Universidad Nacional Autónoma de México
Facultad de Ingeniería

SISTEMA PARA EL FINANCIAMIENTO DE ORGANIZACIONES NO LUCRATIVAS

TESIS QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

PRESENTA

RAÚL MÁRQUEZ ÁVILA

DIRECTOR:

ING. LUCILA PATRICIA ARELLANO MENDOZA

[2009]

Índice

Introducción

La organización nombrada Financiamiento de las Organizaciones No Lucrativas, surge como una necesidad para aquellas fundaciones o instituciones que requieran obtener un fondo para llevar a cabo un bien común de manera altruista, y que de otra forma nunca podrían ser atendidos directamente por el gobierno.

El FONOL se caracteriza por ayudar a cualquier fundación que realmente lo necesite, y lleva operando hace más de 5 años, en los cuales se ha ayudado a más de 5 mil instituciones que lo han requerido.

A pesar de la gran ayuda que ha brindado el FONOL en los años que ha estado activa, ha tenido muchos problemas, ya que la organización no cuenta con fondos suficientes. Uno de los problemas principales es que la organización cuenta con sólo con una cede en el Distrito Federal, por lo cual, si una organización se encuentra en una parte lejana en provincia, necesita trasladarse a la cede en el centro y emitir un oficio para ser financiada por el FONOL, y de igual manera, una vez aceptada la organización necesita emitir oficios cada vez que requiera fondos para un proyecto.

A pesar de que los oficios puedan ser emitidos por fax u otro medio, se necesita de papel para archivarlos y contar con un historial de la institución.

Ante esta situación, el presente trabajo de tesis proporciona de una solución a través del desarrollo de un sistema de información que cumpla con el funcionamiento base de FONOL, además de que brinde de una plataforma de información robusta y confiable.

A lo largo de cada capítulo se explica cómo fue desarrollado el sistema.

Capítulo 1

1. Planteamiento del Problema

En este capítulo, se dará la perspectiva global del problema, para resaltar las necesidades del cliente, y así posteriormente, apoyarse de las herramientas existentes para resolverlo.

1.1. Situación Actual

Una dependencia de gobierno llamada Financiamiento a Organizaciones NO Lucrativas (FONOL), está encargada de financiar a organizaciones no lucrativas. Para llegar a este fin, la organización que requiera ser financiada debe presentarse en las oficinas del FONOL, para que allí, después de llenar un extenso formulario e identificarse realmente como una Organización No Lucrativa, se permita ser candidata junto con otras organizaciones que también requieran ser financiadas. La persona encargada de evaluar tanto a las organizaciones como a los proyectos que ellas propongan, es el delegado, el cual, es el encargado de un estado en específico. Si una organización es aceptada, el FONOL le envía un oficio de aceptación, el cual le indica las instrucciones a seguir dentro de la institución. Por otra parte, ya que una organización es financiada, solicita al FONOL, mediante un oficio, que se le asignen fondos para un proyecto, y de ser aceptado, se le regresa un oficio de aceptación, que le indica como recibir el dinero. Es preciso indicar que todos los oficios pueden tardar días en procesarse.

Es por esto, que el FONOL requiere un sistema que automatice el proceso descrito en el párrafo anterior, y también, sea accesible vía Web, ya que existen organizaciones que no se encuentran en el Distrito Federal (donde se encuentran las oficinas centrales) y requieren estar conectados remotamente.

El flujo principal del FONOL, en su situación actual, es mostrado en las figuras 1.1 y 1.2:

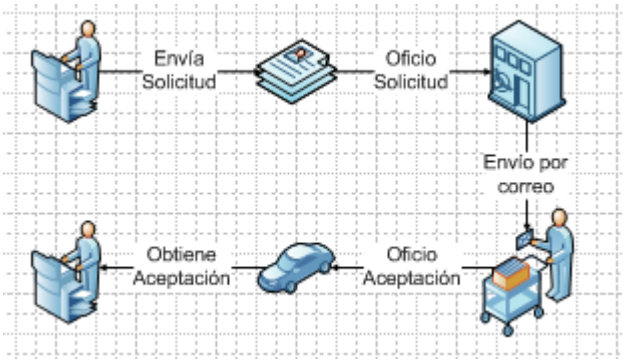


Figura 1.1
Solicitud de ingreso al FONOL, por parte de una compañía.

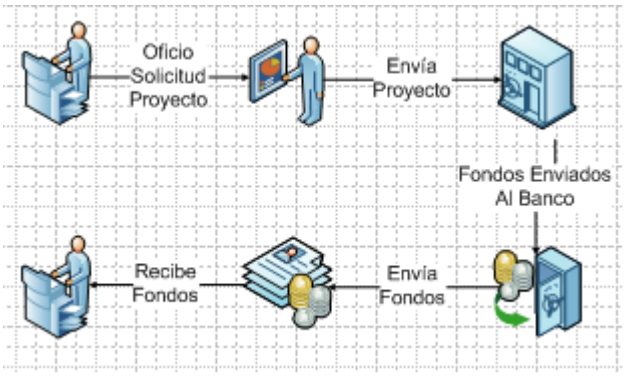


Figura 1.2
Solicitud de fondos para un proyecto en el FONOL.

1.2. Necesidades del sistema

Considerando lo anterior y realizando un análisis previo de la información, se requiere que el sistema realice lo siguiente:

- Dar de alta en el sistema a organizaciones que deseen ser financiadas.
- Capturar los datos en el sistema de la organización y del representante.
- Se requiere que los delegados (representantes del FONOL por cada estado) revisen las propuestas de su estado y acepten a los aspirantes que, de acuerdo al presupuesto, se consideren necesitados económicamente. Los resultados pueden ser los siguientes:
    - Si la organización es aceptada, entonces se convertirá en beneficiado, y podrá acceder al sistema con la contraseña que ya proporcionó.
    - Si la organización no es aceptada, entonces podrá hacer otra petición para el mes siguiente, teniendo en cuenta que solo podrá hacer 3 intentos para ser beneficiado.
- El beneficiado podrá proponer proyectos que requieran fondos, y llenará una solicitud que sustente al proyecto. Un proyecto puede tener cuatro estatus: solicitado, autorizado, viable o rechazado.
- El delegado es la única persona que puede cambiar el estatus de un proyecto.
- Una vez que el proyecto se haya propuesto, el delegado de un estado, o personal del mismo, lo revisará y determinará si el proyecto merece ser financiado.
    - En caso de que el proyecto lo requiera y que haya el presupuesto suficiente para financiarlo, el delegado le cambiará el estatus al proyecto como autorizado, y se le asignará el monto deseado.
    - En caso de que el proyecto no lo merezca, el delegado le asignará el estatus de rechazado, lo cual significa que el proyecto será descartado definitivamente.

- Si el delegado observa un proyecto interesante, pero no cuenta con el presupuesto para financiarlo, el delegado dejará el estatus de viable y lo podrá autorizar después.
- Para que el delegado pueda financiar proyectos, requerirá del presupuesto suficiente para apoyarlos. Este presupuesto le será asignado por el gobierno, que será el que determinará cuanto presupuesto requiere cada entidad federativa.
- Para que el gobierno pueda tomar una decisión más acertada, se apoyará de los reportes que le indiquen como se están distribuyendo los fondos.

Todo el proceso se resume en la figura 1.3:

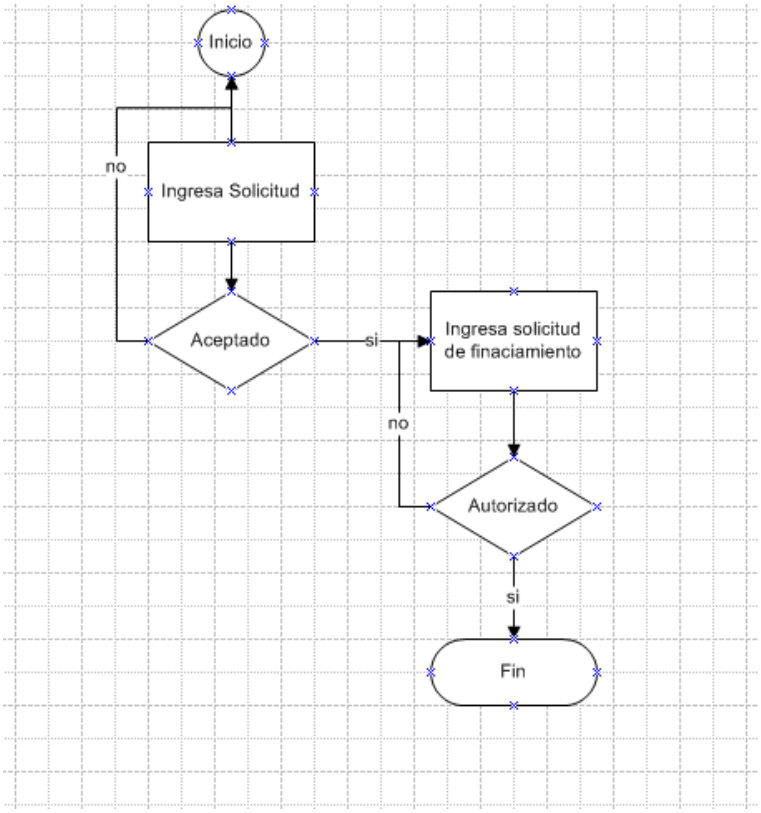


Figura 1.3

Esquema básico del flujo de información requerido.

1.3. Objetivo General del Sistema

- Desarrollar un sistema capaz de administrar fondos para organizaciones no lucrativas.
- Agilizar el proceso de financiamiento de las organizaciones, para no depender de oficios por escrito.
- Brindar acceso vía web, para evitar que los representantes de las organizaciones, tengan que desplazarse desde su estado a las oficinas centrales del FONOL, ya que éste es el único lugar donde se aceptan las solicitudes.
- Proveer de un sistema de seguridad para autentificar a los usuarios, y asignar roles para permitirles o no, consultar partes esenciales del sistema.

Todo esto esta esquematizado en la figura 1.4, en donde, el sistema web FONOL será el único medio para administrar la información:

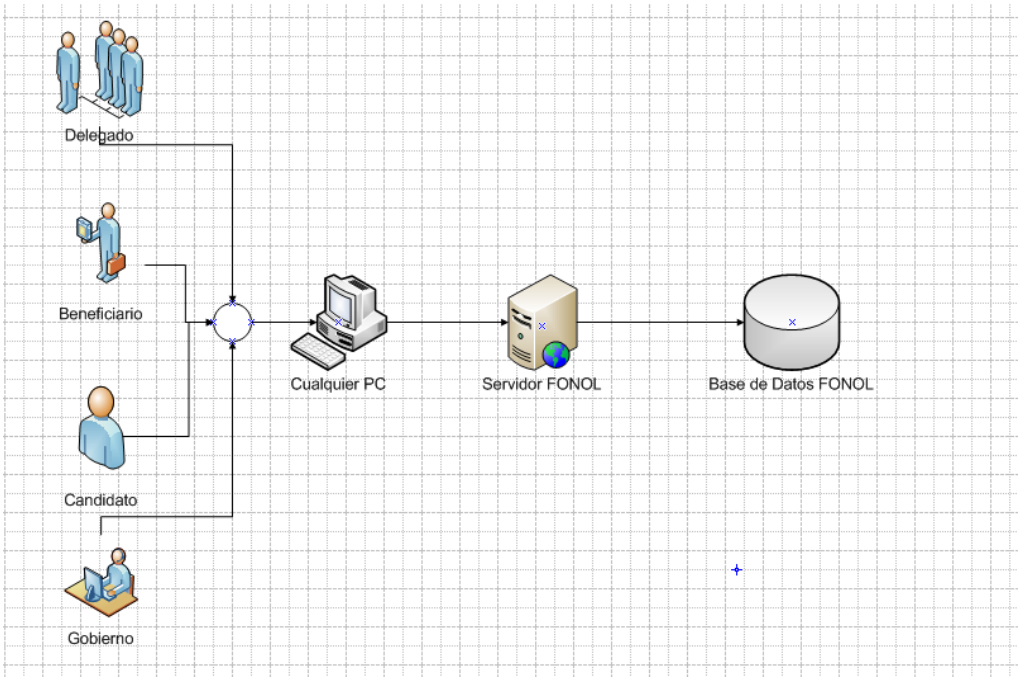


Figura 1.4

Esquema del funcionamiento deseado.

Capítulo 2

2. Marco Teórico

En las secciones siguientes, se dará un panorama de las asignaturas que servirán de base para realizar esta tesis.

2.1. Ingeniería de Software

Ingeniería de software es la disciplina o área de la informática que ofrece métodos y técnicas para desarrollar y mantener software de calidad.
Esta ingeniería trata con áreas muy diversas de la informática y de las ciencias de la computación, tales como construcción de compiladores, sistemas operativos, o desarrollos Intranet/Internet, abordando todas las fases del ciclo de vida del desarrollo de cualquier tipo de sistema de información y aplicables a infinidad de áreas (negocios, investigación científica, medicina, producción, logística, banca, control de tráfico, meteorología, derecho, Internet Intranet, etc.).

Dentro de la Ingeniería de Software existen varias metodologías que nos permiten llevar un control del desarrollo de un sistema, a continuación se mostrarán algunas de las más importantes y sus características.

2.1.1. Metodologías de Ingeniería de Software

La metodología, dentro de la ingeniería de software, se encarga de elaborar estrategias de desarrollo de software que promuevan prácticas adoptativas en vez de predictivas; centradas en las personas o los equipos, orientadas hacia la funcionalidad y la entrega, de comunicación intensiva y que requieren implicación directa del cliente.
A continuación se describirán las metodologías más famosas y sus características.

2.1.1.1. Metodología en cascada

En Ingeniería de software, el desarrollo en cascada, también llamado modelo en cascada, es el enfoque metodológico que ordena rigurosamente las etapas del ciclo de vida del software, de forma tal que el inicio de cada etapa debe esperar a la finalización de la inmediatamente anterior.

2.1.1.1.1. Fases

Las fases que componen la metodología en cascada son:

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

Y lo representamos con figura 2.1:

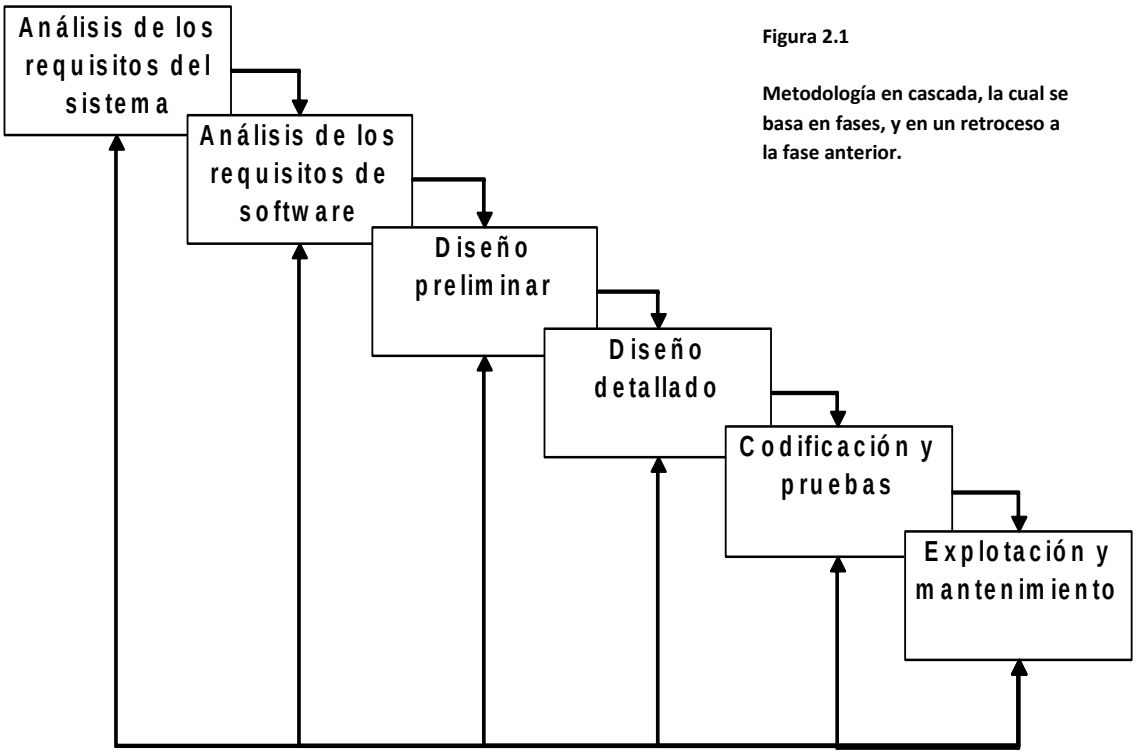


Figura 2.1

Metodología en cascada, la cual se basa en fases, y en un retroceso a la fase anterior.

De esta forma, cualquier error de diseño detectado en la etapa de prueba conduce necesariamente al rediseño y nueva programación del código afectado, aumentando los costes del desarrollo. La palabra cascada sugiere, mediante la metáfora de la fuerza de la gravedad, el esfuerzo necesario para introducir un cambio en las fases más avanzadas de un proyecto.

Si bien ha sido ampliamente criticado desde el ámbito académico y la industria, sigue siendo el paradigma más seguido al día de hoy.

Las fases de la metodología en cascada son explicadas a continuación:

- Análisis de requisitos
    - Se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. De esta fase surge un documento llamado documento de especificación de requisitos, que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos.
    - Es importante señalar que en esta etapa se debe analizar todo lo que se requiere del sistema y será aquello lo que seguirá en las

siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software.

- Diseño del Sistema
    - Se descompone y organiza el sistema en elementos que puedan elaborarse por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el documento de diseño del software, que contiene la descripción de la estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras.
- Diseño del Programa
    - Es la fase en donde se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario, así como también el análisis necesario para saber que herramientas utilizar en la etapa de codificación.
- Codificación
    - Es la fase de programación o implementación, propiamente dicha. Aquí se implementa el código fuente, haciendo uso de prototipos, así como pruebas y ensayos para corregir errores.
    - Dependiendo del lenguaje de programación y su versión, se crean las librerías y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.
- Pruebas
    - Los elementos, ya programados, se ensamblan para componer el sistema y se comprueba que funciona correctamente antes de ser puesto en explotación.
- Implantación
    - El software obtenido se pone en producción. Se implantan los niveles software y hardware que componen el proyecto. La implantación es la fase con más duración y con más cambios en el ciclo de elaboración de un proyecto. Es una de las fases finales del proyecto.

- Durante la explotación del sistema pueden surgir cambios para corregir errores o bien para introducir mejoras. Todo ello se recoge en los documentos de cambios.

Ya explicadas las fases principales es importante comparar sus ventajas y desventajas, para así, determinar cual se escogerá.

2.1.1.1.2. Ventajas y Desventajas

Las desventajas son:

- En la vida real, un proyecto rara vez sigue una secuencia lineal, esto crea una mala implementación del modelo, lo cual hace que lo lleve al fracaso.
- Difícilmente un cliente va a establecer al principio todos los requerimientos necesarios, por lo que provoca un gran atraso trabajando en este modelo, ya que este es muy restrictivo y no permite movilizarse entre fases.
- Los resultados y/o mejoras no son visibles, el producto se ve recién cuando este esté finalizado, lo cual provoca una gran inseguridad por parte del cliente que anda ansioso de consultar avances en el producto. Esto también implica toparse con requerimientos que no se habían tomado en cuenta, y que surgieron al momento de la implementación, lo cual provocara que se regrese nuevamente a la fase de requerimientos.

Por otro lado, las ventajas son:

- Se tiene todo bien organizado y no se mezclan las fases.
- Es perfecto para proyectos que son rígidos, y además donde se especifiquen muy bien los requerimientos y se conozca muy bien la herramienta a utilizar.

2.1.1.2. Metodología incremental

La idea principal detrás de la metodología incremental es el mejoramiento iterativo, el cual trata de desarrollar un sistema de programas de manera incremental, permitiéndole al desarrollador sacar ventaja de lo que se ha aprendido a lo largo del desarrollo anterior, incrementando, versiones entregables del sistema. El aprendizaje viene de dos vertientes: el desarrollo del sistema, y su uso (mientras sea posible). Los pasos claves en el proceso eran comenzar con una implementación simple de los requerimientos del sistema, e iterativamente mejorar la secuencia evolutiva de versiones hasta que el sistema completo esté implementado. En cada iteración, se realizan cambios en el diseño y se agregan nuevas funcionalidades y capacidades al sistema.

A continuación se mostrarán las fases de esta metodología:

2.1.1.2.1. Fases

Las fases de la metodología incremental son:

- Análisis de requerimientos del sistema
- Análisis de requerimientos de software
- Diseño
- Codificación
- Mantenimiento:

La metodología incremental puede ser representada mediante la figura 2.2:

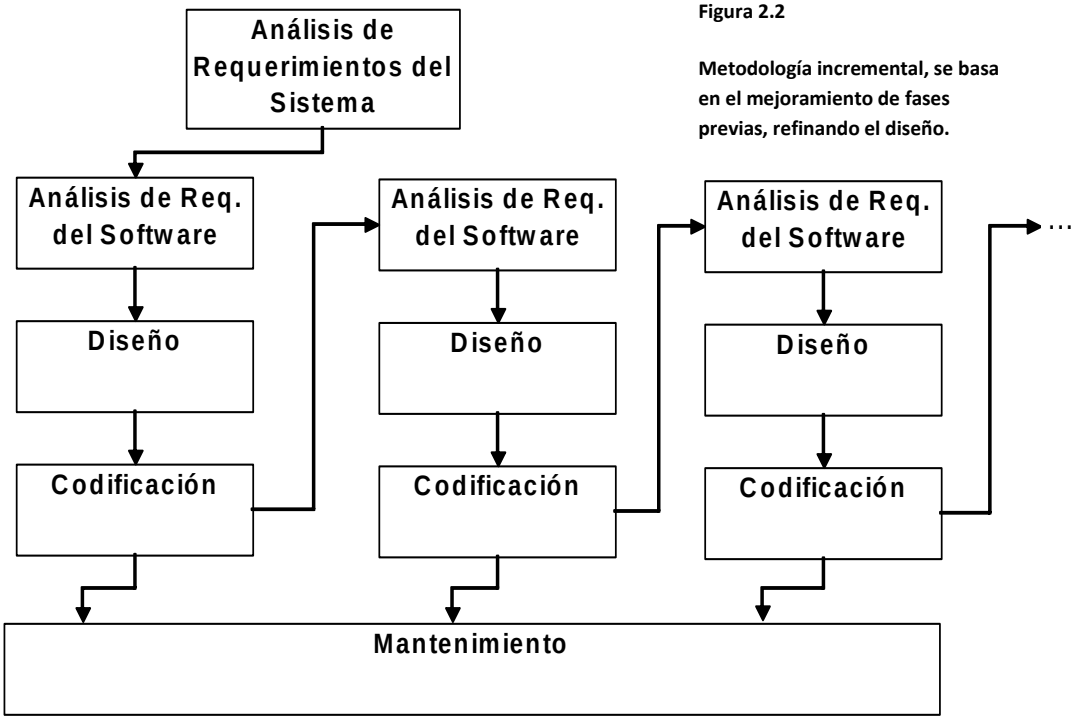


Figura 2.2

Metodología incremental, se basa en el mejoramiento de fases previas, refinando el diseño.

A continuación se mostrarán las ventajas y desventajas del modelo:

2.1.1.2.2. Ventajas y Desventajas

Las ventajas son:

- Se evitan proyectos largos y se entrega “Algo de valor” a los usuarios con cierta frecuencia
- Requiere desarrolladores experimentados
- El resultado puede ser muy positivo

Las desventajas son:

- Se requiere un buen mantenimiento de las versiones, ya que si se descuida, se pueden salir de control.
- Difícil de evaluar el costo total.
- Difícil de aplicar a sistemas transaccionales que tienden a ser integrados y a operar como un todo.
- Los errores en los requisitos se detectan tarde.

2.1.1.3. Metodología en Espiral

Es un modelo de ciclo de vida, el cual se conforma en una espiral con bucles (giros), cada bucle representa un conjunto de actividades. Las actividades no están fijadas a priori, sino que las siguientes se eligen en función del análisis de riesgo, comenzando por el bucle interior.

A continuación se mostrarán las fases del modelo en espiral:

2.1.1.3.1. Fases

Para cada ciclo habrá cuatro fases o actividades:

- Determinar o fijar objetivos
    - Fijar también los productos definidos a obtener: requerimientos, especificación, manual de usuario.
    - Fijar las restricciones.
- Análisis del riesgo
    - Se estudian todos los riesgos potenciales y se seleccionan una o varias alternativas propuestas para reducir o eliminar los riesgos.
- Desarrollar, verificar y validar (probar)
    - Tareas de la actividad propia y de prueba.
    - Análisis de alternativas e identificación resolución de riesgos.
    - Dependiendo del resultado de la evaluación de los riesgos, se elige un modelo para el desarrollo, el que puede ser cualquiera de los otros existentes, como formal, evolutivo, cascada, etc. Así, si los riesgos en la interfaz de usuario son dominantes, un modelo de desarrollo apropiado podría ser la construcción de prototipos evolutivos.
- Planificar

- Revisamos todo lo hecho, evaluándolo, y con ello decidimos si continuamos con las fases siguientes y planificamos la próxima actividad.

El diagrama en espiral lo representamos en la figura 2.3:

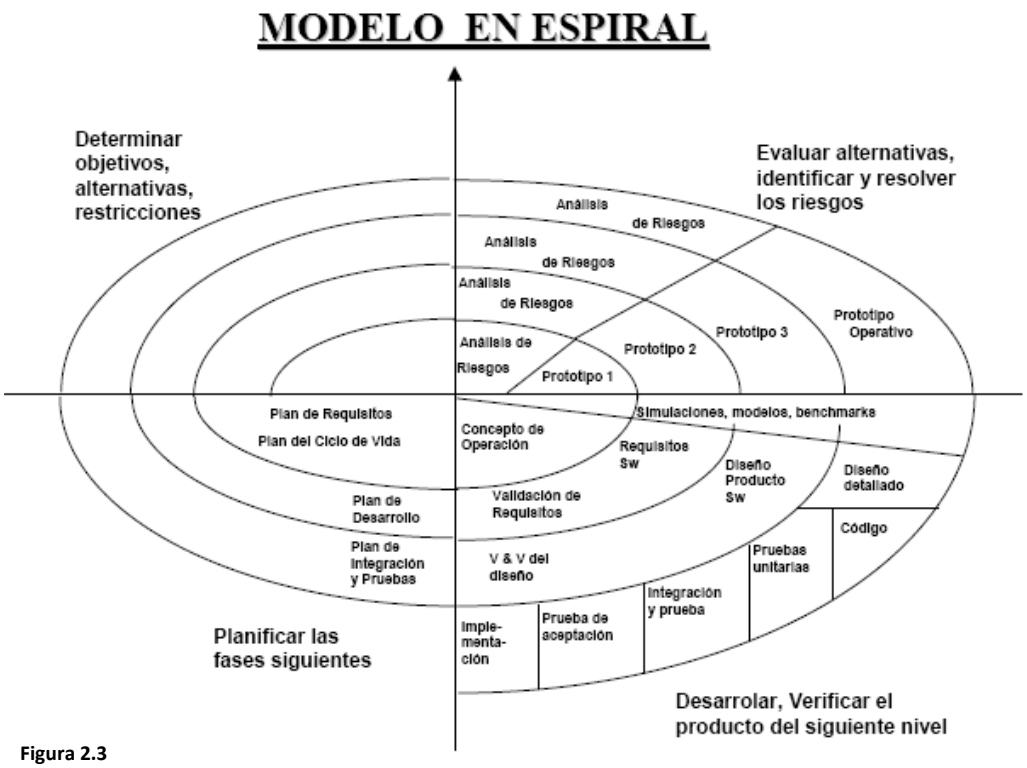


Figura 2.3

Metodología en espiral

2.1.1.3.2. Ventajas y Desventajas

Las ventajas son:

- Proporciona una representación rápida de lo que se requiere
- Se asigna prioridad a una actividad de acuerdo a su riesgo
- Es ágil con proyectos pequeños

Por otro parte, las desventajas son:

- Al analizarse respecto a riesgos, es difícil decir con certeza si el riesgo está bien calificado.

- No es tan organizado, lo cual trae errores
- No es tan aplicable en proyectos grandes

2.1.1.4. Metodología RUP

El Proceso Unificado de Rational (Rational Unified Process en inglés, habitualmente resumido como RUP) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

RUP no es un sistema con pasos establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización.

También se conoce por este nombre al software desarrollado por Rational, hoy propiedad de IBM, el cual incluye información entrelazada de diversos artefactos y descripciones de las diversas actividades.

Originalmente se diseñó un proceso genérico y de dominio público, el Proceso Unificado, y una especificación más detallada, el Rational Unified Process, que se vendiera como producto independiente.

En la siguiente sección, se mostrarán las fases de las cuales está compuesto RUP:

2.1.1.4.1. Fases

Dentro de esta metodología se encuentran las siguientes fases:

- Iniciación
- Elaboración
- Construcción
- Transición

Las primeras iteraciones (en las fases de Inicio y Elaboración) se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y al establecimiento de una arquitectura.

Durante la fase de inicio las iteraciones hacen mayor énfasis en actividades de modelado del negocio y de requerimientos.

En la fase de elaboración, las iteraciones se orientan al desarrollo de la arquitectura, abarcan más los flujos de trabajo de requerimientos, modelo de negocios (refinamiento), análisis, diseño y una parte de implementación.

En la fase de construcción, se lleva a cabo la construcción del producto por medio de una serie de iteraciones.

Para cada iteración se selecciona algunos casos de uso, se refina su análisis y diseño y se procede a su implementación y pruebas. Se realiza una pequeña cascada para cada ciclo. Se realizan tantas iteraciones hasta que se termine la implementación de la nueva versión del producto.

En la fase de transición se pretende garantizar que se tiene un producto preparado para su entrega a la comunidad de usuarios.

Como se puede observar en cada fase participan todas las disciplinas, pero dependiendo de la fase, el esfuerzo dedicado a una disciplina varía.

Todas las fases están representadas en la figura 2.4:

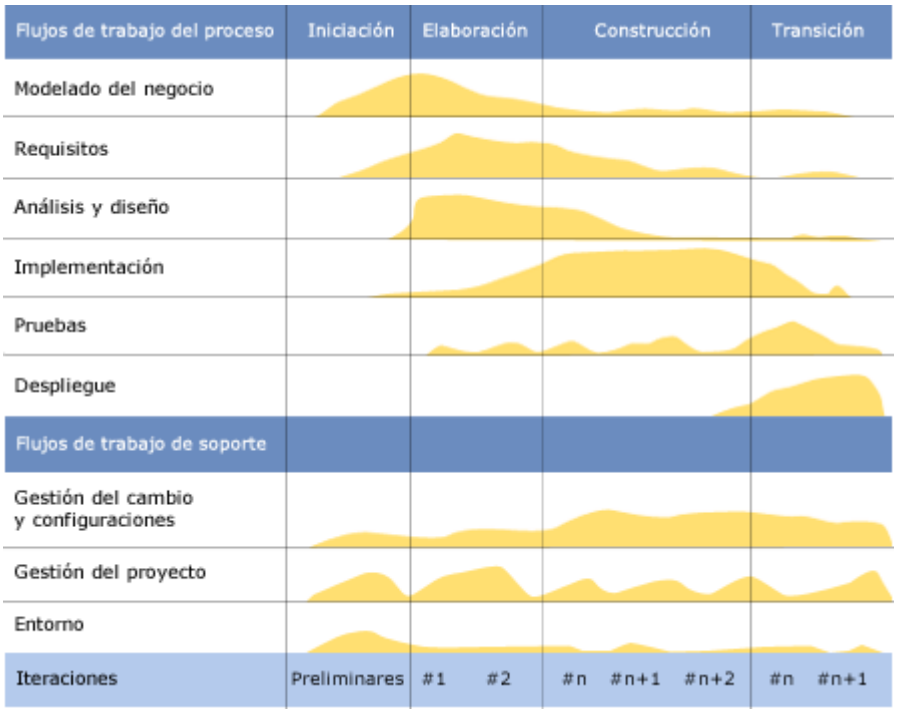


Figura 2.4

Metodología RUP, se basa en 4 fases, cada una abarcando más de una fase de la metodología incremental.

Las fases de RUP generalmente son mal aplicadas en un estilo cascada, por su alta complejidad no es adecuado para proyectos pequeños.

2.1.1.4.2. Ventajas y Desventajas

Las ventajas de la metodología RUP son:

- Mitigación temprana de posibles riesgos altos
- Progreso visible en las primeras etapas
- Temprana retroalimentación que se ajuste a las necesidades reales
- Gestión de la complejidad
- Conocimiento adquirido en una iteración puede aplicarse de iteración a iteración

Por otra parte, las desventajas son:

- Si el conjunto de documentos y artefactos no son concebidos tal y como se plantea en RUP, dicha documentación solo servirá para ser archivada, lo cual no genera valor respecto a la calidad del desarrollo, y evoluciona en problemas más complejos (aplicación que no satisface los requerimientos, diseño de la estructura no coincide con la estructura final de la aplicación, etc...)
- Es necesario incluir a más personas en el equipo de desarrollo: Especialistas en los diseños y evolución de casos de uso, de los modelos de análisis y diseño, de los modelos de implementación, etc.… Cabe indicar que la tendencia es a reducir cada vez más los equipos de desarrollo, esto no se debe a una simple política de reducción de costos, sino a la reducción de la complejidad asociada a la comunicación entre los miembros del equipo.
- Lo más importante en el desarrollo de un producto informático es el propio desarrollo, en RUP se gasta posiblemente demasiado tiempo para pasar a la fase de desarrollo

Una vez explicadas las metodologías de programación en las cuales se apoyará para el desarrollo de la tesis, es momento de mencionar a las Bases de Datos, ya

que ellas con una parte vital del desarrollo del sistema, y es conveniente mencionarlas en el marco teórico.

2.2. Base de datos

Antes de hablar sobre los modelados de Base de Datos es preciso, dar un concepto claro y entendible de lo que es.

Una base de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. En este sentido, una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta. En la actualidad, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos están en formato digital, que ofrece un amplio rango de soluciones al problema de almacenar datos.

Para representar estos datos se utilizan de modelos, los cuales se describen a continuación:

2.2.1. Modelos

El modelo, o modelado, es la forma de representar los datos para su posterior almacenamiento en una base de datos. Está más enfocado al diseño conceptual, que físico de los datos.

Algunos de los modelos más utilizados son los siguientes:

2.2.1.1. Modelo relacional

En este modelo todos los datos son almacenados en relaciones, y como cada relación es un conjunto de datos, el orden en el que estos se almacenen no tiene mayor relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la ventaja de que es más fácil de entender y de utilizar por un usuario no experto. La información puede ser recuperada o almacenada por medio de consultas que ofrecen una amplia flexibilidad y poder para administrar la información.

Este modelo considera la base de datos como una colección de relaciones. De manera simple, una relación representa una tabla que no es más que un conjunto de filas, cada fila es un conjunto de campos y cada campo representa un valor que interpretado describe el mundo real. Cada fila también se puede denominar registro y a cada columna se le puede llamar campo o atributo.

Para manipular la información se utilizará un lenguaje relacional, actualmente se cuenta con dos lenguajes formales el álgebra relacional y el cálculo relacional. El álgebra relacional permite describir la forma de realizar una consulta, en cambio, el cálculo relacional sólo indica lo que se desea devolver.

El lenguaje más común para construir las consultas a bases de datos relacionales es SQL, o Lenguaje Estructurado de Consultas, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

2.2.1.2. Base de datos relacional

En una base de datos relacional, todos los datos se almacenan y se acceden por medio de relaciones. Las relaciones que almacenan datos son llamados "relaciones base" y su implementación es llamada "tabla". Otras relaciones no almacenan datos, pero que son calculadas al aplicar operaciones relacionales. Estas relaciones son llamadas "relaciones derivadas" y su implementación es llamada "vista" o "consulta". Las relaciones derivadas son convenientes ya que expresan información de varias relaciones actuando como si fuera una sola.

Una restricción es una condición que obliga el cumplimiento de ciertas condiciones en la base de datos. Algunas no son determinadas por los usuarios, sino que son inherentemente definidas por el simple hecho de que la base de datos sea relacional. Algunas otras restricciones las puede definir el usuario, por ejemplo, usar un campo con valores enteros entre 1 y 10.

Las restricciones proveen un método de implementar reglas en la base de datos. Las restricciones restringen los datos que pueden ser almacenados en las tablas. Usualmente se definen usando expresiones que dan como resultado un valor booleano, indicando si los datos satisfacen la restricción o no.

Las restricciones no son parte formal del modelo relacional, pero son incluidas porque juegan el rol de organizar mejor los datos. Las restricciones son muy discutidas junto con los conceptos relacionales.

Cada tabla puede tener uno o más campos cuyos valores identifican de forma única cada registro de dicha tabla, es decir, no pueden existir dos o más registros diferentes cuyos valores en dichos campos sean idénticos. Este conjunto de campos se llama clave única.

Pueden existir varias claves únicas en una determinada tabla, y a cada una de éstas suele llamársele candidata a clave primaria.

2.2.1.3. Modelo entidad-relación (MER)

Es una representación conceptual de la información. Mediante una serie de procedimientos se puede pasar del modelo E-R a otros, como por ejemplo el modelo relacional.

El modelado entidad-relación es una técnica para el modelado de datos, no es la única técnica pero sí la más utilizada. Brevemente consiste en los siguientes pasos:

1. Se parte de una descripción textual del problema o sistema de información a automatizar (los requisitos).
2. Se hace una lista de los sustantivos y verbos que aparecen.
3. Los sustantivos son posibles entidades o atributos.
4. Los verbos son posibles relaciones.
5. Analizando las frases se determina la cardinalidad de las relaciones y otros detalles.
6. Se elabora el diagrama (o diagramas) entidad-relación.
7. Se completa el modelo con listas de atributos y una descripción de otras restricciones que no se pueden reflejar en el diagrama.

   Dado lo rudimentario de esta técnica, se necesita cierto entrenamiento y experiencia para lograr buenos modelos de datos.

El modelado de datos no acaba con el uso de esta técnica. Son necesarias otras técnicas para lograr un modelo directamente implementable en una base de datos. Brevemente:

- Transformación de relaciones múltiples en binarias.
- Normalización de una base de datos de relaciones (algunas relaciones pueden transformarse en atributos y viceversa).
- Conversión en tablas (en caso de utilizar una base de datos relacional).

2.2.1.4. Mapeo objeto-relacional (ORM)

El mapeo objeto-relacional, (o sus siglas en inglés O/RM, ORM, y O/R mapping) es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional. En la práctica esto crea una base de datos orientada a objetos virtual, sobre la base de datos relacional. Esto posibilita el uso de las características propias de la orientación a objetos (básicamente herencia y polimorfismo). Hay paquetes comerciales y de uso libre disponibles que desarrollan el mapeo relacional de objetos, aunque algunos programadores prefieren crear sus propias herramientas ORM.

En la programación orientada a objetos, las tareas de manejo de datos son implementadas generalmente por la manipulación de objetos, los cuales son casi siempre valores no escalares. Para ilustrarlo, considere el ejemplo de una entrada en una libreta de direcciones, que representa a una sola persona con cero o más números telefónicos y cero o más direcciones. En una implementación orientada a objetos, esto puede ser modelado por un "objeto persona" con "campos" que almacenan los datos de dicha entrada: el nombre de la persona, una lista de números telefónicos y una lista de direcciones. La lista de números telefónicos estaría compuesta por "objetos de números telefónicos" y así sucesivamente. La entrada de la libreta de direcciones es tratada como un valor único por el lenguaje de programación (puede ser referenciada por una sola variable, una instancia). Varios métodos pueden asociarse con el objeto, como uno que devuelva el número telefónico preferido, la dirección de su casa, etc.

Sin embargo, muchos productos populares de base de datos, como los productos SQL DBMS, solamente puede almacenar y manipular valores escalares como enteros y cadenas, organizados en tablas.

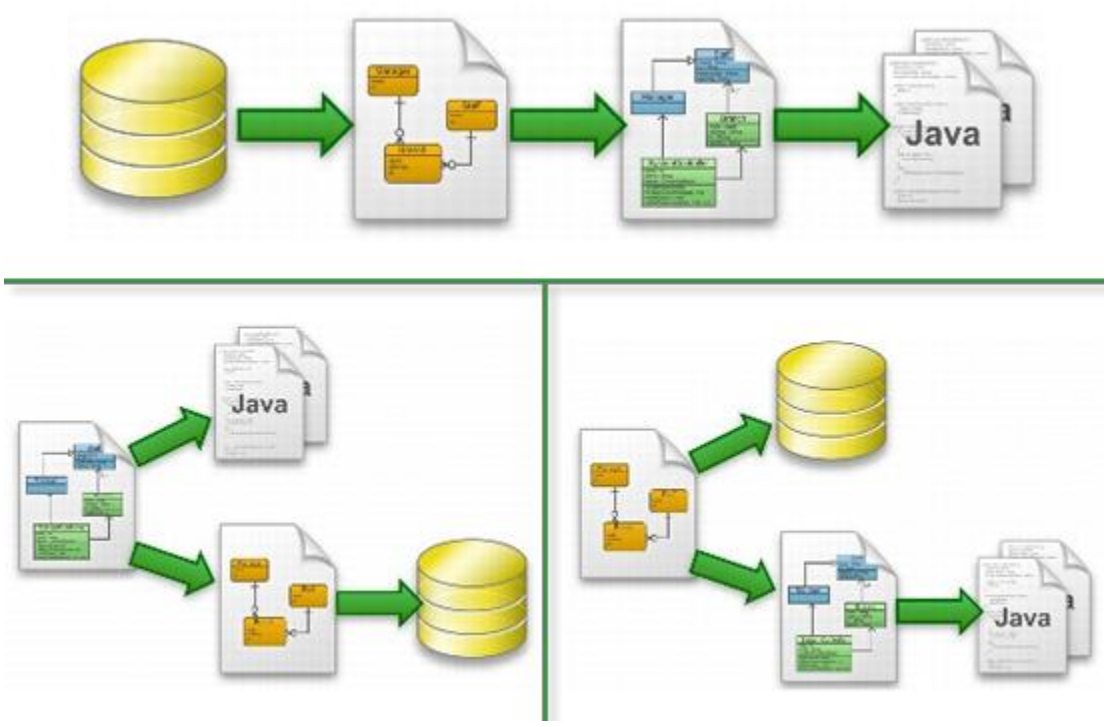
El programador debe convertir los valores de los objetos en grupos de valores simples para almacenarlos en la base de datos (y volverlos a convertir luego de recuperarlos de la base de datos), o solo usar valores escalares simples en el

programa. El mapeo relacional de objetos es utilizado para implementar esta primera aproximación.

En la figura 2.5, podemos observar, una representación gráfica del mapeo Objeto-Relacional en Java. En la parte inferior, se muestra el modelo antiguo en Java para acceder a los datos, en el cual, para acceder a nuestra base de datos tenemos que tratar de construir objetos parecidos al modelo relacional, extraer los datos del modelo relacional, guardarlos en Java, para después hacer las actualizaciones pertinentes y posteriormente regresarlos a su representación relacional. El modelo relacional, nos brinda la ventaja de que sólo es necesario indicarle a Java como se mapean los objetos y las relaciones de la base de datos, para simplemente manipular directamente los objetos, e instantáneamente se reflejen las actualizaciones en base de datos, sin tener que programar todas las modificaciones.

Figura 2.5

Mapeo Objeto Relacional.



Una vez mencionado lo correspondiente a los Modelos de Base de Datos existentes, toca mencionar los Manejadores de Base de Datos que brindarán el soporte al modelo elegido:

2.2.2. Manejadores de Base de Datos

Antes de dar un listado de los manejadores de base de datos, es necesario saber que es un manejador y cuáles son las funciones que realiza.
El sistema manejador de bases de datos es la porción más importante del software de un sistema de base de datos. Un DBMS es una colección de numerosas rutinas de software interrelacionadas, cada una de las cuales es responsable de alguna tarea específica.

Las funciones principales de un DBMS son:

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

El DBMS es conocido también como gestor de base de datos.

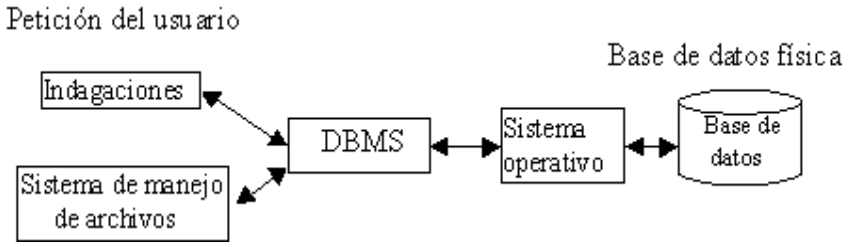


Figura 2.6

DBMS.

La figura 2.6 muestra el DBMS como interface entre la base de datos física y las peticiones del usuario. El DBMS interpreta las peticiones de entrada/salida del usuario y las manda al sistema operativo para la transferencia de datos entre la unidad de memoria secundaria y la memoria principal.
En sí, un sistema manejador de base de datos es el corazón de la base de datos ya que se encarga del control total de los posibles aspectos que la puedan afectar.

Ahora se dará un resumen de las características más sobresalientes de cada uno de ellos.

2.2.2.1. Oracle

Oracle es un sistema de gestión de base de datos relacional (o RDBMS), y se considera como uno de los sistemas de bases de datos más completos, destacando su:

- Soporte de transacciones.
- Estabilidad.
- Escalabilidad.
- Soporte multiplataforma.

Ha sido criticada por algunos especialistas la seguridad de la plataforma, y las políticas de suministro de parches de seguridad, modificadas a comienzos de 2005 y que incrementan el nivel de exposición de los usuarios. En los parches de actualización provistos durante el primer semestre de 2005 fueron corregidas 22 vulnerabilidades públicamente conocidas, algunas de ellas con una antigüedad de más de 2 años.

2.2.2.2. SQL Server

Microsoft SQL Server es un sistema de gestión de bases de datos relacionales (SGBD) basado en el lenguaje Transact-SQL, y específicamente en Sybase IQ, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Las características que ofrecen son:

- Soporte de transacciones.
- Escalabilidad, estabilidad y seguridad.
- Soporta procedimientos almacenados.
- Incluye también un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información.
- Permite administrar información de otros servidores de datos.

Este sistema incluye una versión reducida, llamada MSDE con el mismo motor de base de datos pero orientado a proyectos más pequeños, que en su versión 2005 pasa a ser el SQL Express Edition, que se distribuye en forma gratuita.

Es común desarrollar completos proyectos complementando Microsoft SQL Server y Microsoft Access a través de los llamados ADP (Access Data Project). De esta forma se complementa la base de datos (Microsoft SQL Server), con el entorno de desarrollo (VBA Access), a través de la implementación de aplicaciones de dos capas mediante el uso de formularios Windows.

Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para varias plataformas de desarrollo, entre ellas .NET, pero el servidor sólo está disponible para Sistemas Operativos Windows.

2.2.2.3. PostgreSQL

De las características más sobresalientes de PostgreSQL, destacan:

- Alta concurrencia: Mediante un sistema denominado MVCC (Acceso concurrente multiversión, por sus siglas en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo *commit*. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos.
- Amplia variedad de tipos nativos
- Adicionalmente los usuarios pueden crear sus propios tipos de datos, los que pueden ser por completo indexables gracias a la infraestructura GiST de PostgreSQL. Algunos ejemplos son los tipos de datos GIS creados por el proyecto Postgis.

Ya abordado lo referente a los Manejadores de Base de Datos, se darán los diversos lenguajes de programación existentes, para poder elegir el más adecuado con el sistema:

2.3. Lenguajes de programación

Un lenguaje de programación es un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones, utilizado para controlar el comportamiento físico y lógico de una máquina.

Aunque muchas veces se usan los términos 'lenguaje de programación' y 'lenguaje informático' como si fuesen sinónimos, no tiene por qué ser así, ya que los lenguajes informáticos engloban a los lenguajes de programación y a otros más, como, por ejemplo, el HTML (lenguaje para el marcado de páginas web que no es propiamente un lenguaje de programación).

Un lenguaje de programación permite a uno o más programadores especificar de manera precisa sobre qué datos debe operar una computadora, cómo estos datos deben ser almacenados o transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. Todo esto, a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o natural, tal como sucede con el lenguaje Léxico. Una característica relevante de los lenguajes de programación es precisamente que más de un programador puedan tener un conjunto común de instrucciones que puedan ser comprendidas entre ellos para realizar la construcción del programa de forma colaborativa.

Existen varias formas de conceptualizar un problema, visto desde un punto de vista algorítmico, y a esta forma se le conoce como: metodología o paradigma, que es la forma y los métodos para llegar a un objetivo. A continuación se mostrarán las metodologías más utilizadas en el desarrollo de sistemas.

2.3.1. Metodologías de Programación

En esta sección, se dará un panorama general de las metodologías más importantes, su importancia en el tiempo y sus principales características.

2.3.1.1. Estructurada

La programación estructurada es una forma de programar de forma clara, para ello utiliza únicamente tres estructuras: secuencial, selectiva e iterativa; siendo innecesario y no permitiéndose el uso de la instrucción o instrucciones de transferencia incondicional (GOTO).

Hoy en día, las aplicaciones informáticas son mucho más ambiciosas que las necesidades de programación existentes en los años 60, principalmente debido a las aplicaciones gráficas, por lo que las técnicas de programación estructurada no son suficientes. Ello ha llevado al desarrollo de nuevas técnicas tales como la programación orientada a objetos y el desarrollo de entornos de programación que facilitan la programación de grandes aplicaciones.

2.3.1.2. Orientada a Objetos

La Programación Orientada a Objetos (POO u OOP según sus siglas en inglés) es un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas de computadora. Está basado en varias técnicas, incluyendo herencia, modularidad, polimorfismo y encapsulamiento.
La programación orientada a objetos es una nueva forma de programar que trata de encontrar una solución a estos problemas. Introduce nuevos conceptos, que superan y amplían conceptos antiguos ya conocidos. Entre ellos destacan los siguientes:

- Clase: definiciones de las propiedades y comportamiento de un tipo de objeto concreto. La instanciación es la lectura de estas definiciones y la creación de un objeto a partir de ellas. Es la facilidad mediante la cual la clase D ha definido en ella cada uno de los atributos y operaciones de C, como si esos atributos y operaciones hubiesen sido definidos por la misma D.
- Objeto: entidad provista de un conjunto de propiedades o atributos (datos) y de comportamiento o funcionalidad (métodos). Se corresponde con los objetos reales del mundo que nos rodea, o a objetos internos del sistema (del programa). Es una instancia a una clase.
- Método: algoritmo asociado a un objeto (o a una clase de objetos), cuya ejecución se desencadena tras la recepción de un "mensaje". Desde el punto de vista del comportamiento, es lo que el objeto puede hacer. Un método puede producir un cambio en las propiedades del objeto, o la generación de un "evento" con un nuevo mensaje para otro objeto del sistema.
- Evento: un suceso en el sistema (tal como una interacción del usuario con la máquina, o un mensaje enviado por un objeto). El sistema maneja el evento enviando el mensaje adecuado al objeto pertinente. También se puede definir como evento, a la reacción que puede desencadenar un objeto, es decir la acción que genera.

- Mensaje: una comunicación dirigida a un objeto, que le ordena que ejecute uno de sus métodos con ciertos parámetros asociados al evento que lo generó.
- Propiedad o atributo: contenedor de un tipo de datos asociados a un objeto (o a una clase de objetos), que hace los datos visibles desde fuera del objeto y esto se define como sus características predeterminadas, y cuyo valor puede ser alterado por la ejecución de algún método.
- Estado interno: es una variable que se declara privada, que puede ser únicamente accedida y alterada por un método del objeto, y que se utiliza para indicar distintas situaciones posibles para el objeto (o clase de objetos). No es visible al programador que maneja una instancia de la clase.
- Componentes de un objeto: atributos, identidad, relaciones y métodos.
- Representación de un objeto: un objeto se representa por medio de una tabla o entidad que esté compuesta por sus atributos y funciones correspondientes.

En comparación con un lenguaje imperativo, una "variable", no es más que un contenedor interno del atributo del objeto o de un estado interno, así como la "función" es un procedimiento interno del método del objeto.

Ya comprendidos los paradigmas de programación, en la siguiente sección se hablará acerca de los lenguajes más importantes hoy en día que nos permiten resolver estos paradigmas.

2.3.2. Lenguajes de Programación

En la siguiente sección, se mostrarán los diferentes lenguajes de programación vistos desde el punto de vista de sus fabricantes, y resaltando en todo momento, sus ventajas y características.

2.3.2.1. Microsoft .NET VB

Visual Basic .NET (VB.NET) es un lenguaje de programación orientado a objetos que se puede considerar una evolución de Visual Basic implementada sobre el framework .NET. Un framework, es la manera en la cual un grupo de librerías resuelven un problema común. Su introducción resultó muy controvertida, ya que debido a cambios significativos en el lenguaje VB.NET no es compatible hacia atrás con Visual Basic, cosa que causó gran división en la comunidad de desarrolladores de Visual Basic.
La gran mayoría de programadores de VB.NET utilizan el entorno de programación Microsoft Visual Studio .Net en alguna de sus versiones (Visual Studio .NET, Visual Studio .NET 2003 o Visual Studio .NET 2005),
Como pasa con todos los lenguajes de programación basados en .NET, los programas escritos en VB.NET requieren el Framework .NET para ejecutarse.

Las ventajas que trae consigo Visual Basic son:

- Es un lenguaje RAD (Desarrollo rápido de aplicaciones).
- Posee una curva de aprendizaje muy rápida.
- Integra el diseño e implementación de formularios de Windows.
- Permite usar con suma facilidad la plataforma de los sistemas Windows.
- El código en Visual Basic es fácilmente puede migrarse a otros lenguajes.
- Acostumbra a los desarrolladores a programar con eficiencia.

2.3.2.2. JAVA

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Las aplicaciones Java están típicamente compiladas en un código de máquina llamado bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

Las características más sobresalientes:

- Es un lenguaje que es compilado, generando ficheros de clases compilados, pero estas clases compiladas, son en realidad interpretadas por la máquina virtual de java. Siendo la máquina virtual de java la que mantiene el control sobre las clases que se estén ejecutando.
- Es un lenguaje multiplataforma: El mismo código java que funciona en un sistema operativo, funcionará en cualquier otro sistema operativo que tenga instalada la máquina virtual java.
- Es un lenguaje seguro: La máquina virtual, al ejecutar el código java, realiza comprobaciones de seguridad, además el propio lenguaje carece de características inseguras, como por ejemplo los punteros.
- Gracias al API de java podemos ampliar el lenguaje para que sea capaz de, por ejemplo, comunicarse con equipos mediante red, acceder a bases de datos, crear páginas HTML dinámicas, crear aplicaciones visuales al estilo Windows.

Con el incremento de la tecnología mundial, las aplicaciones empresariales se han incrementado exponencialmente, por lo cual se ha optado por tener aplicaciones

Web en lugar de aplicaciones de escritorio. En las secciones siguientes, se hablará de los sistemas Web para comprender como están construidos.

2.4. Sistemas Web

En la ingeniería software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web (HTML, JavaScript, Java, etc.) en la que se confía la ejecución al navegador.
Las aplicaciones web son populares debido a lo práctico del navegador web como cliente ligero, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales. Existen aplicaciones como los webmails, wikis, weblogs, tiendas en línea y la propia Wikipedia que son ejemplos bien conocidos de aplicaciones web.

Es importante mencionar que una página Web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo.

Por su accesibilidad y escalabilidad, las aplicaciones Web son lo más utilizado últimamente, por lo cual trataré de mostrar algunos servidores de aplicaciones Web más importantes.

2.4.1. Servidores de Aplicaciones

En informática se denomina servidor de aplicaciones a un servidor en una red de computadores que ejecuta ciertas aplicaciones.

Usualmente se trata de un dispositivo de software que proporciona servicios de aplicación a las computadoras cliente. Un servidor de aplicaciones generalmente gestiona la mayor parte (o la totalidad) de las funciones de lógica de negocio y de acceso a los datos de la aplicación. Los principales beneficios de la aplicación de la tecnología de servidores de aplicación son la centralización y la disminución de la complejidad en el desarrollo de aplicaciones.

Capítulo 3

3. Propuesta de solución

La propuesta de solución que se utilizará se ha tomado de acuerdo a las características propias del sistema, así como de las tecnologías disponibles en el mercado que faciliten el trabajo.

Se abordará cada una de ellas en una sección diferente para que se noten las características propias.

Capítulo 3

3. Propuesta de solución

La propuesta de solución que se utilizará se ha tomado de acuerdo a las características propias del sistema, así como de las tecnologías disponibles en el mercado que faciliten el trabajo.

Se abordará cada una de ellas en una sección diferente para que se noten las características propias.

3.1. Metodología a utilizar

De las metodologías mencionadas y evaluadas en el capítulo anterior, se eligió RUP, ya que entre sus características más importantes, destacan:

- Dirigido por casos de uso

  Los casos de usos se especifican, se diseñan, y los casos de uso finales son la fuente a partir de la cual los ingenieros de prueba construyen sus casos de prueba.

- Centrado en la arquitectura:

  Es muy importante esta característica, ya que nos da de una manera más ordenada la forma en que se van ensamblando los componentes de la aplicación, así como sus interrelaciones. La arquitectura del software involucra:

  - La estructura y el comportamiento
  - La funcionalidad
  - La facilidad de comprensión
  - La reutilización
  - La flexibilidad
  - El rendimiento
  - Las restricciones y compromisos económicos y tecnológicos
  - La estética

- Iterativo e incremental

  Es la propiedad más interesante que me hizo_elegir RUP, ya que nos permite iterar cuantas veces se necesite, para pulir nuestra aplicación de manera más limpia, sin afectar todo el conjunto.

3.2. Base de Datos a utilizar

La elección del manejador a utilizar, se evaluó ampliamente, ya que se contaba en el mercado, con manejadores robustos, de licencia muy cara y de difícil mantenimiento. Pero finalmente se utilizó PostgreSQL, las razones fueron las siguientes:

- Opensource: Esta cualidad es muy importante, ya que no se requiere de una licencia cara, como en el caso de SQL Server.
- Robusto: A pesar de que es un software libre, reúne todas las características de un Sistema Gestor de Base de Datos.
- Fácil: Es de muy fácil uso, y no se necesita de un DBA para operarlo. Además, una de las razones más influyentes fue, que es muy portable, ya que con un script se puede levantar todo el sistema.
- Modelo Objeto-Relacional simplificado: Es una característica importante, ya que nos permite de una manera sencilla generar el mapeo de los Objetos con el modelo Relacional, sin problemas de compatibilidad.

3.3. Lenguaje de programación a utilizar

De los lenguajes evaluados en el capítulo anterior, se eligió Java, por varias razones:

- Escalabilidad: Java me permite separar una aplicación Java en varias capas, y así tener una organización más limpia del código, es decir, se puede modificar una capa muy interna del sistema sin afectar a las capas exteriores.
- Independencia de plataforma: se eligió esta característica, ya que es sencillo desarrollar tanto el Linux como en Windows sin afectar la aplicación ni ninguno de sus componentes.
- Robusto: es un lenguaje robusto, ya que mediante sus modificadores de acceso, nos permite restringir estrictamente el acceso a las variables y a los métodos.

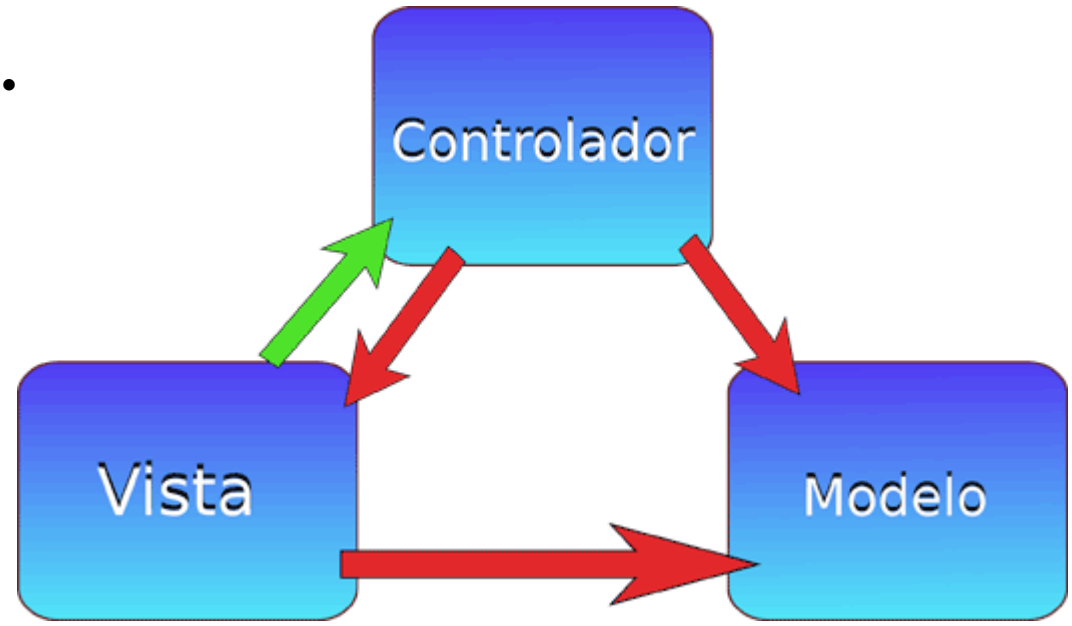
Por otra parte, ya elegido Java como lenguaje de programación, es necesario hacer algunas anotaciones referentes a las tecnologías dentro de Java que se utilizarán para el desarrollo del sistema.

En primera instancia, es necesario definir los Patrones de Diseño, ya que se apoyará de ellos para la solución del sistema. Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Ya con esta información de fondo, se pueden justificar los patrones de diseño que se utilizarán, los cuales se enuncian a continuación:

- MVC (Modelo-Vista-Controlador): Divide un componente o un subsistema en tres partes lógicas: modelo, vista y controlador, facilitando la modificación o personalización de cada parte. El modelo se encarga de la interacción directa con los datos que se encuentran en una Base de Datos, la vista se encarga de aceptar los datos de entrada del usuario, así como

mostrarle los datos de salida; y finalmente, el controlador, el cual se encarga de organizar todas las peticiones del usuario al sistema y viceversa. En la figura 3.1 se muestra un diagrama de este patrón de diseño:

- 

e

  patrón se encarga de comunicar diferentes capas en el desarrollo de aplicaciones empresariales, y más específicamente en el sistema, permitirá conectar la capa Web con la capa empresarial.
- Facade: Provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema. En el caso del FONOL, permitirá aislar las operaciones complicadas, y abstraerlas en una interfaz más sencilla.

Igualmente, se enunciarán los frameworks que se utilizarán para el desarrollo del FONOL, pero antes, se enunciará framework, ya que se necesita tener una idea clara de lo que significa. Un framework es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros softwares para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Ya con esta información, es necesario mencionar los frameworks que se utilizarán en el desarrollo del sistema:

- Struts es una herramienta de soporte para el desarrollo de aplicaciones Web bajo el patrón MVC en la plataforma J2EE (Java 2, Enterprise Edition). Struts se desarrollaba como parte del proyecto Jakarta de la Apache Software Foundation, pero actualmente es un proyecto independiente conocido como Apache Struts. Struts permite reducir el tiempo de desarrollo. Su carácter de "software libre" y su compatibilidad con todas las plataformas en que Java Enterprise esté disponible, lo convierte en una herramienta altamente disponible.
- Java Persistence API, más conocida por su sigla JPA, es la API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, como pasaba con EJB2, y permitir usar objetos regulares (conocidos como POJOs).

Finalmente, en lo que respecta a Java, se apoyará en las tecnologías de vanguardia de Sun, para llevar un control más preciso del desarrollo del sistema, lo cual, se sustentará en los estándares Sun que incluyen la capa Web y la capa de Negocio.

En la capa Web, es necesario mencionar dos tipos de componentes que permitirán construir la capa web: los JSP's y los Servlets. Los JSP, son archivos HTML, en los cuales está embebido código Java, es muy útil para mostrar información del usuario y para aceptar datos del mismo. Por otro lado, los Servlets son un componente escrito puramente en Java, que permite manejar el flujo de la información, y en el caso del FONOL, fungirá como controlador de la información de entrada y de salida.

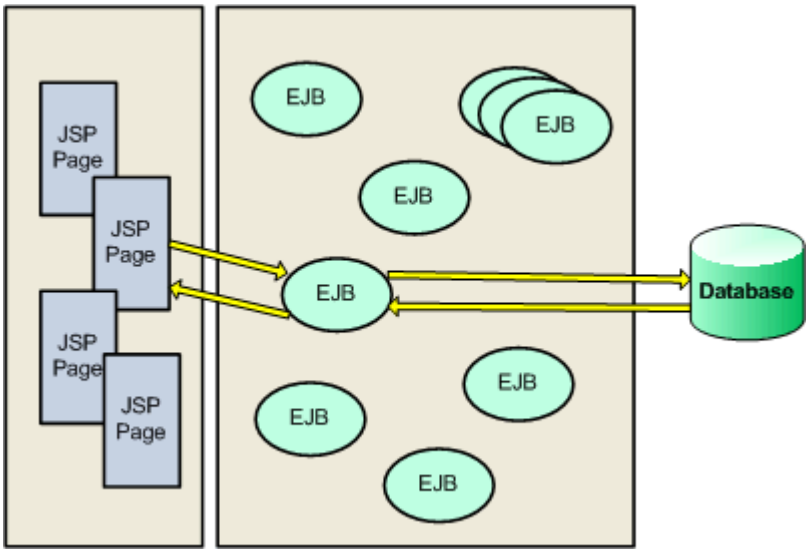
En la capa de Negocio, se deben mencionar dos tipos de componentes que permitirán realizar funciones específicas con los datos: los EJB y las entidades. Los EJB, son componentes de negocio portables, los cuales realizan una función

en especifico pero con el sustento de servicios que por sí ya tiene soportados, como: manejo de transacciones, comunicación remota utilizando CORBA, control de la concurrencia, y seguridad entre otros; todo esto permitirá abstraer la lógica que manipula la base de datos, de una manera limpia y escalable. Finalmente, como último componente, se mencionarán las entidades, que como su nombre lo indica, representan una entidad o una clase, dentro de la base de datos. Este componente es el más importante, ya que permitirá manejar y modificar los datos haciendo uso del mapeo objeto relacional, en el cual, solo se mapearán las relaciones de la base de datos en clases, para su fácil uso y manipulación.

La figura 3.2, da un ejemplo de cómo se conjunta la capa Web y de Negocio:

Figura 3.2

Relación de las capas Web y de Negocio, resaltando que la Vista (capa Web) no se comunica directamente con la Base de Datos.



3.1. Metodología a utilizar

De las metodologías mencionadas y evaluadas en el capítulo anterior, se eligió RUP, ya que entre sus características más importantes, destacan:

- Dirigido por casos de uso

  Los casos de usos se especifican, se diseñan, y los casos de uso finales son la fuente a partir de la cual los ingenieros de prueba construyen sus casos de prueba.

- Centrado en la arquitectura:

  Es muy importante esta característica, ya que nos da de una manera más ordenada la forma en que se van ensamblando los componentes de la aplicación, así como sus interrelaciones. La arquitectura del software involucra:

  - La estructura y el comportamiento
  - La funcionalidad
  - La facilidad de comprensión
  - La reutilización
  - La flexibilidad
  - El rendimiento
  - Las restricciones y compromisos económicos y tecnológicos
  - La estética

- Iterativo e incremental

  Es la propiedad más interesante que me hizo_elegir RUP, ya que nos permite iterar cuantas veces se necesite, para pulir nuestra aplicación de manera más limpia, sin afectar todo el conjunto.

3.2. Base de Datos a utilizar

La elección del manejador a utilizar, se evaluó ampliamente, ya que se contaba en el mercado, con manejadores robustos, de licencia muy cara y de difícil mantenimiento. Pero finalmente se utilizó PostgreSQL, las razones fueron las siguientes:

- Opensource: Esta cualidad es muy importante, ya que no se requiere de una licencia cara, como en el caso de SQL Server.
- Robusto: A pesar de que es un software libre, reúne todas las características de un Sistema Gestor de Base de Datos.
- Fácil: Es de muy fácil uso, y no se necesita de un DBA para operarlo. Además, una de las razones más influyentes fue, que es muy portable, ya que con un script se puede levantar todo el sistema.
- Modelo Objeto-Relacional simplificado: Es una característica importante, ya que nos permite de una manera sencilla generar el mapeo de los Objetos con el modelo Relacional, sin problemas de compatibilidad.

3.3. Lenguaje de programación a utilizar

De los lenguajes evaluados en el capítulo anterior, se eligió Java, por varias razones:

- Escalabilidad: Java me permite separar una aplicación Java en varias capas, y así tener una organización más limpia del código, es decir, se puede modificar una capa muy interna del sistema sin afectar a las capas exteriores.
- Independencia de plataforma: se eligió esta característica, ya que es sencillo desarrollar tanto el Linux como en Windows sin afectar la aplicación ni ninguno de sus componentes.
- Robusto: es un lenguaje robusto, ya que mediante sus modificadores de acceso, nos permite restringir estrictamente el acceso a las variables y a los métodos.

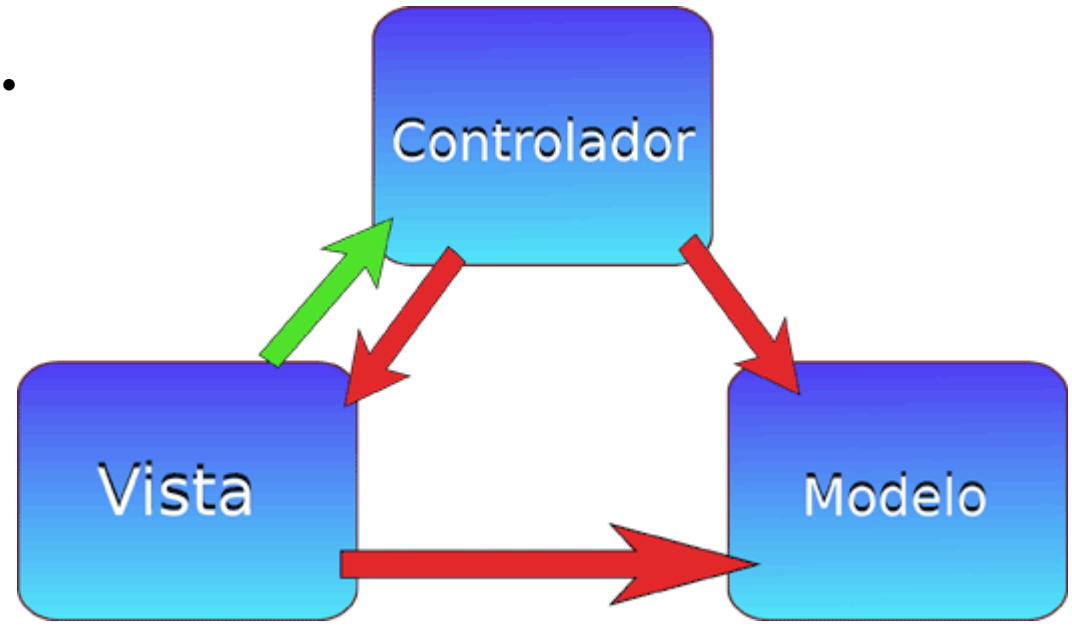
Por otra parte, ya elegido Java como lenguaje de programación, es necesario hacer algunas anotaciones referentes a las tecnologías dentro de Java que se utilizarán para el desarrollo del sistema.

En primera instancia, es necesario definir los Patrones de Diseño, ya que se apoyará de ellos para la solución del sistema. Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Ya con esta información de fondo, se pueden justificar los patrones de diseño que se utilizarán, los cuales se enuncian a continuación:

- MVC (Modelo-Vista-Controlador): Divide un componente o un subsistema en tres partes lógicas: modelo, vista y controlador, facilitando la modificación o personalización de cada parte. El modelo se encarga de la interacción directa con los datos que se encuentran en una Base de Datos, la vista se encarga de aceptar los datos de entrada del usuario, así como

mostrarle los datos de salida; y finalmente, el controlador, el cual se encarga de organizar todas las peticiones del usuario al sistema y viceversa. En la figura 3.1 se muestra un diagrama de este patrón de diseño:

- 

e

  patrón se encarga de comunicar diferentes capas en el desarrollo de aplicaciones empresariales, y más específicamente en el sistema, permitirá conectar la capa Web con la capa empresarial.
- Facade: Provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema. En el caso del FONOL, permitirá aislar las operaciones complicadas, y abstraerlas en una interfaz más sencilla.

Igualmente, se enunciarán los frameworks que se utilizarán para el desarrollo del FONOL, pero antes, se enunciará framework, ya que se necesita tener una idea clara de lo que significa. Un framework es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros softwares para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Ya con esta información, es necesario mencionar los frameworks que se utilizarán en el desarrollo del sistema:

- Struts es una herramienta de soporte para el desarrollo de aplicaciones Web bajo el patrón MVC en la plataforma J2EE (Java 2, Enterprise Edition). Struts se desarrollaba como parte del proyecto Jakarta de la Apache Software Foundation, pero actualmente es un proyecto independiente conocido como Apache Struts. Struts permite reducir el tiempo de desarrollo. Su carácter de "software libre" y su compatibilidad con todas las plataformas en que Java Enterprise esté disponible, lo convierte en una herramienta altamente disponible.
- Java Persistence API, más conocida por su sigla JPA, es la API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, como pasaba con EJB2, y permitir usar objetos regulares (conocidos como POJOs).

Finalmente, en lo que respecta a Java, se apoyará en las tecnologías de vanguardia de Sun, para llevar un control más preciso del desarrollo del sistema, lo cual, se sustentará en los estándares Sun que incluyen la capa Web y la capa de Negocio.

En la capa Web, es necesario mencionar dos tipos de componentes que permitirán construir la capa web: los JSP's y los Servlets. Los JSP, son archivos HTML, en los cuales está embebido código Java, es muy útil para mostrar información del usuario y para aceptar datos del mismo. Por otro lado, los Servlets son un componente escrito puramente en Java, que permite manejar el flujo de la información, y en el caso del FONOL, fungirá como controlador de la información de entrada y de salida.

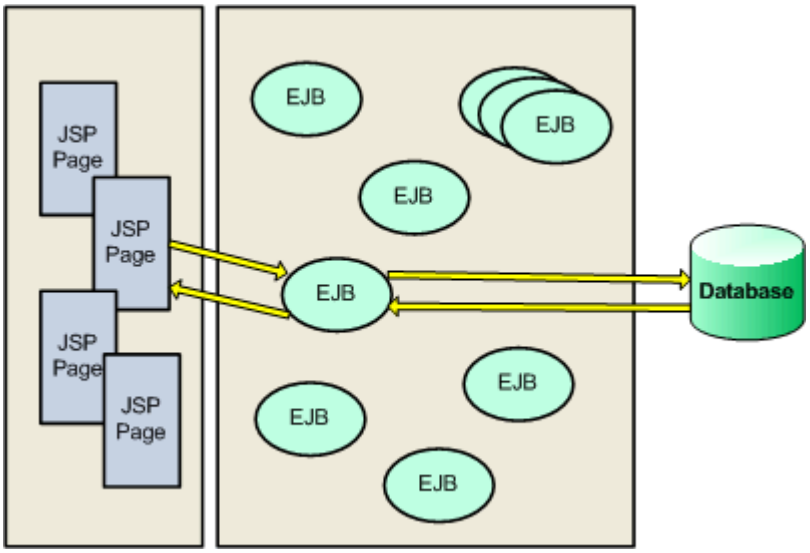
En la capa de Negocio, se deben mencionar dos tipos de componentes que permitirán realizar funciones específicas con los datos: los EJB y las entidades. Los EJB, son componentes de negocio portables, los cuales realizan una función

en especifico pero con el sustento de servicios que por sí ya tiene soportados, como: manejo de transacciones, comunicación remota utilizando CORBA, control de la concurrencia, y seguridad entre otros; todo esto permitirá abstraer la lógica que manipula la base de datos, de una manera limpia y escalable. Finalmente, como último componente, se mencionarán las entidades, que como su nombre lo indica, representan una entidad o una clase, dentro de la base de datos. Este componente es el más importante, ya que permitirá manejar y modificar los datos haciendo uso del mapeo objeto relacional, en el cual, solo se mapearán las relaciones de la base de datos en clases, para su fácil uso y manipulación.

La figura 3.2, da un ejemplo de cómo se conjunta la capa Web y de Negocio:

Figura 3.2

Relación de las capas Web y de Negocio, resaltando que la Vista (capa Web) no se comunica directamente con la Base de Datos.



Capítulo 4

4. Diseño del Sistema aplicado al FONOL orientado a RUP

Antes de comenzar el desarrollo del sistema, es preciso mencionar, que RUP es una metodología de vanguardia, ya que permite de una manera simple el proceso de la construcción de un sistema, en este capítulo se ubicará la metodología con sus diversas fases.

4.1. Análisis de Requisitos

En esta sección se dará un análisis de los requisitos, para cada uno de ellos mostraré los actores, las responsabilidades y necesidades de cada uno de ellos.

4.1.1. Reconocimiento de los actores

Antes de comenzar, se enumerarán los actores principales del sistema, para tener un entendimiento más claro del sistema:

- Organización: Es una organización no l ucrativa, la cual funge como dos actores más, dependiendo del resultado de su solicitud. Si solicitó ser beneficiada es un Candidato; y si es aceptada por el sistema, es un Beneficiario.
- Candidato: Es la organización que quiere ser aceptada por el FONOL, la cual ingresa sus datos de solicitud en el sistema para esperar ser aceptada.
- Beneficiario: Es aquel candidato que es aceptado por el delegado para ser parte del FONOL. Ya puede ingresar proyectos, para que de esta manera, se le otorguen fondos.
- Proyecto: Es ingresado por un Beneficiario, y describe una necesidad en específico por la cual requiere fondos la organización a la que representa.
- Estado: Es cada estado del país.
- Gobierno: Es el encargado de administrar el presupuesto de cada Estado. Para tomar una decisión de agregar o disminuir el presupuesto en un estado, el se apoya de la información del sistema para evaluar si un estado tiene más de lo que necesita, o está desperdiciando el presupuesto otorgado para brindárselo a otro estado que lo necesite.
- Delegado: Es la persona responsable de cada uno de los estados. Es la persona que autoriza o rechaza a cada candidato; y es también la persona que acepta o rechaza los proyectos de cada beneficiario basándose en el presupuesto solicitado y el presupuesto disponible.

4.1.2. Detalle de los Requerimientos del Sistema

En esta sección se dará un panorama de los requerimientos del cliente, a partir de tres tipos de requerimientos diferentes: Requisitos funcionales, Requisitos no funcionales, Requisitos de implementación.

4.1.2.1. Requerimientos funcionales

Los Requerimientos funcionales son:

- Describen las interacciones entre el sistema y su entorno (usuarios u otros sistemas), sin tener en cuenta cuestiones de implementación.
- Se estudian y representan en el Modelo de Casos de Uso

Los requerimientos funcionales del sistema son:

- Brindar de un formulario para que los candidatos puedan ingresar las solicitudes.
- Brindar de una pantalla en la cual se muestren los resultados de los beneficiarios aceptados.
- Construir dos áreas del sistema: una pública y una protegida.
- El área pública debe permitir acceso a cualquier tipo de usuario de propósito general, para ingresar solicitudes, revisar resultados y contactar a la empresa por información.
- El área protegida debe distinguir entre diferentes roles, y servirá para administrar y validar toda la información sensible del FONOL.
- Brindar tres roles de acceso al sistema protegido de administración: beneficiario, gobierno, delegado.
- Para todos los roles se necesita una clave de acceso al sistema, privilegios diferentes y una opción de salirse del sistema para eliminar a sesión.

  Las características del rol delegado se explicarán en los siguientes puntos:

- Un delegado después de ingresar al sistema, tendrá dos Áreas principales: Evaluación de candidatos y Consultar beneficiarios.
- El área de Evaluación de candidatos, para el delegado debe mostrar una lista de los candidatos de su estado que han puesto su solicitud en el sistema. El sistema sólo debe de mostrar las solicitudes que fueron ingresadas con un mes de anterioridad. También, en esta área se deben visualizar los Candidatos que han sido aceptados con un mes de anterioridad, también de su mismo estado, como una ayuda visual para el delegado.
- En la lista de Candidatos que buscan ser aceptados, se debe brindar una opción para consultar los detalles de un Candidato en específico.
- El los detalles del Candidato, sólo el delegado puede cambiar el estatus de cualquier candidato de su estado, y en caso de aceptarlo, e proporcionará un usuario y contraseña disponible para su acceso.
- Si la organización es aceptada, entonces se convertirá en Beneficiado, y podrá acceder al sistema con la contraseña que le proporcionó delegado.
- Si la organización no es aceptada, entonces podrá hacer otra petición para el mes siguiente, teniendo en cuenta que solo podrá hacer 3 intentos para ser beneficiado.
- En el área Consultar Beneficiarios, el delegado puede consultar un listado de todos los Beneficiarios, de su estado, y también cada registro tendrá dos vínculos; uno, para consultar los detalles del beneficiario y otro para consultar los proyectos del beneficiario.
- En los detalles del beneficiario, se mostrarán los datos del beneficiario, y también existirá un formulario para cambiar la contraseña del beneficiario, si se requiere.
- En el vínculo de consultar proyectos para cada beneficiario, se mostrará un listado de todos los proyectos que ha ingresado el beneficiario, así como, un vínculo para consultar los detalles.
- En los detalles del proyecto, se deben mostrar los datos del proyectos, y se debe permitir

En lo que respecta al perfil del beneficiario, se tiene otro rol más restrictivo, el cual se describirá en los puntos siguientes:

- El beneficiario necesita un área en la cual pueda manejar todo el proceso de solicitud de proyectos. Su área es Consultar Proyectos.
- En el área de consultar proyectos, se debe mostrar una lista de todos los proyectos ingresados por el beneficiario, así como, un vínculo para agregar un proyecto.
- Los proyectos tienen 4 estatus: Solicitado, Autorizado, Viable o Rechazado.
- Se debe de crear un formulario para que el beneficiario pueda ingresar un proyecto, y debe especificar el presupuesto requerido y la justificación del proyecto.
- El delegado es la única persona que puede cambiar el estatus de un proyecto. El beneficiario solo consultará el estatus sin poder modificarlo.
- En la misma pantalla de Consultar Proyectos, por cada proyecto que aparezca, existirá un vínculo para consultar y modificar los detalles cada proyecto.
- En los detalles del proyecto, el beneficiario podrá cambiar los datos del proyecto por dos grandes razones: puede haberse equivocado al capturar información y puede reconsiderar el presupuesto, en caso de que el proyecto no haya sido autorizado por esa razón.

Finalmente, para el gobierno, dentro del sistema, se requiere:

- Se requiere una pantalla, en la cual, el Gobierno pueda consultar y evaluar a los delegados. En esta pantalla se enlistarán los delegados existentes por cada estado. De igual manera, se requiere que en esta pantalla exista un vínculo para agregar un delegado, de un estado que no tenga datos ni delegado.
- El desde su sesión puede revisar cualquier sección, desde consultar beneficiarios, proyectos, candidatos, y sus respectivos detalles; con el inconveniente de que no puede modificar los datos de estas secciones.

4.1.2.2. Requerimientos no funcionales

Aquí se describirán los requerimientos, que son parte del diseño, y los criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, es decir, son las cualidades del sistema que son requeridas, muy aparte de la funcionalidad primordial del sistema.

Los requerimientos no funcionales son:

- Todos los formularios deben de ser validados, para evitar el error del usuario que capture la información.
- Deben de validarse con transacciones las operaciones que involucren fondos, es decir, si el Gobierno le otorga un monto a un delegado, solo puedo hacerlo, si cuenta con el presupuesto disponible para otorgarlo; y una vez asignado, se debe incrementar el presupuesto en el estado del delegado, y disminuir el presupuesto del gobierno. Lo mismo aplica para el dinero que un delegado otorga a un beneficiario.

4.2. Diseño

En la etapa de diseño, el objetivo es producir un modelo lógico del sistema a implementar, y se tiene que tomar en cuenta lo siguiente:

- Los requisitos no funcionales, restricciones impuestas por el lenguaje de programación a usar, el sistema operativo donde se va a ejecutar, el tipo de interfaz, etc.
- Capturar requisitos de las clases de análisis para tomarse con punto de partida para la implementación.
- Ser capaz de visualizar y razonar acerca del diseño usando una notación común.

Para apoyarse en esta fase se valdrá de artefactos para llevar un diseño más puntual de la aplicación, los cuales se enlistarán a continuación:

- Clase de Diseño:

  Son abstracciones de clases directamente utilizables en la implementación y son representadas principalmente mediante Diagramas de Clase. El lenguaje utilizado para especificar las clases es UML en el cual se muestra la visibilidad de atributos y operaciones de la clase.
- Realización de Casos de Uso-Diseño: n

  Describe cómo un caso de uso se lleva a cabo en términos de clases de diseño y sus objetos. Hay una correspondencia directa entre la Realización de un Caso de Uso de Diseño y la Realización de un Caso de Uso de Análisis. Estos los diagramas que nos permitirán esquematizar el sistema:

  A. Diagrama de Clases.- que contiene las clases que participan en el caso de uso, aunque algunas de ellas puedan participar en varios.
  B. Diagrama de Secuencia.- que muestra una secuencia detallada de interacción entre los objetos de diseño. En algunos casos es posible incluir Subsistemas dentro de los diagramas. Los diagramas de secuencia visualizan el intercambio de mensajes entre objetos.

C. Descripción del Flujo de Eventos de Diseño.- descripción textual que explica y completa a los diagramas de colaboración

Ya explicada la metodología de esta fase de RUP, se mostrarán mediante los Casos de Uso las necesidades del FONOL.

4.2.1. Descripción de Casos de Uso

Un caso de uso es un documento narrativo que describe la secuencia de eventos de un actor que utiliza el sistema para completar el proceso.

Los actores y las acciones a realizar, nos acercan el contexto del sistema y nos dan una base sólida para la descripción de los Casos de Uso.

Dadas las necesidades del FONOL, se dividirá el sistema en dos partes: la parte pública y la parte protegida. La parte pública, será accedida sin requerir una contraseña. La parte protegida, al manejar información sensible, estará protegida por contraseña para que no esté accesible al público general.

En la siguiente sección, se describirán los Casos de Uso que existirán en el Sistema Público.

4.2.2. Descripción de Casos de Uso del Sistema Público

En esta sección se explicarán los Casos de uso del Sistema Público:

4.2.2.1. Visitar el Sitio Público

Actores:	Cualquier Usuario
Casos de Uso que incluye:	Ninguno
Propósito:	En este caso de uso, el usuario Web accede al sistema por Internet
Resumen:	El usuario accede al sistema por internet, al teclear la dirección del Sistema FONOL.
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1. El usuario accede al portal Web, mediante la URL del FONOL.	2. El servidor le regresa la página principal del Sistema.
Curso Alterno de Eventos:	
Línea 1:	El usuario se equivoca al teclear la dirección del FONOL

4.2.2.2. Revisar Quienes Somos

Actores:	Cualquier usuario
Casos de Uso que incluye:	Ninguno
Propósito:	El Usuario accede a la sección Quienes

	somos, para conocer a la organización FONOL
Resumen:	En este caso se revisan los detalles del FONOL, para conocer su historia
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
El usuario accede a la sección Quienes somos	El sistema presenta la información de FONOL
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.2.3. Contactar FONOL

Actores:	Cualquier usuario
Casos de Uso que incluye:	Ninguno
Propósito:	Se le da la opción al usuario de contactar al FONOL
Resumen:	Se le da la opción al usuario de contactar al FONOL por medio información de contacto, (teléfono, dirección de correo electrónico)
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
El usuario accede a la sección Contactar al FONOL	El sistema proporciona la información necesaria para que la contacten
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.2.4. Revisar Solicitudes

Actores:	Cualquier usuario
Casos de Uso que incluye:	Ninguno
Propósito:	Revisar las solicitudes de las organizaciones que quieren unirse al FONOL.
Resumen:	En este caso de uso, el usuario revisa las organizaciones que han ingresado solicitud y están a la espera de ser aceptados.
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
El usuario accede a la sección Revisar Solicitudes	El sistema le proporciona un listado de todas las organizaciones que han ingresado alguna solicitud para ingresar al FONOL
Curso Alterno de Eventos:	
Línea 1:	La lista no trae datos, ya que para ese periodo no hay solicitudes pendientes.

4.2.2.5. Ingresar Solicitud

Actores:	Candidato
Casos de Uso que incluye:	Procesar información de Ingreso
Propósito:	Se utiliza para ingresar una solicitud al FONOL
Resumen:	En este caso de uso, el Candidato que quiere ser Beneficiado por el FONOL da clic en el botón Solicitudes para acceder al formulario

	de la solicitud
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El candidato accede a la sección Solicitudes al dar clic en el botón Solicitudes. 3.-Deberá ingresar la información de su organización y del representante de la misma con datos correctos que serán validados para verificar su consistencia. La información requerida por el formulario es: longitud de texto, email valido, RFC valido, y además, que el RFC de la organización no esté en otra organización, para evitar fraudes.	2.-Provocará que se abra un formulario para ingresar sus datos. 4.-La información es procesada por el servidor.
Curso Alterno de Eventos:	
Línea 1:	El candidato ingresa información incorrecta. En este escenario, el candidato captura información incorrecta, lo cual provoca que el sistema le indique al usuario el error que cometió para que lo corrija.

4.2.2.6. Procesar información de Ingreso

Actores:	Sistema
Casos de Uso que incluye:	Ninguno
Propósito:	Ingresar información de ingreso a la base de datos
Resumen:	En este caso de uso, el sistema internamente, procesa la información del candidato y le

	aplica las validaciones pertinentes
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
	1.-El sistema valida la información
Curso Alterno de Eventos:	
Línea 1:	Hay un error en la validación, lo cual invalida el envío de información

4.2.2.7. Consultar Detalles del Candidato

Actores:	Candidatos, Usuarios Normales
Casos de Uso que incluye:	Ninguno
Propósito:	El usuario puede consultar los detalles de un Candidato
Resumen:	El usuario puede consultar los detalles de un Candidato
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
El usuario verifica los detalles del Candidato	
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.2.8. Revisar Candidatos Aprobados

Actores:	Candidatos, Usuarios Normales
Casos de Uso que incluye:	Ninguno
Propósito:	Se revisan los candidatos aprobados
Resumen:	Se revisan los candidatos aprobados
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
Se revisan los candidatos aprobados	
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.3. Descripción de Casos de Uso del Sistema Protegido

En esta sección se explicarán los Casos de uso del Sistema Protegido:

4.2.3.1. Ingresar al Sistema Protegido

Actores:	Usuario FONOL
Casos de Uso que incluye:	Validar Usuario
Propósito:	El usuario ingresa al sistema FONOL
Resumen:	El usuario ingresa al sistema FONOL
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
El usuario ingresa al sistema FONOL	
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.3.2. Validar Usuario

Actores:	Sistema
Casos de Uso que incluye:	Ninguno
Propósito:	Validar o no al usuario
Resumen:	El sistema, mediante un modulo de seguridad, permite validar o no al usuario, mostrando un mensaje descriptivo en caso de que no sea posible

Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
	1.-EL sistema valida al usuario con la base de datos
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.3.3. Salir del Sistema

Actores:	Usuario FONOL
Casos de Uso que incluye:	Ninguno
Propósito:	Este caso de uso permite salir del sistema
Resumen:	El usuario que se encuentra adentro del sistema protegido FONOL, mediante un botón puede salir del sistema eliminando su sesión.
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
El usuario selecciona un botón para salir del sistema	El sistema elimina la sesión
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.3.4. Consultar Candidatos

Actores:	Delegado
Casos de Uso que incluye:	Ninguno
Propósito:	Consultar los candidatos que quieren ser beneficiados
Resumen:	En este caso de uso, el delegado consulta una lista de los candidatos que quieren ser beneficiados en su estado
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El usuario ingresa al sistema a la sección de Consultar Candidatos	2.-El sistema muestra una lista de todos los candidatos, junto con su estatus y un vínculo para evaluarlos.
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.3.5. Evaluar Candidato

Actores:	Delegado
Casos de Uso que incluye:	Validar información de Candidato
Propósito:	Evaluar al candidato que quiere ser beneficiado
Resumen:	En este caso de uso, el delegado consulta los detalles del candidato, para evaluar su condición, y de esta manera, aceptarlo o no al FONOL
Curso Normal de Eventos:	

Acción del Usuario	Acción del Sistema
1.-El usuario ingresa al sistema a la sección de Consultar Candidatos	2.-El sistema muestra una lista de todos los candidatos, junto con su estatus y un vínculo para evaluarlos.
3.-El usuario selecciona el vínculo de un candidato	4.-El sistema brinda de un formulario para actualizar la información del candidato, así como, para evaluarlo.
5.-El usuario actualiza la información si lo requiere, y envía el formulario al servidor.	6.-El servidor procesa la información y muestra errores en caso de haberlos
Curso Alterno de Eventos:	
Línea 1:	La información no es válida, lo cual muestra un error descriptivo al delegado.

4.2.3.6. Validar información de Candidato

Actores:	Sistema
Casos de Uso que incluye:	Ninguno
Propósito:	Se valida la información ingresada de un candidato
Resumen:	En este caso de uso, el sistema valida la información ingresada por el delegado del candidato
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
	1.-El sistema mediante una serie de validaciones de dato, permite o no, ingresar la información a la base de datos.

Curso Alterno de Eventos:	
Línea 1:	La información no es válida, lo cual muestra un error descriptivo al delegado.

4.2.3.7. Consultar Delegados

Actores:	Gobierno
Casos de Uso que incluye:	Ninguno
Propósito:	Consultar los delegados que existen
Resumen:	El Gobierno que acaba de entrar en el sistema, puede observar la lista de todos los delegados
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El Gobierno ingresa al sistema	2.-El sistema muestra una lista de todos los delegados, para consultarlos o modificarlos.
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.3.8. Agregar Delegado

Actores:	Gobierno
Casos de Uso que incluye:	Valida presupuesto, Verifica Usuario
Propósito:	Este caso de uso permite agregar un delegado de un estado que no tenga uno

Resumen:	El gobierno llena un formulario para ingresar la información de un delegado de un estado que no tenga uno
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El gobierno ingresa la información del delegado	2.-El sistema verifica que la información sea correcta 3.-El sistema ingresa la información a la base de datos.
Curso Alterno de Eventos:	
Línea 1:	El gobierno ingresa información incorrecta lo cual no permite que el flujo continúe
Línea 2:	El gobierno trata de ingresar un usuario que ya existe, lo cual no permite que el flujo continúe

4.2.3.9. Valida presupuesto

Actores:	Sistema
Casos de Uso que incluye:	Ninguno
Propósito:	Verifica que haya presupuesto disponible para asignarse
Resumen:	El sistema verifica que se cuente con el presupuesto disponible y no asignado, para asignárselo a alguien más
Curso Normal de Eventos:	

Acción del Usuario	Acción del Sistema
	1.-El sistema consulta el presupuesto total con el que cuenta, posteriormente consulta la suma de todos los presupuestos asignados, y verifica que el presupuesto pedido no exceda el presupuesto disponible
Curso Alterno de Eventos:	
Línea 1:	El presupuesto no es suficiente lo cual provoca que se evite asignar presupuesto no válido, así como, enviarle un mensaje descriptivo al usuario.

4.2.3.10. Verifica Usuario

Actores:	Sistema
Casos de Uso que incluye:	Ninguno
Propósito:	Se valida si el usuario ya existe
Resumen:	El sistema hace una consulta con la base de datos, en la tabla usuarios,
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El gobierno ingresa la información del delegado	2.-El sistema verifica que la información sea correcta 3.-El sistema ingresa la información a la base de datos.
Curso Alterno de Eventos:	

Línea 1:	El gobierno ingresa información incorrecta lo cual no permite que el flujo continúe
Línea 2:	El gobierno trata de ingresar un usuario que ya existe, lo cual no permite que el flujo continúe

4.2.3.11. Modificar Delegado

Actores:	Gobierno
Casos de Uso que incluye:	Valida presupuesto
Propósito:	Modificar un delegado existente
Resumen:	Le permite al gobierno modificar un delegado existente, ya sea su presupuesto o sus datos generales
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El usuario modifica los datos o el presupuesto del delegado	2.-El sistema permite recibir la información del delegado. 3.-Si la información es correcta almacena los datos en la base de datos
Curso Alterno de Eventos:	
Línea 1:	El gobierno ingresa información incorrecta lo cual no permite que el flujo continúe
Línea 2:	El gobierno trata de ingresar un usuario que ya existe, lo cual no permite que el flujo continúe

4.2.3.12. Cambiar Contraseña del Delegado

Actores:	Gobierno
Casos de Uso que incluye:	Ninguno
Propósito:	En este caso de uso, el gobierno modifica el contraseña de un delegado
Resumen:	El gobierno modificar la contraseña del delegad, ya sea por seguridad o por cambios de puesto
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El usuario modifica la contraseña de acceso del delegado	2.-El sistema permite modificar la contraseña del delegado y actualizarla en la base de datos
Curso Alterno de Eventos:	
Línea 1:	El gobierno ingresa una contraseña de una longitud incorrecta, lo cual no permite que prosiga el flujo normal del sistema

4.2.3.13. Consultar Beneficiarios

Actores:	Gobierno, Delegados
Casos de Uso que incluye:	Ninguno
Propósito:	En este caso de uso, se consultan los beneficiarios asociados a un Estado.
Resumen:	Se consultan los beneficiarios asociados a un estado, y por consiguiente, están asociados a

	un delegado. Se mostrará una lista de beneficiarios con vínculos para modificar u observar sus atributos.
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El actor, entra con un delegado de interés, a la sección Consultar Beneficiarios	2.-El sistema le muestra una lista de los beneficiarios con vínculos para observar o modificar los atributos del mismo.
Curso Alterno de Eventos:	
Línea 1:	El delegado puede observar el presupuesto disponible.

4.2.3.14. Modificar Beneficiarios

Actores:	Delegados
Casos de Uso que incluye:	Ninguno
Propósito:	En este caso de uso, se modifican los atributos del beneficiario.
Resumen:	Se consultan los beneficiarios asociados a un estado, y por consiguiente, están asociados a un delegado. Se mostrará una lista de beneficiarios con vínculos para modificar u observar sus atributos. Al seleccionarse un beneficiarios se accede a un formulario, el cual, permite la modificación del beneficiario
Curso Normal de Eventos:	

Acción del Usuario	Acción del Sistema
1.-El actor, entra con un delegado de interés, a la sección Consultar Beneficiarios	2.-El sistema le muestra una lista de los beneficiarios con vínculos para observar o modificar los atributos del mismo.
3.-El usuario selecciona un vínculo para observar los detalles del beneficiario	4.-El sistema muestra un formulario con los datos del beneficiario
5.-El usuario modifica los atributos y envía los datos	6.-El sistema procesa la información y la valida antes de actualizarla en la base de datos
Curso Alterno de Eventos:	
Línea 1:	El Usuario ingresa información errónea, lo cual, muestra un error descriptivo al usuario

4.2.3.15. Consultar Detalles del Beneficiario

Actores:	Delegado, Gobierno
Casos de Uso que incluye:	Validar estatus
Propósito:	Se consultan los detalles del beneficiario
Resumen:	Se muestra un formulario para observar los atributos del beneficiario elegido, así como para modificarlos.
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-Se accede a la sección Consultar Beneficiarios	2.-El sistema muestra un listado de los beneficiarios existentes con vínculos para observar detalles
3.-El actor selecciona uno de los vínculos de	4.-El sistema muestra un formulario con la

un beneficiarios de interés	información del beneficiario
Curso Alterno de Eventos:	
Línea 1:	Ninguna

4.2.3.16. Validar estatus

Actores:	Sistema
Casos de Uso que incluye:	Validar estatus
Propósito:	Valida el estatus del candidato
Resumen:	Se valida el estatus del candidato. Tomando en cuanta la disponibilidad de fondos y el criterio del delegado
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
	1.-Se valida el estatus del candidato
Curso Alterno de Eventos:	
Línea 1:	Ninguna

4.2.3.17. Ingresar Proyecto

Actores:	Beneficiario
Casos de Uso que incluye:	Valida Presupuesto
Propósito:	Se ingresa un proyecto
Resumen:	En este caso de uso, el beneficiario ingresa un proyecto para beneficiar a una causa de la

	organización.
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El usuario accede con un beneficiario al sistema	2.-El sistema muestra un vínculo para ir al formulario de ingresar un proyecto
3.-El usuario ingresa los datos del proyecto y los envía al servidor	4.-El sistema procesa la información
Curso Alterno de Eventos:	
Línea 1:	El beneficiario ingresa información incorrecta lo cual no permite que el flujo continúe
Línea 2:	El beneficiario trata de asignar un presupuesto mayor al disponible, lo cual no permite que el flujo continúe

4.2.3.18. Consultar Proyectos

Actores:	Gobierno, Delegado, Beneficiario
Casos de Uso que incluye:	Ninguno
Propósito:	Se utiliza para observar los proyectos asociados a un beneficiario
Resumen:	En este caso de uso, se da un listado de todos los proyectos asociados a un beneficiario, su presupuesto, y vínculos para consultar sus detalles, modificarlos o evaluarlos.
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**

1.-El usuario accede con un beneficiario al sistema	2.-El sistema muestra un listado de los proyectos asociados al beneficiario
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.3.19. Consultar Detalles del Proyecto

Actores:	Gobierno, Delegado, Beneficiario
Casos de Uso que incluye:	Ninguno
Propósito:	Se consultan los detalles del proyecto
Resumen:	En este caso de uso, se muestran los detalles de un proyecto en especifico y también de una interfaz para modificarlo
Curso Normal de Eventos:	
Acción del Usuario	**Acción del Sistema**
1.-El actor accede con un beneficiario	2.-El sistema le brinda de un listado de los proyectos asociados al beneficiario
3.-El usuario observa los detalles del proyecto	
Curso Alterno de Eventos:	
Línea 1:	Ninguno

4.2.4. Diagramas de Casos de Uso

En esta sección se elaborarán los diagramas de casos de uso que mostrarán los escenarios del sistema. Para esto, el sistema se dividirá en paquetes que mostrarán la organización del FONOL, todo esto se explica en la figura 4.1, que se muestra a continuación:

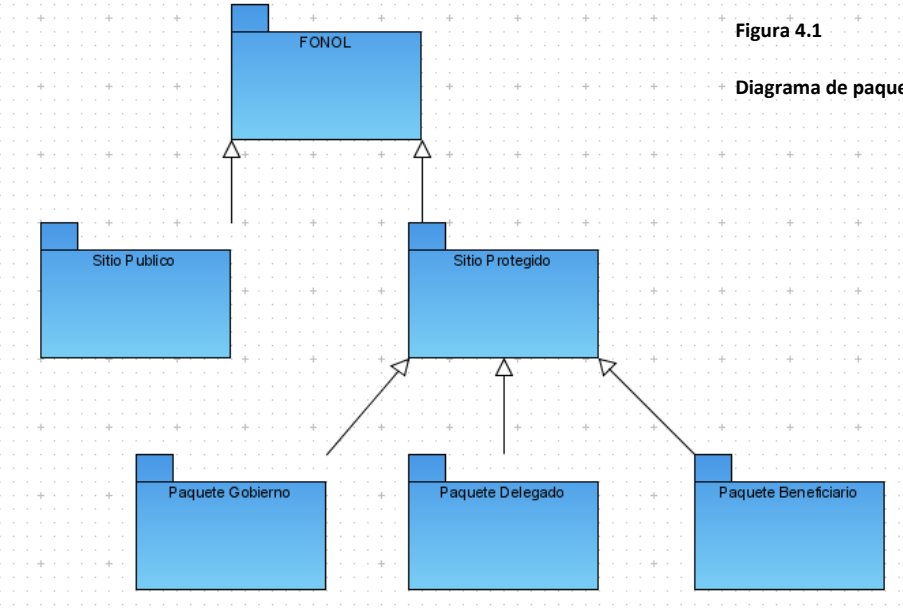


Figura 4.1

Diagrama de paquetes del Sistema.

En la figura 4.1 se muestra el diagrama de paquetes del sistema, el cual, se divide en sitio público y sitio protegido. El sitio protegido de dividirá en tres paquetes que corresponderán a los tres roles del sistema: gobierno, delegado y beneficiario.

En las secciones siguientes, se dará una explicación de cada uno de estos paquetes:

El paquete sitio público es mostrado en la figura 4.2, con sus casos de uso:

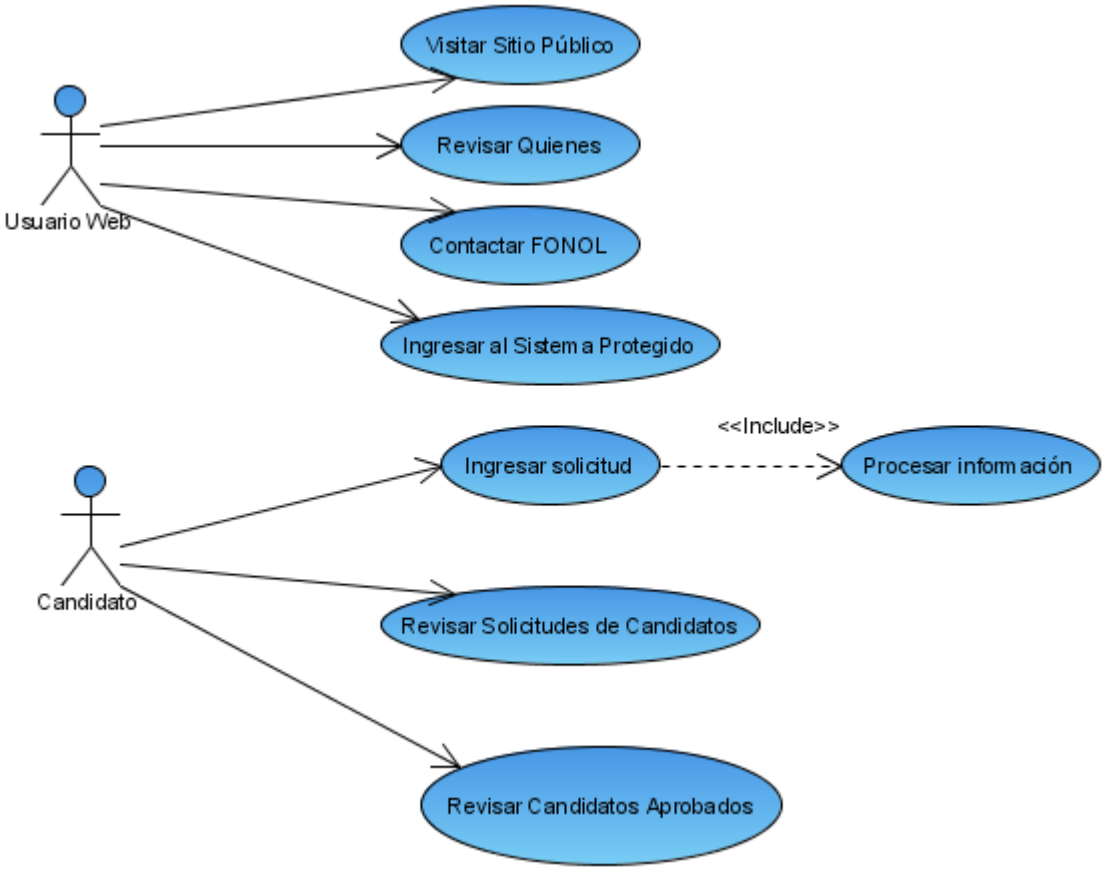


Figura 4.2

Paquete Sitio Protegido, con el Sub-Paquete que le corresponde al rol gobierno.

En la figura 4.3, se muestra el diagrama de casos de uso del paquete sistema protegido del rol gobierno, el cual representa las interacciones del actor gobierno a lo largo del sistema protegido en modo administrador:

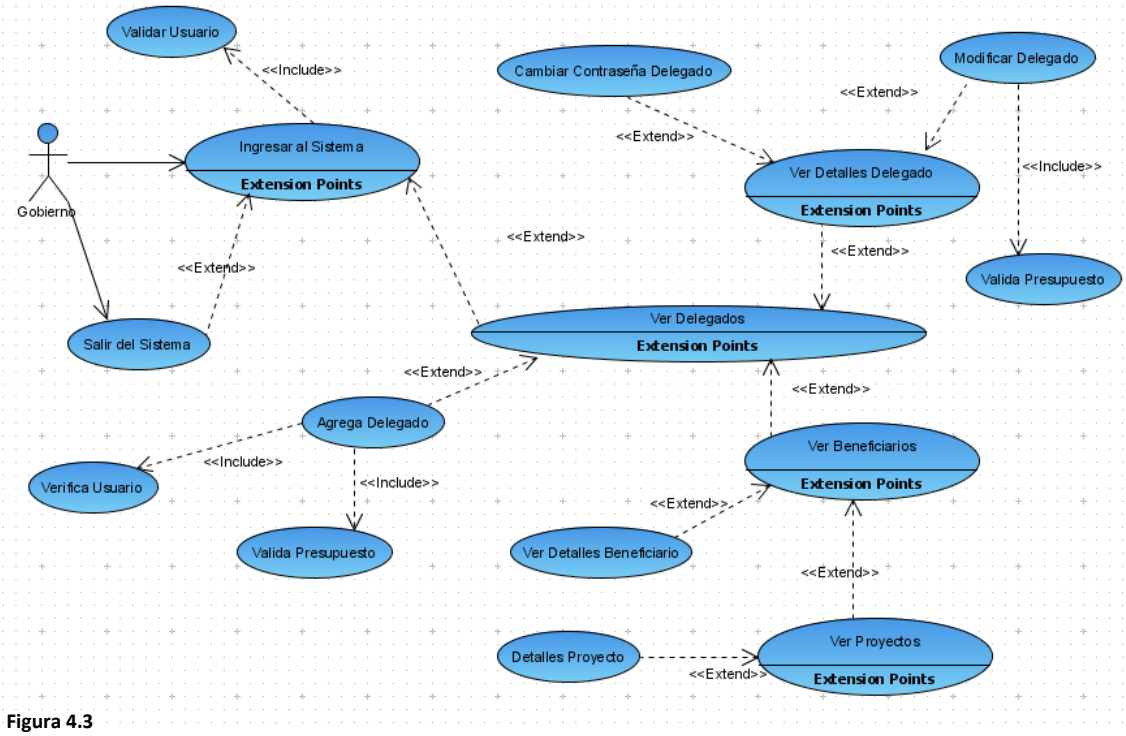


Figura 4.3

Paquete del Sistema Protegido con el rol gobierno.

La figura 4.4, muestra el diagrama de casos de uso del paquete sistema protegido del rol delegado:

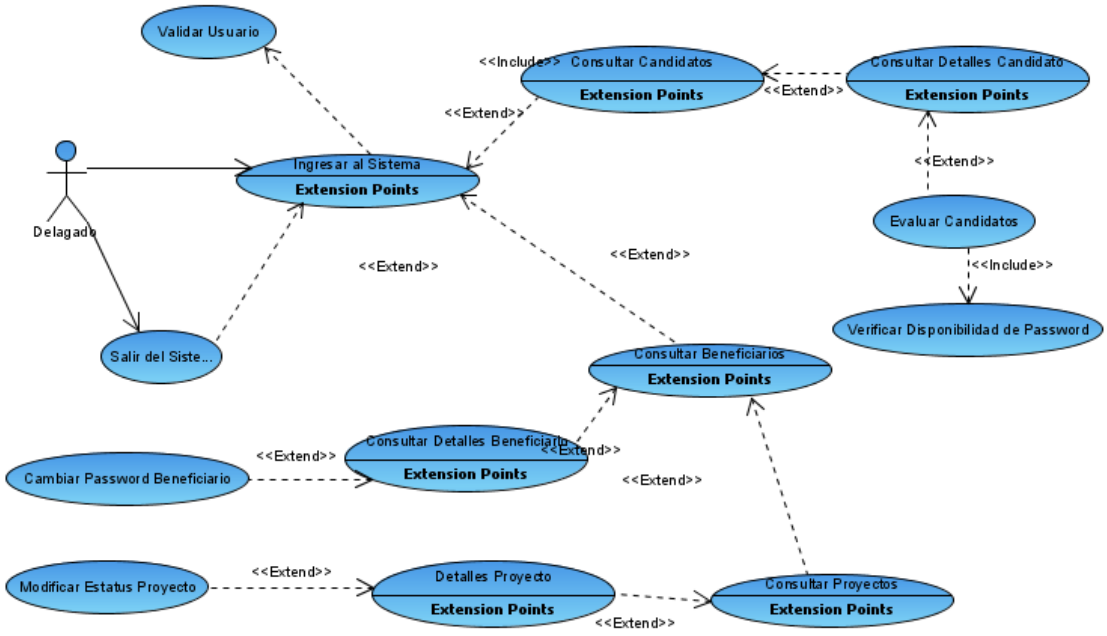


Figura 4.4

Paquete del Sistema Protegido con el rol delegado.

En la figura 4.5, se muestra el diagrama de casos de uso del paquete sistema protegido del rol beneficiario.

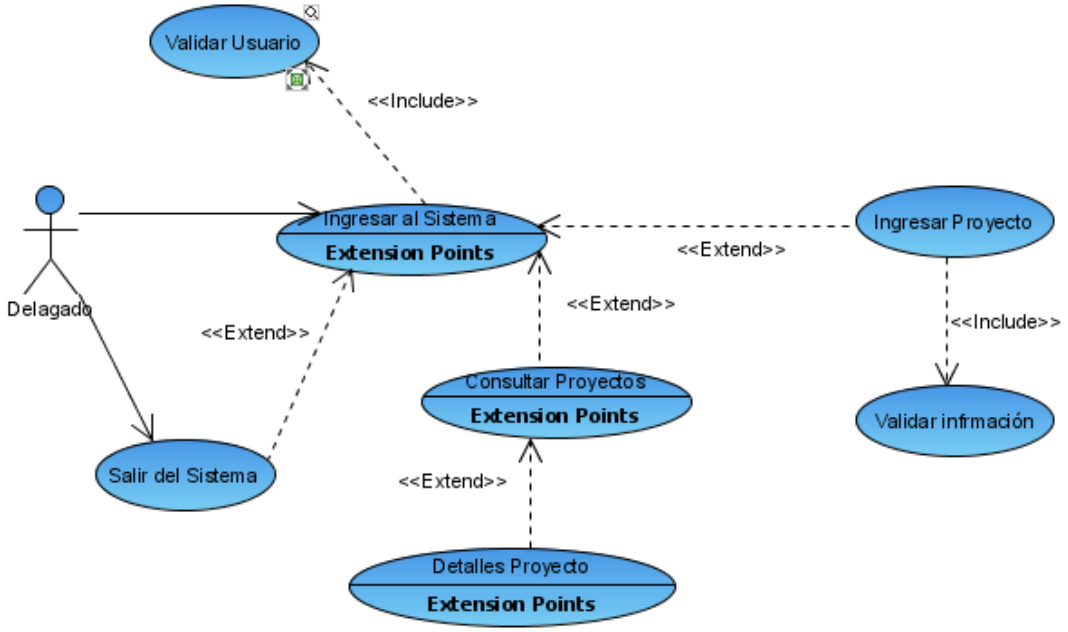


Figura 4.5

Paquete del Sistema Protegido con el rol beneficiario.

A continuación se explicará a detalle cada uno de los casos de uso para su mayor comprensión:

- Caso de uso Revisar Quienes Somos

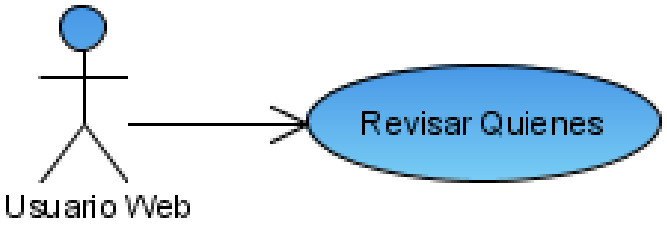


Figura 4.6

Caso de Uso Revisar Quienes Somos

  - Descripción breve:

    En este caso de uso, el usuario Web accede a la sección Quienes somos, para conocer a la organización
  - Flujo Básico de Eventos

    El usuario accede al sistema por internet al darle clic al botón Quienes somos
  - Escenarios
    - El usuario accede a la sección Quienes somos.

- Caso de Uso Contactar FONOL

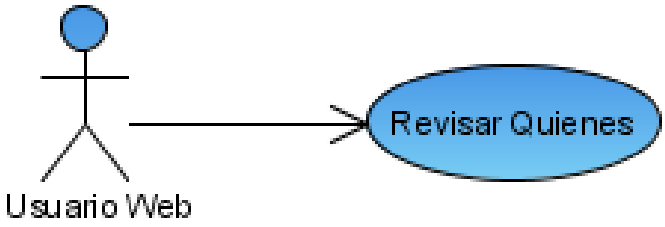


Figura 4.7

Caso de Uso Contactar FONOL

  - Descripción breve:

    En este caso de uso, el usuario web accede a la sección Contáctanos, para contactar a la organización.
  - Flujo Básico de Eventos

    El usuario accede al sistema por internet al darle clic al botón Contáctanos
  - Escenarios

- El usuario accede a la sección Contáctanos.

- Caso de Uso Ingresar al Sistema Protegido

Figura 4.8

Caso de Uso Revisar Quienes Somos

  - Descripción breve:

    En este caso de uso, el usuario web accede a la sección sitio protegido, para ingresar al sistema que gestiona la información.
  - Flujo Básico de Eventos

    El usuario accede al sistema protegido al darle clic al botón Acceso al Sistema
  - Escenarios

    El usuario accede a la sección Sistema Protegido.

- Caso de Uso Ingresar Solicitud

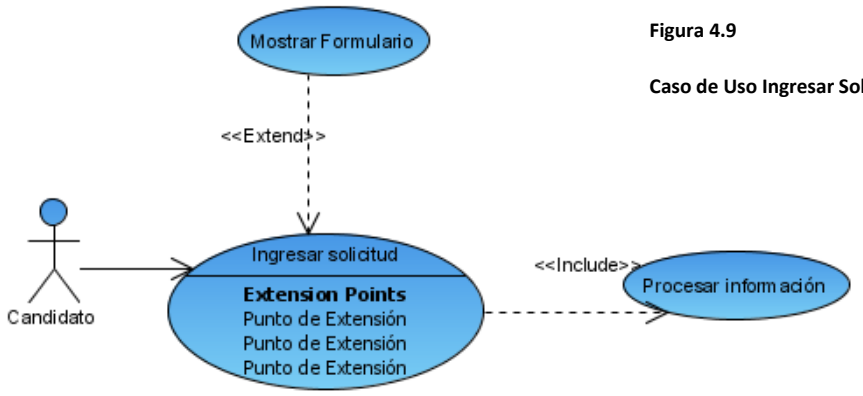


Figura 4.9

Caso de Uso Ingresar Solicitud

  - Descripción breve:

    En este caso de uso, el candidato que quiere ser beneficiado por el FONOL da clic en el botón solicitudes para acceder al formulario de la solicitud.
  - Flujo Básico de Eventos
    - El candidato accede a la sección solicitudes al dar clic en el botón solicitudes, lo cual provocará que se abra un formulario para ingresar sus datos.
    - Después, deberá ingresar la información de su organización y del representante de la misma con datos correctos que serán validados para verificar su consistencia. La información requerida por el formulario es: longitud de texto, email valido, RFC valido, y además, que el RFC de la organización no esté en otra organización, para evitar fraudes.
    - Cuando terminen de capturar la información y quieran enviarla al FONOL, se verificarán los datos, y si hay una validación que no hayan pasado, se les regresará al mismo formulario para corregirla.
    - Si la información es correcta, se permitirá el envío de la información al FONOL.
  - Escenarios

- El candidato ingresa información incorrecta. En este escenario, el candidato captura información incorrecta, lo cual provoca que el sistema le indique al usuario el error que cometió para que lo corrija.
- El candidato ingresa información correcta. En este escenario, el candidato captura información requerida correctamente, lo cual provoca que el sistema le permita enviar a información y redirigirlo a los detalles del candidato.

- Requerimientos Especiales
  - Se pide que el RFC además de ser válido, debe de verificarse que no esté repetido en otra organización candidata.

- Caso de Uso Revisar Solicitudes

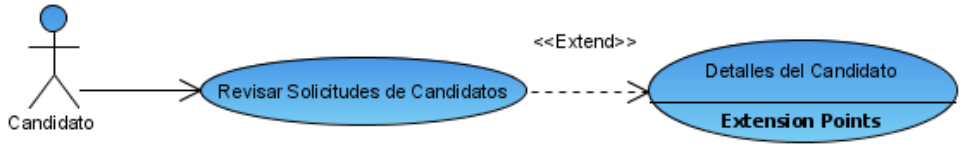


Figura 4.10

Caso de Uso Revisar Solicitud

- Descripción breve

  En este caso de uso, se muestra la lista de los Candidatos que han solicitado ser beneficiaros.

- Flujo Básico de Eventos

  Al dar clic en el botón candidatos, se mostrará una lista de los Candidatos que han ingresado una solicitud en los últimos 30 días. Y se les mostrará un combo en el cual pueden filtrar por estado.

  Si le interesa un Candidato, puede consultar sus detalles en el vínculo Enviar de cada renglón.

- Escenarios
  - El candidato filtra por estado.
  - El candidato que desea consultar los detalles, le da clic al vinculo Enviar y lo llevará al caso de uso Consultar Detalles.

- Caso de Uso Consultar Detalles del Candidato

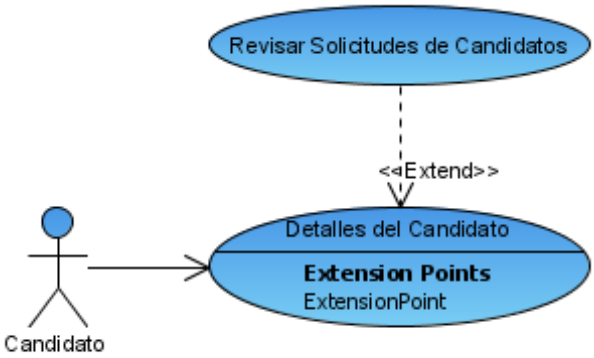


Figura 4.11

Caso de Uso Consultar Detalles del Candidato

  - Descripción breve:

    En este caso de uso, el candidato puede consultar los detalles de un candidato
  - Flujo Básico de Eventos

    El candidato entra a consultar los Detalles del Candidato.
  - Escenarios
    - El candidato entra a consultar los Detalles del Candidato.

- Caso de Uso Revisar Candidatos Aprobados

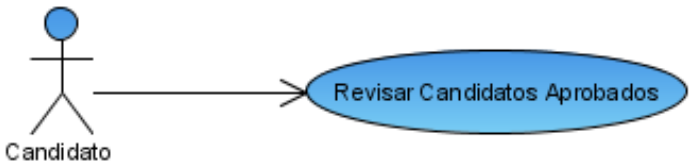


Figura 4.12

Caso de Uso Revisar Candidatos Aprobados

  - Descripción breve:

    En este caso de uso, el Candidato puede consultar la lista de Candidatos aprobados.
  - Flujo Básico de Eventos

    El candidato accede a la sección Resultados al dar clic en el botón Resultados, lo cual provocará que vea una lista de los Candidatos que han sido aceptados por el FONOL.
  - Escenarios

- El candidato ve una lista de los candidatos aceptados.

- Caso de Uso Visitar el Sitio Público

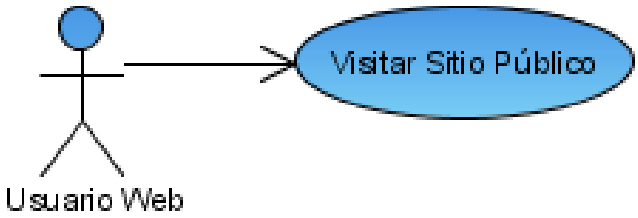


Figura 4.13

Caso de Uso Visitar Sitio Público

  - Descripción breve:

    En este caso de uso, el usuario Web accede al sistema por Internet

  - Flujo Básico de Eventos

    El usuario accede al sistema por internet, al teclear la dirección del Sistema FONOL.

  - Escenarios
    - El usuario accede al portal web.

- Caso de Uso Ingresar al Sistema

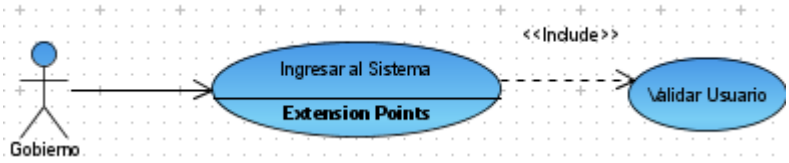


Figura 4.14

Caso de Uso Ingresar al Sistema

  - Descripción breve:

    En este caso de uso, el gobierno ingresa su usuario y su contraseña para ingresar al sistema.

  - Flujo Básico de Eventos

    El gobierno, al teclear el vínculo de Acceso al Sistema desde el sistema público, se le muestra un formulario el cual tiene que ingresar su usuario y contraseña para ingresar al sistema protegido.

  - Escenarios
    - El gobierno ingresa un usuario y una contraseña, los cuales son de longitud diferente de nueve caracteres. El sistema le indica que debe respetar la longitud de nueve caracteres en el campo usuario y contraseña.

- El gobierno ingresa un usuario inválido, o un usuario válido con una contraseña incorrecta. El sistema se conecta con el caso de uso Validar Usuario, el cual valida sus credenciales con la base de datos, y le indica que debe ingresar un usuario y contraseña válidos.
- El gobierno ingresa correctamente su usuario y contraseña, lo cual le permite ingresar al sistema.

- Caso de Uso Salir del Sistema

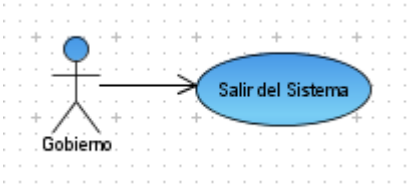


Figura 4.15

Caso de Uso Salir del Sistema

  - Descripción breve:

    En este caso de uso, el gobierno sale del sistema.
  - Flujo Básico de Eventos

    El gobierno selecciona el botón logout el cual le permite salir del sistema y eliminar la sesión por seguridad.
  - Escenarios
    - El gobierno selecciona el botón Logout el cual le permite salir del sistema y eliminar la sesión por seguridad.

- Caso de Uso Consultar Delegados

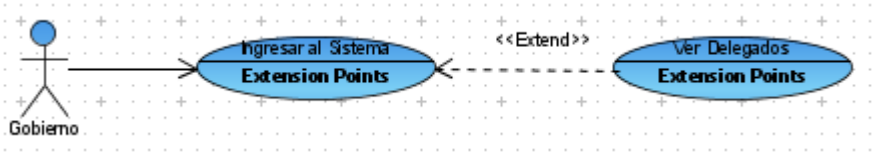


Figura 4.16

Caso de Uso Consultar Delegados

  - Descripción breve:

    En este caso de uso, el gobierno ve una lista de todos los delegados que pertenecen a su estado, respectivamente.

- Flujo Básico de Eventos

  El gobierno, después de ingresar al sistema protegido, selecciona el botón Consultar Delegados, el cual le mostrará una lista de todos los delegados que pertenecen a un estado, así como, sus datos generales, su presupuesto y un vínculo para observar los detalles cada delegado. También, en la misma pantalla, el gobierno tiene un vínculo para agregar un nuevo delegado de un estado que no cuente con uno.

- Escenarios
  - El gobierno selecciona el botón Consultar Delegados, de la página principal del sistema protegido, el cual, le mostrará una lista de todos los delegados.

- Caso de Uso Agregar Delegado

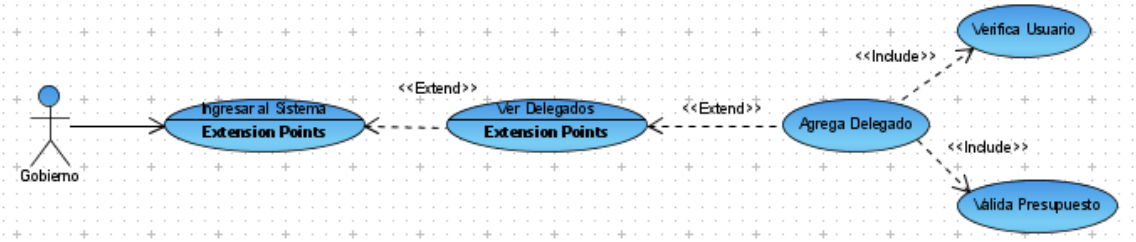


Figura 4.17

Caso de Uso Agregar Delegado

- Descripción breve:

  En este caso de uso, el gobierno asocia un delegado con un Estado que no cuente con uno.

- Flujo Básico de Eventos

  El Gobierno, después de ingresar al sistema protegido, selecciona el botón Consultar Delegados, posteriormente, selecciona el botón Agregar Delegado, lo cual le mostrará un formulario en el cual el gobierno ingresa los datos del delegado, así como el presupuesto que le asignará a éste. El formulario validará la información

ingresada (debe ingresar un correo electrónico válido, un RFC válido, etc.) y si se cuenta con el presupuesto para asignárselo.

- Escenarios
  - El gobierno ingresa la información incorrecta del delegado, lo cual le mostrará un mensaje de error apropiado dependiendo del campo incorrecto.
  - El gobierno asigna un nombre usuario al delegado que ya esté en uso por algún otro usuario del sistema, lo cual, le arrojará un mensaje de error.
  - El gobierno asigna un nombre de usuario y una contraseña al delegado, pero con una longitud diferente de nueve caracteres, lo cual le mostrará un mensaje apropiado a éste.
  - El gobierno ingresa todos los datos generales correctos, pero al momento de ingresar el presupuesto, asigna más presupuesto del que tiene, lo cual le mostrará un mensaje de error correspondiente.
  - El gobierno ingresa todos los datos correctos, lo cual le permite asociar el delegado con un estado disponible.

- Caso de Uso Modificar Delegado

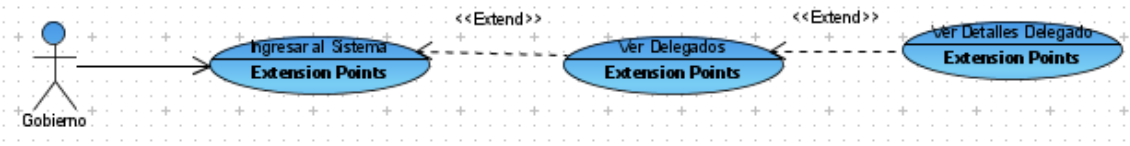


Figura 4.18

Caso de Uso Modificar Delegado

- Descripción breve:

  En este caso de uso, el gobierno observa y puede seleccionar vínculos para modificar aspectos de los delegados existentes.
- Flujo Básico de Eventos

El Gobierno, después de ingresar al sistema protegido, selecciona el botón Consultar Delegados, posteriormente, selecciona el botón Consultar Detalles de un registro de interés, lo cual le mostrará toda la información del delegado.

- Escenarios
  - El gobierno revisa la información del delegado.

- Caso de Uso Cambiar Contraseña del Delegado

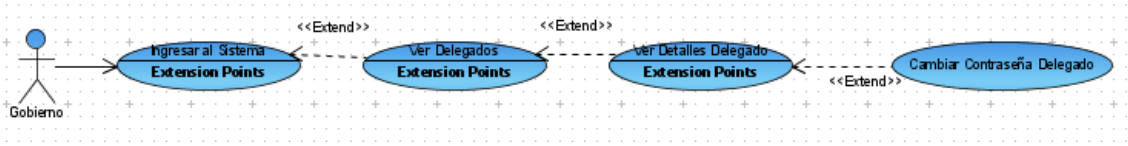


Figura 4.19

Caso de Uso Cambiar Contraseña del Delegado

- Descripción breve:

  En este caso de uso, el gobierno cambia la contraseña del delegado.

- Flujo Básico de Eventos

  El gobierno, después de ingresar al sistema protegido, selecciona el botón Consultar Delegados, posteriormente, selecciona el botón Consultar Detalles de un registro de interés, lo cual le mostrará toda la información del delegado. Y desde esa pantalla modifica el usuario o contraseña del delegado.

- Escenarios
  - El gobierno ingresa o modifica la contraseña existente con una longitud de caracteres diferente de nueve, lo cual le muestra un mensaje de error personalizado.
  - El gobierno ingresa una contraseña válida, lo cual le permite modificar al usuario.

- Caso de Uso Modificar Delegado

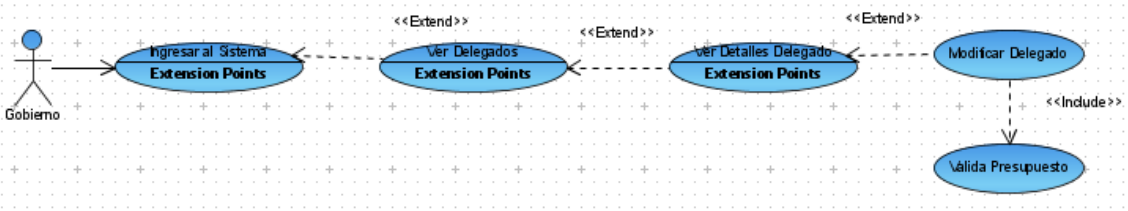


Figura 4.20

Caso de Uso Cambiar Contraseña del Delegado

- Descripción breve:

  En este caso de uso, el gobierno modifica los datos personales de un delegado.

- Flujo Básico de Eventos

  El gobierno, después de ingresar al sistema protegido, selecciona el botón Consultar Delegados, posteriormente, selecciona el botón Consultar Detalles de un registro de interés, lo cual le mostrará toda la información del delegado. Y desde esa pantalla modifican los datos personales del delegado.

- Escenarios
  - El gobierno modifica la información del delegado de forma incorrecta, lo cual le mostrará un mensaje de error apropiado dependiendo del campo incorrecto.
  - El gobierno asigna una contraseña al delegado, pero con una longitud diferente de nueve caracteres, lo cual le mostrará un mensaje apropiado a éste.
  - El gobierno ingresa todos los datos generales correctos, pero al momento de ingresar el presupuesto, asigna más presupuesto del que tiene, lo cual le mostrará un mensaje de error correspondiente.
  - El Gobierno ingresa todos los datos correctos, lo cual le permite asociar el delegado con un estado disponible.

- Caso de Uso Consultar Beneficiarios

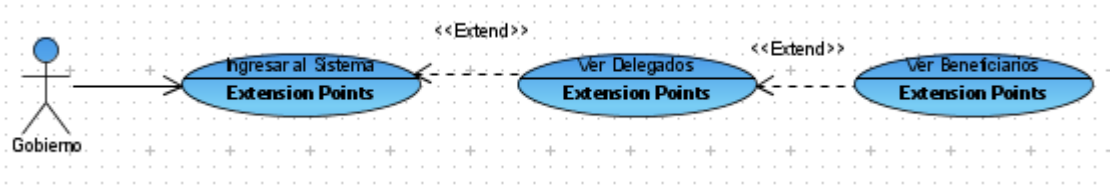


Figura 4.21

Caso de Uso Consultar Beneficiarios

- Descripción breve:

  En este caso de uso, el gobierno ve una lista de todos los Beneficiarios asociados a un delegado o a un estado.

- Flujo Básico de Eventos

  El gobierno, después de ingresar al sistema Protegido, selecciona el botón Consultar Delegados, posteriormente, selecciona el vinculo Consultar Beneficiarios asociado a un registro de un delegado.

  Al seleccionarlo, le mostrará una lista de todos los beneficiarios que pertenecen a un estado, así como, sus datos resumidos, un vínculo para observar los detalles cada beneficiario y un vínculo para observar los proyectos asociados a cada beneficiario.

- Escenarios

  - El gobierno selecciona el vínculo Consultar Beneficiarios, de la pantalla Consultar Delegados, el cual, le mostrará una lista de todos los beneficiarios.

- Caso de Uso Consultar Detalles del Beneficiario

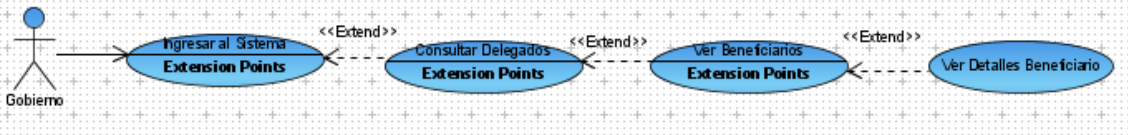


Figura 4.22

Caso de Uso Consultar Detalles del Beneficiario

- Descripción breve:

  En este caso de uso, el gobierno consulta los detalles del beneficiario.

- Flujo Básico de Eventos

  El Gobierno, después de ingresar al sistema protegido, selecciona el botón Consultar Delegados, posteriormente, selecciona el vinculo Consultar Beneficiarios asociado a un registro de un delegado, y finalmente, selecciona el vínculo de un beneficiario de interés para consultar sus detalles.

  Al seleccionarlo, le mostrará todos los detalles del beneficiario.

- Escenarios

  - El gobierno selecciona el vínculo Consultar Detalles del Beneficiario, de la pantalla consultar beneficiarios, el cual, le mostrará todos los detalles del beneficiario.

- Caso de Uso Consultar Proyectos

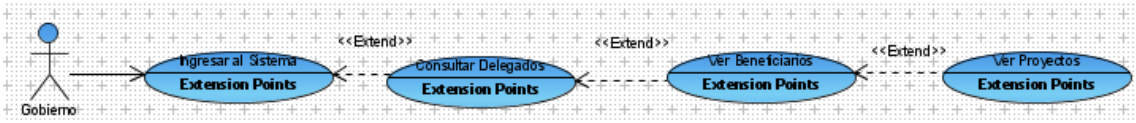


Figura 4.23

Caso de Uso Consultar Proyectos

  - Descripción breve:

    En este caso de uso, el gobierno ve una lista de los proyectos asociados a un beneficiario.

  - Flujo Básico de Eventos

    El gobierno, después de ingresar al sistema protegido, selecciona el botón Consultar Delegados, posteriormente, selecciona el vinculo Consultar Beneficiarios asociado a un registro de un delegado, y finalmente, selecciona el vínculo de un beneficiario de interés para consultar sus proyectos.

    Al seleccionarlo, le mostrará la lista de los proyectos del beneficiario.

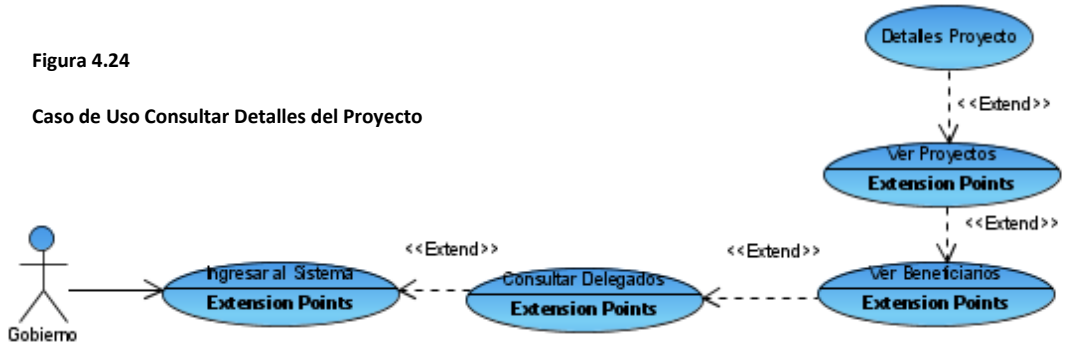
  - Escenarios

    - El gobierno selecciona el vínculo consultar Proyectos, de la pantalla consultar Beneficiarios, el cual, le mostrará la lista de todos los proyectos del beneficiario.

- Caso de Uso Consultar Detalles del Proyecto

Figura 4.24

Caso de Uso Consultar Detalles del Proyecto



- Descripción breve:

  En este caso de uso, el gobierno consulta los Detalles del Proyecto.

- Flujo Básico de Eventos

  El gobierno, después de ingresar al sistema Protegido, selecciona el botón Consultar Delegados, posteriormente, selecciona el vinculo Consultar Beneficiarios asociado a un registro de un delegado, seguido de esto, selecciona el vínculo de un beneficiario de interés para consultar sus proyectos, y finalmente, seleccionamos el vinculo Consultar Detalles del proyecto.

  Al seleccionarlo, le mostrará todos los detalles del proyecto.

- Escenarios
  - El gobierno selecciona el vínculo Consultar Detalles del Proyecto, de la pantalla consultar Proyectos, el cual, le mostrará todos los detalles del proyecto.

En la siguiente sección se analizarán los diagramas de colaboración, como la realización de los diagramas de casos de uso.

4.2.5. Diagramas de Colaboración

4.2.5.1. Visitar el Sitio Público

En la figura 4.25 se muestra el diagrama de colaboración del caso de uso Visitar Sitio Público, y se puede visualizar como el usuario para ingresar al sistema, teclea la dirección del sistema público, lo cual provoca que se despliegue la página del sistema público para navegar.

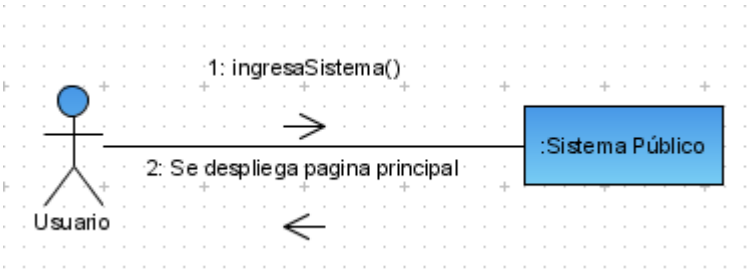


Figura 4.25

Diagrama de colaboración del caso de uso Visitar el sitio público

4.2.5.2. Revisar Quienes Somos

En la figura 4.26 se muestra como el usuario interactúa para revisar la información en general del FONOL.

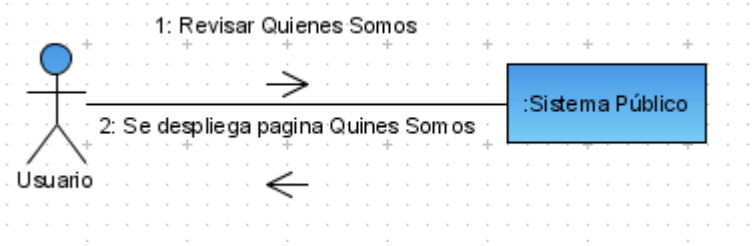


Figura 4.26

Diagrama de colaboración del caso de uso Revisar Quienes Somos

4.2.5.3. Contactar FONOL

En la figura 4.27, se muestra el diagrama de colaboración del caso de uso contactar FONOL, en el cual se puede visualizar como el usuario que quiere contactar al FONOL, para obtener información o por dudas, debe presionar un vínculo del Sistema Público para obtener la información.

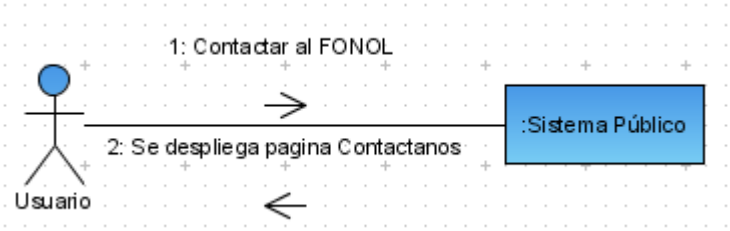


Figura 4.27

Diagrama de colaboración del caso de uso Contactar al FONOL

4.2.5.4. Revisar Solicitudes

En la figura 4.28, se muestra que, para revisar las solicitudes del sistema, se involucran tres capas con sus componentes. En primera instancia, el usuario pide la información al acceder a la página para consultar solicitudes. Después, el objeto acción solicitudes, que es el intermediario entre la capa de usuario y la capa de negocio, maneja la información del formulario y la envía al facade candidato, que a su vez será el responsable de interactuar directamente con el objeto candidato. El objeto candidato, es quien, realiza todas las interacciones con la base de datos, sin tener que ocuparse de las consultas a la base de datos. Finalmente, se enviará al usuario la información de los candidatos, para que verifique los resultados.

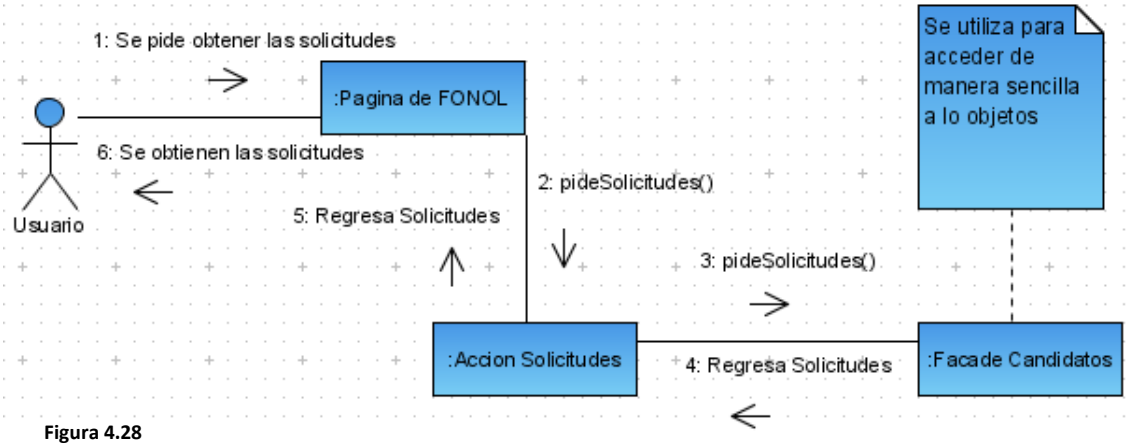


Figura 4.28

Diagrama de colaboración del caso de uso Revisar Solicitudes

4.2.5.5. Ingresar Solicitud

En la figura 4.29, diagrama de colaboración donde interactúan los diferentes objetos del sistema. En primera instancia, el candidato ingresa la información necesaria para su solicitud, al enviar la información la Interfaz gráfica del FONOL, que es un formulario web; el formulario web, envía la información al objeto Acción Ingresar Solicitud que manejará la interacción entre la capa de usuario y la de negocio. Posteriormente, el objeto acción se encargará de ayudarse del objeto Validador para saber si la información es correcta, en caso afirmativo, se continuará con el flujo de información. Seguido de esto, ya con la información correcta se conecta con el facade candidato, el cual traerá una lista de los candidatos disponibles para que, finalmente, se le regrese al usuario y los visualice.

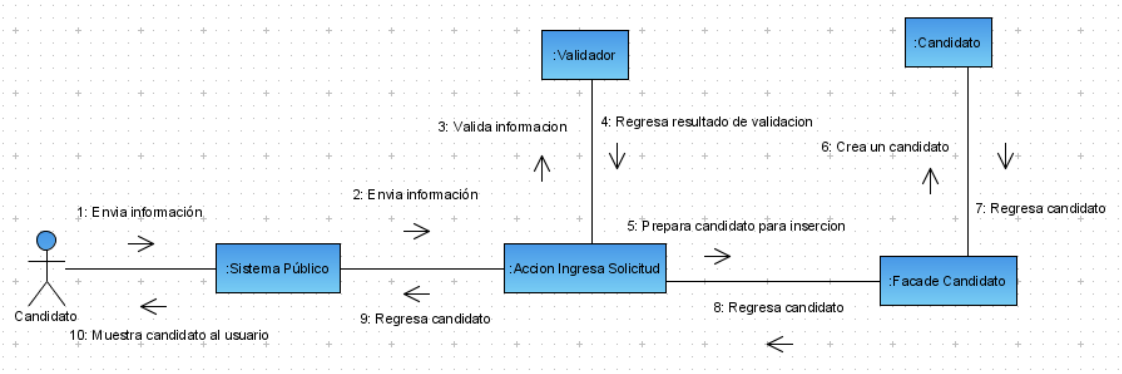


Figura 4.29

Diagrama de colaboración del caso de uso Ingresar Solicitud

4.2.5.6. Consultar Detalles del Candidato

En la figura 4.30, se puede observar cómo interactúan los distintos objetos de este caso de uso. Primero, el usuario selecciona un vínculo para observar los detalles de un candidato. Seguido de esto, el vínculo se comunica con el objeto Acción Solicitudes, el cual interactúa entre la capa de usuario y de negocio, al pedir la información del candidato al objeto Facade Candidatos. Este objeto

interactúa con el candidato de manera simple para obtener los detalles y regresarlos al usuario.

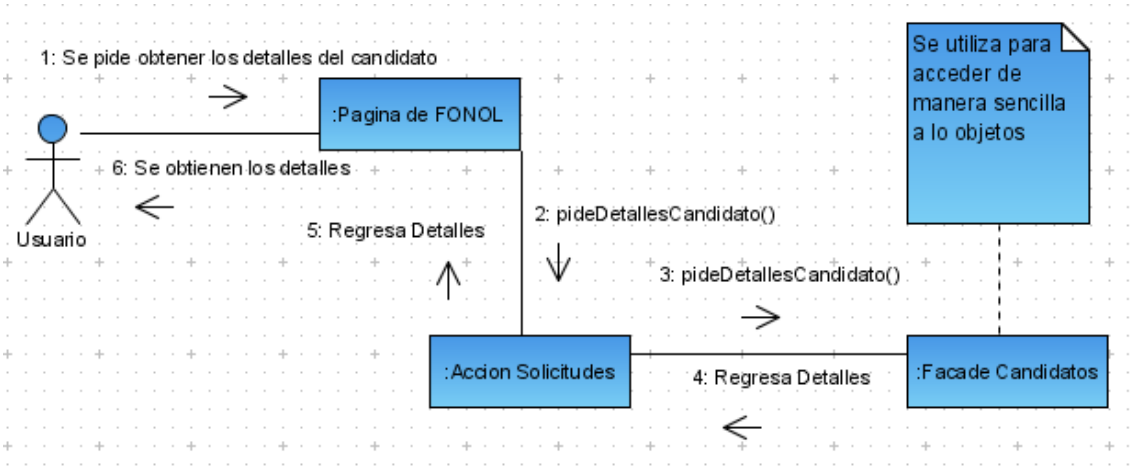


Figura 4.30

Diagrama de colaboración del caso de uso Consulta Detalles del Candidato

4.2.5.7. Revisar Candidatos Aprobados

En la figura 4.31, se muestra como para revisar las solicitudes aceptadas de los candidatos, se involucran las tres capas del sistema con sus componentes. En primera instancia, el usuario pide la información al acceder a la página de consultar solicitudes aceptadas. Después, el objeto Acción Solicitudes, que es el intermediario entre la capa de usuario y la capa de negocio, maneja la información del formulario y la envía al Facade candidato, que a su vez será el responsable de interactuar directamente con el objeto Candidato. El objeto Candidato, es el cual, realiza todas las interacciones con la base de datos, sin tener que preocuparse con las consultas a la base de datos. Finalmente, se enviará al usuario la información de los candidatos aceptados para mostrárselos al usuario.

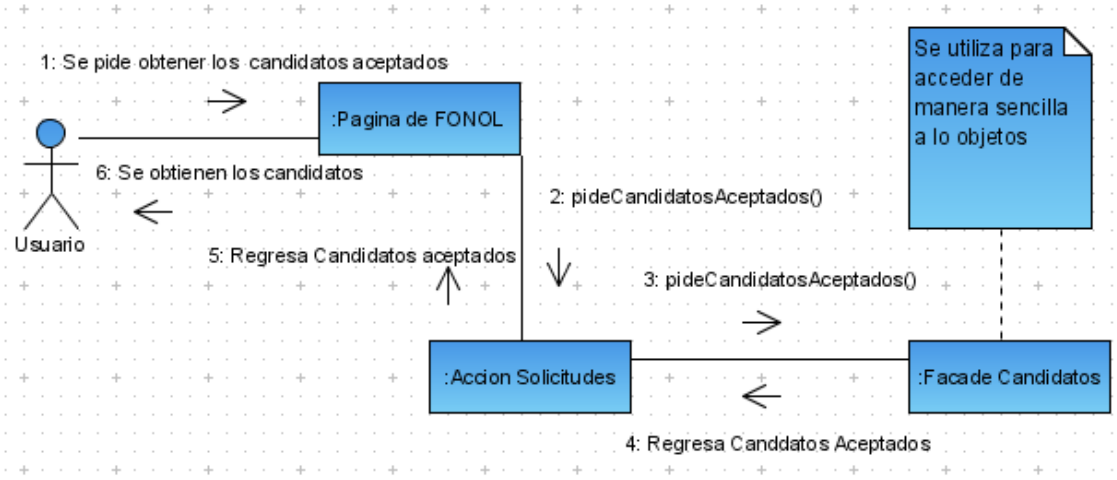


Figura 4.30

Diagrama de colaboración del caso de uso Revisar Candidatos Aceptados

4.2.5.8. Ingresar al Sistema Protegido

En la figura 4.31, se muestra cómo interactúan los objetos del sistema. Primero, se envían las credenciales del usuario en el formulario de ingreso al sistema. Posteriormente, se envían las credenciales del formulario al objeto Acción Valida Usuario, que a su vez, le pedirá al Facade Usuario un usuario válido de ser correctas las credenciales. Finalmente, se regresa el objeto Usuario al usuario si las credenciales son correctas.

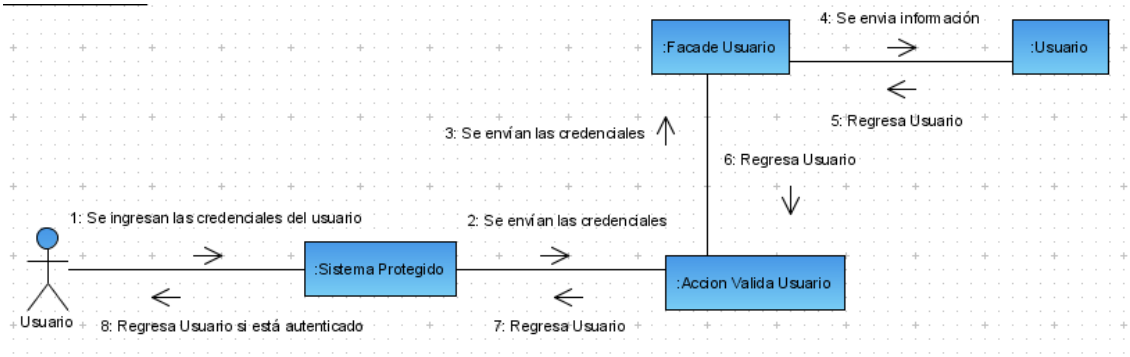


Figura 4.31

Diagrama de colaboración del caso de uso Ingresar al Sistema Protegido

4.2.5.9. Salir del Sistema

En la figura 4.32, se muestra el diagrama de colaboración del caso de uso Salir del Sistema, el cual, representa salir del sistema protegido una vez que se está

adentro. Primero, el usuario en el sistema protegido selecciona el vínculo Salir del sistema, el cual, se comunica con el objeto Acción Logout. El objeto Logout se encarga de la interacción entre la capa de usuario y la capa de negocio, y en este caso, obtiene el objeto Sesión para eliminarlo e invalidar la sesión. Finalmente, ya eliminado el objeto Sesión e invalidada la sesión, se le avisa al usuario y lo saca del sistema.

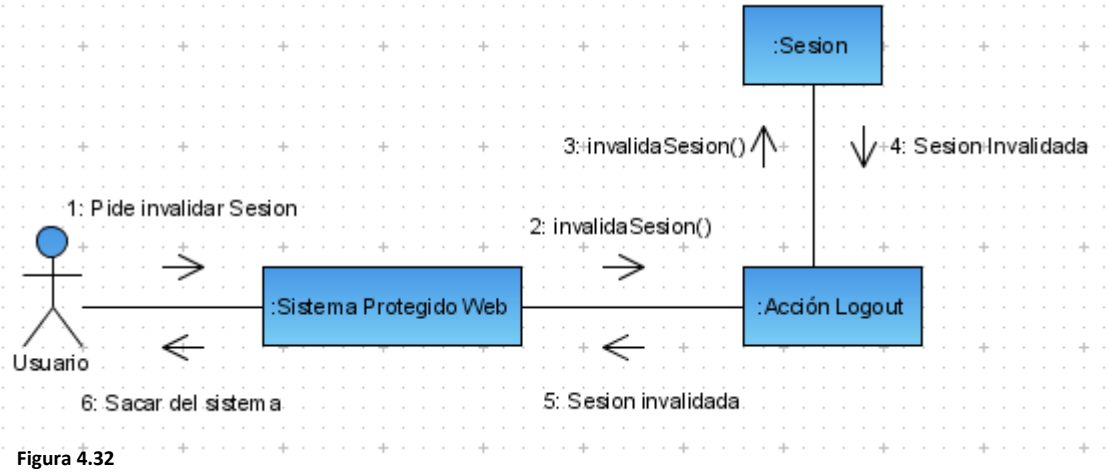


Figura 4.32

Diagrama de colaboración del caso de uso Salir del Sistema Protegido

4.2.5.10. Consultar Candidatos

En la figura 4.33, se muestra como para revisar los candidatos, se involucran las tres capas del sistema con sus componentes. En primera instancia, el usuario pide la información al acceder a la página de Consultar Candidatos. Después, el objeto Acción Consultar Candidatos, que es el intermediario entre la capa de usuario y la capa de negocio, maneja la información del formulario y la envía al Facade candidato, que a su vez será el responsable de interactuar directamente con el objeto Candidato. El objeto Candidato, es el cual, realiza todas las interacciones con la base de datos, sin tener que preocuparse con los queries. Finalmente, se enviará al usuario la lista de los candidatos para mostrárselos al usuario.

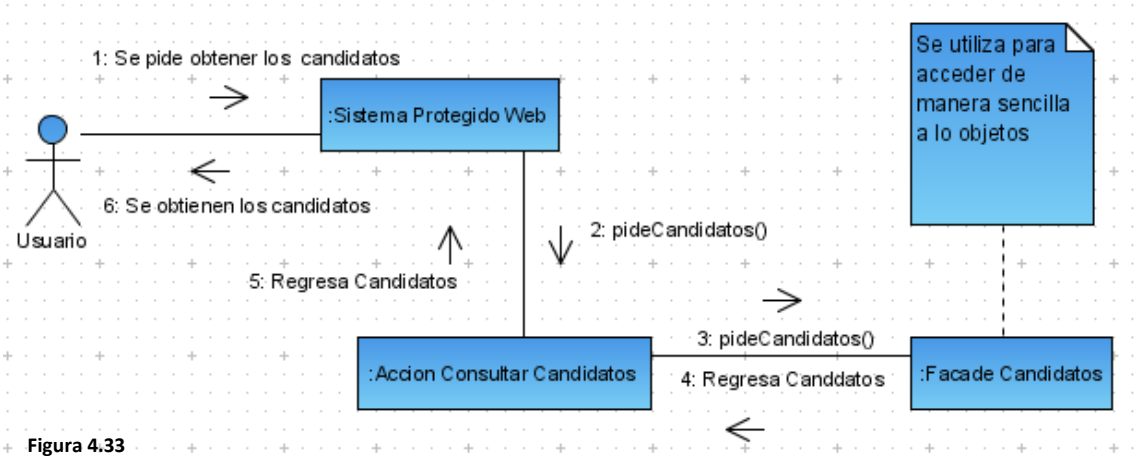


Figura 4.33

Diagrama de colaboración del caso de uso Salir Consultar Candidatos

4.2.5.11. Evaluar Candidato

En la figura 4.34, se muestra el diagrama de colaboración del caso de uso Evaluar Candidato, el cual, se explicará a continuación. En primera instancia, el usuario, que es este caso de uso es el delegado, modifica el estatus del candidato, lo cual provoca que el Sistema Web envíe la información actualizada al objeto Acción Evalúa Candidato. Este objeto, se encargará de la interacción entre la capa de usuario y la de negocio, y en el caso del sistema, se encargará de comunicarse con el objeto Facade para actualice el estatus del candidato. Posteriormente, el Facade Candidato regresa la confirmación que indica que el candidato se ha actualizado correctamente, lo cual, llega al objeto Acción Evalúa Candidato de regreso. Ahí, si el candidato fue aceptado por el estatus que llevaba, el objeto Acción llamará al Facade Beneficiario, el cual creará un nuevo beneficiario y se lo regresará al usuario para que vea la confirmación.

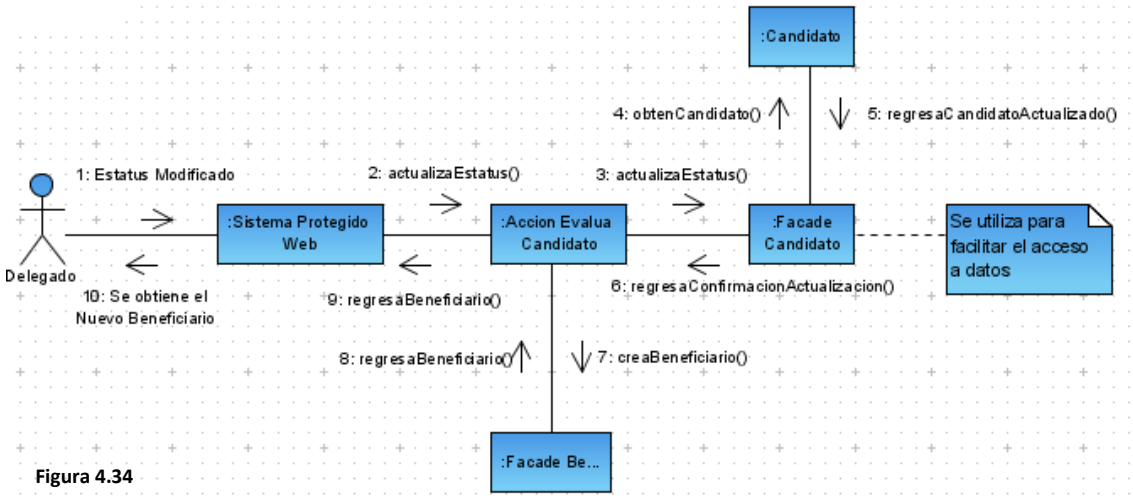


Figura 4.34

Diagrama de colaboración del caso de uso Evaluar Candidato

4.2.5.12. Consultar Delegados

En la figura 4.35, se muestra como para revisar los delegados, se involucran las tres capas del sistema con sus componentes. En primera instancia, el usuario pide la información al acceder a la página de Consultar Delegados. Después, el objeto Acción Consultar Delegados, que es el intermediario entre la capa de usuario y la capa de negocio, maneja la información del formulario y la envía al facade delegado, que a su vez será el responsable de interactuar directamente con el objeto delegado. El objeto delegado, es el cual, realiza todas las interacciones con la base de datos, sin tener que preocuparse con los queries. Finalmente, se enviará al usuario, que en este caso de uso es el gobierno, la lista de los delegados para mostrárselos al usuario.

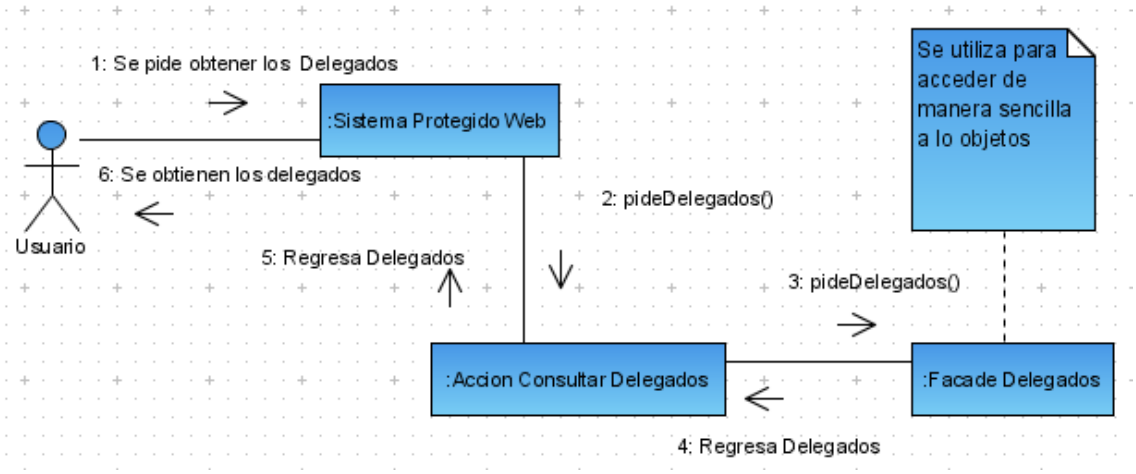


Figura 4.35

Diagrama de colaboración del caso de uso Consultar Delegados

4.2.5.13. Agregar Delegado

En la figura 4.36, se muestra el diagrama de componentes del caso de uso Agregar Delegado, en el cual se notan los objetas y las interacciones existentes. En primera instancia, el usuario, que en este caso es el gobierno, llena la información del delegado en el formulario del Sistema Protegido, al enviar el formulario, el sistema web envía la información al objeto Acción Agrega Delegado, el cual se encargará de validar la información y llevar el flujo de la información; esto lo hace llamando al objeto validador, el cual valida información correcta, y al objeto Valida Presupuesto, el cual se encarga de asegurarse si se cuenta con el presupuesto suficiente para asignárselo al delegado. Una vez que concluyan las validaciones correctamente, el objeto Acción manda la información al objeto facade delegado para que agregue el delegado, ya que el facade se encarga de la interacción con la base de datos y hacer que las operaciones sean más simples, y es este caso se encarga de insertar el delegado en la base de datos. Finalmente, ya ingresado, se retorna el objeto delegado al usuario para mostrar que la operación se llevo a cabo satisfactoriamente.

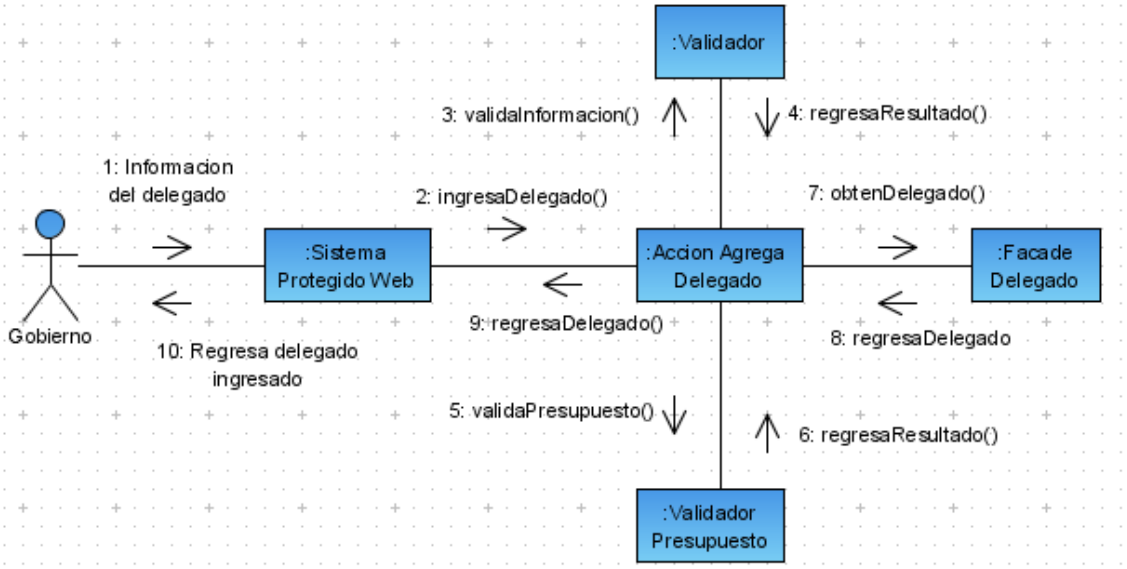


Figura 4.36

Diagrama de colaboración del caso de uso Agregar Delegado

4.2.5.14. Modificar Delegado

En la figura 4.37, se muestra el diagrama de colaboración del caso de uso Modifica Delgado, el cual es semejante al caso de Uso Agregar Delegado, la única diferencia es que en lugar de tomar la información para ingresar el delegado a la base de datos, se utilizará la información para actualizar la base de datos. Igualmente, en este diagrama intervienen los mismos objetos los cuales realizan funciones similares. En primer lugar, el usuario actualiza la información del delgado y la envía al objeto Acción Modifica Delegado, el cual, una vez que valide la información y el presupuesto, enviará la información al objeto facade para que la actualice en la base de datos. Finalmente, ya actualizada la información, se reenvía al usuario para confirmar los cambios y que la operación se ha realizado correctamente.

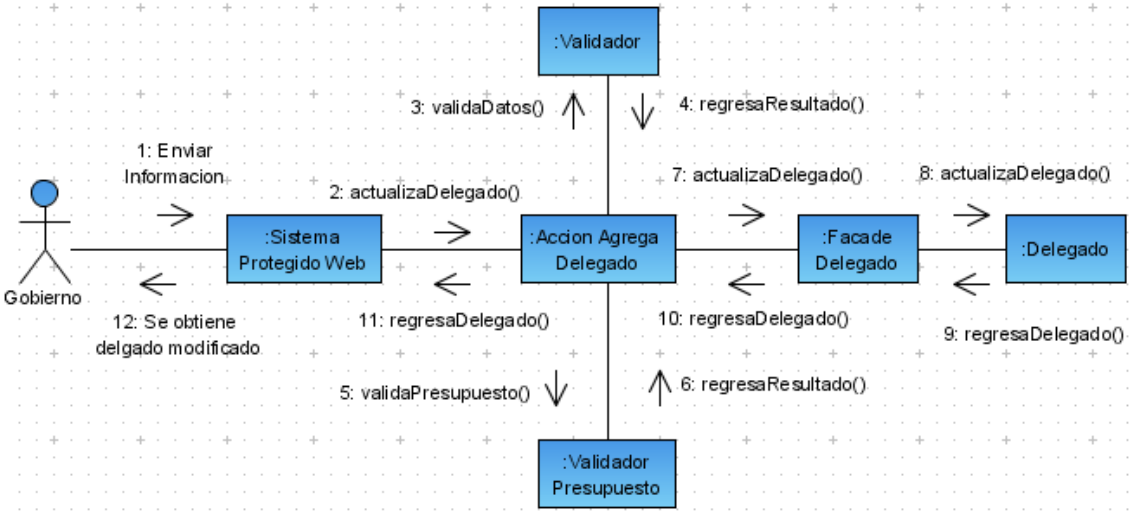


Figura 4.37

Diagrama de colaboración del caso de uso Modificar Delegado

4.2.5.15. Consultar Beneficiarios

En la figura 4.38, se muestra como para revisar los beneficiarios, se involucran las tres capas del sistema con sus componentes. En primera instancia, el usuario pide la información al acceder a la página de Consultar Beneficiarios. Después, el objeto Acción Consultar Beneficiarios, que es el intermediario entre la capa de usuario y la capa de negocio, maneja la información del formulario y la envía al Facade beneficiario, que a su vez será el responsable de interactuar directamente con el objeto beneficiario. El objeto beneficiarios, es el cual, realiza todas las interacciones con la base de datos, sin tener que preocuparse con los queries. Finalmente, se enviará al usuario, que en este caso de uso es el gobierno y el delegado; la lista de los delegados para mostrárselos al usuario.

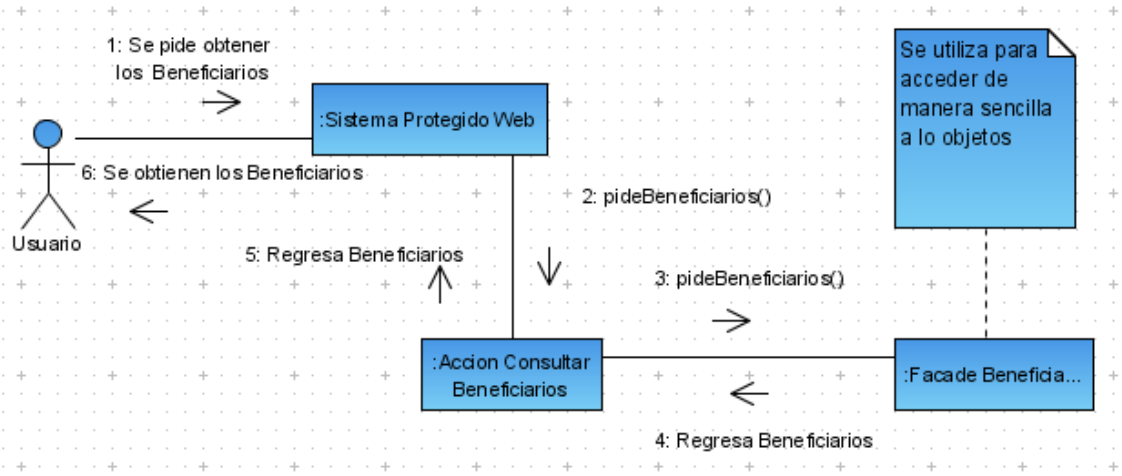


Figura 4.38

Diagrama de colaboración del caso de uso Consultar Beneficiarios

4.2.5.16. Modificar Beneficiario

En la figura 4.39, se muestra el diagrama de colaboración del caso de uso Modifica Beneficiario, el cual es semejante al caso de Uso Agregar Beneficiario, la única diferencia es que en lugar de tomar la información para ingresar el Beneficiario a la base de datos, se utilizará la información para actualizar la base de datos. Igualmente, en este diagrama intervienen los mismos objetos los cuales realizan funciones similares. En primer lugar, el usuario actualiza la información del beneficiario y la envía al objeto Acción Modifica Beneficiario, el cual, una vez que valide la información y el presupuesto, enviará la información al objeto facade para que la actualice en la base de datos. Finalmente, ya actualizada la información, se reenvía al usuario para confirmar los cambios y que la operación se ha realizado correctamente.

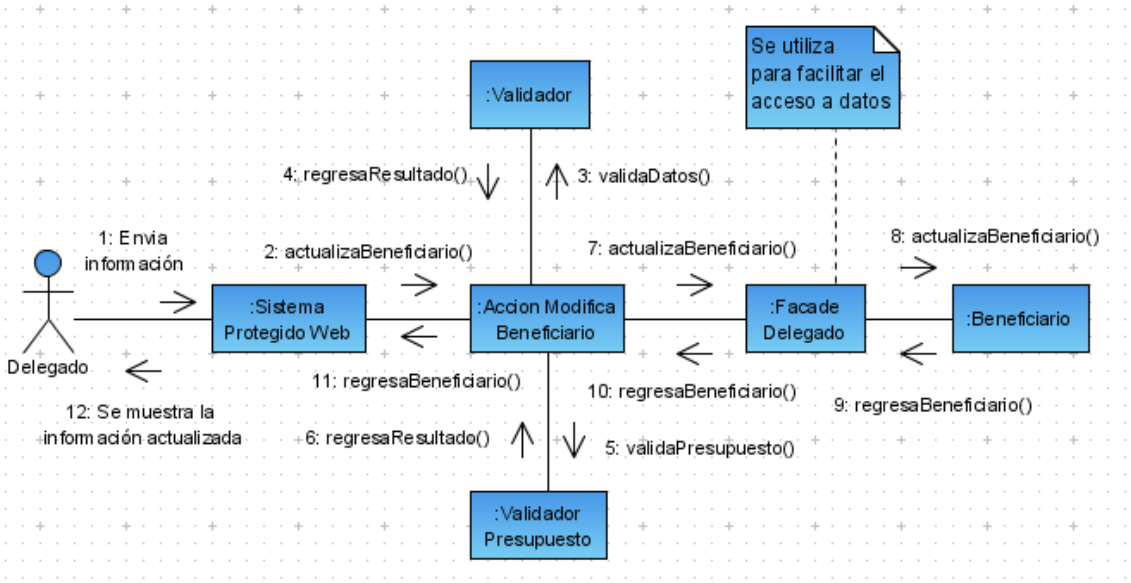


Figura 4.39

Diagrama de colaboración del caso de uso Modificar Beneficiario

4.2.5.17. Consultar Detalles del Beneficiario

En la figura 4.40, se muestra el diagrama de colaboración del caso de uso Consultar detalles del beneficiario. En primera instancia, el usuario al seleccionar el vínculo Web en un proyecto en específico, el Sistema Web Protegido enviará la petición al objeto Acción Consultar Detalles Beneficiario, el cual interactúa entre la capa de usuario y la capa de negocio. Este objeto acción pedirá los beneficiario al objeto facade beneficiario, y se los regresará para que posteriormente sean enviados al usuario.

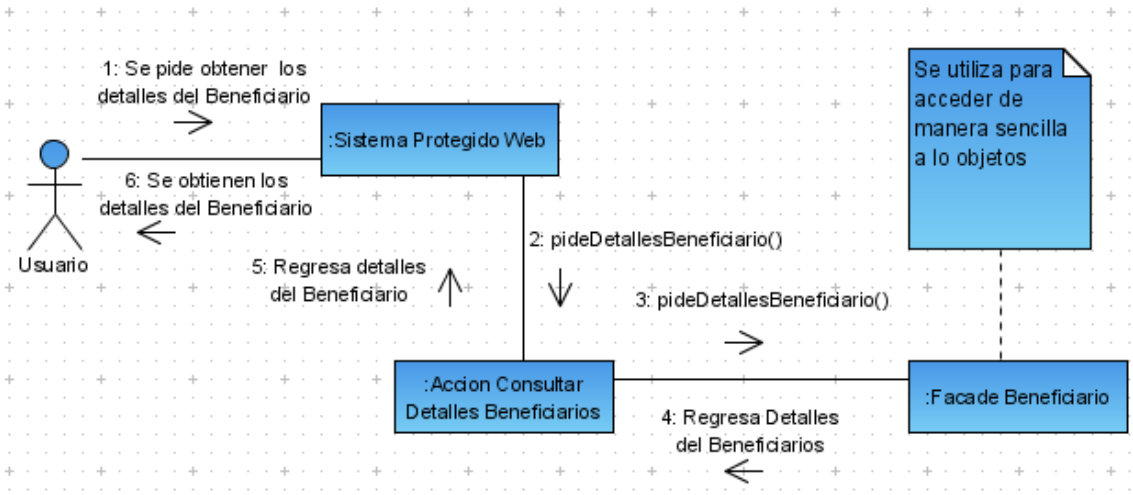


Figura 4.40

Diagrama de colaboración del caso de uso Consultar Detalles del Beneficiario

4.2.5.18. Ingresar Proyecto

En la figura 4.41, se muestra el diagrama de colaboración del caso de uso Ingresa Proyecto y la interacción entre sus objetos. En primera instancia, el usuario ingresa la información el formulario correspondiente y la envía al servidor por medio del sistema protegido web. Este último, manda la información al objeto Acción Ingresar Proyecto, el cual se encargará del flujo de información entre la capa de usuario y la de negocio. Posteriormente, el objeto Acción de encarga de llamar a dos objetos Validadores, el Validador de Datos y el Validador de Presupuesto, que de regresar resultados satisfactorios de las validaciones el objeto Acción, le enviará la información al Facade Proyecto para que se encargue de la interacción con la base de datos. Finalmente, el facade regresa la información del proyecto al usuario para confirmar que todo resultó satisfactoriamente.

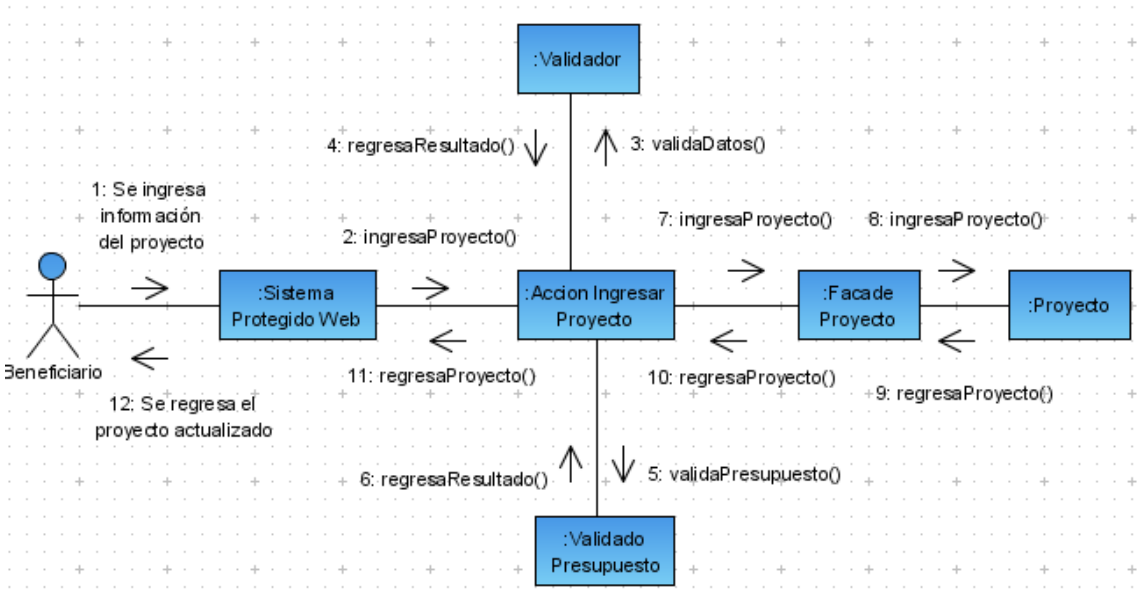


Figura 4.41

Diagrama de colaboración del caso de uso Ingresar Proyecto

4.2.5.19. Consultar Proyectos

En la figura 4.42, se muestra como para revisar los proyectos, se involucran las tres capas del sistema con sus componentes. En primera instancia, el usuario pide la información al acceder a la página de Consultar Proyectos. Después, el objeto Acción Consultar Proyectos, que es el intermediario entre la capa de usuario y la capa de negocio, maneja la información del formulario y la envía al Facade proyecto, que a su vez será el responsable de interactuar directamente con el objeto proyecto. El objeto proyecto, es el cual, realiza todas las interacciones con la base de datos, sin tener que preocuparse con los queries. Finalmente, se enviará al usuario, (que en este caso de uso es el gobierno, el delegado y el beneficiario), la lista de los delegados para mostrárselos al usuario.

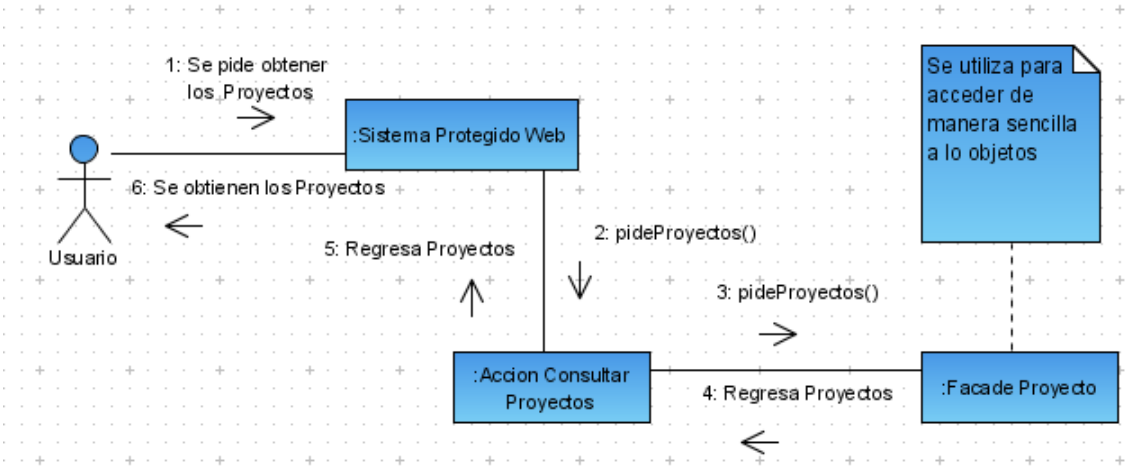


Figura 4.42

Diagrama de colaboración del caso de uso Consultar Proyectos

4.2.5.20. Consultar Detalles del Proyecto

En la figura 4.43, se muestra el diagrama de colaboración del caso de uso Consultar detalles del Proyecto. En primera instancia, el usuario al seleccionar el vínculo Web en un proyecto en específico, el sistema web protegido enviará la petición al objeto Acción Consultar Detalles Proyecto, el cual interactúa entre la capa de usuario y la capa de negocio. Este objeto Acción pedirá los proyectos al objeto Facade proyecto, y se los regresará para que posteriormente sean enviados al usuario.

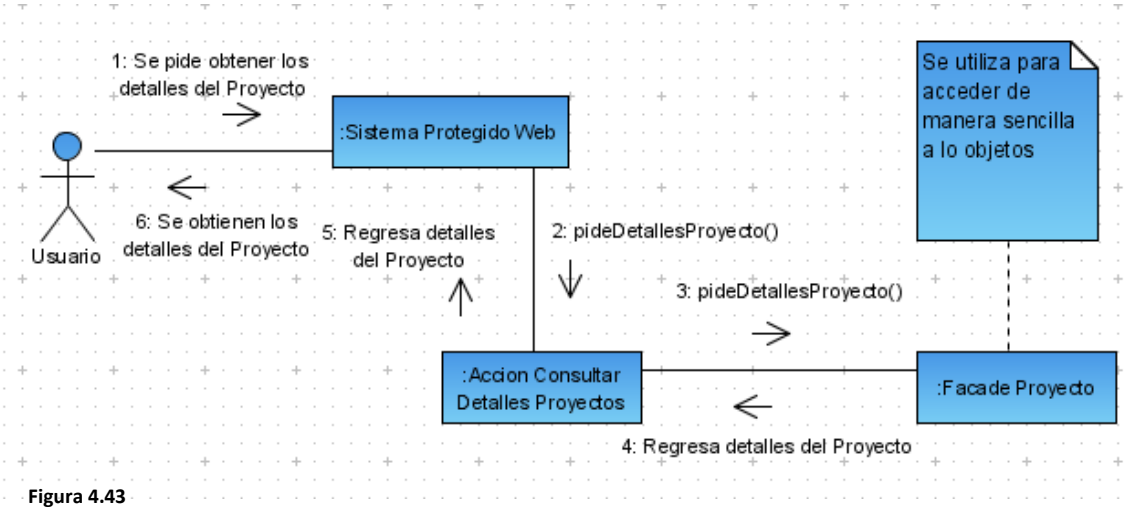


Figura 4.43

Diagrama de colaboración del caso de uso Consultar Detalles del Proyecto

A continuación se mostrarán los diagramas de secuencias del sistema, los cuales son utilizados para mostrar a los objetos y las interacciones entre ellos, pero mostrando los mensajes entre ellos en una organización temporal.
Todo esto se consultará en la sección siguiente.

4.2.6. Diagramas de Secuencia

El diagrama de secuencia es uno de los diagramas más efectivos para modelar interacción entre objetos en un sistema. Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada método de la clase. Mientras que el diagrama de casos de uso permite el modelado de una vista del negocio del escenario, el diagrama de secuencia contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes pasados entre los objetos.

Un ejemplo de ello es mostrado en la figura 4.44, el cual muestra el patrón de diseño MVC (Modelo Vista Controlador), todo esto se muestra a continuación:

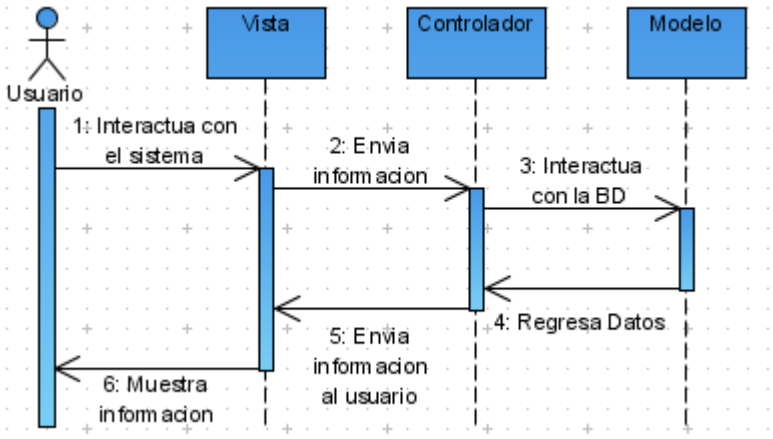


Figura 4.44

Diagrama de secuencia del paradigma MVC

En la siguiente sección se muestran los diagramas de secuencia del sistema FONOL, correspondientes a los casos de uso.

4.2.6.1. Visitar el Sitio Público

La figura 4.45, muestra como el usuario accede al sistema Público, el cual se puede acceder por internet por cualquier usuario.

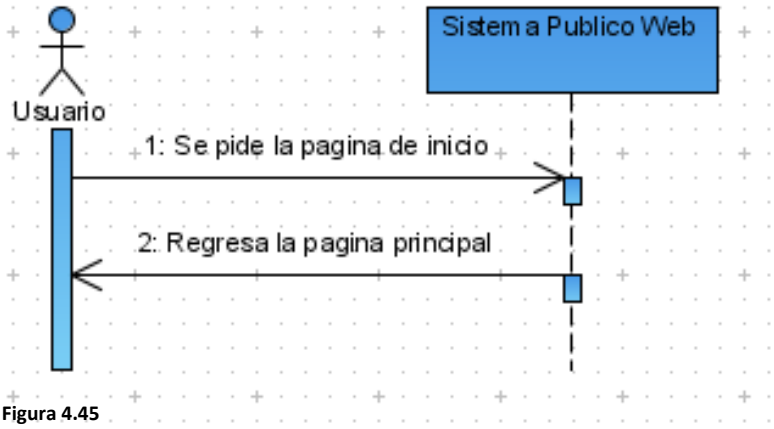


Figura 4.45

Diagrama de secuencia del caso de uso Visitar Sitio Publico

4.2.6.2. Revisar Quienes Somos

La figura 4.46, muestra como el usuario accede al sistema público y solicita la pantalla Quienes Somos, la cual se puede acceder por internet por cualquier usuario y brinda una breve historia del FONOL en el tiempo.

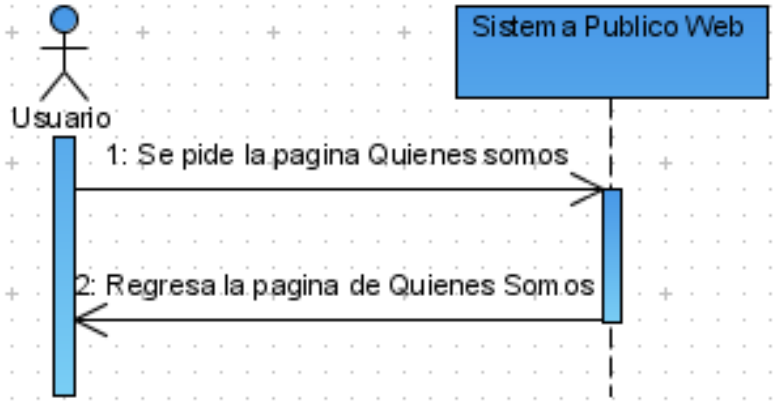


Figura 4.46

Diagrama de secuencia del caso de uso Visitar Quienes Somos

4.2.6.3. Contactar FONOL

La figura 4.47, muestra como el usuario accede al sistema público y solicita la pantalla Contáctanos, la cual se puede acceder por internet por cualquier usuario y brinda de información necesaria para comunicarse con personal del FONOL.

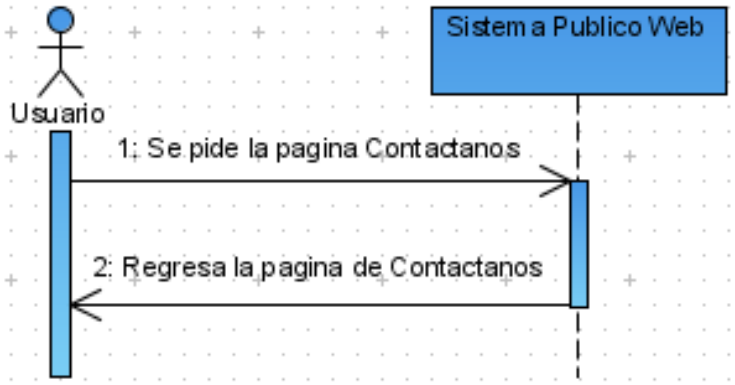


Figura 4.47

Diagrama de secuencia del caso de uso Visitar Contáctanos

4.2.6.4. Revisar Solicitudes

La figura 4.48, muestra como el usuario accede al sistema público y solicita la pantalla Revisar Solicitudes, la cual se puede acceder por internet por cualquier usuario y brinda un listado de las solicitudes que se han emitido recientemente.

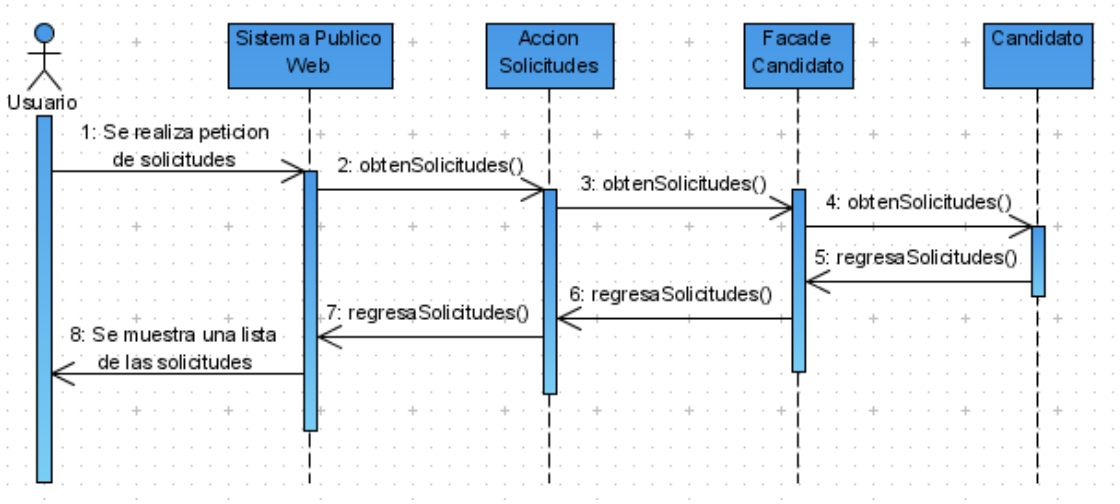


Figura 4.48

Diagrama de secuencia del caso de uso Revisar Solicitudes

4.2.6.5. Ingresar Solicitud

La figura 4.49, muestra como el usuario accede al sistema público y solicita la pantalla Ingresar Solicitud, la cual se puede acceder por internet por cualquier usuario y brinda un formulario para ingresar los datos de una organización que quiere ser parte del FONOL.

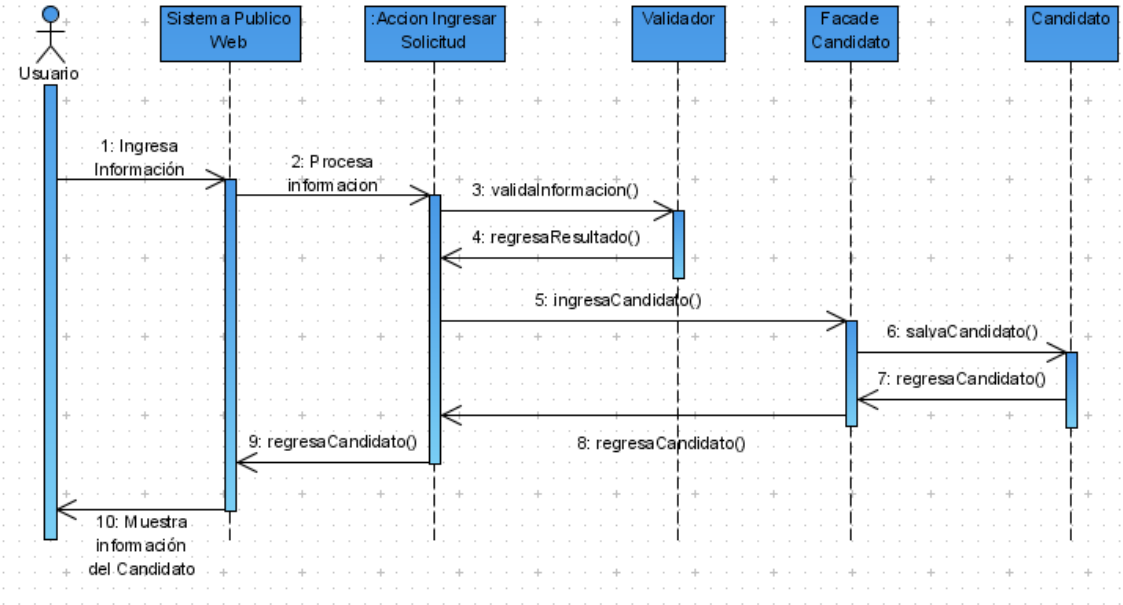


Figura 4.49

Diagrama de secuencia del caso de uso Ingresar Solicitud

4.2.6.6. Consultar Detalles del Candidato

La figura 4.50, muestra como el usuario accede al sistema público y solicita la pantalla Detalles del Candidato, la cual se puede acceder por internet por cualquier usuario y brinda de la información general de un candidato, así como, de su representante.

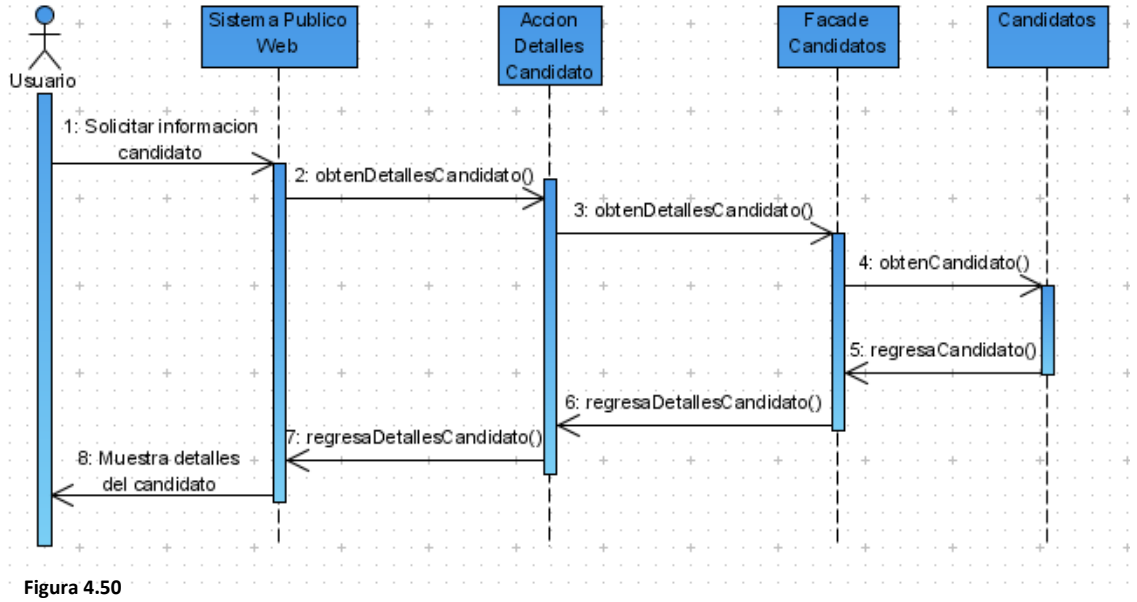


Figura 4.50

Diagrama de secuencia del caso de uso Consultar Detalles Candidato

4.2.6.7. Revisar Candidatos Aprobados

La figura 4.51, muestra como el usuario accede al sistema Público y solicita la pantalla Candidatos Aprobados, la cual se puede acceder por internet por cualquier usuario y brinda un listado de las solicitudes de los candidatos que han sido aceptados recientemente.

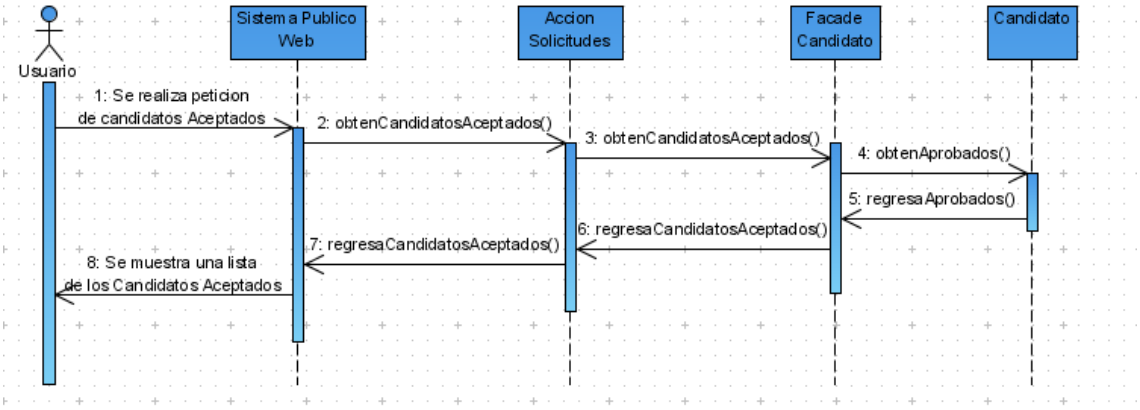


Figura 4.51

Diagrama de secuencia del caso de uso Revisar Candidatos Aprobados

4.2.6.8. Ingresar al Sistema Protegido

La figura 4.52, muestra como el usuario accede al sistema Protegido y solicita la pantalla Ingresar al sistema Protegido, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario y una contraseña autorizada, y brinda acceso al módulo administrador del sistema.

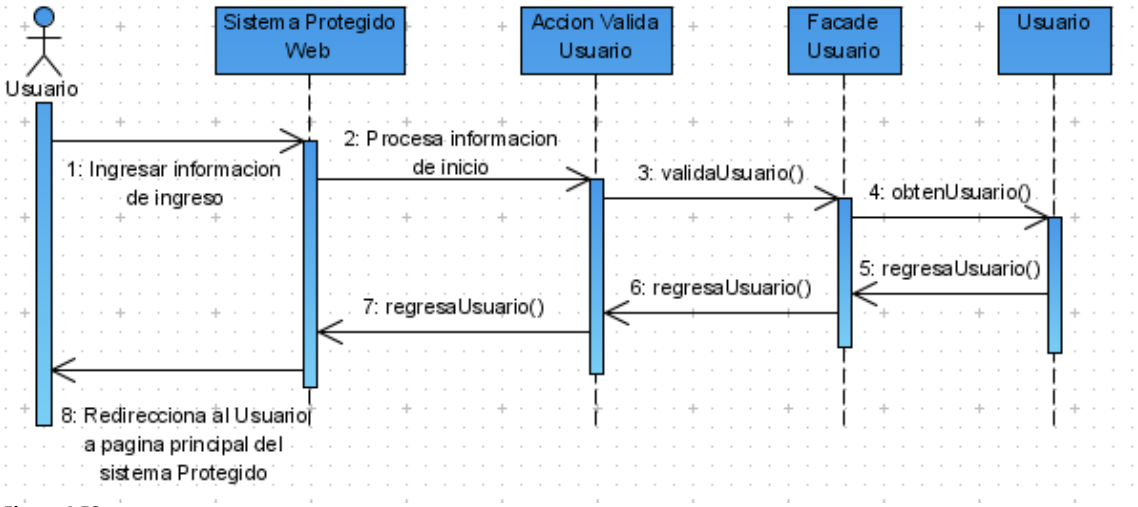


Figura 4.52

Diagrama de secuencia del caso de uso Ingresar al Sistema Protegido

4.2.6.9. Salir del Sistema

La figura 4.53, muestra como el usuario para salir del sistema Protegido solicita la pantalla Salir del sistema Protegido, la cual se puede acceder una vez que se encuentre dentro del sistema, y brinda de una forma clara y segura para salir del sistema.

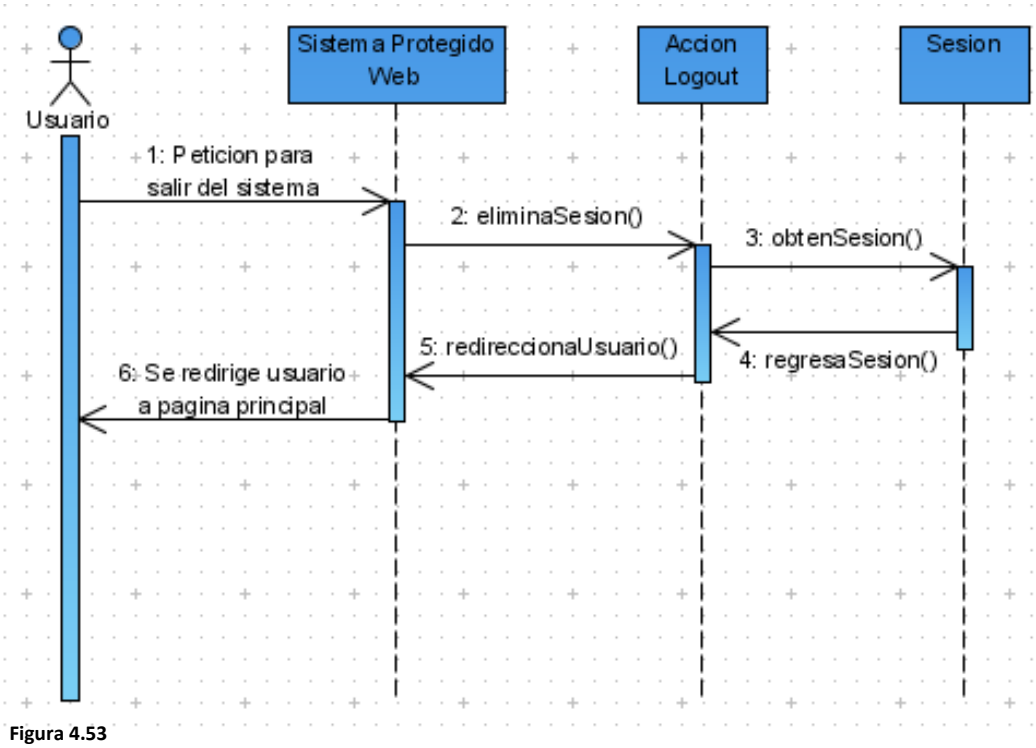


Figura 4.53

Diagrama de secuencia del caso de uso Salir del Sistema

4.2.6.10. Consultar Candidatos

La figura 4.54, muestra como el usuario accede al sistema protegido y solicita la pantalla Consultar Candidatos, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol gobierno o delegado, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para consultar las solicitudes de los candidatos que han quedado registradas recientemente.

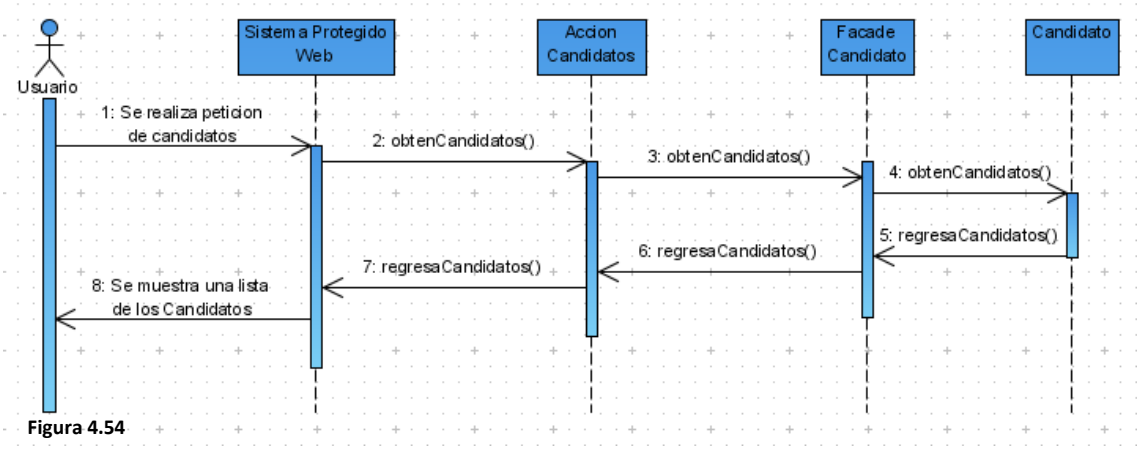


Figura 4.54

Diagrama de secuencia del caso de uso Consultar Candidatos

4.2.6.11. Evaluar Candidato

La figura 4.55, muestra como el usuario accede al sistema protegido y solicita la pantalla Evaluar Candidatos, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol delegado, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para consultar las solicitudes de los candidatos que han quedado registradas recientemente, e igualmente, para evaluar si se quedan o no dentro del FONOL.

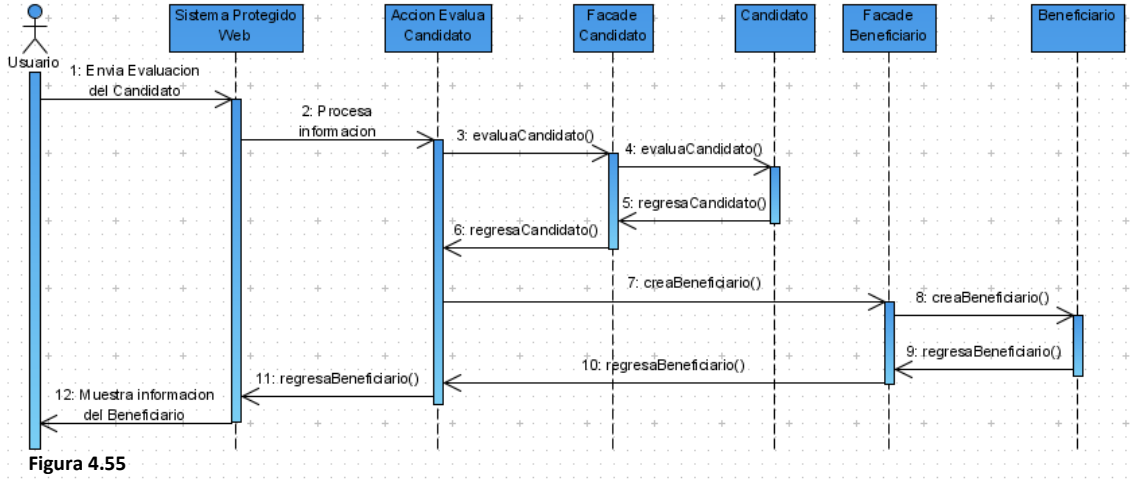


Figura 4.55

Diagrama de secuencia del caso de uso Evaluar Candidato

4.2.6.12. Consultar Delegados

La figura 4.56, muestra como el usuario accede al sistema protegido y solicita la pantalla Consultar Delegados, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol gobierno, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para consultar a los delegados que se encuentren como responsables del FONOL dentro de cada estado.

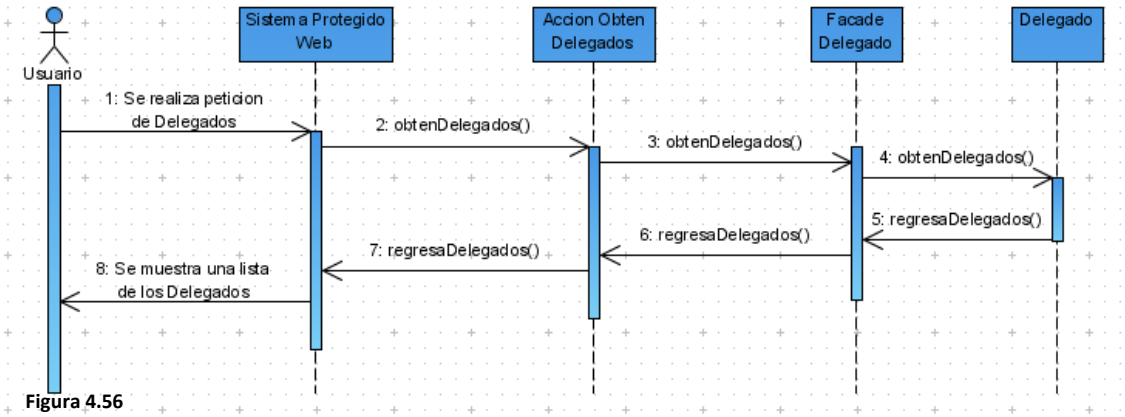


Figura 4.56

Diagrama de secuencia del caso de uso Consultar Delegados

4.2.6.13. Agregar Delegado

La figura 4.57, muestra como el usuario accede al sistema protegido y solicita la pantalla Agregar Delegado, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol gobierno, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para agregar un delegado en un estado en específico.

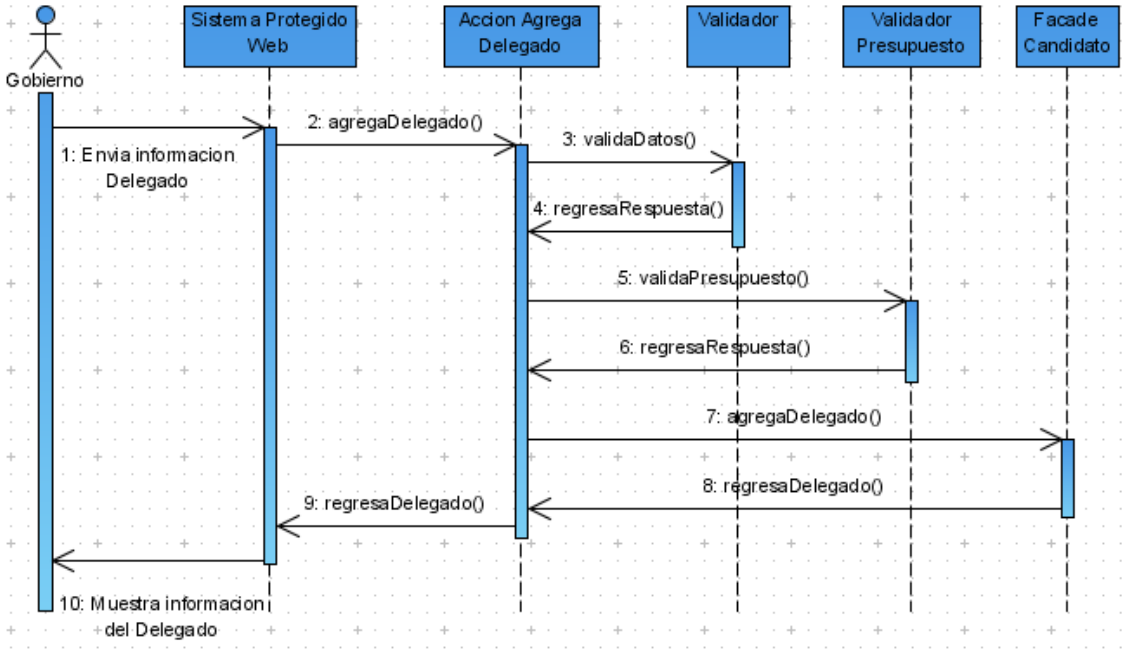


Figura 4.57

Diagrama de secuencia del caso de uso Agregar Delegado

4.2.6.14. Modificar Delegado

La figura 4.58, muestra como el usuario accede al sistema protegido y solicita la pantalla Modificar Delegado, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol gobierno, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para modificar un delegado en un estado existente.

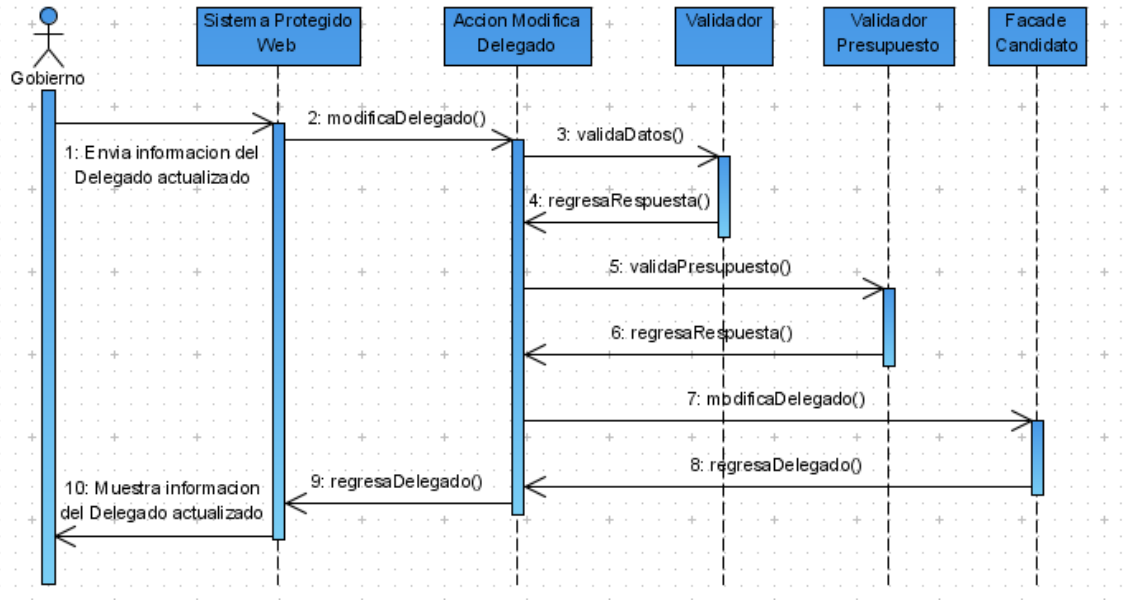


Figura 4.58

Diagrama de secuencia del caso de uso Modificar Delegado

4.2.6.15. Consultar Beneficiarios

La figura 4.59, muestra como el usuario accede al sistema protegido y solicita la pantalla Consultar Beneficiarios, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol gobierno o rol delegado, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para consultar a los beneficiarios que se encuentren dentro de un estado.

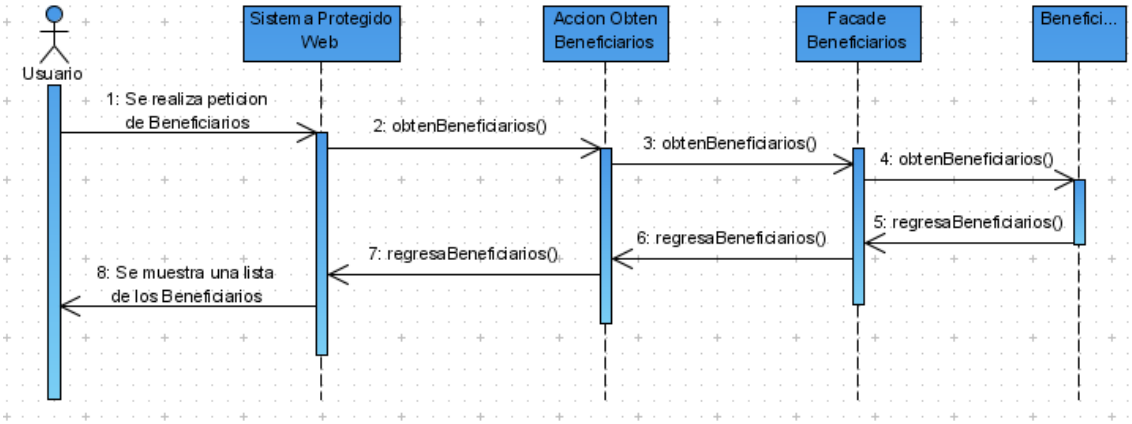


Figura 4.59

Diagrama de secuencia del caso de uso Consultar Beneficiarios

4.2.6.16. Modificar Beneficiarios

La figura 4.60, muestra como el usuario accede al sistema protegido y solicita la pantalla Modificar Beneficiarios, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol delegado, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para modificar a los beneficiarios que se encuentren dentro de un estado.

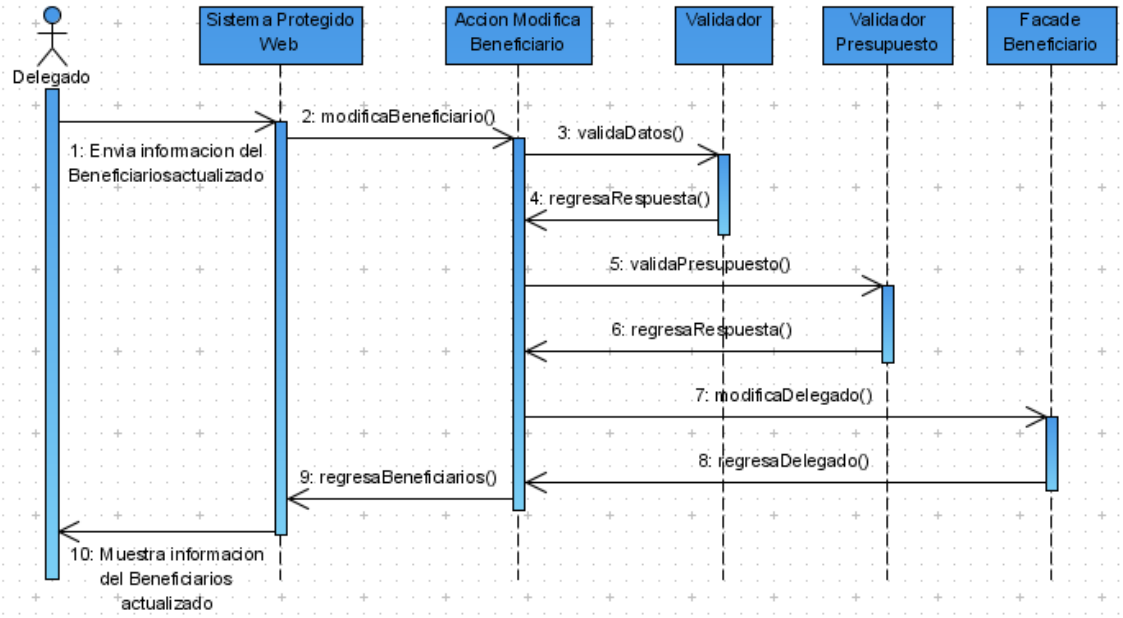


Figura 4.60

Diagrama de secuencia del caso de uso Modificar Beneficiario

4.2.6.17. Consultar Detalles del Beneficiario

La figura 4.61, muestra como el usuario accede al sistema protegido y solicita la pantalla Consultar Detalles del beneficiario, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol delegado o rol gobierno, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para consultar los detalles de un beneficiario de interés.

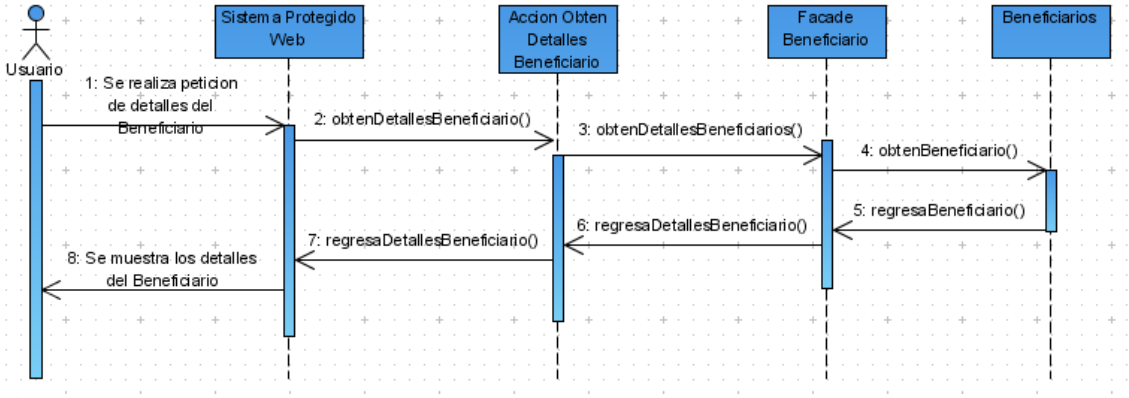


Figura 4.61

Diagrama de secuencia del caso de uso Consultar Detalles del Beneficiario

4.2.6.18. Ingresar Proyecto

La figura 4.62, muestra como el usuario accede al sistema protegido y solicita la pantalla Ingresar proyecto, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol beneficiario, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para ingresar un proyecto de interés de su organización.

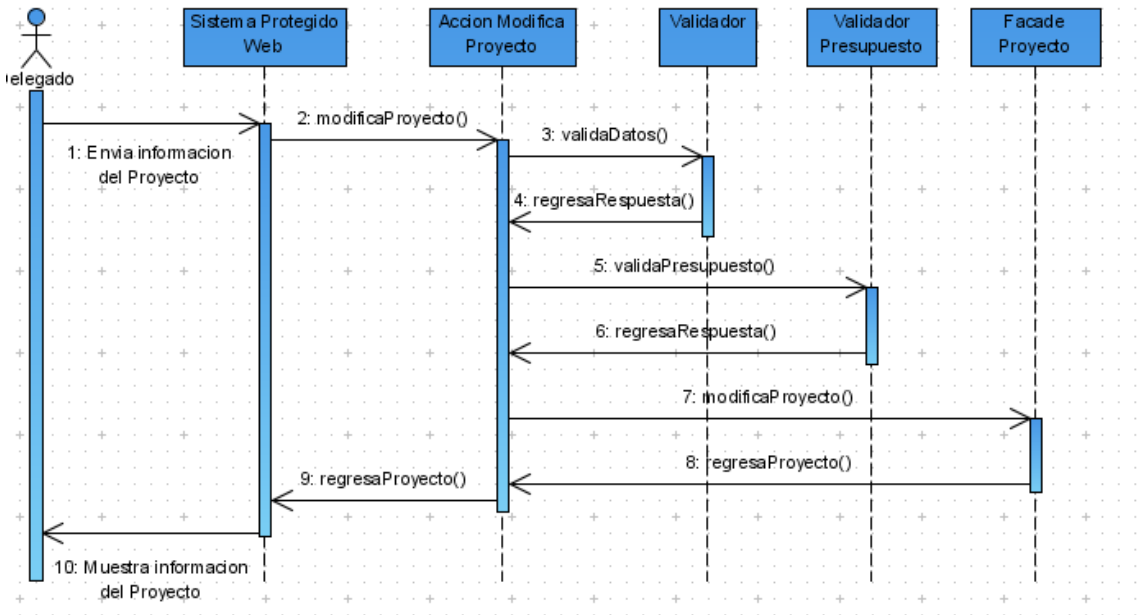


Figura 4.62

Diagrama de secuencia del caso de uso Ingresar Proyecto

4.2.6.19. Consultar Proyectos

La figura 4.63, muestra como el usuario accede al sistema protegido y solicita la pantalla Consultar Proyectos, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol beneficiario, rol delegado o rol gobierno, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para consultar la información un proyecto de interés dentro de un estado.

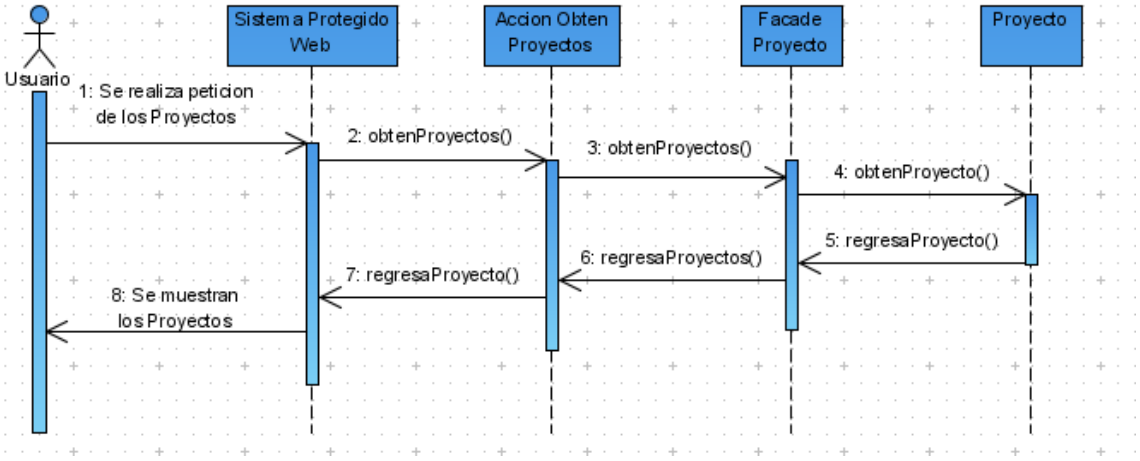


Figura 4.63

En la figura, se muestra el diagrama de secuencia del caso de uso Consultar Proyectos

4.2.6.20. Consultar Detalles del Proyecto

La figura 4.64, muestra como el usuario accede al sistema Protegido y solicita la pantalla Consultar Detalles del Proyecto, la cual se puede acceder por internet por cualquier usuario que cuente con un usuario con rol beneficiario, rol delegado o rol gobierno, además de una contraseña autorizada, y brinda acceso al módulo administrador del sistema para consultar la información extendida de un proyecto de interés dentro de un estado.

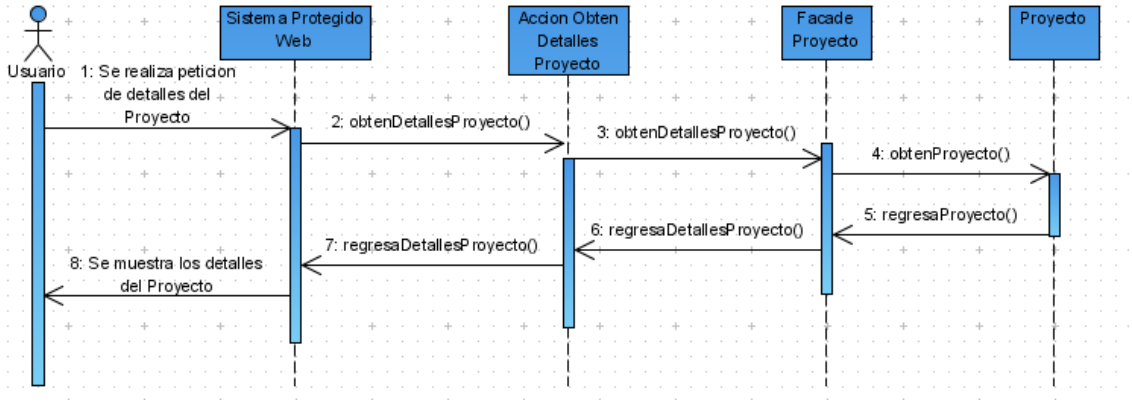


Figura 4.64

Diagrama de secuencia del caso de uso Consultar Detalles del Proyecto

A continuación se mostrará el diagrama de actividades del sistema FONOL, el propósito del diagrama de actividad es modelar un proceso de flujo de trabajo y/o modelar operaciones. Una operación es un servicio proporcionado por un objeto, que está disponible a través de una interfaz. Una Interfaz es un grupo de operaciones relacionadas con la semántica.

4.2.7. Diagrama de Actividades

Un diagrama de actividades representa los flujos de trabajo paso a paso de negocio y operacionales de los componentes en un sistema. Un diagrama de actividades muestra el flujo de control general, en la figura 4.65, se muestra el diagrama de actividades del sistema FONOL en su modulo de Sistema Protegido, ya que es donde se realizan las funciones más importantes:

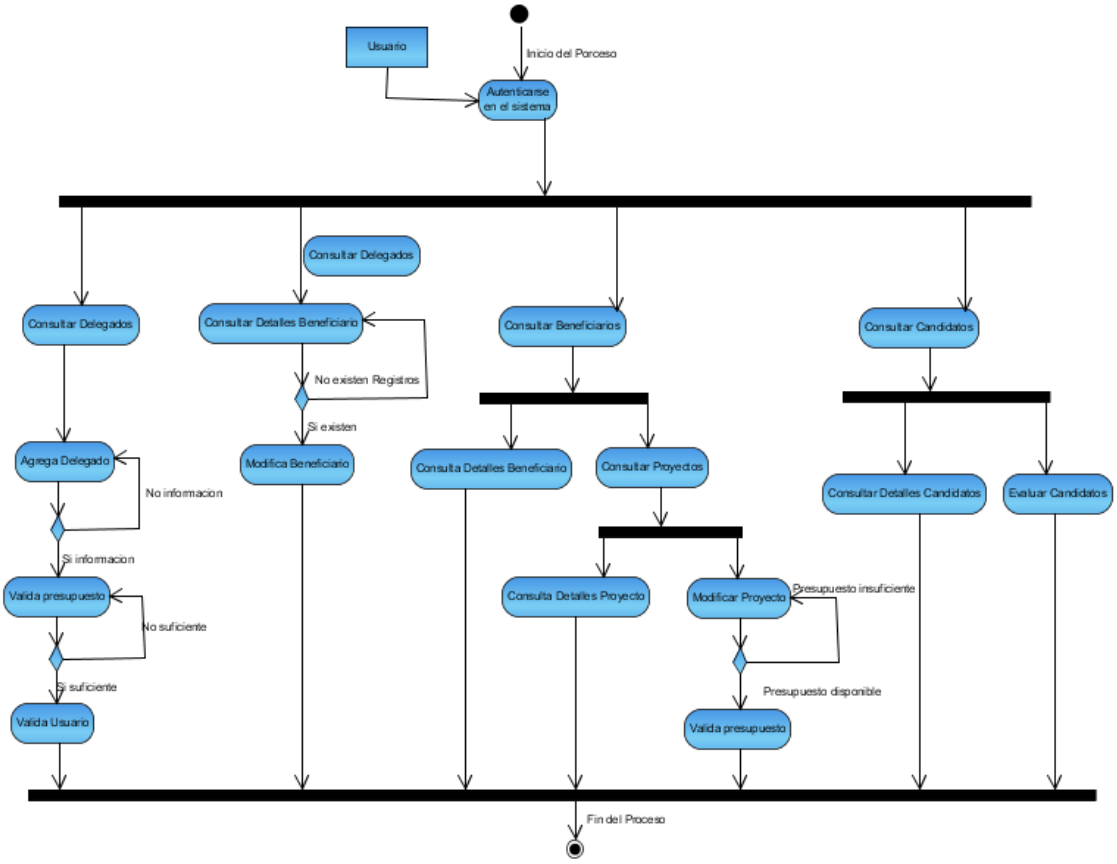


Figura 4.65

Diagrama de actividades del Sistema FONOL

4.2.8. Diagrama Entidad-Relación

El Modelo Entidad-Relación una herramienta para el modelado de datos de un sistema de información. Estos modelos expresan entidades relevantes para un sistema de información, sus inter-relaciones y propiedades.

En la figura 4.66, se mostrarán a los actores mencionados en la sección 4.1.1, la cual enlista los actores del sistema, y en esta sección de muestran integrados al modelo ER, el cual se mostrará a continuación:

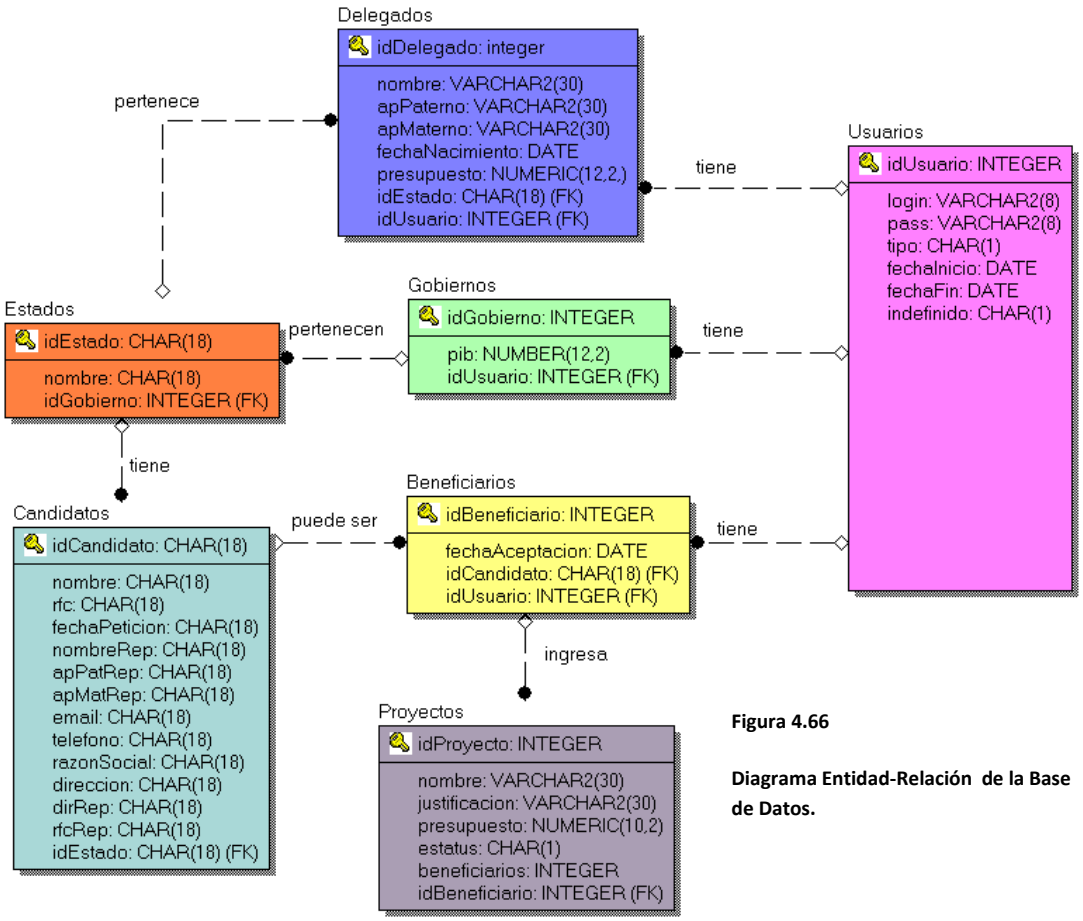


Figura 4.66

Diagrama Entidad-Relación de la Base de Datos.

Del diagrama se pueden extraer los objetos y las relaciones entre ellos. Es muy útil para modelar los datos en la base de datos y para implementarlos en el sistema.

En la siguiente sección, se muestra el diccionario de datos, el cual nos muestra la definición de los datos de las Entidades del sistema.

4.2.9. Diccionario de Datos

El diccionario de datos que se visualizará a continuación, incluirá todos los campos, los índices y las llaves foráneas.

La estrategia que se utilizará para automatizar la generación de las llaves primarias de las tablas del sistema será crear secuencias. Y se creará una secuencia por cada tabla del sistema.

A continuación, se mostrará la definición de cada una de las tablas del sistema FONOL:

4.2.9.1. Estados

La tabla Estados representa a cada una de las entidades federativas de la República Mexicana, actúa como un catalogo y una referencia al gobierno actual.

Campos

Campo	Dato	PK	FK	NN	Descripción
idestado	int4	✓		✓	Identificador único
nombre	varchar(255)				Nombre del estado
idgobierno	int4		✓		Referencia al gobierno actual

Índices

Index	Campos	Primary	Unique
estados_pkey	idestado	✓	

Llaves foráneas

Llave foránea	Campos	Ref Table	Ref Campos
fk_estados_idgobierno	idgobierno	gobiernos	idgobierno

Referencias

Llave foránea
fk_candidatos_idestado
fk_delegados_idestado

4.2.9.2. Usuarios

La tabla Usuarios representa cada uno de los usuarios que pueden ingresar al sistema protegido del FONOL. En ella se puede observar que el usuario debe estar en un periodo válido para poder ingresar en el sistema, o en su defecto, debe tener la bandera de Indefinido activado.

Campos

Campo	Dato	PK	FK	NN	Descripción
idusuario	int4	✓		✓	Identificador único del registro
fechainicio	timestamp				La fecha de inicio valido de acceso
fechafin	date				La fecha de caducidad de acceso
login	varchar(9)				El nombre de usuario
indefinido	int4				Bandera que indica si es indefinido su acceso
tipo	int4				Tipo de usuario
pass	varchar(9)				La contraseña de acceso

Indices

Index	Campos	Primary	Unique
usuarios_pkey	idusuario	✓	

Llaves foráneas

No hay llaves foráneas

Referencias

Llave foránea
fk_beneficiarios_idusuario
fk_delegados_idusuario
fk_gobiernos_idusuario
fk_historialpantallas_idusuario

4.2.9.3. Gobiernos

La tabla Gobierno representa al gobierno actual. En ella se puede observar que contiene el presupuesto nacional, el cual, representa todo el dinero que le ha sido asignado al FONOL para repartir entre sus entidades. El gobierno actual se encarga de agregar y modificar a los delegados y el presupuesto que se le asignará a cada uno de los estados.

Campos

Campo	Dato	PK	FK	NN	Descripción

idgobierno	int4	✓		✓	Identificador único del registro
pib	numeric(38)				El presupuesto total nacional
idusuario	int4		✓		Referencia a un usuario de acceso

Indices

Index	Campos	Primary	Unique
gobiernos_pkey	idgobierno	✓	

Llaves foráneas

Llave foránea	Campos	Ref Table	Ref Campos
fk_gobiernos_idusuario	idusuario	public.usuarios	idusuario

Referencias

Llave foránea
fk_estados_idgobierno

4.2.9.4. Delegados

La tabla Delegados, representa a cada uno de los encargados de cada estado, el cual es responsable de evaluar y agregar tanto a candidatos como a los proyectos ingresados por los beneficiarios. También, contiene el presupuesto disponible para el estado, así como los datos personales del delegado.

Campos

Campo	Dato	PK	FK	NN	Descrpción
iddelegado	int4	✓		✓	Identificador único del registro
apmaterno	varchar(255)				Apellido materno del delegado
fechanacimiento	date				Fecha de nacimiento del delegado
nombre	varchar(255)				Nombre del delegado
presupuesto	numeric(38)				Presupuesto disponible para el estado
appaterno	varchar(255)				Apellido paterno del delegado
idusuario	int4		✓		Referencia a un usuario
idestado	int4		✓		Referencia a un estado

Indices

Index	Campos	Primary	Unique
delegados_pkey	iddelegado	✓	

Llaves foráneas

Llave foránea	Campos	Ref Table	Ref Campos
fk_delegados_idestado	idestado	public.estados	idestado
fk_delegados_idusuario	idusuario	public.usuarios	idusuario

4.2.9.5. Candidatos

La tabla Candidatos, representa a cada uno de los candidatos que ingresaron una solicitud para ser parte del FONOL. En ella, se ingresan los datos de la solicitud, así como los datos de la organización candidata y los datos del representante.

Campos

Campo	Dato	PK	FK	NN	Descripción
idcandidato	int4	✓		✓	Identificador único de registro
email	varchar(255)				Correo electrónico de la organización
telefono	varchar(255)				Teléfono de la organización
nombre	varchar(255)				Nombre de la organización
razonsocial	varchar(255)				RFC de la organización
fechapeticion	date				Fecha de petición de la organización
direccion	varchar(255)				Dirección de la organización
appatrep	varchar(255)				Apellido paterno del representante
dirrep	varchar(255)				Dirección del representante
rfcrep	varchar(255)				RFC del representante
nombrerep	varchar(255)				Nombre del representante
estatus	int4				Estatus del representante
rfc	varchar(255)				RFC de la organización
apmatrep	varchar(255)				Apellido materno del representante
idestado	int4		✓		Referencia a un estado

Indices

Index	Campos	Primary	Unique
candidatos_pkey	idcandidato	✓	

Llaves foráneas

Llave foránea	Campos	Ref Table	Ref Campos
fk_candidatos_idestado	idestado	public.estados	idestado

Referencias

Llave foránea
fk_beneficiarios_idcandidato

4.2.9.6. Beneficiarios

La tabla Beneficiarios, se encarga de almacenar los beneficiarios que ya son parte del FONOL, tiene una referencia al candidato del que proviene, ya que, al tener esta referencia, se permite obtener toda la información de la organización y de su representante.

Campos

Campo	Dato	PK	FK	NN	Descripción
idbeneficiario	int4	✓		✓	Identificador único del registro
fechaaceptacion	date				Fecha de aceptación del beneficiario
idcandidato	int4		✓		Referencia a un candidato
idusuario	int4		✓		Referencia a un usuario

Indices

Index	Campos	Primary	Unique
beneficiarios_pkey	idbeneficiario	✓	

Llaves foráneas

Llave foránea	Campos	Ref Table	Ref Campos
fk_beneficiarios_idcandidato	idcandidato	public.candidatos	idcandidato
fk_beneficiarios_idusuario	idusuario	public.usuarios	idusuario

Referencias

Llave foránea
fk_proyectos_idbeneficiario

4.2.9.7. Proyectos

En la tabla Proyectos, se almacenan todos los proyectos ingresados por el beneficiario. Esta tabla contiene el monto requerido para llevar a cabo el proyecto, así como el estatus requerido para que el beneficiario lo acepte o rechace.

Campos

Campo	Dato	PK	FK	NN	Descripción
idproyecto	int4	✓		✓	Identificador único del proyecto
presupuesto	numeric(38)				Presupuesto requerido para proyecto
comentario	varchar(255)				Comentario del delegado
nombre	varchar(255)				Nombre del proyecto
estatus	int4				Estatus del proyecto
beneficiarios	int4				Numero de beneficiados por el proyecto
justificacion	varchar(255)				Justificación del proyecto
idbeneficiario	int4		✓		Referencia a su beneficiario

Indices

Index	Campos	Primary	Unique
proyectos_pkey	idproyecto	✓	

Llaves foráneas

Llave foránea	Campos	Ref Table	Ref Campos

fk_proyectos_idbeneficiario	idbeneficiario	public.beneficiarios	idbeneficiario

Ya completado el diseño, el paso siguiente es implementar todos los casos de uso del análisis y el diseño. En la siguiente sección, se describirá la etapa de implementación enfocada a RUP.

4.3. Implementación

En la implementación, las clases y subsistemas encontrados durante el diseño, son implementados como componentes de archivos que contienen código fuente, los cuales posteriormente se prueban individualmente y se integran compilándolos y enlazándolos en uno o más ejecutables.
Para realizar esta tarea se hará uso del modelo de implementación.

A continuación se darán todos los elementos necesarios para crear el modelo de implementación, comenzando con su arquitectura.

4.3.1. Arquitectura JEE aplicada al FONOL

Para crear una arquitectura robusta, se eligió un patrón de diseño que resuelve las necesidades del sistema: el Modelo-Vista-Controlador. Este patrón de diseño separa los datos de una aplicación, la interfaz de usuario, y la lógica de negocio en tres componentes distintos. El modelo, es el encargado de la interacción con los datos que residen en la base de datos; la Vista, la cual brinda la interfaz de usuario para la entrada del usuario y mostrar los datos del servidor al usuario; y finalmente, el Controlador, el cual se encarga de la interacción entre la Vista y el Modelo.
Para implementar el FONOL y aplicarlo al patrón de diseño MVC, se creará una aplicación Empresarial Java JEE, la cual constará de dos módulos: uno Web y uno EJB. Por una parte, el modulo Web constará de toda la Interfaz Grafica de Usuario, que actuará como la Vista, con tecnología JSP; y el Controlador del flujo de información, que contará con tecnología Servlet. Por otra parte, el modulo EJB se encargará de realizar toda la lógica negocio, y lo hará con tecnología EJB 3.

Ya definida la arquitectura del FONOL, se crearán los componentes requeridos para implementar los casos de uso. Esto se consultará en la siguiente sección.

4.3.2. Componentes

En esta sección, se mostrarán los distintos componentes que se utilizarán para construir todo el sistema FONOL, que en su totalidad, corresponden a componentes de Java EE, los cuales serán descritos a continuación:

4.3.2.1. JSP

Los componentes JSP (Java Server Pages), son una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo.
Las JSP's permiten la utilización de código Java mediante scripts. Además es posible utilizar algunas acciones JSP predefinidas mediante etiquetas. Estas etiquetas pueden ser enriquecidas mediante la utilización de Librerías de Etiquetas (TagLibs o Tag Libraries) externas e incluso personalizadas.
En el caso del sistema FONOL, estos componentes representan la Vista del patrón de diseño MVC.

4.3.2.2. Servlets

Un servlet es un objeto que se ejecuta en un servidor o contenedor JEE, especialmente diseñado para ofrecer contenido dinámico desde un servidor web, generalmente HTML. Forman parte de JEE (Java Enterprise Edition), que es una ampliación de JSE (Java Standard Edition).
Entre el servidor de aplicaciones y el servlet existe un contrato que determina cómo han de interactuar. La especificación de éste se encuentra en los JSR (Especificación Sun de Java) de Sun Microsystems.
En el caso del sistema FONOL, estos componentes representan al Controlador del patrón de diseño MVC.

4.3.2.3. EJB

Los Enterprise JavaBeans (también conocidos por sus siglas EJB) son una de las API que forman parte del estándar de construcción de aplicaciones empresariales J2EE de Sun Microsystems (ahora JEE 5.0). Su especificación detalla cómo los servidores de aplicaciones proveen objetos desde el lado del servidor, tales funciones son:

- Comunicación remota utilizando CORBA
- Transacciones
- Control de la concurrencia
- Eventos utilizando JMS (Java messaging service)
- Servicios de nombres y de directorio
- Seguridad
- Ubicación de componentes en un servidor de aplicaciones.

En el caso del sistema FONOL, estos componentes representan la lógica del negocio que interactúa directamente con el Modelo del patrón de diseño MVC.

4.3.2.4. JPA Entidades

Las entidades JPA, son componentes simples, representados por Clases Java Simples, las cuales no dependen de la lógica y del comportamiento de otras clases, aunque se relacionan con otras. Representan un objeto almacenado en la base de datos y permiten realizar las operaciones básicas de manera sencilla.
En el caso del sistema FONOL, estos componentes representan el Modelo del patrón de diseño MVC.

4.3.2.5. Clases Utilidad

Este tipo de componentes son importantes para brindar una funcionalidad genérica y desacoplada de la lógica de negocio. Asimismo, proveen un servicio independiente y reutilizable, que puede ser utilizado por otra aplicación.

4.3.2.6. Facades

Provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema. Será útil ya que me permitirá realizar operaciones simples con las entidades de la Base de Datos.

4.3.2.7. Archivos de Configuración

Los archivos de configuración no son propiamente componentes, pero para propósitos prácticos si lo son, ya que en el sistema FONOL se apoya de ellos para realizar toda su funcionalidad y el flujo de la información.
En el caso del sistema FONOL, estos componentes no representan ninguna parte del patrón de diseño MVC.

Ya explicados los diferentes tipos de componentes, en la siguiente sección se mostrarán los casos de uso más relevantes, pero unidos en diagramas de componentes.

4.3.3. Diagramas de Componentes

4.3.3.1. Solicitudes

La figura 4.67, muestra los componentes que interactúan en todos los procesos de ingresar y modificar solicitudes del sistema, del cual, se pueden señalar las capas en las cuales está separada la aplicación FONOL: la capa cliente, la capa web, la capa negocio, la capa integración y la capa recursos empresariales.

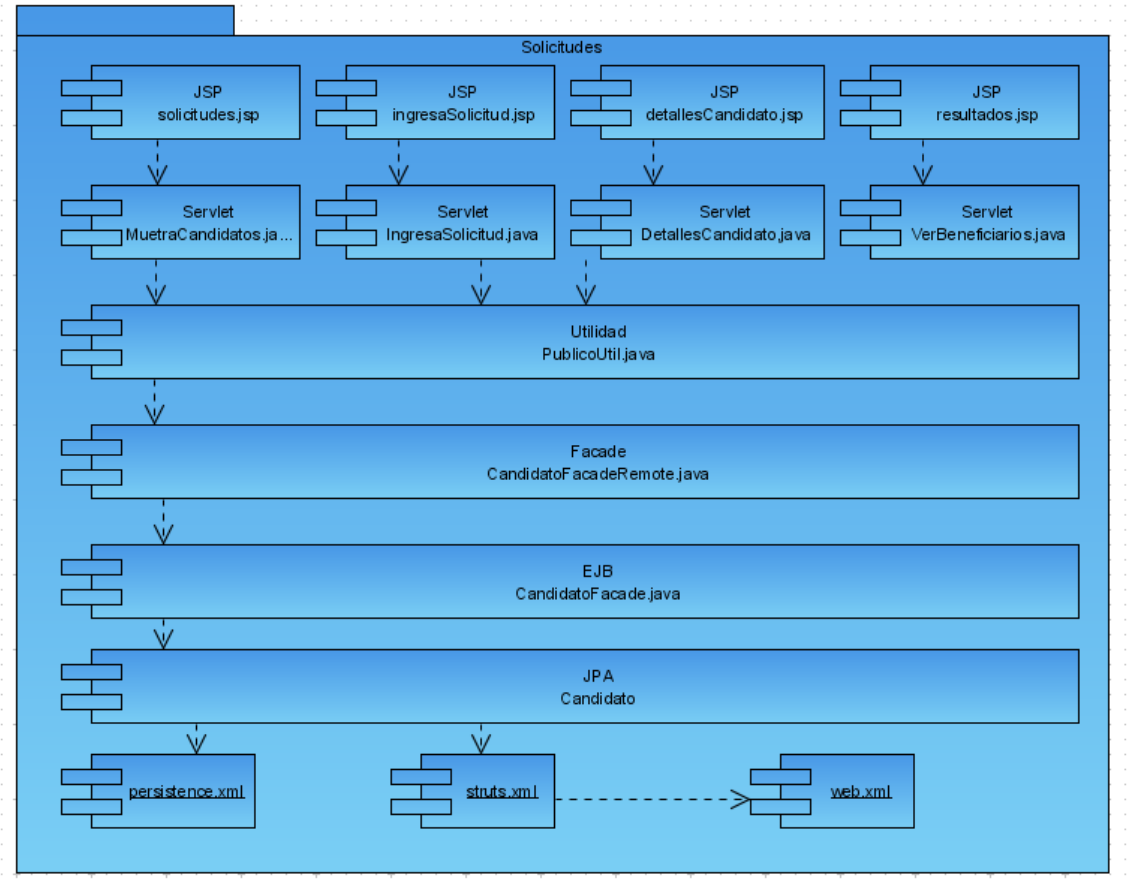


Figura 4.67

Diagrama de Componentes de lo referente a las solicitudes.

4.3.3.2. Ingresar y Salir del Sistema Protegido

La figura 4.68, muestra los componentes que interactúan en todos los procesos de ingresar y salir del sistema, resaltando la arquitectura que mantiene, además de que se unen los archivos de configuración que forman parte del diagrama.

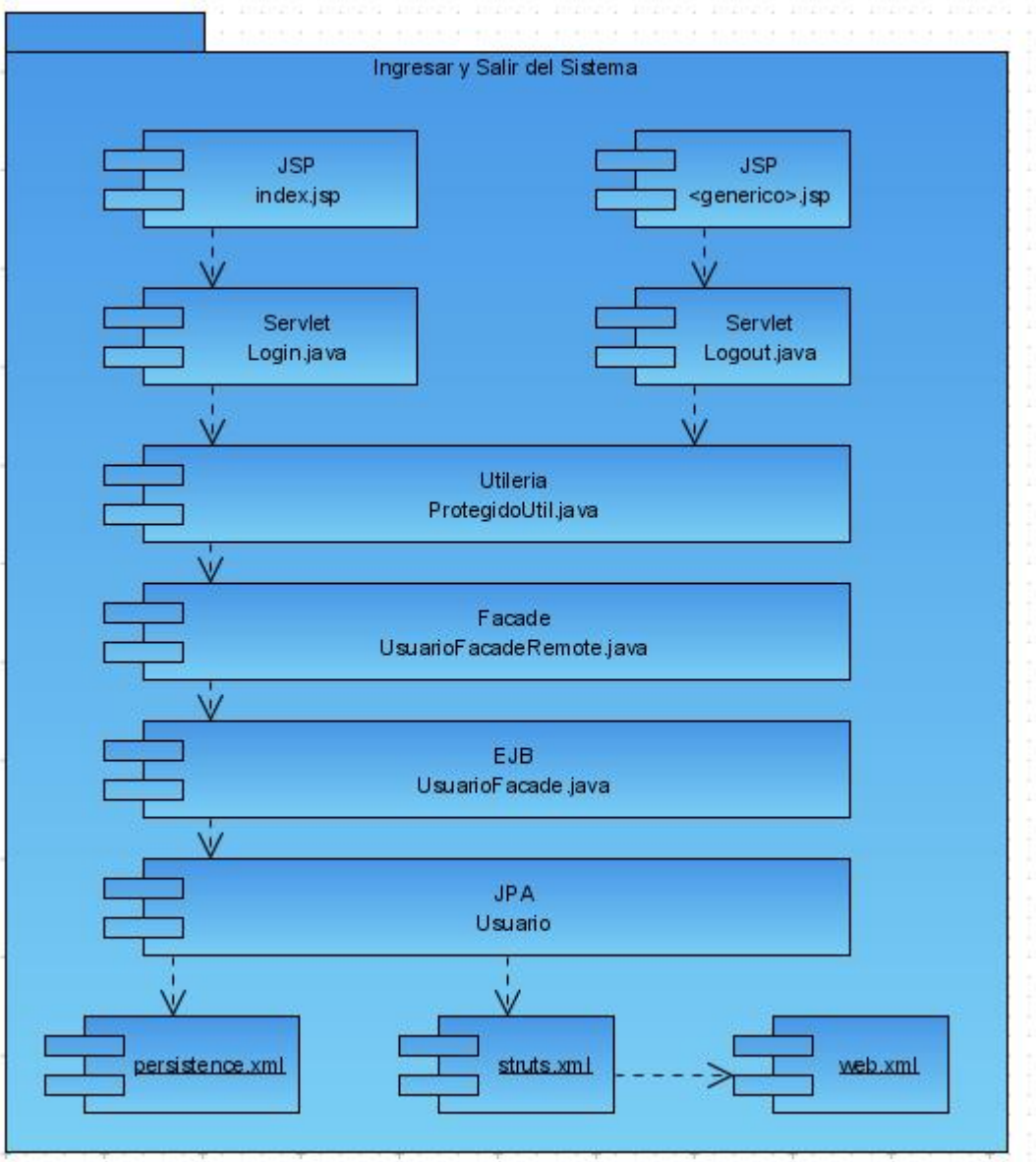


Figura 4.68

Diagrama de Componentes de lo referente a ingresar y salir del sistema.

4.3.3.3. Consultar Información de los Actores

La figura 4.69, muestra los componentes que interactúan en todos los procesos de consultar información de los actores del sistema. En la capa web se encuentran los JSP, en la capa web se encuentran los Servlet, en la capa de negocio se encuentran los EJB, en la capa de integración se encuentran las entidades Entity y finalmente en la capa de recursos se encuentran las base de datos.

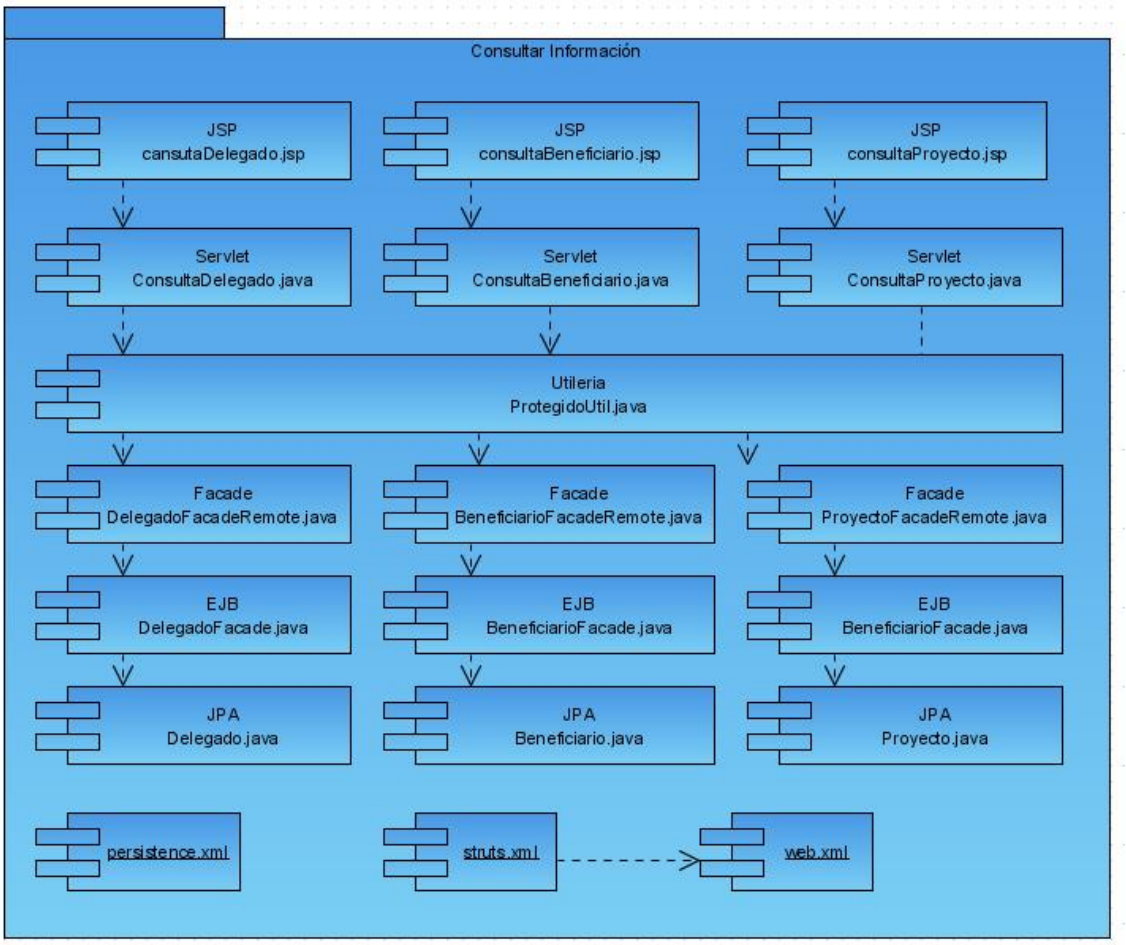


Figura 4.69

Diagrama de Componentes de lo referente a consultar información de actores del sistema.

4.3.3.4. Evaluar Candidato

La figura 4.70, muestra los componentes que interactúan en todos los procesos del caso de Uso Evaluar Candidato, de los cuales, se pueden señalar las mismas capas nuestra arquitectura, conservando las mismas capas de componentes.

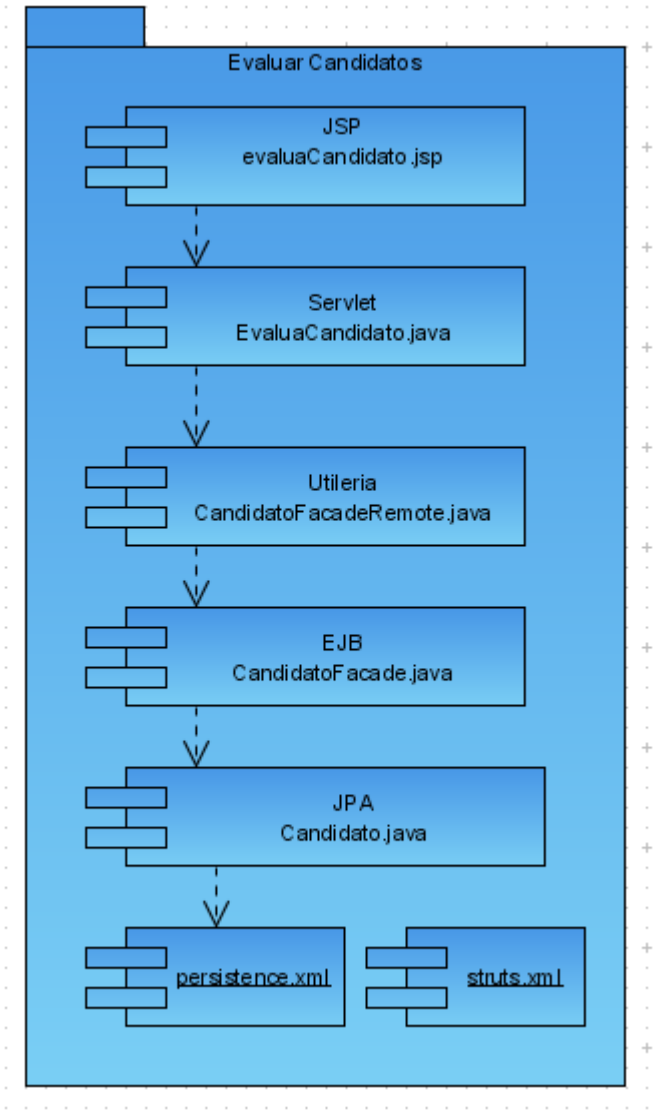


Figura 4.70

Diagrama de Componentes de lo referente a evaluar candidatos.

4.3.3.5. Agregar y Modificar Delegado

La figura 4.70, muestra al caso de Uso Agregar y Modificar Delegado, el cual sólo puede ser realizado por el rol gobierno. El diagrama muestra la implementación del caso de uso, así como los componentes utilizados. En este diagrama también se hace uso de clases validadoras como utilidad dentro de la arquitectura.

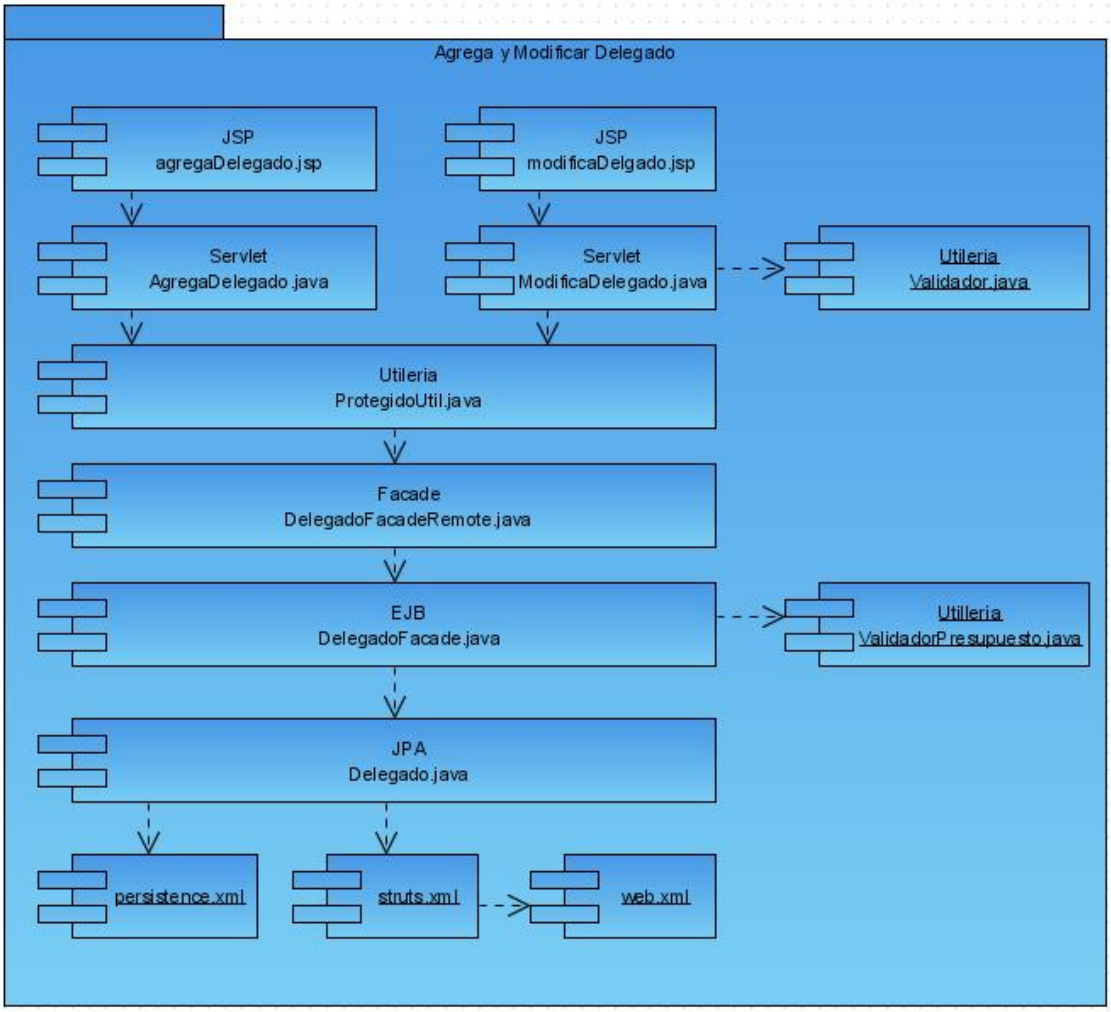


Figura 4.70

Diagrama de Componentes de lo referente a agregar y modificar delegados.

4.3.3.6. Ingresar Proyecto

La figura 4.71, muestra lo requerido a nivel de componentes del caso de Uso Ingresar Proyecto. Aquí se toma en cuenta el rol que ejecuta la acción, ya que solo puede realizar este caso de uso el beneficiario. El diagrama muestra las mismas capas de la arquitectura resaltando los componentes de utilería, que en este caso son para validar los datos.

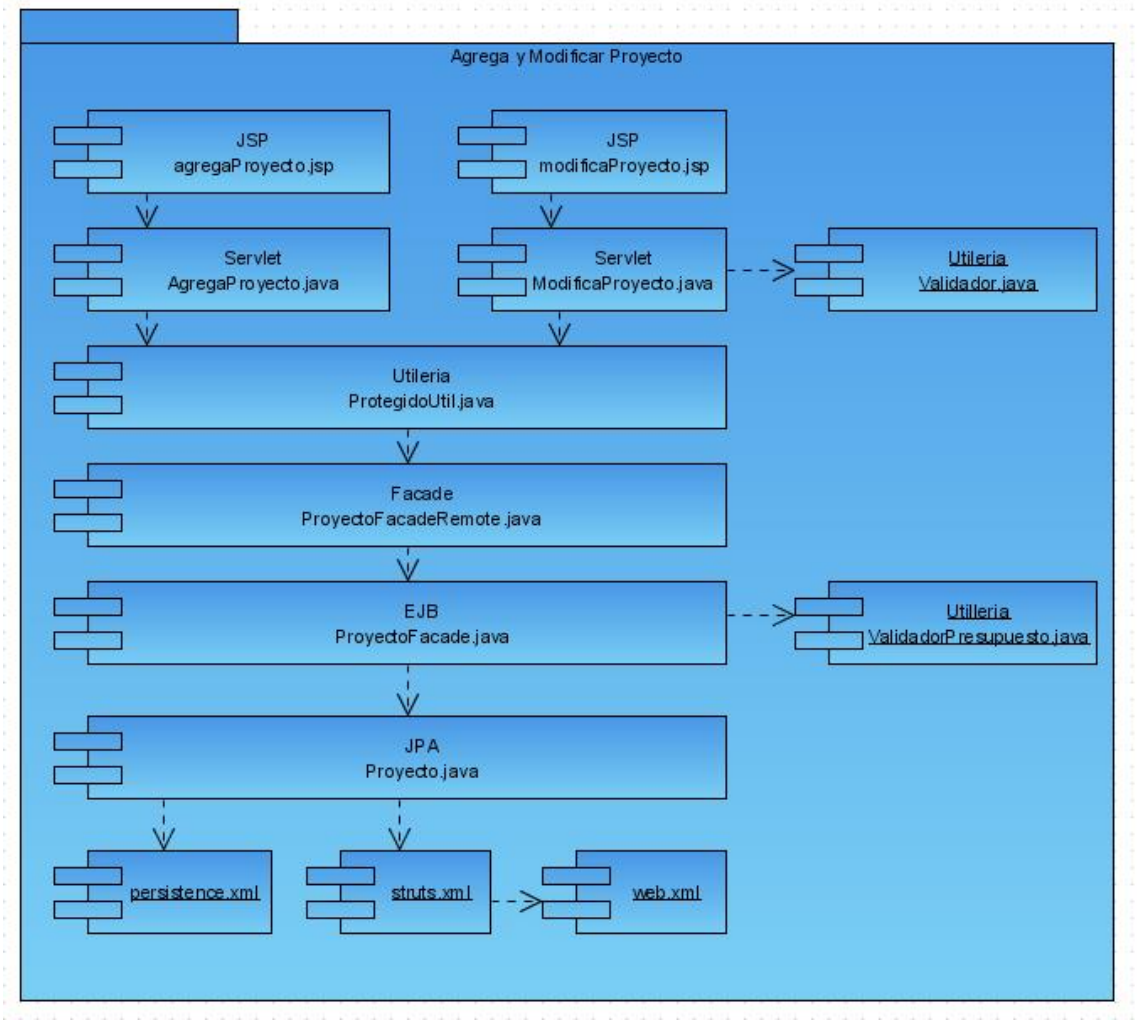


Figura 4.71

Diagrama de Componentes de lo referente a ingresar un proyecto.

En la siguiente sección se abordará la etapa de verificación de la metodología RUP, la cual consta de artefactos creados para verificar el buen funcionamiento del sistema.

4.4. Verificación

En esta sección de diseñaran las pruebas que se realizarán para analizar el bien funcionamiento del sistema.

4.4.1. Modelo de pruebas

El modelo de pruebas, describe como se prueban los componentes obtenidos en el modelo de implementación a través de los casos de prueba por unidad, integración y sistema.

4.4.2. Casos de prueba

Un caso de prueba, especifica una forma de probar el sistema, incluye las entradas, los resultados y las condiciones bajo las cuales ha de evaluarse el sistema. Un caso de prueba se deriva de un caso de uso. Esto quiere decir que por cada caso de uso, existe una o varias pruebas.

A continuación se hace una descripción de los principales casos de prueba que se llevarán a cabo en el FONOL:

4.4.2.1. Pruebas de unidad

Las pruebas de unidad se realizarán en la implementación de cada componente del FONOL con el propósito de probarlos como unidades individuales. Se llevaron a cabo los siguientes tipos de prueba de unidad:

4.4.2.1.1. Pruebas de especificación o de caja negra

Este tipo de prueba unitaria, se realizará para verificar el comportamiento del componente sin tener en cuenta cómo se implementa dicho componente. Es decir,

sólo se tomará en cuenta la salida que el componente devuelve cuando se le da una determinada entrada.

A continuación se diseñarán las pruebas de caja negra de los casos de uso más relevantes y sus respectivas pantallas, comenzando con el caso práctico Ingresar solicitud de ingreso.

Caso práctico: Ingresar solicitud de ingreso al FONOL (Caso de uso: Ingresar Solicitud).

Objetivo: El candidato llena un formulario para poder acceder como beneficiario al FONOL.

Planeación de la prueba

Actividades a realizar:

- No ingresar la información requerida
- Ingresar un correo electrónico incorrecto
- Ingresar un RFC inválido
- Ingresar un RFC de una organización que ya esté registrada

Diseño de la prueba

Caso de Prueba	Procedimiento de prueba
No ingresar la información requerida	El usuario entra en la sección ingresar solicitud. El usuario trata de enviar el formulario sin llenar uno de los campos obligatorios.
Ingresar un correo electrónico incorrecto	El usuario entra en la sección ingresar solicitud.

	El usuario trata de enviar el formulario llenando el correo electrónico incorrectamente.
Ingresar un RFC inválido	El usuario entra en la sección ingresar solicitud. El usuario trata de enviar el formulario llenando el RFC incorrectamente.
Ingresar un RFC de una organización que ya esté registrada	El usuario entra en la sección ingresar solicitud. El usuario trata de enviar el formulario ingresando un RFC que ya ha sido ingresado anteriormente.

Caso práctico: Agregar o Modificar un Delegado (Caso de uso: Agregar o Modificar un Delegado).

Objetivo: El gobierno llena un formulario para agregar o modificar un delegado.

Planeación de la prueba

Actividades a realizar:

- No ingresar la información requerida
- Ingresar un presupuesto no numérico
- Sobrepasar el presupuesto
- Asignar un nombre se usuario que ya esté siendo utilizado

Diseño de la prueba

Caso de Prueba	Procedimiento de prueba
No ingresar la información requerida	El gobierno entra en la sección agregar o modificar delegado.

	El gobierno trata de enviar el formulario sin llenar uno de los campos obligatorios.
Ingresar un presupuesto no numérico	El gobierno entra en la sección agregar o modificar delegado. El gobierno ingresa un presupuesto con un formato no numérico.
Sobrepasar el presupuesto	El gobierno entra en la sección agregar o modificar delegado. El gobierno ingresa un presupuesto que sobrepasa al presupuesto disponible.
Asignar un nombre se usuario que ya esté siendo utilizado	El gobierno entra en la sección agregar o modificar delegado. El gobierno ingresa un nombre se usuario que ya esté siendo utilizado.

Caso práctico: Agrega o modifica proyecto (Caso de uso: Agregar o Modificar un Proyecto).

Objetivo: El beneficiario llena un formulario para agregar o modificar un proyecto.

Planeación de la prueba

Actividades a realizar:

- No ingresar la información requerida
- Ingresar un presupuesto no numérico
- Sobrepasar el tope de presupuesto

Diseño de la prueba

Caso de Prueba	Procedimiento de prueba
No ingresar la información requerida	El beneficiario entra en la sección agregar o modificar proyecto. El beneficiario trata de enviar el formulario sin llenar uno de los campos obligatorios.
Ingresar un presupuesto no numérico	El beneficiario entra en la sección agregar o modificar proyecto. El beneficiario ingresa un presupuesto con un formato no numérico.
Sobrepasar el tope de presupuesto	El beneficiario entra en la sección agregar o modificar proyecto. El beneficiario ingresa un presupuesto que sobrepasa al presupuesto disponible por estado.

A continuación, se planearán las pruebas de integración para posteriormente ejecutarlas y verificar el estado del sistema como un todo.

4.4.2.2. Pruebas de integración

Pruebas integrales o pruebas de integración, son aquellas que se realizan en el ámbito del desarrollo de software una vez que se han aprobado las pruebas unitarias. Únicamente se refieren a la prueba o pruebas de todos los elementos unitarios que componen un proceso, hecha en conjunto, de una sola vez. Consiste en realizar pruebas para consultarificar que un gran conjunto de partes de software funcionan juntos.

En el caso del FONOL, se integrarán todos los módulos para mostrar el comportamiento en conjunto del sistema. Todo esto se realizará tomando en cuenta un caso de uso, que posiblemente sea el más importante: ingresar

solicitud. Se tomará este caso de uso, ya que involucra más validaciones e interacción con más componentes.

En la siguiente sección, se abordara la etapa de liberación de la metodología RUP, y se realizará a nivel de diseño conceptual de la liberación tomando en cuenta el entorno requerido para la prueba.

4.5. Liberación

En esta sección se abordará todo lo necesario para que el entorno del sistema quede configurado, además lo necesario para su buen funcionamiento.

4.5.1. Pasos para el entorno del sistema FONOL

Los pasos requeridos para el funcionamiento correcto del FONOL, serán:

- Base de datos Postgres
- Script de creación de tablas e inserciones a la tabla. (véase Apéndice 1)
- Un servidor de aplicaciones Java EE 5
- Los ejecutables de la aplicación.
- Una fuente de datos registrada dentro del servidor de aplicaciones para que apunte a la base de datos del FONOL

En la siguiente sección se mostrarán los pasos requeridos para poder administrar el sistema FONOL, para que, de esta manera, se garantice el buen funcionamiento de la aplicación.

4.6. Control

En esta sección se darán los pasos a seguir para dar un buen control a la aplicación del sistema FONOL:

4.6.1. Pasos para el control del FONOL

Los pasos son los siguientes:

- Uso de la herramientas de monitoreo de JConsole
- Monitoreo interno del servidor de aplicaciones
- Monitoreo de base de datos

En la siguiente sección, se demostrará con pruebas, el funcionamiento por unidad y por bloque del sistema. Asimismo, con las pruebas se conjuntarán los módulos del FONOL para su puesta en marcha.

Capítulo 5

5. Pruebas y Puesta en Marcha

5.1. Pruebas

En las siguientes secciones se realizarán las pruebas que se diseñaron en la sección anterior, y se mostrará el resultado dado:

5.1.1. Casos de prueba

Caso práctico: Ingresar solicitud de ingreso al FONOL (Caso de uso: Ingresar Solicitud).

Ejecución de la prueba

Caso de Prueba	Resultado
No ingresar la información requerida	Éxito.
Ingresar un correo electrónico incorrecto	Éxito.
Ingresar un RFC inválido	Éxito.
Ingresar un RFC de una organización que ya esté registrada	Éxito.

Interfaces graficas de los resultados de las pruebas

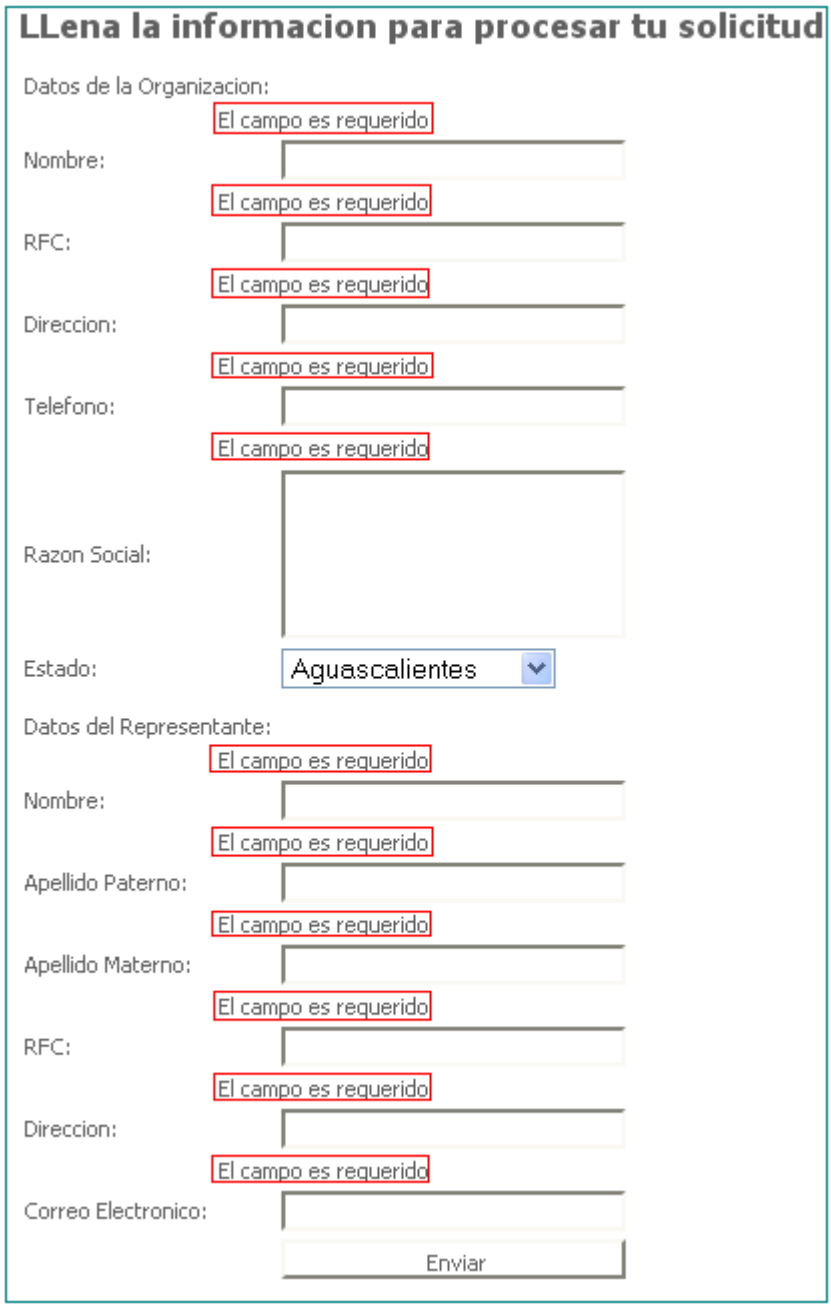


Figura 5.1

Prueba de caja negra de Ingresar una Solicitud, en la cual, no se ingresa la información obligatoria.

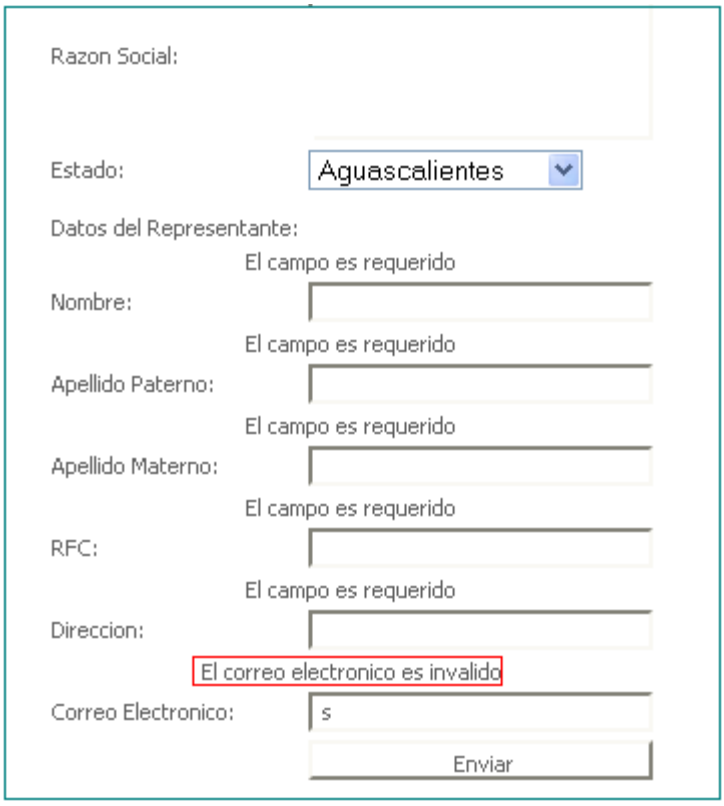


Figura 5.2

Prueba de caja negra de Ingresar una Solicitud, en la cual, no se ingresa un correo electrónico válido.

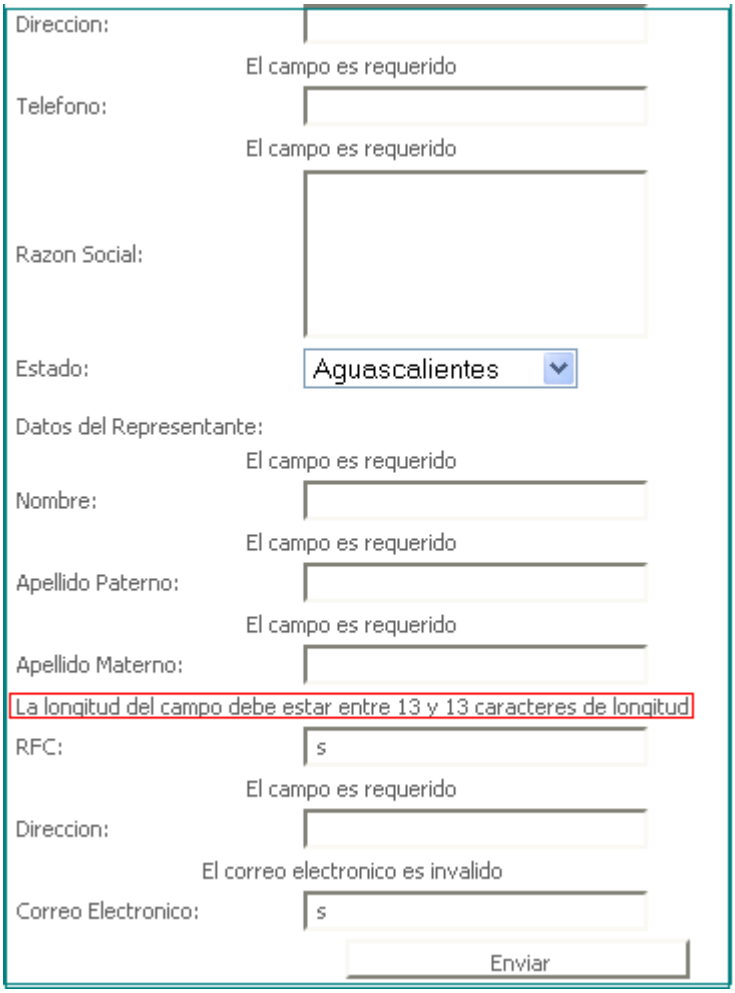


Figura 5.3

Prueba de caja negra de Ingresar una Solicitud, en la cual, no se ingresa un RFC válido.

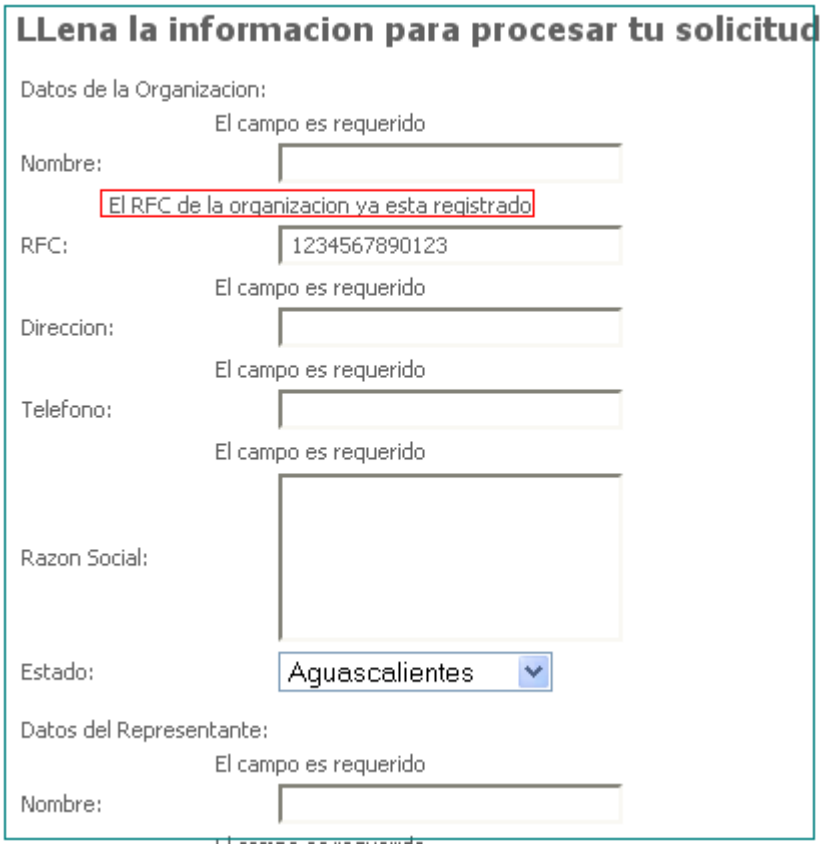


Figura 5.4

Prueba de caja negra de Ingresar una Solicitud, en la cual, no se ingresa un RFC que ya haya sido registrado anteriormente.

Caso práctico: Agregar o Modificar un Delegado (Caso de uso: Agregar o Modificar un Delegado).

Ejecución de la prueba

Caso de Prueba	Resultado
No ingresar la información requerida	Éxito.
Ingresar un presupuesto no numérico	Éxito.
Sobrepasar el presupuesto	Éxito.
Asignar un nombre se usuario que ya esté siendo utilizado	Éxito.

Interfaces graficas de los resultados de las pruebas

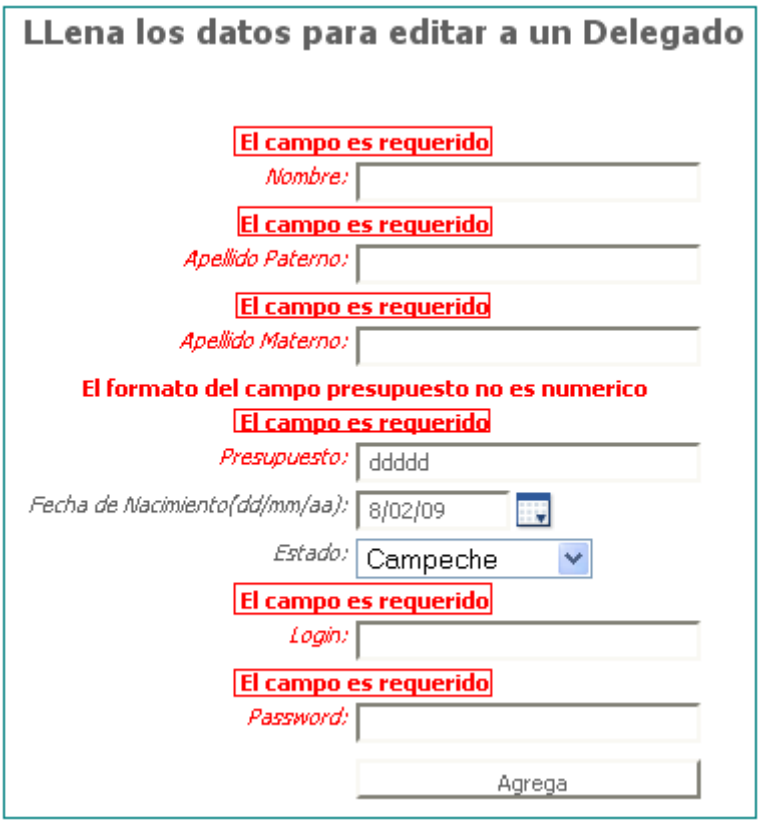


Figura 5.5

Prueba de caja negra de Agregar o Modificar un Delegado en la cual, no se ingresan los datos requeridos y obligados.

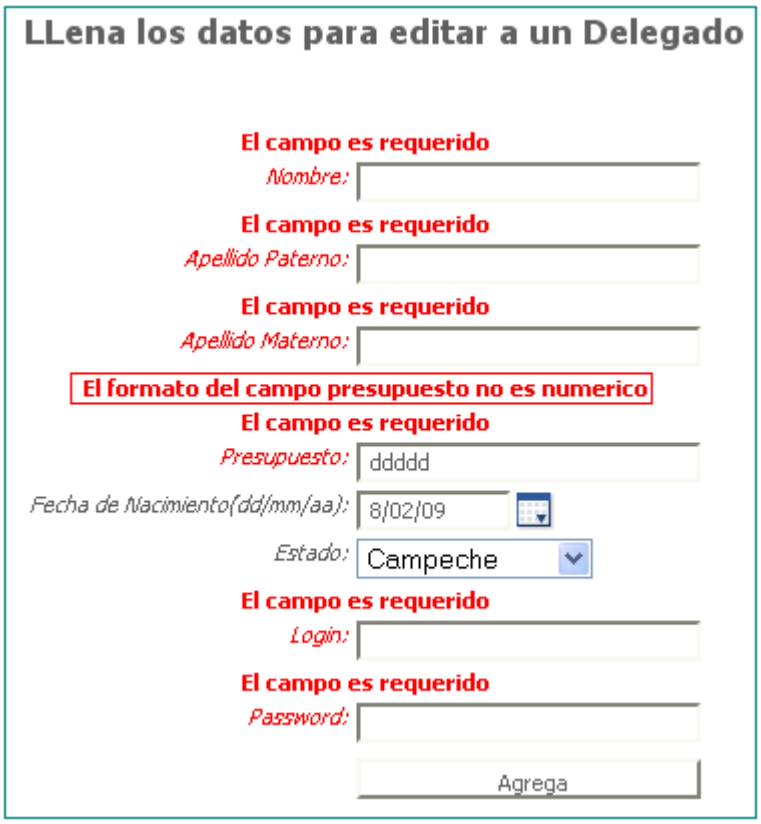


Figura 5.6

Prueba de caja negra de Agregar o Modificar un Delegado en la cual, se ingresa un presupuesto que no está en formato numérico.

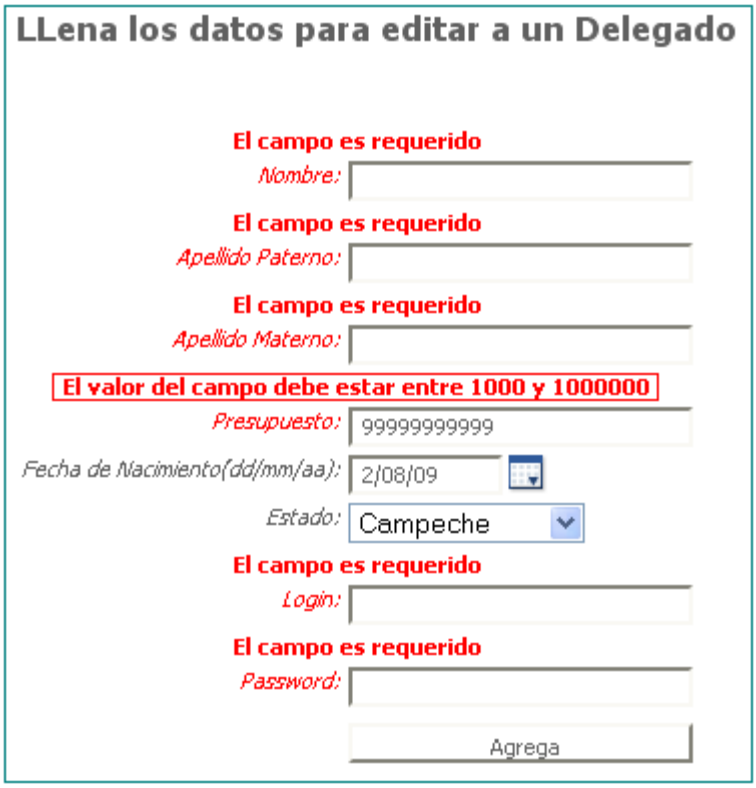


Figura 5.7

Prueba de caja negra de Agregar o Modificar un Delegado en la cual, se ingresa un presupuesto que está fuera del tope del presupuesto.

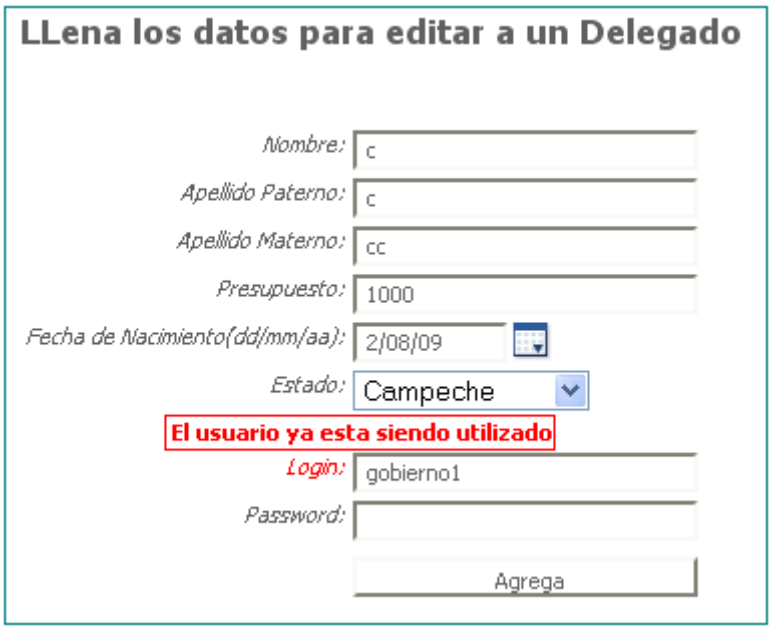


Figura 5.8

Prueba de caja negra de Agregar o Modificar un Delegado en la cual, se ingresa un usuario que ya ha sido utilizado anteriormente.

Caso práctico: Agrega o modifica proyecto (Caso de uso: Agregar o Modificar un Proyecto).

Ejecución de la prueba

Caso de Prueba	Resultado
No ingresar la información requerida	Éxito.
Ingresar un presupuesto no numérico	Éxito.
Sobrepasar el tope de presupuesto	Éxito.

Interfaces graficas de los resultados de las pruebas

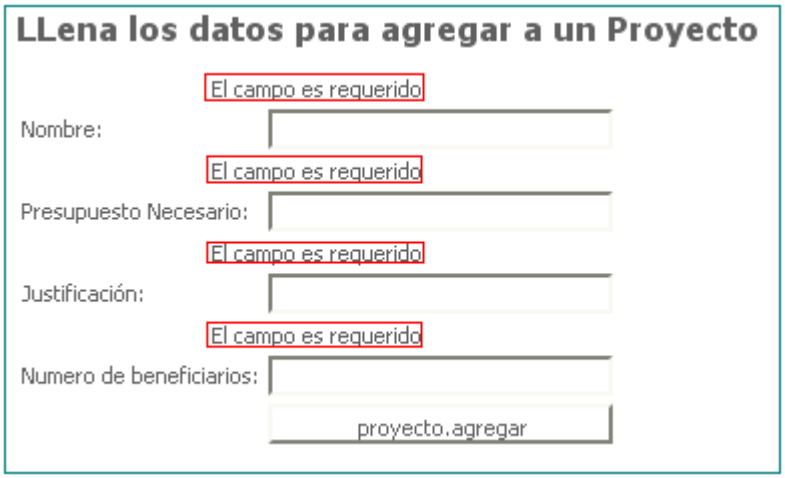


Figura 5.9

Prueba de caja negra de Agregar o Modificar un Proyecto en la cual, no se ingresan los datos requeridos.

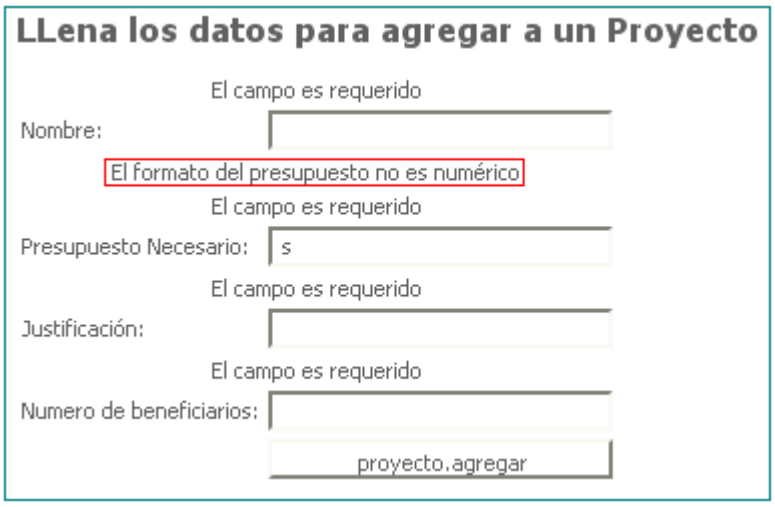


Figura 5.10

Prueba de caja negra de Agregar o Modificar un Proyecto en la cual, se ingresa el presupuesto con un formato que no es numérico.

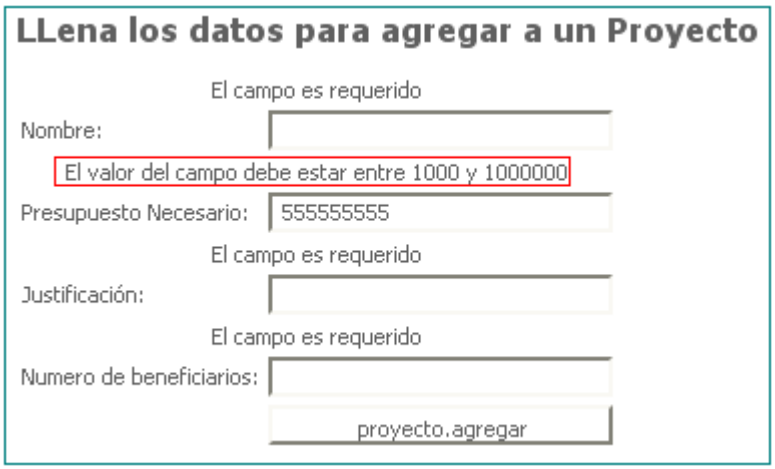


Figura 5.11

Prueba de caja negra de Agregar o Modificar un Proyecto en la cual, se ingresa un presupuesto que no se encuentre dentro del rango de presupuesto permitido.

5.1.1.1.1. Pruebas de caja blanca

La prueba de caja blanca que se realizará, será sobre la clase más importante del sistema, la cual es **protegido.util.ProtegidoUtil.java.**

En la figura 5.12 se muestran las partes más importantes del código:

```
public Object getObjeto(Usuarios usuarios) {
    TipoUsuario tipo=usuarios.getTipo();
    switch(tipo){
        case GOBIERNO:
            return getUsuarioFacade().dameGobierno(usuarios);
        case BENEFICIARIO:
            return getUsuarioFacade().dameBeneficiario(usuarios);
        case DELEGADO:
            return getUsuarioFacade().dameDelegado(usuarios);
    }
    return null;
}
public Beneficiarios calificaProyecto(Integer id, String estatuse, String comentario) {
    Proyectos pro=getProyectosFacade().find(id);
    logea("ProtegidoUtil.calificaProyecto.proyecto:"+pro);
    if(estatuse.equals("ESPERA"))
        pro.setEstatus(EstatusCandidato.ESPERA);
    else if(estatuse.equals("RECHAZADO"))
        pro.setEstatus(EstatusCandidato.RECHAZADO);
    else if(estatuse.equals("ACEPTADO")){
        pro.setEstatus(EstatusCandidato.ACEPTADO);
    }
    pro.setComentario(comentario);
    getProyectosFacade().edit(pro);
    Beneficiarios ben=pro.getIdbeneficiario();
    logea("ProtegidoUtil.calificaProyecto.beneficiarios:"+ben);
    return ben;
}
```

Figura 5.12

Código de ProtegidoUtil.java, la cual realiza una de las funciones más importantes: conectar la parte Web con la parte Empresarial.

Ahora se mostrará la prueba de caja blanca para mostrar la efectividad de la clase, esto se realizará con JUnit, el cual es una herramienta para realizar pruebas unitarias a clases Java. A continuación, se muestra en la figura 5.13 el código de JUnit:

```
package test;

import static org.junit.Assert.*;

public class ComparaFechasTest {

    @Test
    public void testComparaValidezFecha() {
        assertTrue(ComparaFechas.comparaValidezFecha(new Date(), new Date()));
    }

    @Test(expected=ArithmeticException.class)
    public void aritmetico(){
        assertNull(4/0);
    }

    @Test
    public void validaPresupuesto(){
        assertTrue(Presupuesto.obtenPresupuesto()>10000);
    }

}
```

Figura 5.13

Código de ComparaFechasTest.java, la cual realiza las pruebas unitarias.

Finalmente, se muestra el resultado de la prueba, en la figura 5.14.

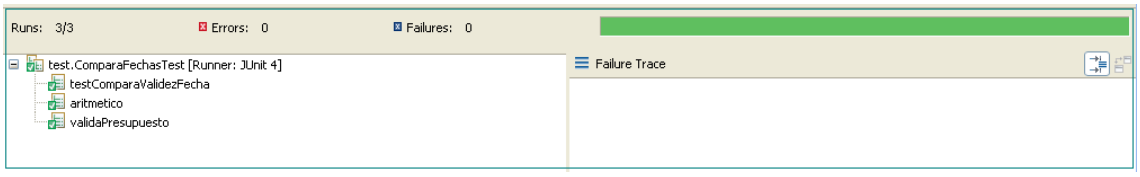


Figura 5.14

Resultado de la ruaba unitaria con JUnit.

En la siguiente sección se mostrará la puesta en marcha del sistema, con toda la configuración realizada para que esté en pleno funcionamiento.

5.2. Puesta en Marcha

5.2.1. Instalación del entorno

En esta sección se mostrarán los pasos de instalación del sistema FONOL:

- Se instaló una instancia de Postgres para que aloje los datos del FONOL
- Se ejecutó el script del Apéndice A, para la instalación de los esquemas de la base de datos del FONOL, en la base de datos Postgres instalada anteriormente
- Se instaló el servidor de aplicaciones Glassfish
- Se instaló la aplicación en el Servidor de Aplicaciones Glassfish de Sun
- Se configuró la fuente de datos desde el servidor de aplicaciones para que pudiera conectarse a la base de datos de Postgres
- Se probó que la aplicación estuviera arriba tecleando la dirección http://<servidor>:<puerto>/FONOL-war, tal y como se muestra en la figura 5.15:

Figura 5.15

Pantalla principal del FONOL Web

5.2.2. Monitoreo y control de la aplicación

Para llevar un control más preciso del FONOL, se utilizarán dos herramientas para monitorear el funcionamiento de la aplicación, una es JConsole y la otra es el perfilador del IDE Netbeans.

Con JConsole se puede monitorear cualquier aplicación Java corriendo local o remotamente. En el caso del FONOL, se ejecuto la aplicación JConsole (incluida en la instalación del entorno de desarrollo estándar de Java) y se le configuró para que apuntara al servidor de aplicaciones Glassfish, en la figura 5.16 se muestra el resultado:

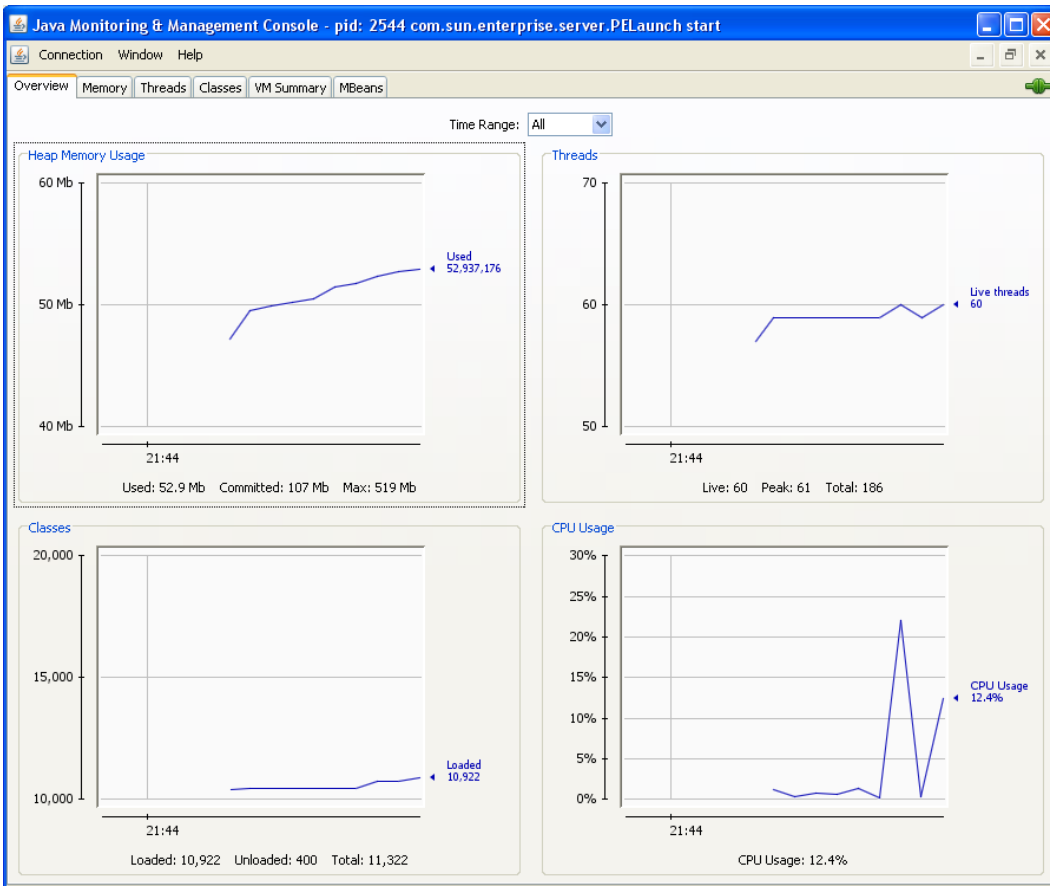


Figura 5.16

Aplicación JConsole, utilizado para monitorear aplicaciones Java

Por otra parte, se utilizó el perfilador de Netbeans, el cual solo se puede ejecutar si se le indica a donde apunte con unos parámetros específicos de la aplicación, en la figura 5.17 se muestra el resultado:

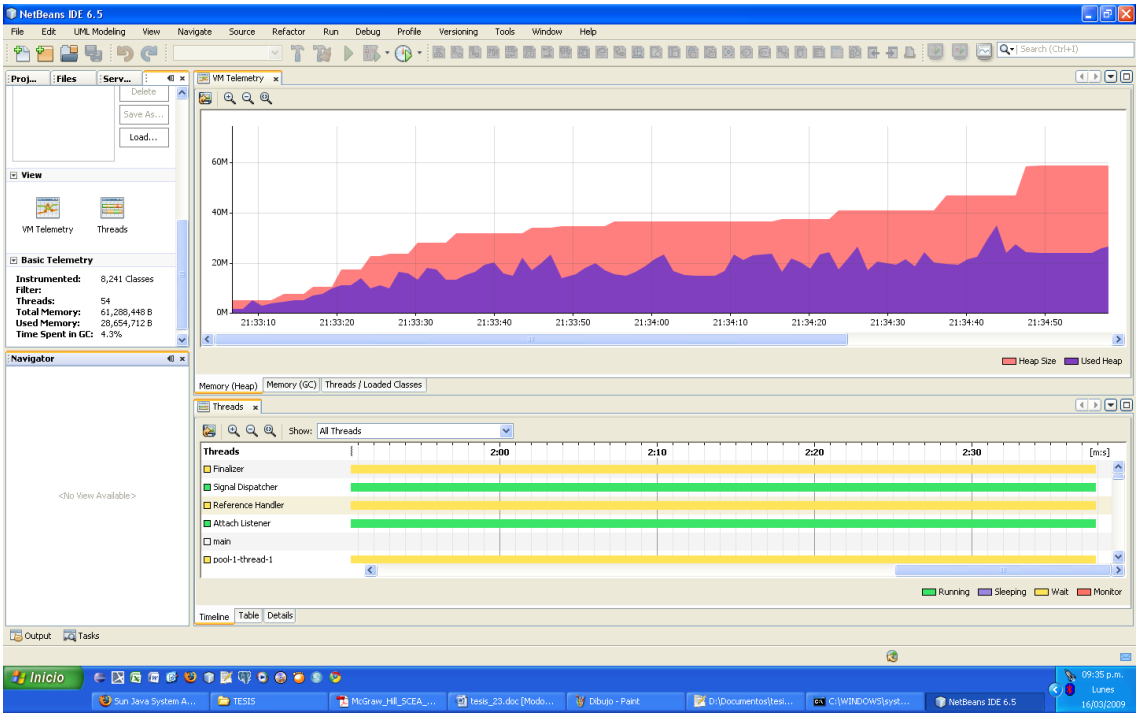


Figura 5.17

Perfilador de Netbeans, utilizado para monitorear aplicaciones Java

Conclusión

A lo largo del desarrollo de la tesis, se han cumplido con los objetivos previstos inicialmente. Se desarrollo un sistema capaz de administrar fondos para organizaciones no lucrativas de una manera sencilla, lo cual es de vital importancia, ya que anteriormente cada organización tenía que solicitar un oficio por escrito para ser aceptada en el FONOL y esto tardaba varios días en procesarse. El proceso se hizo más ágil y rápido, ya que al tratarse de un sistema Web, la organización ya no necesitará desplazarse personalmente de un estado a otro para ingresar y verificar el estado de sus solicitudes.

Finalmente, cabe mencionar que las expectativas se superaron ya que se pretendió obtener un modelo seguro, rápido, confiable y remoto, lo cual se logró y de una mejor manera a la esperada. Además de cumplir con los objetivos y las expectativas del sistema, se logró una arquitectura robusta y escalable, lo cual permitirá que en un futuro se incorporen nuevas funcionalidades sin afectar el desarrollo ya hecho. De igual manera, se desarrollo con la última tecnología JEE 5.0, lo cual garantiza que el sistema permanezca vigente al menos cinco años en adelante.

Apéndices

Apéndice A: Script de creación de tablas

```
CREATE SEQUENCE candidatos_idcandidato_seq
    INCREMENT BY 1
    NO MAXVALUE
    NO MINVALUE
    CACHE 1;

CREATE TABLE candidatos (
    idcandidato integer DEFAULT nextval('candidatos_idcandidato_seq'::regclass) NOT NULL,
    email varchar(255),
    telefono varchar(255),
    nombre varchar(255),
    razonsocial varchar(255),
    fechapeticion date,
    direccion varchar(255),
    appatrep varchar(255),
    dirrep varchar(255),
    rfcrep varchar(255),
    nombrerep varchar(255),
    estatus integer,
    rfc varchar(255),
    apmatrep varchar(255),
    idestado integer
) WITHOUT OIDS;

CREATE SEQUENCE gobiernos_idgobierno_seq
    START WITH 1
    INCREMENT BY 1
    NO MAXVALUE
    NO MINVALUE
    CACHE 1;

CREATE TABLE gobiernos (
    idgobierno integer DEFAULT nextval('gobiernos_idgobierno_seq'::regclass) NOT NULL,
    pib numeric(38,0),
    idusuario integer
) WITHOUT OIDS;

CREATE SEQUENCE beneficiarios_idbeneficiario_seq
    INCREMENT BY 1
    NO MAXVALUE
```

```
    NO MINVALUE
    CACHE 1;


CREATE TABLE beneficiarios (
    idbeneficiario integer DEFAULT nextval('beneficiarios_idbeneficiario_seq'::regclass) NOT NULL,
    fechaaceptacion date,
    idcandidato integer,
    idusuario integer
) WITHOUT OIDS;

CREATE SEQUENCE usuarios_idusuario_seq
    INCREMENT BY 1
    NO MAXVALUE
    NO MINVALUE
    CACHE 1;

CREATE TABLE usuarios (
    idusuario integer DEFAULT nextval('usuarios_idusuario_seq'::regclass) NOT NULL,
    fechainicio timestamp without time zone,
    fechafin date,
    login varchar(9),
    indefinido integer,
    tipo integer,
    pass varchar(9)
) WITHOUT OIDS;

CREATE SEQUENCE estados_idestado_seq
    INCREMENT BY 1
    NO MAXVALUE
    NO MINVALUE
    CACHE 1;

CREATE TABLE estados (
    idestado integer DEFAULT nextval('estados_idestado_seq'::regclass) NOT NULL,
    nombre varchar(255),
    idgobierno integer
) WITHOUT OIDS;

CREATE SEQUENCE delegados_iddelegado_seq
    INCREMENT BY 1
    NO MAXVALUE
    NO MINVALUE
    CACHE 1;

CREATE TABLE delegados (
    iddelegado integer DEFAULT nextval('delegados_iddelegado_seq'::regclass) NOT NULL,
```

```
    apmaterno varchar(255),
    fechanacimiento date,
    nombre varchar(255),
    presupuesto numeric(38,0),
    appaterno varchar(255),
    idusuario integer,
    idestado integer
) WITHOUT OIDS;

CREATE SEQUENCE historialpantallas_idhistorial_seq
    START WITH 1
    INCREMENT BY 1
    NO MAXVALUE
    NO MINVALUE
    CACHE 1;

CREATE TABLE historialpantallas (
    idhistorial integer DEFAULT nextval('historialpantallas_idhistorial_seq'::regclass) NOT NULL,
    operacion varchar(255),
    fecha date,
    pantalla varchar(255),
    idusuario integer
) WITHOUT OIDS;

CREATE SEQUENCE proyectos_idproyecto_seq
    INCREMENT BY 1
    NO MAXVALUE
    NO MINVALUE
    CACHE 1;

CREATE TABLE proyectos (
    idproyecto integer DEFAULT nextval('proyectos_idproyecto_seq'::regclass) NOT NULL,
    presupuesto numeric(38,0),
    comentario varchar(255),
    nombre varchar(255),
    estatus integer,
    beneficiarios integer,
    justificacion varchar(255),
    idbeneficiario integer
) WITHOUT OIDS;
```

Bibliografía

- G. Booch, I. Jacobson, J. Rumbaugh. El Lenguaje Unificado de Modelado. Guía del usuario. Addison-Wesley/Diaz de Santos,1999.
- J. Rumbaugh, I. Jacobson, G. Booch, El Lenguaje Unificado de Modelado. Manual de referencia. Addison-Wesley,2000.
- OMG. "*Unified Modeling Language. Notation Guide*". Version 1.5. 2003.
- Larman, Craig , UML y patrones : introduccion al análisis y diseño orientado a abjetos Prentice-Hall, 2ª ed. 2002.
- Jacobson, Ivar, Booch, Grady and Rumbaugh, James. "*El Proceso Unificado de Modelado*". Addison-Wesley, 2000.