facultad de Estudios Superiores campus Aragón

DIRECCION GENERAL DE ORIENTACION
Y SERVICIOS EDUCATIVOS
SUBDIRECCION DE SERVICIO SOCIAL
Y VINCULACIÓN LABORAL
DEPARTAMENTO DE PROGRAMAS
MULTIDISCIPLINARIOS

Of. No. DGOR/DPM/051/06

Asunto: Carta de Aceptación

MTRO. FERNANDO GARCIA HERNANDEZ
JEFE DEL DEPARTAMENTO DE SERVICIO SOCIAL
FACULTAD DE ESTUDIOS SUPERIORES ARAGON
P R E S E N T E .

Por este conducto hacemos de su conocimiento que el (la) alumno (a) **DAVID MONTES GONZALEZ**, con número de cuenta **099114338** de la carrera de **Ingeniería en Computación** , ha sido aceptado (a) para realizar su Servicio Social en esta institución dentro del Programa Universitario de Servicio Social UNAM-PERAJ "Adopta un amigo" clave 2006-12/238-1514, a partir del 03 de octubre de 2007 al 03 de octubre de 2008 cubriendo un total de 480 horas y realizará las siguientes actividades:

- Orientación Vocacional.
- Tutorías a nivel básico.
- Planeación de actividades culturales
- Planeación de actividades recreativas

Agradeciendo de antemano la atención que se sirva prestar a la presente, envío a usted un cordial saludo.

A T E N T A M E N T E
"POR MI RAZA HABLARÁ EL ESPÍRITU"
Cd. Universitaria, D.F., 3 de octubre de 2007.

LA JEFA DE DEPARTAMENTO

Figura 18. Ejemplo de carta de registro de servicio social.

Y por ultimo se queremos cerrar la sesión del administrador solo tenemos que presionar el boton salir que se encuentra a un lado de l a barra de tareas.

A continuación se describe lo necesario para poder instalar el sistema en una plataforma Windows, los requirimientos son minimos, pero estos pueden cambiar si el alcance del sistema es muy grande.

INSTALACIÓN Y CONFIGURACIÒN DE APACHE

Existen dos versiones de Apache, la versión 1.3 y la versión 2, instalaremos y configuraremos esta versión. Se puede descargar de distintos lugares, pero la mejor es la pagina oficial para realizar la descarga.

http://www.apache.org

La instalación de Apache es sencilla, al ejecutar el archivo que se descargue se mostrara una ventana de bienvenida, para continuar presionaremos **Next >.** Despues tendemos que aceptar el acuerdo de licencia y presionaremos **Next >.** En "**Network Domain"** y **"Server Name"** escribiremos la dirección IP de nuestra computadora (127.0.0.1) y en **"Administrator's Email Address"** una dirección de correo electrónico, en cuanto al tipo de instalación elegiremos la Típica (Typical).

A partir de aquí solo presionaremos Next >, Install y Finish en este orden sin modificar nada.

Una vez instalado se mostrara un nuevo icono en barra del reloj.

Ya instalado, ahora toca configurar el archivo **httpd***. Este archivo se encuentra en la carpeta* confg *en donde se ha instalado Apache.*

Abrimos el archivo y editamos los siguientes parámetros:

Buscamos "Dynamic Shared Object (DSO) Support", tras esto aparecen estas líneas:

EXAMPLE:

LOADMODULE FOO_MODULE MODULES/MOD_FOO.SO

#

Debajo del signo (#) añadimos las siguientes líneas:

LOADMODULE PHP5_MODULE C:/PHP/PHP5APACHE2.DLL

Esto es para cargar el modulo de PHP al iniciar Apache, sin esto PHP no funcionaria con lo cual cualquier sistema en este lenguaje no podrían ser interpretados.

También tenemos que configurar el directorio en el cual guardamos nuestros sitios web, por defecto Apache lo define como

C:\ARCHIVOS DE PROGRAMA\APACHE GROUP\APACHE2\HTDOCS\

Para poder modificar lo anterior en el archivo httpd buscamos las siguientes líneas:

DocumentRoot: The directory out of which you will serve your

documents. By default, all requests are taken from this directory, but

symbolic links and aliases may be used to point to other locations.

#

DocumentRoot "C:/Archivos de programa/Apache Group/Apache2/htdocs"

Substituimos el contenido de esta ultima línea por la ruta de la carpeta en la cual esta almacenada el sistema, podemos crear una carpeta en la cual se vaya a almacenar los archivos del sitio web, por ejemplo crea una carpeta llamada “sitios” en el disco duro.

Entonces quedaría algo parecido a esto:

…

symbolic links and aliases may be used to point to other locations.

#

DocumentRoot "C:/sitios"

También modificaremos nuestra pagina de Index (index es la pagina principal de nuestro sitio web) puede ser, php, php3, htm, html, y otros formatos, busca las siguientes líneas:

DirectoryIndex: sets the file that Apache will serve if a directory

is requested.

#

The index.html.var file (a type-map) is used to deliver content-

negotiated documents. The MultiViews Option can be used for the

same purpose, but it is much slower.

#

DirectoryIndex index.html index.html.var

Substituimos la ultima línea por esto

DirectoryIndex index.html index.htm index.php index.php3 index.phtml index.html.var

Y añadimos las siguientes líneas:

AddType application/x-httpd-php .php .php3 .phtml

AddType application/x-httpd-php-source .phps

Para nuestro sistema hemos utilizado la versión 8.2 de postgreSQL instalaremos y configuraremos esta versión. Se puede descargar de distintos lugares, pero que mejor que la pagina oficial para realizar la descarga.

http://www.postgresql.org

Para la versión en Windows, podemos descargar el archivo postgresql-8.2.6-1.zip, que contiene cuatro archivos, tres ejecutables y uno de texto, el primero que ejecutaremos es el archivo postgresql-8.2, seleccionaremos el idioma ingles y presionamos el botón START > y en la siguiente pantalla presionaremos NEXT > y en las notas de instalación también presionaremos NEXT >, en seguida aparecerán las opciones de instalación, dejamos la selección que se proporciona y presionaremos el botón NEXT >, después aparecerá la configuración del servicio , presionaremos el botón NEXT > y creara un usuario para este servicio con un password de manera aleatoria, esto es para administración de servicios de windows, en la pantalla siguiente podremos establecer el súper usuario y la contraseña con las cuales podremos realizar la conexión con este SGDB , después podremos activar los lenguajes procedurales, nosotros solo necesitaremos el PL/pgsql, enseguida elegiremos los modulos que se van a instalar, solo instalaremos el adminpack, y presionamos NEXT > y la siguiente pantalla de igual manera para completar la instalación , cuando termine el proceso presionamos el boton FINISH >

El siguiente paso es ejecutar el archivo UPGRADE.BAT que viene en nuestra archivo en zip , cuando este se ejecuta, se presiona una tecla para continuar con el proceso y se continua con la instalación, cuando este termina debemos reiniciar la computadora.

INSTALACIÓN Y CONFIGURACIÓN DE PHP.

Utilizaremos la última versión que ha salido de PHP, que es la 5. Una vez descargado el archivo (el zip alrededor de 7’3 MB, no el auto-ejecutable que también se puede descargar), lo descomprimiremos en C:\PHP. Esto lo podremos encontrar en http://www.php.net

Encontraremos una gran cantidad de archivos en la carpeta descomprimida. Uno de ellos es *php5ts.dll*, copiaremos este archivo en la carpeta *C:Windows\System* (Windows9x) o en *C:WINNT\System32* (XP, 2000 o NT).

Además también debemos hacer lo mismo con los siguientes archivos con la extensión dll de la carpeta *C:\PHP* a tu correspondiente directorio anterior:

- fdftk.dll
- fribidi.dll
- gds32.dll
- libeay32.dll
- libmhash.dll
- libmysql.dll
- libmysqli.dll
- msql.dll
- ntwdblib.dll

- ssleay32.dll
- yaz.dll
- php_pgsql.dll

También hay que configurar el archivo con extensión ini de php el cual tras haberlo editado lo guardaremos en la carpeta Windows o WINNT según corresponda a tu sistema operativo.

Para ello abriremos con algún editor de texto el archivo *php.ini-dist*, tendremos que editar una serie de líneas como con el httpd de Apache.

Buscaremos estas líneas:

REGISTER_GLOBALS = OFF

en lugar de Off escribiremos On.

Ahora hay que indicarle a PHP la carpeta en la cual se encuentran las extensiones, en este caso la carpeta se llama *"ext"* .

Buscamos estas líneas

; DIRECTORY IN WHICH THE LOADABLE EXTENSIONS (MODULES) RESIDE.

EXTENSION_DIR = "./"

Sustituimos *"./"* por C:/PHP/EXT

Ahora buscamos estas líneas:

;WINDOWS EXTENSIONS

Aparecerán a continuación una serie de *dll* precedidas por punto y coma, deberemos quitar el punto y coma que precede a *php_pgsql.dll*. Con esto haremos posible la correcta utilización de postgreSQL con PHP.A continuación crearemos dos nuevas carpetas en *C:\PHP*, una la llamaremos uploads y la otra sessions.Ahora buscaremos la siguientes líneas en el archivo ini de PHP que estábamos editando:

; TEMPORARY DIRECTORY FOR HTTP UPLOADED FILES (WILL USE SYSTEM DEFAULT IF NOT

; SPECIFIED).

UPLOAD_TMP_DIR =

Detrás de *upload_tmp_dir=* escribiremos C:\PHP\UPLOADS

Ahora buscamos esto:

SESSION.SAVE_PATH = "N;/PATH"

y en lugar de *"N;/path"* escribimos C:\PHP\SESSIONS

Con este último paso tendremos todo configurado, ahora solo quedaría guardar el archivo que estamos editando como *php.ini* en Windows o WINNT como antes se ha indicado

ANEXO III. HTML.

En 1989 existían dos técnicas que permitían vincular documentos electrónicos, por un lado los hipervínculos (links) y por otro lado un poderoso lenguaje de etiquetas denominado SGML. Por entonces un usuario conocedor de ambas opciones, Tim Berners-Lee físico nuclear del Centro Europeo para la Investigación Nuclear da a conocer a la prensa que estaba trabajando en un sistema que permitirá acceder a ficheros en línea, funcionando sobre redes de computadoras o máquinas electrónicas basadas en el protocolo TCP/IP.

A principios de 1990, Tim Berners-Lee define por fin el HTML como un subconjunto del conocido SGML y crea algo más valioso aún, el World Wide Web. En 1991, Tim Berners-Lee crea el primer navegador de HTML que funcionaría en modo texto y para UNIX.

HTML

HTML es el acrónimo inglés de **H**yper**T**ext **M**arkup **L**anguage, que se traduce al español como *Lenguaje de Etiquetas de Hipertexto*. Es un lenguaje de marcado diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas web. Gracias a Internet y a los navegadores como Internet Explorer, Opera, Firefox, Netscape o Safari, el HTML se ha convertido en uno de los formatos más populares y fáciles de aprender que existen para la elaboración de documentos para web.

HTML no es un Lenguaje de Programación, aunque si permite incluirle código en Lenguajes de Programación, bajo ciertos criterios, extendiendo su capacidad y funcionalidad, aunque eso se logre excediendo los alcances del HTML en si.

En pocas palabras, simplemente usando el NOTEPAD de Windows podemos realizar páginas Web, ya que una pagina Web es simplemente un texto al que se le añaden diferentes etiquetas con el objeto de indicar diferentes propiedades. De cualquier manera existen editores específicos para realizar páginas Web. Dichos editores permiten por medio de menús e iconos incluir etiquetas de HTML sin necesidad de teclearlas.

Tags HTML y Atributos

Para indicarle al navegador como vamos a utilizar la información que contiene nuestro fichero existen códigos o etiquetas, también llamados tags que se encargan de dicha tarea, la manera de utilizar estos códigos es mediante los símbolos “**<**” y “**>**” y dentro de ellos la etiqueta que queremos utilizar, por ejemplo <table>, que indica que utilizaremos una tabla para organizar lo que viene adelante.

Normalmente una etiqueta HTML debe ser cerrada para indicarle al navegador que deseamos terminar de utilizarla, por ejemplo para cerrar el código HTML que utilizamos antes se utiliza </table>, si nos fijamos bien es la misma palabra solo antecedida por un “/”, normalmente esta es la manera de cerrar los códigos HTML.

Para quedar un poco mas claro ilustremos un poco más el ejemplo.

```
<TABLE BORDER=1 WIDTH=300>
<TR><TD WIDTH=100>prueba</TD><TD WIDTH=100>prueba</TD>

<TD WIDTH=100>prueba</TD></TR><TR><TD WIDTH=100>prueba</TD>
<TD WIDTH=100>prueba</TD><TD WIDTH=100>prueba</TD></TR>
</TABLE>
```

Dentro del tag <TABLE>, podemos ver las palabras BORDER y WIDTH, que le indican al navegador atributos de visualización para ese tag en este caso el tamaño del borde de la tabla a 1 y la anchura de la tabla a 300; cada una de las etiquetas HTML contiene atributos que

proporcionan características (por ejemplo, altura, ancho, color, etc.) algunas veces muy importantes que finalmente definirán cómo debe ser interpretado el tag.

Entonces el código HTML del ejemplo nos desplegara algo más o menos así en nuestro navegador:

prueba	prueba	prueba
prueba	prueba	prueba

Ahora pasemos a definir las etiquetas básicas de HTML para crear una pagina web.

Etiqueta < HTML >: Identifica al documento como uno del tipo HTML e indica el comienzo del mismo. Todas las etiquetas que forman el documento deben estar encerradas dentro de las etiquetas HTML (en el formato de inicio y cierre de etiquetas). Dicho mas en cristiano: La primera instrucción que debemos colocar en nuestro editor es la etiqueta HTML en su formato de apertura; asimismo, la última será la misma etiqueta en su formato de cierre. Esto siempre tiene que ser así.

Etiqueta < HEAD >: Esta etiqueta significa cabecera y en ella se encierra la información técnica del documento que se esta elaborando. El término información técnica se refiere a aquellos datos que a la persona que va a visitar la página no le interesan, pero que el software navegador requiere para poder reconocer la página. Básicamente, la información técnica que con mayor frecuencia manipulamos es la referente al título de la página.

Etiqueta < TITLE >: Significa título y aquí se encierra el nombre del documento o página web que se esta elaborando. Esta información se considera de tipo técnica y va, a su vez, encerrada dentro de la etiqueta < HEAD >.

Etiqueta < BODY >: Comprende el cuerpo del documento o Página Web en si. Aquí se encierra todo el texto, imágenes, tablas, formatos, etc, que serán visibles en la pantalla. A partir de esta etiqueta comienza propiamente el diseño de nuestra página.

Esta etiqueta puede ser complementada por medio de extensiones o *atributos*. A continuación presentamos algunas de ellas:

EXTENSIONES:

BGCOLOR, BGPROPERTIES, TEXTBACKGROUND, TEXTCOLOR. Estas son las mas importantes ,y brevemente explicaremos la utilidad de algunas de ellas:

- BGCOLOR se utiliza para indicar el color de fondo que queremos para nuestra página: se indica con un código hexadecimal de 6 caracteres donde:
  Los 2 primeros caracteres indican la cantidad de ROJO. Van desde 33 (rojo mas claro) hasta ff (rojo mas intenso); Los 2 caracteres intermedios indican la cantidad de VERDE.; Los 2 caracteres finales indican la cantidad de AZUL.
  El color también puede indicarse colocando su nombre en ingles.

- TEXTBACKGROUND se utiliza cuando, en vez de querer que nuestra página tenga un color de fondo, presente como fondo una imagen. Esta imagen tiene que ser del tipo JPG o GIF.

Dentro del cuerpo <body> podemos encontrar numerosas etiquetas. A continuación se indican algunas a modo de ejemplo:

- **<h1>, <h2>, <h3>, <h4>, <h5>, <h6>:** encabezados o títulos del documento con diferente relevancia.
- **<table>:** define una tabla
- **<tr>:** fila de una tabla
- **<td>:** celda de datos de una tabla
- **<a>:** Hipervínculo o enlace, dentro o fuera del sitio web. Debe definirse el parámetro de pasada por medio del atributo *href*. Por ejemplo: Fes Aragón se representa como Fes Aragón.
- ** :** área de la página
- **<img>:** imagen. Requiere del atributo *src*, que indica la ruta en la que se encuentra la imagen
- **<li>, <ol>, <ul>:** Etiquetas para listas.
- <b>: texto en negrita (*Etiqueta descartada. Se recomienda usar la etiqueta *)
- **<i>:** texto en cursiva
- **<u>:** texto subrayado

Etiqueta < P >: Su uso es muy simple, tan solo indica el comienzo y el final de un párrafo. Un párrafo puede ser escrito obviando esta etiqueta, pero el utilizarla nos garantizará que el escrito al momento de ser visualizado tenga mejor apariencia, pues mediante algunas extensiones que tiene esta etiqueta podrá ser colocado de una forma que se adapte mas a nuestros intereses. La extensión mas importante que emplea es ALIGN, la cual se emplea para indicar el alineamiento que queremos tenga nuestro párrafo con respecto a algún otro elemento presente en la tabla (*una imagen, una tabla, etc*). Los atributos que puede tomar ALIGN son: Left, Right, Center, justify

Etiqueta < BR >: Baja el texto que le sigue a la siguiente línea. Funciona, en esencia como un *ENTER*. Obviamente, no requiere etiqueta de cierre, ni de atributos. Podemos colocar tantos saltos de línea como queramos.

Etiqueta < HR >: Su función es similar a la de la etiqueta < BR > , en cuanto a que permite saltar de una línea a otra; Sin embargo, con < HR > se presentan algunas diferencias importantes: el espacio de una línea a otra es mayor y se incluye una línea visible que permite hacer mas notable el salto. El hecho de que se observe una línea separadora, ha hecho que entre los programadores de HTML sea una convención utilizar esta etiqueta cuando desean separarse dos parrafos que no son directamente dependientes el uno del otro.

COLOR, SIZE, WIDTH. Se utilizan para:

- **COLOR** Establecer un color para la línea.
- **SIZE** Se utiliza para establecer el grosor que queremos para la línea.
- **WIDTH** Indica el largo de la línea. Por defecto la línea abarca toda la pantalla, pero si no queremos que sea así, y que solo este presente en una porción de la misma, debemos utilizar esta extensión.

EVENTOS

Los eventos son ampliamente usados en documentos HTML como una forma de asociar tags HTML a scripts. De manera simple, los scripts son programas del lado cliente que cumplen una cierta tarea y los eventos son las cosas que ocurren y ejecutan los scripts (por ejemplo, el puntero del mouse pasa sobre un elemento, la página termina de cargarse, etc.).

La sintaxis usada para definir un evento es muy similar a la de los atributos. En este ejemplo se muestra al tag HTML a, con el atributo "href" y los eventos "onmouseover" y "onmouseout".

Hay que notar que las funciones "comenzar_funcion()" y "detener_funcion()" deben ser escritas en algún lenguaje del lado cliente, por ejemplo JavaScript.

CONTENIDO

El contenido de un tag es en la mayoría de los casos la parte afectada por el efecto del tag (por ejemplo, el texto mostrado en negrita para el tag HTML b), y va en medio de los tags de apertura y cierre.

Esto va en Negritas

Por su naturaleza y funcionalidad, no todos los tags tienen contenido (por ejemplo, el tag HTML img). Estos tags vacíos deben ser correctamente cerrados para hacer el documento compatible con el estándar de código XHTML. Existen dos formas de cerrar un tag vacío: la primera es utilizando un tag de cierre normal (</nombre_tag>) y la otra es usando una barra al final del tag de apertura. Observemos estos ejemplos:

- <img src="logo.jpg" onmouseover="comenzar_funcion()"></img>
-

Notar que en el segundo caso, la última barra es considerada por los navegadores antiguos como un atributo desconocido por lo cual es simplemente ignorado. Por esta razón se debe separar el último atributo de la barra.

El contenido de un tag puede ser otro tag o hasta trozos de documentos HTML, aunque no todos los tags pueden contener a otros tags y algunos de ellos pueden contener solo ciertos tags. Como regla general, elementos de línea no pueden contener elementos de bloque, elementos de bloque pueden contener elementos de línea, y elementos de bloque pueden contener elementos de bloque. Esta es una regla muy general que tiene muchas excepciones pero es suficiente para obtener una idea general.

Observa este ejemplo:

Código	Visualización
Este tag contiene un tag de línea. 	Este tag contiene un **tag de línea**. Aquí tenemos un elemento de bloque conteniendo a otro elemento de bloque que

Aquí tenemos un elemento de bloque conteniendo a otro elemento de bloque que está conteniendo otros elementos de línea.	está conteniendo otros *elementos de línea*.

Hay que recordar que para hacer el código compatible con el estándar de código XHTML debe respetarse el orden en que los tags son abiertos y cerrados (esto significa, el primer tag en abrirse es el último en cerrarse).

ANEXO IV. JAVASCRIPT

Javascript es un lenguaje de programación utilizado para crear pequeños programas encargados de realizar acciones dentro del ámbito de una página web. Con Javascript podemos crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar

estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

Entre las acciones típicas que se pueden realizar en Javascript tenemos dos vertientes. Por un lado los efectos especiales sobre páginas web, para crear contenidos dinámicos y elementos de la página que tengan movimiento, cambien de color o cualquier otro dinamismo. Por el otro, javascript nos permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que podemos crear páginas interactivas con programas como calculadoras, agendas, o tablas de cálculo.

Javascript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Toda esta potencia de Javascript se pone a disposición del programador, que se convierte en el verdadero dueño y controlador de cada cosa que ocurre en la página.

BREVE HISTORIA

JavaScript es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C.

El lenguaje fue inventado por Brendan Eich en la empresa Netscape Communications, que es la que desarrolló los primeros navegadores web comerciales. Apareció por primera vez en el producto de Netscape llamado Netscape Navigator 2.0.

Tradicionalmente, se venía utilizando en páginas web HTML, para realizar tareas y operaciones en el marco de la aplicación únicamente cliente, sin acceso a funciones del servidor. JavaScript se ejecuta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

Los autores inicialmente lo llamaron Mocha y más tarde LiveScript pero fue rebautizado como JavaScript en un anuncio conjunto entre Sun Microsystems y Netscape, el 4 de diciembre de 1995.

En 1997 los autores propusieron JavaScript para que fuera adoptado como estándar de la European Computer Manufacturers' Association ECMA, que a pesar de su nombre no es europeo sino internacional, con sede en Ginebra. En junio de 1997 fue adoptado como un estándar ECMA, con el nombre de ECMAScript. Poco después también lo fue como un estándar ISO.

JScript es la implementación de ECMAScript de Microsoft, muy similar al JavaScript de Netscape, pero con ciertas diferencias en el modelo de objetos del navegador que hacen a ambas versiones con frecuencia incompatibles.

Para evitar estas incompatibilidades, el World Wide Web Consortium diseñó el estándar Document Object Model (DOM, ó Modelo de Objetos del Documento), que incorporan Konqueror, las versiones 6 de Internet Explorer y Netscape Navigator, Opera versión 7, y Mozilla desde su primera versión.

Lenguaje JavaScript

Antes que nada debemos decir que para programar en Javascript necesitamos básicamente lo mismo que para programar páginas web con HTML. Un editor de textos y un navegador compatible con Javascript. Un usuario de Windows posee de salida todo lo necesario para

poder programar en Javascript, puesto que dispone dentro de su instalación típica de sistema operativo, de un editor de textos, el Bloc de notas, y de un navegador: Internet Explorer.

Lo más importante y básico que podemos destacar en este momento es que la programación de Javascript se realiza dentro del propio documento HTML. Esto quiere decir que en la página se mezclan los dos lenguajes de programación, y para que estos dos lenguajes se puedan mezclar sin problemas se han de incluir unos delimitadores que separan las etiquetas HTML de las instrucciones Javascript. Estos delimitadores son las etiquetas <SCRIPT> y </SCRIPT>. Todo el código Javascript que pongamos en la página ha de ser introducido entre estas dos etiquetas. Existen **dos maneras de ejecutar scripts** en la página. La primera de estas maneras se trata de ejecución directa de scripts, la segunda es una ejecución como respuesta a la acción de un usuario. Veremos ahora cada una de ellas.

Ejecución directa

Es el método de ejecutar scripts más básico. En este caso se incluyen las instrucciones dentro de la etiqueta <SCRIPT>, tal como hemos comentado anteriormente. Cuando el navegador lee la página y encuentra un script va interpretando las líneas de código y las va ejecutando una después de otra. Llamamos a esta manera ejecución directa pues cuando se lee la página se ejecutan directamente los scripts.

Respuesta a un evento

Es la otra manera de ejecutar scripts, pero antes de verla debemos hablar sobre los eventos. Los eventos son acciones que realiza el usuario. Los programas como Javascript están preparados para atrapar determinadas acciones realizadas, en este caso sobre la página, y realizar acciones como respuesta. De este modo se pueden realizar programas interactivos, ya que controlamos los movimientos del usuario y respondemos a ellos. Existen muchos tipos de eventos distintos, por ejemplo la pulsación de un botón, el movimiento del ratón o la selección de texto de la página.

Las acciones que queremos realizar como respuesta a un evento se han de indicar dentro del mismo código HTML, pero en este caso se indican en atributos HTML que se colocan dentro de la etiqueta que queremos que responda a las acciones del usuario. Por ejemplo, si queríamos que un botón realizase acciones cuando se pulsase sobre el, debíamos indicarlas dentro del atributo onclick del botón como se muestra a continuación.

```
<INPUT TYPE=BUTTON VALUE=ATRÁS ONCLICK="HISTORY.GO(-1)">
```

La etiqueta <SCRIPT> tiene un atributo que sirve para indicar el lenguaje que estamos utilizando, así como la versión de este. Por ejemplo, podemos indicar que estamos programando en Javascript 1.2 o Visual Basic Script, que es otro lenguaje para programar scripts en el navegador cliente que sólo es compatible con Internet Explorer.
El atributo en cuestión es lenguaje y lo más habitual es indicar simplemente el lenguaje con el que se han programado los scripts. El lenguaje por defecto es Javascript, por lo que si no utilizamos este atributo, el navegador entenderá que el lenguaje con el que se está programando es Javascript.

```
<SCRIPT LANGUAGE=JAVASCRIPT>
```

ANEXO V.SCRIPT SQL

El script que a continuación se presenta es para la creación de las tablas una vez creada la base de datos llamada PERAJ.

```
CREATE TABLE ss_areas
(  idarea integer NOT NULL DEFAULT 0,
```

```
  nombre character varying(70) NOT NULL DEFAULT 0,
  CONSTRAINT ss_areas_pkey PRIMARY KEY (idarea) ) ;
CREATE TABLE ss_carreras
(  idcarrera integer NOT NULL DEFAULT 0,
  nombre character varying(200) NOT NULL DEFAULT 0,
  idarea integer NOT NULL DEFAULT 0,
  CONSTRAINT ss_carreras_pkey PRIMARY KEY (idcarrera) );
CREATE TABLE ss_facultades
(  idfac integer NOT NULL DEFAULT 0,
  nombre character varying(80) NOT NULL DEFAULT 0,
  CONSTRAINT ss_facultades_pkey PRIMARY KEY (idfac),
  CONSTRAINT fk_sscarreras FOREIGN KEY (carrera)
     REFERENCES ss_carreras (idcarrera) MATCH SIMPLE
     ON UPDATE NO ACTION ON DELETE NO ACTION
 );
CREATE TABLE ss_usuario
(  id_usuario serial NOT NULL,
  nombre_usuario character varying(100) NOT NULL DEFAULT 0,
  nombre_completo character varying(100) NOT NULL DEFAULT 0,
  active boolean NOT NULL DEFAULT false,
  contrasenia character varying(100) NOT NULL DEFAULT 0,
  CONSTRAINT pk_ss_usuario PRIMARY KEY (id_usuario) );
CREATE TABLE ssp_academicoalumno
(  numcuenta character varying(9) NOT NULL DEFAULT 0,
  facultad character varying(30) NOT NULL DEFAULT 0,
 carrera character varying(30) NOT NULL DEFAULT 0,
creditos character varying(20) NOT NULL DEFAULT 0,
promedio character varying(20) NOT NULL DEFAULT 0,
character varying(20) NOT NULL DEFAULT 0,
  area character varying(10) NOT NULL DEFAULT 0,
  beca character varying(40) NOT NULL DEFAULT 0,
  CONSTRAINT pk_academicoalumno PRIMARY KEY (numcuenta),
  CONSTRAINT fk_academicoalumno FOREIGN KEY (numcuenta)
     REFERENCES ssp_personalalumno (numcuenta) MATCH SIMPLE
     ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT fk_ss_facultades FOREIGN KEY (facultad)
     REFERENCES ss_facultades (idfac) MATCH SIMPLE
     ON UPDATE NO ACTION ON DELETE NO ACTION
);
CREATE TABLE ssp_academicoamigo
(  idamigo integer NOT NULL,
  escuela character varying(80),
  grado character varying(80),
  grupo character varying(80),
  promedio character varying(80),
  nom_prof character varying(80),
  mat_pref character varying(80),
  mat_pref_razon character varying(80),
  mat_nopref character varying(80),
  mat_nopref_razon character varying(80),
  CONSTRAINT pk_ssp_academicoamigo PRIMARY KEY (idamigo),
  CONSTRAINT fk_ssp_academicoamigo FOREIGN KEY (idamigo)
```

```
    REFERENCES ssp_personalamigo (idamigo) MATCH SIMPLE
    ON UPDATE CASCADE ON DELETE CASCADE );
CREATE TABLE ssp_adicionalamigo
(  idamigo integer NOT NULL,
  nombrepapa character varying(80),
  nombremama character varying(80),
  nombre_ref character varying(80),
  tel_ref character varying(80),
  parentesco character varying(80),
  rel_fam character varying(80),
  fam1 character varying(80),
  fam2 character varying(80),
  nomfam1 character varying(80),
  nomfam2 character varying(80),
  profesion character varying(80),
  tiempo_libre character varying(80),
  aspiracion character varying(80),
  nomaspiracion character varying(80),
  CONSTRAINT pk_ssp_adicionalamigo PRIMARY KEY (idamigo),
  CONSTRAINT fk_ssp_adicionalamigo FOREIGN KEY (idamigo)
    REFERENCES ssp_personalamigo (idamigo) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION );
CREATE TABLE ssp_personalalumno
(  nombre character varying(80) NOT NULL DEFAULT 0,
  ap_paterno character varying(80) NOT NULL DEFAULT 0,
  ap_materno character varying(80) NOT NULL DEFAULT 0,
  calle character varying(80) NOT NULL DEFAULT 0,
  no_int character varying(80), -- Numero interior del domicilio del alumno
  no_ext character varying(80) NOT NULL DEFAULT 0,
  colonia character varying(80) NOT NULL DEFAULT 0
  cp character varying(30) NOT NULL DEFAULT 0,
  delegacion character varying(80) NOT NULL DEFAULT 0,
  entidad character varying(80) NOT NULL DEFAULT 0,
  tel_casa character varying(40) NOT NULL DEFAULT 0,
  rfc character varying(40) NOT NULL DEFAULT 0,
  curp character varying(40), -- curp del alumno
  sexo character varying(1) NOT NULL DEFAULT 0,
  email character varying(60) NOT NULL DEFAULT 0,
  fecha_nacimiento character varying(20) NOT NULL DEFAULT 0,
  edad character varying(2) NOT NULL DEFAULT 0,
  numcuenta character varying(9) NOT NULL DEFAULT 0,
  tel_cel character varying(40),
  perfil smallint NOT NULL DEFAULT 0,
  num_asignacion integer(5) DEFAULT 0,
  CONSTRAINT pk_personalalumno PRIMARY KEY (numcuenta) );
CREATE TABLE ssp_personalamigo
( nombre character varying(80),
  ap_paterno character varying(80),
  ap_materno character varying(80),
  calle character varying(80),
  no_int character varying(80),
  no_ext character varying(80),
```

```
 colonia character varying(80),
 cp character varying(30),
 delegacion character varying(80),
 entidad character varying(80),
 tel_casa character varying(40),
 sexo character varying(1),
 fecha_nacimiento character varying(20),
 edad character varying(2),
 idamigo integer NOT NULL DEFAULT
nextval(('public.seq_idamigo_ssp_personalamigo'::text)::regclass),
 num_asignacion integer(4) DEFAULT 0,
 CONSTRAINT pk_ssp_personalamigo PRIMARY KEY (idamigo) );

Create table ssp_tutorias
(
 numcuenta character varying(9) NOT NULL DEFAULT 0,
idamigo integer NOT NULL DEFAULT,
num_asignacion integer NOT NULL DEFAULT
nextval(('public.seq_numasignacion_ssp_tutorias'::text)::regclass),
CONSTRAINT pk_ssp_tutorias PRIMARY KEY (idamigo),
CONSTRAINT pk_ssp_tutorias PRIMARY KEY (numcuenta),
 CONSTRAINT fk_ssp_tutoriasamigo FOREIGN KEY (idamigo)
    REFERENCES ssp_personalamigo (idamigo) MATCH SIMPLE
    ON UPDATE CASCADE ON DELETE CASCADE,
 CONSTRAINT fk_ssp_tutoriasalumno FOREIGN KEY (numcuenta)
    REFERENCES ssp_personalalumno (numcuenta) MATCH SIMPLE
    ON UPDATE CASCADE ON DELETE CASCADE.
);
```

BIBLIOGRAFÍA

1. *"Analisis Estructurado Moderno",Edward Yourdon*

   Primera Edición en Español, México, 1993

   Ed. Prentice Hall HispanoAmericana S.A.

2. *"Análisis y diseño de sistemas", Kendall, Kenneth E.*

   Sexta Edición, *México.* 2005

   Ed. Prentice Hall Hispanoamericana.

3. *"Beginning PHP and PostgreSQL 8 : from novice to professional" , W. Jason Gilmore and Robert H. Treat*

   Primera Edicion, Berkeley, California, 2006

   Ed. Apress.

4. *"Domine HTML y DHTML" , Charte Ojeda, Francisco*

   México, 2003

   Ed. Ra-Ma, Librería y Editorial Microinformática

5. *"Domine JavaScript", José López Quijado*

   Madrid, 2004

   Ed. Alfa-Omega.

6. "*Introducción a los sistemas de base de datos", Date C,J.*

   Septima Edición , Año 2001

   Ed. Alhambra Mexicana, S.A.

7. *"SQL : the complete reference" , James R. Groff, Paul N. Weinberg*

   Segunda Edición, Berkeley,California

   Ed. Osborne McGraw-Hill

8. *"PHP5 a través de ejemplos" , Abraham Gutiérrez Rodríguez, Ginés Bravo García.*

   Primera Edición, Paracuellos de Jarama, Madrid , 2005.

   Ed. Ra-Ma.