UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Sistema de gestión vía web para servicio social

TESIS

Que para obtener el título de

Ingeniero en Computación

P R E S E N T A

Guillermo Galán García

DIRECTORA DE TESIS

Ing. Maricela Castañeda Perdomo

Ciudad Universitaria, Cd. Mx., 2017

AGRADECIMIENTOS

A la Universidad Nacional Autónoma de México

Por haberme abierto las puertas y darme la oportunidad para realizar mis metas.

A mi padre †

Porque no existen palabras suficientes para agradecerle toda la ayuda brindada en el transcurso de mis estudios y por todo lo que hizo por mí.

A mi familia

A quienes con toda su confianza, empeño y motivación, consiguieron que pudiera terminar mis estudios. Gracias por todo.

A mis amigas

Por su apoyo en los momentos difíciles, quienes me demostraron lo valioso que resulta la constancia en toda tarea.

A mi asesora

Con agradecimiento a la Ing. Maricela Castañeda Perdomo por haberme brindado su tiempo para la realización del presente trabajo.

A mis profesores

Quienes con sus enseñanzas lograron que terminara de recorrer este gran camino.

SISTEMA DE GESTIÓN VÍA WEB PARA SERVICIO SOCIAL

ÍNDICE

INTRODUCCIÓN

Un sistema de gestión es un programa que es capaz de crear una estructura de soporte llamada framework, para la creación y administración de contenidos que en su mayoría son de aplicaciones web, hechas por los programadores. El sistema tiene la ventaja de poder manejar de manera independiente el contenido y el diseño, si el diseño se desea cambiar, no será necesario eliminar el contenido. Otra de las ventajas es que permite controlar el acceso a los usuarios del sistema, existiendo desde usuarios que puedan crear contenido y modificarlo, así como aquellos que solo podrán visualizarlo. Los sistemas de gestión siempre funcionan en el servidor web donde se encuentran alojados y se accede a ellos desde algún navegador web.

El uso de los sistemas de gestión, permite que el acceso al contenido sea visualizado por un mayor número de usuarios, y por lo tanto, la colaboración sobre un mismo trabajo también sea entre varios usuarios.

Conforme han pasado los años, el número de alumnos de la Facultad de Psicología que se han inscrito a un programa de servicio social ha ido en aumento, los medios de almacenamiento de datos con los que contaban eran poco prácticos y de difícil acceso, por lo que recientemente surgió la necesidad de contar con un sistema que sea eficiente para poder registrar a los alumnos y realizar consultas de datos específicos en el menor tiempo posible, además de fácil manejo y que permita el acceso a la información a todo el personal de la Jefatura de servicio social de la Facultad de Psicología.

Por lo tanto, la presente tesis trata sobre el proceso de desarrollo de una aplicación web con una base de datos, para la Jefatura de servicio social de la Facultad de Psicología, perteneciente a la Universidad Nacional Autónoma de México.

Esta aplicación web debe ser capaz de cumplir con las necesidades ya mencionadas para reducir el tiempo de captura y consulta de la información de los alumnos prestadores de servicio social.

Una vez que la aplicación web esté terminada, el personal de la Jefatura de servicio social podrá tener acceso a la información desde cualquier lugar y en cualquier momento siempre y cuando se cuente con una conexión a Internet.

En el resto del contenido de esta tesis, se describen a detalle cada una de las fases que conforman el desarrollo de la aplicación web y el diseño de la base de datos.

Por último, se muestran las pruebas de funcionamiento de la aplicación web.

CAPÍTULO 1. ANTECEDENTES

La Jefatura de servicio social de la Facultad de Psicología es un área encargada de llevar el registro y control de los alumnos prestadores de servicio social, con el paso del tiempo el personal encargado de la captura de información, fue dejando a un lado el formato físico como un medio de almacenamiento de información, ya que mientras pasaban los años, el número de expedientes iba aumentando.

Posteriormente se elaboró una aplicación para escritorio que contenía una base de datos llamada por el mismo personal como la "base histórica", por que contenía registros de alumnos que habían hecho su servicio social muchos años atrás, pero el problema persistió porque la aplicación estaba instalada en solo dos computadoras y el resto del personal no tenía acceso a ella, ni la consulta de la información era de manera práctica.

Para solucionar momentáneamente dicho problema, se creó un archivo en Excel que sirvió como una base de datos que se subió a Google Drive, y se compartió a todo el personal de la Jefatura, pero al ser un trabajo laborioso el hacer consultas de datos específicos, se requirió de la elaboración de un sistema al cual tuvieran acceso todo el personal de la Jefatura de servicio social y así el ingreso y consulta de los datos a la base fuera de manera rápida y práctica.

La principal labor en dicha dependencia, fue desarrollar en base a las especificaciones dadas por el jefe de la Jefatura de servicio social, y al trabajo ya hecho por antiguos prestadores de servicio social y del resto del personal, una aplicación web con una base de datos para que mediante una interfaz sencilla y de fácil manejo, se lleve el registro y control de los alumnos prestadores de servicio social, de las Instituciones que requieren los servicios de los alumnos de la Facultad de Psicología y de los programas a los que dichos alumnos pueden inscribirse, además de realizar consultas al instante de datos específicos, reducir el tiempo de labor de captura de datos y tener acceso desde cualquier dispositivo que cuente con conexión a internet, para finalmente sustituir las antiguas bases de datos que resultaron no ser eficientes.

La aplicación web que se desarrolló, es un sistema distribuido que utiliza la arquitectura cliente-servidor de tres capas, la comunicación con el servidor se hace mediante el protocolo HTTP y cuenta con un middleware que soporta consultas a la base de datos en SQL. El procesamiento de la información no se encuentra en un solo equipo, lo que comúnmente aumenta la probabilidad de pérdida de la información en caso de alguna falla irreparable como sucedía con la base de datos que el personal llamaba "base histórica".

CAPÍTULO 2. SISTEMA DE GESTIÓN VÍA WEB PARA SERVICIO SOCIAL

En el ámbito de la ingeniería de software (disciplina de la ingeniería que comprende todas las etapas de la producción de un software), al sistema que se desarrolló se le conoce como “aplicación web”, esta contiene un conjunto de funciones a las que el usuario puede acceder, ingresando al servidor web donde se encuentra alojada dicha aplicación mediante un navegador, siempre y cuando se tenga conexión a internet. Además está codificada en algún lenguaje de programación que es soportado por los navegadores web.

Todo desarrollo de software tiene una serie de actividades que se deben de seguir, estas son:

- **Especificación del software.** El(los) cliente(s) y el(los) ingeniero(s) definen las funciones que tendrá el software que se producirá y las restricciones sobre su operación.
- **Desarrollo del software.** Se realiza el diseño y programación del software.
- **Validación del software.** El software debe de ser validado para asegurar que realiza las funciones que el cliente solicitó.
- **Evolución del software.** Son todas aquellas modificaciones que se le realizan al software para adaptarlo a los cambios que solicita el cliente.

A dicha serie se le conoce como “proceso de software”. Una descripción simplificada de un proceso de software se conoce como “modelo de proceso del software”. Los modelos de software pueden incluir actividades que son parte de los procesos de software, algunos ejemplos de los tipos de modelos que se pueden producir son:

- **Modelo de flujo de trabajo.** Muestra la secuencia de actividades en el proceso, junto con sus entradas, salidas y dependencias. Las actividades en este modelo representan acciones humanas.
- **Modelo de flujo de datos o de actividad.** Representa el proceso como un conjunto de actividades, donde cada una realiza alguna transformación en los datos. La entrada del proceso la cual es la especificación, se transforma en una salida, siendo esta el diseño.
- **Modelo de rolación.** Representa los roles de las personas involucradas en el proceso del software y las actividades de las que son responsables.

La mayoría de los modelos de procesos de software se basan en uno de los siguientes tres modelos generales:

- **Modelo en cascada.** Engloba las actividades que se mencionan arriba y son representadas como fases de procesos separados, una vez que cada etapa queda definida, el desarrollo continúa con la siguiente etapa.
- **Desarrollo evolutivo.** Las actividades de especificación, desarrollo y validación se entrelazan. El sistema se desarrolla a partir de las especificaciones y se va refinando a través

de las peticiones del cliente, hasta que se produce el software que satisface las necesidades de dicho cliente.

- **Ingeniería del software basada en componentes (ISBC).** Este modelo se basa en la existencia de un número significativo de componentes reutilizables. La etapa de desarrollo consiste en la integración los componentes ya existentes en vez de desarrollarlos desde cero.

El uso del modelo en cascada, tiene muchas desventajas que hace que no sea muy útil para sistemas que requieren estar en constante mantenimiento, porque se requiere rehacer el trabajo desde la primera etapa.

Sin embargo, para el desarrollo de la aplicación web, se decidió utilizar dicho modelo por las siguientes razones:

- Las funciones principales que debe realizar el sistema, las cuales fueron establecidas en los requerimientos son inamovibles, lo que garantiza no rehacer todo el trabajo cuando se solicite realizar mantenimiento a la aplicación web.
- El mantenimiento de la aplicación web, únicamente se basa en agregar nuevas consultas que muestren la información que requiere el usuario, así como la creación de tablas adicionales donde se almacenará información que ya no es tan relevante.
- Las funciones que realiza la aplicación web no pueden irse probando, integrando y entregando al usuario individualmente, ya que las funciones dependen una de otra y no era factible usar el modelo incremental que obliga a ir poniendo en funcionamiento parte por parte.

Las etapas en las que se divide el modelo en cascada se muestran en la figura 2.1:

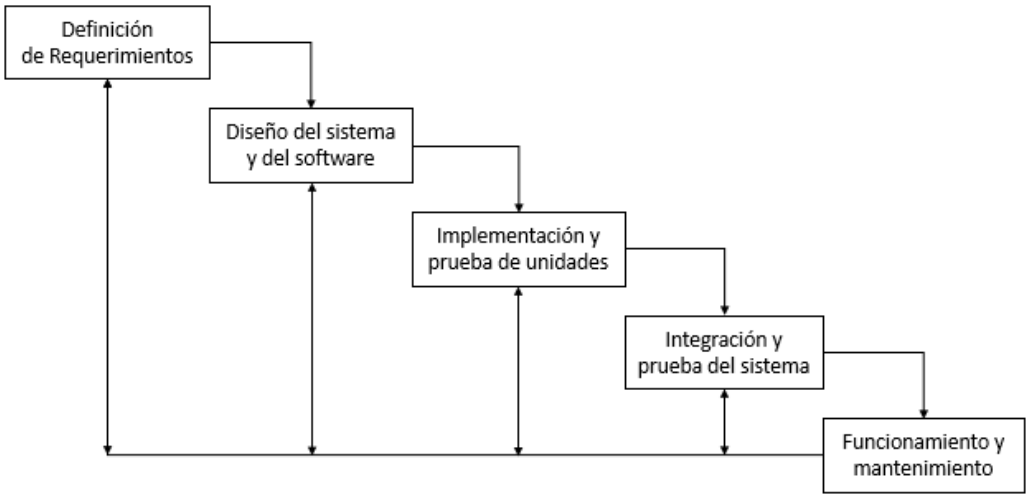


FIGURA 2.1 Diagrama de las etapas del Modelo en cascada

- **Definición de requerimientos.** Los servicios, restricciones y los objetivos con los que debe de cumplir el sistema a desarrollar, se definen a partir de consultas con los usuarios. Una vez que se definen, se convierten en las especificaciones del sistema.

- **Diseño del sistema y del software.** En esta etapa el sistema divide los requerimientos en sistemas hardware y software. Se establece la arquitectura del sistema. El diseño del software identifica y describe las abstracciones fundamentales del sistema.
- **Implementación y prueba de unidades.** El diseño del software se desarrolla como un conjunto o unidades de programa. La prueba de unidades es cuando se verifica que cada unidad cumpla con su especificación.
- **Integración y prueba de sistema.** Los programas o las unidades individuales se integran y se prueban como un sistema completo, para así garantizar que se cumplan los requerimientos. Después de que se han realizado las pruebas, el sistema es entregado al cliente.
- **Funcionamiento y mantenimiento.** El sistema se instala y es puesto en funcionamiento. El mantenimiento consiste en corregir los errores no detectados en las etapas anteriores, así como también implementar mejoras en el sistema.

Otro modelo que se hubiera elegido para el desarrollo de la aplicación web es el "desarrollo evolutivo", que se basa en la idea de desarrollar una implementación inicial, esta se expone a los comentarios de los usuarios y se va perfeccionando a través de las diferentes versiones hasta que se obtiene un sistema adecuado. Las actividades de especificación, desarrollo y validación se entrelazan. El tipo de desarrollo evolutivo que se hubiera adecuado a la forma de trabajar es el "desarrollo exploratorio", que consiste en trabajar con el cliente para explorar sus requerimientos y entregar un sistema final. El desarrollo se inicia con las partes del sistema que mejor se comprendan. El sistema va evolucionando conforme se van agregando nuevas propuestas por el cliente.

Entre las razones por las cuales el modelo mencionado hubiera sido utilizado, son:

- Cuando una de las funciones del sistema era presentada al jefe de la Jefatura, por lo regular había que modificarla o añadirle detalles adicionales posteriormente, dichos cambios en su mayoría tenían que ver con la interfaz de usuario.
- En las presentaciones del avance del sistema, surgían nuevos requerimientos que se tenían que implementar de último momento, añadiendo al sistema nuevas consultas de datos.

La gestión de proyectos de software es parte esencial de la ingeniería de software, si se lleva a cabo una buena gestión es posible garantizar el éxito del proyecto, en caso contrario, el proyecto podría fracasar. La función de los gestores es realizar la planificación y temporalización del desarrollo de los proyectos.

La planificación de proyectos consiste en identificar las actividades y entregas de un proyecto. Por lo tanto, se debe de bosquejar un plan que sirva como guía para las entregas del proyecto.

En la calendarización del proyecto se estiman el tiempo y los recursos requeridos para completar las actividades y organizarlas en una sucesión coherente. La calendarización del proyecto consiste en separar todo el trabajo del proyecto en actividades complementarias y además se debe de considerar el tiempo requerido para completar dichas actividades. Es muy común que algunas actividades se lleven a cabo en paralelo, estas actividades se deben de coordinar y organizar el trabajo para que la mano de obra se utilice de manera óptima. También se deben de evitar situaciones que provoquen que el proyecto sufra un retraso debido a que no se ha terminado una

actividad crítica. El tiempo que lleva realizar las actividades de un proyecto debe de ser por lo menos de una semana. Cuando se está realizando el proceso de la calendarización se deben de contemplar los problemas que puedan ocurrir en cada etapa del proyecto, como por ejemplo la falta de personal por enfermedad o renuncia, fallos en el hardware, retraso en adquisición de nuevo software, etc., además de los recursos con los que se cuenta.

El calendario del proyecto se representa como un conjunto de gráficos en donde se muestra la división del trabajo, las actividades y la asignación del personal.

Los gráficos de barras y las redes de actividades son notaciones gráficas que se utilizan para ilustrar la calendarización del proyecto. En los gráficos de barras se muestran quienes son los responsables de cada actividad y cuando debe de comenzar y terminar dicha actividad, mientras que en las redes de actividades se muestran las dependencias entre las diferentes actividades que conforman el proyecto.

Para representar la información del calendario del desarrollo de la aplicación web, se utilizó el gráfico de barras que suele ser llamado “diagrama de Gantt”, en el diagrama se muestra el calendario del proyecto además de las fechas iniciales y finales de las actividades.

En la figura 2.2, se muestra la calendarización que se elaboró para el desarrollo de la aplicación web.

Num	Actividades	Agosto (1 - 31)					Septiembre (1 - 30)				Octubre (1 al 31)				Noviembre (1 al 30)					Diciembre (1 al 31)				Enero (1 al 31)				Feb.
		1	2	3	4	5	1	2	3	4	1	2	3	4	1	2	3	4	5	1	2	3	4	1	2	3	4	1
1	Recopilación de requerimientos																											
2	Elección e instalación del software correspondiente																											
3	Diseño de la interfaz de usuario de la Aplicación Web																											
4	Diseño de la base de datos																											
5	Implementación del sistema de registro e inicio de sesión																											
6	Implementación de la interfaz de la Aplicación Web																											
7	Implementación de la función de insertar información																											
8	Implementación de la función de modificar información																											
9	Implementación de la función de eliminar información																											
10	Implementación de la función de buscar información																											
11	Implementación de las consultas de datos específicos																											
12	Alojamiento de la aplicación web en un hosting gratuito																											
13	Elaboración del manual técnico y manual de usuario																											
14	Capacitación del personal de la jefatura de servicio social																											

FIGURA 2.2 Calendarización del desarrollo de la Aplicación Web

El tiempo de duración de cada una de las actividades es por semana, en todas las actividades se contemplaron los posibles retrasos que pudieran existir.

2.1 REQUERIMIENTOS

La etapa de ingeniería de requerimientos es donde se comprenden y se definen los servicios que el sistema requerirá, se identifican las restricciones de funcionamiento y desarrollo del mismo.

El proceso de la ingeniería de requerimientos se divide en cuatro fases:

- **Estudio de vialidad.** Es la fase donde se estima si las necesidades del usuario se pueden satisfacer con las tecnologías que actualmente existen de software y hardware. Se analiza si el sistema es rentable, el estudio debe ser económico y rápido de elaborar.
- **Obtención y análisis de los requerimientos.** Los requerimientos del sistema se obtienen por medio de la observación de los sistemas existentes, discusiones con los usuarios potenciales y proveedores, el análisis de tareas, etc.
- **Especificación de requerimientos.** Toda la información recopilada durante el análisis es traducida a un documento que define un conjunto de requerimientos. En dicho documento se pueden incluir los requerimientos del usuario que contiene los requerimientos del cliente y del usuario final del sistema, y los requerimientos del sistema que describen de una manera detallada la funcionalidad a proporcionar.
- **Validación de los requerimientos.** Es la actividad donde se comprueba la veracidad, consistencia y completitud de los requerimientos. Se detectan y se corrigen los errores en el documento de los requerimientos.

El orden en que se llevan a cabo cada una de las fases de la ingeniería de requerimientos no es estrictamente secuencial. En la figura 2.3 se muestra el diagrama del proceso de la ingeniería de requerimientos:

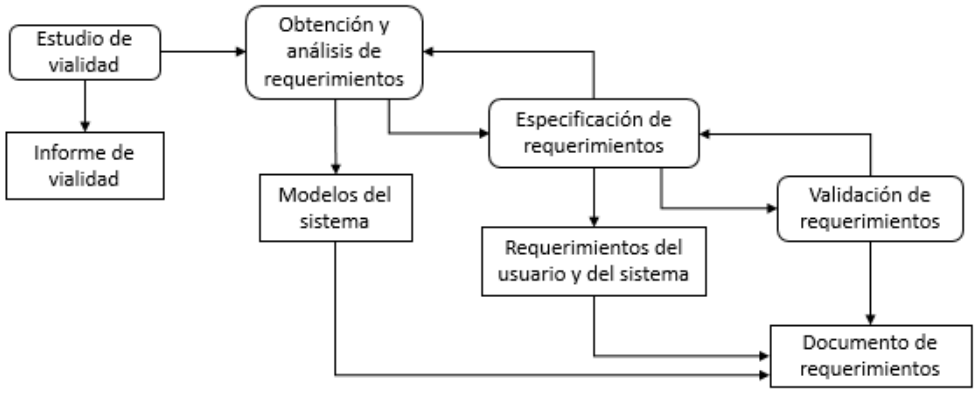


FIGURA 2.3 Diagrama del proceso de la ingeniería de requerimientos

2.1.1 ESTUDIO DE VIALIDAD

Para crear el informe del estudio de vialidad, se plantean y se responden las siguientes preguntas:

1. ¿Contribuye el sistema a los objetivos generales de la organización?

2. ¿Se puede implementar el sistema utilizando la tecnología actual y dentro de las restricciones de coste y tiempo?
3. ¿Puede integrarse el sistema con otros sistemas existentes en la organización?

Para responderlas se debe consultar con las fuentes de información, ya sea con los jefes de departamentos donde se utilizará el sistema, los ingenieros de software que están involucrados con el sistema a desarrollar, usuarios finales, etc. El lapso para concluir el estudio de vialidad es de dos o tres semanas.

La aplicación web que se desarrolló, es un sistema que cumple con las necesidades de la Jefatura de servicio social, permite el almacenamiento de la información de los programas de servicio social y de los alumnos prestadores de servicio social inscritos en algún programa a la base de datos, cuenta con una interfaz sencilla de manejar para llevar a cabo fácilmente los procesos de guardar, modificar y eliminar información, así como también de realizar las consultas de datos específicos en segundos, mientras que en las bases de datos que existían antes se invertían horas para realizar consultas, ya que el proceso era manual.

La implementación se llevó a cabo haciendo uso de la tecnología existente, y no se generó algún gasto extra porque las herramientas (software y hardware) con las que se implementó son gratuitas. El tiempo que se invirtió en el desarrollo de la aplicación web, fue dentro del rango pactado de 6 meses, dentro de ese período se implementaron las funciones esenciales que se requieren para poner en funcionamiento la aplicación web.

El desarrollo de la aplicación tuvo que ser desde cero, porque la aplicación con la que se contaba era obsoleta y la información almacenada está incompleta y a la vez innecesaria que no se usará en la nueva aplicación. La información para su desarrollo se tomó de la base de datos hecha en Excel para la creación de cada una de las tablas. La información que maneja la aplicación web única y exclusivamente es sobre los programas de servicio social y de los alumnos inscritos a dichos programas, no se puede obtener información relacionada con su trayectoria escolar, ni con otra clase de información personal, ni mucho menos se puede hacer pública dicha información a otras áreas de la misma dependencia.

2.1.2 OBTENCIÓN Y ANÁLISIS DE REQUERIMIENTOS

Tanto los ingenieros de software, como los clientes y los usuarios finales, trabajan en conjunto para determinar los servicios que proporcionará el sistema, el rendimiento, restricciones de hardware, etc. El término “stakeholder” se usa para referirse a aquellas personas que serán afectadas por el sistema de manera directa o indirecta. Entre los stakeholders se encuentran los usuarios finales del sistema y todos aquellos que pueden ser afectados por su instalación. Las actividades que se llevan a cabo en el proceso de obtención y análisis de requerimientos son las siguientes:

- **Descubrimiento de requerimientos.** En esta actividad se interactúa con los stakeholders para recopilar los requerimientos.
- **Clasificación y organización de los requerimientos.** Los requerimientos recopilados se van organizando en grupos coherentes.

- **Ordenación por prioridades y negociación de requerimientos.** Es una actividad en la que los requerimientos pueden entrar en conflicto cuando hay una gran cantidad de stakeholders. Los requerimientos se ordenan por prioridad y aquellos que están en conflicto se negocian.
- **Documentación de requerimientos.** Los requerimientos son documentados.

El personal de la Jefatura de servicio social está formado por el jefe de servicio social, secretarias y asesores de alumnos prestadores de servicio social. Cada miembro del personal de la Jefatura expuso sus necesidades (requerimientos) que deseaba satisfacer con la creación de la aplicación web. Los requerimientos recopilados según su prioridad, se enlistan enseguida:

1. Sistema de registro de usuarios para llevar un control del personal que hace uso de la aplicación web.
2. Almacenar información de los programas de servicio social dirigidos a los alumnos de la Facultad de Psicología.
3. Almacenar información de los supervisores de los programas de servicio social.
4. Almacenar información de los alumnos inscritos a algún programa de servicio social.
5. Almacenar información de los asesores de los alumnos prestadores de servicio social.
6. Funciones de modificar, eliminar, y buscar información en la base de datos.
7. Elaboración de constancias llamadas PRIDES para los supervisores de programas de servicio social, que se puedan obtener a partir de consultas en el menor tiempo posible.
8. Elaboración de constancias para los asesores de los alumnos prestadores de servicio social.
9. Elaboración de consultas para la obtención de datos específicos con fines estadísticos: número de alumnos escritos a algún programa de servicio social en determinado mes, número de programas de servicio social que ofrece la Facultad de Psicología, número de alumnos por sistema al que están sujetos, ya sea Escolarizado o SUA, etc.
10. Informar mediante alertas, de los problemas que surjan en los programas de servicio social.
11. Tomar la información de las bases antiguas para facilitar la captura de datos.

2.1.3 ESPECIFICACIÓN DE REQUERIMIENTOS

Una vez que se han obtenido los requerimientos, se agrupan en requerimientos del usuario y requerimientos del sistema.

- **Requerimientos del usuario.** Son todas aquellas declaraciones en lenguaje natural y en diagramas, de los servicios que el sistema proporcionará junto con sus respectivas restricciones.
- **Requerimientos del sistema.** En los requerimientos del sistema se deben de definir exactamente qué es lo que se va a implementar.

2.1.3.1 REQUERIMIENTOS DEL USUARIO

Todos los requerimientos que se fueron recabando del personal, se convierten en los requerimientos funcionales del usuario.

Los requerimientos funcionales, son los servicios que debe de proporcionar el sistema donde se especifique su comportamiento ante entradas particulares y situaciones particulares.

Los requerimientos funcionales del usuario se mencionan enseguida:

- Todo usuario que desee ingresar al sistema, se le pedirá realizar su registro previamente, una vez hecho podrá ingresar cuando quiera ingresando su usuario y contraseña.
- El usuario podrá ingresar la información a la base de datos de cada uno de los programas de servicio social vigentes, dependencias donde se puede realizar el servicio social, los supervisores de cada programa de servicio social y de los asesores de los alumnos.
- Cuando un alumno solicite iniciar su servicio social, ciertos datos personales serán almacenados a la base de datos, una vez hecho este procedimiento, el alumno podrá ser inscrito al programa de servicio social de su preferencia, así como también se le asignará un asesor.
- Para cualquier modificación de los datos que se hayan almacenado ya sea por corrección de errores o por actualización, se puede hacer uso del buscador para encontrar un registro en particular de la base de datos y realizar las modificaciones deseadas.
- Las secretarias podrán crear constancias llamadas PRIDES para los supervisores de programas de servicio social a partir de información que consultarán de la base de datos, realizando una búsqueda por clave del programa supervisado, el sistema mostrará todos aquellos alumnos que fueron inscritos a dicho programa del cual se muestra el nombre del programa, así como los datos de inicio y finalización del servicio social, nombre, número de cuenta y número de expediente del alumno.
- Las secretarias podrán también generar constancias a los asesores de los alumnos, realizando una búsqueda por nombre del asesor, el sistema mostrará los mismos datos de los alumnos asesorados tal y como se mencionó en el punto anterior, agregando el nombre y clave del programa de servicio social al que cada alumno se inscribió.
- El sistema cuenta con la función de poder realizar consultas para obtener información particular de la base de datos, uno de los ejemplos es la búsqueda por mes de captura, al ingresar uno de los meses, se mostrarán los datos de los alumnos que fueron registrados en dicho mes.

2.1.3.2 REQUERIMIENTOS DEL SISTEMA

Los requerimientos del sistema se dividieron en requerimientos funcionales y requerimientos no funcionales.

2.1.3.2.1 REQUERIMIENTOS FUNCIONALES DEL SISTEMA

Los requerimientos funcionales del sistema que se enlistan enseguida, explican a detalle el funcionamiento de cada una de los servicios que proporciona el sistema:

- Para ingresar a la aplicación web que se desarrolló, se debe de contar con un equipo de cómputo o incluso un smartphone o tablet, siempre y cuando se tenga conexión a internet.

Se recomienda hacer uso del navegador Google Chrome para una mejor visualización de la aplicación.

- Cuando el usuario desee almacenar información, se le solicitará llenar un formulario donde ingresará la información en los campos correspondientes, al guardarla, esta permanecerá en la base de datos.
- Para cualquier modificación o consulta de un registro en particular, hará uso del buscador y visualizará un formulario con la información deseada.
- Para todas aquellas consultas particulares y las constancias para los supervisores y los asesores, se visualizará una tabla que mostrará la información deseada, la consulta solo tomará unos segundos realizarla.

2.1.3.2.2 REQUERIMIENTOS NO FUNCIONALES DEL SISTEMA

Los requerimientos no funcionales son por lo regular, aquellos requerimientos que no hacen referencia a las funciones que realiza el sistema.

Los requerimientos que se enlistan enseguida, son conocidos como requerimientos del producto, que se refieren a las especificaciones de cómo se implementará el sistema y los requerimientos para que sea funcional. Otro tipo de requerimientos no funcionales son los requerimientos externos, estos requerimientos se derivan de los factores externos al sistema y su proceso de desarrollo. Estos incluyen, los requerimientos de interoperabilidad que definen la manera en que el sistema interactúa con sistemas de otras organizaciones, los requerimientos legislativos que deben de asegurar que el sistema funcione dentro de la ley, y por último los requerimientos étnicos, los cuales deben de asegurar que será aceptado por los usuarios y por el público en general:

- La aplicación web se implementó usando el IDE “Visual Studio”, los lenguajes HTML y CSS, así como el lenguaje de programación C# y la arquitectura de “N Capas”.
- El manejador de base de datos que se usó, fue SQL SERVER mediante la interfaz gráfica SQL Server Management Studio.
- La aplicación web para la realización de pruebas, se alojó en un hosting gratuito del proveedor SOMEE.
- La información almacenada en la base de datos de la aplicación web, no se proporciona a personal externo a la Jefatura de servicio social, ni comparte información con algún otro sistema dentro de la institución.

2.1.4 VERIFICACIÓN Y VALIDACIÓN DE REQUERIMIENTOS

Todos los requerimientos que se fueron recopilando anteriormente deben de definir el sistema que el cliente realmente necesita. Es una fase muy importante para identificar los errores desde un principio, y así evitar el tener que rehacer el trabajo al identificarlos durante el desarrollo o incluso una vez que se ha puesto en funcionamiento el sistema. Si surge un cambio en los requerimientos, el diseño y la implementación del sistema cambiarán, además se tendrá que probar nuevamente.

Durante el proceso de la validación de los requerimientos, se deben llevar a cabo verificaciones sobre requerimientos en el documento de requerimientos. Las verificaciones comprenden:

- **Verificaciones de validez.** Se verifica si el sistema que el cliente solicita realiza todas las funciones que necesita. Mediante el análisis y razonamiento se puede identificar si se requieren funciones adicionales.
- **Verificación de consistencia.** Los requerimientos recopilados no deben de contradecirse. No debe haber restricciones o descripciones contradictorias de la misma función del sistema.
- **Verificación de completitud.** El documento de los requerimientos debe de incluir todos los requerimientos dados por el usuario.
- **Verificaciones de realismo.** Los requerimientos deben de ser verificados para que se pueda garantizar su implementación de acuerdo a la tecnología existente.
- **Verificabilidad.** Los requerimientos del sistema siempre deben de redactarse en un documento para ser verificados y así reducir las discusiones que puedan suceder con el cliente.

Las técnicas de validación que existen son:

- **Revisiones de requerimientos.** Los requerimientos son analizados sistemáticamente por un equipo de revisores.
- **Construcción de prototipos.** Se muestra un modelo ejecutable del sistema a los usuarios finales y a los clientes. Se puede experimentar con el modelo para ver si cumple con las necesidades reales.
- **Generación de casos de prueba.** Los requerimientos deben de poder probarse. Si una prueba es difícil o imposible de diseñar, puede significar que los requerimientos serán difíciles de implementar y deberán ser considerados nuevamente.

Después de que se realizó una revisión de cada una de los requerimientos, se identificó que algunos no podía implementarse, uno de los primeros requerimientos que no podía llevarse a cabo, era por falta de información esencial para hacer la consulta deseada, también hubo una inconsistencia porque el jefe deseaba que la información que ya existía en la base de datos antigua, fuera tomada tal cual y pudiera ingresarse de manera casi automática a la nueva base de datos, esto es imposible porque la información ya existente en la base hecha en Excel no estaba bien organizada y hacía falta agregar campos esenciales que se iban a agregar en la nueva base de datos. También se realizó una verificación de realismo, el jefe quería que se implementara una especie de semáforo que mostrara una señal de alerta cuando existieran problemas en algún programa de servicio social, el requerimiento se salía del objetivo principal del sistema, la función principal es poder almacenar información, pero no mostrar alertas ante ciertos comentarios ya que eso no es posible implementar en un sistema como el solicitado, que además entra en contradicción ante el resto de requerimientos, también surgió otro problema cuando se pretendía que el sistema generara automáticamente la fecha de término de servicio social, omitiendo fines de semana, vacaciones o días festivos, fue un requerimiento imposible de realizar ya que no se puede implementar dicha función en la que el sistema sea capaz de determinar días o fechas ante tantas restricciones. El resto de requerimientos que incluyen el almacenar, buscar, modificar y eliminar información, así como los de las consultas de datos específicos, y el sistema de registro de usuarios, fueron verificados y posteriormente validados junto con el jefe de la Jefatura.

2.2 DISEÑO DEL SOFTWARE

Es una descripción de la estructura del software que se va a implementar, los datos del sistema, las interfaces entre los componentes y a veces de los algoritmos que se van a utilizar. Los diseñadores no llegan a un diseño detallado al momento, sino que se desarrolla de manera iterativa a través de todos los cambios que se van aplicando. Este proceso requiere formalidad y detalle. El proceso de diseño puede implicar el desarrollo de varios modelos del sistema con diferentes niveles de abstracción.

Las actividades del diseño de software son:

- **Diseño arquitectónico.** Los subsistemas que conforman el sistema y sus relaciones se identifican y documentan.
- **Especificación abstracta.** Para cada subsistema se produce una especificación donde se indiquen los servicios y las restricciones bajo las cuales debe funcionar.
- **Diseño de la interfaz.** Se diseña y se documenta la interfaz de cada subsistema.
- **Diseño de los componentes.** Se asignan servicios a los componentes y se diseñan las interfaces.
- **Diseño de la estructura de datos.** Se diseña y se especifica la estructura de datos utilizada para la implementación del sistema.
- **Diseño de algoritmos.** Se diseñan a detalle y se especifican los algoritmos utilizados para proporcionar los servicios.

El diagrama de la figura 2.4, muestra cada una de las actividades del diseño del software:

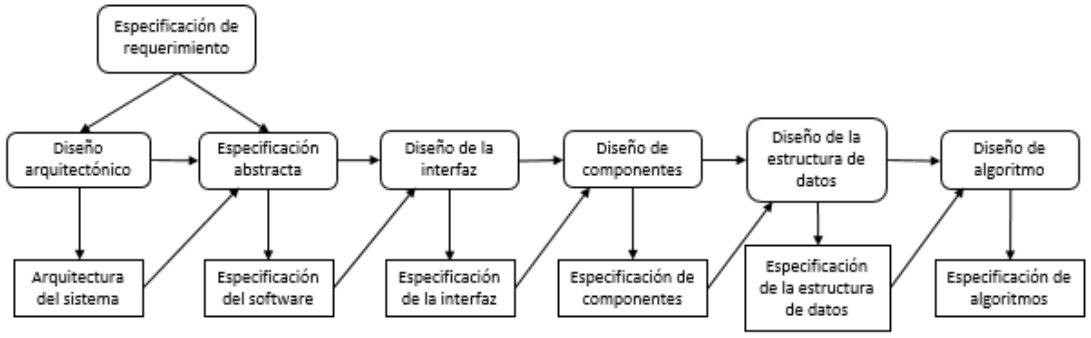


FIGURA 2.4 Diagrama de las actividades del diseño de software

Del modelo general mencionado anteriormente se pueden hacer algunas adaptaciones, las adaptaciones posibles son las siguientes:

- Las dos últimas actividades pueden ser pospuestas hasta la etapa de implementación.
- Si se utiliza un enfoque exploratorio de diseño, las interfaces del sistema se pueden diseñar después de que se especifiquen las estructuras de datos.
- La actividad de la especificación abstracta se puede omitir, aunque es una actividad fundamental de sistemas críticos.

2.2.1 DISEÑO ARQUITECTÓNICO

Es el proceso de diseño inicial en el que se identifican los subsistemas y se establece un marco para el control y comunicación de los subsistemas. Como resultado, se obtiene una descripción de la arquitectura del software.

Algunas de las ventajas que tiene el realizar y documentar el diseño arquitectónico son:

- **Comunicación de stakeholders.** La arquitectura constituye una presentación del sistema que puede usarse como punto de discusión por varios stakeholders.
- **Análisis del sistema.** Las decisiones que se tomen sobre el diseño arquitectónico tiene efecto sobre si el sistema puede cumplir con los requerimientos críticos como el rendimiento, fiabilidad y mantenibilidad.
- **Reutilización a gran escala.** La arquitectura del sistema llega a ser la misma para otros sistemas con requerimientos similares, por lo tanto, puede soportar reutilización de software a gran escala.

El realizar el diseño arquitectónico, puede servir para negociar los requerimientos del sistema y una forma de estructurar las discusiones con los clientes, desarrolladores y gestores.

Para los arquitectos del sistema se tienen que tomar varias decisiones que afecten al sistema y al proceso de su desarrollo. En base a la experiencia y conocimiento, los arquitectos del sistema tienen que responder las siguientes preguntas fundamentales:

1. ¿Existe una arquitectura de aplicación genérica que pueda actuar como una plantilla para el sistema que se están diseñando?
2. ¿Cómo se distribuirá el sistema entre varios procesadores?
3. ¿Qué estilo o estilos arquitectónicos son apropiados para el sistema?
4. ¿Cuál será la aproximación fundamental utilizada para estructurar el sistema?
5. ¿Cómo se descompondrán en módulos las unidades estructurales del sistema?
6. ¿Qué estrategia se usará para controlar el funcionamiento de las unidades del sistema?
7. ¿Cómo se evaluará el diseño arquitectónico?
8. ¿Cómo debería documentarse la arquitectura del sistema?

Tres tipos de organización de sistemas que se usan ampliamente, son:

- **Modelo repositorio.** Es un modelo utilizado para aplicaciones en las que los datos son generados por un subsistema y son usados por otro. Los subsistemas que forman un sistema, deben intercambiar información para que puedan trabajar conjuntamente de forma efectiva.
- **Modelo cliente-servidor.** Es una arquitectura distribuida, donde se puede hacer un uso efectivo de los sistemas en red con muchos procesadores distribuidos. Es fácil añadir un nuevo servidor e integrarlo con el resto del sistema o actualizar los servidores de forma transparente sin afectar al resto del sistema.
- **Modelo de capas.** El sistema es organizado en capas, donde cada una proporciona un conjunto de servicios. Cada capa puede pensarse como una máquina abstracta cuyo lenguaje máquina se define por los servicios proporcionados por la capa.

Ante la necesidad de que al sistema se pudiera acceder desde cualquier parte y en cualquier momento, el modelo que se eligió para satisfacer las necesidades del personal de servicio social fue el modelo arquitectónico cliente-servidor.

2.2.1.1 MODELO CLIENTE-SERVIDOR

Es un modelo en el que el sistema se organiza como un conjunto de servicios y servidores asociados, más los clientes que acceden y usan los servicios. Los componentes del modelo son:

- Conjunto de servidores que ofrecen servicios a otros subsistemas.
- Conjunto de clientes que llaman a los servicios que los servidores ofrecen.
- La red que permite a los clientes acceder a los servicios.

Los clientes pueden conocer los nombres de los servidores disponibles y los servicios que proporcionan. Los clientes acceden a los servicios que proporciona un servidor a través de llamadas a procedimientos remotos, usando un protocolo de petición-respuesta tal como el protocolo http usado en la WWW (World Wide Web). El cliente hace una petición al servidor y este le da una respuesta al cliente.

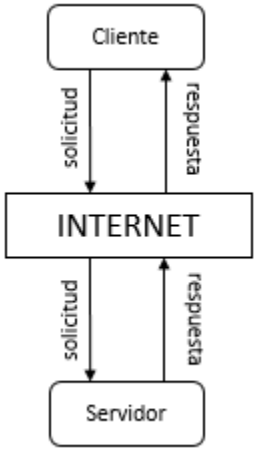


FIGURA 2.5 Modelo Cliente-Servidor

La aplicación web para la etapa de pruebas, se colocó en un hosting gratuito que provee SOMEE. Del mismo lado del servidor, se encuentra la base de datos que gestionará la información que irá almacenando el cliente (usuario).

Adquirir un plan gratuito para alojar la aplicación web con el proveedor SOMEE, cuenta con ventajas y desventajas que deben de tomarse en cuenta y se mencionan enseguida:

Ventajas:

- Capacidad de almacenamiento de 150 MB y para la base de datos 15 MB.
- No está permitido que el sitio creado sea con fines de piratería informática.

- Servicio de soporte para problemas relaciones con el acceso a la cuenta las 24 horas del día, los 7 días de la semana.
- Acceso inmediato al hosting una vez que se ha realizado el registro.
- Soporta almacenamiento de aplicaciones ASP.NET, PHP, entre otras.

Desventajas:

- Publicidad forzada que impide visualizar correctamente el sitio.
- Si no se ingresa al sitio en un lapso de 30 días, el sitio automáticamente será eliminado.
- Problemas relacionados con el hosting gratuito no son atendidos, únicamente se brinda asistencia cuando se ha contratado un plan de paga.

2.2.2 ESPECIFICACIÓN DEL SOFTWARE

El diagrama mostrado en la figura 2.5 es la arquitectura de la aplicación web, a continuación se explican las funciones de cada uno de los componentes de la arquitectura.

Servidor web. Es un equipo (físico o virtual) que está conectado a Internet para ofrecer diversos recursos. El servidor web es quien procesa la aplicación realizando conexiones bidireccionales o unidireccionales con el cliente, generando una respuesta. El protocolo que es usado para este tipo de comunicaciones es el protocolo HTTP que pertenece a la capa de aplicación del modelo OSI.

Hosting. Es un servicio que se proporciona a los usuarios para almacenar información como imágenes, video, o cualquier otro tipo, al cual se accede vía web. El hosting es un espacio de alojamiento en el servidor. En un servidor pueden estar alojados varios hosting, así que llega a suceder que en una misma dirección IP del servidor se encuentren diferentes hosting. El uso de los hosting puede ahorrar las tareas de mantenimiento técnico que son realizadas por la empresa con quien se contrató el servicio.

Base de datos. Es una colección de información, organizada y clasificada que se relaciona entre sí y que pertenece a un mismo contexto. El Sistema Manejador de Bases de Datos (SMBD) es quien permite interactuar con la base de datos, ofreciendo:

1. **Seguridad.** Protección de la base de datos ante accesos no autorizados.
2. **Concurrencia.** Permite la ejecución de los procesos en paralelo, accediendo a la información compartida.
3. **Integridad.** Corrección y coherencia de los datos almacenados.

Algunas de las ventajas del SMBD son:

- Control de redundancia de datos
- Coherencia de datos
- Mejor accesibilidad de los datos y mayor capacidad de respuesta
- Mantenimiento simplificado
- Simplifica la comunicación
- Permite la seguridad
- Garantiza la integridad

Entre las desventajas del SMBD se encuentran:

- Complejidad
- Tamaño
- Costo del SMBD
- Costo del hardware adicional
- Mayor impacto de los fallos

Lenguaje SQL. El Lenguaje de Consulta Estructurado (Structured Query Language, SQL por sus siglas en inglés) es el lenguaje estándar de las bases de datos relacionales que se utiliza para definir, gestionar y manipular la información contenida en la base de datos.

El Lenguaje de Definición de Datos (Data Definition Language, DDL por sus siglas en inglés) es el que permite la creación, eliminación y modificación de un objeto de la base de datos como son las tablas, vistas, usuarios, etc.

El Lenguaje de Manipulación de Datos (Data Manipulation Language, DML por sus siglas en inglés) permite acceder, agregar, modificar o eliminar datos de las tablas que se han creado.

Para el desarrollo de la aplicación web se utilizaron los siguientes lenguajes:

Lenguaje de Programación C#. Es un lenguaje de programación diseñado por Microsoft para su plataforma .NET, aunque existen otros lenguajes para dicha plataforma, C# es diseñado exclusivamente para ser utilizado en ella. La sintaxis y la estructura de C# es muy similar a lenguajes como C++ y Java. Algunas de las características del lenguaje de programación C# son:

- **Sencillez.** Elimina muchos elementos que otros lenguajes de programación incluyen en su código, como los ficheros adicionales (ficheros de cabecera o ficheros IDL), el tamaño de los tipos de datos básicos es fijo e independiente del compilador, no se incluyen elementos poco útiles como macros, herencia múltiple o el uso de un operador diferente del punto.
- **Orientado a objetos.** C# es un lenguaje de programación de propósito general, por lo que es orientado a objetos. Es un lenguaje puro que no admite funciones ni variables globales, todo el código y datos se definen dentro de las definiciones de tipos de datos, lo que evita problemas por conflictos de nombres y facilita la legibilidad del código.
- **Orientación a componentes.** C# define cómodamente propiedades, eventos o atributos. Incluye elementos propios del diseño de componentes.
- **Gestión automática de memoria.** No es necesario incluir instrucciones de destrucción de objetos, C# cuenta con un recolector de basura del CLR y también proporciona un mecanismo de liberación de recursos determinista a través de la instrucción using.
- **Seguridad de tipos.** Se asegura que los accesos a tipos de datos siempre se realicen correctamente, así que evita que ocurran errores difíciles de detectar por acceso a memoria no perteneciente a ningún objeto. Las medidas que se toman son: únicamente se admiten conversiones entre tipos compatibles, no se pueden usar variables no inicializadas, se comprueba que todo acceso a los elementos de una tabla se realice con índices que se encuentren dentro del rango de la misma y se controla la producción de desbordamientos en operaciones aritméticas.

Lenguaje CSS (en inglés, Cascading Style Sheets). Es un lenguaje de hoja de estilos que se usa para definir mediante una sintaxis, la forma de presentación que se aplicará ya sea a un sitio web, un documento HTML, solo una porción de documento o a una etiqueta. Entre los elementos que define se encuentran: color, tamaño y tipo de letra, separación horizontal y vertical entre elementos, posición de cada elemento en la página, etc.

Lenguaje HTML (en inglés, HiperText Markup Language). Es el lenguaje que se utiliza para crear las páginas web. El usuario puede visualizar las páginas web a través de un navegador. HTML es un lenguaje interpretado. Convierte un archivo de texto en una página web con diferentes tipos y tamaños de letra, imágenes, animaciones, formularios, etc. Una de las ventajas que tiene el lenguaje HTML es que no se requiere un programa especial para crear una página web, lo único que se necesita es un editor de texto.

En el lado del cliente se encuentran todos aquellos usuarios que harán uso de los servicios que proporciona la aplicación web.

Cliente. El cliente es quien realiza peticiones al servidor, quien le da una respuesta a través de un navegador. La interacción entre el servidor y el cliente es mediante una interfaz gráfica de usuario.

2.2.3 DISEÑO DE LA INTERFAZ

El diseño de la interfaz es la parte fundamental del diseño general del software, para que alcance su máximo potencial, es esencial que la interfaz de usuario sea diseñada para ajustarse a las habilidades, experiencia y expectativas de los usuarios finales. Los problemas que suelen surgir, entre ellos los llamados “errores de usuarios”, son causados por el hecho de que la interfaz de usuario no considera las habilidades de los usuarios y su entorno de trabajo. El hacer un mal diseño de una interfaz de usuario, provoca que los usuarios no puedan acceder a ciertas funciones del sistema, se cometan errores y se les dificultará manejar el sistema en vez de ser una ayuda para facilitarles el trabajo.

Cuando se toman las decisiones para el diseño de la interfaz de usuario, se deben de contemplar las aptitudes de los usuarios que utilizarán el sistema. Algunos factores que se deben de tomar en cuenta son:

- Las personas tienen una memoria limitada a corto plazo, si a los usuarios se les presenta demasiada información al mismo tiempo, es muy probable que no la puedan asimilar.
- Todas las personas cometen errores cuando manejan grandes cantidades de información o están estresados. Cuando se cometen errores durante el manejo del sistema y este muestra un mensaje de aviso o alarma, el estrés en los usuarios aumenta, incrementando la posibilidad de que cometan errores.
- Las personas poseen diferentes capacidades físicas, por lo tanto las cosas no se deben de diseñar para nuestras propias capacidades y suponer que los usuarios serán capaces de adaptarse.

- Tenemos diferentes preferencias de interacción, algunos les gusta trabajar con imágenes, otros con solo texto. Así como también existen personas que les gusta la manipulación directa y a otros la interacción basada en emitir comandos al sistema.

Los factores anteriores son la base para los principios de diseño que se mencionan en la figura 2.6, y se aplican a todos los diseños de interfaces de usuarios.

PRINCIPIO	DESCRIPCIÓN
Familiaridad de usuario	La interfaz debe de utilizar términos y conceptos obtenidos de la experiencia de las personas que más utilizan el sistema.
Uniformidad	Siempre que sea posible, la interfaz debe ser uniforme en el sentido de que las operaciones comparables se activen de la misma forma.
Mínima sorpresa	El comportamiento del sistema no debe provocar sorpresa a los usuarios.
Recuperabilidad	La interfaz debe incluir mecanismos para permitir a los usuarios recuperarse de los errores.
Guía de usuario	Cuando ocurran errores, la interfaz debe proporcionar retroalimentación significativa y características de ayuda sensible al contexto.
Diversidad de usuarios	La interfaz debe proporcionar características de interacción apropiadas para los diferentes tipos de usuarios del sistema

FIGURA 2.6 Principios de diseño de interfaces de usuario

Un diseñador de la interfaz de usuario debe de plantearse dos preguntas claves:

1. ¿Cómo debe interactuar el usuario con el sistema?
2. ¿Cómo se debe de presentar la información del sistema al usuario?

Una buena interfaz de usuario debe de integrar la interacción del usuario y la presentación de la información, aunque es difícil debido a que los diseñadores tienen que encontrar un término medio entre los estilos más adecuados de interacción y la presentación, la formación y la experiencia de los usuarios del sistema.

2.2.3.1 INTERACCIÓN DEL USUARIO

La interacción del usuario es emitir comandos y datos asociados al sistema. Existen cinco estilos principales de interacción:

- **Manipulación directa.** El usuario interactúa directamente con los objetos de la pantalla. Este tipo de manipulación implica un dispositivo apuntador, ya sea un mouse, un lápiz óptico, una pantalla táctil o un dedo que indique el objeto a manipular y la acción. Su implementación es difícil, pero su interacción es rápida e intuitiva. Su aplicación es en videojuegos y en sistemas CAD.
- **Selección de menús.** El usuario selecciona un comando de una lista de posibilidades llamada menú. También puede seleccionar otro objeto de la pantalla por manipulación directa y el comando actúa sobre él. Disminuye los errores del usuario, pero es un estilo lento para los

usuarios experimentados, y además es complejo si se manejan demasiadas opciones en un menú. Se aplica en la mayoría de sistemas de propósito general.

- **Rellenado de formularios.** El usuario llena los campos de un formulario. Algunos de los campos del formulario tienen un menú asociado, además siempre los formularios deben de tener un botón de acción, si se oprime, se realiza una determinada acción. La inserción de datos al formulario es un proceso sencillo, pero ocupa mucho espacio en la pantalla. Pueden surgir problemas cuando las opciones del usuario no se ajustan a los campos del formulario. El rellenado de formularios se utiliza en procesamiento de préstamos personales.
- **Lenguaje de comandos.** El usuario emite un comando y los parámetros asociados para indicar al sistema la acción que se desea llevar a cabo. Es flexible, pero difícil de aprender. Se aplica en sistemas operativos, sistemas de comandos y control.
- **Lenguaje natural.** El usuario emite un comando en lenguaje natural. El lenguaje natural se analiza y se traduce a comandos del sistema. Es un estilo accesible a usuarios casuales, pero se teclea más y los sistemas de comprensión de lenguaje natural no son fiables. Es un estilo que se aplica en sistemas de recuperación de información.

Los estilos de interacción tienen la facilidad de que se pueden utilizar varios estilos en la misma aplicación.

2.2.3.2 PRESENTACIÓN DE LA INFORMACIÓN

Todos los sistemas interactivos tienen que proporcionar alguna forma de presentar la información a los usuarios. La presentación puede ser de manera directa o de manera gráfica, en la figura 2.7, se muestra un ejemplo.

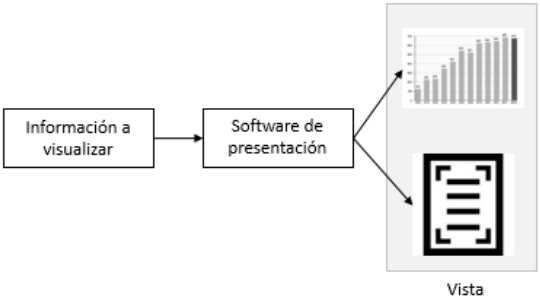


FIGURA 2.7 Presentación de la información

Para poder encontrar la mejor presentación de la información, se necesita conocer a los usuarios de la información y conocer como utilizarán el sistema. Cuando se decide cómo se representará la información, se deben de tener presentes las siguientes preguntas:

1. ¿El usuario está interesado en información precisa o en las relaciones entre valores de los datos?
2. ¿Con qué frecuencia cambian los valores de la información?, ¿Se indicarán de forma inmediata al usuario los cambios en un valor?

3. ¿El usuario debe llevar a cabo alguna acción en respuesta a los cambios de la información?
4. ¿El usuario necesita interactuar con la información visualizada a través de una interfaz de manipulación directa?
5. ¿La información que se va a visualizar es textual o numérica?, ¿Son importantes los valores relativos de los elementos de la información?

Además de la forma en que se desea presentar la información, también se debe de pensar en el color que tendrá la interfaz, los colores pueden mejorar las interfaces de usuario ayudando a los usuarios a comprender y mejorar la complejidad. En una interfaz de usuario se debe de limitar el número de colores a utilizar y ser conservador en la forma de utilizarlos, se recomienda usar un cambio de color para mostrar un cambio en el estado del sistema y que sea de manera uniforme y consciente.

Los errores, son mensajes que el sistema utiliza para comunicarse con los usuarios. Los factores de diseño que se deben de tomar en cuenta en la redacción de errores, son los que se muestran en la figura 2.8:

FACTOR	DESCRIPCIÓN
Contexto	Los mensajes generados por el sistema, deben reflejar el contexto actual del usuario. El sistema debe ser consciente de lo que está haciendo el usuario y generar mensajes relacionados con la actividad actual.
Experiencia	Se deben de presentar mensajes largos y significativos, así como cortos y concisos, de acuerdo al tipo de usuarios del sistema.
Nivel de habilidad	Los mensajes se deben adaptar a las habilidades del usuario, así como a su experiencia.
Estilo	Los mensajes siempre deben de ser positivos. No deben de ser insultantes o graciosos.
Cultura	El diseñador de mensajes debe estar familiarizado con la cultura del país donde se vende el sistema.

FIGURA 2.8 Factores de diseño en la redacción de mensajes

2.2.3.3 PROCESO DE DISEÑO DE LA INTERFAZ DE USUARIO

El proceso de diseño de la interfaz de usuario es donde los usuarios deben de interactuar con los diseñadores y los prototipos de la interfaz, para tomar una decisión de las características, apariencia y funcionamiento de la interfaz de usuario del sistema. Las actividades esenciales del proceso son:

- **Análisis del usuario.** Se desarrolla una comprensión de las tareas que el usuario realiza, su entorno de trabajo, de los demás sistemas que utiliza, etc.
- **Prototipo del sistema.** El diseño y desarrollo de la interfaz es un proceso iterativo. Se diseñan varios prototipos y se muestran a los usuarios para que ellos elijan el adecuado y guiar al diseñador en la evolución de la interfaz.
- **Evaluación de la interfaz.** Es la recopilación de las experiencias de los usuarios con la interfaz, aunque ya se haya hablado con ellos en la actividad del prototipo.

2.2.3.3.1 ANÁLISIS DE USUARIO

El análisis del usuario es una actividad crítica del diseño de la interfaz de usuario, las actividades que realiza el usuario deben de ser soportadas por el sistema. En dado caso que no se entienda lo que el usuario quiere hacer con el sistema, no será posible llevar a cabo un diseño de la interfaz de usuario. Para solucionar esto, se pueden utilizar técnicas como el análisis de tareas, estudios etnográficos, entrevistas de usuarios, observaciones, o una combinación de todas.

Uno de los retos para los ingenieros que realizan el análisis de usuario, es encontrar una forma de describir los análisis de modo que comuniquen la esencia de las tareas al resto de diseñadores y los usuarios finales. Como los usuarios también están involucrados en el diseño, se recomienda desarrollar los escenarios en lenguaje natural para describir las actividades del usuario.

La técnica de análisis que se utilizó, fue el de las entrevistas, al interactuar con todos los usuarios, se obtuvo información sobre las actividades que realizan.

El personal del servicio social realiza la tarea de captura de información de los alumnos que desean inscribirse a algún programa de servicio social, además de consultas de datos concretos de los registros de los alumnos con fines estadísticos y para expedir constancias. Antes del desarrollo de la aplicación web, el registro de los alumnos se hacía en una hoja de cálculo en Excel, donde cada columna era un campo y las filas eran los registros de cada uno de los alumnos. Ellos solicitaron que la información a insertar en la aplicación web, sea por medio de formularios, y sea visualizada de la misma manera en caso de realizar modificaciones. Para la presentación de la información de las consultas de datos específicos, se pidió que los datos se mostraran en tablas. Por último, solicitaron que para la búsqueda de un registro en concreto se haga por medio de un buscador en cada una de las siguientes opciones que conformarán el menú: alumnos, supervisores, asesores y programas de servicio social.

Para el sistema de registro de usuarios, solicitaron que únicamente se ingresaran datos concretos de los usuarios para poder acceder a la aplicación web, entre los datos están: id de usuario, nombre completo, usuario y contraseña, estos últimos datos son para iniciar sesión. Estos datos deben de ingresarse por medio de un formulario.

2.2.3.3.2 PROTOTIPO DE LA INTERFAZ DE USUARIO

De la figura 2.9 a la 2.13 se muestra el prototipo de la interfaz de usuario de la aplicación web, las cuales fueron mostradas y aprobadas por el personal de la Jefatura de servicio social, para luego ser implementada. Cada una de las figuras son las funciones que realiza el sistema.

El diseño estuvo pensado en todo momento, en las capacidades del personal, de tal manera que el manejo del sistema no fuera complicado y no surjan tantos errores durante el proceso.

Registro de usuarios.

Formulario para el registro del usuario, el usuario ingresa su información personal para poder iniciar sesión en la aplicación web.

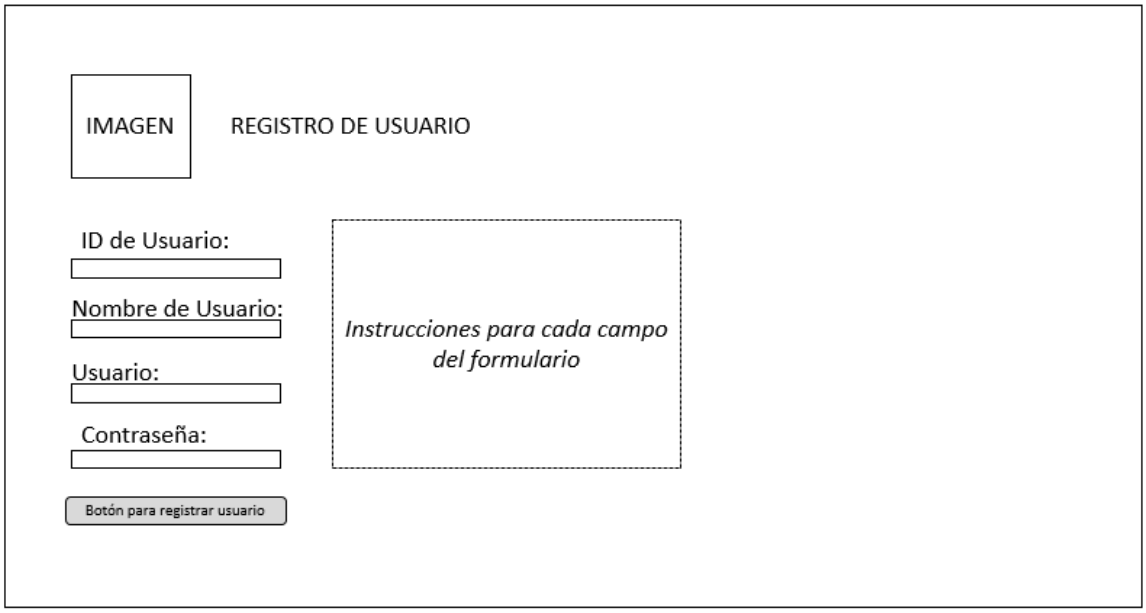


FIGURA 2.9 Prototipo del formulario de registro de usuarios

Inicio de sesión

Formulario para iniciar sesión e ingresar a la página principal de la aplicación web.

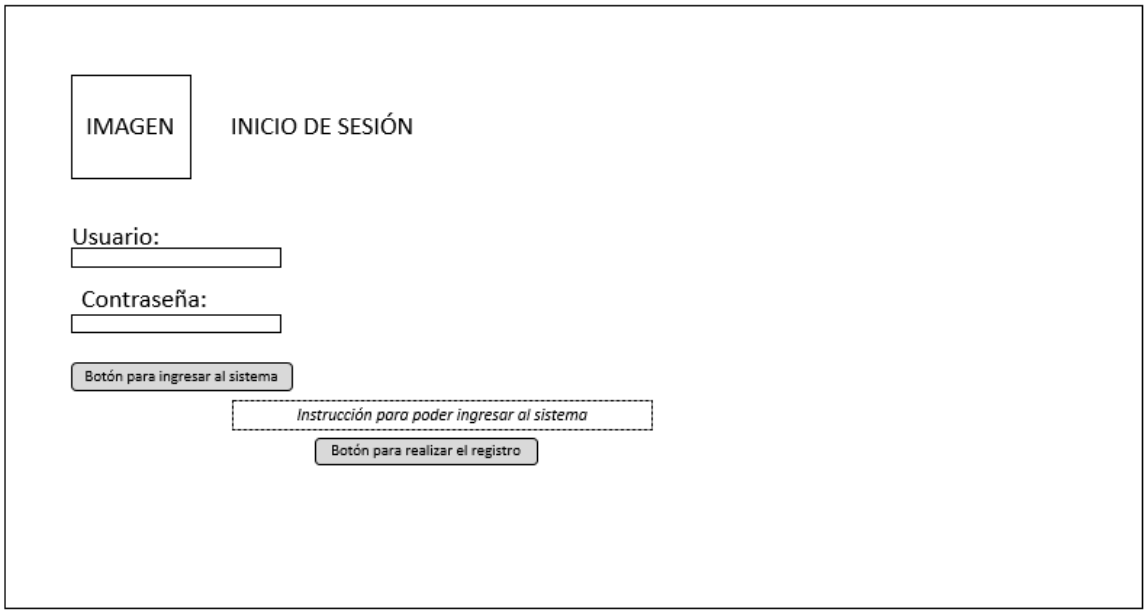


FIGURA 2.10 Prototipo del formulario de inicio de sesión

Opciones del menú

Cada una de las opciones del menú que en su mayoría son las tablas de las que está conformada la base de datos, muestra un buscador para encontrar el registro deseado, así como la opción de realizar alguna consulta particular o ingresar un nuevo registro.

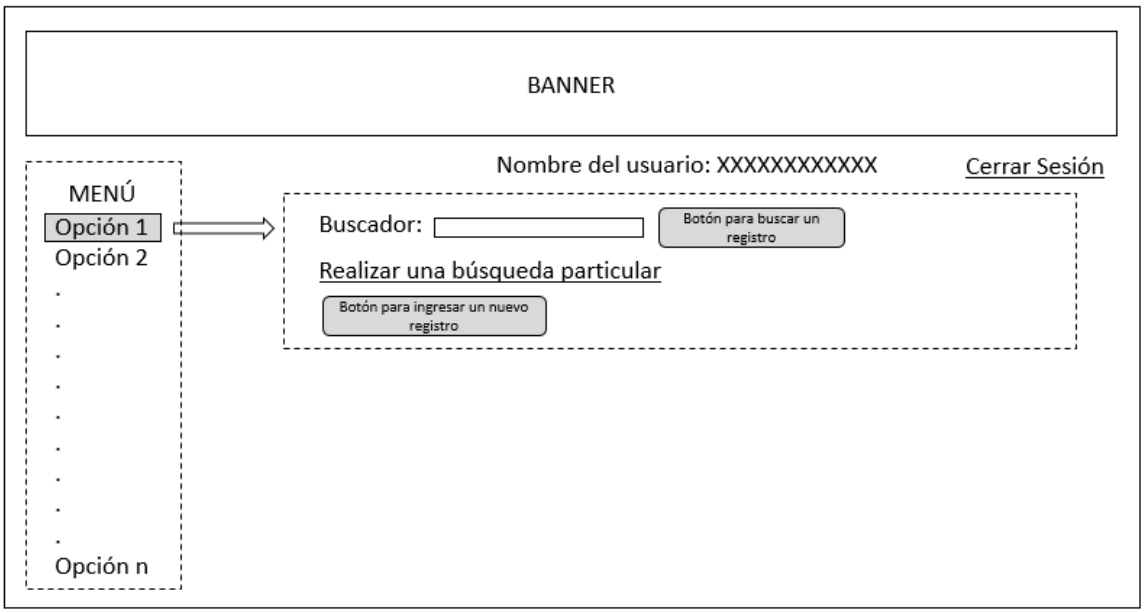


FIGURA 2.11 Prototipo de cada una de las opciones del menú

Formulario para insertar o modificar un registro

Para ingresar o modificar un registro de la base de datos, se realiza mediante un formulario.

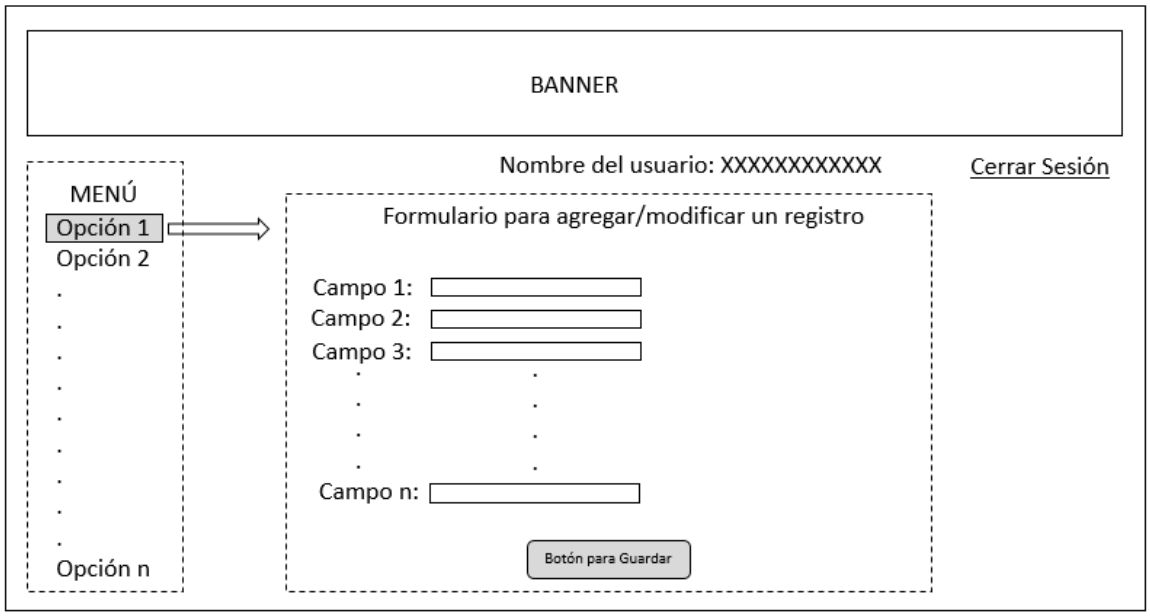


FIGURA 2.12 Prototipo del formulario para agregar o modificar un registro

Consultas particulares

Para realizar una consulta particular con fines estadísticos o para las constancias, la pantalla es similar a la que se muestra en la figura 2.13, se ingresa el parámetro y la información resultante se muestra en una tabla.

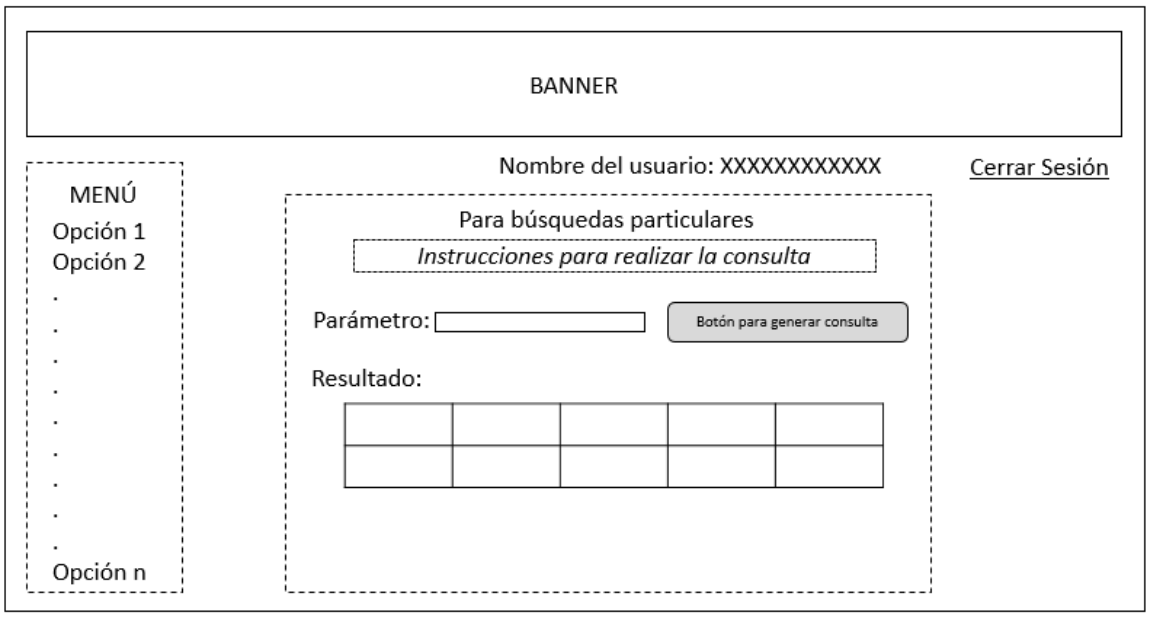


FIGURA 2.13 Prototipo de la pantalla para las consultas particulares

2.2.4 DISEÑO DE LA BASE DE DATOS

El diseño de una base de datos consta de tres fases:

- **Diseño conceptual.** Parte de la especificación de los requisitos y es independiente del modelo de datos y del Sistema Manejador de Base de Datos (SMBD) que se usará para implementarlo.
- **Diseño lógico.** Esta segunda fase parte del diseño conceptual y se describe la estructura de la base de datos que se basa en el modelo de datos que se eligió para su implementación.
- **Diseño físico.** Esta tercera fase parte del diseño lógico y es donde se describe la implementación de la base de datos usando un determinado SMBD.

2.2.4.1 DISEÑO CONCEPTUAL

Esta fase se divide en dos etapas:

- Análisis de requisitos
- Generación del esquema conceptual

2.2.4.1.1 ANÁLISIS DE REQUISITOS

Esta etapa se trata de percepción, identificación y descripción del entorno del contexto a analizar. Se responde a la pregunta ¿qué representar?. Una vez que se han estudiado las reglas de la empresa y entrevistas se elabora un esquema descriptivo de la realidad.

Para esta fase, al jefe de la Jefatura de servicio social se le solicitó que proporcionara los requisitos con los cuales se debía elaborar la base de datos, una vez que proporcionó toda la información que manejan junto con las especificaciones necesarias, se elaboró el esquema descriptivo que se muestra a continuación con el cual se partiría para elaborar la base de datos.

Se desea crear una base de datos para el registro, control y evaluación de los alumnos prestadores de servicio social de la Facultad de Psicología, de las Instituciones de servicio social y de los programas de servicio social.

Del "alumno" prestador de servicio social se tienen los siguientes datos: nombre completo, número de cuenta, número de expediente el cual es un número único que se va asignando de manera consecutiva de acuerdo a como se vayan registrando los alumnos, número de captura, mes en el que se capturaron los datos, nombre de la secretaria que capturó sus datos, sistema al que están sujetos, ya sea "Escolarizado" o "SUA (Sistema de Universidad Abierta)", plan de estudios "1971" o "2008", y su área de estudio elegida entre 6 opciones que ofrece la Facultad de Psicología, "Educación", "Clínica y de la Salud", "Psicobiología y Neurociencias", "Procesos Psicosociales y Culturales", "Organizacional" y "Ciencias Cognitivas y del Comportamiento".

También se tiene fecha de inicio y término de su servicio social, y si le fue "entregada" o está en "proceso" su carta de liberación, semestre en el que inició su servicio social, ya sea en "8vo semestre" durante el tiempo en el que sigue estudiando, "9no semestre" en el cual el alumno lo está realizando inmediatamente después de haber terminado de cursar todas sus materias, o está en estado de "pasante" por haber dejado un semestre entre el haber terminado todos sus créditos y el haber iniciado su servicio. Y entre sus datos de contacto se tiene su teléfono (celular o local) y correo electrónico (un único correo electrónico).

Entre los alumnos registrados hay algunos casos especiales en los que se libera su servicio social por alguno de los dos artículos siguientes:

- *Artículo 52: El servicio social le es liberado por ser una persona de la tercera edad o por tener alguna discapacidad.*
- *Artículo 91: La persona trabaja en alguna Institución Gubernamental y ha estado laborando por más de un año realizando actividades que tienen que ver con la Psicología.*

Cualquier movimiento (bajas y cambios de servicio social) que haga el alumno durante el tiempo en que esté realizando su servicio social, deberá ser registrado como parte de las "observaciones".

De los "programas de servicio social" que eligen los alumnos, se tiene su clave, nombre del programa, nombre de la Institución donde se encuentra el programa de servicio social y el nombre de la Dependencia. Si la Institución es la UNAM, se debe de indicar en un campo

aparte, y por último, el nombre de la Facultad o Instituto donde se encuentra dicho programa. Cada alumno puede escoger un solo programa para la realización de su servicio y cada programa puede tener inscritos a más de un alumno.

De los "supervisores" de los programas de servicio social se tiene un ID de supervisor, su nombre, el grado de estudios, ya sea el de "Lic.", "Mtro." o "Dr.", departamento en el que se encuentran laborando, área de servicio, las cuales son las mismas áreas que los alumnos eligen (área de estudio) y turno en el que se encuentra disponible el supervisor, para el turno se tienen tres opciones, "matutino", "vespertino" o "mixto". Cada programa solo es supervisado por un solo supervisor, pero el supervisor puede tener a su cargo más de un programa.

De los "asesores" solo se tiene un ID de asesor y su nombre; estos pueden asesorar a más de un alumno, pero cada alumno solo tiene asignado a un único asesor, además los asesores pueden colocar los "comentarios" convenientes sobre sus alumnos que estén asesorando.

2.2.4.1.2 ESQUEMA CONCEPTUAL

En esta segunda etapa es donde se transforma el esquema descriptivo, para esto se utiliza un modelo de datos que sea simple, pero que tenga coherencia y no sea redundante.

Los tipos de modelos conceptuales son:

- Modelo Entidad-Relación
- Modelo binario
- Modelo semántico de datos
- Modelo funcional de datos
- Orientado a objetos

El modelo elegido fue el "Modelo Entidad-Relación (MER)", ya que es el modelo más común para el diseño de bases de datos, favorece que exista una mejor comunicación entre los diseñadores por ser sencillo de entender, evita la existencia de registros duplicados y permite tener una visión global del diseño de la base de datos.

El modelo Entidad-Relación que fue introducido por Peter Chen en 1976, permite describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas. Consiste en un conjunto de objetos llamados entidades y de relaciones entre estos objetos.

La simbología que se utiliza para elaborar el MER es la que se muestra en la figura 2.14, basada en la notación Chen:

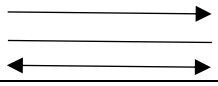
SIMBOLOGÍA	
ENTIDAD	
ATRIBUTO	
RELACIÓN	
CARDINALIDAD	

FIGURA 2.14 Simbología según Chen para el MER.

Entidad. Es cualquier objeto concreto del que se desea almacenar información, se distingue de los demás objetos por sus características que posee, que la hacen única. Existen dos tipos de entidades:

- **Entidad fuerte.** Es una entidad que tiene los atributos necesarios para distinguirse de otra.
- **Entidad débil.** Es aquella que no tiene los atributos suficientes para distinguirse de otras. Su existencia depende de otra entidad.

Atributo. Es una propiedad descriptiva que tiene cada entidad. Hay diferentes tipos de atributos:

- **Simples.** No se subdividen.
- **Compuestos.** Se dividen en otros atributos.
- **Monovalorados.** Hacen referencia a un valor único.
- **Multivalorados.** Es el atributo que tiene un conjunto de valores para una entidad concreta.
- **Derivados.** Es un atributo donde el valor puede calcularse a partir de otro.
- **Discriminante.** Es el atributo de una entidad débil que la identifica.

La **"clave primaria"** es un atributo que identifica de forma única a una entidad, mientras que la **"clave candidata"** es un atributo o conjunto de atributos que por sus características podría ser usado como llave primaria.

Relación. Es la relación entre una o más entidades. Una relación puede tener un atributo propio llamado **"atributo descriptivo"** que ayuda a describir la relación.

Cardinalidad. Es el número máximo y mínimo de ejemplares de un tipo de entidad que puede asociarse con un ejemplar del otro. Para una relación binaria, hay tres tipos de correspondencia:

- **1:1 uno a uno.** Un ejemplar de la entidad A se asocia a un ejemplar de la entidad B y viceversa.

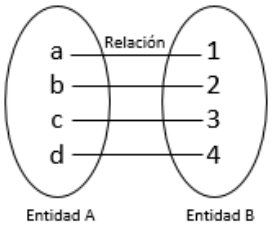


FIGURA 2.15 Correspondencia 1:1

- **1:m uno a muchos.** Un ejemplar de la entidad A se asocia con muchos ejemplares de la entidad B, y un ejemplar de la entidad B se asocia con un solo ejemplar de la entidad A.

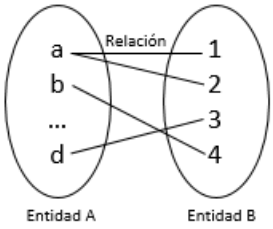


FIGURA 2.16 Correspondencia 1:m

- **m:m muchos a muchos.** Un ejemplar de la entidad A se asocia a muchos ejemplares de la entidad B y viceversa.

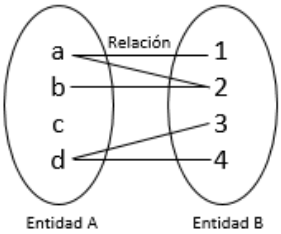


FIGURA 2.17 Correspondencia m:m

El modelo Entidad-Relación obtenido a partir de los requisitos, se muestra en la figura 2.18:

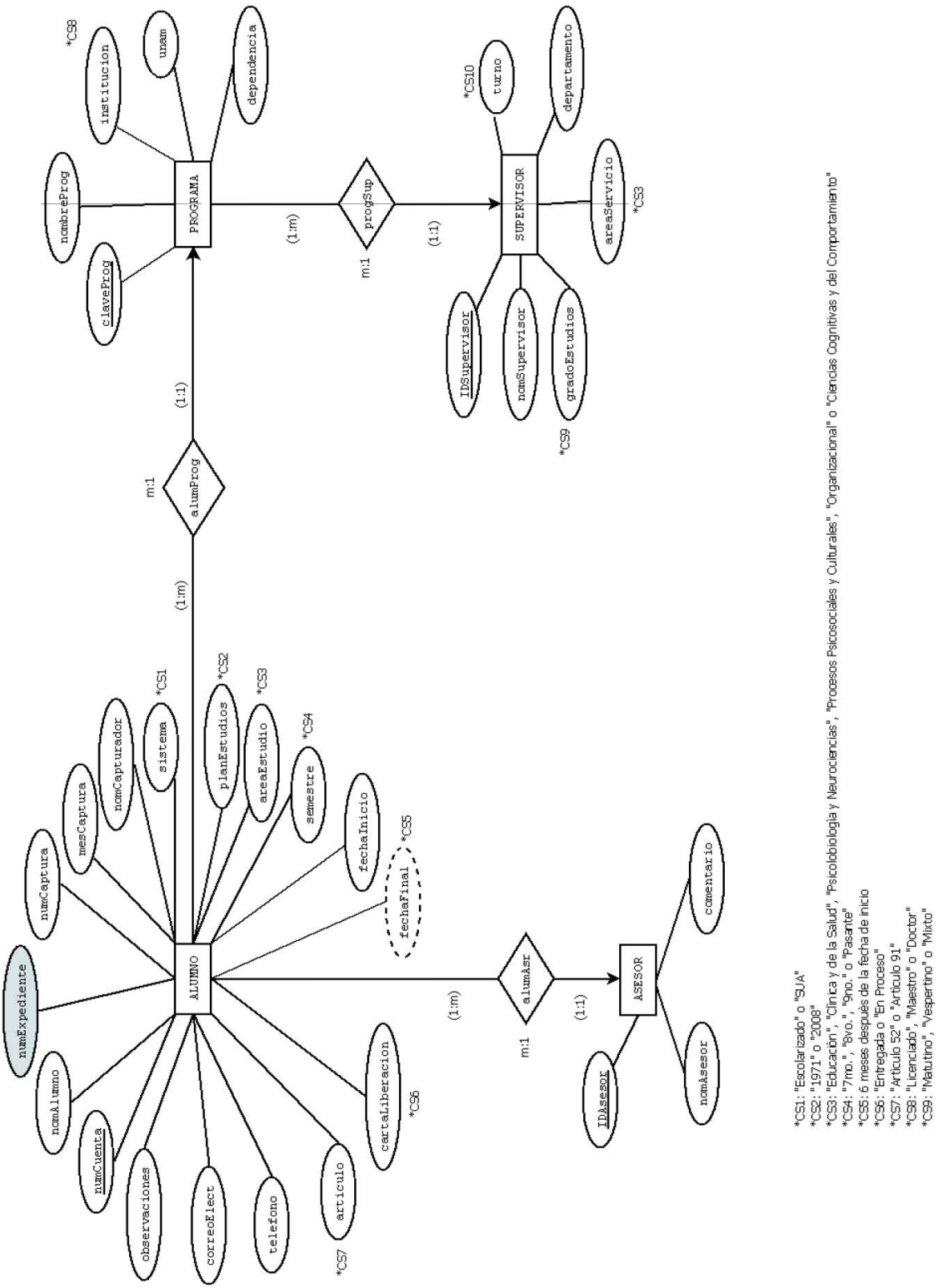


FIGURA 2.18 Modelo Entidad-Relación de la base de datos diseñada.

2.2.4.2 DISEÑO LÓGICO

El esquema lógico global se crea a partir del modelo conceptual y representa la estructura general de la base de datos.

Los tipos de modelos lógicos basados en registros son:

- **Modelo de red.** Los datos son representados como colecciones de registros arbitrarios, mientras que las relaciones como ligas lógicas.
- **Modelo jerárquico.** Los registros están organizados como conjuntos de árboles en vez de gráficas arbitrarias.
- **Modelo relacional.** Los datos y las relaciones entre los datos se representan por medio de una serie de tablas, cada una de las cuales tiene varias columnas con nombres únicos.

2.2.4.2.1 ESQUEMA LÓGICO GLOBAL

El modelo de datos que se utilizó fue el modelo relacional. Este modelo fue propuesto por Edgar Frank Codd. EL modelo relacional permite representar la información del mundo real de una manera intuitiva, que puede ser fácil de entender por cualquier persona.

Los datos en el modelo relacional se representan en forma de tablas como la que se muestra en la figura 2.19.

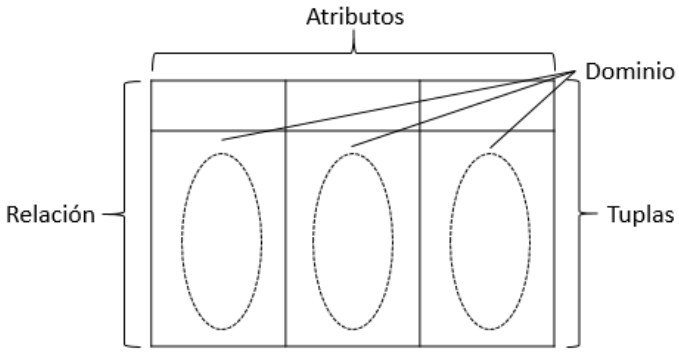


FIGURA 2.19 Ejemplo de una Tabla.

Relación. Es la estructura básica del modelo relacional, una relación es representada como una tabla bidimensional que contiene filas y columnas.

Atributo. Es una columna en una relación.

Grado de la relación. Número de atributos en una relación.

Dominio. El conjunto de valores válidos que toma un atributo.

Tupla. Es una fila en una relación.

Cardinalidad de la relación. Es el número de tuplas que contiene una relación.

Restricción. Una regla que restringe los valores en una base de datos.

Valor por defecto. Es un valor que se inserta automáticamente si el usuario no especifica la entrada.

Valor nulo. Hay ausencia de valor en un atributo.

Existen distintos tipos de llaves:

- **Llave candidata.** Cualquier conjunto de atributos que identifican unívoca y mínimamente cada tupla de una relación.
- **Llave primaria.** Es la llave candidata que elige el diseñador como llave de la relación.
- **Llave alternativa.** Es aquella llave candidata que no fue elegida como llave primaria, pero se debe de considerar en el modelo.
- **Llave foránea.** Es un conjunto de atributos en una relación que constituyen una llave primaria en otra relación.

El modelo relacional se obtuvo a partir de modelo Entidad-Relación que se había elaborado previamente, basándose en las siguientes reglas:

1.- Toda entidad fuerte se transforma en una relación, en donde se conservan todos los atributos y la llave primaria, para las llaves candidatas se establece restricción de unicidad, atributos compuestos se colocan en forma individual, para los atributos multivaluados se crea una nueva relación, propagando la llave primaria de la relación como foránea a la nueva relación, los atributos derivados o calculados se establecen como atributos calculados, y se establecen restricciones sobre los atributos.

2.- Para las entidades débiles se crea una relación conservando todos sus atributos y se propaga la llave principal de la entidad fuerte de la que depende, y la llave principal es formada por la llave primaria de la entidad fuerte y el discriminante de la entidad débil.

3.- Para las relaciones se tienen tres casos:

- **1:1.** La llave primaria de una de las dos entidades se propaga a la otra entidad como llave foránea, dependiendo del contexto.
- **m:1 o 1:m.** La llave primaria de la entidad que tiene la cardinalidad **1**, se propaga a la relación de **m** como llave foránea.
- **m:m.** Se genera una nueva relación que contendrá la llave primaria de las entidades que une, pero como llaves foráneas, además de sus atributos propios (descriptivos) de la relación.

Para las relaciones **1:1**, **1:m**, **m:1** que tienen atributos descriptivos, existen dos opciones:

- Generar una nueva relación que contendrá la llave primaria de cada una de las entidades que une, pero como llaves foráneas, además de sus atributos descriptivos.
- Propagar la llave primaria de la entidad con cardinalidad **1** junto con todos los atributos de la relación a la entidad con cardinalidad **m**.

La simbología que se utilizó es la que se muestra en la figura 2.20:

PRIMARY KEY	(PK) ó ———
UNIQUE	(U) ó ----------
FOREIGN KEY	(FK) ó ═══
CHECK	(CK)
Calculados	(C)
Discriminantes	(D)

FIGURA 2.20 Simbología utilizada para la elaboración del Modelo Relacional

Por lo tanto, el modelo relacional de la base de datos que se diseñó, es el siguiente:

ALUMNO= {numCuenta(PK), nomAlumno, numExpediente(U), numCaptura, mesCaptura, nomCapturador, sistema(CK)^{*CS1}, planEstudios(CK)^{*CS2}, areaEstudio(CK)^{*CS3}, semestre(CK)^{*CS4}, fechaInicio, fechaFinal(C)^{*CS5}, cartaLiberacion(CK)^{*CS6}, articulo(CK)^{*CS7}, telefono, correoElect, observaciones, claveProg(FK), idAsesor(Fk)**}**

PROGRAMA= {claveProg(PK), nombreProg, institución, unam, dependencia, idSupervisor(FK)**}**

SUPERVISOR= {idSupervisor(PK)**,** nomSupervisor, gradoEstudios(CK)^{*CS8}, areaServicio(CK)^{*CS3}, departamento, turno(CK)^{*CS9}**}**

ASESOR= {idAsesor(PK), nomAsesor, comentarios**}**

2.2.4.2.2 NORMALIZACIÓN

El siguiente paso que se realizó una vez que se obtuvo el modelo relacional, fue la normalización, este proceso surge debido a problemas que se presentan en el diseño de la base de datos como:

- Redundancia
- Anomalías de modificación, inserción y borrado de datos.

La teoría de la normalización se encarga de eliminar los comportamientos anormales de las relaciones durante las actualizaciones, los datos redundantes y facilita la comprensión de las relaciones semánticas entre los datos.

Para la base de datos que se diseñó, se normalizó hasta la 3FN, a continuación se presenta el proceso de normalización que se tuvo que realizar:

1ra. forma normal. Una relación está en 1FN si y solo sí, cada renglón columna contiene valores atómicos. Eso quiere decir que únicamente todas las celdas de las tablas contienen valores simples (un solo valor por celda), además también cada columna debe de tener un nombre único.

La base de datos en 1FN queda de la siguiente forma:

ALUMNO
numCuenta
nomAlumno
~~ApPatAlumno~~
~~ApMatAlumno~~
numExpediente
numCaptura
mesCaptura
nomCapturador
sistema
planEstudios
areaEstudio
semestre
fechaInicio
fechaFinal
cartaLiberacion
articulo
telefono
correoElect
observaciones
claveProg
idAsesor

PROGRAMA
claveProg
nombreProg
institución
unam
dependencia
idSupervisor

SUPERVISOR
idSupervisor
nomSupervisor
~~ApPatSupervisor~~
~~ApMatSupervisor~~
gradoEstudios
areaServicio
departamento
turno

ASESOR
idAsesor
nomAsesor
~~ApPatAsesor~~
~~ApMatAsesor~~
comentarios

FIGURA 2.21 Base de datos normalizada hasta la 1FN

Estrictamente como se muestra en la figura 2.21, la base de datos ya está en 1FN, sin embargo por indicaciones del usuario final, el nombre del alumno, supervisor y asesor no tenía que ser separado en tres campos como se debe, sino en un solo campo para facilitar el trabajo a la hora de la inserción o búsqueda del nombre.

2da. forma normal. Una relación está en 2FN si y solo sí está en 1FN y todos los atributos no clave dependen por completo de la llave primaria. Esta 2FN consiste en identificar que atributos dependen de otros.

Para poder normalizar a la 2FN, se identificó la llave primaria de cada una de las tablas y después los atributos que dependen de ella, dando como resultado las siguientes tablas donde se observa que el nombre de alumno, supervisor y asesor ya no fueron divididos:

numCuenta -> nomAlumno, sistema, planEstudios, areaEstudio, teléfono, correoElect

claveProg -> nombreProg, institucion, unam, dependencia

idSupervisor -> nomSupervisor, gradoEstudios, areaServicio, departamento, turno

idAsesor -> nomAsesor

numCuenta, claveProg, idSupervisor -> numExpediente, numCaptura, mesCaptura, nomCapturador, semestre, fechaInicio, fechaFinal, cartaLiberacion, articulo, observaciones

numCuenta, idAsesor -> comentarios

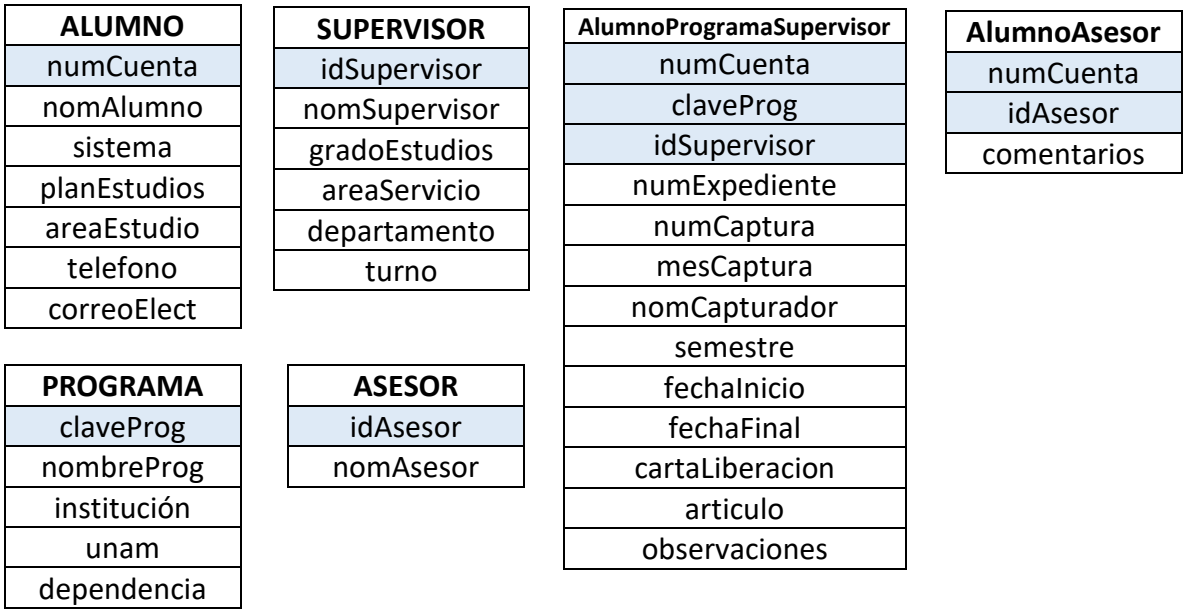


FIGURA 2.22 Base de datos normalizada hasta la 2FN

Luego de aplicar a la base de datos la 2FN, se obtuvieron dos nuevas tablas, la tabla "AlumnoProgramaSupervisor" que cuenta con los datos del servicio social al que está inscrito el alumno, y la segunda tabla "AlumnoAsesor" cuenta con el "idAsesor" junto con el atributo "comentarios", los cuales proporciona el asesor del alumno; sin embargo, se tuvo que aplicar la 3FN para eliminar las dependencias transitivas.

3ra. forma normal. Una relación está en 3FN sí y solo sí está en 2FN y todos los atributos no clave dependen en forma no transitiva de la llave primaria.

Al término de haber aplicado la 3FN, se garantizó tener una base de datos bien hecha, sin redundancia y sin pérdida de información. Para evitar la creación de dos tablas con nombre distinto, pero mismo contenido, los atributos "areaEstudio" y "areaServicio" se convirtieron en un solo atributo llamado "areaPsicologia" contando ahora con un "idArea"; además de la creación de cuatro nuevas tablas, Sistema, Institución, Dependencia y Departamento, y las tablas "AlumnoProgramaSupervisor", "AlumnoAsesor" se renombraron como "registroSS" y "comentarioAsesor" respectivamente.

En la figura 2.23, se muestra la relación entre las tablas:

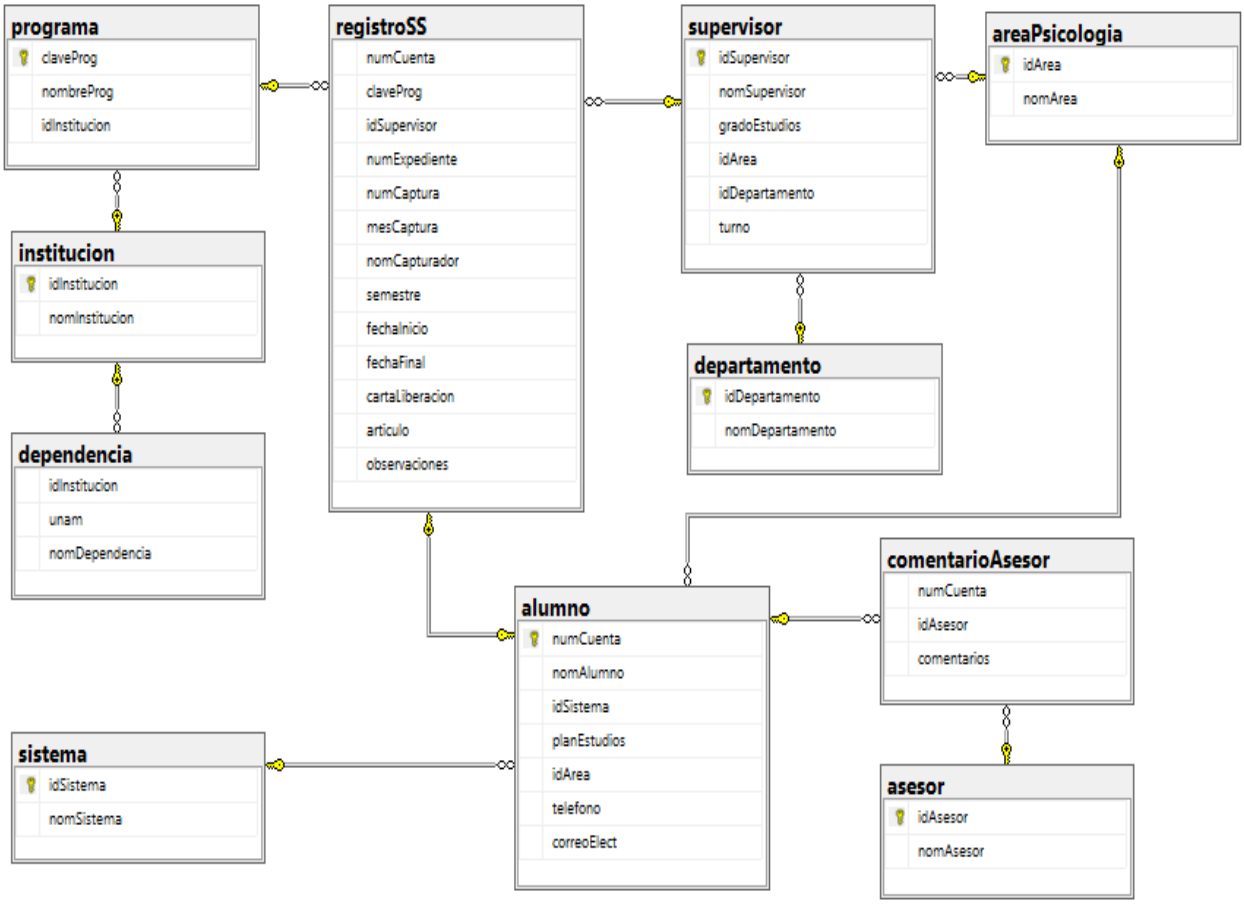


FIGURA 2.23 Base de datos normalizada hasta la 3FN

2.2.4.2.3 ESQUEMA LÓGICO

Después de que la base de datos se normalizó hasta la 3FN, se diseñó el esquema lógico definitivo que consta de un conjunto de relaciones, estas están formadas por:

- Nombre de la relación
- Lista de atributos entre paréntesis
- La clave primaria y claves ajenas si las tiene
- Las reglas de integridad de las claves ajenas

El esquema lógico obtenido, fue el siguiente:

ALUMNO= {numCuenta(PK), nomAlumno, idSistema(FK), planEstudios(CK)^{*CS2}, idArea(FK), telefono, correoElect**}**

PROGRAMA= {claveProg(PK), nombreProg, idInstitucion(FK)**}**

SUPERVISOR= {idSupervisor(PK)**,** nomSupervisor, gradoEstudios(CK)*CS8, idArea(FK), idDepartamento(FK), turno(CK)*CS9**}**

ASESOR= {idAsesor(PK), nomAsesor**}**

REGISTROSS= {numCuenta(FK), claveProg(FK), idSupervisor(FK), numExpediente(U), numCaptura, mesCaptura, nomCapturador, semestre(CK)*CS4, fechaInicio, fechaFinal(C)*CS5, cartaLiberacion(CK)*CS6, articulo(CK)*CS7, observaciones**}**

COMENTARIOASESOR= {numCuenta(FK), idAsesor(FK), comentarios **}**

INSTITUCION= {idInstitucion(PK), nomInstitucion**}**

DEPENDENCIA= {idInsticuion(FK), unam, nomDependencia**}**

SISTEMA= {idSistema(PK), nomSistema(CK)*CS1**}**

AREAPSICOLOGIA= {idArea(PK), nomAreaPsicologia**}**

DEPARTAMENTO= {idDepartamento(PK), nomDepartamento**}**

A la base de datos se le añadió una tabla más, sin relación alguna con el resto, esta tabla almacena la información de los usuarios que se registran en la aplicación web diseñada.

USUARIO= {idUsuario(PK), nombreUsuario, usuario, contraseña**}**

2.2.4.2.4 DICCIONARIO DE DATOS

En el diccionario de datos se describen por esquema (tabla) los atributos, y para cada atributo se tiene:

- **Dominio:** tipo de dato, longitud y restricciones de dominio.
- Valor por defecto el cual es opcional.
- Si admite nulos.
- Si es derivado, en caso de que así sea se coloca como se calcula su valor.

El esquema que se utilizó es el que se muestra en la figura 2.24:

Nombre atributo	Tipo de Dato	Llave	Restricciones De Dominio	Omisión	Obligatorio	Integridad	Derivado

FIGURA 2.24 Esquema utilizado para elaborar el diccionario de datos

ALUMNO

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
numCuenta	INT	PK	NO	NO	SI	NO	NO
nomAlumno	VARCHAR(50)	NO	NO	NO	SI	NO	NO
idSistema	CHAR(10)	FK	NO	NO	NO	valor nulo	NO
planEstudios	INT	NO	1971 o 2008	NO	SI	CK	NO
idArea	CHAR(10)	FK	NO	NO	NO	valor nulo	NO
telefono	INT	NO	NO	NO	SI	NO	NO
correoElect	VARCHAR(40)	NO	NO	NO	SI	NO	NO

PROGRAMA

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
claveProg	VARCHAR(30)	PK	NO	NO	SI	NO	NO
nombreProg	VARCHAR(110)	NO	NO	NO	SI	NO	NO
idInstitucion	CHAR(10)	FK	NO	NO	NO	cascada	NO

SUPERVISOR

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
idSupervisor	CHAR(10)	PK	NO	NO	SI	NO	NO
nomSupervisor	VARCHAR(50)	NO	NO	NO	SI	NO	NO
gradoEstudios	VARCHAR(15)	NO	Lic., Mtro. o Dr.	NO	SI	CK	NO
idArea	CHAR(10)	FK	NO	NO	NO	valor nulo	NO
idDepartamento	CHAR(10)	FK	NO	NO	NO	valor nulo	NO
turno	VARCHAR(15)	NO	Mat., Vesp., o Mixto	NO	SI	CK	NO

ASESOR

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
idAsesor	CHAR(10)	PK	NO	NO	SI	NO	NO
nomAsesor	VARCHAR(50)	NO	NO	NO	SI	NO	NO

REGISTROSS

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
numCuenta	INT	FK	NO	NO	NO	cascada	NO
claveProg	VARCHAR(30)	FK	NO	NO	NO	D-valor nulo U-cascada	NO
idSupervisor	CHAR(10)	FK	NO	NO	NO	valor nulo	NO
numExpediente	INT	U	Número sin repeticiones	NO	SI	NO	NO
numCaptura	INT	NO	NO	NO	SI	NO	NO
mesCaptura	VARCHAR(15)	NO	NO	NO	SI	NO	NO
nomCapturador	VARCHAR(50)	NO	NO	NO	SI	NO	NO
semestre	VARCHAR(20)	NO	7mo., 8vo., 9no., o Pasante	NO	SI	CK	NO
fechaInicio	VARCHAR(15)	NO	NO	NO	SI	NO	NO
fechaFinal	VARCHAR(15)	NO	NO	NO	SI	NO	6 meses después de la fechaInicio
cartaLiberacion	VARCHAR(30)	NO	Entregada o en Proceso	NO	NO	CK	NO
articulo	VARCHAR(20)	NO	Art. 52, o Art. 91	NO	NO	CK	NO
observaciones	VARCHAR(200)	NO	NO56t	NO	NO	NO	NO

COMENTARIOASESOR

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
numCuenta	INT	FK	NO	NO	NO	cascada	NO
idAsesor	CHAR(10)	FK	NO	NO	NO	D-valor nulo U-cascada	NO
comentarios	VARCHAR(200)	NO	NO	NO	NO	NO	NO

INSTITUCION

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
idInstitucion	CHAR(10)	PK	NO	NO	SI	NO	NO
nomInstitucion	VARCHAR(100)	NO	NO	NO	SI	NO	NO

DEPENDENCIA

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
idInstitucion	CHAR(10)	FK	NO	NO	NO	cascada	NO
unam	VARCHAR(100)	NO	NO	NO	NO	NO	NO
dependencia	VARCHAR(100)	NO	NO	NO	SI	NO	NO

SISTEMA

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
idSistema	CHAR(10)	PK	NO	NO	SI	NO	NO
nomSistema	VARCHAR(20)	NO	Escolarizado o SUA	NO	SI	CK	NO

AREAPSICOLOGIA

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
idArea	CHAR(10)	PK	NO	NO	SI	NO	NO
nomArea	VARCHAR(70)	NO	6 áreas de la Psicología	NO	SI	CK	NO

6 áreas de la Psicología: “Educación”, “Clínica y de la Salud”, “Psicobiología y Neurociencias”, “Procesos Psicosociales y Culturales”, “Organizacional” y “Ciencias Cognitivas y del Comportamiento”.

DEPARTAMENTO

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
idDepartamento	CHAR(10)	PK	NO	NO	SI	NO	NO
nomDepartamento	VARCHAR(100)	NO	NO	NO	SI	NO	NO

USUARIO

Nombre Atributo	Tipo Dato	Llave	Restr. De Dominio	Omisión	Obligatorio	Integridad	Derivado
idUsuario	CHAR(10)	PK	NO	NO	SI	NO	NO
nombreUsuario	VARCHAR(50)	NO	NO	NO	SI	NO	NO
usuario	VARCHAR(100)	NO	NO	NO	SI	NO	NO
contraseña	VARCHAR(100)	NO	NO	NO	SI	NO	NO

Simbología utilizada. U: Actualización, **D:** Borrado

2.2.4.3 DISEÑO FÍSICO

La última fase del diseño de una base de datos está dividida en cuatro etapas:

- **Traducir el esquema lógico global para el SMBD específico.-** En esta etapa es donde se utiliza la información obtenida en el esquema lógico global y en el diccionario de datos.
- **Diseñar las relaciones base para el SMBD específico.-** Las relaciones base se definen mediante el lenguaje de definición de datos del SMBD.
- **Diseñar las reglas de negocio para el SMBD específico.-** Las actualizaciones que se realizan sobre las relaciones de las bases de datos, deben observar ciertas restricciones que imponen las reglas de negocio de la empresa. Existen algunas restricciones que no pueden ser manejadas por los SMBD, para estas restricciones se deben de escribir programas de aplicación específicos.
- **Diseñar las vistas de los usuarios.-** Una vista es una tabla cuyo contenido está definido por una consulta. Al igual que una tabla base, una vista consta de un conjunto de atributos y tuplas con un nombre. Las vistas suelen utilizarse para restringir el acceso de usuarios a ciertos datos de las tablas. Pueden usarse como un mecanismo de seguridad.

2.2.4.3.1 TRADUCCIÓN DEL ESQUEMA LÓGICO GLOBAL PARA EL SMBD ESPECÍFICO

Una vez que se obtuvieron el esquema lógico y el diccionario de datos, se eligió el SMBD que cumpliera con las funciones necesarias, para la implementación de la base de datos se eligió **"SQL SERVER"** de Microsoft, ya que es un manejador de base de datos relacionales que soporta procedimientos almacenados, uso de comandos DDL y DML, puede administrar la información de otros servidores de datos, se puede trabajar mediante línea de comandos o por interfaz gráfica al instalar **"SQL SERVER MANAGEMENT STUDIO",** y trabaja usando el modo cliente-servidor donde la base de datos queda alojada en el servidor y el cliente es el que accede a la información (datos).

2.2.4.3.2 DISEÑO DE LAS RELACIONES BASE PARA EL SMBD ESPECÍFICO

La implementación de la base de datos se hizo mediante interfaz gráfica, así que se instaló "SQL SERVER MANAGEMENT STUDIO". En esta parte se utilizó el Lenguaje de Definición de Datos para la creación de las tablas, con apoyo de la información del esquema global y del diccionario de datos. También se definieron los tipos de datos, los tipos de llave y las reglas de integridad.

Con el uso del Lenguaje de Manipulación de Datos se diseñaron los procedimientos de inserción, modificación (actualización), borrado y consulta de datos, para cada una de las tablas de la base de datos.

2.2.4.3.3 DISEÑO DE LAS REGLAS DE NEGOCIO PARA EL SMBD ESPECÍFICO

La integridad de datos es una forma de asegurar que la base de datos sólo tiene información exacta y aceptable. La integridad referencial se usa para asegurar que cada valor de la llave foránea concuerde con el valor de la llave primaria de otra relación.

Para declarar las restricciones de integridad de una tabla se usa la cláusula CONSTRAINT del comando CREATE TABLE.

En varias de las columnas de las tablas de la base de datos, se declararon restricciones de integridad para asegurar que el usuario únicamente introduzca información correcta y concreta, e incluso sin repeticiones. Las columnas junto con sus restricciones se muestran en la figura 2.25:

COLUMNA	RESTRICCIÓN
planEstudios	1971, 2008
gradoEstudios	Lic., Mtro., Dr.
turno	Mat., Vesp., Mixto
numExpediente	Sin repeticiones
semestre	7mo., 8vo., 9no., pasante
cartaLiberacion	Entregada, En proceso
articulo	Art. 52, Art. 91
nomSisrema	Escolarizado, SUA
nomArea	"Educación", "Clínica y de la Salud", "Psicobiología y Neurociencias", "Procesos Psicosociales y Culturales", "Organizacional", "Ciencias Cognitivas y del Comportamiento"

FIGURA 2.25 Restricciones de integridad

También se declararon constraint de llave foránea, tanto de borrado o actualización en cascada y de valor nulo que se muestran en la figura 2.26:

COLUMNA	RESTRICCIÓN	
idSistema	DELETE - valor nulo	Si se elimina un registro de la tabla **sistema**, en la tabla **alumno** donde aparezca la clave idSistema será nula.
idArea	DELETE - valor nulo	Si se elimina un registro de la tabla **areaPsicologia**, en las tablas **alumno** y **supervisor** donde aparezca la clave idArea será nula.
idInstitucion	DELETE – cascada	Si se elimina un registro de la tabla **institucion**, todos los registros donde aparezca el idInstitucion en la tabla **dependencia** se eliminarán.
idDepartamento	DELETE – valor nulo	Si se elimina un registro de la tabla **departamento**, en la tabla **supervisor** la clave idDepartamento será nula.
numCuenta	DELETE – cascada	Si se elimina un registro de la tabla **alumno**, el registro de servicio social en la tabla **registroSS**, y el registro en la tabla **comentarioAsesor** se eliminarán.
claveProg	DELETE – valor nulo	Si se elimina un registro de la tabla **programa**, en la tabla **registroSS** la clave claveProg será nula.
idSupervisor	DELETE – valor nulo	Si se elimina un registro de la tabla **supervisor**, en la tabla **registroSS** donde aparezca la clave idSupervisor será nula.

	UPDATE – cascada	Si se modifica un registro en la tabla **supervisor**, en la tabla **registroSS** donde aparezca la clave idSupervisor también se modificará.
idAsesor	DELETE – valor nulo	Si se elimina un registro de la tabla **asesor**, en la tabla **comentarioAsesor** donde aparezca la clave idAsesor será nula.
	UPDATE – cascada	Si se modifica un registro en la tabla **asesor**, en la tabla **comentarioAsesor** donde aparezca la clave idAsesor también se modificará.

FIGURA 2.26 Constraint de llave foránea

2.2.4.3.4 DISEÑO DE LAS VISTAS DE LOS USUARIOS

De acuerdo a las indicaciones dadas por la dependencia, se diseñaron las vistas de usuarios, estas vistas son consultas de datos específicos, algunas consultas son con fines estadísticos, mientras que otras son para reducir el tiempo de búsqueda de información.

Los datos que se presentan en pantalla provienen de diferentes tablas, para la elaboración de dichas vistas, se utilizó el álgebra relacional.

A continuación se enlistan cada una de las vistas creadas:

- Datos de los alumnos inscritos a un programa de servicio social por “mes de captura”.
- Número de alumnos por “sistema”, “área de estudio” y “plan de estudios” en función del sistema seleccionado.
- Número de programas de servicio social de cada dependencia por “Institución”.
- Número y nombre de los supervisores de programas de servicio social por “dependencia”.
- PRIDES (Datos de los alumnos inscritos a determinado programa de servicio social, nombre del supervisor y nombre del programa de servicio social por “clave de dicho programa”).
- Constancias de asesores (Datos de los alumnos asesorados por “asesor”).

2.3 IMPLEMENTACIÓN Y PRUEBA DE UNIDADES

2.3.1 IMPLEMENTACIÓN

La etapa de implementación toma en cuenta las etapas de requerimientos y diseño del sistema. Los procesos de desarrollo rápido de software, están diseñados para producir software útil de forma rápida. Las etapas de especificación, diseño, desarrollo y pruebas se entrelazan.

2.3.1.1 TÉCNICAS DE DESARROLLO RÁPIDO DE APLICACIONES

Las técnicas de desarrollo rápido de aplicaciones (Rapid Application Development, RAD por sus siglas en inglés) evolucionaron de los lenguajes de cuarta generación en los años 80’s y se utilizan para desarrollar aplicaciones con un uso intensivo de datos. Las técnicas están organizadas como un

conjunto de herramientas que permiten crear datos, buscarlos, visualizarlos y presentarlos en informes.

Las herramientas que se incluyen en un entorno RAD son:

- **Un lenguaje de programación de bases de datos.** El lenguaje tiene conocimiento de la estructura de la base de datos, además incluye las operaciones básicas de manipulación de bases de datos. El lenguaje de programación de bases de datos es SQL.
- **Un generador de interfaces.** Se utiliza para crear formularios de introducción y visualización de datos.
- **Enlaces a aplicaciones de oficina.** Una hoja de cálculo para el análisis y manipulación de información numérica o un procesador de texto para la creación de las plantillas de informes.
- **Un generador de informes.**- Se utiliza para definir y crear informes a partir de la información de la base de datos.

Las aplicaciones de negocio a menudo comprenden la actualización de una base de datos y la producción de informes a partir de la información existente en ella. Para las entradas y salidas se utilizan formularios estándar.

Los sistemas RAD están dirigidos a la elaboración de aplicaciones interactivas que se apoyan en la abstracción de información en una base de datos organizacional, presentando la información a los usuarios en su terminal o estación de trabajo y actualizando la base de datos con los cambios de los usuarios.

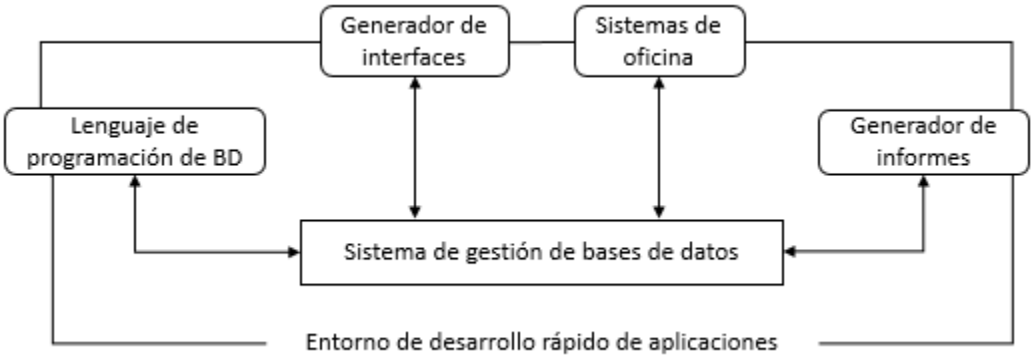


FIGURA 2.27 Entorno de desarrollo rápido de aplicaciones

Actualmente, muchos entornos RAD permiten el desarrollo de interfaces de base de datos basadas en navegadores web, el acceso a la base de datos se hace desde cualquier lugar a través de una conexión a internet. El costo de software se reduce y permite a usuarios externos tener acceso a la base de datos.

La mayoría de los sistemas RAD incluyen herramientas de programación visual que permiten desarrollar de forma interactiva. El desarrollador del sistema sustituye el escribir un programa secuencial, por la manipulación de íconos gráficos que representan funciones, datos o componentes de interfaces de usuario, y asocia el procesamiento de secuencias de comandos con estos íconos. Los programadores constituyen el sistema definiendo la interfaz en términos de pantallas, campos,

botones y menús. El programa ejecutable se genera automáticamente a partir de la representación visual del sistema.

Como se mencionó en el diseño físico de la base de datos, el manejador que se utilizó fue SQL Server, para el desarrollo de la aplicación web se instaló el IDE **"Visual Studio"** de Microsoft, el cual soporta lenguajes de programación como C++, C#, Visual Basic, .NET, F#, Java, Python, PHP entre otros, así como también entornos de desarrollo web como ASP.NET, Django, etc. Visual Studio permite crear sitios y aplicaciones web que soporten la plataforma .NET. Se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos, entre otros.

Una vez que se instaló Visual Studio, se hizo uso del entorno de desarrollo **ASP.NET** que es usado para crear sitios web dinámicos, aplicaciones web y servicios web XML. A las páginas de ASP.NET se les conoce como "Web Forms" (formularios web, en español) que son el principal medio para la creación de aplicaciones web. Los Web Forms tienen la extensión .aspx, los cuales contienen etiquetas HTML o XHTML, así como también etiquetas definiendo controles web que se procesan del lado del servidor y controles de usuario donde los desarrolladores colocan todo el código estático y dinámico requerido por la página web.

2.3.1.2 METODOLOGÍAS PARA EL DESARROLLO ÁGIL DE SOFTWARE

Los métodos ágiles universalmente dependen de un enfoque iterativo para la especificación, desarrollo y entrega de software, fueron diseñados para apoyar al desarrollo de aplicaciones de negocio donde los requerimientos del sistema normalmente cambiaban rápidamente durante el proceso de desarrollo.

Algunos de los métodos ágiles conocidos, son:

- **Programación extrema.** Es uno de los métodos más conocido y usado. El nombre de programación extrema se debe a Beck, debido a que el enfoque fue desarrollado utilizando las mejores prácticas del desarrollo iterativo con la participación extrema del cliente. Todos los requerimientos son expresados como escenarios, los cuales se implementan como una serie de tareas.
- **Scrum.** Es un método que aplica las mismas premisas conceptuales que la programación extrema pero para resolver un problema ligeramente distinto como es el de desarrollo evolutivo de aplicaciones. Sirve para gestionar el desarrollo de software, el objetivo es maximizar el retorno de la inversión para la empresa. Se basa principalmente en construir la funcionalidad de mayor valor para el cliente y en los principios de inspección continua, adaptación, auto-gestión e innovación.
- **Cristal.** La metodología Cristal identifica con colores diferentes cada método, y su elección debe ser consecuencia del tamaño y criticidad del proyecto, de forma que los de mayor tamaño, o aquellos en los que la presencia de errores o desbordamiento de agendas implique consecuencias graves, deben adoptar metodologías más pesadas. De esta forma se busca lograr mayor rentabilidad en el desarrollo de proyectos de software.
- **Desarrollo adaptativo de software.** Fue propuesto por Jim Highsmith en 1998, es una técnica que se utiliza para construir software y sistemas complejos, se enfoca en la colaboración y organización propia del equipo. Un enfoque de desarrollo ágil y adaptativo

basado en la colaboración, es una fuente de orden en las complejas interacciones entre disciplina e ingeniería.

La metodología que se utilizó fue la programación extrema, por ser uno de los métodos más usuales y sencillo de aplicar, además permitió tener una mejor organización en la etapa de implementación de cada uno de los requerimientos.

2.3.1.2.1 PROGRAMACIÓN EXTREMA

La programación extrema es un método ampliamente utilizado, donde los requerimientos se expresan como escenarios llamados historias de usuarios, los cuales se implementan directamente como una serie de tareas. Las actividades del ciclo de entrega en la programación extrema se muestran en la figura 2.28:

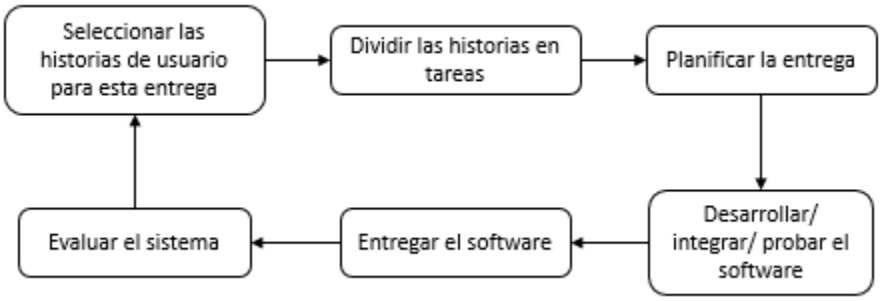


FIGURA 2.28 Ciclo de entrega de la programación extrema

La programación extrema implica varias prácticas, algunas de las que se utilizaron en el desarrollo del sistema, se enlistan enseguida:

- **Planificación incremental.** Los requerimientos se registran en tarjetas de historias y las historias a incluir en una entrega se determinan según el tiempo disponible y su prioridad relativa. Los desarrolladores dividen las historias en tareas de desarrollo.
- **Entregas pequeñas.** El mínimo conjunto útil de funcionalidad que proporcione valor de negocio se desarrolla primero. Las entregas del sistema son frecuentes e incrementalmente añaden funcionalidad a la primera entrega.
- **Diseño sencillo.** Sólo se lleva a cabo el diseño necesario para cumplir los requerimientos actuales.
- **Desarrollo previamente probado.** Se utiliza un sistema de pruebas de unidad automatizado para escribir pruebas para nuevas funcionalidades antes de que éstas se implementen.
- **Integración continua.** En cuanto acaba el trabajo en una tarea, se integra en el sistema entero. Después de la integración, se deben pasar al sistema todas las pruebas de unidad.
- **Cliente presente.** Debe estar disponible al equipo de la XP, un representante de los usuarios finales del sistema (cliente) a tiempo completo. En un proceso de la programación extrema, el cliente es miembro del equipo de desarrollo y es responsable de formular al equipo los requerimientos del sistema para su implementación.

Cada una de las prácticas de la programación extrema fueron adaptadas al modelo en cascada que se eligió, las entregas de cada uno de los requerimientos no se anexaban al sistema como tal para

de inmediato ponerlo en funcionamiento para los usuarios, porque los requerimientos dependían unos de otros para que el sistema funcionara correctamente y fuera eficiente. Cada requerimiento se terminaba y se presentaba al representante de los usuarios, el representante era el jefe de la Jefatura de servicio social; la presentación de cada uno de los requerimientos con sus respectiva prueba de funcionamiento se realizaban de manera demostrativa con videos, una vez que el representante lo aprobaba ya sea al momento o después de haber solicitado alguna modificación, se continuaba con el siguiente requerimiento, repitiendo el mismo proceso y presentando el avance cada cierto periodo de tiempo, los requerimientos se iban anexando al sistema, una vez que concluyeron todos los requerimientos, se entregó el sistema completo.

Respetando la prioridad de los requerimientos tratados ahora como escenarios, las tareas que representan cada una de las funciones que realiza el sistema son las siguientes:

TAREA 1: Sistema de registro de usuarios para ingresar a la aplicación web
La primera página que muestra la aplicación al ingresar, es un formulario de inicio de sesión, si es la primera vez que se ingresa, se debe realizar el registro en la aplicación siguiendo las instrucciones en pantalla. Para realizar el registro se muestra un formulario en el que se ingresará un “ID de usuario”, “nombre de usuario”, “usuario”, “contraseña”, una vez hecho el registro se podrá acceder a la aplicación, ingresando los dos últimos datos proporcionados en el registro que contendrá únicamente información personal y confidencial. Tanto en el registro de usuario como en el inicio de sesión, si los datos que se ingresaron no cumplen con el formato indicado, no coinciden con los que se registraron, o algún campo se quedó vacío, el sistema mostrará un mensaje de error notificando la causa por la cual no se puede llevar a cabo el registro o el inicio de sesión.

TAREA 2: Registro de alumnos, supervisores, asesores y programas de servicio social
Una vez que se ha ingresado a la aplicación web, se tendrá que seleccionar la opción deseada del menú de la página principal para ingresar información, se mostrará un formulario que llenará, una vez que se hayan llenado los campos correspondientes, el usuario guardará la información en la base de datos. Si el registro ya existe, se han quedado vacíos los campos obligatorios o el formato que se está introduciendo no es el correcto, el sistema mostrará un mensaje de error, notificando la causa por la cual el registro no puede ser almacenado. Si en la base ya se cuenta con la información de los supervisores, asesores y programas de servicio social vigentes, y un alumno desea iniciar su servicio social, se ingresarán sus datos personales, y se le registrará su servicio social, así como se le asignará un asesor.

TAREA 3: Búsqueda de registros en la base de datos
Si se desea consultar algún registro almacenado en la base de datos para modificarlo o eliminarlo, cada opción del menú cuenta con un buscador, una caja de texto en donde se ingresará ya sea el número de cuenta del alumno, id del supervisor o asesor, clave del programa de servicio social, entre otros datos clave. Al buscar el registro deseado, el sistema mostrará un formulario con la información deseada. Si el registro no existe en la base de datos o se introduce información errónea, el sistema mostrará un mensaje notificando la causa por la cual no encuentra el registro.

TAREA 4: Modificar registros en la base de datos
Al consultar el registro deseado, el sistema mostrará un formulario con la información deseada, los campos se podrán modificar y posteriormente guardar dichos cambios que se hayan realizado. Las modificaciones se pueden dar por las siguientes razones: ❖ Surgen errores de captura, lo que requiere que algunos campos sean modificados. ❖ Las claves de los programas de servicio social son actualizadas porque dicho programa aún seguirá vigente. ❖ El supervisor de algún programa de servicio social es cambiado. ❖ El asesor de algún alumno es cambiado. Si se han dejado campos obligatorios sin llenar o se ha introducido información errónea, el sistema mostrará un mensaje notificando la causa por la cual no se pueden guardar los cambios.

TAREA 5: Eliminar registros en la base de datos
Al consultar el registro deseado, el sistema mostrará un formulario con la información deseada, el registro consultado se podrá eliminar si la información que contiene ya no es necesaria por alguna de las siguientes causas: ❖ El alumno ha solicitado cambio de programa de servicio social, su registro actual es eliminado y es dado de alta en un nuevo programa de servicio social anotando en el campo de "observaciones" el movimiento que ha realizado el alumno. ❖ Al comenzar el año, algún programa de servicio social ya no es renovado. ❖ El supervisor o asesor han dejado de laborar. ❖ La institución o dependencia ya no solicita alumnos prestadores de servicio social. Si alguno de los registros no se puede eliminar, el sistema mostrará un mensaje notificando la causa por la cual no se puede eliminar el registro.

TAREA 6: Consulta de datos para la elaboración de PRIDES
Desde el menú se ingresa a la opción PRIDES, el sistema mostrará las indicaciones a seguir, en la caja de texto se ingresará la clave del programa de servicio social deseado, y el sistema mostrará el nombre del programa de servicio social, el nombre del supervisor de dicho programa en una caja de texto, además en una tabla mostrará los datos de los alumnos inscritos como su nombre, número de cuenta, número de expediente, fecha de inicio y término del servicio social. Si la clave no es encontrada por no existir o estar incorrectamente escrita, el sistema mostrará un mensaje notificando el error.

TAREA 7: Consulta de datos para la elaboración de constancias para los asesores
En un apartado dentro de la opción "asesores" del menú, se puede realizar la consulta de los alumnos asesorados, el sistema mostrará las indicaciones a seguir, en una caja de texto se ingresará el nombre del asesor y el sistema mostrará los datos de los alumnos asesorados como su nombre, número de cuenta, número de expediente, fecha de inicio y término del servicio social, clave y nombre del programa de servicio social al que estuvieron inscritos. Si el asesor no es encontrado o está mal escrito el nombre del asesor, el sistema mostrará un mensaje notificando el error.

TAREA 8: Consulta de datos de los alumnos inscritos a un programa de servicio social por mes de captura
En un apartado dentro de la opción “alumnos” del menú, se puede realizar la consulta de los alumnos inscritos a un programa de servicio social por mes de captura, el usuario debe de seleccionar un mes del año de una lista desplegable, el sistema mostrará en una tabla los datos de los alumnos como su nombre, número de cuenta, número de expediente y clave del programa de servicio social al que están inscritos. Si no hubo alumnos inscritos en el mes seleccionado, el sistema no mostrará información.

TAREA 9: Consulta de número de alumnos por sistema, área de estudio y plan de estudios en función del sistema seleccionado
En un apartado dentro de la opción “alumnos” del menú, se puede realizar la consulta del número de alumnos por sistema, área de estudio y plan de estudios de acuerdo al sistema seleccionado. El usuario seleccionará el tipo de sistema de una lista, el cual será el parámetro para mostrar el resto de la información y el sistema mostrará en una caja de texto, el total de alumnos que están inscritos al sistema seleccionado, en una tabla mostrará las áreas de estudio junto con el número de alumnos por cada área, y en otra tabla los planes de estudios junto con el número de alumnos por cada plan de estudios.

TAREA 10: Consulta del número de programas de servicio social de cada dependencia por Institución
En un apartado dentro de la opción “programas” del menú, se puede realizar la consulta del número de programas de cada dependencia por institución. El usuario escribirá dentro de la caja de texto el nombre de la institución y el sistema mostrará en una tabla el nombre de las dependencias de dicha institución, junto con el número de programas de servicio social por cada dependencia.

TAREA 11: Consulta del número y nombre de los supervisores por dependencia
En un apartado dentro de la opción “supervisores” del menú, se puede realizar la consulta del número y nombre de supervisores por dependencia. El usuario escribirá dentro de la caja de texto el nombre de la dependencia y el sistema mostrará en una caja de texto el número de supervisores y en una tabla el nombre de todos los supervisores por dependencia.

2.3.1.3 PROGRAMACIÓN DE “N CAPAS”

Una vez que se establecieron todas las tareas que se debían realizar, se implementó cada tarea usando el lenguaje de programación C#, mediante la técnica de la programación de “N capas”.

La programación de “N capas” es una arquitectura cliente-servidor, que tiene como objetivo principal, separar la lógica de negocios de la lógica de diseño, un ejemplo de ello es separar la capa de datos de la capa de presentación al usuario.

Entre las ventajas que existe en utilizar esta técnica, es que el desarrollo se lleva a cabo en varios niveles, en caso de que se requiera realizar un cambio, únicamente se realiza en el nivel deseado sin tener que realizarlo entre todo el código mezclado.

El diagrama de la arquitectura de “N capas” que se utilizó, es el que se muestra en la figura 2.29:

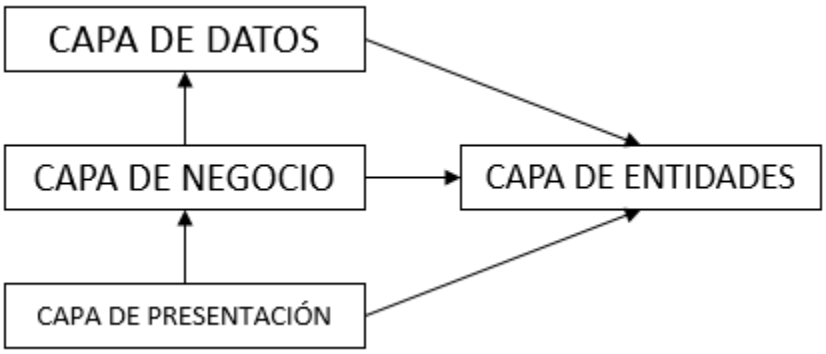


FIGURA 2.29 Diagrama de la arquitectura de “N capas”

Las capas de presentación, de negocio y de datos son bibliotecas de clases creadas, donde se usó código en C#.

FIGURA 2.30 Capas

Capa de presentación. Es la capa que ve el usuario, le muestra la información solicitada y captura la información que el usuario ingresa. La capa de presentación es conocida con el nombre de interfaz gráfica, en la aplicación web, son el menú de opciones y los formularios que el usuario utiliza para almacenar la información de los alumnos, supervisores, asesores y programas de servicio social en la base de datos. Hace referencia a la capa de negocio a la que se le envían los requerimientos del usuario como el almacenar, buscar, modificar o eliminar registros de la base de datos; y también hace referencia a la capa de entidades.

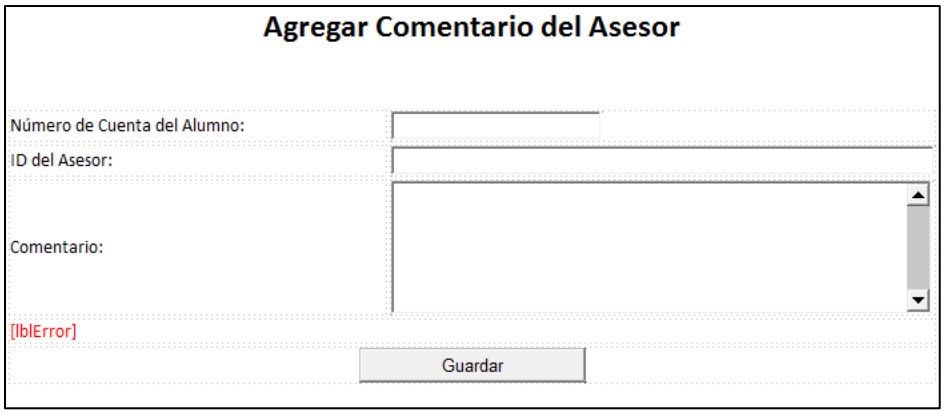


FIGURA 2.31 Ejemplo de formulario

Capa de negocio. Es la capa que recibe las peticiones del usuario, provenientes de la capa de presentación y envía las respuestas tras el proceso, también hace referencia a la capa de datos para solicitarle al manejador de datos, almacenar o recuperar datos. El nombre de capa de negocio se debe porque es la capa en donde se establecen las reglas que deben de cumplirse, contiene los objetos definidos por las clases que se utilizan una y otra vez de acuerdo a las necesidades del usuario.

```
//Para Buscar un Alumno
2 references | 0 excepciones
public static entAlumno BuscarAlumnos(String numCuenta)
{
    return daoAlumno.BuscarAlumnos(numCuenta);
}

//Para Agregar un Nuevo Alumno
1 reference | 0 excepciones
public static int AgregarAlumnos(entAlumno obj)
{
    return daoAlumno.AgregarAlumnos(obj);
}

//Para Modificar un Alumno
1 reference | 0 excepciones
public static int ModificarAlumnos(entAlumno obj)
{
    return daoAlumno.ModificarAlumnos(obj);
}

//Para Eliminar un Alumno
1 reference | 0 excepciones
public static int EliminarAlumnos(entAlumno obj)
{
    return daoAlumno.EliminarAlumnos(obj);
}
```

FIGURA 2.32 Ejemplos de objetos

Capa de Datos. Es donde se encuentran alojados todos los datos y es la capa que se encarga de acceder a ellos. Se realiza la conexión a la base de datos y llama a los procedimientos almacenados que reciben las solicitudes de almacenamiento o recuperación de la información. Hace referencia a la capa de entidades. Los métodos que se encuentran en dicha capa son todas las funciones que realiza el sistema, esto engloba el insertar, buscar, modificar o eliminar registros de la base de datos.

```
public static entAlumno BuscarRegistroSS(String numCuenta)
{
    entAlumno obj = null;
    SqlCommand cmd = null;
    SqlDataReader dr = null;
    try
    {
        Conexion cn = new Conexion();
        SqlConnection cnx = cn.conectar();
        cmd = new SqlCommand("BuscarRegistroSS", cnx);
        cmd.Parameters.AddWithValue("@numCuenta", numCuenta);
        cmd.CommandType = CommandType.StoredProcedure;
        cnx.Open();
        dr = cmd.ExecuteReader();
        obj = new entAlumno();
        dr.Read();
        obj.numCuenta = dr["numCuenta"].ToString();
        obj.claveProg = dr["claveProg"].ToString();
        obj.numExpediente = dr["numExpediente"].ToString();
        obj.numCaptura = dr["numCaptura"].ToString();
        obj.mesCaptura = dr["mesCaptura"].ToString();
        obj.nomCapturador = dr["nomCapturador"].ToString();
        obj.sistema = dr["sistema"].ToString();
        obj.semestre = dr["semestre"].ToString();
        obj.fechaInicio = dr["fechaInicio"].ToString();
        obj.fechaFinal = dr["fechaFinal"].ToString();
        obj.cartaLiberacion = dr["cartaLiberacion"].ToString();
        obj.articulo = dr["articulo"].ToString();
        obj.observaciones = dr["observaciones"].ToString();
    }
    catch (Exception e)
    {
        obj = null;
    }
    finally
    {
        cmd.Connection.Close();
    }
    return obj;
```

FIGURA 2.33 Ejemplo del método para buscar el registro de un alumno de servicio social

Capa de entidades. Es la capa que contiene todos los atributos de las clases creadas. Dichos atributos son cada una de las columnas de las tablas creadas en la base de datos.

```
public String mesCaptura { get; set; }
7 references | 0 excepciones
public String nomCapturador { get; set; }
7 references | 0 excepciones
public String sistema { get; set; }
8 references | 0 excepciones
public String planEstudios { get; set; }
8 references | 0 excepciones
public String areaEstudio { get; set; }
7 references | 0 excepciones
public String semestre { get; set; }
7 references | 0 excepciones
public String fechaInicio { get; set; }
7 references | 0 excepciones
public String fechaFinal { get; set; }
7 references | 0 excepciones
public String cartaLiberacion { get; set; }
```

FIGURA 2.34 Ejemplos de atributos

2.3.2 VERIFICACIÓN Y VALIDACIÓN

Aunque el proceso de verificación y validación es un proceso que se lleva a cabo durante y después de la implementación, en el desarrollo de la aplicación web, se llevó a cabo después del proceso de integración. Cabe resaltar que se mencionó en la elaboración de las tareas, que la presentación de cada tarea al representante de los usuarios era nada más de manera demostrativa, justo en este proceso de presentación con pruebas de unidades ya realizadas previamente, se llevaba a cabo el proceso de verificación y validación.

El programa que se está desarrollando debe de ser comprobado para asegurar que satisface su especificación y tiene la funcionalidad esperada por el usuario. La verificación y la validación es el nombre que se le da a los procesos de análisis y pruebas, a menudo se llegan a confundir ambos conceptos, pero no son lo mismo, una diferencia entre ambos conceptos es la siguiente:

- **Validación:** ¿Estamos construyendo el producto correcto?
- **Verificación:** ¿Estamos construyendo el producto correctamente?

Lo anterior significa que la verificación es para comprobar que el software está de acuerdo a su especificación, satisface los requerimientos funcionales y no funcionales. La validación es asegurar que el sistema satisface las expectativas del cliente.

El objetivo del proceso de verificación y validación, es establecer la seguridad de que el sistema está hecho para su propósito. En el proceso existen dos aproximaciones complementarias para el análisis y comprobación de los sistemas:

1. Las inspecciones de software, analizan y comprueban las representaciones del sistema tales como el documento de requerimientos, los diagramas de diseño y el código fuente del programa. Las inspecciones de software y los análisis automáticos, son técnicas de

verificación y validación estática, ya que no se necesita ejecutar el software en una computadora.

2. Las pruebas del software, implican ejecutar una implementación del software con datos de prueba. Las salidas del software y su entorno operacional, es examinado para comprobar que funciona como es requerido. Las pruebas son una técnica dinámica de verificación y validación.

El sistema solo puede probarse cuando ya se cuenta con un prototipo o una versión ejecutable. Las funcionalidades se pueden probar conforme se van añadiendo al sistema, por lo que no tiene que realizarse una implementación completa antes de que comiencen las pruebas.

El proceso de verificación y validación únicamente se llevó a cabo realizando las pruebas del software, en la demostración del funcionamiento de cada una de las tareas, se aplicó esta técnica. En este caso, la técnica de inspección de software no se llevó a cabo ya que el representante de los usuarios quien era el jefe de la jefatura de servicio social, la persona que veía las demostraciones de las tareas, no tenía conocimiento en el área, ni tampoco existía personal entre ellos que participara en cada una de las etapas del proceso de desarrollo de la aplicación web para que esta técnica se llevara a cabo.

2.3.3 DISEÑO DE CASOS DE PRUEBAS

Las pruebas de sistema son fundamentales en XP, en la que se ha desarrollado un enfoque que reduce la probabilidad de producir nuevos incrementos del sistema y se introduzcan errores en el software existente. Las características de las pruebas en XP son:

- Desarrollo previamente probado.
- Desarrollo de pruebas incrementales a partir de los escenarios.
- Participación del usuario en el desarrollo de las pruebas y en la validación.
- El uso de bancos de pruebas automatizados.

Para cada tarea que representa una función del sistema se debe de diseñar una prueba de unidad. Cada tarea puede tener más de una prueba que verifique la implementación descrita en la tarea. Las **pruebas de aceptación**, son el proceso en el que se prueba el sistema utilizando datos del cliente para verificar que cumple con las necesidades reales. El objetivo del proceso de diseño de casos de prueba, es crear un conjunto de casos que ayuden a descubrir defectos en el sistema y a mostrar que satisface los requerimientos.

Para diseñar un caso de prueba, se debe de seleccionar un componente del sistema, en este caso, se seleccionaron cada una de las tareas que se desarrollaron y que se mencionaron anteriormente, las pruebas están basadas en las tareas de registro de usuarios, inicio de sesión, ingreso de información, búsqueda, modificación y eliminación, así como de las consultas de datos específicos en la base de datos. Posteriormente se selecciona un conjunto de entradas que ejecutan dicha característica, se documentan las salidas esperadas, y se diseña una prueba, que pruebe que las salidas esperadas son las mismas.

Las aproximaciones existentes que pueden seguirse para diseñar los casos de prueba son:

- **Pruebas basadas en requerimientos.** Los casos de prueba se diseñan para probar los requerimientos del sistema. Esta aproximación es utilizada en la etapa de pruebas del sistema. Para cada requerimiento se identifican casos de prueba que puedan demostrar que el sistema satisface el requerimiento.
- **Pruebas de particiones.** Se identifican las particiones de entrada y salida y se diseñan las pruebas para que el sistema ejecute entradas de todas las particiones y genere salidas en todas las particiones. Las particiones son grupos de datos que tienen características comunes.
- **Pruebas estructurales.** Se utiliza el conocimiento de la estructura del programa para diseñar pruebas que ejecuten todas las partes del programa. Cuando se prueba un programa se debe intentar ejecutar cada sentencia al menos una vez.

La aproximación que se utilizó para el diseño de casos de prueba, fue la "basada en requerimientos", el uso del método de la programación extrema implica elaborar para cada tarea sus respectivas pruebas para poder entregarla cada cierto período. Las pruebas para cada una de las tareas que se deben de ejecutar de manera consecutiva, se mencionan enseguida:

Prueba 1: Prueba de la validez de los datos del usuario para registrarse
Entrada: Cadena de caracteres menor o igual a lo solicitado en las indicaciones para realizar el registro de usuarios. **Pruebas:** Comprobar que se ingresó en todas las cajas de texto una cadena de caracteres. Comprobar que la cadena de caracteres no exceda el límite de caracteres solicitados. Comprobar que los datos proporcionados no coincidan con un registro hecho previamente. **Salida:** Si no hay error al hacer el registro, se redireccionará a la página de inicio de sesión. Si hay algún error, indicará con un mensaje que la cadena de caracteres sobrepasa el límite permitido, o un mensaje que indique que faltan campos del formulario por llenar, así como un mensaje que indique que el registro no pudo realizarse.

Prueba 2: Prueba de la validez de los datos del usuario para iniciar sesión
Entrada: Cadena de caracteres que coincida con la información proporcionada en el registro de usuarios **Pruebas:** Comprobar que los datos que se ingresaron coincidan con los proporcionados en el registro de usuarios. Comprobar que ninguno de los campos del formulario de inicio de sesión queden vacíos. **Salida:** Si hay algún error al iniciar sesión, se mostrará un mensaje indicando que alguno de los campos contiene información errónea, así como también si alguno de los campos ha quedado vacío. Si no hay error al iniciar sesión, se redireccionará a la página principal.

Prueba 3: Prueba de validez de la información en el formulario
Entrada: Cadena de caracteres con la información requerida en cada uno de los campos del formulario. **Prueba:** Comprobar que el registro introducido, no coincida con algún otro almacenado en la base de datos. Comprobar que los campos obligatorios no queden vacíos. Comprobar que la cadena de caracteres introducida en algunos campos cumpla con el formato indicado. **Salida:** Si un registro se está repitiendo, se debe de mostrar un mensaje de error que indique que el registro no puede ser almacenado. Se debe de mostrar un mensaje de error si campos obligatorios han quedado vacíos, así como también si la cadena de caracteres no cumple con el formato indicado. Si no hay ningún error, se redireccionará a la página principal.

Prueba 4: Prueba de búsqueda de registros
Entrada: Cadena de caracteres con la información requerida para iniciar la búsqueda del registro **Prueba:** Comprobar que el registro a buscar existe en la base de datos. **Salida:** Si el registro no se encuentra, mostrar mensaje que indique que el registro no fue encontrado. Si el registro es encontrado, mostrar el formulario con la información solicitada.

Prueba 5: Prueba de validez de modificaciones realizadas en los registros
Entrada: Cadena de caracteres en los campos que se deseen llenar o en campos donde la información se quiera modificar. **Prueba:** Comprobar que los campos obligatorios no queden vacíos. Comprobar que la cadena de caracteres introducida en algunos campos cumpla con el formato indicado. **Salida:** Se debe de mostrar un mensaje de error si campos obligatorios han quedado vacíos, así como también si la cadena de caracteres no cumple con el formato indicado. Si no hay ningún error, se redireccionará a la página principal.

Prueba 6: Prueba de validez de eliminación de registros
Entrada: Cadena de caracteres con la información requerida para iniciar la búsqueda del registro. **Prueba:** Comprobar que una vez que se elimina el registro deseado y realizar su búsqueda nuevamente, el registro ya no exista en la base de datos. **Salida:**

Si el proceso de eliminación del registro es exitoso, se mostrará un mensaje indicando que el registro no existe.

Para las consultas de datos en específicos en la base de datos para la elaboración de PRIDES, constancias y el resto de consultas, se indica enseguida una sola prueba que se aplica para las tareas de la 6 a la 11.

Prueba 7: Prueba de la validez de la información en el formulario
Entrada: Cadena de caracteres con la información requerida para iniciar la consulta de datos. **Prueba:** Comprobar que la consulta solicitada se ha realizado correctamente. **Salida:** Conjunto de datos, en base a la cadena de caracteres introducida.

Todas las pruebas que se realizan una vez que se ha desarrollado código pueden a veces mostrar defectos que se deben de eliminar, al proceso de eliminar los defectos se le conoce como depuración. Las pruebas y la depuración son dos procesos distintos, el primero establece la existencia de defectos y la depuración es la localización y corrección de los defectos.

La figura 2.35 muestra el proceso de depuración que se tiene que llevar a cabo:

FIGURA 2.35 Diagrama del proceso de depuración

Los defectos en el código deben de ser localizados y reparados para que el programa cumpla con los requerimientos, una vez hecho esto, es necesario de nuevo realizar pruebas para verificar que los cambios se han realizado correctamente. El proceso de depuración forma parte tanto del desarrollo, como de las pruebas de software.

Al realizar el proceso de la depuración surgen hipótesis sobre el comportamiento y se observa el programa, posteriormente se prueban las hipótesis con el fin de que el defecto pueda ser encontrado. Pueden escribirse también nuevos casos de prueba para localizar el problema. Como ayuda al proceso de la depuración, se pueden utilizar herramientas de depuración interactiva que muestran los valores intermedios de las variables del programa y una traza de las sentencias ejecutadas.

Todas y cada una de las pruebas que se mencionaron anteriormente, se fueron realizando conforme se iba terminando cada una de las tareas, al mismo tiempo que culminaba una, se iba integrando al sistema, ya que cada tarea depende de la anterior para poderse llevar a cabo. Previamente al proceso de prueba del subsistema, se elaboraba otra prueba con datos que se ingresaban a la base de datos, al término de elaborar el procedimiento en SQL, se ejecutaba en el manejador de base de datos para corroborar que la tarea que se estaba por implementar iba a cumplir con el requerimiento establecido.

2.4 INTEGRACIÓN Y PRUEBA DEL SISTEMA

2.4.1 INTEGRACIÓN

En el proceso de integración, todos los subsistemas desarrollados independientemente, se conjuntan para crear un sistema completo. Hay dos formas de hacer la integración, una es utilizando el enfoque "big bang", que consiste en integrar todos los subsistemas al mismo tiempo, la otra forma es el proceso de integración creciente, donde los subsistemas se van integrando uno por uno; este enfoque es mejor debido a las siguientes dos razones:

1. Es imposible por lo general, confeccionar una agenda para el desarrollo de todos los subsistemas de tal forma que todos terminen al mismo tiempo.
2. Reduce el costo de la localización de errores. Si varios subsistemas se integran al mismo tiempo y surge un error durante la prueba, es complicado detectar el error que se encuentre en cualquiera de los subsistemas. En caso contrario, cuando solo se integra un subsistema y en la prueba se encuentra un error, es más fácil localizar el error, puesto estará únicamente en el subsistema que se integró.

Los defectos entre los subsistemas pueden aparecer durante el proceso de la integración del sistema.

Como se mencionó anteriormente, una vez que se terminaba de realizar una tarea, se integraba al sistema, para esto se utilizó el proceso de integración creciente, por la dependencia que hay entre las tareas para que el sistema funcione adecuadamente.

2.4.2 PRUEBA DEL SISTEMA

Una vez que los componentes han sido integrados, se llevan a cabo las pruebas del sistema, las pruebas tienen la finalidad de probar las interfaces entre los componentes y el comportamiento del sistema. Las dos actividades fundamentales de pruebas son, la prueba de componentes (probar cada una de las partes del sistema), y la prueba del sistema (probar el sistema como un todo). En la etapa de la prueba del sistema se debe de establecer que el sistema satisface los requerimientos funcionales y no funcionales, y el comportamiento del sistema no es anormal. Es inevitable que los defectos que no se detectaron durante la prueba de componentes se detecten cuando el sistema completo es probado.

El proceso de pruebas de software tiene dos objetivos:

1. Demostrar al desarrollador y al cliente, que el software satisface los requerimientos. Para cada requerimiento debe de existir al menos una prueba. Algunos sistemas pueden tener una fase de pruebas de aceptación explícita en la que el cliente comprueba que el sistema entregado cumple con su especificación.
2. Descubrir defectos en el software que provocan un comportamiento incorrecto y que no cumple con su especificación. Los defectos deben de ser eliminados como caídas del sistema, interacciones no permitidas con otros sistemas, cálculos incorrectos o corrupción de datos.

El primer objetivo hace referencia a las pruebas de validación en las que se espera que el sistema funcione correctamente usando casos de prueba que reflejen el uso esperado. El segundo objetivo es para la prueba de defectos, en los que los casos de prueba se diseñan para exponer los defectos, una prueba con éxito, es cuando se muestra un defecto que ocasiona que el sistema funcione incorrectamente. El objetivo de las pruebas, es convencer a los clientes que el sistema funciona correctamente y está listo para su uso operacional.

Al usar el modelo en cascada, las pruebas del sistema se llevan a cabo una vez que el sistema está completo.

Para la mayoría de los sistemas existen dos fases distintas de pruebas:

- **Pruebas de integración.** El equipo de pruebas tiene acceso al código fuente del sistema. En dado caso de que se descubra un problema, el equipo de integración intenta encontrar el origen del problema e identificar los componentes que tienen que ser depurados. Esta fase se encarga de encontrar defectos en el sistema.
- **Pruebas de entrega.** Se prueba una versión del sistema que podría ser entregada a los usuarios**.** El equipo de pruebas se encarga de validar que el sistema cumple con los requerimientos y asegura que el sistema es confiable. A las pruebas de entrega se les conoce como pruebas de caja negra, donde el equipo de pruebas se encarga de demostrar que el sistema funciona. Si hay problemas, estos son comunicados al equipo de desarrollo quienes depurarán el programa. Cuando los clientes se involucran en las pruebas de entrega, se les conoce como “pruebas de aceptación”.

2.4.2.1 PRUEBAS DE INTEGRACIÓN

La integración del sistema se realiza a partir de sus componentes, el sistema resultante es probado para encontrar problemas que puedan surgir debido a la integración de componentes. Los componentes que se integran pueden ser componentes comerciales, componentes reutilizables que se han ido adaptando a un sistema en particular o componentes nuevos desarrollados.

Las pruebas de integración comprueban e identifican que los componentes realmente funcionan juntos al añadir código, son llamados correctamente y transfieren datos correctos en el tiempo preciso y a través de sus interfaces. Se recomienda siempre integrar primero los componentes que implementan las funcionalidades que el usuario usará constantemente.

Si surgen errores como se mencionó, la reparación resulta complicada debido a que un grupo de componentes pueden llegar a tener que cambiarse y además la integración y prueba de un nuevo componente puede cambiar el patrón de las interacciones de componentes ya probados. Es importante que cuando se integra un nuevo incremento, se vuelvan a ejecutar pruebas para incrementos previos, y se realicen las pruebas a la nueva funcionalidad del sistema. Al hecho de volver a ejecutar un conjunto de pruebas se les conoce como pruebas de regresión.

Cuando se utiliza el método de la programación extrema, en todas las pruebas se especifican las entradas y las salidas. Un principio básico de la programación extrema es que el conjunto completo de pruebas se ejecute siempre que se ingrese nuevo código, dicho código no puede ser aceptado hasta que todas las pruebas se hayan ejecutado con éxito.

Al término del proceso de haber mostrado cada una de las funciones que realiza la aplicación web al jefe de la Jefatura de servicio social, la aplicación se subió al hosting gratuito que provee SOMEE para seguir realizando pruebas, ingresando desde un navegador a la aplicación web.

Las pruebas de integración que se realizaron, fueron las siguientes:

Prueba de la conexión a la aplicación web

Una vez que se obtuvo un dominio para acceder a la aplicación web, usando un equipo de cómputo, se ingresó a la dirección web, usando el navegador Google Chrome para poder tener una mejor visualización de la aplicación. Con esta prueba se comprobó que es posible ingresar a la aplicación web desde cualquier dispositivo que cuente con conexión a internet.

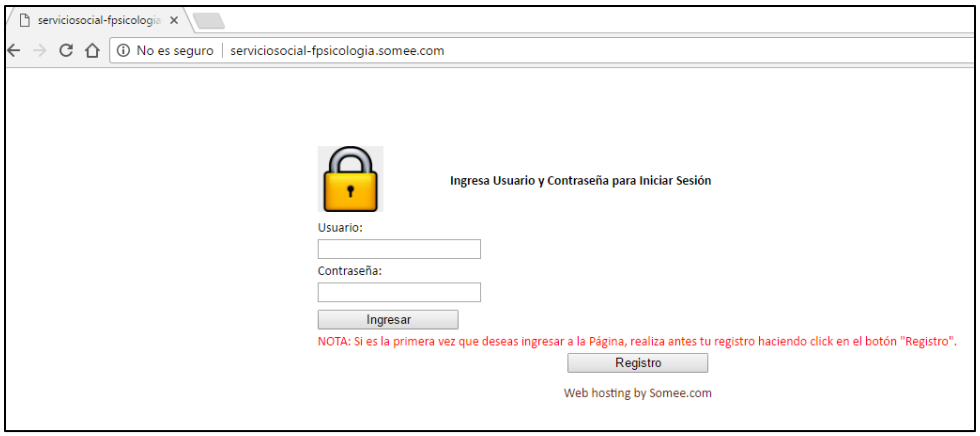


FIGURA 2.36 Página de inicio de sesión de la aplicación web

Prueba de inicio de sesión en la aplicación web

Al iniciar sesión e ingresar a la página principal de la aplicación web exitosamente, se comprobó que la conexión a la base de datos era correcta. En caso contrario que el inicio de sesión no hubiera sido posible, el sistema hubiera mostrado un mensaje de error.

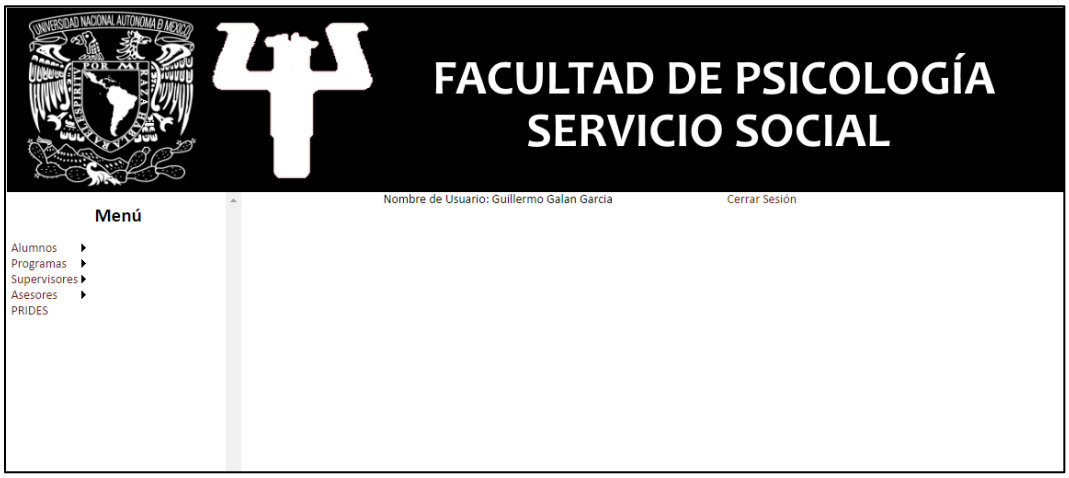


FIGURA 2.37 Página principal de la aplicación web

Para comprobar que la información era almacenada en la base de datos, se ingresaron los comentarios que el asesor puede hacer al alumno asesorado, en la figura 2.38 se muestra una tabla colocada con fines de comprobación, donde se confirma que la información fue almacenada con éxito en la base de datos.

FIGURA 2.38 Prueba de la conexión a la base de datos ingresando información

2.4.2.2 PRUEBAS DE ENTREGA

Las pruebas de entrega también llamadas pruebas funcionales, porque al probador solo le interesa la funcionalidad y no la implementación del software, son el proceso de probar una entrega del sistema que se proporcionará a los clientes. El objetivo de este proceso de entregas, es incrementar la confianza del suministrador, en que el sistema satisface sus requerimientos, si esto se cumple, el sistema se puede entregar como un producto al cliente. Para demostrar que el sistema satisface sus requerimientos se debe de mostrar su funcionalidad especificada, rendimiento y confiabilidad, y no debe de fallar durante su uso.

Las pruebas de entregas se derivan de una especificación del sistema, y es un proceso de pruebas de caja negra. El sistema es tratado como una caja negra donde su comportamiento solo es determinado estudiando sus entradas y sus salidas relacionadas.

El diagrama de la figura 2.39, muestra el modelo de un sistema que se admite en las pruebas de caja negra. El probador es el que presenta las entradas al componente o al sistema y examina las correspondientes salidas. En caso de que las salidas no sean las esperadas, entonces la prueba ha detectado un problema en el sistema. Cuando se realizan las pruebas del sistema, se realizan casos de prueba que provoquen anomalías en él. El objetivo es precisamente seleccionar entradas que muy probablemente generen fallos en la ejecución del sistema.

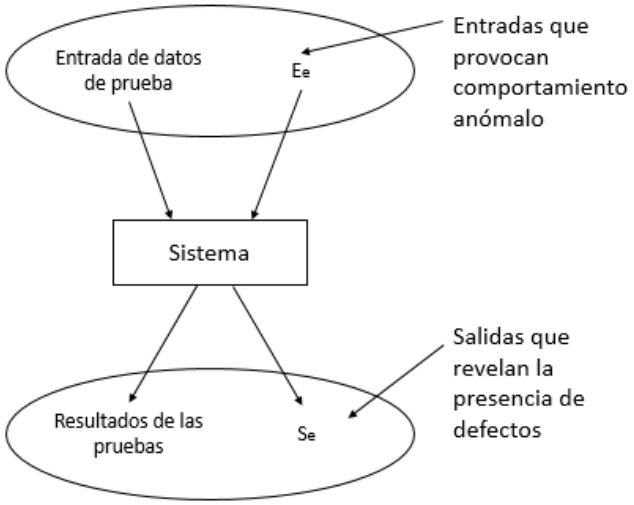


FIGURA 2.39 Pruebas de caja negra

Una de las mejores pruebas para validar que el sistema satisface los requerimientos, es usar las pruebas basadas en escenarios. Estos escenarios que se diseñaron previamente en los que se desarrollaron los casos de prueba, tanto casos que provoquen anomalías en el sistema, como casos en los que se muestre el funcionamiento correcto del sistema. Este conjunto de pruebas debe de estar conformado por entradas válidas e inválidas y que generen salidas válidas e inválidas.

PRUEBA 1. Prueba de la validez de los datos del usuario para registrarse

- **Caso 1:** Ingreso en todas las cajas de texto, una cadena de caracteres para realizar el registro de usuarios.

Si no existe error alguno al realizar el registro, se reedirecciona a la página de incio de sesión. Sin embargo, si algún campo ha quedado vacío, se muestra un mensaje de error indicando que faltan campos por ser llenados.

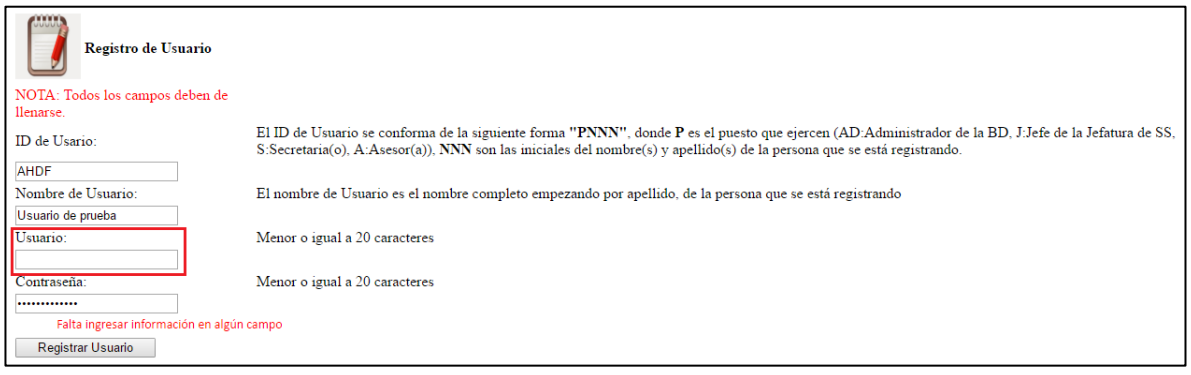


FIGURA 2.40 Prueba que verifica que los campos no queden vacíos

- **Caso 2:** Prueba que verifica que la cadena de caracteres excede el límite permitido.

Si en el campo de “Usuario” o “Contraseña” la cadena de caracteres es mayor a 20 caracteres, el registro no puede realizarse, el sistema envía un mensaje de error indicando que alguno de los campos ya mencionados, excede el número de caracteres permitidos.

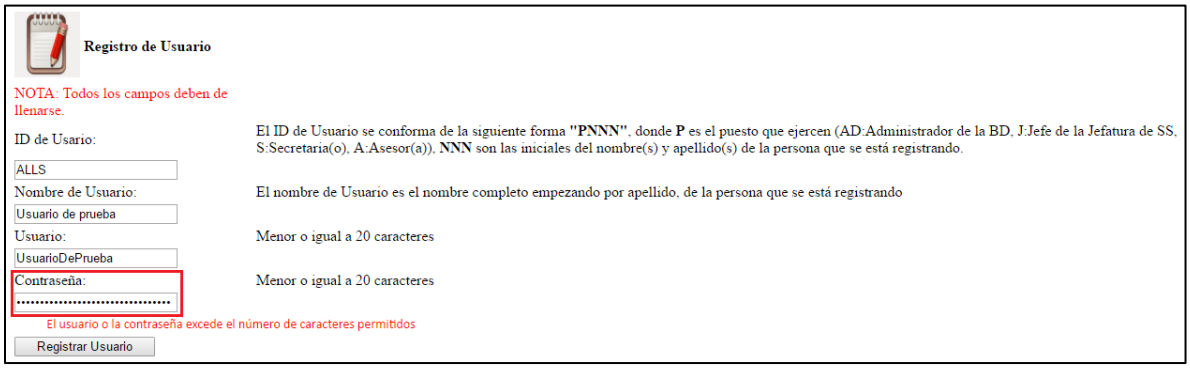


FIGURA 2.41 Prueba que verifica la cadena de caracteres permitida en el campo de “Usuario” y “Contraseña”.

- **Caso 3:** Prueba para comprobar que los datos proporcionados no coincidan con un registro hecho previamente.

Si los datos son idénticos, a un registro previo, se mostrará un mensaje donde se indica que no se puede realizar el registro del usuario.

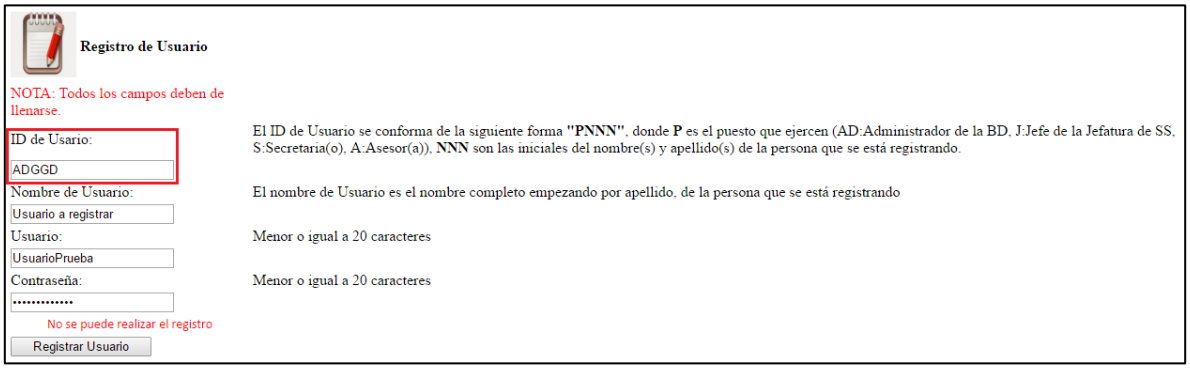


FIGURA 2.42 Prueba que verifica si los datos proporcionados no coinciden con un registro previo.

PRUEBA 2. Prueba de la validez de los datos del usuario para iniciar sesión

- **Caso 1:** Prueba que valida que la información proporcionada para iniciar sesión es errónea.

Si en los datos proporcionados hay un error por falta de un carácter o uno erróneo, el sistema muestra un mensaje indicando que el “Usuario” o la “Contraseña” no son válidos.

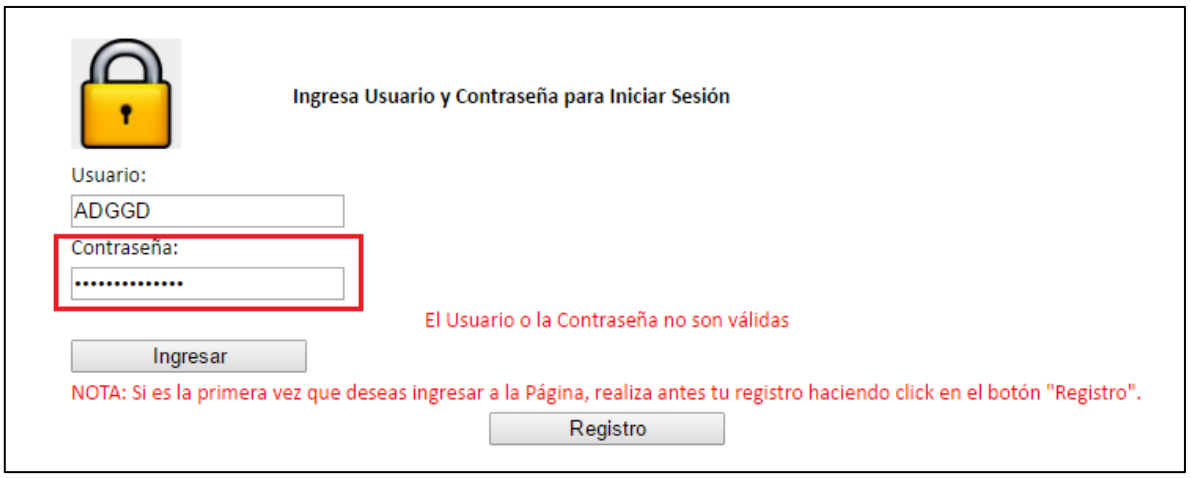


FIGURA 2.43 Prueba que verifica si los datos proporcionados son erróneos

- **Caso 2:** Prueba que verifica que el inicio de sesión es correcto.

Si los datos que se ingresaron para iniciar sesión en la aplicación web son correctos, se redirecciona a la página principal.

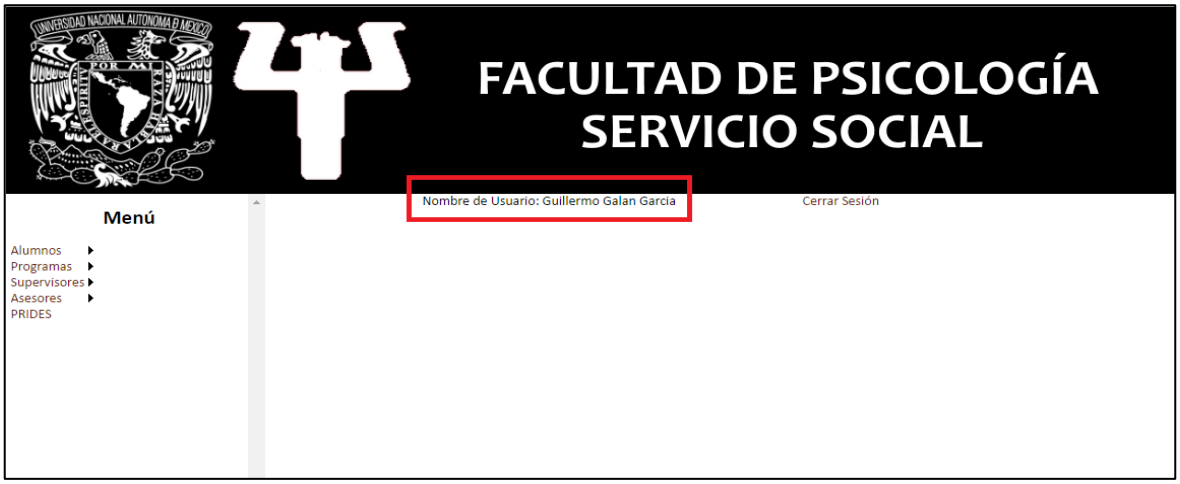


FIGURA 2.44 Prueba que verifica si los datos proporcionados son correctos

- **Caso 3:** Prueba que verifica que ningún campo quede vacío.

Si alguno de los campos de inicio de sesión queda vacío, el sistema muestra un mensaje indicando que falta información en algún campo.

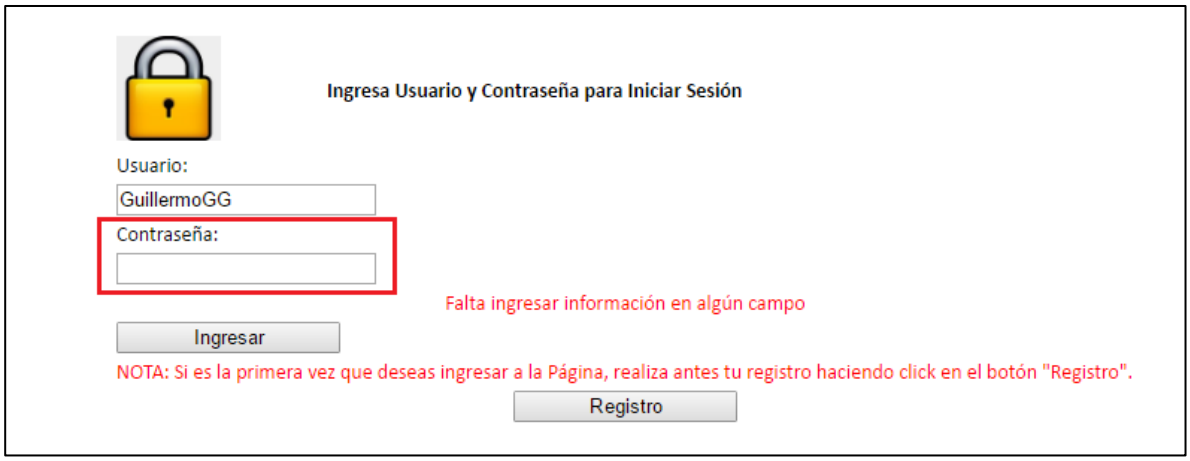


FIGURA 2.45 Prueba que verifica que los campos no queden vacíos

PRUEBA 3. Prueba de validez de la información en el formulario

- **Caso 1:** Prueba que verifica que no exista información repetida.

Si el número de cuenta del alumno, la clave del programa, ID de supervisor, ID asesor y demás, se repiten al almacenar un nuevo registro en la base de datos, el sistema muestra el mensaje indicando que el registro ya existe.

FIGURA 2.46 Prueba que verifica que no existan registros repetidos

- **Caso 2:** Prueba que verifica que ningún campo quede vacío.

Si alguno de los campos "obligatorios" de los formularios queda vacío, el sistema muestra un mensaje indicando que falta información en algún campo.

FIGURA 2.47 Prueba que verifica que los campos obligatorios no queden vacíos

- **Caso 3:** Prueba que verifica que la cadena de caracteres en algunos campos, cumpla con el formato indicado.

Si en algunos campos del formulario no se cumple con el formato indicado, el sistema muestra un mensaje de error indicando que no se cumple con él.

FIGURA 2.48 Prueba que verifica que algunos campos cumplan con el formato indicado

PRUEBA 4. Prueba de búsqueda de registros

Si el registro a buscar en la base de datos no existe, el sistema muestra un mensaje de error indicando que el registro no fue encontrado.

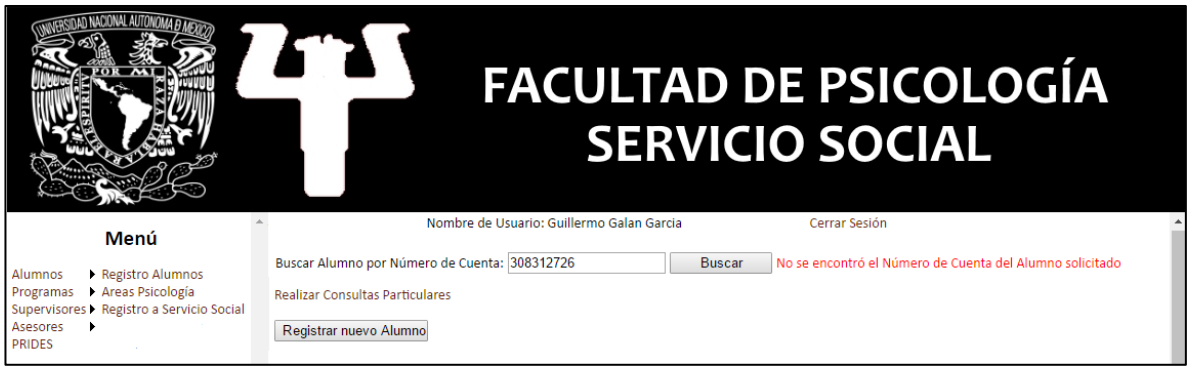


FIGURA 2.49 Prueba que verifica que si el registro a buscar, existe en la base de datos

PRUEBA 5. Prueba de validez de modificaciones realizadas en los registros

- **Caso 1:** Prueba que verifica que ningún campo quede vacío.

Si alguno de los campos "obligatorios" de los formularios queda vacío, el sistema muestra un mensaje indicando que falta información en algún campo y la modificación no puede llevarse a cabo.

FIGURA 2.50 Prueba que verifica que los campos obligatorios no queden vacíos

- **Caso 2:** Prueba que verifica que la cadena de caracteres en algunos campos, cumpla con el formato indicado.

Si en algunos campos del formulario no se cumple con el formato indicado, el sistema muestra un mensaje de error indicando que no se cumple con él y la modificación no puede llevarse a cabo.

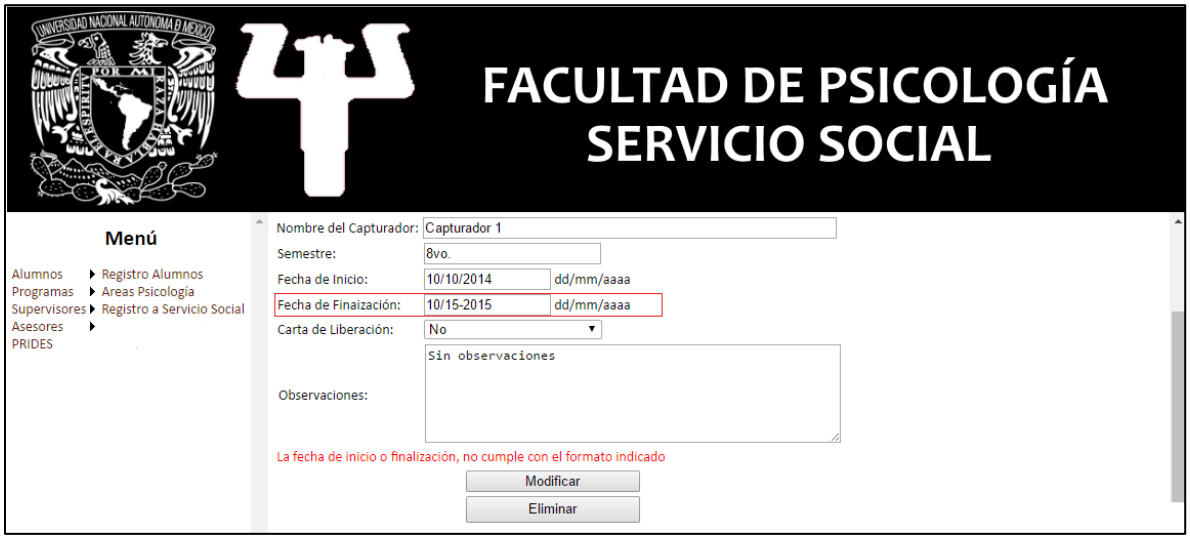


FIGURA 2.51 Prueba que verifica que algunos campos cumplan con el formato indicado

PRUEBA 6. Prueba de validez de eliminación de registros

Una vez que se ha eliminado el registro deseado, y se busca dicho registro, el sistema debe de mostrar un mensaje indicando que el registro solicitado no fue encontrado, garantizando que el registro no existe en la base de datos.

FIGURA 2.52 Registro de la base de datos a eliminar

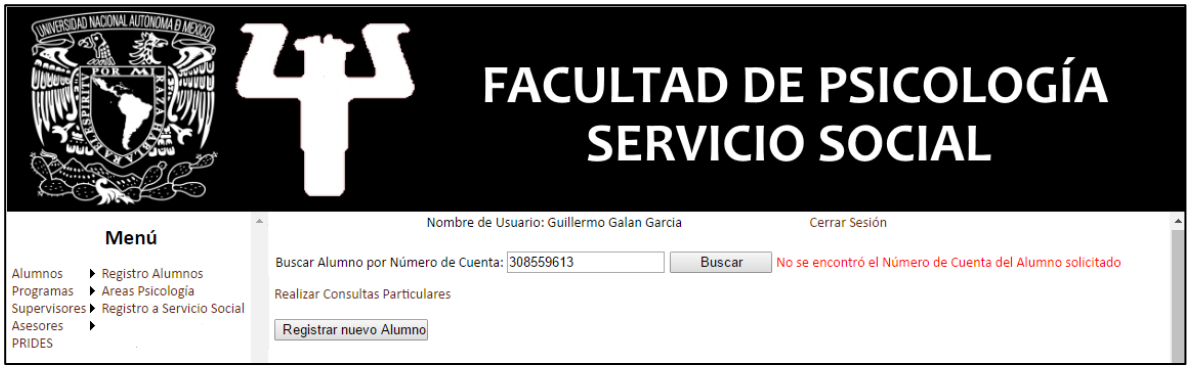


FIGURA 2.53 Prueba que verifica que el registro eliminado ya no se encuentra en la base de datos.

PRUEBA 7. Prueba de la validez de la información en el formulario

- **Consulta de datos para la elaboración de PRIDES:** Si se ha introducido una clave de programa de servicio social correcta, el sistema mostrará el nombre del programa de servicio social, el nombre del supervisor y los datos del alumno que se muestran en la figura 2.54.

FIGURA 2.54 Prueba que verifica la consulta de datos para la elaboración de PRIDES

- **Consulta de datos para la elaboración de constancias para los asesores:** Si se ha introducido el nombre del asesor correctamente, el sistema muestra los datos de los alumnos asesorados por dicho asesor.

FIGURA 2.55 Prueba que verifica la consulta de datos para la elaboración de constancias para los asesores

- **Consulta de datos de los alumnos inscritos a un programa de servicio social por mes de captura:** Si el mes seleccionado contiene expedientes registrados, el sistema muestra los datos del alumno que fueron inscritos a un programa de servicio social en dicho mes.

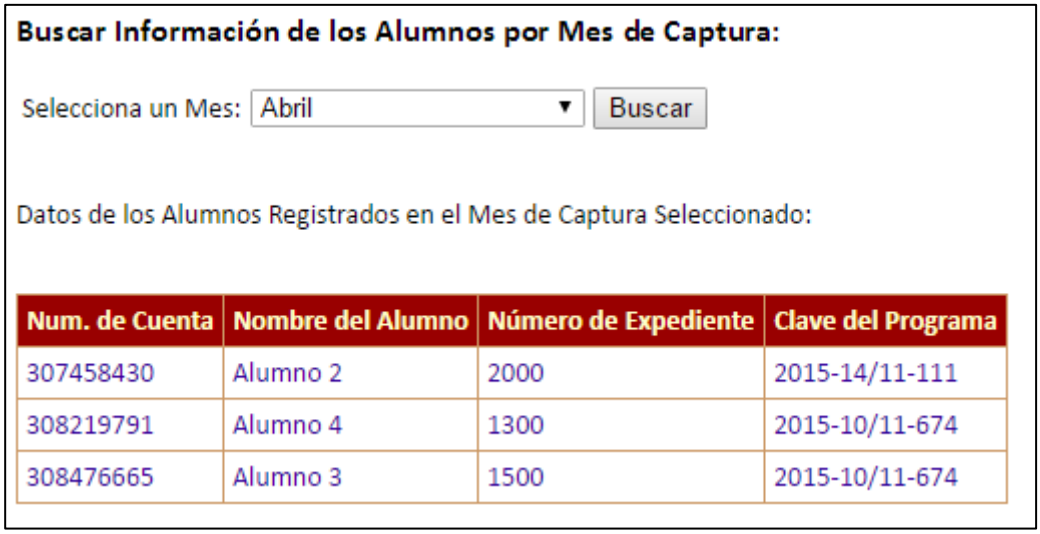


FIGURA 2.56 Prueba que verifica la consulta de datos para mostrar los alumnos inscritos a un programa de servicio social por mes de captura

- **Consulta de número de alumnos por sistema, área de estudio y plan de estudios en función del sistema seleccionado:** Al seleccionar el sistema al que están inscritos los alumnos, ya sea "Escolarizado" o "SUA", se muestra el total de alumnos inscritos, número de alumnos por cada área de estudio y por plan de estudios.

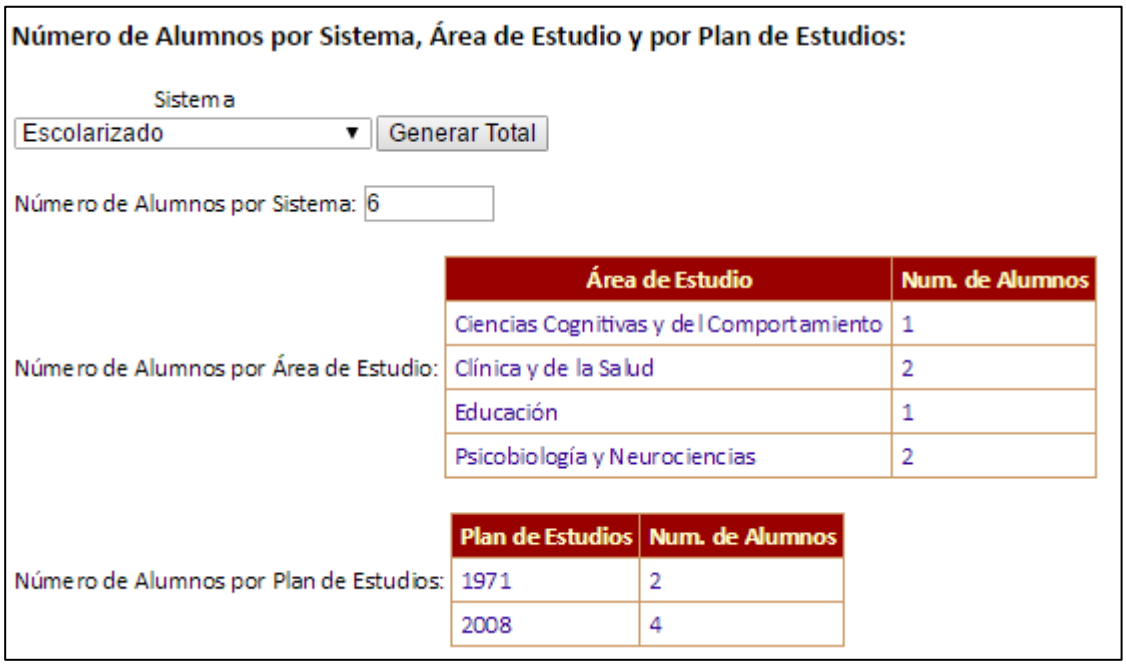


FIGURA 2.57 Prueba que verifica la consulta de datos para mostrar el número de alumnos por sistema, área de estudios y plan de estudios en función del sistema seleccionado.

- **Consulta del número de programas de servicio social de cada dependencia por Institución:** Si se ha introducido la institución que cuenta con programas de servicio social, el sistema muestra los nombres de las dependencias y el número de programas de cada una.

FIGURA 2.58 Prueba que verifica la consulta de datos para mostrar el número de programas de servicio social de cada dependencia por Institución.

- **Consulta del número y nombre de los supervisores por dependencia:** Si se ingresa el nombre de la dependencia correctamente, el sistema muestra el número de supervisores con los que cuenta y sus nombres.

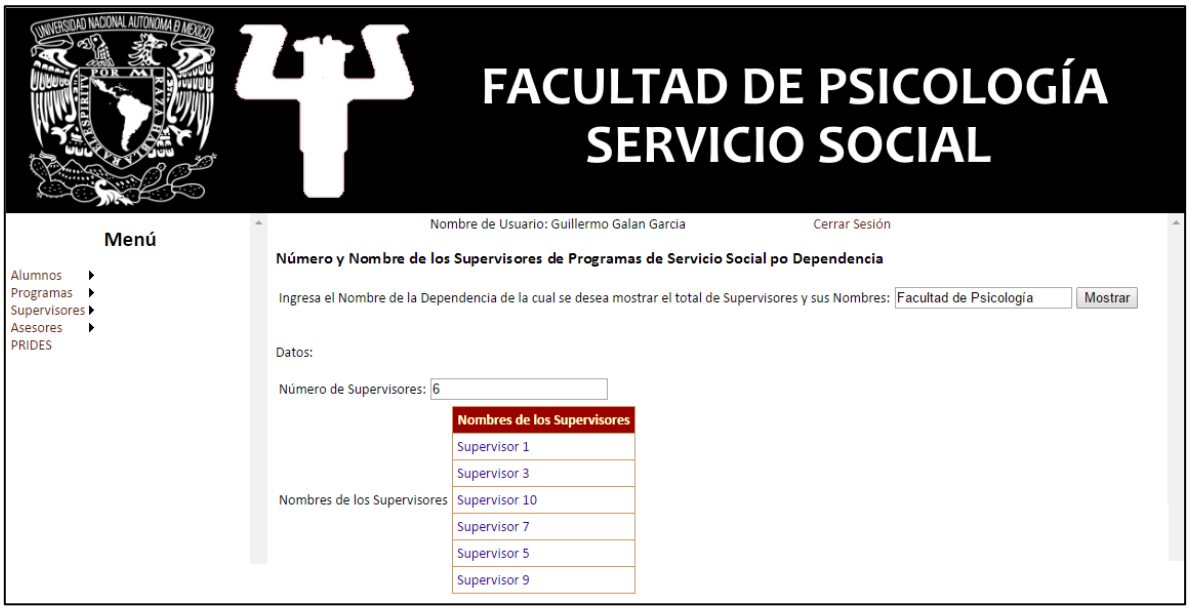


FIGURA 2.59 Prueba que verifica la consulta de datos para mostrar el número y nombre de los supervisores por dependencia.

2.5 FUNCIONAMIENTO Y MANTENIMIENTO

En la etapa del mantenimiento, el sistema sufre cambios para corregir errores en los requerimientos iniciales o para implementar nuevos requerimientos que surgen, aunque los cambios de software se pueden realizar durante o después del desarrollo del sistema.

La evolución que puede tener el sistema, inevitablemente es costosa por las siguientes razones:

- Los cambios que se propongan, tienen que contribuir a los objetivos del sistema.
- Se debe de cuidar que los cambios en los subsistemas no deben de afectar el funcionamiento o comportamiento de otros subsistemas porque son totalmente dependientes entre sí.
- Con el paso del tiempo se corrompe la estructura del sistema original por los cambios realizados.

Además, muchas veces se considera al mantenimiento del software como un proceso laborioso, pero a la vez es un proceso menos problemático que el desarrollo del mismo.

El diagrama de la figura 2.60, muestra el proceso de la evolución del sistema:

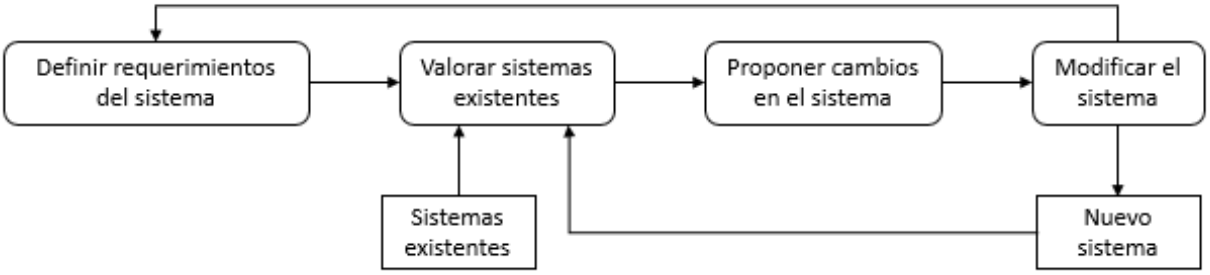


FIGURA 2.60 Proceso de evolución del software

2.5.1 MANTENIMIENTO DEL SOFTWARE

El mantenimiento del software es el proceso de cambiar un sistema después de que ha sido entregado. Lo cambios pueden ser desde corregir errores de código, errores de diseño, errores de especificación o para implementar nuevos requerimientos.

Los tres tipos de mantenimiento de software son:

1. **Mantenimiento para reparar defectos del software.** La corrección de los errores de código son los más baratos de corregir, los errores de diseño son aún más caros porque se requiere reescribir varios componentes de los programas y por último los errores de requerimientos son los más caros porque pueden implicar un rediseño del sistema.
2. **Mantenimiento para adaptar el software a diferentes entornos operativos.** Este tipo de mantenimiento es requerido cuando se cambia el entorno del sistema, como el hardware, el sistema operativo, o algún software de soporte. Las modificaciones deben de ser hechas para que se adapten al nuevo entorno.
3. **Mantenimiento para añadir o modificar las funcionalidades del sistema.** Este mantenimiento surge cuando los requerimientos del sistema cambian por cambios organizacionales o del negocio. El número de cambios es mucho mayor que los otros tipos de mantenimiento.

A los tipos de mantenimiento también se les suele conocer de otra forma, el **mantenimiento correctivo** se utiliza para referirse a la reparación de defectos, el **mantenimiento adaptativo** algunas veces significa la adaptación a un nuevo entorno y adaptar el software a nuevos requerimientos, y el **mantenimiento perfectivo** significa perfeccionar el software, implementando nuevos requerimientos manteniendo la funcionalidad del sistema, pero mejora su estructura y rendimiento.

Los factores que conducen a costos de mantenimiento elevados, son:

- **Estabilidad del equipo.** Es muy común que después de la entrega del sistema, el equipo de desarrollo se disuelva, provocando que el nuevo equipo no entienda del todo el sistema o las razones de las decisiones sobre el diseño. Se requiere tiempo y esfuerzo para comprender el sistema antes de que se realicen cambios en él.

- **Responsabilidad contractual.** El contrato de mantenimiento suele darse con una compañía diferente a la que desarrolló el software, si el equipo que lo desarrolló, no documentó dicho software, el costo por mantenimiento será elevado.
- **Habilidades del personal.** El personal de mantenimiento suele no estar familiarizado con el dominio de la aplicación, al no ser el mantenimiento una etapa de mucha importancia, a menudo se contrata personal principiante, además a veces existen sistemas antiguos que fueron escritos en lenguajes de programación obsoletos, por lo que el personal de mantenimiento puede no tener experiencia en dichos lenguajes y se requiere tiempo para que los aprendan para darle mantenimiento al sistema.
- **Edad y estructura del programa.** La estructura del programa tiende a degradarse por los cambios que se le han realizado o desde el principio nunca hubo una estructura correcta, también existen sistemas que no fueron desarrollados con técnicas modernas de la ingeniería de software, no existe una documentación del sistema o se perdió, lo que provoca que se invierta tiempo en encontrar versiones correctas de los componentes del sistema a modificar.

Para evitar que el costo del mantenimiento en un futuro sea demasiado elevado, se elaboró un manual técnico para la persona que inicie dicha etapa, tenga conocimiento sobre el desarrollo de la aplicación web, además de un manual de usuario para el propio personal de la Jefatura de servicio social que hará uso de dicha aplicación. El manual técnico se le entregó al jefe de la Jefatura de servicio de social, mientras que el manual de usuario puede ser descargado desde la aplicación web. El manual técnico contiene el lenguaje de programación utilizado, la estructura del programa, el diseño de la base de datos y la información del hosting donde fue alojada la aplicación web.

Durante los primeros años en los que se lleve a cabo la etapa de mantenimiento, el tipo de mantenimiento que debe de realizarse, es el mantenimiento para añadir o modificar las funcionalidades del sistema. Debido a los requerimientos que fueron mencionados luego de la entrega, estas modificaciones planean ayudarle al personal del servicio social a tener una mejor estructura de la información almacenada en la base de datos para facilitar aún más la consulta de información.

CAPÍTULO 3. RESULTADOS

A partir de que la aplicación web estuvo en funcionamiento, el personal de la Jefatura de servicio social tuvo un medio de almacenamiento eficiente, que sustituyó completamente al archivo en Excel que hacía la función de una base de datos para almacenar los registros de los alumnos prestadores de servicio social y a la aplicación que se encontraba únicamente instalada en dos computadoras que era de difícil acceso y limitado.

Cualquier miembro del personal de la Jefatura de servicio social, logró tener acceso a la aplicación web en cualquier momento desde cualquier dispositivo que tuviera conexión a internet, y los cambios que se realizaban en la base de datos podían ser vistos por el resto del personal, sin necesidad de repetir el trabajo de captura o modificación de datos como sucedía con la base de datos anterior al archivo de Excel.

El tiempo de captura de la información se redujo al tener los formularios como un medio rápido de captura, y las consultas de los datos dejaron de ser una labor complicada al estar buscando la información requerida en cada una de las filas y columnas del archivo Excel, ya que la aplicación web logró también una mejor organización en los datos, separando la información en las siguientes categorías: alumnos, programas de servicio social, supervisores, y asesores, de esta manera el acceso a la información fue más fácil.

El riesgo de perder la información por causas como fallas en el equipo de cómputo o en el hosting, debido a un mal uso de parte del mismo personal de la jefatura o de los técnicos que dan mantenimiento en el área, como había sucedido en ocasiones anteriores, es casi nulo, aunque el administrador de la base de datos puede realizar una copia de seguridad de la base de datos para prevenir la pérdida de información, ya sea aprovechando el servicio de copia de seguridad que brinda el hosting, o lo puede hacer en la computadora.

CONCLUSIONES

Siguiendo estrictamente el proceso del desarrollo de software, las etapas del diseño de bases de datos y algunas técnicas adicionales, se logró crear la aplicación web que la Jefatura de servicio social solicitó. Para su desarrollo, se aplicaron los conocimientos adquiridos de los temas básicos de bases de datos y de ingeniería de software, así como también de lenguajes de programación, especialmente de aquellos lenguajes utilizados para el desarrollo de aplicaciones web.

El uso del modelo en cascada se adaptó al proceso que requirió el desarrollo de la aplicación web, ya que dicha aplicación se entregó y se puso en funcionamiento una vez que se habían desarrollado cada una de las funciones que realiza, el utilizar la técnica de la programación extrema ayudó a llevar una mejor organización en los requerimientos que se solicitaron, además de diseñar con facilidad las pruebas que se realizarían para garantizar el correcto funcionamiento.

La estructura del código quedó ordenada y legible gracias a la técnica de la arquitectura de “N capas”, que permitirá a futuras personas que lleven a cabo la etapa de mantenimiento entender el código fuente y realizar las modificaciones necesarias o escribir nuevo código.

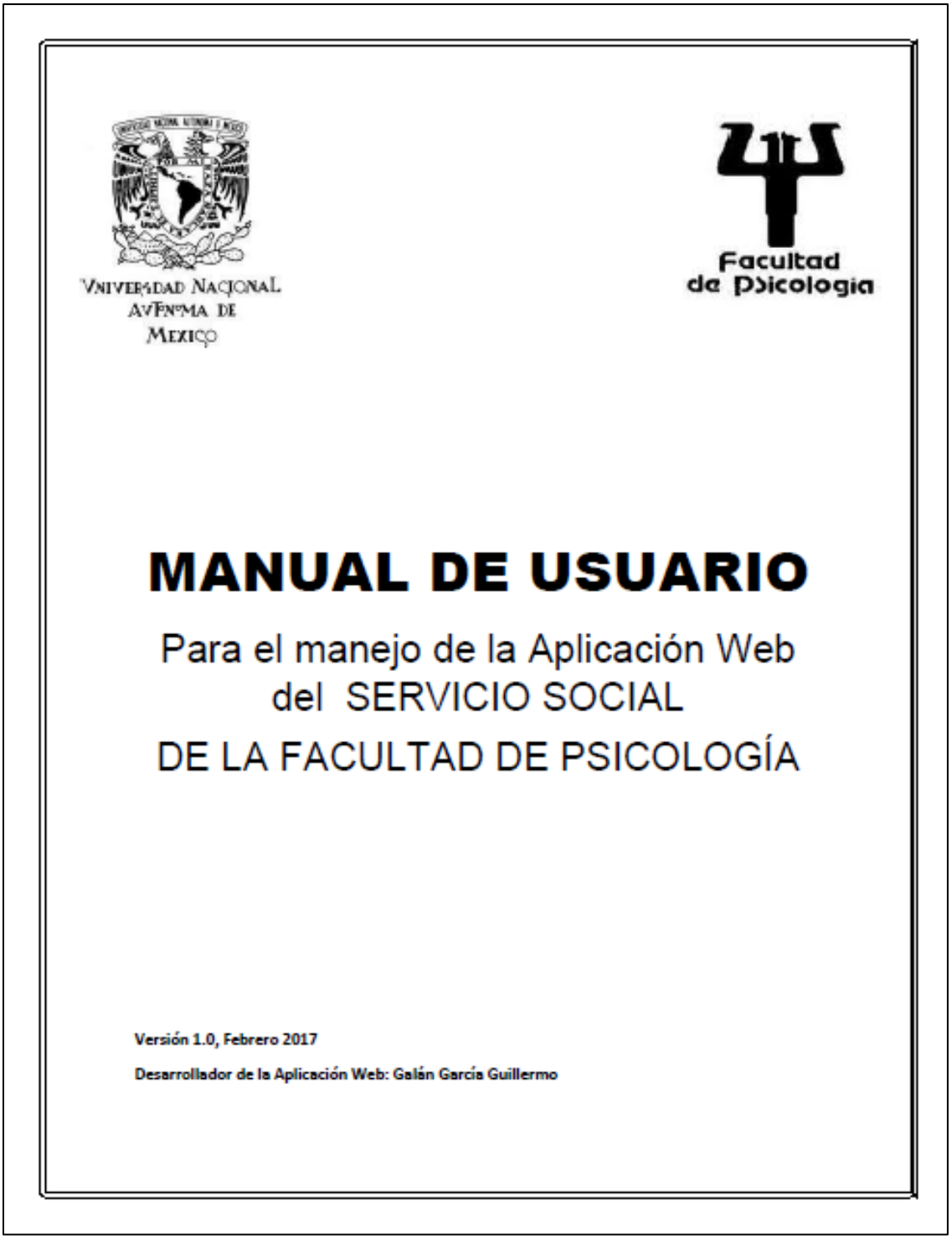
El desarrollo de la aplicación web ayudó al personal de la Jefatura de servicio social a solucionar su problema que llevaba varios años sin poderse resolver por falta de personal de planta, por personas que realizaron anteriormente sistemas con demasiadas deficiencias, así como también de difícil acceso y manejo.

La aplicación web aún puede mejorarse, afinando ciertas funciones incluidas, así como añadiendo funciones adicionales que el personal de la Jefatura de servicio social, requiera en un futuro.

Se logró adquirir mayor habilidad en el desarrollo de aplicaciones web, así como de aprender y aplicar el proceso formal que requiere el desarrollo de un sistema, que obligan a uno a llevar un orden estricto que requiere mucha disciplina.

APÉNDICES

Manual de usuario para el manejo de la aplicación web.



VNIVERSIDAD NACIONAL
AVFNOMA DE
MEXICO

Facultad
de Psicología

MANUAL DE USUARIO

Para el manejo de la Aplicación Web
del SERVICIO SOCIAL
DE LA FACULTAD DE PSICOLOGÍA

Versión 1.0, Febrero 2017

Desarrollador de la Aplicación Web: Galán García Guillermo

Manual técnico para el mantenimiento y actualización de la aplicación web.

VNIVERSIDAD NACIONAL
AVTNOMA DE
MEXICO

Facultad
de Psicología

MANUAL TÉCNICO

Para el mantenimiento y actualización de la Aplicación Web del SERVICIO SOCIAL DE LA FACULTAD DE PSICOLOGÍA

Versión 1.0, Febrero 2017

Desarrollador de la Aplicación Web: Galán García Guillermo

BIBLIOGRAFÍA

SOMMERVILLE, Ian. *Ingeniería del software*. Madrid: Pearson Addison Wesley, 2005.

ARELLANO MENDOZA, Lucila P. y Luciralia Hernández Hernández. *Manual de prácticas de laboratorio de Bases de datos*. México: Universidad Nacional Autónoma de México, Facultad de Ingeniería, 2010.

GÓMEZ FUENTES, María del Carmen. *Notas del curso Bases de datos*. México: Universidad Autónoma Metropolitana, 2013.

SILBERSCHATZ, Abraham, Henry F. Korth y S. Sudarsham. *Fundamentos de Bases de datos*. España: McGraw-Hill, 2002.

DATE, D. J. *Introducción a los sistemas de bases de datos*. México: Pearson Educación, 2001.

MOQUILLAZA HENRÍQUEZ, Santiago Domingo, Hugo Vega Huerta y Luis Guerra Grados. *Programación en N capas*. España: Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistema e Informática, 2010.

MESOGRAFÍA

Romero, V. F. (s.f.). *Diseño Físico de Bases de Datos* [archivo PDF]. Recuperado de: http://av.iesromerovargas.com/pluginfile.php/982/mod_resource/content/4/T3.pdf

García, S. B. (s.f.). *Diseño de Bases de Datos Relacionales* [archivo PDF]. Recuperado de: https://cursos.aiu.edu/base%20de%20datos%20SOG/Sesi%C3%B3n%208.pdf

Montero, F. (s.f.). *Lenguaje SQL* [archivo PDF]. Recuperado de: http://www.v-espino.com/~chema/daw1/tutoriales/SQL3.pdf

Barzanallana A. R. (s.f.). *Desarrollo de Aplicaciones Web* [archivo PDF]. Recuperado de: http://www.um.es/docencia/barzana/DAWEB/Lenguaje-de-programacion-HTML-2.pdf

González S. J. A. (s.f.). *Lenguaje de Programación C#* [archivo PDF]. Recuperado de: http://users.dsic.upv.es/~jlinares/csharp/lenguajeCsharp.pdf

Besteiro M. (s.f.). *Introducción a la programación con C#* [archivo PDF]. Recuperado de: http://www.ehu.eus/mrodriguez/archivos/csharppdf/Lenguaje/HolaMundoConCSharp.pdf

García V. (s.f.). *Curso completo de HTML* [archivo PDF]. Recuperado de: http://es.tldp.org/Manuales-LuCAS/doc-curso-html/doc-curso-html.pdf

All Technology-JC. (2014, Febrero 21). Crear Sitio Web con Visual Studio 2010 ASP.NET con Base de Datos en SQL SERVER 2008 [Archivo de video]. Recuperado de: https://www.youtube.com/watch?v=771KavoGpCY

All Technology-JC. (2013, Diciembre 24). Crear un programa completo en Visual Studio 2010 con Base de Datos en SQL SERVER 2008 [Archivo de video]. Recuperado de: https://www.youtube.com/watch?v=zt8X81TUjSI&t=1017s

All Technology-JC. (2014, Agosto 13). Subir aplicación ASP.NET con base de datos SQL en un hosting gratuito [Archivo de video]. Recuperado de: https://www.youtube.com/watch?v=AX6N4se0ZWU