10 Modelo de Pruebas

Probar un producto es relativamente independiente de la metodología de desarrollo utilizada para construirlo.
Existen diversos tipos de pruebas aplicados durante las diferentes actividades del proceso de desarrollo. Estas pruebas requieren de tiempo y presupuesto adicional, pudiendo llegar a significar entre un 30% y un 50% del costo total de desarrollo. Por tal motivo, el modelo de pruebas debe ser planificado con anticipación y de manera integral junto con el propio desarrollo del sistema. Es un error pensar que las pruebas son la última actividad del desarrollo ya que no se puede lograr software de alta calidad sólo mediante pruebas finales y depuraciones. Las pruebas deben hacerse en paralelo al desarrollo del sistema, teniendo pruebas finales únicamente como certificación final de la calidad del producto y no como la oportunidad para encontrar errores. Encontrar errores al final del desarrollo es bastante problemático dado que requerirá regresar a etapas anteriores para resolverlos. Se considera que "evitar defectos" es más poderoso que "remover defectos".

10.1 Definición de Conceptos

Las siguiente definiciones de IEEE (1983) pueden utilizarse para definir ciertos conceptos conocidos de manera informal como “bugs”:

?? Una *falla* (*failure*) ocurre cuando un programa no se comporta bien, siendo la falla una propiedad (estadística) de un sistema en ejecución.
?? Una *falta* (*fault*) existe en el código del programa, el cual si se presenta, puede ocasionar una *falla* (*failure*). No puede haber una *falta* si el programa no puede *fallar* (*fail*).
?? Un *error* es una acción humana que resulta en que un software contenga una falta, por lo cual un *error* puede significar la inclusión de una *falta* en el programa, haciendo que el sistema *falle*.

Un aspecto importante con los conceptos anteriores es que no se puede garantizar ni probar que un sistema jamás *falle*, solo se puede demostrar que contiene *faltas*. En otras palabras, no encontrar faltas no significa que la prueba haya sido exitosa. Se debe considerar una prueba como exitosa sólo si esta ha encontrado faltas. Sin embargo, pruebas exitosas significarán que no se ha desarrollado un buen sistema. Dada la dificultad de probar un sistema y de encontrar faltas, el encargado de encontrar las faltas en el código es generalmente una persona distinta al desarrollador del sistema. Esto también significa un costo adicional en el desarrollo del sistema, por lo cual a veces sólo se prueban las partes principales del sistema.
Es un fenómeno muy conocido que cuando se corrigen las faltas detectadas, se introducen nuevas faltas en el sistema, requiriendo probar nuevamente todo el sistema completo, esperando que el número de faltas introducidas sea menor que el número anterior. Según Levendel (1990), generalmente se introduce una nueva falta por cada tercera falta corregida.

10.2 Tipos de Pruebas

Los tipos de pruebas se dividen de manera general en pruebas de *verificación* y *validación*. En el caso de la *verificación* se revisa si el resultado corresponde a la especificación del sistema, en otras palabras, si se está construyendo el sistema correctamente, algo por si sólo no garantiza la satisfacción de los clientes. En el caso de la *validación* se revisa si el resultado es realmente lo que el cliente quería, en otras palabras, si se está construyendo el sistema correcto de manera que tanto la especificación como el resultado sean los correctos. En este capítulo nos concentraremos principalmente en la verificación del sistema. La validación del sistema debiera hacerse durante la especificación inicial del sistema a través de prototipos que deben ser aprobados por el cliente y que correspondan a la funcionalidad deseada. El sistema debe validarse continuamente durante el proceso de desarrollo del sistema de manera siempre corresponda con lo especificado. La validación se basa en el modelo de casos de uso.
Existen también diferentes técnicas y niveles de pruebas que pueden aplicarse. Estas se describen en las siguientes secciones.

Técnicas de Pruebas

Las técnicas utilizadas para las pruebas son muy variadas incluyendo las siguientes:

?? **Prueba de Regresión** tiene como propósito verificar el sistema luego de haber hecho cambios, por ejemplo después de corregir una falta, de manera que se mantenga la funcionalidad especificada originalmente.
?? **Prueba de Operación** tiene como propósito verificar el sistema en operación por un período largo de tiempo bajo condiciones normales de uso. Este tipo de prueba mide la *confiabilidad* (*reliability*) del sistema.
?? **Prueba de Escala Completa** tiene como propósito verificar el sistema en su carga máxima asignando los parámetros a su valor límite, interconectando el sistema con un máximo de equipos y usuarios simultáneos. Un

extremo de esto es la *prueba de estrés* (*stressing*) que significa que se prueba el sistema en los límites extremos para ver que tanto aguanta y si ocurre algún tipo de falla.

?? **Prueba de Rendimiento** (*performance*) o **Prueba de Capacidad** tiene como propósito medir la capacidad de procesamiento del sistema bajo diferentes cargas, incluyendo espacio de almacenamiento y utilización del CPU. Los valores medidos se comparan con los valores requeridos.

?? **Prueba de Sobrecarga** tiene como propósito ver cómo se comporta el sistema cuando se le aplica una sobrecarga, más allá que de las pruebas de escala completa y rendimiento. Aunque no se puede esperar que el sistema soporte estas pruebas, este debiera ejecutar correctamente, sobrevivir a picos de carga, evitando que ocurra una catástrofe. Es siempre importante saber en que momento y de que manera cae el rendimiento del sistema.

?? **Prueba Negativa** tiene como propósito medir el estrés del sistema en situaciones inesperadas, como casos de uso que normalmente no serían invocados simultáneamente. El sistema se usa intencionalmente y sistemáticamente de manera incorrecta. Este maltrato debe ser cuidadosamente planeado para probar aspectos especialmente críticos.

?? **Prueba Basada en Requisitos** o **Prueba de Casos de Uso** tiene como propósito hacer pruebas basadas directamente en la especificación de requisitos. Pueden utilizarse los mismos casos de uso originales como casos de prueba. También pueden hacerse pruebas para verificar las especificaciones de rendimiento o de escala completa. Se busca verificar que el sistema final cumple con las especificaciones funcionales descritas por los casos de uso originales.

?? **Pruebas Ergonómicas** tiene como propósito probar los aspectos ergonómicos del sistema, en otras palabras, las interfaces hombre-máquina en el caso de que estas existan. Por ejemplo, se prueba si las interfaces son consistentes con los casos de uso a los cuales corresponden o entre diferentes casos de uso, si los menús son lógicos y legibles, si los mensajes del sistema son visibles, si se puede entender los mensajes de falla, etc.

?? **Prueba de Documentación de Usuario** tiene como propósito probar la documentación de usuario, incluyendo el manual de usuario y documentación de mantenimiento y servicio. Se prueba que los manuales y comportamiento del sistema sean consistentes entre si, que los manuales sean legibles, que tengan una buena redacción y en general que sean comprensibles.

?? **Prueba de Aceptación** o **Prueba de Validación** tiene como propósito lograr una revisión final por parte de la organización que solicitó el sistema, siendo esto a menudo la validación del sistema. El sistema se prueba en su ambiente real por un período largo de tiempo. Cuando se termina la prueba, se toma la decisión si se acepta o no el producto. Este tipo de prueba es a veces conocida como *prueba alfa*. Si no existe un cliente particular que haya solicitado el sistema, por ejemplo en el caso de un producto de software de venta al público, a menudo se hace una *prueba beta*. Esto significa que antes de enviarlo al público en general, el producto es probado por clientes seleccionados que utilizan el sistema y reportan fallas experimentadas.

Nivel de Pruebas

Existen principalmente tres niveles para la aplicación de las diversas técnicas de pruebas:

?? **Prueba de Unidad** donde solamente una unidad es probada como tal, típicamente una clase, paquete de servicio, o subsistema.

?? **Prueba de Integración** donde se verifica que las unidades trabajen juntas correctamente. Pruebas de unidad e integración pueden ser hechas mediante casos de uso de pruebas, los cuales pueden ser aplicados a clases, paquetes de servicio, subsistemas y el sistema completo.

?? **Prueba de Sistema** donde se prueba el sistema completo o la aplicación como tal. Se toma el punto de vista del usuario final y los casos de uso de pruebas ejecutan acciones típicas del usuario.

Prueba de Unidad

Una prueba de unidad es la prueba de más bajo nivel. En un sistema tradicional una prueba de unidad es a menudo una prueba de procedimientos o subrutinas. En un sistema orientado a objetos, las pruebas de unidad se hacen a un nivel más alto, a partir de las clases. Por lo tanto, una prueba de unidad en un sistema orientado a objetos es más complejo que en sistemas estructurados, dado que los objetos involucran estados encapsulados que puede afectar el comportamiento y corrección de la unidad. Además, aspectos como herencia y polimorfismo agregan complejidad adicional a las pruebas.

Tradicionalmente una prueba de unidad consiste en una **prueba estructural** (o prueba de caja blanca), lo cual requiere conocer cómo la unidad está diseñada internamente, y una **prueba de especificación** (prueba de caja negra), basada sólo en la especificación del comportamiento externamente visible de la unidad. Normalmente ambas

pruebas son necesarias y se complementan entre si. Los sistemas orientados a objetos pueden utilizar estas pruebas y además requerir **pruebas basadas en estado**, correspondiente al estado encapsulado del objeto y la interacción de las operaciones.

Como las pruebas estructurales dependen de la estructura del sistema, mientras que la prueba basada en estado y de especificación puede afectar la estructura del sistema, es preferible hacer la prueba estructural al final. Si se encuentra una falta en cualquiera de las pruebas anteriores, se necesitaría modificar de manera correspondiente el sistema, afectando la prueba estructural. Las pruebas son descritas con mayor detalle a continuación y pudiendo seguirse en el orden descrito:

?? **Prueba de especificación**, o de caja negra, tiene como propósito verificar las relaciones de entrada y salida de una unidad. El objetivo es verificar "qué" hace la unidad, pero sin saber "cómo" lo hace. Se envían estímulos con diferentes parámetros como entrada y se comparan con las salidas esperadas. Se revisa que estos sean correctos, como en el caso de operaciones matemáticas. Dado que las unidades se comunican mediante interfaces bien definidas, la prueba de especificación es bastante directa.

?? **Prueba basada en estado** verifica las interacciones entre operaciones de una clase según cambios en los atributos de un objeto. No se puede ver a las operaciones de un objeto como aisladas e independientes de los valores de los atributos. Se debe hacer pruebas del objeto de acuerdo a su "ciclo de vida". Esto es especialmente importante para objetos controlados por estado, descritos usando diagramas de transición de estados. En la realidad es imposible revisar todas las posibles combinaciones de valores de atributos en combinación con todos los posibles estímulos. Es suficiente revisar los estados identificados en los diagramas de transición de estados de los objetos. Adicionalmente, algunas combinaciones particulares de atributos pueden ser de mayor interés que otras. Otras combinaciones se pueden considerar *conjuntos de equivalencia* (*equivalence set*) de comportamiento evitando revisiones adicionales. Por lo tanto, es deseable revisar valores particulares significativos para luego asignar un conjunto de equivalencias para cada grupo de valores relacionados. Por otro lado, algunas operaciones no afectan el estado, como son las operaciones puramente de lectura, por lo cual no necesitan ser consideradas en la prueba basada en estados. Las demás operaciones deberán ejecutarse para todos los posibles estados iniciales.

?? **Prueba estructural** tiene como propósito verificar que la estructura interna de la unidad sea correcta. Esta prueba se conoce también como prueba basada en programa o de caja blanca, dado que debe conocerse cómo está implementada internamente la unidad. Es deseable cubrir todas las posibles combinaciones de parámetros, valores de variables y flujos en el código, de manera que todas las instrucciones se ejecuten. Para examinar la efectividad de los casos de prueba se debe medir la *cobertura de prueba* (*coverage test*) donde cada ruta en el código sea cubierta o probada al menos una vez. La mayoría de los problemas provienen de combinaciones de rutas inusuales como aquellas que involucran ciclos. Los casos de prueba se ilustran mediante diagramas de flujo (*flowcharts*).

Prueba de Integración

Después de haber probado todas las unidades, éstas deben ser integradas en unidades más grandes hasta generar el sistema completo. El propósito de la prueba de integración es probar que las diferentes unidades estén trabajando juntas de manera apropiada. En la prueba de integración se incluye la prueba de paquetes de servicio, casos de uso, subsistemas y el sistema completo. Durante las pruebas de combinación de unidades, algo que se incrementa exponencialmente, se podrá detectar fallas imposibles de detectar durante las pruebas de una sola unidad. No se debe comenzar la prueba de integración hasta que las pruebas de unidad estén listas.

La prueba de integración se basa principalmente en la *prueba de casos de uso*, partiendo de los diagramas de interacción, donde se pueden identificar los estímulos enviados entre el usuario y el sistema, y entre los objetos en el sistema. La prueba de casos de uso se divide en pruebas de curso básico, cursos alterno y documentación de usuario. Se prueba primero los flujos básicos, los esenciales del sistema, probando luego los flujos alternos, correspondientes a flujos que ocurren de manera inusual como en el caso de manejo de excepciones. Los casos de uso que extienden o son incluidos en otros casos de uso se prueban después de probar los casos de uso básicos donde estos se insertan.

Prueba de Sistema

Las pruebas de casos de uso se hacen inicialmente de manera independiente, basadas en el modelo de requisitos. Después de probar todos los casos de uso aislados, se prueba el sistema entero como uno solo. Se ejecutan varios casos de uso en paralelo y se somete el sistema a diferentes cargas. Las pruebas de sistema pueden involucrar *pruebas de operación*, *pruebas de escala completa*, *pruebas negativas*, *pruebas basadas en requisitos* o *casos de uso*, y *pruebas de documentación de usuario*.

10.3 Proceso de Pruebas

El proceso de prueba involucra consideraciones similares al del propio proceso de desarrollo de software, incluyendo estrategias, actividades y métodos los cuales deber ser aplicados de manera concurrente al proceso de desarrollo de software. En particular, las actividades de prueba abarcan los siguientes aspectos: planeación, construcción y ejecución. Por lo general, se mantiene una *bitácora de prueba* (*test log*) durante todo el proceso de prueba.

Estrategia de Prueba

Existen diversas estrategias para el proceso de pruebas, incluyendo el orden en que se van a hacer, la partición de equivalencias de pruebas que se van a aplicar y la posibilidad de automatizarlas.

?? **Orden de Pruebas** tiene como propósito definir en qué momento y en qué orden se aplicarán las pruebas. Aunque las pruebas deben ser planeadas con anterioridad, las verificaciones típicamente se aplican durante el diseño, implementación y operación del sistema. Existen dos enfoques generales para el orden en que se llevarán a cabo las pruebas: de *arriba hacia abajo* y de *abajo hacia arriba*. Este orden depende de gran manera de la estrategia de diseño ya que debe lograr una buena correspondencia con el proceso de desarrollo utilizado. Si se hace pruebas correspondientes a un diseño de arriba hacia abajo, entonces se desarrolla inicialmente las interfaces entre subsistemas, donde se busca probar los protocolos a alto nivel antes de ir a los niveles más bajos. Si se hace un diseño de abajo hacia arriba se puede certificar primero las unidades de bajo nivel y luego las interfaces entre estas. Esta técnica se aprovecha que una vez probadas las unidades se elimina la necesidad de crear servidores especializados para pruebas. En el de pruebas de arriba hacia a bajo, se necesitan servidores de pruebas especiales que simulen el ambiente alrededor de la unidad que se prueba. En el caso extremo se debe construir una estructura completa para simular todos los objetos del sistema que aún no existen. Normalmente, es suficiente tener una base de pruebas que sea una magnitud de orden menor que lo que se está probando, como una clase de prueba por cada paquete de servicio (contratos), un contrato para cada subsistema, etc.

?? **Alcance de Pruebas** tiene como propósito identificar el tipo, número y casos de pruebas que se aplicarán para revisar el sistema en sus diferentes aspectos. Si se considera que los tipos de pruebas son bastante amplios y bastante extensos, el objetivo es seleccionar un número pequeño pero razonable de casos de prueba dentro de un gran número de posibles pruebas donde la probabilidad de encontrar faltas sea alto. Se define la *partición de equivalencias* según *conjuntos equivalentes* (*equivalent set*) de pruebas definiendo un grupo de condiciones donde el sistema o algún componente se comportará de manera similar para diferentes pruebas. La idea es escribir casos de prueba que cubran todos los conjuntos de equivalencia, teniendo un caso de prueba para cada conjunto de equivalencia.

?? **Automatización de Pruebas** tiene como propósito reducir el esfuerzo y costo de las pruebas mediante la *automatización* del proceso o aspectos de él. Esto puede hacerse mediante programas de pruebas especiales asociados a un conjunto de datos de pruebas. El programa de prueba debe tomar conjuntos de datos de entrada y observar la respuesta del sistema, la cual puede guardarse directamente en un archivo de salida o comparado con alguna respuesta esperada. El objetivo es tener un programa de prueba los mas general e independiente posible de los datos de prueba, los cuales a su vez pudieran generarse automáticamente. Para probar el sistema completo son necesarios diferentes programas de prueba. Cuando se desarrollan los programas y datos de prueba se deben utilizar las mismas técnicas usadas para el desarrollo del sistema y deben desarrollarse en paralelo con el propio sistema. Los programas de prueba se consideran parte del sistema, pudiéndose usar para el mantenimiento del sistema, simplificando la tarea de buscar fallas y faltas cuando se haya instalado el sistema.

Planeación de la Prueba

La planeación de la prueba inicia con el establecimiento de las estrategias de pruebas, incluyendo consideraciones si la prueba se hará automáticamente o manualmente y si existen programas y datos de prueba que puedan ser usados o posiblemente modificados o desarrollados nuevamente. Se determina el alcance de las pruebas mediante conjuntos de equivalencia y se identifican los tipos de pruebas necesarios. Por lo general la planeación se hace durante el modelo de análisis luego de finalizar el modelo de requisitos. La prueba en si se diseña hasta durante la propia etapa de diseño del sistema.

Esta etapa de planeación permite estimar los recursos requeridos, sirviendo como guía para la especificación y ejecución de la prueba. Cuando los recursos de prueba están restringidos, cada caso de prueba debe maximizar la probabilidad estadística para detectar fallas buscando primero las fallas mayores.

Construcción de la Prueba

Una vez planificadas las pruebas, éstas se deben construir diseñándolas a un nivel funcional donde se describa cada prueba y su propósito de manera general y detallada. Se describe exactamente cómo se deberá ejecutar el caso de prueba de manera que personas no familiarizadas con la aplicación, o incluso el sistema, puedan ejecutar el caso de prueba. En el caso ideal, las especificaciones del diseño de las pruebas deben servir también como las propias especificaciones de la prueba.
Cada caso de prueba, con excepción de las pruebas de unidad de más bajo nivel, debe ser documentado. Las condiciones para la prueba deben ser especificadas, por ejemplo, si la prueba debe ser hecha en un ambiente de desarrollo o real, con qué software, hardware y equipo de prueba, y bajo qué versión del sistema. Se debe también describir cómo debe ejecutarse la prueba, en qué orden, cual es el resultado esperado y cuáles son los criterios para aprobar la prueba. También se debe describen cómo preparar los reportes de prueba requeridos para documentar los resultados de la prueba.
Si la documentación del diseño del sistema está descrita como especificaciones, éstas pueden ser usadas también como especificaciones de prueba. Una especificación de prueba es también a menudo una especificación de diseño. La etapa de diseño de pruebas también ayuda a encontrar problemas en el diseño del sistema e incluso faltas. En tal caso, éstas deben ser comunicadas al diseñador el cual debe corregir de manera adecuada para luego volver a aplicar las pruebas anteriores nuevamente. Por ejemplo, se puede tener como objetivo general detectar el máximo posible de faltas presentes en el sistema, por lo cual se puede tener como objetivo particular del diseño de pruebas minimizar el número de faltas por cada 1,000 líneas de código, o algo similar.

Ejecución de la Prueba

Se utiliza la especificación del diseño de prueba y los reportes de prueba durante la ejecución de las pruebas. La estrategia es probar en paralelo el mayor caso de pruebas posible. Se ejecutan las pruebas automáticas y manuales de manera correspondiente, e indicando los resultados esperados. Si alguna prueba particular fallara, se interrumpe la prueba y se anota el resultado para luego analizar el defecto y corregirlo si es posible. Una vez corregido, se ejecuta la prueba nuevamente. Todas las pruebas son ponderadas según su importancia, y se puede determinar si la prueba completa ha sido aprobada o no según su resultado. Se analizan los resultados de la prueba completa y si se anota en los reportes de prueba, el resumen de la prueba, los recursos utilizados, los resultados individuales y si los resultados fueron aprobados o no. El reporte de la prueba debe también mostrar el resultado de cada prueba individual, recursos utilizadas y acciones tomadas.
Si al ejecutar las pruebas se detectan fallas, el resultado de la prueba debe ser analizado y la razón para la falta identificada. La falta no tiene que ser por culpa del sistema, puede ser por otras causas, incluyendo si se hizo la prueba correctamente, si existe una falta en los datos de prueba o en el programa de prueba, si existe una falla causada por la base de prueba, o si el software del sistema se comporta de manera incorrecta. Si la falla no fue por causa del objeto en prueba, entonces se debe corregir y hacer la prueba nuevamente. Si la falla fue por causa del sistema, se busca identificar qué clases o paquetes de servicios deficientes deben ser devueltos al diseñador. Se puede facilitar el proceso de detección de faltas si las unidades probadas ofrecen facilidades apropiadas, por ejemplo, contadores o bitácoras de faltas.

10.4 Pruebas para el Sistema de Reservaciones de Vuelos

En lo que respecta al Sistema de Reservaciones de Vuelos desarrollado a lo largo del libro, nos limitaremos a verificar el sistema de acuerdo a la prueba de requisitos o casos de uso. Como objetivo de la prueba revisaremos que la funcionalidad implementada corresponda a los casos de uso correspondientes especificados durante el modelo de requisitos. A continuación revisaremos los casos de uso principales, básicos y de extensión, los cuales fueron descritos durante el diseño: *Registrar Usuario* y *Registrar Tarjeta*. Nótese que los casos de uso *Validar Usuario* y *Ofrecer Servicios* no los describimos de manera independiente ya que son inclusiones de los demás.

Registrar Usuario

Se prueban la secuencia más importante del casos de uso *Registrar Usuario*: *Crear Registro Usuario*, *Actualizar Registro Usuario* y *Eliminar Registro Usuario*.

Crear Registro Usuario

La secuencia *Crear Registro Usuario*, se muestra a nivel funcional en la Figura 10.1.

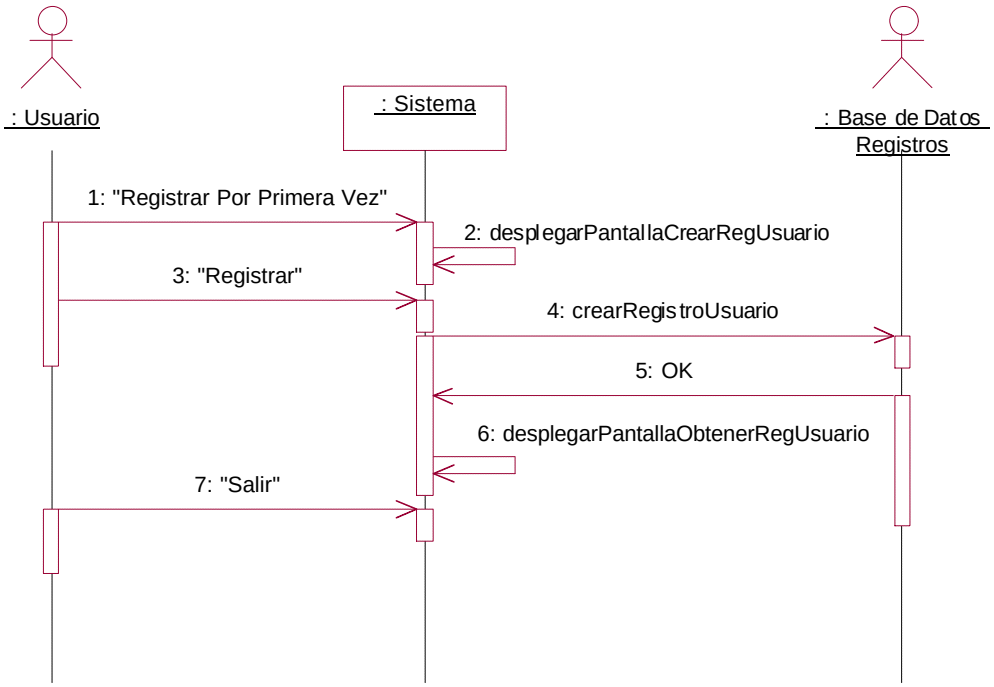

Figura 10.1 Diagrama de secuencia *Crear Registro Usuario* del caso de uso *Registrar Usuario*.

La secuencia comienza con el usuario presionando el botón "Registrarse por Primera Vez" en la PantallaPrincipal (P-1), como se muestra en la Figura 10.2.

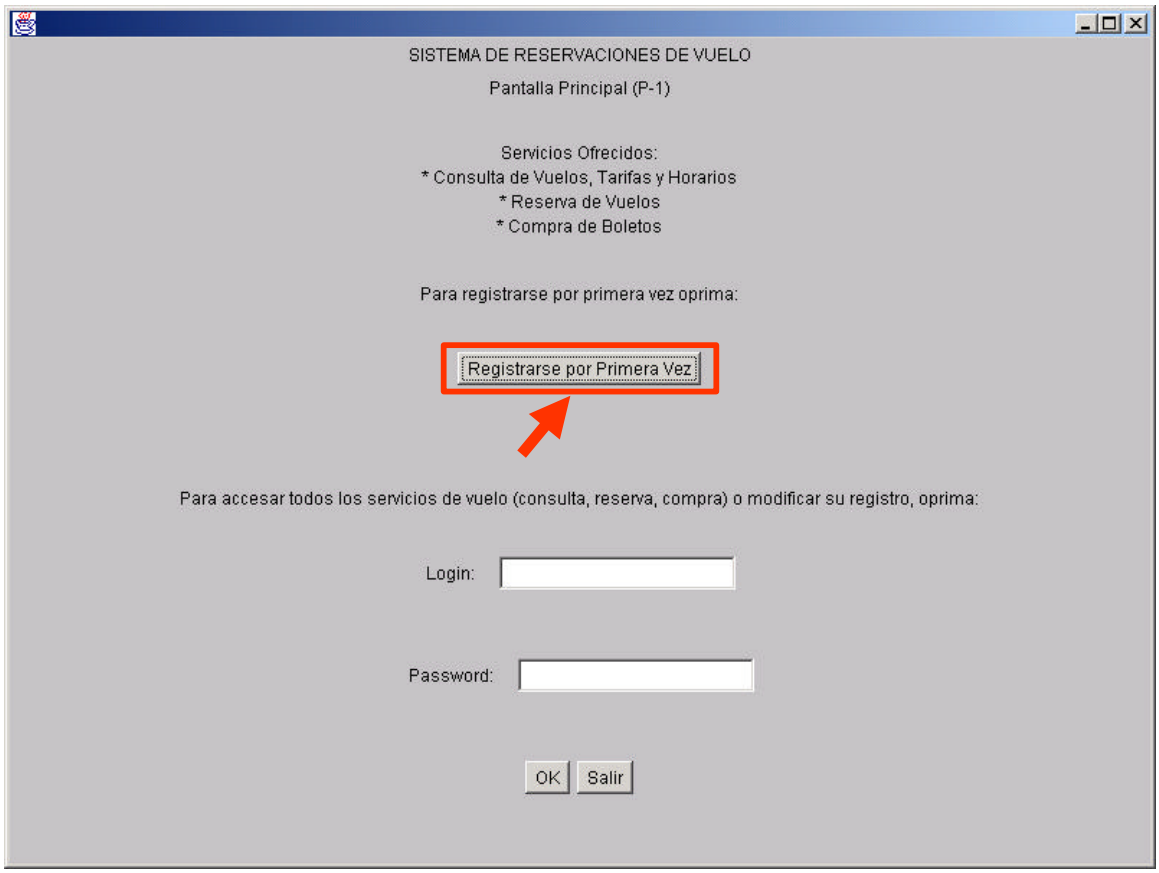

Figura 10.2 La secuencia *Crear Registro Usuario* comienza con el usuario oprimiendo el botón "Registrarse por Primera Vez" en la PantallaPrincipal (P-1).

A continuación el sistema presenta la PantallaCrearRegistroUsuario (P-3) la cual debe ser llenada por el usuario con datos de registro. Al finalizar la inserción de los datos el usuario debe presionar el botón "Registrar", como se muestra en la Figura 10.3.

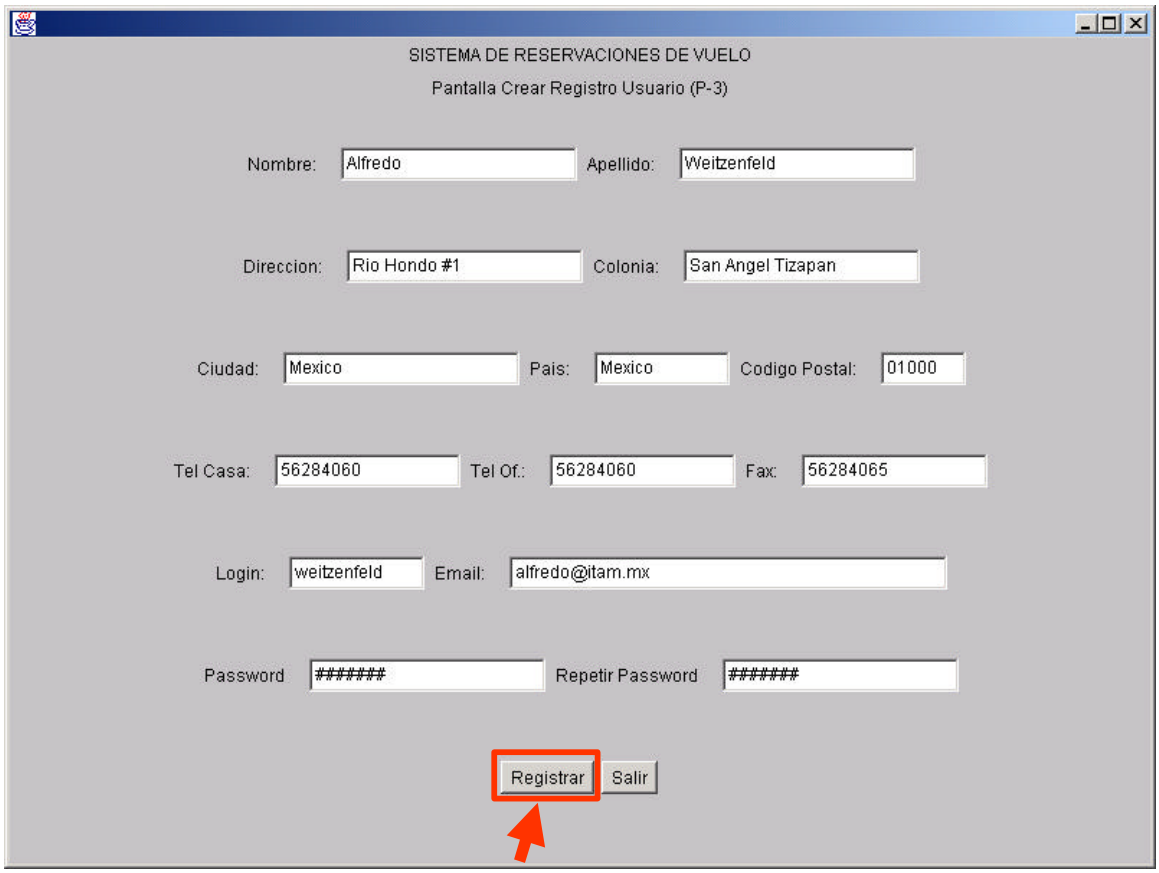

Figura 10.3. La secuencia *Crear Registro Usuario* continúa con el usuario oprimiendo el botón "Registrar" en la PantallaCrearRegUsuario (P-3) una vez llenado los campos correspondientes.

Se puede revisar en la *Base de Datos Registro* que los valores han sido creados e insertados correctamente a la base de datos, como se puede observar en la Figura 10.4.

RegistroUsuario : Table

login	password	nombre	apellido	direccion	colonia	ciudad	pais	CP	telCasa	telOficina	fax	email
weitzenfeld	alfredo	Alfredo	Weitzenfeld	Rio Hondo #1	San Angel Tizapan	Mexico	Mexico	01000	56284060	56284060	56284065	alfredo@itam.mx

Record: 2 of 2

Figura 10.4. Imagen de la Base de Datos de Registro mostrando el registro de usuario recién insertado.

A continuación el sistema despliega la PantallaObtenerRegUsuario (P-4), como se muestra en la Figura 10.5.

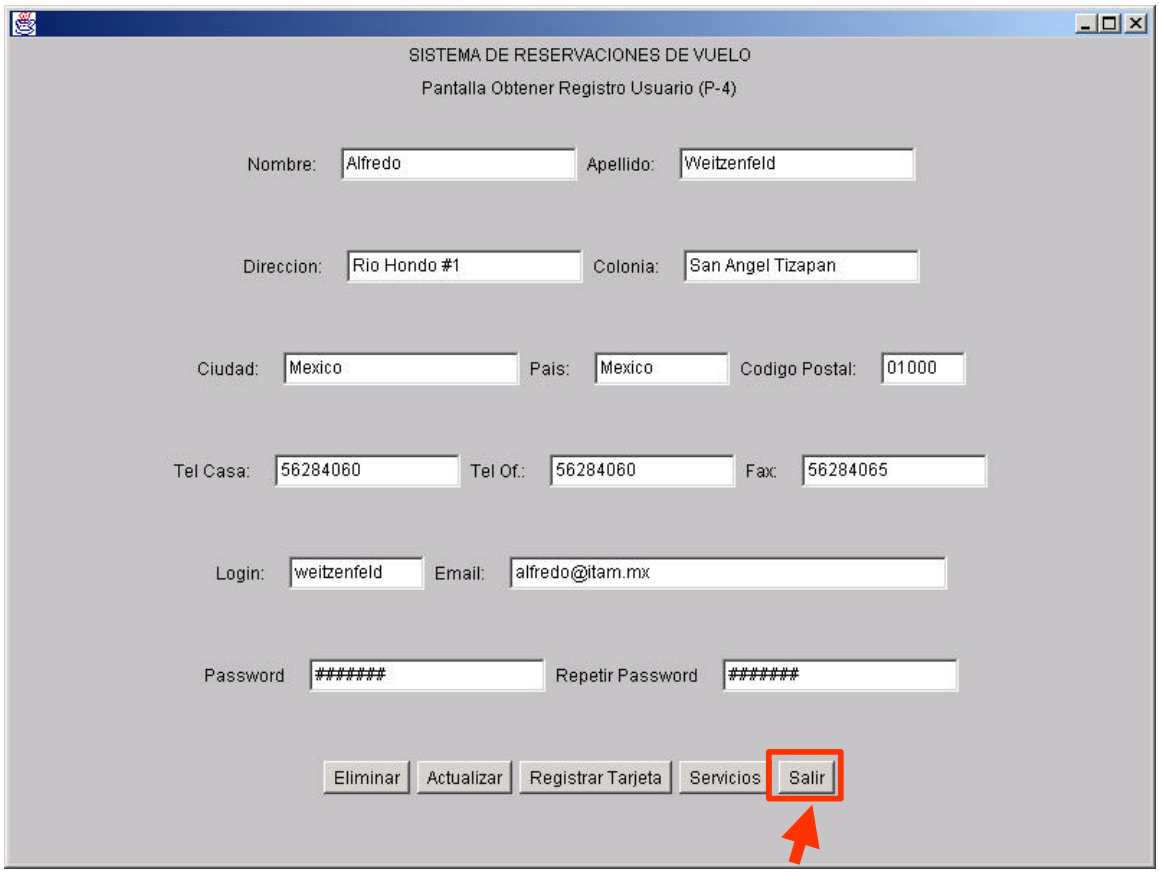

Figura 10.5. La secuencia *Crear Registro Usuario* continúa con el despliegue de la PantallaObtenerRegUsuario (P-4). Una vez desplegada esta pantalla, el usuario puede finalizar el caso de uso presionando del botón "Salir".

El usuario puede finalizar el caso de uso presionando el botón "Salir" en la PantallaObtenerRegUsuario (P-4).

Actualizar Registro Usuario

La secuencia *Actualizar Registro Usuario*, se muestra a nivel funcional en la Figura 10.6.

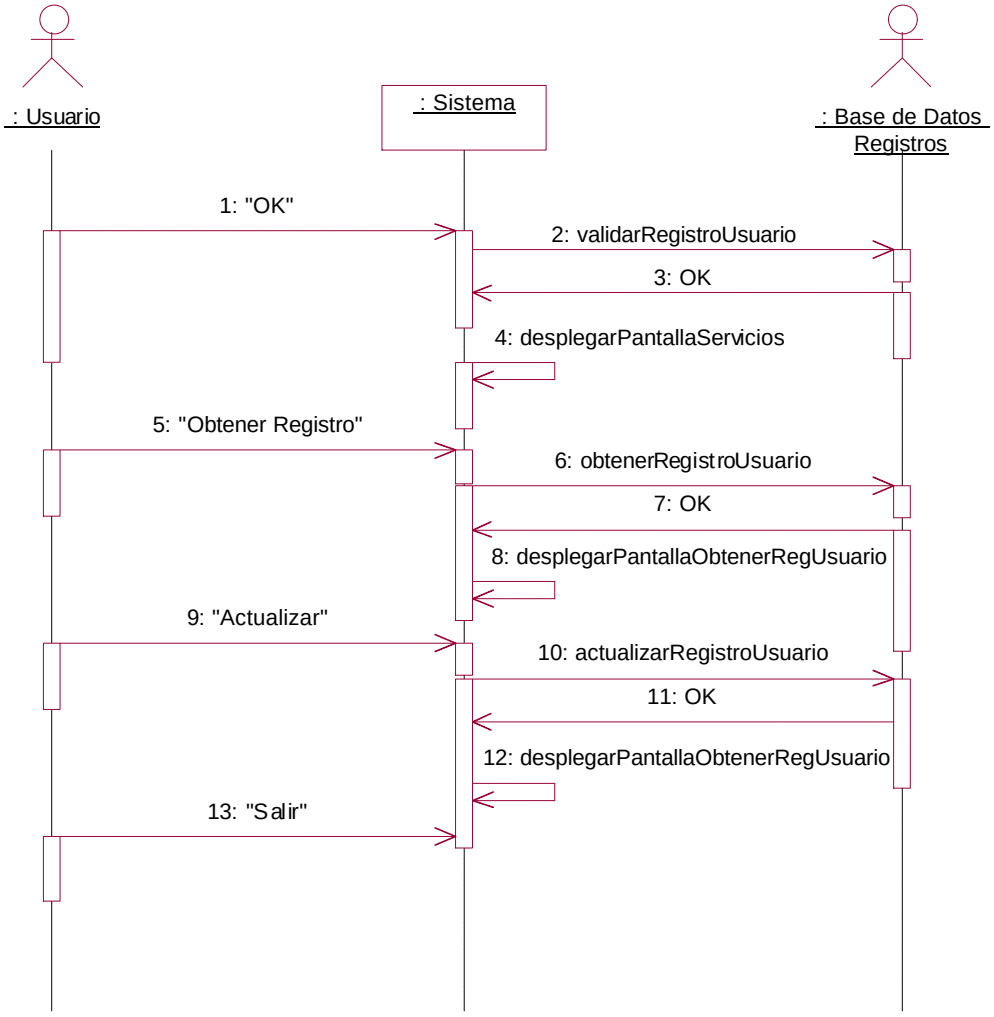

Figura 10.6 Diagrama de secuencias para *Actualizar Registro Usuario* del caso de uso *Registrar Usuario*.
La secuencia comienza con el usuario presionando el botón "OK" en la PantallaPrincipal (P-1), como se muestra en la Figura 10.7.

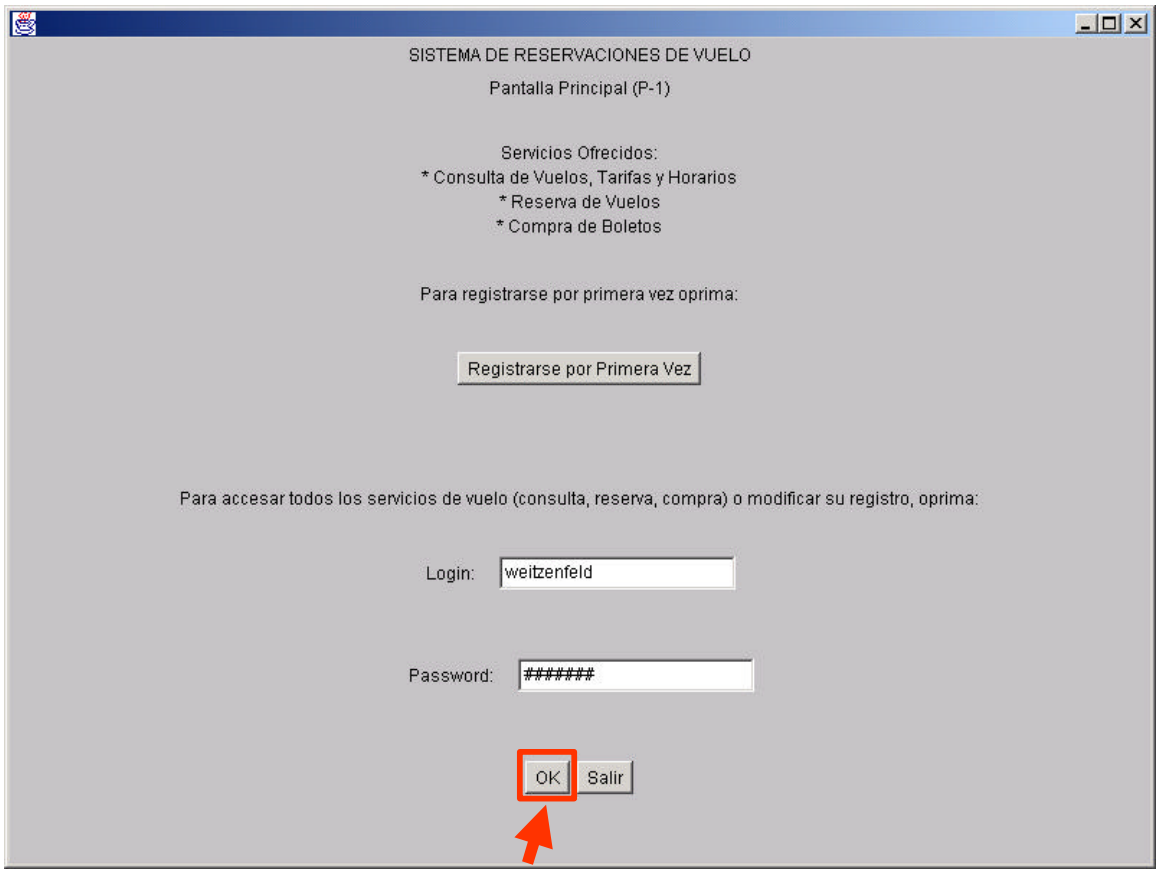

Figura 10.7. La secuencia *Actualziar Registro Usuario* comienza con el usuario oprimiendo el botón "OK en la PantallaPrincipal (P-1) luego de haber insertado los datos de *login* y *password*, "weitzenfeld" y "alfredo" respectivamente. Nótese que se despliega el *password* en la pantalla mediante caracteres de tipo '#'.

A continuación el sistema presenta la PantallaServicio (P-2). Para este caso de uso, la opción de "Obtener Registro" es la apropiada, como se muestra en la Figura 10.8.

Figura 10.8. La secuencia *Actualizar Registro Usuario* continúa con el usuario oprimiendo el botón "Obtener Registro" en la PantallaServicio (P-2).

A continuación el sistema despliega la PantallaObtenerRegUsuario (P-4). El usuario puede hacer las modificaciones deseadas, como cambios en el número de teléfono de la oficina, para luego presionar el botón "Actualizar", como se muestra en la Figura 10.9.

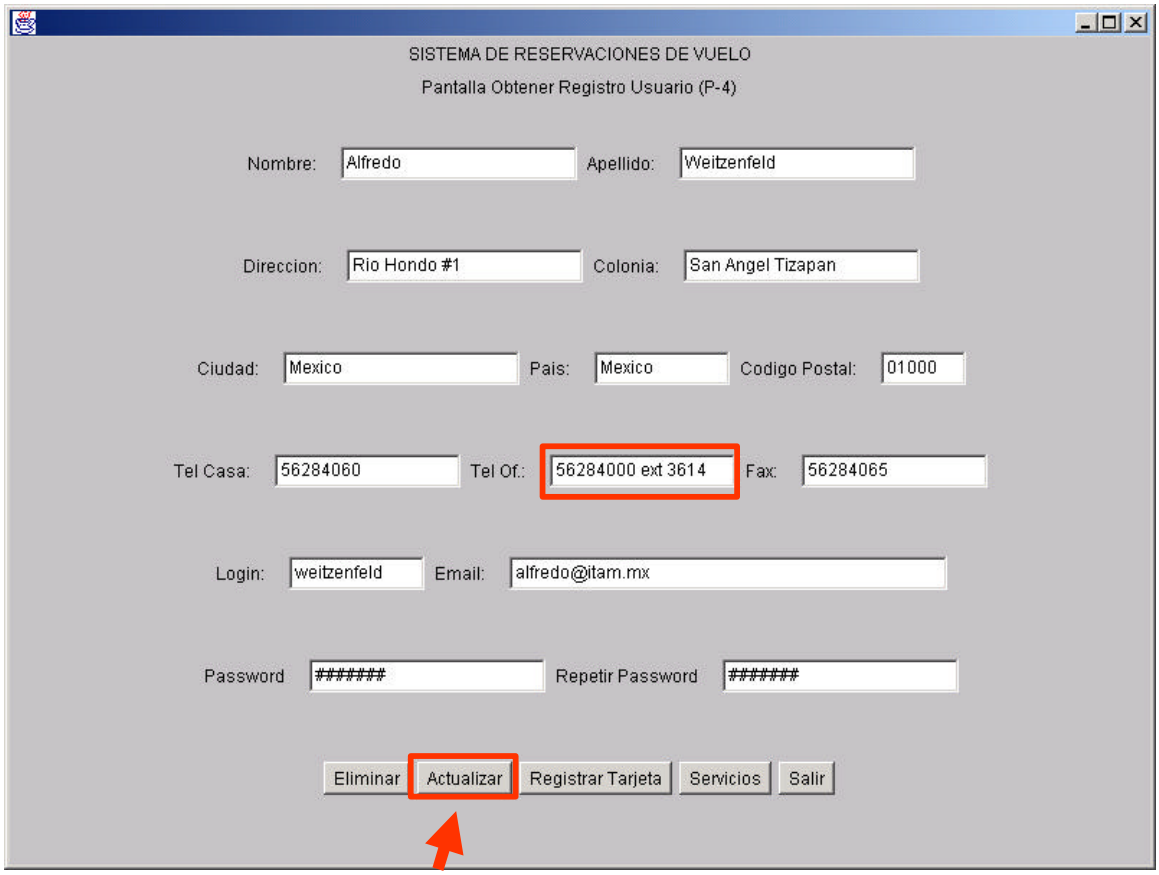

Figura 10.9. La secuencia *Actualizar Registro Usuario* continúa con el usuario oprimiendo el botón "Actualizar" en la PantallaObtenerRegUsuario (P-4) una vez modificados los campos deseados, como el teléfono de la oficina.

El sistema vuelve a desplegar la PantallaObtenerRegUsuario (P-4) luego de proceder con la actualización anterior. Se puede revisar en la *Base de Datos Registro* que los valores han sido actualziados correctamente a la base de datos, como se puede observar en la Figura 10.10.

RegistroUsuario : Table

login	passwor	nombre	apellido	direccion	colonia	ciudad	pais	CP	telCasa	telOficina	fax	email
+ weitzenfeld	alfredo	Alfredo	Weitzenfeld	Rio Hondo #1	San Angel Tizapan	Mexico	Mexico	01000	56284060	56284000 ext 3614	56284065	alfredo@itam.mx

Record: 2 of 2

Figura 10.10. Imagen de la Base de Datos de Registro mostrando el registro de usuario modificados.

El usuario puede finalizar el caso de uso presionando el botón "Salir" en esa misma pantalla, como se mostró anteriormente en la Figura 10.5.

Eliminar Registro Usuario

La secuencia *Eliminar Registro Usuario*, se muestra a nivel funcional en la Figura 10.11.

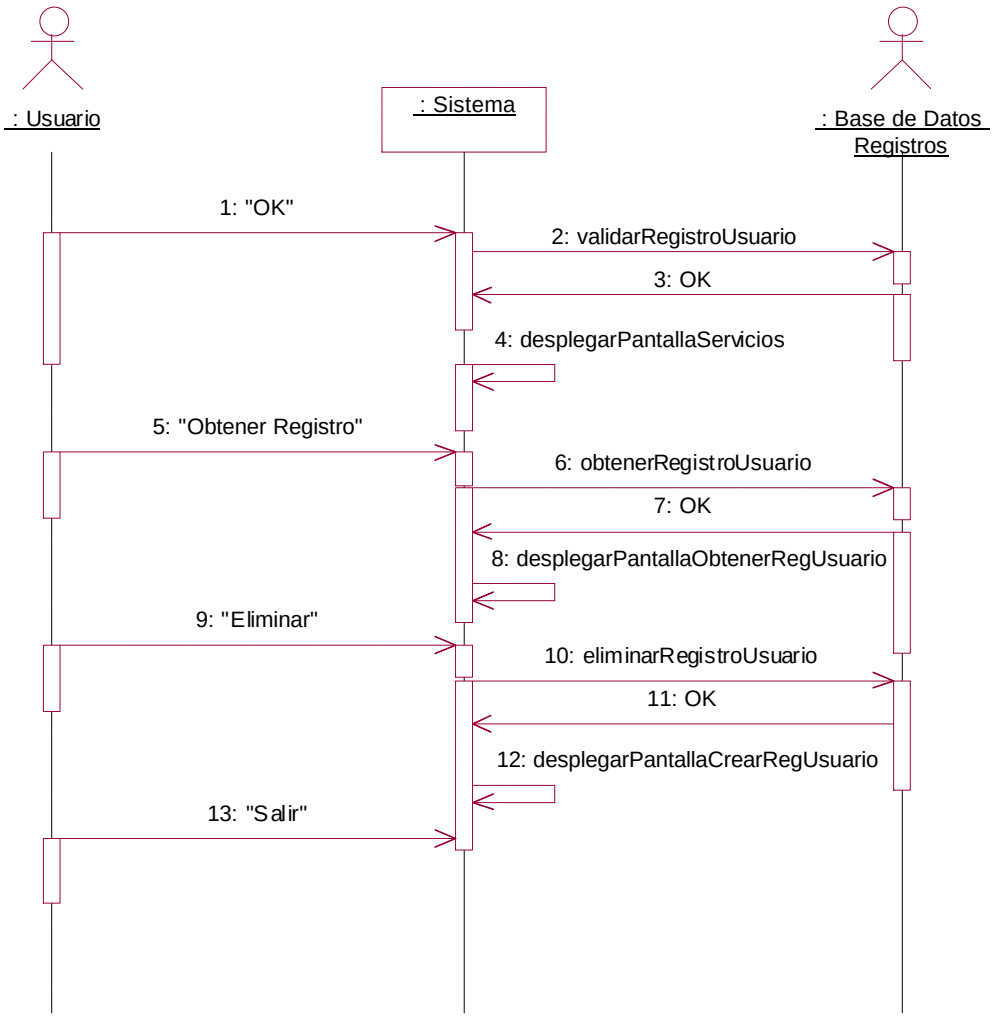

Figura 10.11 Diagrama de secuencias *Eliminar Registro Usuario* del caso de uso *Registrar Usuario*.

La secuencia comienza de manera similar a *Actualizar Registro Usuario*, como se mostró anteriormente en la Figura 10.7, con el usuario insertando sus datos de *login* y *password* para luego presionar el botón "OK" en la PantallaPrincipal (P-1). Se continúa con el usuario presionando el botón "Obtener Registro" en la PantallaServicio (P-2), como se mostró anteriormente en la Figura 10.8. A continuación el sistema presenta la pantalla PantallaObtenerRegUsuario (P-4). Para eliminar el registro, el usuario deberá presionar el botón "Eliminar", como se muestra en la Figura 10.12.

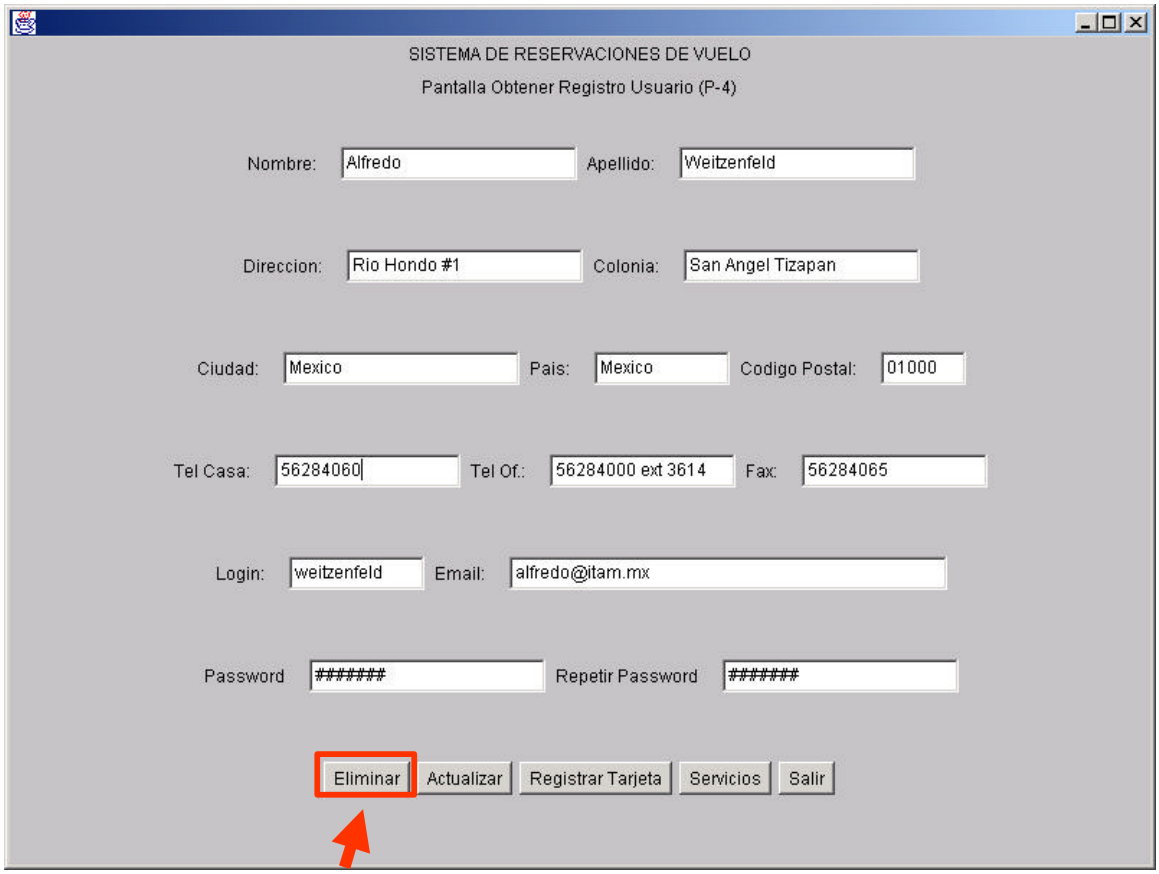

Figura 10.12. La secuencia *Eliminar Registro Usuario* continúa con el usuario oprimiendo el botón "Eliminar" en la PantallaObtenerRegUsuario (P-4).

El sistema despliega la PantallaCrearRegistroUsuario (P-3), dando la posibilidad de crear un nuevo registro de usuario.

Se puede revisar en la *Base de Datos Registro* que los valores han sido eliminados correctamente a la base de datos, como se puede observar en la Figura 10.13.

Figura 10.13. Imagen de la Base de Datos de Registro mostrando el registro de usuario eliminados.

El usuario puede finalizar el caso de uso presionando el botón "Salir" en la PantallaCrearRegistroUsuario (P-3), como se muestra en la Figura 10.14.

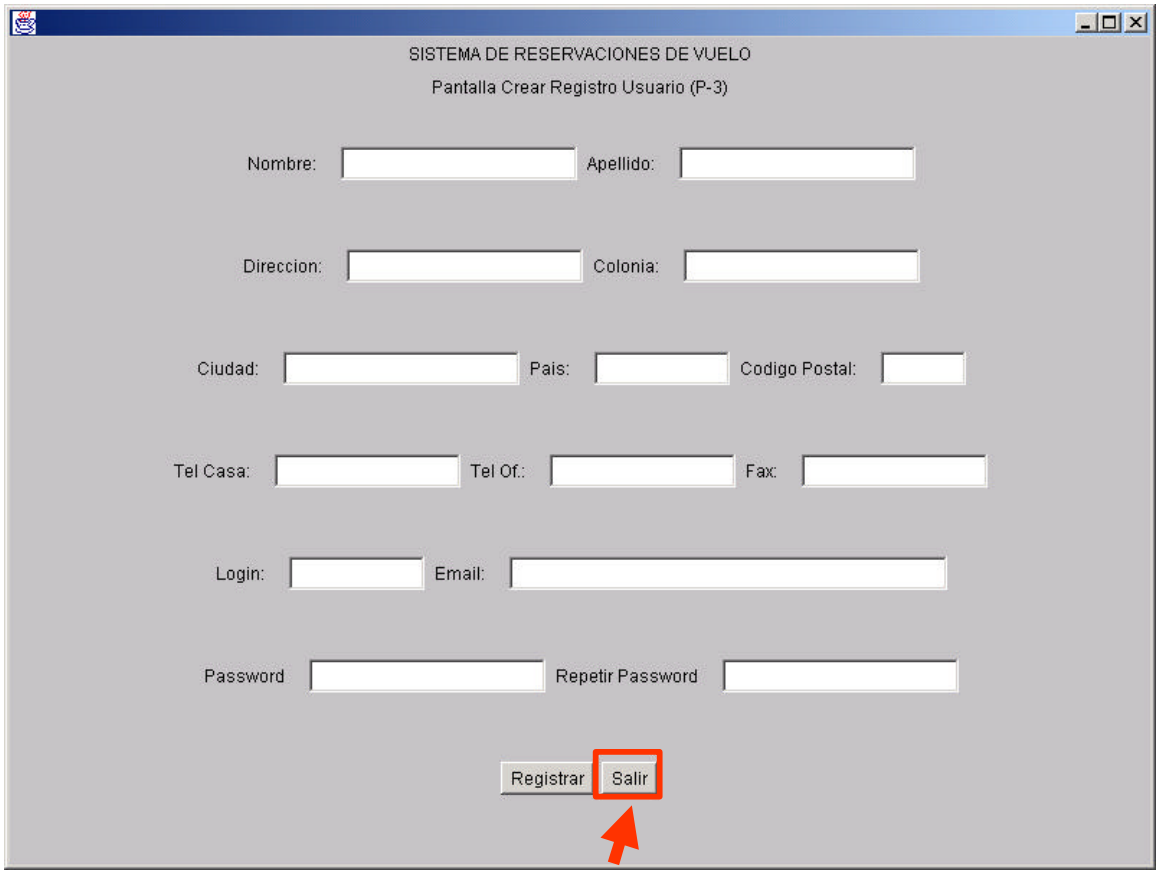

Figura 10.14. La secuencia *Eliminar Registro Usuario* termina con el usuario oprimiendo el botón "Salir" en la PantallaCrearRegUsuario (P-3).

Registrar Tarjeta

Se prueban las secuencia más importante del caso de uso *Registrar Tarjeta*: *Crear Registro Tarjeta*, *Actualizar Registro Tarjeta* y *Eliminar Registro Tarjeta*.

Crear Registro Tarjeta

El caso de uso *Registrar Tarjeta* es una extensión al caso de uso *Registrar Usuario*. La secuencia *Crear Registro Tarjeta* depende de que no exista un registro anterior de tarjeta para el usuario actual. El diagrama de secuencia de alto nivel se muestra en la Figura 10.15.

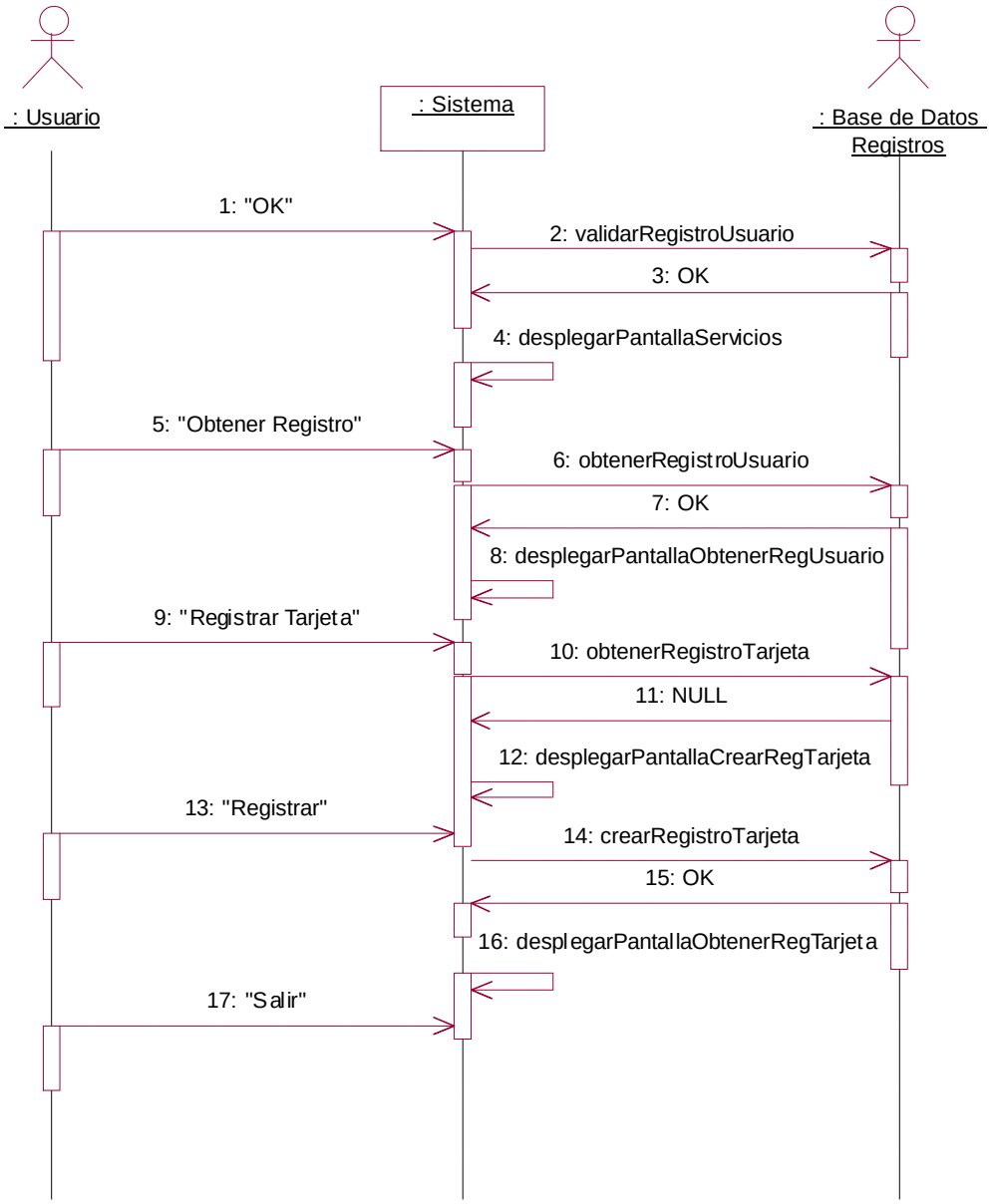

Figura 10.15 Diagrama de secuencias *Crear Registro Tarjeta* del caso de uso *Registrar Tarjeta*.

La secuencia comienza con de manera similar a *Actualizar Registro Usuario*. En lugar de hacer una actualización de registro de usuario, se presiona el botón "Registrar Tarjeta" en la PantallaObtenerRegUsuario (P-4), como se muestra en la Figura 10.16.

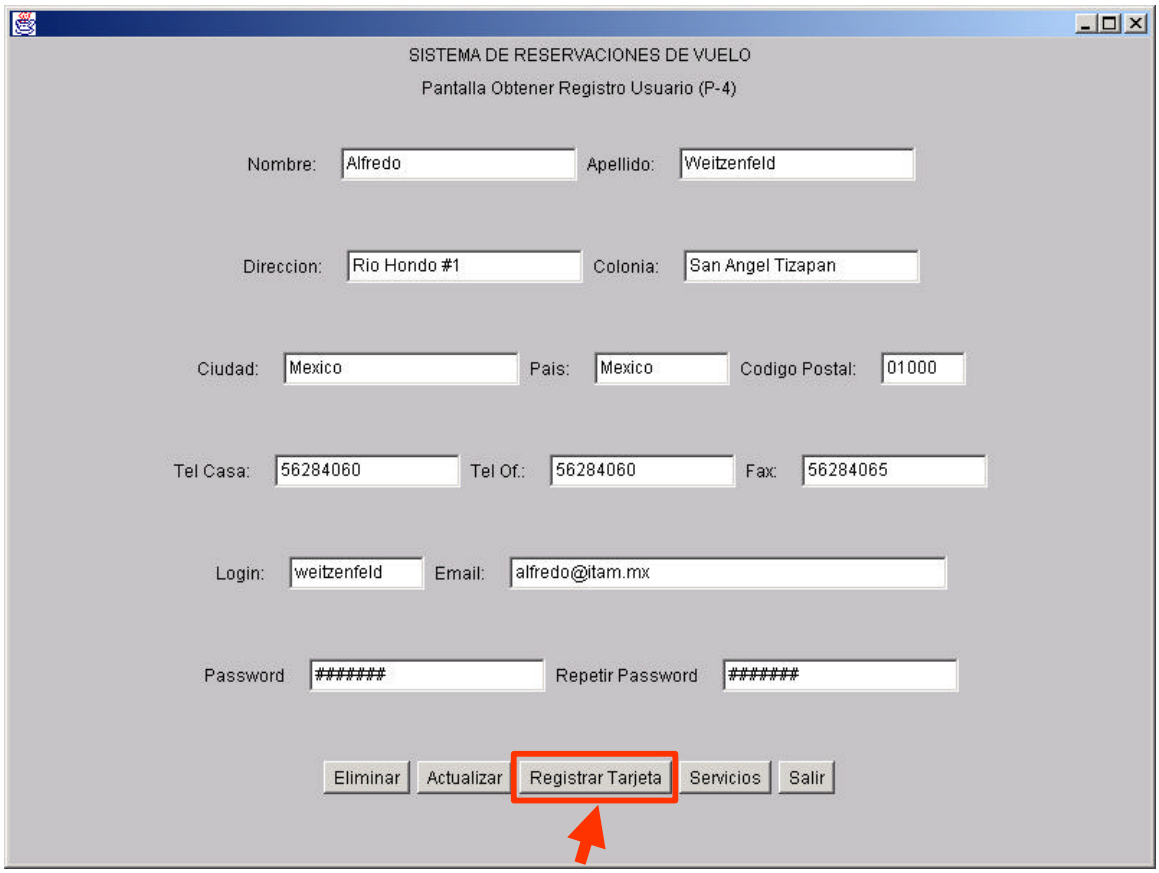

Figura 10.16. La secuencia *Crear Registro Tarjeta* comienza con el usuario oprimiendo el botón "Registrar Tarjeta" en la PantallaObtenerRegUsuario (P-4).

El caso de uso continúa con el sistema presentando la PantallaCrearRegTarjeta (P-5) la cual deberá ser llenada con los datos de tarjeta de crédito solicitados. Nótese, que en este subflujo aún no existe una tarjeta de registro para el usuario. El usuario deberá presionar el botón "Registrar", como se muestra en la Figura 10.17.

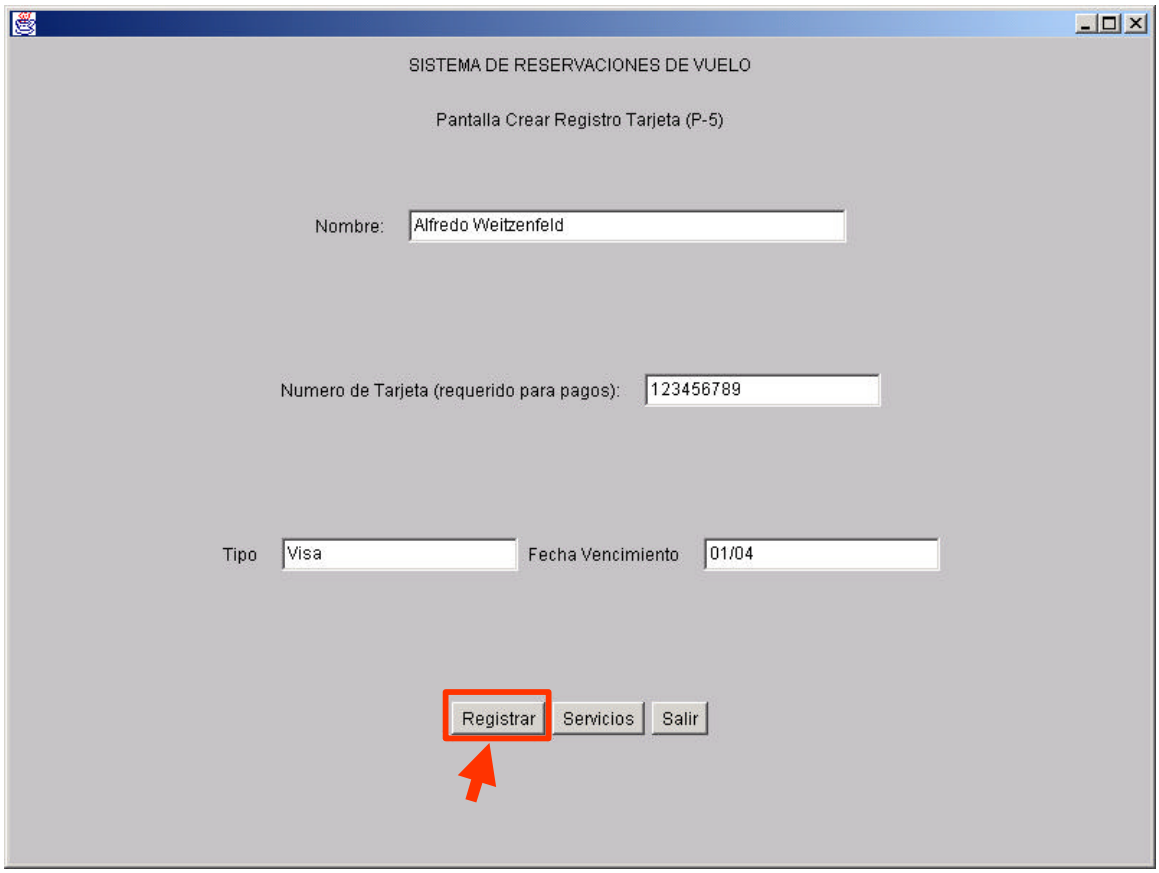

Figura 10.17. La secuencia *Crear Registro Tarjeta* continúa con el usuario insertando los datos de tarjeta de crédito solicitados para luego presionar el botón "Registrar" en la PantallaCrearRegTarjeta (P-5).

Se puede revisar en la *Base de Datos Registro* que los valores han sido creados e insertados correctamente a la base de datos, como se puede observar en la Figura 10.18.

RegistroTarjeta : Table

login	nombre	numero	tipo	fecha
weitzenfeld	Alfredo Weitzenfeld	123456789	Visa	01/04

Record: 2 of 2

Figura 10.18. Imagen de la Base de Datos de Registro mostrando el registro de tarjeta recién insertado.

A continuación el sistema despliega la PantallaObtenerRegTarjeta (P-6), como se muestra en la Figura 10.19.

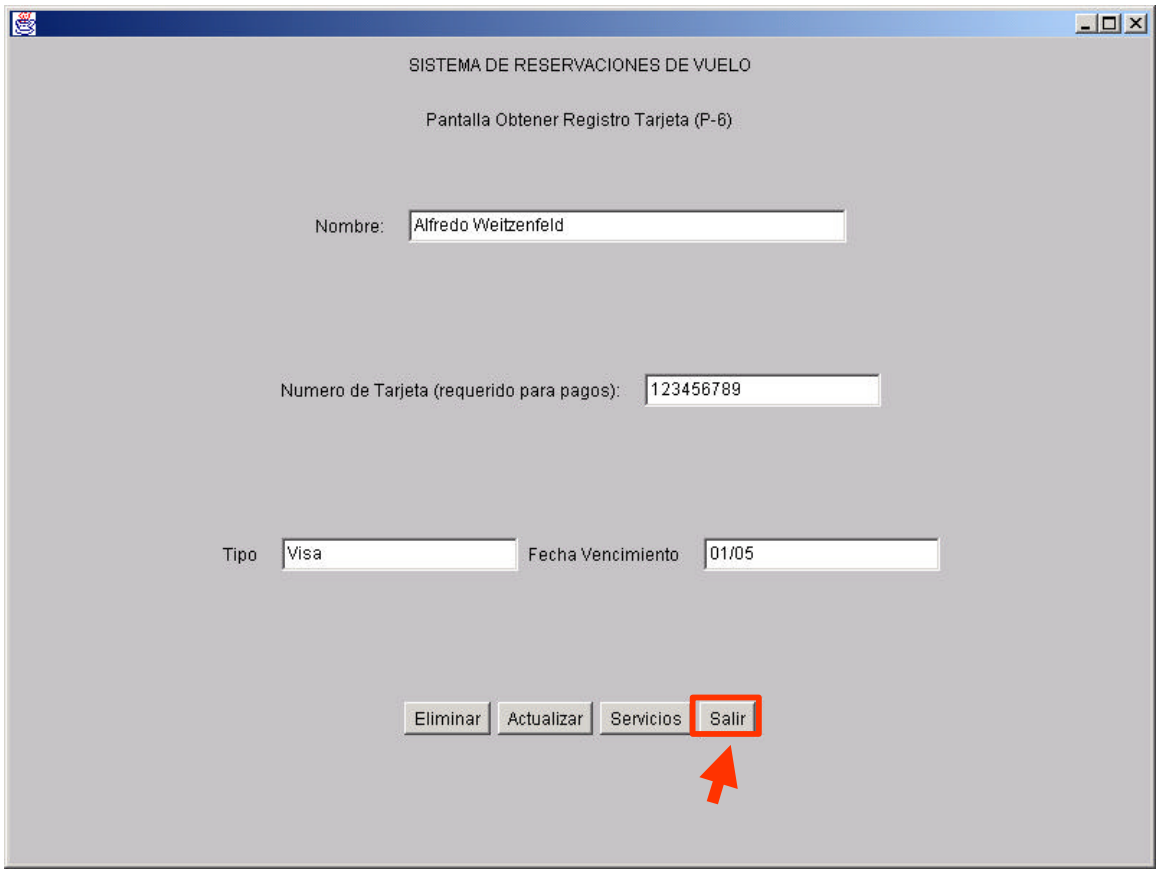

Figura 10.19. La secuencia *Crear Registro Tarjeta* termina con el usuario presionando el botón "Salir" en la PantallaObtenerRegTarjeta (P-6).

El usuario puede finalizar el caso de uso presionando el botón "Salir" en la PantallaObtenerRegTarjeta (P-6).

Actualizar Registro Tarjeta

La secuencia *Actualizar Registro Tarjeta* depende de que ya exista un registro anterior de tarjeta para el usuario actual. El diagrama de secuencia a nivel funcional se muestra en la Figura 10.20.

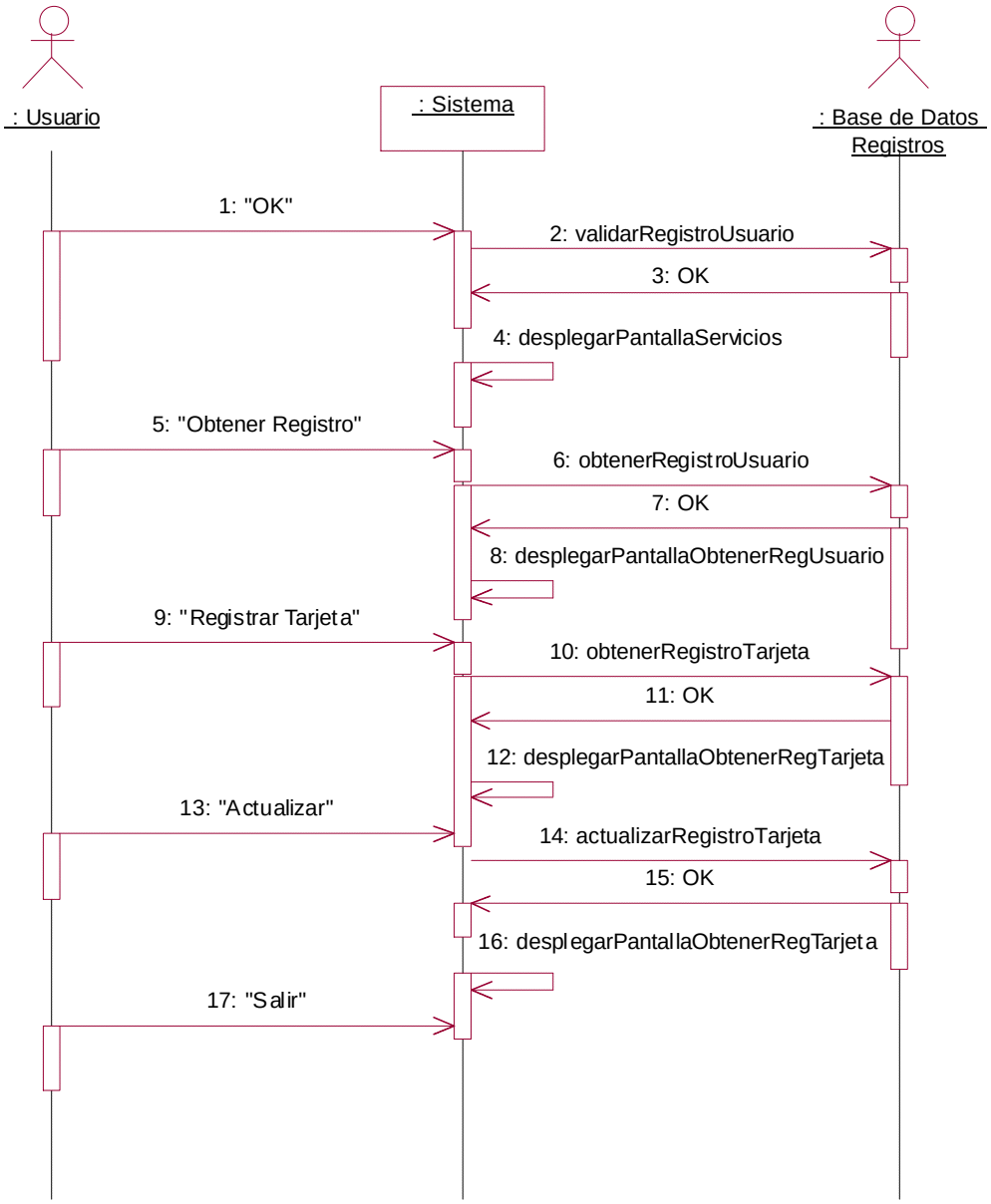

Figura 10.20 Diagrama de secuencias *Actualizar Registro Tarjeta* del caso de uso *Registrar Tarjeta*.

La secuencia inicia de manera similar a *Actualizar Registro Usuario*, donde en la PantallaObtenerRegUsuario (P-4) se presiona el botón "Registrar Tarjeta", como se mostró anteriormente en la Figura 10.16. El caso de uso continúa con el sistema presentando la PantallaObtenerRegTarjeta (P-6) donde el usuario podrá modificar los datos deseados de la tarjeta de crédito, tal como la fecha de vencimiento. El usuario deberá presionar el botón "Actualizar", como se muestra en la Figura 10.21.

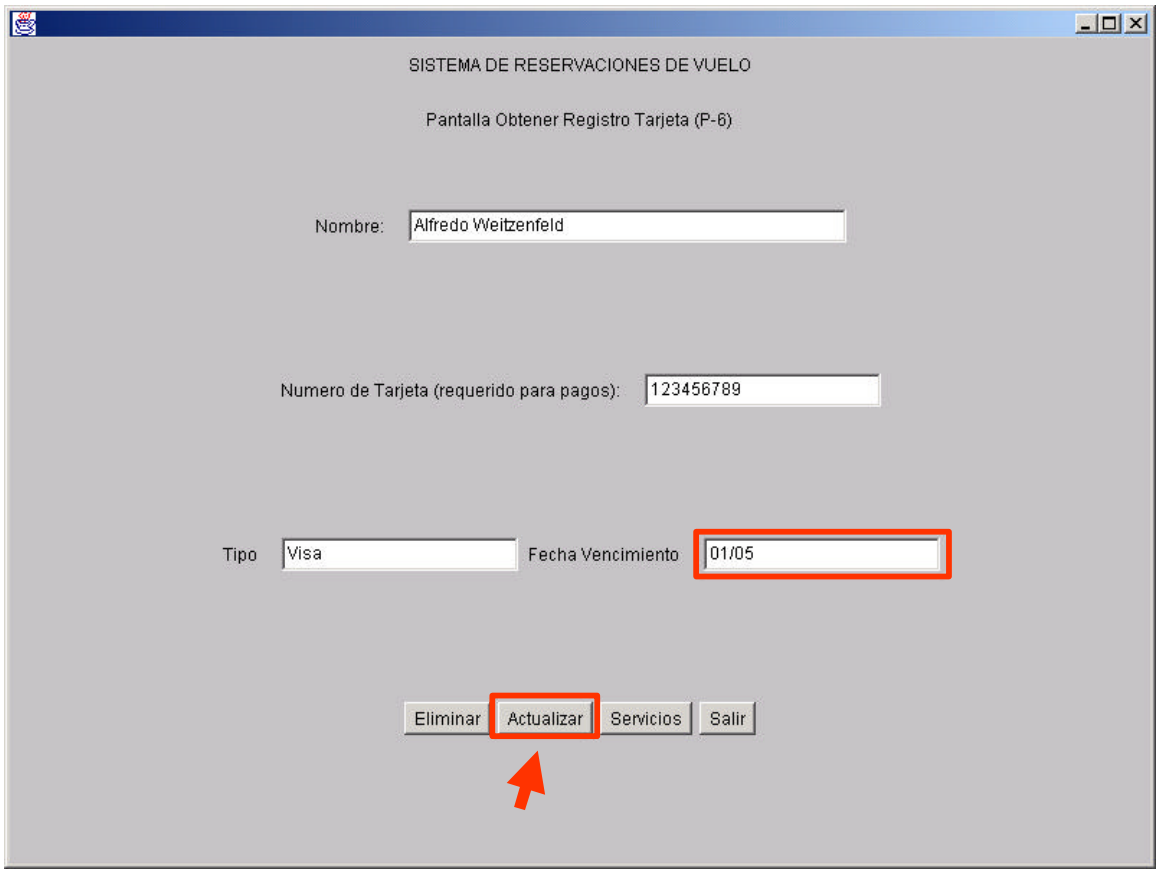

Figura 10.21. La secuencia *Actualizar Registro Tarjeta* permite al usuario modificar los datos de registro de la tarjeta, tales como la fecha de vencimiento. El usuario deberá presionar el botón "Actualizar" en la PantallaObtenerRegTarjeta (P-6).

Se puede revisar en la *Base de Datos Registro* que los valores han sido actualizados correctamente a la base de datos, como se puede observar en la Figura 10.22.

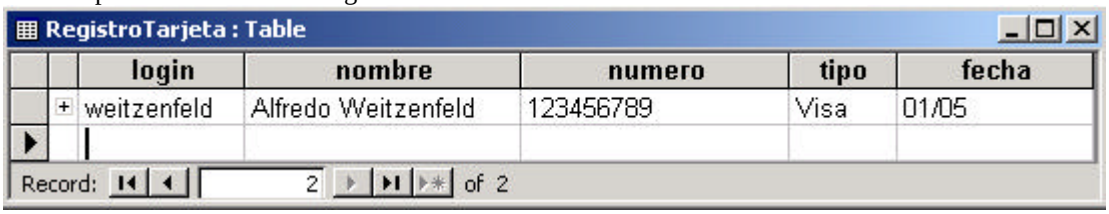

Figura 10.22. Imagen de la Base de Datos de Registro mostrando el registro de tarjeta recién actualizada.

El usuario puede finalizar el caso de uso presionando el botón "Salir" en la PantallaObtenerRegTarjeta (P-6), como se mostró anteriormente en la Figura 10.19.

Eliminar Registro Tarjeta

La secuenciaflujo *Eliminar Registro Tarjeta* se muestra a nivel funcional en la Figura 10.23.

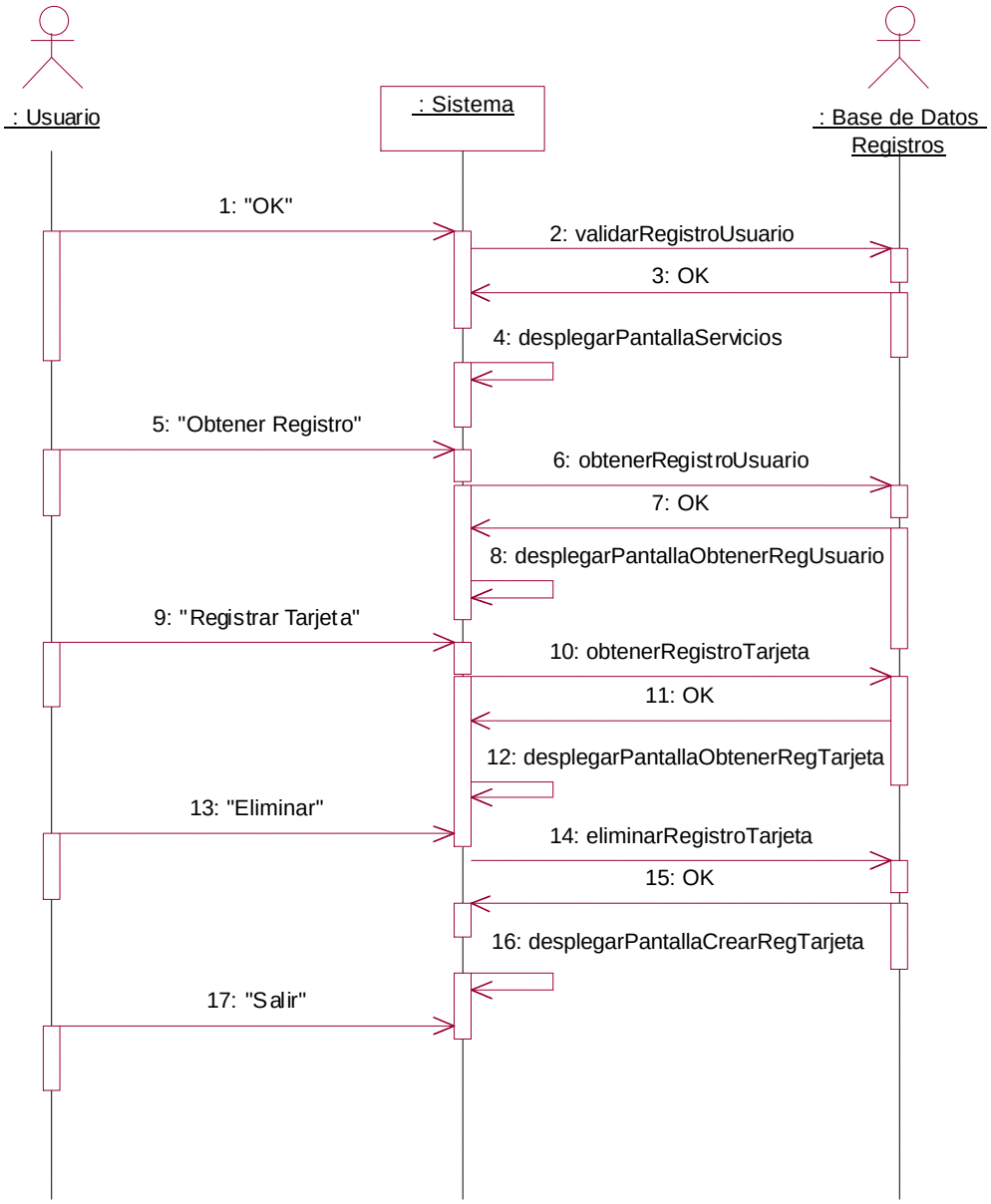

Figura 10.23 Diagrama de secuencias *Eliminar Registro Tarjeta* del caso de uso *Registrar Tarjeta*.

La secuencia comienza de manera similar a *Actualizar Registro Tarjeta*, presionando el botón "Registrar Tarjeta" en la PantallaObtenerRegUsuario (P-4), como se mostró anteriormente en la Figura 10.16. A continuación el sistema presenta la pantalla PantallaObtenerRegTarjeta (P-6). Para eliminar el registro de tarjeta, el usuario deberá presionar el botón "Eliminar", como se muestra en la Figura 10.24.

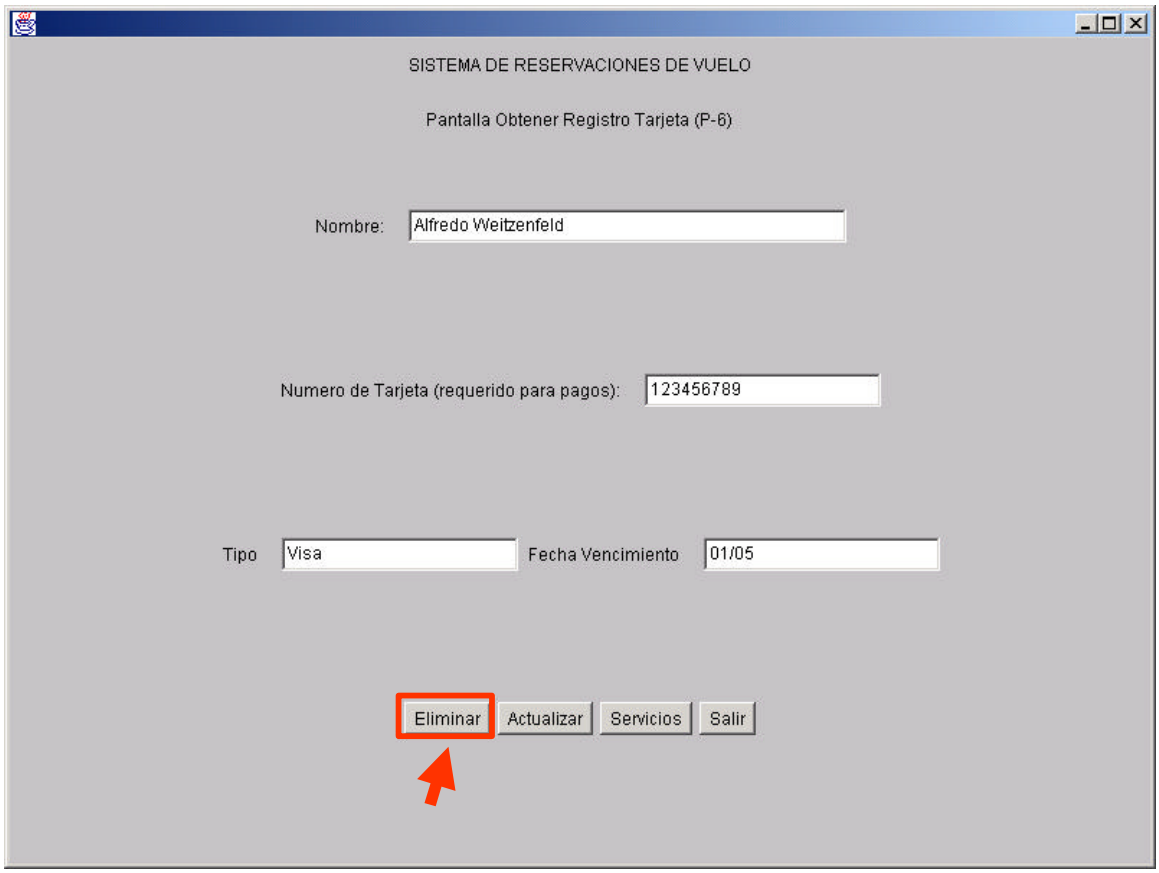

Figura 10.24. La secuencia *Eliminar Registro Tarjeta* continúa con el usuario presionando el botón "Eliminar" en la PantallaObtenerRegTarjeta (P-6).

El sistema presenta la PantallaCrearRegTarjeta (P-5), dando la posibilidad de crear un nuevo registro de tarjeta. Se puede revisar en la *Base de Datos Registro* que los valores han sido eliminados correctamente a la base de datos, como se puede observar en la Figura 10.25.

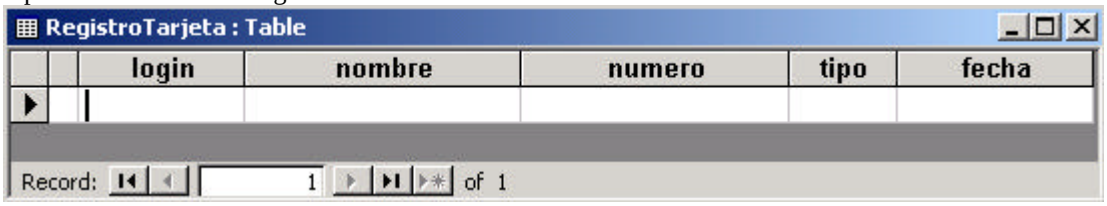

Figura 10.25. Imagen de la Base de Datos de Registro mostrando el registro de tarjeta eliminados.

El usuario puede finalizar el caso de uso presionando el botón "Salir", como se muestra en la Figura 10.26.

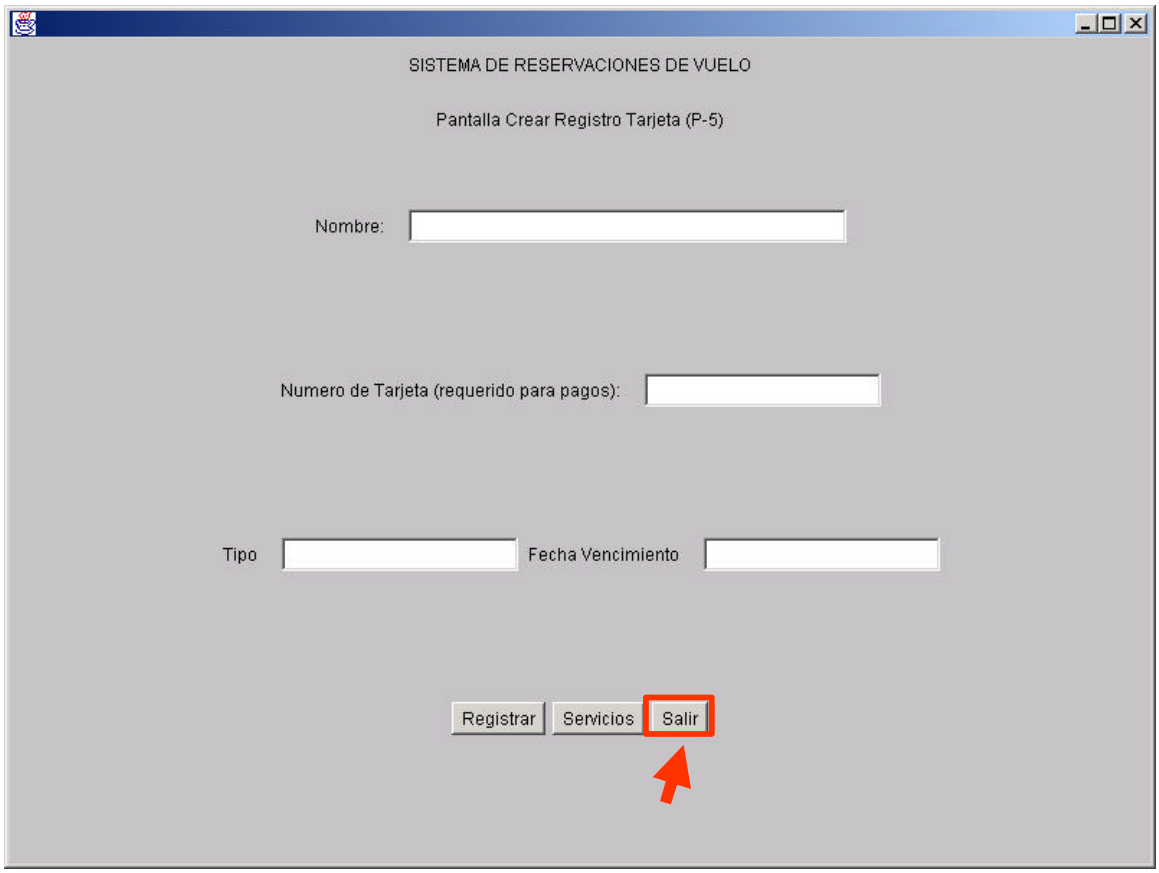

Figura 10.26. La secuencia *Eliminar Registro Tarjeta* termina con el usuario presionando el botón "Salir" en la PantallaCrearRegTarjeta (P-5).