Universidad de las Ciencias Informáticas

Facultad 3

Título:

Análisis y diseño del Sistema de atención a solicitudes de revisiones técnicas.

Trabajo de Diploma para optar por el título de
Ingeniero Informático

Autor: Carlos Arian Cruz Chapotín
Tutor: Jorge Luis Valdés

La Habana, Junio de 2011.

Frase

Aprendí a considerar más el aspecto brillante de mi situación que lo que me faltaba, y este recurso, a veces, me proporcionó tan inefable consuelo, que apenas puedo expresarlo.

Daniel Defoe (1660-1731)

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Carlos Arian Cruz Chapotín **Jorge Luis Valdés**

_____ _____

Firma del Autor Firma del Tutor

Agradecimientos

Creo que si menciono los nombres de todas aquellas personas que de alguna que otra forma tuvieron que ver en la realización de este trabajo, muchos dirían que no fui yo quien lo realizó o muchos podrían irse convertidos en ingenieros. Por tanto, gracias a todos los que de alguna forma u otra me ayudaron y queden seguros de que mi deuda jamás podrá ser cumplida, porque las deudas de gratitud no pueden ser pagadas si siquiera en especie.

Dedicatoria

Dedico este trabajo a mi familia a los cuales ya no puedo llamar ni tíos ni primos, porque todos se han comportado como si fueran madres o padres de su único hijo, yo. A mi honey que lleva soportando mi "peso" durante 5 años, a mis amigos que estuvieron conmigo en las malas, así que ni decir de las buenas.

A mi abuelo...

A mi padre…

Y a la Mariana que me dio a luz...mi madre.

Resumen

Estableciendo como blanco fundamental la creación de una infraestructura informática que permita la implantación calificada y fiable de una instancia de Gobierno Electrónico en la Universidad de Ciencias Informáticas, se realizan en el Centro de Gobierno Electrónico (CEGEL) las primeras tareas de organización interna que permitan la consolidación necesaria para el desarrollo de soluciones encaminadas el Gobierno Electrónico dentro y fuera de nuestro país.

La revisión y calificación de los artefactos que se generan a través del proceso de desarrollo de software dentro del centro, son algunas de las tareas más importantes que se realizan. Para el cumplimiento de este objetivo en un ámbito especificado por tres disciplinas fundamentales: arquitectura, tecnología y seguridad, surge Grupo de Arquitectura-Tecnología-Seguridad (GATS), creando un proceso llamado *atención a solicitudes de revisión técnica*.

Enfatizando en la mejora de este proceso, se propone el análisis y diseño de un sistema capaz de gestionar cada revisión técnica que se genera en el ámbito concerniente a GATS. Realizando entonces en cada capítulo los entregables y tareas necesarias para dar cumplimiento a la anterior propuesta de análisis y diseño, por tanto, se realiza un estudio de estado del arte sobre las herramientas que posibiliten una gestión similar a la que se desea, se analiza el negocio para determinar los requisitos y obtener los casos de uso que guiaran el proceso de desarrollo, además de proveer lo necesario para iniciar fases superiores de desarrollo, dentro de las que se encuentran el análisis y diseño del sistema.

PALABRAS CLAVE

Gobierno Electrónico, arquitectura, Gestión, Análisis, Diseño.

Índice

Índice de tablas

Índice de figuras

Introducción

Con el objetivo de producir soluciones informáticas orientadas al Gobierno Electrónico surge en la Universidad de Ciencias Informáticas (UCI) el Centro de Gobierno Electrónico (CEGEL), el cual deberá estandarizar y normalizar cada proyecto que surja con este perfil en la UCI.

Como parte de la organización interna de CEGEL, surge el Grupo de Arquitectura-Tecnología-Seguridad (GATS), el cual se encargará de la normalización y evaluación de los entornos tecnológicos, arquitecturas de sistema, además de la aplicación de políticas de seguridad en el centro.

Específicamente, GATS trae consigo varios objetivos a cumplir, los cuales marcarán las tareas a realizar dentro de CEGEL, entre estos se encuentran:

1. Definir entornos normativos basados en estándares internacionales para la implantación de políticas de seguridad informática.
2. Proponer marcos tecnológicos genéricos a partir de tipologías de proyectos, que puedan ser adoptadas y/o adaptadas al desarrollo de sistemas informáticos en el Centro de Gobierno Electrónico.
3. Brindar asesoramiento técnico a proyectos en la toma de decisiones para la definición de marcos tecnológicos.
4. Proveer un mecanismo institucionalizado para el control del cumplimiento de los estándares establecidos.
5. Crear un marco formativo de pregrado y posgrado sobre temas de arquitectura, tecnología y seguridad informática.

Conjuntamente con los objetivos anteriormente mencionados existen una serie de tareas que el grupo debe llevar a cabo, dentro de las cuales está atender las Solicitudes de Revisiones Técnicas**,** donde se revisan una serie de artefactos que responden a aspectos fundamentales del proceso desarrollo de software en un proyecto productivo determinado, y que generalmente se manejan en forma de documentación. Algunos de estos artefactos son:

1. Documentos de Solicitud de equipamientos tecnológicos
2. Proyectos Técnicos

3. Documentos de Arquitectura de Software
4. Documentos de Plataforma Tecnológica de Seguridad Informática
5. Documentos sobre Normas y Estándares

En la actualidad, el proceso de solicitudes de revisiones técnicas comienza cuando la dirección de cualquier proyecto productivo en el centro solicita la evaluación de un artefacto especifico, el cual primeramente debe ser enviado por vía correo electrónico a los Jefes de Departamento de CEGEL donde es reenviado a la gerencia de GATS. Una vez recibido el documento se procede con su revisión, al terminar esta se emite un Dictamen Técnico (DT) en el que se recoge la información positiva o negativa respecto al artefacto revisado, sirviendo esto como motivo de rechazo o aprobación según sea el caso. El DT generado es enviado por la misma vía que ingresó al grupo, hasta llegar a su emisor inicial, la dirección del proyecto que solicitó la revisión. En caso de que el DT emitido sea de rechazo, los solicitantes de la revisión pueden solicitar además un taller informativo con el grupo, donde se discuten los criterios bajo los cuales fue rechazado el documento.

Durante el tiempo que demore el artefacto a revisar, en llegar a manos del GATS, hasta que culmina la revisión técnica, no es posible por parte del solicitante visualizar el estado en el que se encuentra la solicitud realizada, por tanto, el proyecto ve afectada su planificación, teniendo que paralizar otras tareas dependientes de la revisión del artefacto. Disminuye además la eficiencia del proceso debido a que se depende del servicio de correo para manejar cada uno de los pasos referentes a la solicitud de revisión técnica, y al envío del DT generado.

El proceso de evaluación y validación de artefactos realizados por GATS permite un aumento en la calidad con la que se desarrollan los proyectos dentro del grupo CEGEL, además de incentivar la estandarización en cuanto a la definición de arquitecturas y plataformas tecnológicas se refiere. Es por esto que la eficiencia, agilidad y calidad con la que se realiza este proceso debe ser mejorada.

Debido a las razones anteriormente expuestas se plantea el siguiente **problema de investigación**: ¿Cómo obtener los entregables necesarios para la posterior implementación del Sistema de Atención a Solicitudes de Revisión Técnica?
Para la resolución del problema existente en la investigación se define como **objeto de estudio**: el proceso de desarrollo de software.

Estableciendo como **objetivo general,** la realización del análisis y diseño de un sistema web, desarrollado sobre una plataforma libre, que permita la informatización del proceso de atención a solicitudes de revisión técnica por el grupo de tecnología del centro CEGEL.

Para lograr dicho objetivo se hace necesario investigar sobre los sistemas de gestión de proyectos encargados de las revisiones técnicas de los artefactos del desarrollo, como una subcategoría de los sistemas de gestión de información siendo este el **campo de acción** en el que se sintetiza la investigación.

Se determina como **idea a defender** que: Con el análisis y diseño de un sistema para la atención y seguimiento de revisiones técnicas en CEGEL, permitirá dar paso a la implementación del mismo.

Se hace primordial para el equipo de desarrollo satisfacer el objetivo general de la investigación, para esto es necesario cumplir con los siguientes **objetivos específicos:**

1. Elaborar el marco teórico de la investigación a partir del estudio del proceso de revisiones técnicas en el contexto de la gestión de proyectos.
2. Realizar el análisis y diseño del sistema a partir de la metodología RUP.
3. Validación de la propuesta de análisis y diseño.

Para el cumplimiento cabal de los objetivos anteriormente expuestos, se trazan las siguientes **tareas de investigación:**

1. Realización del estudio del estado del arte de las herramientas para la gestión y seguimiento de revisiones técnicas.

2. Realización de la descripción del negocio.

3. Modelación del negocio.

4. Identificación de los requisitos del sistema, a partir de lo que plantea el proceso de revisiones técnicas definidos por el Proceso de Mejora de la UCI.

5. Realización del estudio de factibilidad tecnológica para la definición del entorno tecnológico donde se desarrollará el sistema.

6. Realización del análisis y diseño de la solución siguiendo la metodología RUP y a partir del expediente de proyecto aprobado por Calidad UCI.

7. Validación del análisis y el diseño con la utilización de métricas para la medición de software.

8. Realización de entrevistas de aceptación con el cliente del análisis y diseño propuestos.

En el cumplimiento de las tareas se usarán los siguientes métodos teóricos:

- El **Histórico-Lógico** permitirá una mayor comprensión del estado actual de los sistemas de gestión Web a partir del análisis de su progreso y las etapas principales por las que han cursado.
- La **Modelación** mediante el lenguaje de modelado UML 2.0 se utilizará para reflejar la estructura, relaciones internas y características de la solución a través de diagramas.
- La **Entrevista** posibilitará un mejor entendimiento de los objetivos que persigue el cliente con el desarrollo del sistema, permitiendo un cumplimiento cabal de los requisitos.

El presente documento consta de tres capítulos:

Capítulo 1, en el cual se tratarán aquellos temas relacionados con la propuesta de la estructura tecnológica y de herramientas, tanto como para el desarrollo del análisis y diseño de SIGET, como para su desarrollo final. En este capítulo se pueden obtener además, información relacionada con las herramientas similares al sistema que se diseña, dividida en dos porciones o ámbitos fundamentales, el internacional y el nacional. Luego de este análisis se comienza con aquellas propuestas de herramientas metodologías y leguajes que serán utilizados en todo el proceso de desarrollo de software.

Capítulo 2, a través de este capítulo se trata de llegar a la determinación de los requisitos funcionales, a través del análisis de las actividades, procesos y subprocesos que conforman el negocio perteneciente a la organización del cliente. Para lograr dicho objetivo, se realizan las actividades necesarias propuestas por RUP para el flujo de trabajo Modelamiento del negocio. Luego de concluida esta parte del capítulo, fueron obtenidos los procesos y actividades que es posible automatizar para obtener una primera versión de los requisitos propuestos por el usuario, los cuales una vez refinados y obtenidos los diagramas de casos de uso del negocio proveen las herramientas necesarias para proseguir con la siguiente fase de desarrollo.

Capítulo 3, análisis y diseño del sistema de atención a solicitudes de revisiones técnicas, permite observar claramente, a través de diagramas la representación de los casos de uso que más tarde se transformarán en funcionalidades. Se inicia dicha representación en el análisis, con la realización de los diagramas y descripciones generadas en este flujo de trabajo, donde inicialmente se describe la dinámica general que poseerá el sistema sin profundizar en los aspectos específicos de la estructura del sistema ni los aportados por las herramientas a utilizar. Después de concluido este proceso, es posible entonces continuar con el diseño del sistema en el cual se tratan de vincular de manera profunda la dinámica general del sistema con los aspectos específicos de la implementación, se consideran entonces todas las herramientas físicas como *frameworks*, lenguajes de programación, IDEs de desarrollo y librerías. Se toman en cuenta además las herramientas lógicas tales como arquitecturas, patrones de diseño y arquitectónicos.

Capítulo 1. Fundamentación teórica.

1.1 Introducción.

El siguiente capítulo tiene como objetivo mostrar el estado actual en el que se encuentran las herramientas similares a la que se desea obtener, así como una breve conceptualización de las áreas en las que va a operar la aplicación en desarrollo, las cuales son los Sistemas de Gestión de Información, y los Sistemas de Gestión de Proyectos. Se desglosan además las herramientas y tecnologías a utilizar para el desarrollo del sistema así como sus características fundamentales, ventajas y desventajas de su uso.

1.2 Marco conceptual

En este epígrafe se darán a conocer conceptos esenciales que permitirán una mayor comprensión del tema de la investigación en transcurso.

Sistema de gestión de información

Se puede definir como sistema a “cualquier conjunto organizado de recursos y procedimientos unidos y regulados por interacción o interdependencia para lograr un conjunto de funciones específicas”.[1]

Según el Diccionario Manual de la Lengua Española gestión puede definirse como “acción o trámite que se lleva a cabo para conseguir o resolver algo”. [2]

Es posible determinar entonces que un sistema de gestión de información en cuanto a informática se refiere, no es más que un grupo organizado de recursos y procedimientos informáticos, relacionados y coordinados entre sí, para recopilar, almacenar, organizar y relacionar datos con un contexto y objetivo específico.

Sistema de gestión de proyecto

Según Domingo (2005), “un proyecto no es más que el conjunto de actividades, planificadas, ejecutadas y supervisadas que con recursos finitos tiene como objeto crear un servicio”, establece además que “para que exista un proyecto, debe existir una coordinación de actos orientados a la consecución de uno o varios objetivos, integrados entre si y estructurados”. [3]

De manera que un sistema de gestión de proyectos no es más que una serie de elementos y procedimientos informáticos organizados, que relacionados entre sí se encargan de brindar soporte a la mayoría de las tareas que se generan durante el transcurso de un proyecto para que este cumpla con sus objetivos de manera cabal y eficiente.

Revisiones Técnicas

Una revisión técnica es el proceso mediante el cual es evaluado uno o más aspectos dentro del desarrollo de un proyecto, sea informático o no. Estos aspectos pueden ser solicitudes de equipamiento técnico, o la aprobación de determinada arquitectura por un especialista. Dentro de esta categoría de revisión técnica podrían aparecer las relacionadas con la gestión de la calidad en los proyectos informáticos.

Aplicaciones Web

El surgimiento de la Web se le es atribuido a Tim Berners-Lee, en el año 1991, quien construye un sistema basado en el concepto del hipervínculo, o sea, un sistema en el cual cada documento manejado allí posee enlaces a otros documentos dentro del sistema. Este sistema surge utilizando dos conceptos que marcarían la web 1.0, el primero, HTTP a través del cual el programa gestionaba los documentos desde el servidor y el segundo, el formato en el cual se escribían dichos documentos HTML.

El autor de dicho sistema explica que sus aspiraciones eran encontrar “formas nuevas de trabajar en equipo de manera eficaz, y especialmente, atravesando barreras geográficas” (entrevista publicada en el diario El País). Pues no se hicieron esperar dichas aspiraciones puesto que en tan solo 10 años, la web había evolucionado tanto que era fácil encontrarla dentro de cualquier ámbito posible.

Una aplicación está diseñada e implementada como una herramienta para un propósito específico, dando respuesta a una necesidad de un usuario determinado.

Ahora, haciendo coincidir los conceptos anteriormente relacionados se puede concluir que una aplicación Web no es más que una aplicación que se rige por las potencialidades, tecnologías, estándares y normas que define el W3C, cuyos contenidos además pueden ser vistos o gestionados a través de una interfaz gráfica, la cual es posible observar gracias a un navegador.

Ventajas y desventajas de la web.

Ventajas:

- Son independientes del hardware y software que se utilicen.
- No es necesario descargarlas, instalarlas o configurarlas, sino que se accede a ellas a través de un navegador.
- Es posible la detección de errores de forma centralizada sin necesidad de que el cliente obtenga un parche o una nueva versión para solucionar el problema.
- Es posible utilizar diferentes tecnologías compatibles y diferentes lenguajes de programación, en cuyo caso el cliente no advierte problema alguno, y se mantiene como un proceso transparente.

Desventajas:

- Es necesaria la conexión de red constante, elemento que falta en algunos casos.
- No suelen ser confiables a los usuarios en cuanto a datos personales se refiere.
- Posee deficiencias en comparación con las aplicaciones de escritorio, relacionadas generalmente con la velocidad de gestión, ya que esta generalmente depende de la velocidad de la conexión.

1.3. Trabajos similares

Antes de comenzar el desarrollo del sistema de atención a solicitudes de revisiones técnicas se hace necesaria la búsqueda de un sistema de gestión de información en general y más específico para la gestión de proyectos, ya que estos son los sistemas que más similitudes presentan. Por lo que es necesario observar qué ocurre en cuanto al desarrollo de estos sistemas, tanto a nivel mundial como en el área donde se desea desarrollar el sistema, para tomar las experiencias locales como guía, acción que es recomendada por expertos a nivel internacional.

1.3.1.Ámbito internacional

Sobre los sistemas de gestión de proyecto se trae una muestra de algunas de las herramientas cuyas funcionalidades soporten en alguna medida el proceso de atención a solicitudes de revisiones técnicas. Herramientas que sean orientadas a las web, desarrolladas y liberadas bajo licencias que permitan su uso en la UCI.

Collabtive

Collabtive[1]es un software para administración de proyectos basado en la web, publicado como software libre. Es una alternativa *Open Source* ante herramientas propietarias como *Basecamp* y *ActiveCollab*. Fue escrita en PHP5 y para el trabajo con la interfaz de usuario se utilizó AJAX. Es multilenguaje, soportando más de 30 lenguajes, se integra además con otros *webservices*.

Características

- Monitorización de los hitos definidos.
- Permite visualizar el calendario con las tareas definidas en este.
- Permite el intercambio de mensajes de forma instantánea.
- Administración de documentos.
- Permisos basados en roles.
- Permite búsquedas internas.
- Reportes en diferentes formatos (Excel, PDF).
- Capacidad de exportar en diferentes formatos (ZIP, XML, RSS, iCal, vCard).
- Es posible importar XML desde otras herramientas como *Basecamp*.
- Interfaz multilenguaje.

Trac

Trac[2] es una herramienta para la gestión de proyectos y el seguimiento de errores escrita en Python, inspirado en *CVSTrac*. Su nombre original fue *svntrac*, debido a su fuerte dependencia de *Subversion* está desarrollada y mantenida por *Edgewall Software*, y es un software libre de código abierto.

Hasta mediados de 2005 estaba disponible bajo la Licencia Pública General de GNU, pero desde su versión 0.9 se distribuye bajo la Licencia BSD modificada. (La licencia BSD al contrario que la GPL permite el uso del código fuente en software no libre.)

Características

- Permite enlazar información entre una base de datos (BD) de errores de software, un sistema de control de versiones y el contenido de un wiki.

[1] Sitio Oficial: http://collabtive.o-dyn.de

[2] Sitio Oficial: http://trac.edgewall.org/

- Sirve como interfaz web de un sistema de control de versiones como *Subversion, Git, Mercurial, Bazaar o Darcs.*
- Utiliza un sistema de plantillas web propio llamado *Genshi*.

Redmine

Redmine[3] es una aplicación web flexible para la administración de proyectos. Fue implementada bajo el framework *Ruby on Rails*, es multiplataforma, además fue liberado bajo GPL. El diseño de Redmine está significativamente influenciado por Trac.

Características

- Soporta múltiples proyectos.
- Roles flexibles basados en control de acceso.
- Sistema de seguimiento de errores flexible.
- Diagramas de Gantt y calendario.
- Administración de noticias, documentos y archivos.
- Fuentes web y notificaciones por correo electrónico.
- Soporta diferentes SGBD (SQLite, MySQL, PostgreSQL).
- Integración con diferentes sistemas de gestión de configuración de software (SCM) como pueden ser: *Subversion*, *CVS, Git, Mercurial, Bazaar y Darcs*).

De las herramientas caracterizadas anteriormente se pueden mencionar algunas funcionalidades que tienen en común y coinciden además con características que son deseadas para el sistema de atención a solicitudes de revisiones técnicas, estas particularidades son:

- Administración de documentos y archivos
- Sistema de acceso basado en roles
- Generación de reportes
- Notificaciones e-mail.

[3] Sitio Oficial: http://www.redmine.org

No obstante estos sistemas analizados en el ámbito internacional, están diseñados para el apoyo a la gestión de proyectos, brindando funcionalidades que nunca serán usadas para la atención de solicitudes de revisiones técnicas debido a que no se ajustan a este proceso.

1.3.2 Ámbito nacional

En el ámbito nacional la investigación realizada arrojó que las herramientas utilizadas para la gestión o administración de proyectos no son nativas del área nacional sino que toman otras que han sido desarrolladas por otras empresas y comunidades de desarrollo de otros países. Un ejemplo es Redmine, el cual es usado para la gestión de proyectos en instituciones como la UCI.

1.4 Tecnologías a usar

La selección de las tecnologías correctas para el desarrollo de la aplicación en cuestión influye directamente en aspectos fundamentales tales como el tiempo de desarrollo, costo y la calidad del producto final, es por eso que la selección de las herramientas debe estar dada por aspectos tales como el conocimiento de las mismas, para evitar demora por concepto de capacitación. Otro aspecto importante para elegir, puede ser las características o funcionalidades de las herramientas dadas para no afectar la calidad del producto final. Y el último y no menos importante aspecto es el costo, para esto una solución sencilla puede ser el uso de herramientas libres, ya que estas poseen muchas de las funcionalidades que se necesita y demás pueden ser utilizadas sin costo alguno.

1.4.1 Metodologías de desarrollo.

Metodología de desarrollo de software se define como una guía para conducir a un proceso de desarrollo de software al éxito. Una metodología es un proceso, que según Jacobson Booch Rumbaugh [4] define “quién está haciendo qué, cuando, y cómo alcanzar un determinado objetivo”. Se puede concluir sin dudas que el objetivo de las metodologías de desarrollo de software es guiar el proceso de desarrollo de software de un proyecto hasta su completitud, brindando un conjunto de actividades y técnicas que permitan alcanzar los objetivos propuestos, minimizando el tiempo y el costo.[5]

Las metodologías para el desarrollo de software por su parte han necesitado evolucionar cada vez más para la obtención de sistemas más sofisticados dando lugar a metodologías al surgimiento de metodologías capaces de dar soporte a diferentes tipos de desarrollo.

Las metodologías para el desarrollo de software se dividen en dos categorías importantes: Metodologías Ágiles y Metodologías pesadas o tradicionales. A continuación se relacionan algunas de las metodologías de ambas clasificaciones.

- **Ágiles:** XP, Scrum y Crystal.
- **Tradicionales:** Rational Unified Process (RUP), Métrica 3 y OPEN.

Características de XP

XP (eXtreme Programing) programación extrema en español, fue diseñada para dar soporte a proyectos con requisitos difusos y con tendencias a cambios durante el desarrollo, incluso a finales del proceso de desarrollo. XP se caracteriza por ser un proceso ligero ágil y flexible, orientado a los clientes y desarrolladores, por lo que requiere la existencia de una gran comunicación entre los que desarrollan y usan el software que se produce, para lograr una retroalimentación constante que permita que el desarrollo no se aparte de los objetivos propuestos al inicio.

Todo el proceso de desarrollo que provee XP como guía se encuentra basado en cuatro actividades fundamentales: Codificar, Probar, Atender y Diseñar. A partir de estas actividades se definen un conjunto de prácticas. [6]

- Realizar pequeños Releases.
- Diseñar simple.
- Probar – Testear.
- Rearmar – Refactorizar.
- Programar en pares.
- Propiedad Colectiva.
- Integrar Continuamente.
- Cliente *On-Site*.
- Usar Estándares de Codificación

Características de Scrum

Scrum es una metodología de desarrollo de software perteneciente a los procesos de desarrollo ágiles. Se caracteriza por guiar el desarrollo de manera iterativa e incremental, concluyendo cada iteración con la

entrega de una parte pequeña del software. Como la gran mayoría de las metodologías de desarrollo ágil, está diseñada para adaptarse a los cambios de requerimientos además de priorizar aquellas funcionalidades que satisfacen aspectos importantes en el negocio. Esto trae consigo que el equipo de desarrollo conozca las expectativas del cliente hasta el momento con respecto al producto en desarrollo, tenido como referencias resultados tangibles.

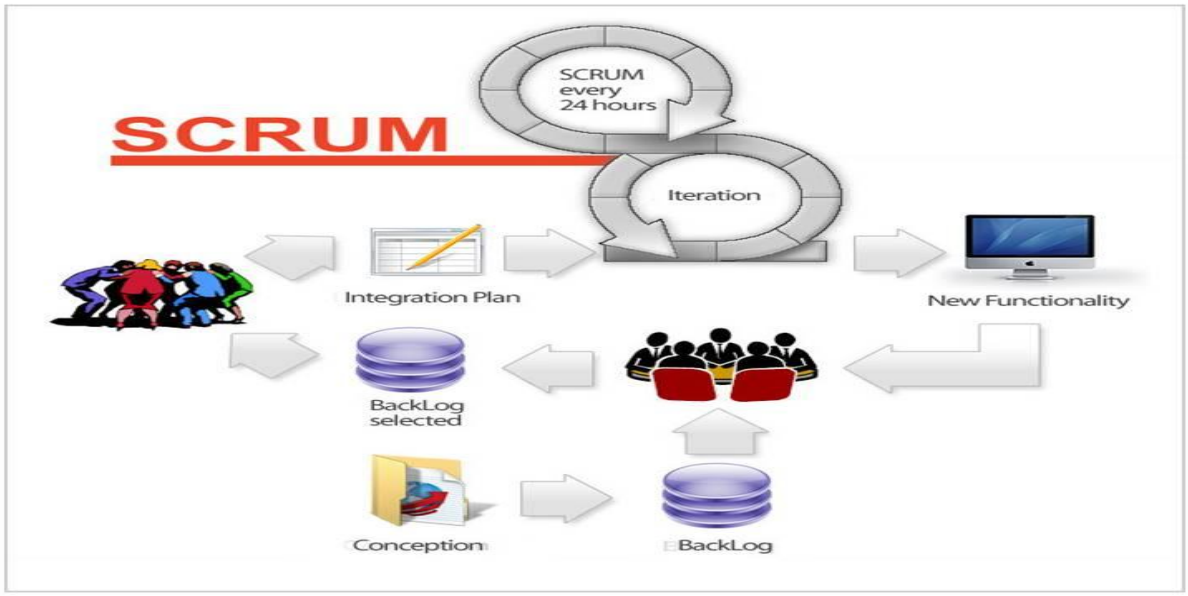


Figura 1 Ciclo de trabajo de Scrum

Principios de Scrum

- Colaboración estrecha con el cliente.
- Predisposición y respuesta al cambio
- Prefiere el conocimiento tácito de las personas al explícito de los procesos
- Desarrollo incremental con entregas funcionales frecuentes
- Comunicación verbal directa entre los implicados en el proyecto
- Motivación y responsabilidad de los equipos por la auto-gestión, auto-organización y compromiso.
- Simplicidad. Supresión de artefactos innecesarios en la gestión del proyecto. [7]

Como ventajas de Scrum es posible destacar la flexibilidad y adaptación respecto a las necesidades del cliente y cambios en el mercado además de la mitigación sistemática de los riesgos del proyecto, todo esto debido a la estrategia de inspección continua.

Características de RUP

RUP es una metodología de desarrollo de software tradicional o pesada, la cual divide su desarrollo en 9 flujos de trabajo, 6 de ingeniería y 3 de apoyo, priorizando cada uno de estos flujos según la fase de desarrollo en la que se encuentre el producto. Estas fases son, Inicio, Elaboración, Construcción y Transición, se definen por cada una los respectivos hitos: obtener visión de los objetivos, obtener un prototipo de la arquitectura, capacidad operacional inicial y liberación del producto. Este proceso puede repetirse hasta alcanzar un *release* definitivo.

Principios de RUP

- **Orientado por Casos de Uso**: Utiliza a los casos de uso (CU) como el orientador de las actividades a realizar, conservando así la consecución de los objetivos planeados. Se basa en las funcionalidades que el sistema debe poseer para satisfacer las necesidades de un usuario (persona, sistema externo) que van a interactuar con el sistema en cuestión.
- **Centrado en la arquitectura**: Establece una analogía con la arquitectura convencional de un edificio, en la que es necesario para tener una vista completa del edificio, realizar varias vistas del mismo edificio solo que desde diferentes aspectos o puntos de vistas, son las llamadas vistas, para más tarde comenzar la construcción.
- **Iterativo e incremental**: Establece una jerarquía de proyectos mayores dentro de los que se encuentran subproyectos, siendo cada subproyecto una iteración donde cada iteración se encargará de englobar y tratar un grupo específico de casos de uso.

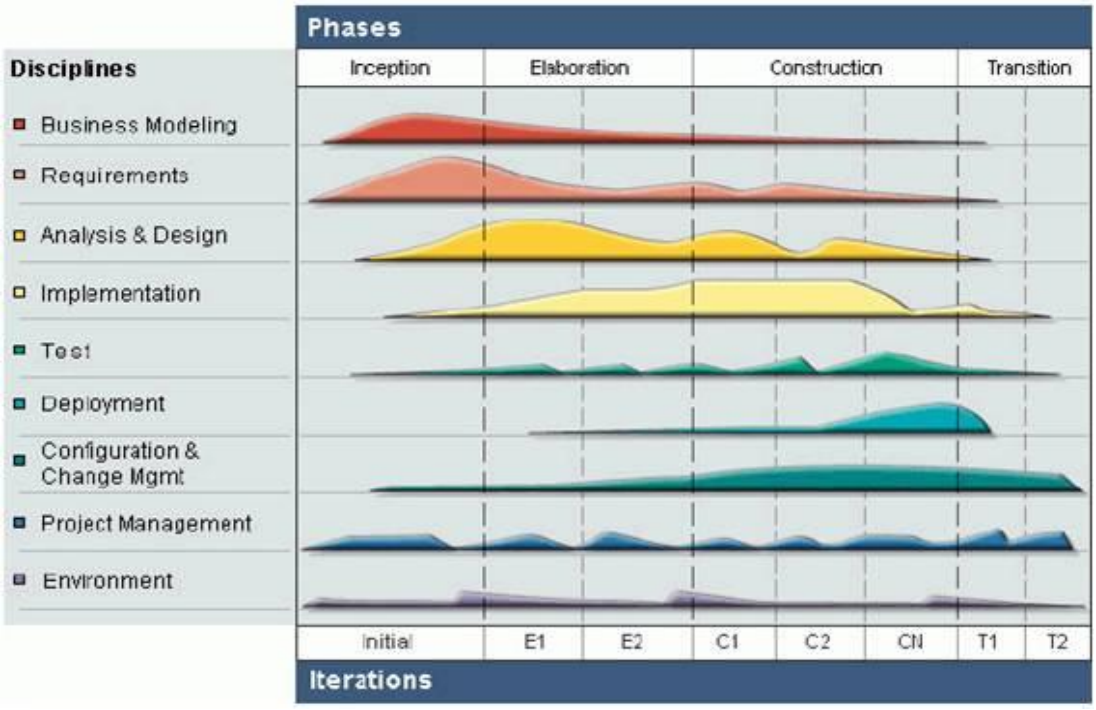


Figura 2 Ciclo de trabajo de RUP

Justificación de RUP

Se selecciona para esta investigación el trabajo con la metodología de desarrollo RUP, por las características antes mencionadas, además de generar una gran cantidad de documentación que generalmente es descriptiva permitiendo la implementación desde un punto de partida mucho más amplio, descrito y pormenorizado que los propuestos por las metodologías ágiles. Conjuntamente a estas razones se encuentra el dominio de esta metodología por parte del equipo de desarrollo, algo que no sucede de la misma manera con otras metodologías.

1.4.2 Herramientas CASE

En su traducción al español CASE significa, Ingeniería de Software Asistida por Computación. Su definición más general, la reconoce como la aplicación de métodos y técnicas a través de las cuales se hacen útiles a las personas comprender las capacidades de las computadoras, por medio de programas, de procedimientos y su respectiva documentación. Centrando la atención en el uso de estas herramientas, para el desarrollo de proyectos informáticos que tengan como objetivo la automatización de procedimientos administrativos; se puede decir que estas herramientas representan una forma que

permite modelar los procesos de negocios de las empresas y desarrollar los Sistemas de Información Gerenciales. [8] Estas herramientas se comportan dinámicamente en el mundo del desarrollo del software, a continuación se explican algunas de las más utilizadas.

Enterprise Architect (EA)

EA provee una generación poderosa de documentos y herramientas de reporte con un editor de plantilla completo WYSIWYG. Genera reportes detallados y complejos de EA con la información que se necesita y en el formato que sea solicitado.

EA soporta generación e ingeniería inversa de código fuente para muchos lenguajes populares, incluyendo: C++, C#, Java, Delphi, Visual Basic y PHP. EA le permite navegar y explorar su modelo de código fuente en el mismo ambiente. Para el trabajo en Eclipse o Visual Studio.Net, Sparx Systems también vende puentes livianos para estas IDE's, permitiendo modelar en EA y generar directamente el código fuente un editor determinado. Las plantillas de generación de código le permiten personalizar el código fuente generado a las especificaciones de su compañía.

Ventajas de EA:

- Diseño y construcción de UML.
- Casos de Uso, Modelos Lógico, Dinámico y Físico.
- Extensiones personalizadas para modelado de procesos y más.
- Documentación de alta calidad compatible con MS Word.
- Intuitivo y simple de usar. [9]

Visual Paradigm

Visual Paradigm es una herramienta visual de Ingeniería de Software para el modelado, brinda una colección de menús, barras de herramientas y ventanas que forman el área de trabajo, lo cual permite crear diferentes tipos de diagramas en un ambiente completamente visual. Está diseñada para una gran variedad de usuarios, incluyendo Ingenieros de Software, Analistas de Sistema, Analistas de Negocios, por nombrar algunos.

Provee una manera intuitiva para llevar a cabo sistemas de análisis y diseño orientado a objetos (OO). Soporta los últimos estándares de anotaciones de Java, UML, provee soporte para la generación de código y la ingeniería inversa para Java, se integra con algunas herramientas de este lenguaje, como: Eclipse, Netbeans, Jbuilder y Oracle. Es una herramienta gratis y multiplataforma. [10]

Justificación de Visual Paradigm

Se seleccionó para la presente investigación, el trabajo con la herramienta Visual Paradigm, además de las características que presenta, que se puede integrar al IDE escogido, el NetBeans. Otro factor que fundamentó esta selección es que facilitará el entendimiento por parte de todos los roles implicados en el desarrollo del software.

1.4.3. Lenguajes de programación

En la actualidad para el desarrollo de sistemas web existen muchos lenguajes de programación. Algunos de los más importantes son PHP, Perl, Ruby, Python y Java por nombrar los más utilizados. La elección de alguno de los mencionados anteriormente se basaría en los conocimientos de los desarrolladores de dicho lenguaje, además de las particularidades de la aplicación a desarrollar.

Características de Java

Java ha sido probado, mejorado y ampliado por una comunidad especializada millones de desarrolladores, Gracias a su versatilidad, eficiencia y portabilidad, permite a los desarrolladores:

- Apegarse estrictamente a la teoría de la programación OO. El polimorfismo y la encapsulación de datos son características intrínsecas de Java.
- Interactuar con los protocolos ya establecidos del conjunto de protocolos TCP/IP, como http y ftp, lo cual permite el acceso a información que no está necesariamente en la misma zona geográfica.
- Java es robusto, puesto que no permite el manejo directo de memoria.
- En Java no hay aritmética de apuntadores y no se permite la conversión de un entero arbitrario en una referencia de objeto.
- Fue pensado para ser un lenguaje de red, por lo que intrínsecamente cuenta con características de seguridad que evitan la infección por virus o trampas de programación, comunes en programas hechos con otros lenguajes.

- Java es dinámico, se adapta con suma facilidad a los cambios en los componentes fundamentales del lenguaje. [11]

Ventajas

- Desarrollar software en una plataforma y ejecutarlo en prácticamente cualquier otra plataforma
- Crear programas para que funcionen en un navegador web y en servicios web
- Desarrollar aplicaciones para servidores como foros en línea, tiendas, encuestas, procesamiento de formularios HTML.
- Combinar aplicaciones o servicios que usan el lenguaje Java para crear servicios o aplicaciones totalmente personalizados.
- Desarrollar potentes y eficientes aplicaciones para teléfonos móviles, procesadores remotos, productos de consumo de bajo coste y prácticamente cualquier tipo de dispositivo digital. [12]

Desventajas

- Tiene como limitante la necesidad de la máquina virtual de Java para la ejecución de los sistemas desarrollados bajo este lenguaje, algo que limita mucho la portabilidad.
- Posee una curva de aprendizaje más elevada frente a otros lenguajes como PHP.
- Algunos de sus *servlets* pueden quedar del lado del cliente, haciéndose necesario el conocimiento avanzado del lenguaje y sus tecnologías para desarrollarlos del lado del servidor

Características de PHP

Ventajas:

- Es multiplataforma, ya que es posible utilizarlo en la gran mayoría de las versiones existentes de plataformas Unix/Linux, Windows y MacOS.
- Su código se encuentra abierto a modificaciones por parte del usuario, o sea es *Open Source*.
- Debido a que su código es abierto los errores que este posee son rápidamente solucionados por la extensa comunidad que lo desarrolla a nivel mundial, además de que las actualizaciones son realizadas muy a menudo sin necesidad de pagar por ellas.

- Es rápido y sin afectar en gran medida el rendimiento del servidor, además de que posee gran interoperabilidad con Apache.
- Es posible utilizarlo también bajo AOLServer y THTTPD.
- Puede utilizar cualquiera de los tres gestores de bases de datos más importantes a nivel mundial Oracle, MySQL y PostgreSQL.

Desventajas:

- Se considera que el código PHP embebido en el documento HTML es problemático debido a la dificultad que existe para mantenerlo y optimizarlo.
- No puede ser compilado.
- No maneja adecuadamente los estándares relacionados con el *Unicode.*

Se pensaría que con semejantes desventajas no se debería usar este lenguaje, sin embargo existen frameworks de desarrollo que permiten solucionar algunos de los problemas que dicho lenguaje posee.

PHP fue elegido, ya que cuenta en la universidad con una comunidad amplia, incluyendo la gran cantidad de documentación existente.

1.4.4 Entornos de desarrollo integrado (IDE)

Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDE pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. Existen una variedad de IDEs, entre los IDE *Open Source* más usados a nivel internacional se encuentran: Eclipse y NetBeans. De ambos IDEs, se seleccionó NetBeans, ya que en su versión 6.9 tiene integración con Symfony. A continuación se muestran algunas características de NetBeans, concernientes al tipo de desarrollo que se desea realizar.

Características de NetBeans

- Creación de Proyectos PHP

NetBeans provee de una estructura para los proyectos, propone un esqueleto para organizar el código fuente, además integra lenguajes como HTML**,** JavaScript y CSS, definitivos en cuanto a desarrollo web se refiere**.**

- Integración con Symfony

La integración del IDE con Symfony se enlaza a los niveles muy bajos, permitiendo así la creación de proyectos sin la utilización de la línea de comandos, así como el autocompletamiento de código, tanto para PHP, como lenguaje como para funciones propias implementadas por el Framework.

- Depuración de PHP

NetBeans utiliza *Xdebug* para inspeccionar y examinar cada variable local, establecer puntos de interrupción y evaluar el código.

NetBeans 6.9 es un IDE integrado con Symfony desde al menos 4 versiones anteriores sin necesidad de usar librerías para el trabajo con el framework, algo que es necesario hacer con Eclipse.

1.4.5 Sistema gestor de BD y servidor Web

Un Sistema Gestor o Manejador de Bases de Datos (SGBD) es un conjunto de programas que permite a los usuarios crear y mantener una BD, por lo tanto, el SGBD es un software de propósito general que facilita el proceso de definir, construir y manipular la BD para diversas aplicaciones. Pueden ser de propósito general o específico.

Entre las clasificaciones más conocidas de un SGBD se encuentran.

Según modelos conceptuales

- Relacional
- Redes
- Jerárquico
- OO

Según número de sitios

- Centralizado.
- Distribuido. [13]

Características de PostgreSQL

Para el desarrollo de la presente investigación se seleccionó PostgreSQL[4], como SGBD, ya que está bajo licencia *Berkeley Software Distribution* (BSD), esta licencia BSD al contrario que la GPL permite el uso del código fuente en software no libre.

Características principales:

- Corre en casi todos los principales sistemas operativos: Linux, Unix, BSD, Mac OS, Beos, Windows.
- Posee una documentación bien organizada, pública y libre, con comentarios de los propios usuarios.
- Comunidades muy activas, varias comunidades en castellano.
- Bajo "Costo de Propiedad Total" (TCO) y rápido "Retorno de la Inversión Inicial" (ROI)
- Altamente adaptable a las necesidades del cliente.
- Soporte nativo para los lenguajes más populares del medio: PHP, C, C++, Perl, Python.
- Soporte de protocolo de comunicación encriptado por SSL
- Extensiones para alta disponibilidad, nuevos tipos de índices, datos espaciales y minería de datos.
- Utilidades para limpieza de la BD (Vacuum)
- Utilidades para análisis y optimización de *Querys*.
- Almacenaje especial para tipos de datos grandes (TOAST)
- El mejor sistema operativo para correr PostgreSQL es BSD y Unix, por su sistema dinámico de I/O (más eficiente que en otros sistema operativo).

Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido, más tarde en otros sistemas de gestión comerciales. PostGreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. A pesar de esto, PostGreSQL no es un sistema de gestión de bases de datos puramente OO.

Servidores web

[4] http://www.postgresql.org

Un servidor web es un sistema para la atención de las peticiones provenientes del navegador de un cliente específico, a través del protocolo HTTP en todas sus variantes. El funcionamiento de los servidores web se puede resumir de la siguiente manera.

1. Espera las peticiones a través del puerto TCP indicado (generalmente 8080)
2. Recibe una petición.
3. Busca el recurso.
4. Envía el recurso utilizando la misma conexión por la que recibió la petición.
5. Vuelve al segundo paso. [14]

Entre los servidores web más conocidos se encuentran Apache, AOLServer y THTTPD. Se seleccionó como servidor de aplicación web Apache por las características que a continuación se relacionan.

Características de Apache

- Apache es un servidor altamente configurable de diseño modular.
- Apache es una tecnología gratuita de código fuente abierto.
- Apache trabaja con gran cantidad de Perl, PHP y otros lenguajes de script.
- Apache funciona en Linux y en otros sistemas de Unix.
- Apache también funciona en Windows.
- Soporte de scripts PHP.
- Soporte de Secured Socket Layer (SSL).
- Soporte para CGI (Common Gateway Interface). [15]

Ventajas:

- Altamente configurable: Cualquier actividad que se necesite realizar con un servidor web se logra con Apache.
- Estabilidad: Ha sido probado por diferentes proyectos, que verifican su funcionamiento y estabilidad.
- Independencia de la plataforma: Está disponible en varias plataformas.

Desventajas:

- Complejidad**:** Puede resultar difícil de configurar incluso para tareas sencillas y por ello muchos novatos le escapan a su uso.

- Formatos de configuración no estándar: Esto dificulta un poco la automatización y el procesamiento de la configuración al no estar basada en formatos más conocidos como el XML.
- Falta de integración**:** Al ser un producto multiplataforma, el servidor no aprovecha al máximo las posibilidades que ofrece el sistema operativo.
- Administración: Como la mayoría de los programas open-source, uno depende de configurar los archivos a mano o tener que instalarse herramientas adicionales para las tareas de administración. Apache viene en una suerte de Kit para armar. [16]

1.4.6. Frameworks para el desarrollo Web

La palabra *framework* deviene del inglés, *frame* (marco) + *work* (trabajo), no obstante en el desarrollo de software, *framework* se refiere a un conjunto de normas y prácticas ya definidas que ayudan a resolver problemas presentes con técnicas aprendidas de problemas anteriores pero de índole similar. Es además una estructura base o arquitectura ya definida para la organización de otros proyectos sobre ella, generalmente provee al nuevo proyecto programas, bibliotecas además de un lenguaje que permita la interoperabilidad entre otros programas así como unir a todos los componentes que componen al proyecto que se desarrolla sobre el *framework* en cuestión. Estos *frameworks* permiten en su gran mayoría, encapsular operaciones complejas en pocas instrucciones, generalmente ayudan al desarrollador, ya que, se puede dedicar a los aspectos más específicos del proyecto que desarrolla puesto que las operaciones más sencillas están implementadas en estos marcos de trabajo.

Para la selección del framework se toma en cuenta principalmente que en su base funcione utilizando PHP. Existen disimiles marcos de trabajo para el desarrollo web basados en este lenguaje, *ZendFramework, CakePHP, CodeIgniter*, Symfony, entre otros.

CodeIgniter[5]

CodeIgniter es un potente framework PHP con un tamaño muy pequeño, construido para programadores PHP que necesitan un conjunto de herramientas sencillas y elegantes para crear todas las funciones de las aplicaciones web.

[5] Sitio Oficial: http://codeigniter.com/

Es un poderoso framework para PHP que facilita la escritura de código repetitivo, y a comparación de otros frameworks cómo Cake PHP, Symfony o ZendFramework, es más rápido pero menos fácil, ya que carece de algunas librerías que los otros frameworks tienen, pero aún así no deja de ser un buen framework además de que es totalmente extensible y altamente compatible con gran variedad de versiones y configuraciones de PHP. [17]

Symfony

Características

- Fácil de instalar y configurar en sistemas Windows, Mac y Linux
- Esta implementado usando PHP5, el cual es superior a sus versiones anteriores, debido a que incluye o trabaja bajo el paradigma de programación OO, fueron agregados modificadores de acceso para los atributos y métodos estos son *private, protected y public.* Funciona con todas las bases de datos comunes (MySQL, PostgreSQL, SQLite, Oracle, MS SQL Server).
- Está basado en el patrón arquitectónico *Modelo Vista Controlador* (MVC), unos de los patrones más eficaces en cuanto a programación web se refiere.
- Compatible solamente con PHP5 desde el año 2006, para asegurar el mayor rendimiento y acceso a las características más avanzadas de PHP.
- Basado en la premisa de *"convenir en vez de configurar"*, en la que el desarrollador solo debe configurar aquello que no es convencional.
- Preparado para aplicaciones empresariales, ya que se puede adaptar con facilidad a las políticas y arquitecturas propias de cada empresa u organización.
- Flexible hasta cualquier límite y extensible mediante un completo mecanismo de *plugins*.
- Publicado bajo licencia MIT de software libre y apoyado por una empresa comprometida con su desarrollo.

Justificación de framework

Se selecciona este *framework* de desarrollo, además de sus características, porque posee una amplia documentación que facilita la comprensión del mismo. El uso de este framework en el desarrollo web

en la UCI es casi extensivo, lo que posibilita tener la experiencia en el desarrollo con Symfony al alcance de la mano.

1.4.7 Arquitectura de Software

Según la IEEE 1471-2000, “Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución”. Se puede definir también, como una vista de alto nivel del sistema, que establece un estilo arquitectónico o la combinación de estos para solucionar un problema específico en el sistema. Además de que se concentra en los requisitos no funcionales. [18]

1.4.8 Patrones arquitectónicos

“Un patrón arquitectónico expresa un esquema estructural fundamental para la organización de sistemas de software. Provee una serie de subsistemas predefinidos, especifica sus responsabilidades, e incluye reglas y guías para la organización de las relaciones entre ellos” [19]

Los patrones arquitectónicos tienen diferentes categorías, entre las que se destacan:

- Patrones Simples: Layers, Tuberías y Filtros, Pizarrón, Repositorio
- Sistemas Distribuidos: Broker (Microkernel, Tuberías y Filtros), CAGS, Cliente-Servidor
- Sistemas Interactivos : Modelo Vista Controlador (MVC), Presentación-Abstracción-Control
- Patrones Adaptables: Reflexión [20]

De los patrones arquitectónicos relacionados anteriormente se seleccionó el patrón MVC. Este es conocido por ser uno de los más utilizados para el desarrollo de aplicaciones web, debido a la capacidad que aporta a la aplicación de separarse en tres componentes, **Modelo** que se encargara del almacenamiento y gestión de los datos, **Controlador** se encargara de manejar la lógica que posee el negocio del proyecto en cuestión y por último la **Vista** será el componente encargado de proveer una interfaz gráfica al usuario desde donde los demás procesos serán transparentes al usuario.

1.4.9 Patrones de diseño

“Un patrón de diseño es una buena práctica documentada o el núcleo de una solución que ha sido aplicada exitosamente en múltiples ambientes para resolver un problema que se repite en un número específico de soluciones”. [21]

A su vez los patrones de diseño se dividen en dos ramas según su tipo: General Responsibility Asignment Software Patterns (GRASP) y Gang of Four (GoF). A continuación se relacionan y explican los patrones de diseño más utilizados.

GRASP

Experto: Consiste en la asignación de la responsabilidad a la clase que posee la información necesaria para llevarla a cabo.
Creador: Asigna a una clase Y la responsabilidad de crear una instancia de la clase X si Y contiene, agrega, registra, tiene los datos de inicialización de X o la utiliza de muy cerca.

Controlador: Consiste en asignar a una clase la responsabilidad del manejo de los mensajes generados por los eventos de un sistema.

Bajo acoplamiento: Asignar una responsabilidad para mantener bajo acoplamiento.

Alta cohesión: Asignar una responsabilidad de modo que la cohesión siga siendo alta. [22]

GoF

Los patrones de diseño GoF se clasifican en tres ramas fundamentales estas son: Creacionales, Estructurales y de Comportamiento.

Creacionales

Fábrica Abstracta / Abstract Factory: Proporcionar una interfaz para la creación de familias de objetos interdependientes o interrelacionados, sin especificar sus clases concretas.

Solitario / Singleton: Garantizar que una clase sólo tiene una única instancia, proporcionando un punto de acceso global a la misma.

Estructurales

Envoltorio / Decorator: Añadir responsabilidades adicionales a un objeto dinámicamente, proporcionando una alternativa flexible a la especialización mediante herencia, cuando se trata de añadir funcionalidades.

Objeto compuesto / Composite: Su objetivo es tratar uniformemente a los objetos simples y compuestos de una estructura jerárquica recursiva. [23]

Para la construcción del diseño se utilizaron de los patrones GRASP: Experto, Controlador, Bajo Acoplamiento, Alta Cohesión. Y se utilizaron de los patrones GoF el Singleton y el Decorator.

1.5 Conclusiones

Durante el desarrollo de este Marco Teórico se abordaron los temas que soportarán y aportarán además a la solución del problema planteado. El cual surge gracias a la ineficacia con que se realiza el proceso de *Solicitudes de Revisión Técnica.* Se plantea que una posible solución sería la creación de un sistema para la gestión de dichas solicitudes, partiendo entonces de la información brindada por el estado del arte de dichas herramientas no fue satisfactoria dado que arrojo como resultado de que existían muchos sistemas de gestión pero todos realizados a la medida de una empresa determinada o para el proceso de gestión de proyectos en general, sin llegar a enfocarse totalmente un proceso parecido al de solicitudes de revisión técnica siendo entonces imposible la reutilización de ningún otro sistema, quedando al descubierto la necesidad de la creación desde cero de un sistema para la gestión de las solicitudes de revisión técnica, eligiéndose entonces como lenguaje de programación PHP5, además de un marco de trabajo para el desarrollo de sistemas web, Symfony. Se agregan a la selección de herramientas el IDE de desarrollo, NetBeans 6.9, que posee la confianza y potencia suficiente para realizar las operaciones necesarias durante el desarrollo del sistema.

Capítulo 2. Características del sistema

Como parte de del proceso de desarrollo de software es necesario comprender el funcionamiento del proceso de atención a solicitudes de revisiones técnicas, además de las posibles mejoras a introducirle. Es necesario identificar los involucrados que de alguna forma intervienen en el negocio, sean personas o no. Para cumplir con este objetivo en el capítulo se realizan varias tareas fundamentales, como son obtener un listado de las funcionalidades que el sistema que se analiza y diseña deberá tener, llamadas comúnmente requisitos funcionales, además se representan las acciones y las relaciones con las personas que las realizan, y lograr así un consenso con el cliente.

2.1 Descripción de negocio

2.1.1 Identificación de actores y trabajadores

Actores

Parte solicitante

Son los encargados de realizar o comenzar la cadena de sucesos por las que pasa el proceso de gestión de solicitudes de revisión técnicas. Comienzan enviando las solicitudes de revisión técnica a la dirección del centro.

Trabajadores

- **Jefe del GATS.**

Es el encargado de asignar las solicitudes de revisión técnica recibidas por el GATS a los revisores técnicos, además de inspeccionar la calidad del trabajo realizado por estos.

- **Revisor Técnico.**

Una vez que recibe la solicitud de revisión técnica y el documento proceden a revisar y emitir el DT para el tipo de documento o artefacto que recibieron.

2.1.2 Identificación de los procesos de negocio

- **Solicitud de Revisión Técnica:** Es un proceso mediante el cual un solicitante realiza una solicitud en la cual requiere una revisión técnica de un artefacto, perteneciente al proceso de desarrollo de software de un proyecto dentro de CEGEL.

- **Asignación de solicitudes de revisión técnica:** Proceso en el cual la gerencia del GATS asigna a los revisores técnicos las solicitudes que llegan al grupo, según las competencias de cada revisor técnico.
- **Realización de la revisión técnica:** Es la acción que realiza el revisor técnico una vez que le ha sido asignada la revisión técnica correspondiente a su competencia, la cual deberá realizar en un tiempo límite.
- **Emisión de DT:** Es en este proceso donde se emite un DT para al artefacto que recibió el revisor técnico, al momento en que le fue asignada la revisión técnica.
- **Realización de seguimiento a las revisiones técnicas:** A través de esta acción la gerencia del GATS verifica el cumplimiento de las revisiones asignadas además de verificar la calidad del trabajo realizado.
- **Solicitud de Taller Informativo:** Este proceso no es más que la solicitud de un encuentro físico entre el solicitante y el revisor técnico asignado, taller en el cual se dan argumentos al solicitante sobre el porqué se ha denegado el artefacto que anteriormente envió.

2.2 Modelo de negocio

EL modelado de negocio en el proceso de desarrollo de software definido por RUP posee un objetivo fundamental para la comprensión del funcionamiento estructural de la organización para la cual se desarrolla determinada solución. Tiene además otra tarea importante y es potenciar que tanto clientes como desarrolladores, alcancen el mismo nivel de comprensión de los problemas que existan en la organización y las posibles mejoras que sobre esta se puedan realizar. A través del modelo de negocio es posible además obtener los requisitos del sistema que la organización solicita.

Con el marcado interés de lograr los objetivos anteriormente expuestos, el modelo de negocio permite definir roles y responsabilidades para cada proceso identificado, apoyándose principalmente en los modelos de caso de uso de negocio y de objetos.

2.2.1 Reglas del negocio.

RN1. El solicitante debe pertenecer obligatoriamente a cualquier proyecto dentro de CEGEL.
RN2. El solicitante es el único capacitado por el negocio para requerir una revisión técnica.

RN3. El revisor técnico es el único encargado de revisar los artefactos que le son asignados y generar un dictamen técnico una vez concluida la revisión técnica.

RN4. La revisión técnica debe ser concluida en un tiempo determinado, el cual es establecido por la gerencia de GATS.

RN5. El solicitante es responsable de reclamar el taller informativo para saber el porqué fue rechazado el artefacto.

RN6. Solo tienen acceso al estado de una revisión técnica la persona que realizó la solicitud de revisión técnica, el revisor técnico que la revisa y la gerencia de GATS que asignó dicha solicitud al revisor.

RN7. El dictamen técnico generado debe ser de carácter confidencial.

RN8. Los revisores técnicos son aquellas personas que por sus competencias en los campos de arquitectura, tecnología y seguridad son considerados especialistas, además de pertenecer al GATS.

RN9. Un dictamen técnico será anulado en caso de que se demuestre en el taller informativo que el revisor cometió un error durante la revisión del artefacto.

RN10. En caso de anular un dictamen técnico, deberá quedar registrada dicha acción en la revisión técnica correspondiente.

2.2.2 Actores del negocio

Es posible definir a un actor del negocio como cualquier agente externo que interactúe con el negocio, este agente podría ser un individuo, organización, máquina e inclusive un sistema de información externo. Es posible identificarlo también como el único rol que se beneficia del negocio.

Actor	Descripción

Solicitante	Es el encargado de iniciar la cadena de sucesos por las que pasa el proceso de gestión de solicitudes de revisión técnicas. Comienza enviando las solicitudes de revisión técnica a la dirección GATS. Puede además solicitar talleres informativos en función del resultado de los dictámenes técnicos generados para cada revisión técnica que solicita.

Tabla 1 Actores del Negocio

2.2.3 Trabajadores del negocio

Un trabajador del negocio puede definirse como una persona, o un grupo de estas, máquinas hasta sistemas informáticos que realicen una determinada acción o tengan responsabilidades dentro del negocio.

Trabajador	Descripción
Gerencia de GATS	Asigna a los revisores técnicos determinada solicitud de revisión.
Revisor Técnico	Emite un dictamen técnico luego de aceptar y realizar la revisión del artefacto presente en la solicitud de revisión técnica que le fue asignada.

Tabla 2 Trabajadores del Negocio

2.2.4 Diagrama de CU del negocio

Los diagramas de CU de negocio son utilizados para hacer visible la relación de los actores con el proceso que se analiza en este caso, proceso de atención a las solicitudes de revisión técnica. A su vez cada CU representa la interacción de un actor con el negocio en espera de obtener algo de este.

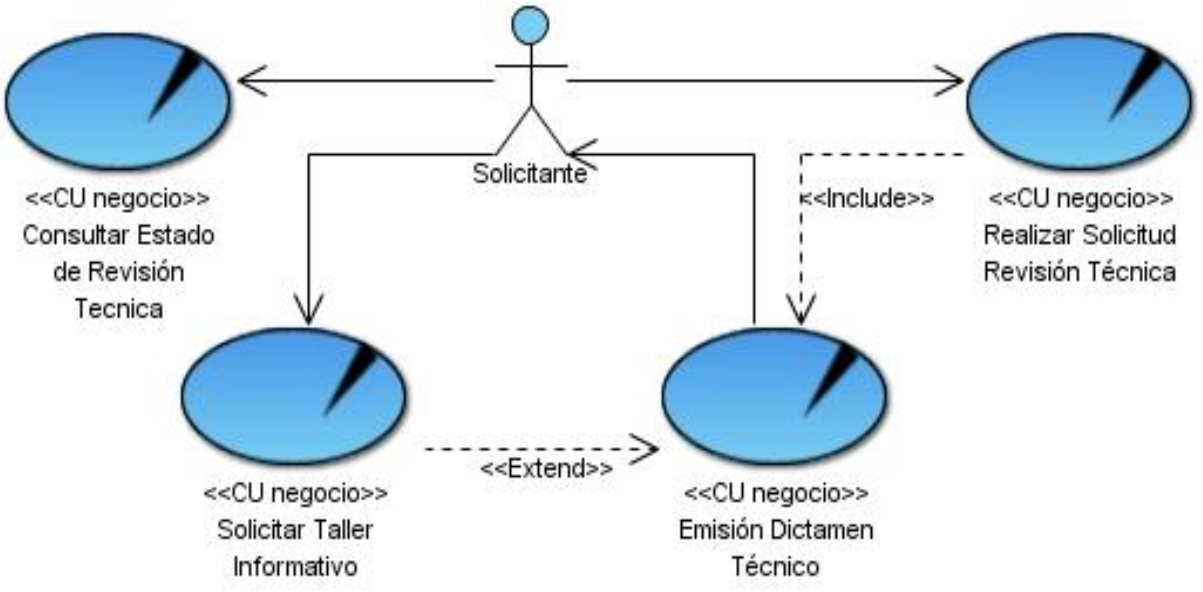


Figura 3 Diagrama del CU del Negocio.

2.2.5 Descripción textual de los CU del negocio

La descripción textual de los CU de negocio persigue comprender cada uno de los procesos y subprocesos del negocio, especifica también la relación entre actores, casos de uso y trabajadores.

Caso de Uso	
CU1:	Realizar Solicitud de Revisión Técnica.
Propósito:	Que el solicitante realice una solicitud de revisión técnica acompañada de un artefacto, para su posterior revisión por parte de un revisor técnico.
Actores:	Solicitante
Resumen:	Este caso de uso inicia cuando el solicitante realiza una petición al GATS de una revisión técnica sobre un artefacto de desarrollo generado en el proyecto productivo donde trabaja el solicitante.
Acción del actor	**Respuesta del Negocio**
1 El solicitante realiza una solicitud de	2 La gerencia del GATS asigna la solicitud a un revisor

revisión técnica al GATS.	técnico específico. 3 El revisor técnico realiza una revisión técnica al artefacto recibido y emite un dictamen técnico para el artefacto enviado y lo hace llegar a la gerencia del GATS. 4 La gerencia del GATS se lo envía al solicitante que exigió la revisión técnica.
5 EL solicitante recibe el DT emitido, para su solicitud de revisión técnica.	

Tabla 3 Descripción del CU: Realizar Solicitud de Revisión Técnica

Caso de Uso	
CU2:	Solicitar Taller Informativo
Propósito:	Solicitar un Taller Informativo en el cual se presenten los argumentos que justifiquen el rechazo del artefacto.
Actores:	Solicitante
Resumen:	Este caso de uso es iniciado por un solicitante después de haber exigido una revisión técnica para determinado artefacto y este fue rechazado.

Acción del actor	**Respuesta del Negocio**
1 El solicitante realiza una solicitud de taller informativo a la gerencia del GATS.	2 La gerencia del GATS hace coordina con el solicitante el lugar y fecha del taller.
3 EL solicitante recibe la especificación del momento y lugar del taller.	

Tabla 4 Descripción del CU: Solicitar Taller Informativo

Caso de Uso	
CU3:	Emisión Dictamen Técnico
Propósito:	Generar un dictamen técnico a partir de la revisión técnica realizada a un artefacto.
Actores:	Solicitante
Resumen:	Se emite un dictamen técnico para informarle al actor el resultado final de la revisión técnica realizada al artefacto sobre el cual solicitó la revisión técnica.

Acción del actor	**Respuesta del Negocio**
	1 El revisor técnico emite un dictamen técnico para el artefacto proveniente de la revisión exigida por el solicitante. 2 La gerencia del GATS envía este dictamen al solicitante.
3 El solicitante recibe el dictamen técnico emitido, en caso de inconformidad con este ver: **CU Solicitar Taller Informativo**.	

Tabla 5 Descripción del CU Emisión Dictamen Técnico

2.2.6 Diagrama de actividades del negocio

El diagrama de actividad es un grafo (grafo de actividades) que contiene estados en los que puede encontrarse una actividad. Un estado de actividad representa la ejecución de una sentencia de un procedimiento, o el funcionamiento de una actividad en un flujo de trabajo. En vez de esperar un evento, como en un estado de espera normal, un estado de actividad espera la terminación de su cómputo. Cuando la actividad termina, entonces la ejecución procede al siguiente estado de actividad dentro del grafo. Una transición de terminación es activada en un diagrama de actividades cuando se completa la actividad

precedente. Un diagrama de actividades puede contener bifurcaciones las cuales especifican el cumplimiento de una condición.

El diagrama de actividad describe un proceso que explora el orden de las tareas o actividades que logran los objetivos del negocio. Es similar a un diagrama de estados en el cual todos o la mayoría de los estados son *estados de actividad* y en la cual todas o la mayoría de las *transiciones* se disparan al completarse las acciones en los estados fuentes precedentes. [24]

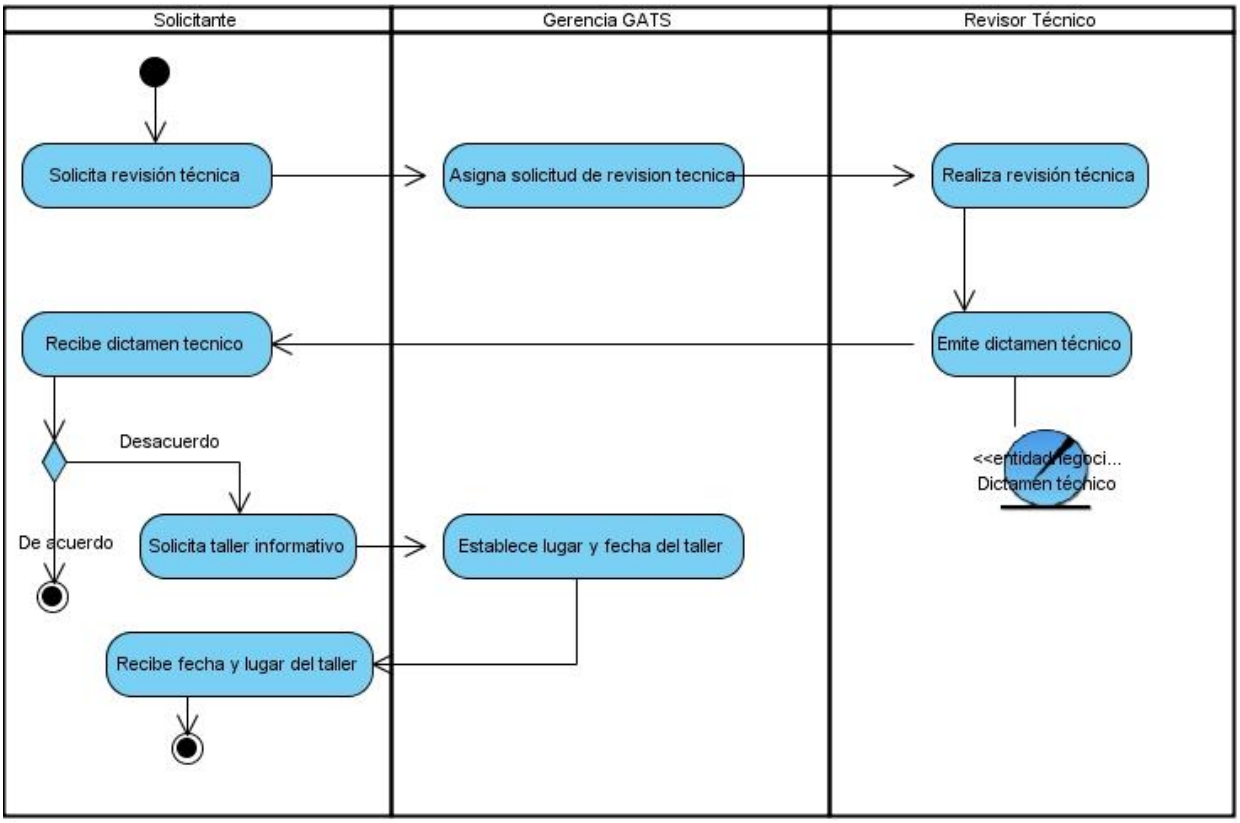


Figura 4 Diagrama de Actividades

2.2.7 Modelo de objetos

En un modelo de objetos se describe la colaboración existente entre los trabajadores y las entidades del negocio dentro del flujo de trabajo modelación de negocio.

Se pueden representar, además de la asociación, los distintos tipos de relaciones entre las entidades de negocio (agregación, composición y generalización / especialización), la cardinalidad y navegabilidad de las

relaciones, pero para efectos de su utilización posterior es suficiente con mostrar la relación entre los trabajadores y estos con las entidades.[25]

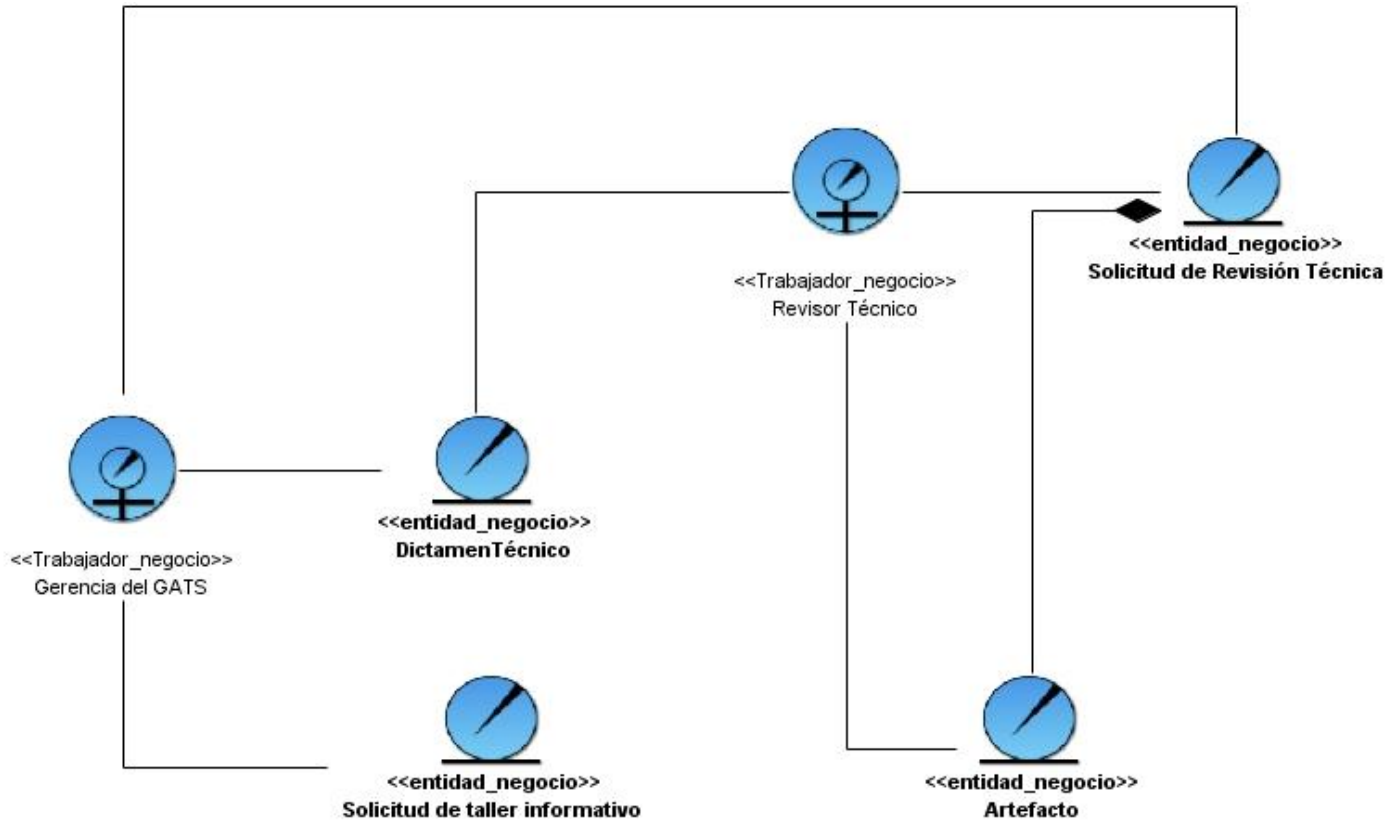


Figura 5 Modelo de Objetos

2.3 Especificación de los requisitos del sistema

2.3.1 Requisitos funcionales del sistema

RF.1. Autenticar usuario.
RF.2. Solicitar revisión técnica.
RF.3. Asignar solicitud al revisor técnico
RF.4. Modificar revisión técnica.
RF.5. Modificar estado de la revisión técnica asignada.
RF.6. Generar un DT.
RF.7. Anular DT.

RF.8. Visualizar revisión técnica.
RF.9. Buscar DT.
RF.10. Solicitar taller informativo.
RF.11. Modificar perfil de usuario.
RF.12. Dar de alta un usuario.
RF.13. Dar de baja un usuario.
RF.14. Buscar usuario.
RF.15. Modificar privilegios.
RF.16. Salvar Log.
RF.17. Visualizar Log.
RF.18. Solicitar prórroga de tiempo.

2.3.2 Requisitos no funcionales del sistema

Los requisitos no funcionales de un software son aquellas funcionalidades que debe tener el producto aún cuando el usuario no las ha solicitado, ya que están enfocadas a los aspectos que convierten al software en un producto confiable, robusto, rápido, usable y atractivo. Generalmente estos requisitos convierten al producto en un éxito.

A continuación se hace referencia a los requisitos no funcionales que el sistema debe cumplir.

RNF1 Usabilidad

Según la norma ISO 9241 “la usabilidad es el rango en el cual un producto puede ser usado por usuarios específicos para alcanzar ciertas metas especificadas con efectividad, eficiencia y satisfacción en un contexto de uso especificado”.

- El sistema deberá mostrar la operación que está realizando en el momento (Procesando solicitud de revisión técnica, generando DT, etc.)
- La interfaz debe permitir operar el sistema de manera fácil e intuitiva, así como brindar ayuda al usuario para las operaciones más complicadas.
- Se debe mostrar un enlace de ayuda según el rol que sirva para la realización de cada operación en el sistema en consecuencia con sus privilegios.

RNF2 Seguridad

A consideración del autor de la presente investigación, la seguridad informática se encuentra basada en 3 principios fundamentales, la Confidencialidad que expresa los intercambios de información entre el usuario que utiliza el sistema y el sistema no deben ser vistos por terceros que no tengan relación con el proceso que el sistema maneja, la Integridad que se encarga de mantener en todo momento la información gestionada ya sea relacionada con el proceso que se atiende o del usuario que utiliza el sistema, sin alteraciones y la Disponibilidad que se refiere a la disponibilidad de la información que se gestiona sin interrupciones; por tanto la gran mayoría de los requisitos no funcionales de seguridad integran dichos principios.

- Almacenar la información en la BD, además de dejar registro de cada operación realizada en ella.
- Serán usados certificados digitales para avalar cada operación de solicitud y emisión dentro del sistema.
- La información almacenada será modificada por los roles establecidos.
- Se realizaran salvas periódicas del sistema y la BD, para la restauración del sistema en caso de fallas.
- Los usuarios deberán tener acceso al sistema en todo momento.

RNF3 De interfaz o apariencia externa

- El sistema deberá poseer una interfaz de usuario lógica e intuitiva.
- EL sistema deberá mostrar las funcionalidades al usuario según el rol con el cual se identificó.

RNF4 De hardware

Para la instalación del sistema se deberá contar con un servidor para la aplicación web con los siguientes requisitos:

Servidor

- 120 GB de disco duro
- 2 GB RAM

Cliente

- 5 GB de disco duro
- 256 MB de RAM

RNF5 De software

Servidor

- Sistema operativo Linux, Windows o MacOS si se desea, ya que el sistema será desarrollado bajo el lenguaje PHP, usando el framework Symfony convirtiéndolo en multiplataforma.
- Servidor de BD PostgreSQL 9.x.

Cliente

- Sistema operativo Linux, Windows XP o superior, o MacOS.
- Navegador Mozilla Firefox preferentemente, aunque es posible utilizar cualquiera de los disponibles en el sistema operativo utilizado por el cliente.

2.4. Solución propuesta

La respuesta para dar solución a la problemática planteada consiste en elaborar el análisis y diseño para la implementación de una aplicación para la gestión y atención a las solicitudes de revisión técnica.

2.4.1 Definición de los actores del sistema

Actor	Descripción
Administrador del Sistema	Establece privilegios de usuarios, gestiona usuarios, y el Log de seguridad del sistema.
Solicitante	Solicita una revisión técnica para determinado artefacto, además de talleres informativos.
Revisor Técnico	Emite un DT luego de aceptar la solicitud de revisión técnica.
Jefe GATS	Asigna a los revisores técnicos determinada solicitud de revisión.

Usuario general	Es una representación de los usuarios pueden interactuar con el sistema. (Jefe GATS, Revisor Técnico, Solicitante, Administrador del Sistema)

Tabla 6 Definición de los actores

2.4.2 Diagrama de CU del sistema

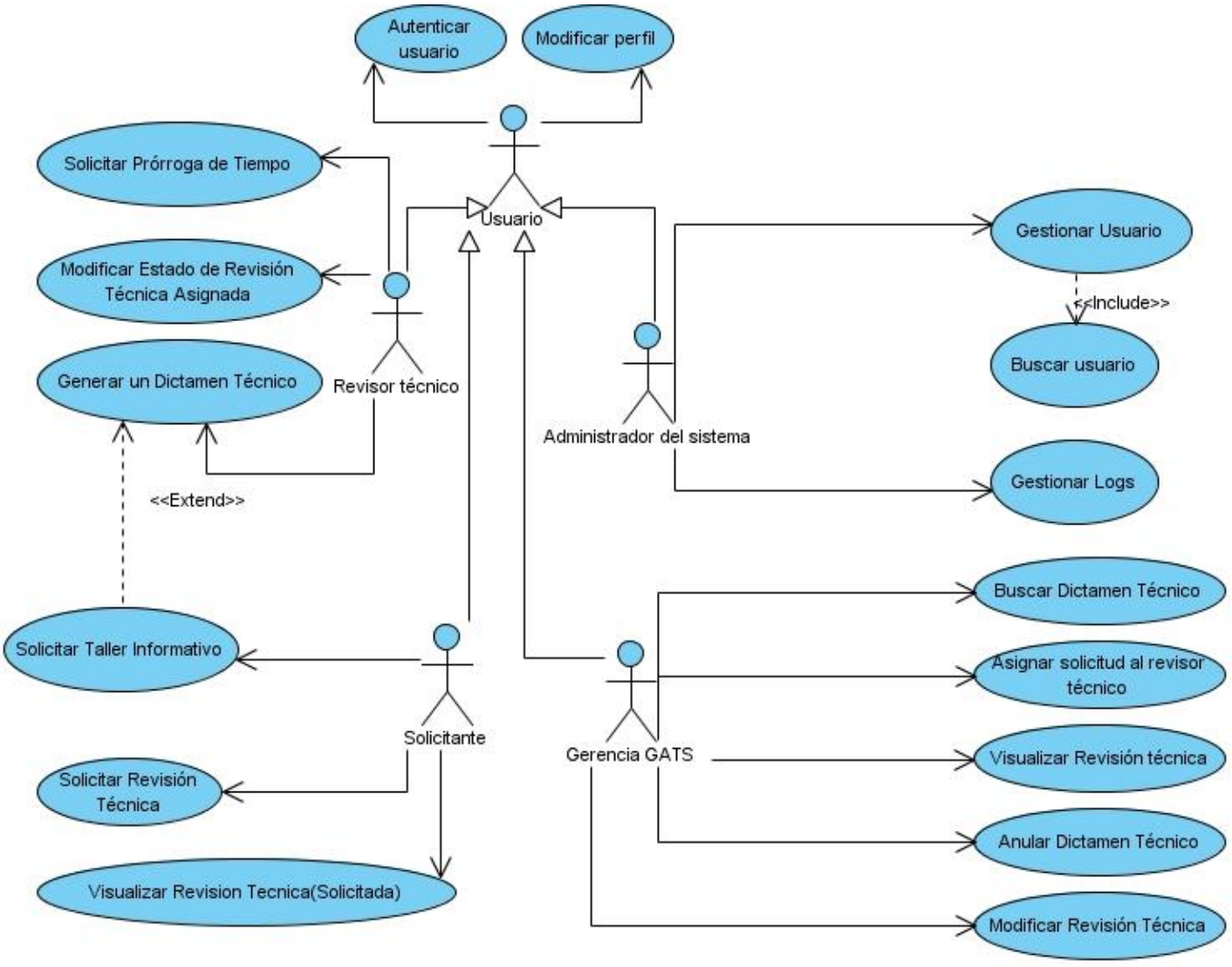


Figura 6 Diagrama de CU del Sistema

Patrones de CU utilizados en el diagrama de CU del Sistema

- Concordancia (Adición), se manifiesta en el CU Solicitar taller informativo. Este patrón es preferible usarlo cuando otros CU se encuentran propiamente completos, o sea, que no requieran de una subsecuencia común de acciones para modificar los usos completos del sistema.

- CRUD (completo), se manifiesta en el CU Gestionar usuario. Este patrón consta de un CU que modela todas las operaciones que puedan ser realizadas sobre una parte de la información de un tipo específico, tales como la creación, lectura, actualización y eliminación; suele ser utilizado cuando los flujos contribuyen al mismo valor del negocio, y estos a su vez son cortos y simples.
- CRUD (parcial), se manifiesta en el CU Gestionar Logs. Este patrón es preferible utilizarlo cuando una de las alternativas de los CU es más significativa que las otras.
- Múltiples actores (Roles comunes), se manifiesta en la generalización/especialización de Usuario y de los usuarios: Solicitante, Gerencia GATS, Administrador del sistema y Revisor técnico. Se utiliza este patrón cuando varios actores jueguen el mismo rol sobre el CU, y entonces se representa ese rol por otro actor heredado por los actores que comparten el rol. Es aplicable cuando, desde el punto de vista del CU, solo existe una entidad externa interactuando con cada una de las instancias del CU.

2.4.3 Casos de Uso Expandidos

CU Autenticar Usuario

Caso de Uso:	Autenticar Usuario
Actores:	Usuario general
Resumen:	El usuario introduce sus datos (usuario y contraseña) y se le permite la entrada al sistema si están correctos.
Precondiciones:	
Referencias	RF 1.
Prioridad	Media
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El usuario accede por medio de la dirección URL a la portada del sistema.	2 El sistema le muestra una ventana con los campos: usuario, contraseña.

3 El usuario introduce sus datos.	4 El sistema verifica que ambos datos se introdujeron. Si no están completos ver Flujo Alterno 1. 5 El sistema comprueba que los datos estén correctos. Si no están correctos ver Flujo Alterno 2.
Flujos Alternos	
Flujo Alterno 1	
Acción del Actor	**Respuesta del Sistema**
	1 El sistema muestra un mensaje: Debe llenar todos los campos. 2 Volver al paso 3 del Flujo Normal de Eventos.
Flujo Alterno 2	
Acción del Actor	**Respuesta del Sistema**
	1 El sistema muestra un mensaje: Sus datos están incorrectos. 2 Volver al paso 3 del Flujo Normal de Eventos.
Pos condiciones	El usuario queda autenticado en el sistema.

Tabla 7 CU Autenticar usuario.

CU Gestionar usuario

Caso de Uso:	Gestionar usuario
Actores:	Administrador del sistema

Resumen:	El administrador tiene las opciones de adicionar, eliminar y establecer privilegios a los usuarios del sistema.
Precondiciones:	El administrador tiene que estar autenticado en el sistema.
Referencias	RF 12, RF 13, RF 14, CU Buscar Usuario (Extendido)
Prioridad	Alta

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El administrador escoge la opción Gestionar Usuario.	2 Muestra una lista de usuarios existentes en el sistema. 3 Brinda las opciones de Adicionar, Eliminar y Establecer Privilegios.
4 El administrador escoge una de las opciones. Si selecciona: • Adicionar Usuario ver Sección 1. • Eliminar Usuario ver Sección 2. • Establecer Privilegios ver Sección 3.	

Sección 1 "Adicionar Usuario"	
Acción del Actor	**Respuesta del Sistema**
	1 Muestra una página con los siguientes campos: nombre y apellidos, y usuario.
2 El administrador del sistema llena los campos.	3 Verifica que no hayan campos en blanco. Si los hay ver Flujo Alterno 1. 4 Validar usuario y contraseña. Si no están correctos ver

	Flujo Alterno 2. 5 Adiciona el Usuario en la base de datos.
Sección 2 "Eliminar Usuario"	
Acción del Actor	**Respuesta del Sistema**
	1 Llama al CU Buscar Usuario mostrando así los datos del usuario a eliminar.
2 El administrador selecciona la opción eliminar.	3 Muestra un mensaje: ¿Desea eliminar el usuario seleccionado? y las opciones: Aceptar o Cancelar. En caso de escoger cancelar se termina el caso de uso. 4 Elimina el usuario de la base de datos.
Sección 3"Establecer Privilegios"	
Acción del Actor	**Respuesta del Sistema**
	1 Muestra los usuarios adicionados.
2 El administrador selecciona la opción asignar rol.	3 Muestra un mensaje: El rol ha sido asignado satisfactoriamente.

Tabla 8 CU Gestionar usuario

CU Buscar usuario

Caso de Uso:	Buscar Usuario
Actores:	Administrador
Resumen:	El administrador inserta un usuario y el sistema muestra sus datos.

Precondiciones:	El administrador tiene que estar autenticado en el sistema.
Referencias	RF 14
Prioridad	Media
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El administrador selecciona la opción Buscar Usuario.	2 Muestra una ventana con el campo usuario.
3 El administrador introduce el usuario a buscar.	4 Muestra los datos del usuario especificado.
Pos condiciones	

Tabla 9 CU Buscar usuario

Los restantes diagramas pueden encontrarse en el Anexo A.

2.5 Conclusiones

En este capítulo se cumplieron varios objetivos importantes para el desarrollo del sistema, entre ellos se obtuvieron los requisitos funcionales del sistema, se logró además, la obtención de un lenguaje común entre el cliente y el equipo de desarrollo a través del modelo de negocio. Y una parte fundamental fue llegar a una propuesta de solución que permitiera el cumplimiento de las necesidades del cliente. Al concluir este capítulo el equipo de desarrollo puede comenzar un análisis más profundo del sistema a desarrollar, además de comenzar con los aspectos fundamentales del diseño.

Capítulo 3. Análisis y diseño del sistema

3.1 Introducción

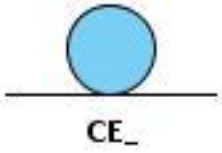
Este capítulo trata de ilustrar la transformación de los requisitos obtenidos anteriormente en una descripción a través de diagramas de cómo deberá ser implementado el sistema. Este objetivo fundamental del análisis y diseño de un sistema se trata de alcanzar a través de los diagramas de clases del análisis, para mostrar las relaciones entre las clases que se han obtenido y que interactúan aquí, a partir de los diagramas enunciados anteriormente, se muestran los diagramas de secuencia que estas generan para que pueda ser observado el orden de las acciones en los casos de uso obtenidos al momento en el que son realizados por el usuario que los invoca. Se ilustra también a través del diagrama de clases del diseño de cada caso de uso, el patrón arquitectónico que se escogió para desarrollar el sistema.

3.2 Modelo del análisis

El objetivo del modelo de análisis en el proceso de desarrollo de software no es más que alcanzar una visión de los requisitos funcionales en un lenguaje orientado al desarrollador final que deberá implementar la solución, en fin concretar una visión mucho más técnica que las descripciones anteriores. Al término del modelo de análisis se puede llegar al diseño de la solución de manera más fácil y moderada posible, permitiendo alcanzar un dominio general del funcionamiento que deberá tener el sistema.

Se representan a continuación varios diagramas de clases del análisis los cuales pertenecen a algunas de las funcionalidades más importantes que deberá tener el sistema, las restantes podrán ser vistas en el anexo correspondiente.

Estereotipos de representación en el modelo de análisis.

Nombre	Características	Representación
Clase Entidad	Modela la información del sistema y el comportamiento asociado a una información que por lo regular es persistente.	

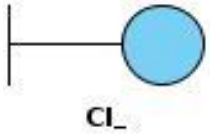
Clase Interfaz	Modela la interacción entre el Actores y el Sistema, como las ventanas y formularios.	
Clase Control	Esta clase coordina el trabajo de las otras clases, encapsulan el comportamiento de un CU, y sus funciones son complejas.	

Tabla 10 Estereotipos del Análisis.

3.2.1 Diagramas de clases del análisis

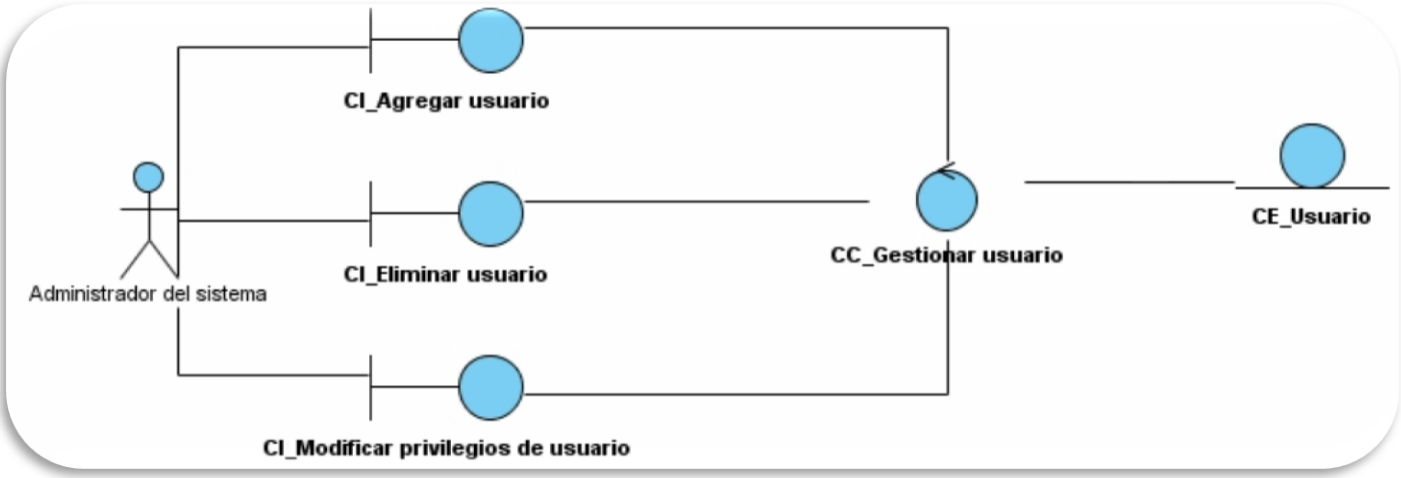


Figura 7 Diagrama de clase de análisis: CU Gestionar usuario

Figura 8 Diagrama de clase de análisis: CU Asignar solicitud al revisor técnico

Figura 9 Diagrama de clase de análisis: CU Modificar dictamen técnico

Figura 10 Diagrama de clase de análisis: CU Modificar estado de revisión técnica

Figura 11 Diagrama de clase de análisis: CU Generar dictamen técnico

Figura 12 Diagrama de clase de análisis: CU Realizar revisión de solicitud

Figura 13 Diagrama de clase de análisis: CU Solicitar revisión técnica

3.2.2 Diagramas de Interacción.

Los diagramas de interacción son utilizados para modelar o ilustrar el dinamismo que determinado proceso posee. Dejando entrever su comportamiento, utilizando para esto estereotipos que representan interfaces, nodos, componentes, entre otros. Se representan además las relaciones que dichos estereotipos tienen unos con otros, relaciones que poseen un significado especifico dentro del proceso que se representa, con la inserción de los mensajes que explican la naturaleza de la relación. Existen en RUP dos tipos de diagramas de interacción.

Diagrama de Secuencia y **de Colaboración**.

- Diagrama de Secuencia muestra las relaciones entre objetos de manera secuencial, en un espacio determinado de tiempo.
- Diagrama de Colaboración representa además de las relaciones existentes entre ellos, los mensajes que intercambian así como la estructura que poseen.

3.2.3 Diagramas de Secuencia del Análisis

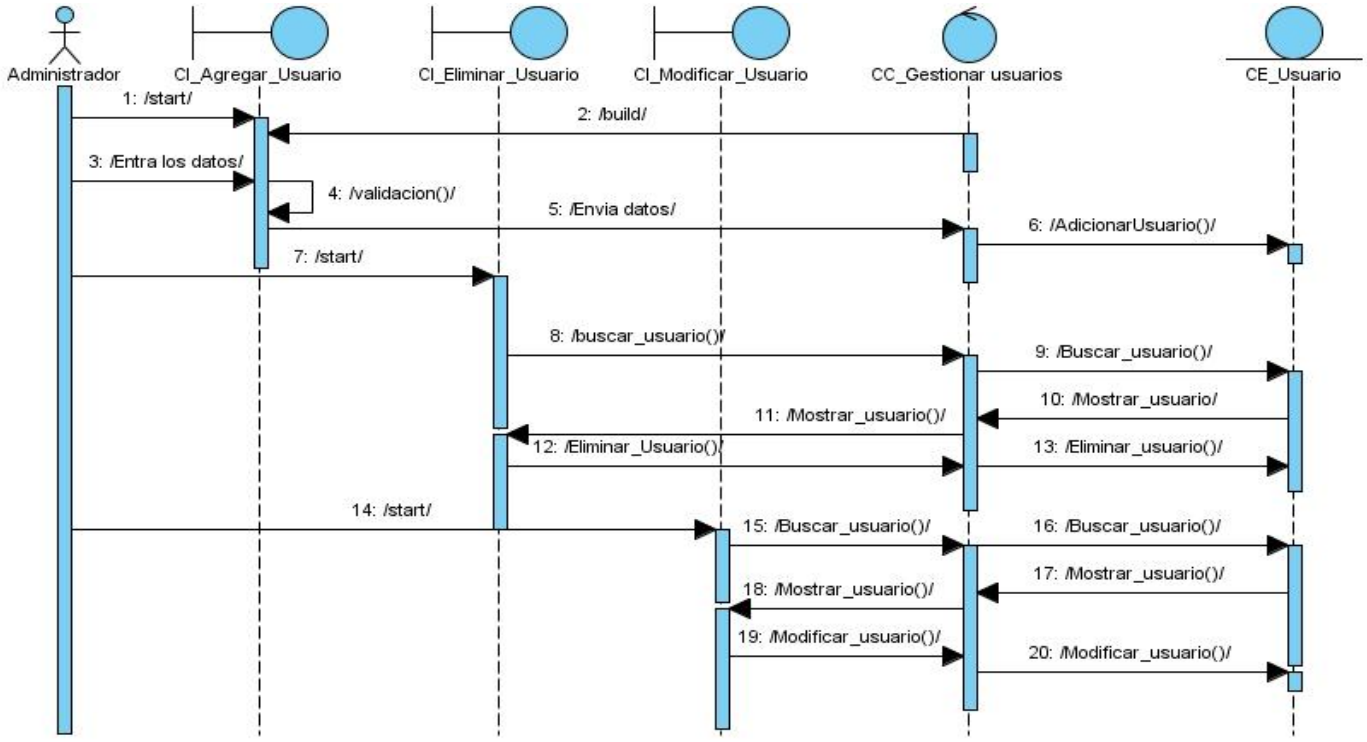


Figura 14 CU Gestionar Usuario

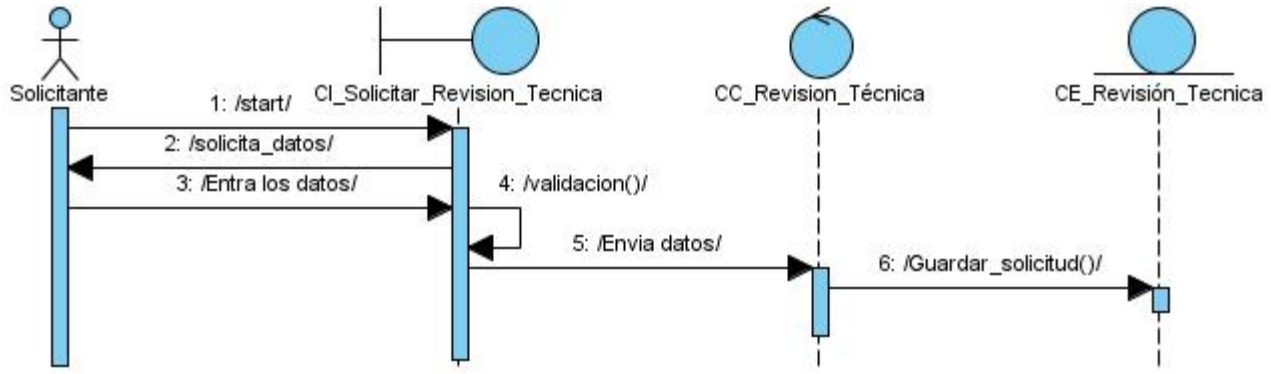


Figura 15 CU Solicitar Revisión Técnica

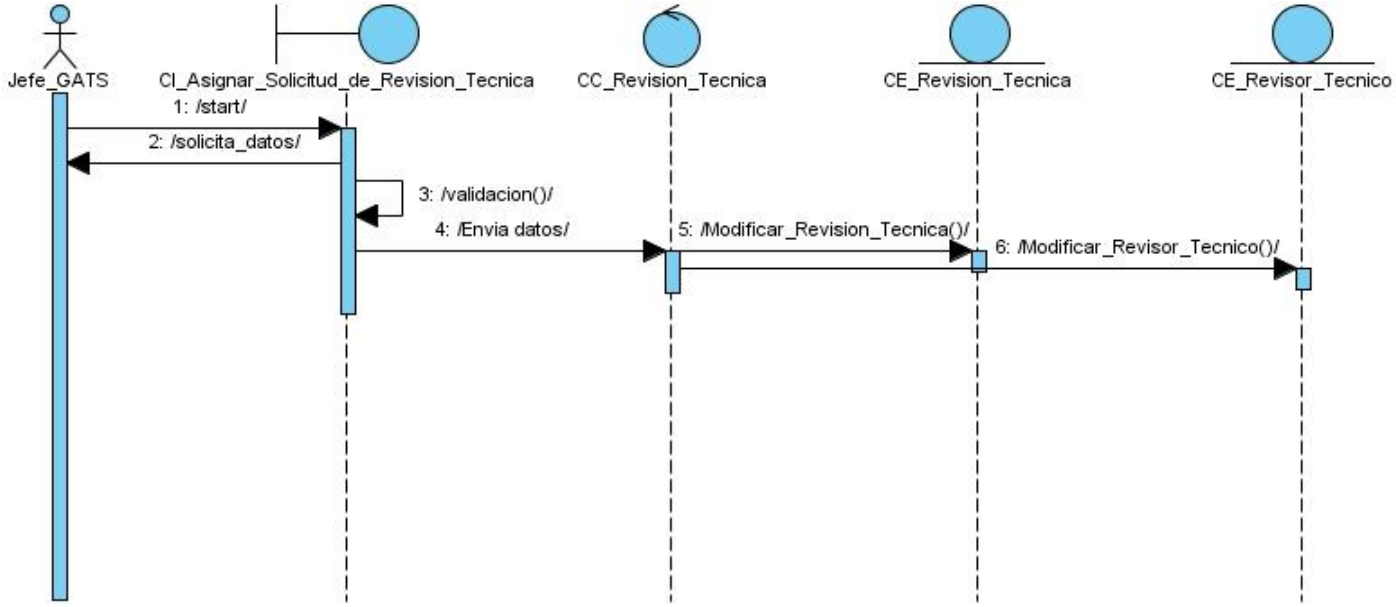


Figura 16 CU Asignar Solicitud de Revisión Técnica

Los restantes diagramas se pueden encontrar en el Anexo B.

3.2.4 Validación del Análisis

Al término del análisis es posible confirmar que el número de diagramas de clases del análisis asciende a 14, y 7 de ellos con CU críticos. Así sucede también con los diagramas de secuencia del análisis donde cada uno corresponde a los casos de uso del sistema definidos en el Capítulo 2.

Antes de proseguir con el diseño se hace necesaria la validación del análisis para evitar la inclusión de errores cometidos aquí en un diseño posterior. Según Sommerville. La validación responde a la pregunta de que si ¿estamos construyendo el software correcto? Algo sin duda vital para el cumplimiento de los requisitos funcionales planteados por el cliente.

Matrices de trazabilidad

Son utilizadas para mostrar la relación existente entre dos tipos de objetos, en este caso para la relación entre los casos de uso de sistema con los diagramas de clases del análisis y los diagramas de interacción.

Casos de Uso del Sistema (CUS)

- CUS 1. Autenticar usuario
- CUS 2. Solicitar revisión técnica
- CUS 3. Asignar solicitud al revisor técnico
- CUS 4. Visualizar revisión técnica
- CUS 5. Modificar estado de revisión técnica
- CUS 6. Modificar revisión técnica
- CUS 7. Generar un dictamen técnico
- CUS 8. Buscar dictamen técnico
- CUS 9. Anular Dictamen Técnico
- CUS 10. Visualizar Revisión Técnica Solicitada
- CUS 11. Modificar perfil de usuario
- CUS 12. Gestionar usuario
- CUS 13. Gestionar Logs
- CUS 14. Solicitar prorroga de tiempo
- CUS 15. Buscar usuario

- CUS 16. Solicitar taller informativo

Diagrama de clases del Análisis	Casos de Uso(CU)															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DCA-Autenticar_Usuario	x															
DCA-Solicitar_Revisión_Técnica		x														
DCA-Asignar_Solicitud_al_revisor_técnico			x													
DCA-Visualizar_Revisiones_Técnicas				x												
DCA-Modificar_Estado_de_Revisión_Técnica					x											
DCA-Modificar_Revision_Técnica						x										
DCA-Generar_Dictamen_Técnico							x									
DCA-Buscar_Dictamen_Técnico								x								
DCA-Anular_Dictamen_Técnico									x							
DCA-Visualizar_Revision_Técnica Solicitada										x						
DCA-Modificar_Perfil											x					
DCA-Gestionar_Usuario												x				
DCA-Gestionar_Logs													x			
DCA-Solicitar Prorroga de tiempo														x		
DCA-Buscar_Usuario															x	
DCA-Solicitar_Taller_Informativo																x

Figura 17 Matriz de trazabilidad CUS contra DCA

Diagrama de Secuencia del Análisis	Casos de Uso(CU)															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DSA-Autenticar_Usuario	x															
DSA-Solicitar_Revisión_Técnica		x														
DSA-Asignar_Solicitud_al_revisor_técnico			x													
DSA-Visualizar_Revisiones_Técnicas				x												
DSA-Modificar_Estado_de_Revisión_Técnica					x											
DSA-Modificar_Revision_Técnica						x										
DSA-Generar_Dictamen_Técnico							x									
DSA-Buscar_Dictamen_Técnico								x								
DSA-Anular_Dictamen_Técnico									x							
DSA-Visualizar_Revision_Técnica Solicitada										x						
DSA-Modificar_Perfil											x					
DSA-Agregar_Usuario												x				
DSA-Eliminar_Usuario												x				
DSA-Modificar_Privilegios												x				
DSA-Salvar_Logs													x			
DSA-Visualizar_Logs													x			
DSA-Eliminar_logs													x			
DSA-Solicitar Prorroga de tiempo														x		
DSA-Buscar_Usuario															x	
DSA-Solicitar_Taller_Informativo																x

Figura 18 Matriz de trazabilidad CUS contra DCA

Teniendo en cuenta que cada caso de uso del sistema representa al menos un requisito funcional y que cada uno de los diagramas tanto de clases como se secuencia del análisis responde como mínimo a un caso de uso del sistema es posible comenzar el diseño, satisfaciendo cada uno de los casos de uso definidos a partir de los requisitos funcionales del sistema.

3.3 Modelo del Diseño

El modelo de dominio tiene como misión principal describir la forma en la que se realizan los casos de uso a través de un modelo de objetos, acercándose más a la implementación o el código fuente final. Se usa además como una entrada a las actividades de implementación y prueba, ya que este está compuesto por artefactos que abarcan todos los paquetes, subsistemas, colaboraciones, clases y relaciones entre estas últimas.

3.3.1 Arquitectura y Diseño con Symfony como *framework* de desarrollo

El desarrollo de aplicaciones Web a través de Symfony como framework de desarrollo está dado por la combinación de varios factores importantes dentro el proceso de desarrollo de software: calidad, rapidez, seguridad entre otros factores. Esto se debe a la adición por parte de este framework de una nueva capa

por encima de PHP, compuesta por herramientas que simplifica el desarrollo de aplicaciones web complejas.

Patrones Arquitectónicos a través de Symfony

Symfony basa su implementación en uno de los patrones arquitectónicos más conocidos en el desarrollo de aplicaciones Web: el patrón Modelo-Vista-Controlador (MVC), el cual se encuentra formado por tres capas o niveles específicos.

El **Modelo** representa la información que se maneja en el negocio además de encapsular las operaciones que se realizan sobre esta.

La **Vista** tiene como función fundamental representar la información perteneciente al modelo a través de una página web para el usuario que interactúa con ella.

El **Controlador** es el mediador entre las capas anteriores, procesa las peticiones del usuario, realizando los cambios pertinentes en el Modelo y la Vista para resolver dichas peticiones.

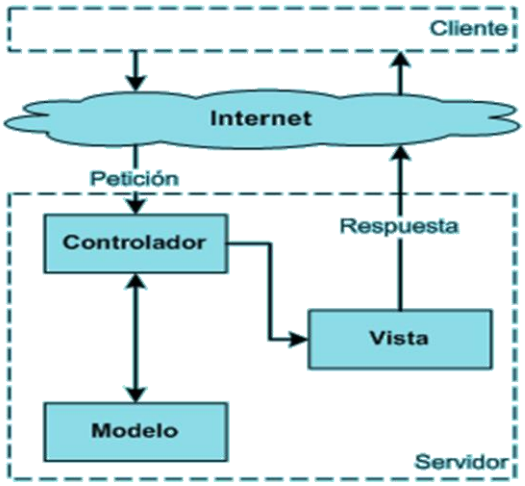


Figura 19 Representación del MVC por Symfony

Symfony establece además una estructura básica para toda aplicación que se desarrolle a través de él. Para esto genera varios componentes generales para todo el proyecto, cada cual en el nivel correspondiente a las operaciones que realiza.

Figura 20 Implementación del MVC por Symfony

En el Modelo se encuentran:

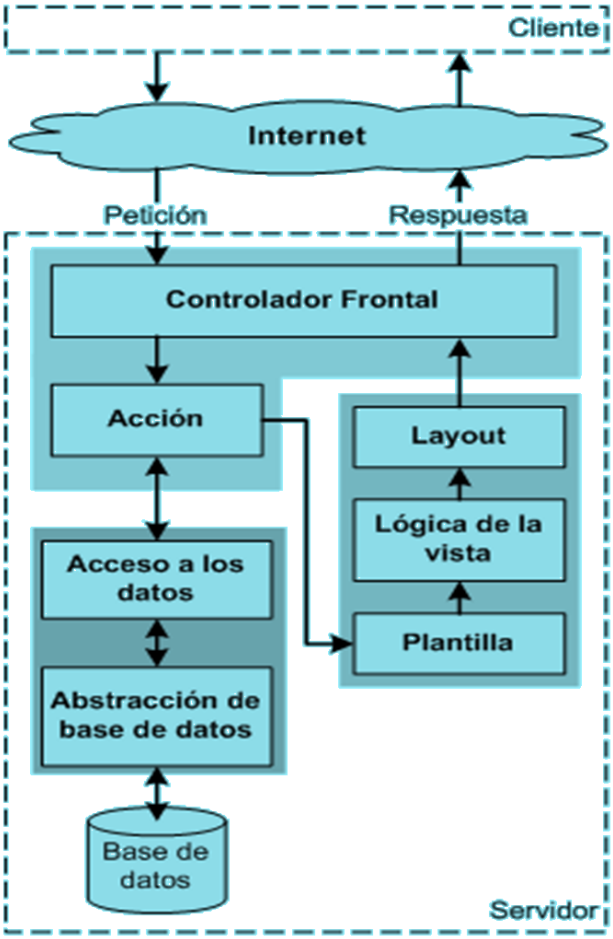
- El componente de abstracción de la base de datos.
- Y el de acceso a datos

La Vista posee:

- Vista
- Plantilla
- Layout

En el Controlador tiene:

- Controlador frontal
- Acción



Paquetes de Diseño con MVC

Utilizando el patrón MVC es posible la organización de las clases y relaciones que se manejan durante el proceso de diseño en la metodología RUP. Se podrán separar cada una de estas clases y relaciones en paquetes según sus responsabilidades. Para garantizar la similitud de cada diagrama de diseño con el patrón arquitectónico bajo el cual se construye el sistema, cada paquete llevará el nombre de uno de los niveles pertenecientes al patrón además, dentro de cada cual se podrán encontrar cada uno de los elementos asociados al paquete según las operaciones que realiza.

Patrones de diseño a través de Symfony

Durante la confección del diseño se utilizan diferentes tipos de patrones como pueden ser los Arquitectónicos, GoF y GRASP. Se podría definir que “un patrón de diseño provee un esquema para el refinamiento de subsistemas o componentes de un sistema de software o las relaciones en ellos. Describe una estructura recurrente común de componentes de comunicación que resuelven un problema general de diseño, sin un contexto particular”[26]

Decorador

Como patrón estructural Symfony implementa posiblemente uno de los más usados en cuanto a desarrollo Web se refiere, Decorador (o *decorator* en inglés) responde a la necesidad de añadir dinámicamente funcionalidad a un objeto o clase. Esto nos permite no tener que crear sucesivas clases que hereden de la primera incorporando la nueva funcionalidad, sino otras que la implementan y se asocian a la primera.

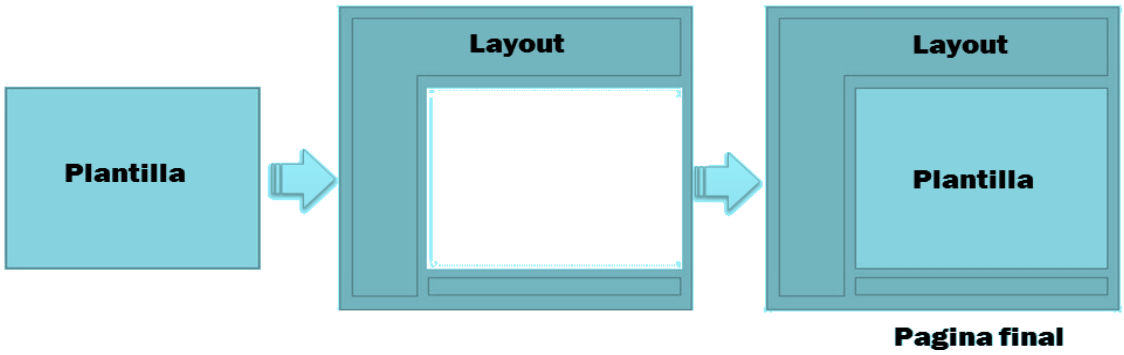


Figura 21 Representación del patrón de diseño *Decorator* por Symfony

Singleton

Sin embargo en la categoría de los patrones creacionales Symfony utiliza el Singleton. Este patrón es utilizado para garantizar que determinada clase posea solo una instancia de ella, proporcionando para sí un solo punto de acceso, o un punto global de acceso.[27]

3.3.2Diagramas de clases del diseño

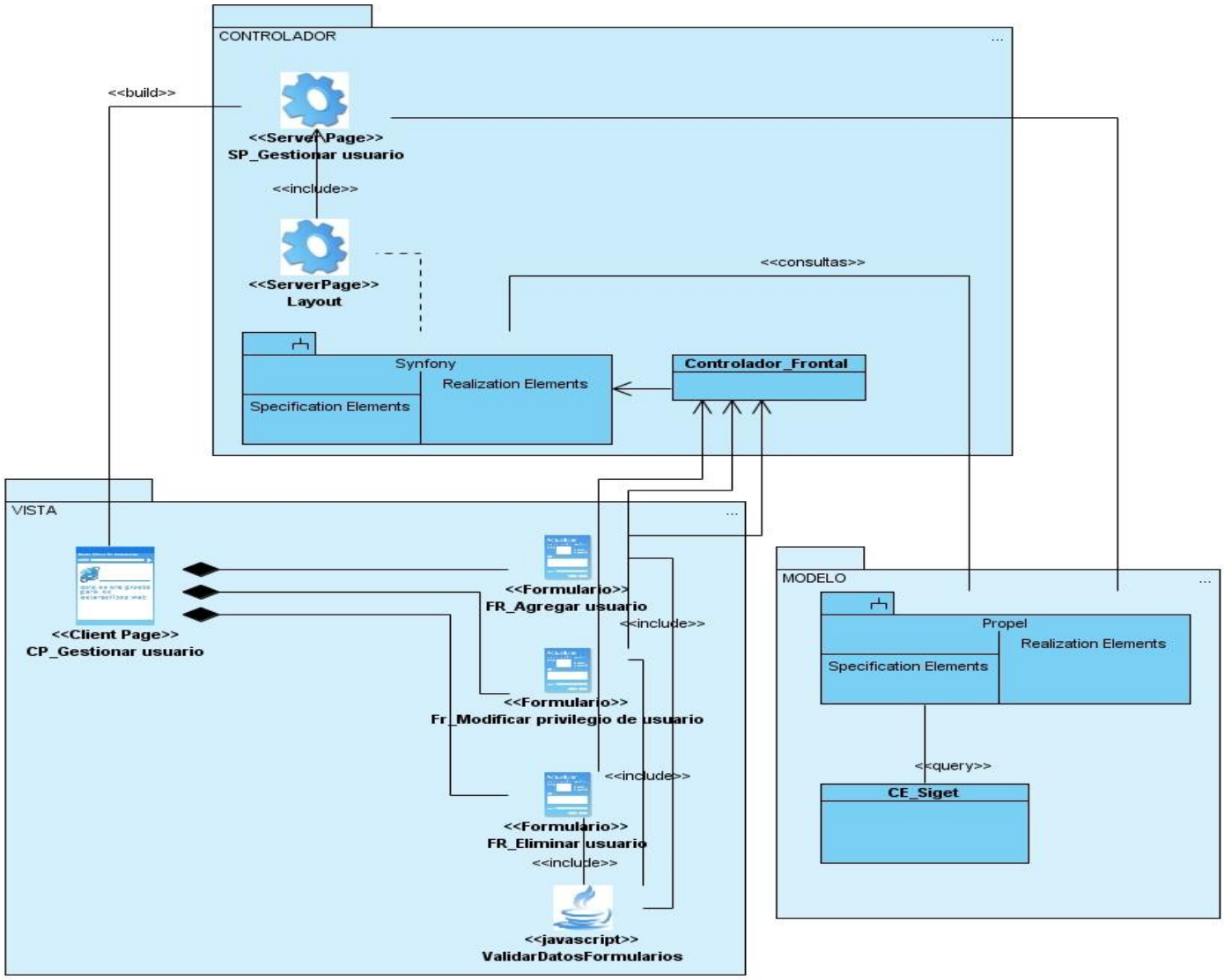


Figura 22 Diagrama de clase del diseño CU

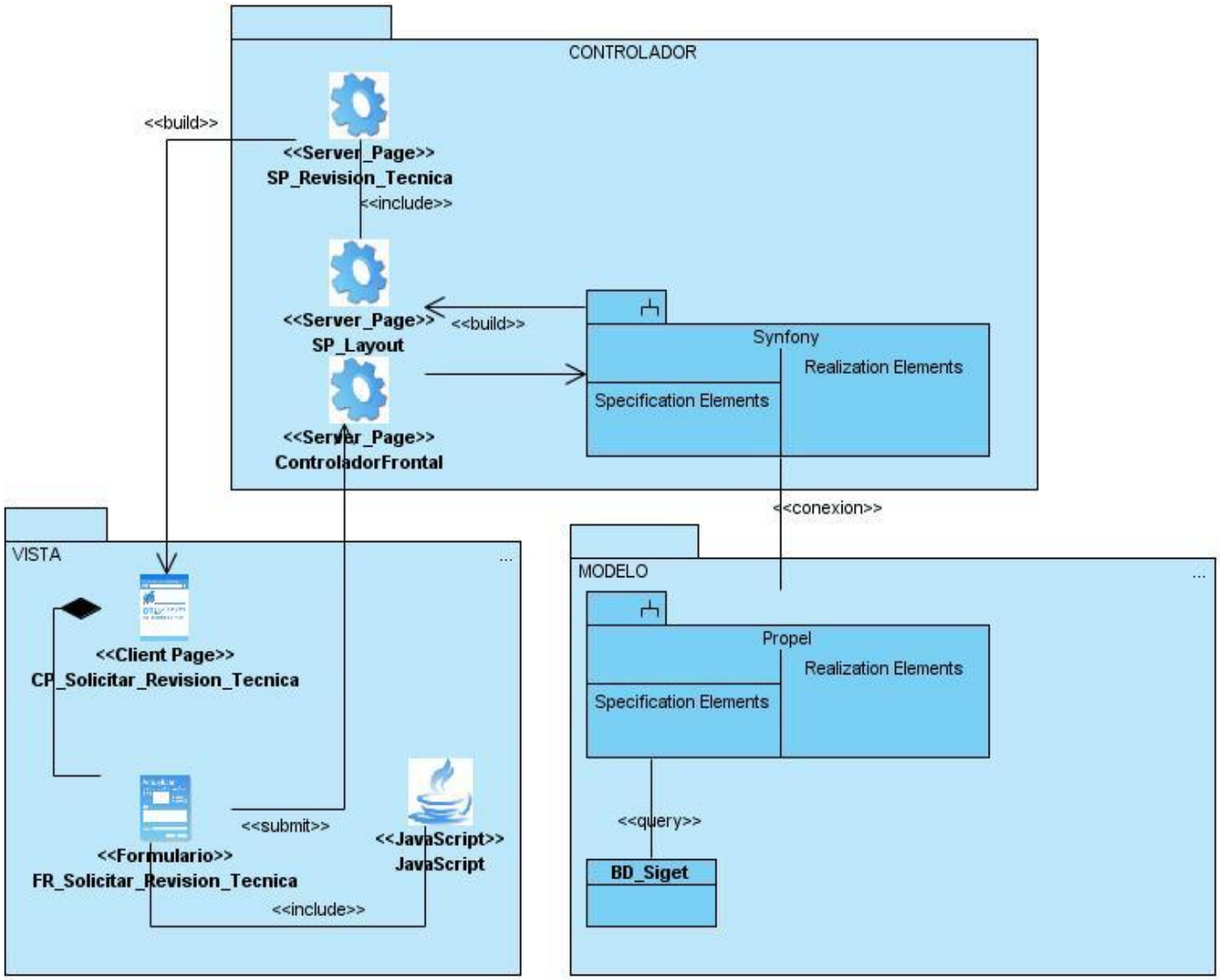


Figura 23 Diagrama de clase del diseño CU Solicitar Revisión Técnica

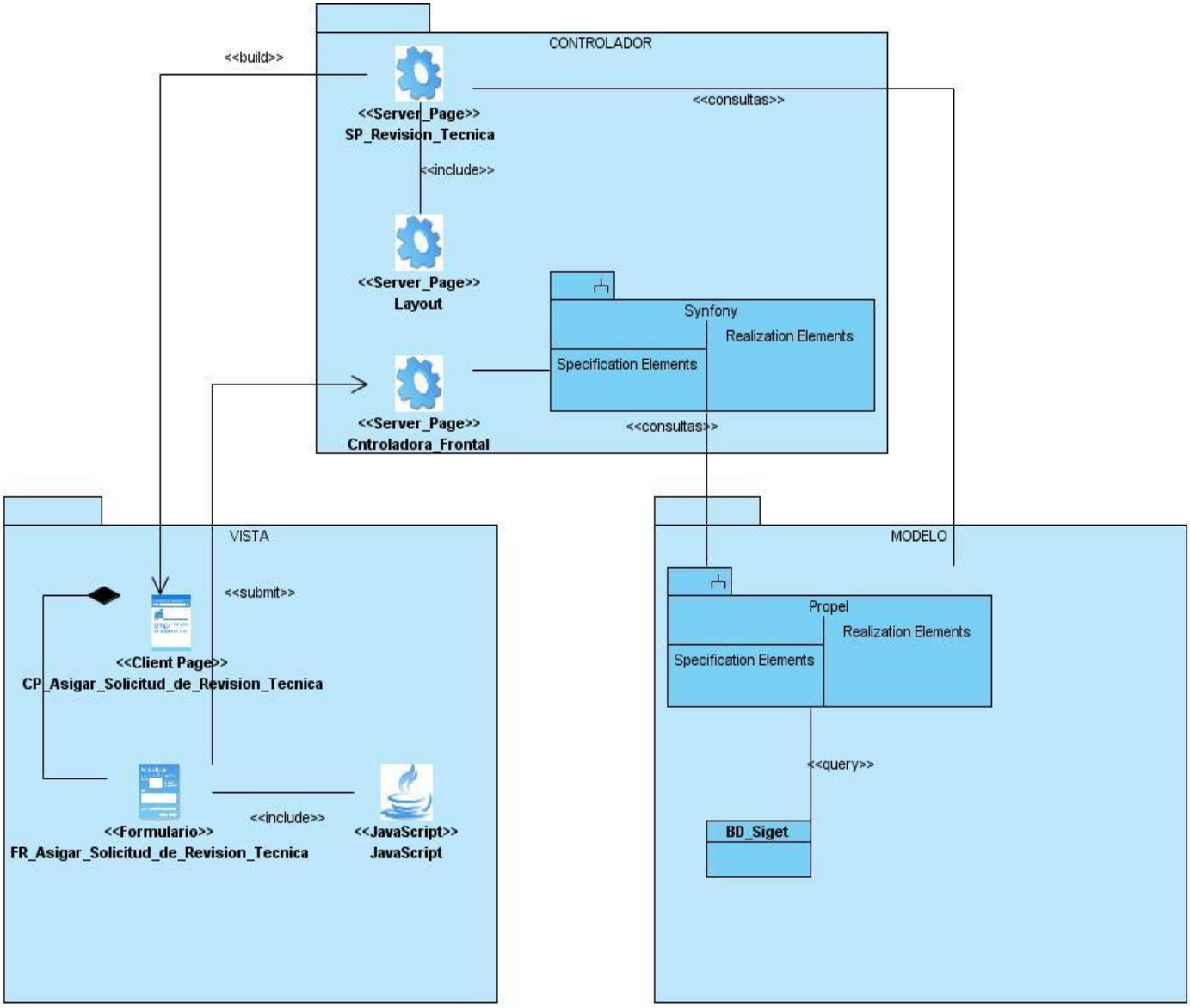


Figura 24 Diagrama de clase del diseño CU Asignar Solicitud de Revisión Técnica

Los restantes diagramas se pueden encontrar en el Anexo C.

3.3.3 Diagramas de secuencia

Un diagrama de secuencia tiene como objetivo principal mostrar al desarrollador la interacción de los objetos que deberán componer al sistema, especificando el orden en el cual deberá realizarse cada

interacción en un momento o instante dado, generalmente durante la realización de determinado caso de uso. Su importancia está determinada por la semejanza que llega a tener con el funcionamiento real del sistema, sirviendo como apoyo durante la implementación.

A continuación se pueden observar algunos de los diagramas de secuencia pertenecientes a las funcionalidades más importantes que el sistema que se diseña deberá cumplir.

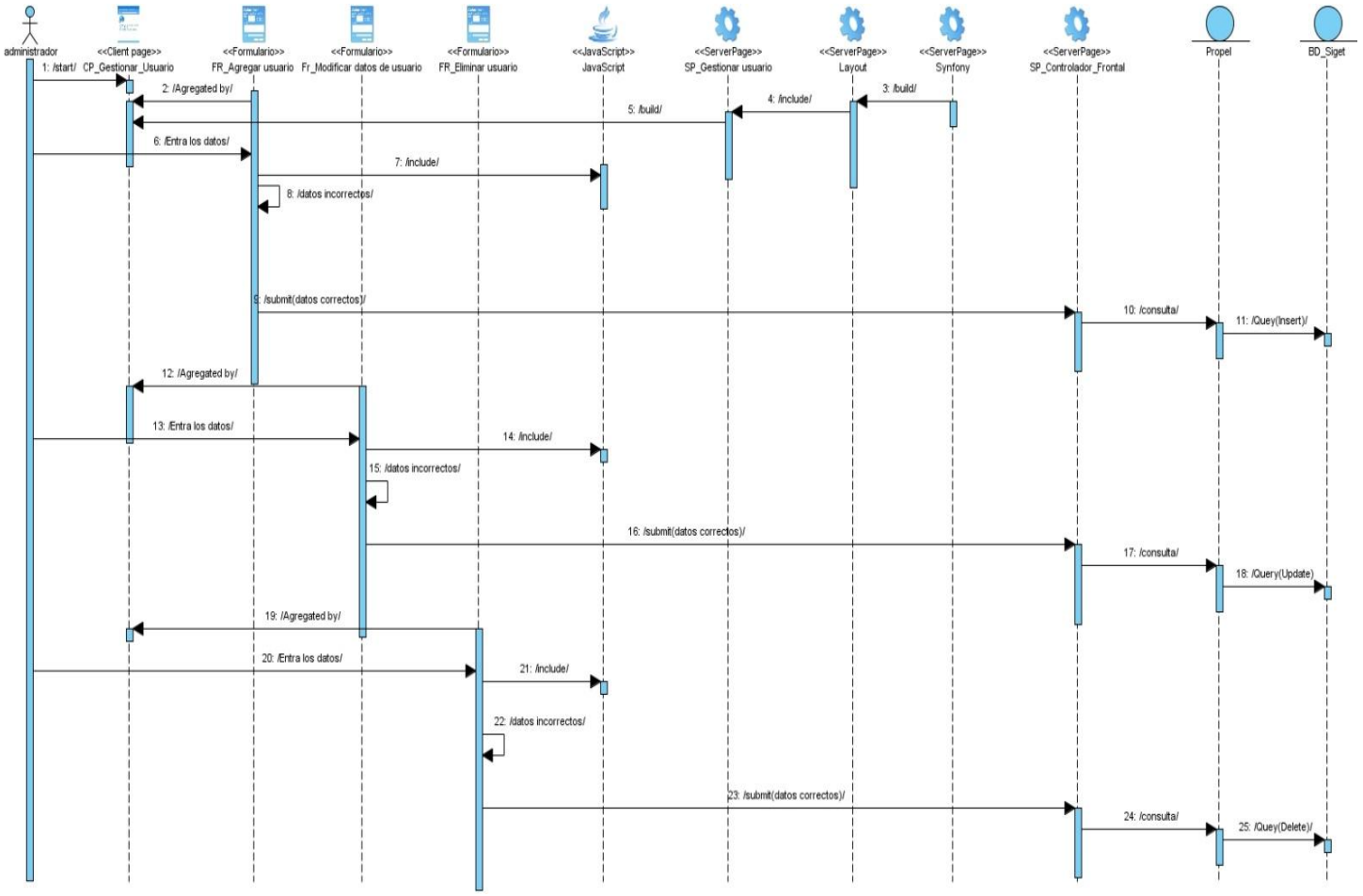


Figura 25 Diagrama de secuencia CU Gestionar Usuario

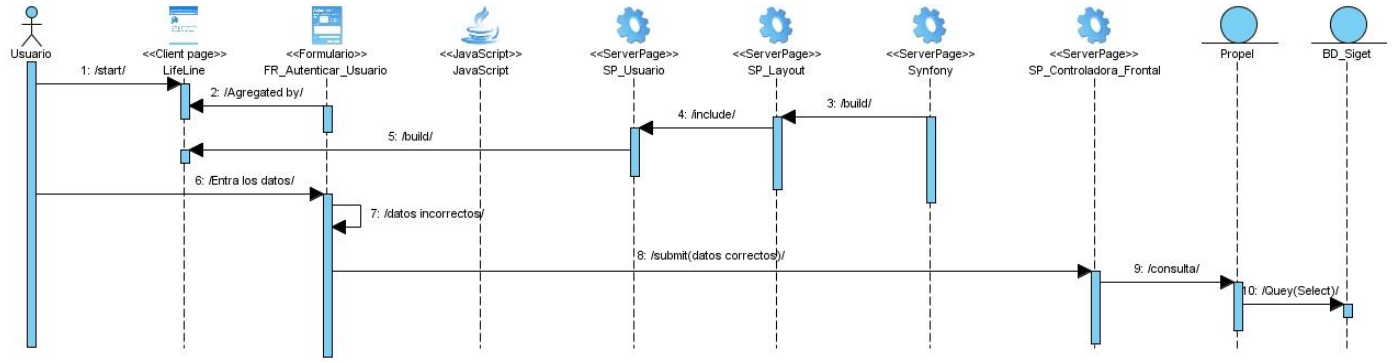


Figura 26 Diagrama de secuencia CU Autenticar Usuario

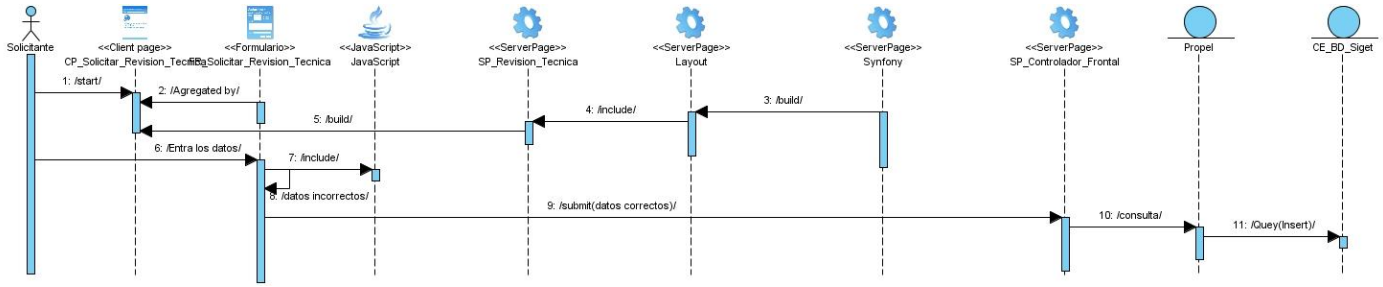


Figura 27 Diagrama de secuencia CU Solicitar Revisión Técnica

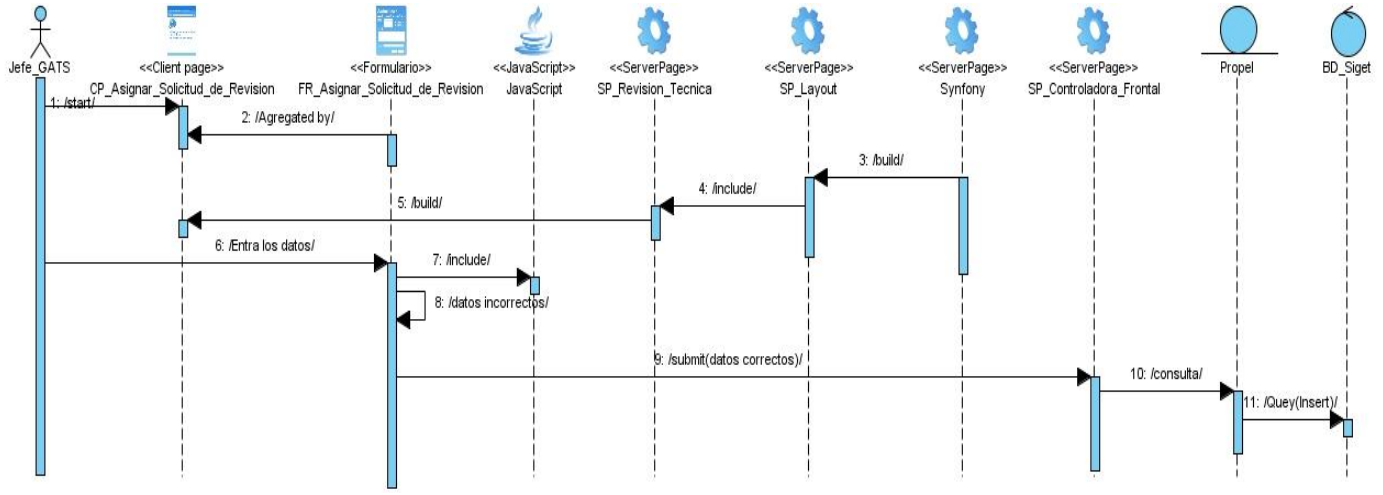


Figura 28 Diagrama de secuencia CU Asignar Revisión de solicitud de Revisión Técnica

Los restantes diagramas se pueden encontrar en el Anexo D.

3.3.4 Validación del Diseño

La validación es un paso importante dentro del proceso de desarrollo de software y más específicamente en los flujos de trabajo de análisis y diseño. Esto es debido a que la calidad de dichos flujos de trabajo influirá positiva o negativamente en la calidad de la futura implementación.

El objetivo de la validación es asegurar que el sistema satisface las expectativas del cliente. Va más allá de la comprobación de que el sistema satisface su especificación para demostrar que el software hace lo que el cliente espera que haga. [28]

Para darle cumplimiento al objetivo de la validación del diseño, es necesaria la utilización de métricas, que según (Pressman, 2008) van a ayudar a la evaluación de los modelos de análisis y de diseño, en donde proporcionarán una indicación de la complejidad de diseños procedimentales y de código fuente, y ayudaran en el diseño de pruebas más efectivas; es por eso que propone un proceso de medición.

Las métricas para sistemas OO deben de ajustarse a las características que distinguen el software OO del software convencional. Estas métricas hacen hincapié en el encapsulamiento, la herencia, complejidad de clases y polimorfismo. Por lo tanto las métricas OO se centran en métricas que se pueden aplicar a las características de encapsulamiento, ocultamiento de información, herencia y técnicas de abstracción de objetos que hagan única a esa clase. [29]

Para la emisión de un criterio evaluativo sobre la calidad del diseño que se pretende validar es necesario el uso de métricas, y la selección de una de ellas, estuvo dada por el tipo de desarrollo que se ha realizado, el cual es sin dudas OO. Existen métricas para la evaluación de software desarrollado bajo dicho paradigma, por tanto las métricas seleccionadas tienen que estar diseñadas para evaluar este tipo de software.

Métricas de evaluación OO

Fueron utilizadas en la presente investigación las métricas: Tamaño Operacional de Clase (TOC) y Árbol de Profundidad de Herencia (APH).

La métrica de TOC, es propuesta por Lorenz y Kidd (1994) y APH es producto al trabajo de Chidamber y Kemererer (1994) [30].

El tamaño operacional de clase permite determinar a partir de los valores de tamaño de clase si la responsabilidad de una clase es alta o baja. Por ejemplo, un valor alto de TC significaría responsabilidad alta por parte de una clase, reducción de la reutilizabilidad, llegando a complicar la implementación y las pruebas.

Clase	Cantidad de Procedimientos	Responsabilidad
CC_Usuario	5	Alta
CC_Agregar_Usuario	2	Media
CC_Eliminar_Usuario	2	Media
CC_Modificar_Rol_de_Usuario	2	Media
CE_usuario	5	Media
CC_RevisiónTécnica	5	Alta
CC_Solicitar_Revisión_Técnica	2	Media
CC_Asignar_Revisión_Técnica	2	Media
CC_Visualizar_Revisión_Técnica	2	Media
CC_Modificar_Estado_de _Revisión	2	Media
CE_revisióntécnica	5	Media
CC_Dictamen_Técnico	3	Alta
CC_Generar_Dictamen_Técnico	2	Media
CC_Modificar_Dictamen_Técnico	2	Media
CE_dictamenTécnico	3	Media
CC_ldapLoguin	1	Media
CC_GestionarLogs	3	Baja

Figura 29 Representación de tamaño de clases

Se obtuvo un promedio de **2.82** operaciones por cada clase del total de 17.

Responsabilidad	Cantidad de clases	Porciento
Baja	1	5.882352941
Media	13	76.47058824
Alta	3	17.64705882

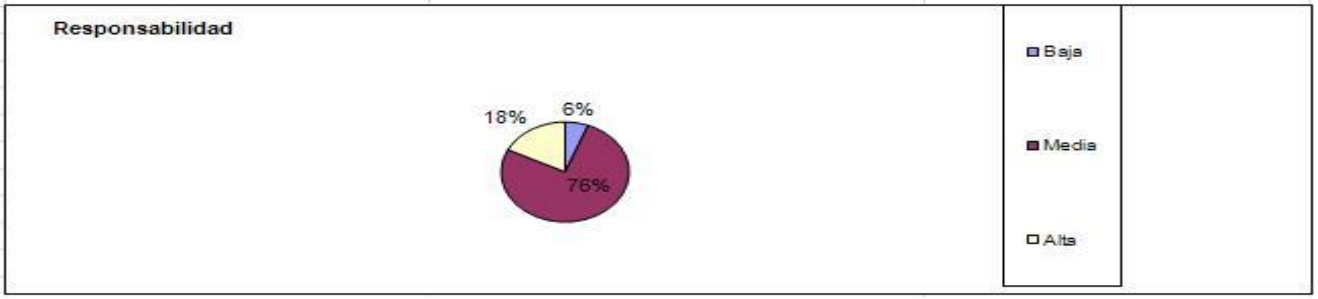


Figura 30 Responsabilidad de clases

Reutilización	Cantidad de clases	Porciento
Alta	2	11.76470588
Media	14	82.35294118
Baja	1	5.882352941

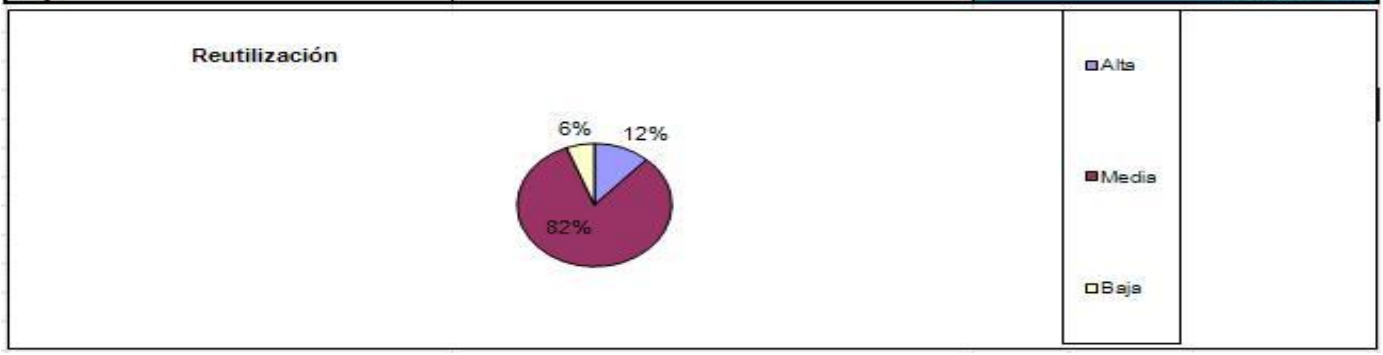


Figura 31 Reutilización de clases

Complejidad	Cantidad de clases	Porciento
Baja	1	5.882352941
Media	13	76.47058824
Alta	3	17.64705882

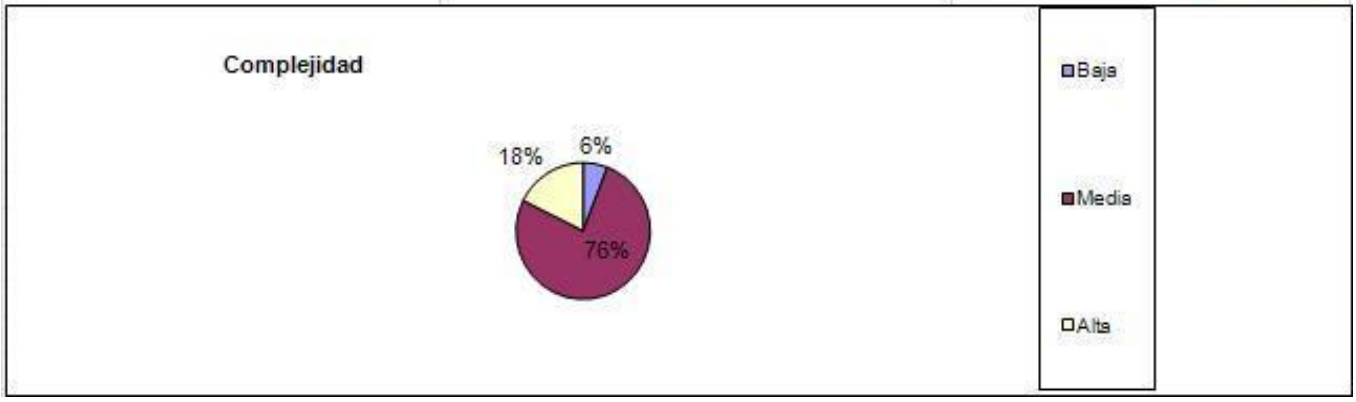


Figura 32 Complejidad de clases

APH posibilita la obtención de una visión del sistema diseñado orientado a la herencia. APH se puede definir como la longitud máxima desde un nodo hasta la raíz del árbol. El aumento de APH implica que los niveles inferiores heredan más funcionalidades, dando lugar a dificultades en cuanto a la determinación del comportamiento de la clase además de una mayor complejidad al momento del diseño.

Profundidad de Herencia	
Criterio	Cantidad de Clases
Nivel 1	8
Nivel 2	9
Nivel 3	0
Total	17

Figura 33 Profundidad de herencia

El nivel máximo de profundidad de herencia es 2 por consiguiente es posible garantizar un diseño sencillo gracias al bajo acoplamiento existente en Siget.

3.3.5 Diseño de la base de datos

Diagrama de clases persistentes

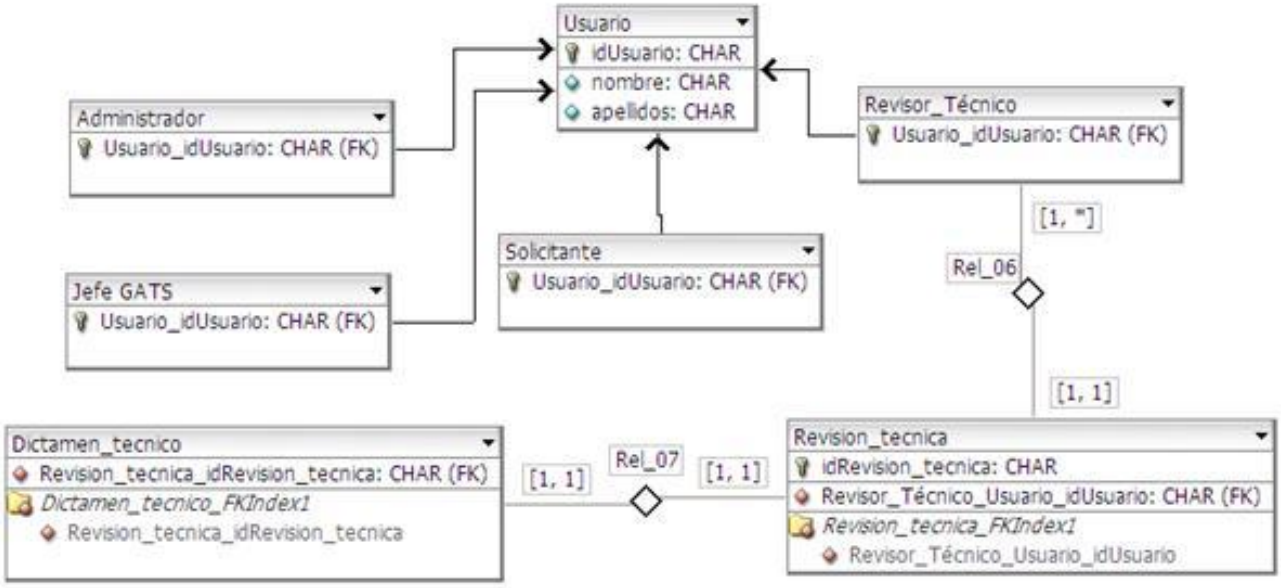


Figura 34 Diagrama entidad relación de Siget

3.4 Conclusiones

Este capítulo estuvo enfocado a detallar los diagramas de clases del diseño generados por los casos de uso, proporcionando un lenguaje orientado al desarrollador, definido por la complementación de la dinámica que deberá tener el sistema y las herramientas que serán utilizadas para el desarrollo del sistema que se diseña. Fueron definidos los patrones a utilizar, tanto arquitectónicos como de diseño, mostrando a través de los diagramas la estructura y potencialidades que brindan.

Conclusiones finales

A raíz de la necesidad existente en el Centro de Gobierno Electrónico (GATS) y específicamente en el Grupo de Arquitectura Tecnología y Seguridad (GATS), de mejorar el proceso de atención a solicitudes de revisión técnica, con la utilización de un sistema de gestión. Para esto se realizó un estudio de estado del arte, donde se analizaron los sistemas de gestión de información y como subcategoría los sistemas de gestión de proyectos, dada la similitud de estos con el proceso que se desea automatizar. Después de observar que los sistemas desarrollados para la gestión de proyectos no se adecuaban completamente al proceso, se decide comenzar con desarrollo a través de RUP.

En la etapa inicial se realizó el modelo de negocio para el proceso de atención a solicitudes de revisión técnica, para comprender el proceso con el cual se operaba en GATS, y luego obtener los requisitos funcionales que generaron los casos de uso de sistema que guiarían todo el proceso de desarrollo según la metodología seleccionada. Se realizó además el análisis para identificar la dinámica que debería tener el sistema en un lenguaje orientado a la comprensión del desarrollador. Más tarde se efectúa el diseño en el cual se toman en cuenta ya las herramientas a utilizar, la dinámica general del sistema y la estructura sugerida por el uso de patrones de diseño y arquitectónicos para mezclarlas en cada uno de los diagramas de clases y secuencia del diseño, brindando finalmente todos los elementos necesarios para que los desarrolladores puedan implementar la solución.

Recomendaciones

Teniendo en cuenta que todo desarrollo de software no está ajeno a cambios que añadan nuevas funcionalidades que contribuyan a enriquecer el software, el autor de la presente investigación, propone las siguientes recomendaciones:

- Realizar la implementación al sistema modelado en el GATS, y así validar la solución propuesta.
- Mantener esta solución como referencia para sistemas cuyas características sean similares.
- Utilizar la propuesta de solución no solo dentro del GATS sino en cualquier proyecto que posea un proceso similar al de atención de solicitudes de revisión técnica.

Referencias Bibliográficas

[1] [Internet] Disponible en <http://www.its.bldrdoc.gov/fs-1037/dir-036/_5255.htm> [Citado en: 14 Junio 2011].

[2] LAROUSSE *Diccionario Manual de la Lengua Española*. Larousse [Citado en: 20 Febrero 2011].

[3] DOMINGO Ajenjo, Alberto *Dirección y Gestión de Proyectos: un enfoque práctico 2005* .RA-MA 2005 [Citado en: 20 Febrero 2011].

[4] JACOBSON, IVAR; BOOCH, GRADY y RUMBAUGH, JAMES. *El Proceso Unificado de Desarrollo de Software.* Madrid 2000. [Citado en: 14 Junio 2011].

[5] Cortizo Pérez, José Carlos; Expósito Gil, Diego y Ruiz Leyva; Miguel. *eXtreme Programming* [Internet] Disponible en <http://www.esp.uem.es/jccortizo/xp.pdf>[Citado en: 14 Junio 2011].

[6] DÍAZ Flores, Mirian Milagros. *RUP vs Xp*. ESCUELA DE INGENIERÍA DE SISTEMAS (http://www.extremeprogramming.org/, http://www.programacionextrema.org/, http://www.geocities.com/chuidiang/metodologia/extrema.html) [Citado en: 14 Junio 2011].

[7] Dr.C Profesor Auxiliar Piñero Pérez, Pedro Y; MsC Leyva Vázquez, Maikel Yelandi. *Metodologías ágiles y formales o robustas*. [Citado en: 14 Junio 2011].

[8] Instituto Nacional de Estadística e Informática. *Herramientas CASE*. Colección Cultura Informática. [Internet] Disponible en http://docs.google.com/viewer?a=v&q=cache:Mn-wR-O8mNIJ:www.inei.gob.pe/biblioineipub/bancopub/Inf/Lib5103/Libro.pdf+herramientas+case&hl=es&pid=bl&srcid=ADGEESgA_8HOs8sES_cEotYfMJ7iGTuplYaAkSZqdLXbtNs751Neib4aT0ZME2oyrUYm1UmGMNtba2sJAOyPAqBLoCN-7nxcRroqxvr3Og4tXT5vBfmmZ5lBLHD78UHwAng9F8Kavq0A&sig=AHIEtbRGYE3xJKTm-JObjgM28h5PyM50Rg [Citado en: 14 Junio 2011].

[9] [Internet] Disponible en http://www.sparxsystems.com.ar/products/ea.html [Citado en: 14 Junio 2011].

[10] Disponible en <http://eva.uci.cu/file.php/102/Curso_2010-2011/Clases/Semana_09/Laboratorio_2/Materiales_basicos/Caracteristicas_de_Herramientas.pdf>[Citado en: 14 Junio 2011].

[11] Becerril C, Francisco. *Java a su alcance*.1998 [Citado en: 14 Junio 2011].

[12] [Internet] Disponible en <http://www.java.com/es/about> [Citado en: 14 Junio 2011].

[13] Disponible en <http://eva.uci.cu/file.php/624/2._Clases/Semana_1/1ra_frecuencia/MApoyo/C1_Introductoria.pdf>[Citado en: 14 Junio 2011].

[14] [Internet]. Disponible en: <http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php> [Citado 9 Abril 2011].

[15] Mohamed, J. Kabir. *La biblia de Apache.* Disponible en: <http://bibliodoc.uci.cu/pdf/reg01737.pdf> [Citado 9 Abril 2011].

[16] Disponible en: <http://www.cerocerouno.com.ar/?p=228>[Citado 9 Abril 2011].

[17] Leopoldo, Carlos (17 de julio de 2008). *Manual de CodeIgniter en español.* [Citado 9 Abril 2011].

[18] Billy Reynoso, Carlos. *Introducción a la arquitectura de software*. [Citado 9 Abril 2011].

[19] Buschmann, Frank, et al. *Pattern-Oriented Software Architecture.* John Wiley & Sons 1996. [Citado 9 Abril 2011].

[20] [Internet] Disponible en: <http://webcache.googleusercontent.com/search?q=cache:t_93rv7xSmcJ:https://www.u-cursos.cl/ingenieria/2005/2/CC51A/1/material_docente/objeto/76454+patrones+arquitectonicos&cd=6&hl=es&ct=clnk&gl=cu&source=www.google.com.cu> [Citado 9 Abril 2011].

[21] Kuchana, P. *Software architecture design patterns in java*. [Citado 9 Abril 2011].

[22] Larman, Craig 99. *UML y patrones Introducción al análisis y diseño orientado a objetos.* [Citado 9 Abril 2011].

[23] *Patrones del "Gang of Four".* Facultad de Informática de la Universidad Politécnica de Madrid. Disponible en <http://eva.uci.cu/file.php/259/Curso_2010-2011/Semana_4/Seminario_1/Materiales_Basicos/patrones_gof.pdf>[Citado 11 Junio 2011].

[24] Disponible en <http://eva.uci.cu/file.php/102/Curso_2010-2011/Clases/Semana_04/Conferencia_6/Materiales_complementarios/Fase_de_Inicio._Disciplina_de_Modelamiento_del_Negocio.pdf> > [Citado 11 Junio 2011].

[25] Disponible en < http://eva.uci.cu/file.php/102/Curso_2010-2011/Clases/Semana_04/Conferencia_6/Materiales_complementarios/Fase_de_Inicio._Disciplina_de_Modelamiento_del_Negocio.pdf> [Citado 11 Junio 2011].

[26] Gamma, Helm, Johnson, and Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995. [Citado en: 15 Diciembre 2010].

[27] [Internet] Disponible en: http://msdn.microsoft.com/es-es/library/bb972272.aspx [Citado en: 15 Diciembre 2010].

[28] Sommerville, Ian (2005). *Ingeniería del software* (Sétima edición). [Citado en: 15 Diciembre 2010].

[29] Pressman. CAPÍTULO 6 Métricas para Sistemas Orientados a Objetos. [Citado en: 15 Junio 2011].

[30] Lorenz y Kidd (1994). *A Metrics Suite for Object- Oriented Design*. [Citado en: 15 Junio 2011].

Bibliografía Consultada

1. JACOBSON, Ivar;BOOCH Grady;RUMBAUGH James. *El Proceso Unificado de Desarrollo de Software.*
2. PRESSMAN, Roger S. *Ingeniería del Software. Un enfoque práctico.*
3. LARMAN, C. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.*
4. http://www.ceslcam.com/conocelo/analisis-de-aplicaciones/Redmine
5. http://pymecrunch.com/aplicaciones-de-codigo-abierto-para-cubrir-las-necesidades-de-gestion-de-tu-empresa

Anexo A

CU Solicitar Prórroga de Tiempo

Caso de Uso:	Solicitar Prórroga de Tiempo
Actores:	Revisor técnico
Resumen:	El actor solicita una extensión del plazo de entrega de la revisión técnica.
Precondiciones:	El actor tiene que estar autenticado en el sistema.
Referencias	RF 18.
Prioridad	Media

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Revisor técnico selecciona la opción Solicitar Prórroga de Tiempo.	2 Muestra un formulario con los campos correspondientes a la solicitud de prórroga.
3 El Revisor técnico introduce los datos.	4 el sistema valida la solicitud, la no existencia de campos vacíos, límite de cantidad de caracteres de la descripción y de los demás campos. El sistema guarda los cambios realizados en la solicitud, y notifica a la Gerencia del GATS de dicha prórroga.

Tabla 11 CU Realizar revisión de la solicitud.

CU Generar un dictamen técnico

Caso de Uso:	Generar un dictamen técnico
Actores:	Revisor técnico
Resumen:	El actor se identifica en el sistema y procede a generar el dictamen técnico asociado a dicha solicitud y al artefacto que esta posee.
Precondiciones:	El actor tiene que estar autenticado en el sistema, CU Realizar revisión de solicitud.
Referencias	RF 6.
Prioridad	Alta

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Revisor técnico selecciona la opción generar dictamen técnico.	2 Muestra una ventana con las solicitudes asignadas cuyo estado de revisión sea 100%.
3 El Revisor técnico selecciona la solicitud que desee para generar dictamen técnico de dicha solicitud.	4 Muestra un formulario con los campos (ID de revisión, revisor, etc.) a generar en el dictamen.
5 Llena los campos en el formulario mostrado y pulsa generar dictamen técnico.	6 El sistema guarda los cambios realizados en la solicitud, y envía al solicitante el dictamen técnico generado.

Tabla 12 CU Generar dictamen técnico

CU Solicitar taller informativo.

Caso de Uso:	Solicitar taller informativo
Actores:	Solicitante
Resumen:	El actor se identifica en el sistema, y tras recibir un Dictamen Técnico no satisfactorio, procede a solicitar un taller informativo, donde se le informa las particularidades del dictamen emitido.
Precondiciones:	El actor tiene que estar autenticado en el sistema, CU Emitir Dictamen Técnico
Referencias	RF 9.
Prioridad	Alta

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El actor selecciona la opción solicitar taller informativo.	2 Muestra un formulario donde deberá introducir el identificador de la solicitud a la cual le fue emitido el dictamen en cuestión.
3 El actor llena los campos mostrados anteriormente.	4 El sistema verifica que el identificador sea correcto y pertenezca a alguna solicitud presentada por este solicitante.
5 Llena los campos en el formulario mostrado y pulsa generar dictamen técnico.	6 El sistema guarda los cambios realizados en la solicitud, y envía al solicitante el dictamen técnico generado.

Tabla 13 CU Solicitar taller informativo

CU Solicitar revisión técnica

Caso de Uso:	Solicitar revisión técnica
Actores:	Solicitante
Resumen:	El actor se identifica en el sistema, y selecciona la opción solicitar revisión técnica, y llena los datos correspondientes para procesar la solicitud.
Precondiciones:	El actor tiene que estar autenticado en el sistema.
Referencias	RF 2.
Prioridad	Alta

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El actor selecciona la opción solicitar revisión técnica.	2 Muestra un formulario donde deberá llenar una descripción, y cargar el artefacto al sistema.
3 El actor llena los campos de la descripción y carga el archivo al sistema.	4 El sistema verifica que la descripción se haya llenado con la cantidad de caracteres mínima o máxima posible y que el archivo haya sido cargado correctamente. Luego muestra un mensaje: Solicitud procesada correctamente

	5 El sistema guarda los cambios realizados en la solicitud, y envía una notificación confirmando la solicitud realizada.

Tabla 14 CU Solicitar revisión técnica

CU Asignar solicitud al revisor técnico

Caso de Uso:	Asignar solicitud al revisor técnico
Actores:	Gerencia del GATS
Resumen:	El actor se identifica en el sistema, selecciona la opción Asignar solicitud al revisor técnico y el sistema muestra las solicitudes que no han sido asignadas, el actor selecciona una de ellas y se la asigna al revisor técnico correspondiente
Precondiciones:	El actor tiene que estar autenticado en el sistema.
Referencias	RF 3.
Prioridad	Alta

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El actor selecciona la opción Asignar solicitud al revisor técnico.	2 Muestra aquellas revisiones que aún no han sido asignadas.
3 El actor selecciona la solicitud que desea asignar.	4 Muestra un formulario con la opción asignar revisión técnica a: y a continuación un combo Box con los revisores existentes en el sistema

5 El actor selecciona el revisor correspondiente	6 El sistema guarda los cambios realizados en la solicitud, y envía una notificación al revisor al le fue asignada dicha solicitud.

Tabla 15 CU Asignar solicitud al revisor técnico

CU Visualizar revisión técnica.

Caso de Uso:	Visualizar revisión técnica
Actores:	Gerencia del GATS
Resumen:	El actor se identifica en el sistema, selecciona la opción Visualizar revisión técnica y el sistema muestra las solicitudes que han sido asignadas, el actor selecciona una de ellas y procede a revisar el estado en que se encuentran estas.
Precondiciones:	El actor tiene que estar autenticado en el sistema.
Referencias	RF 8.
Prioridad	Media
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El actor selecciona la opción Visualizar revisión técnica.	2 Muestra las revisiones que se encuentran asignadas.
3 El actor selecciona la revisión que desee monitorizar.	4 El sistema muestra los datos de la revisión y el estado en el que se encuentra dicha revisión.

Tabla 16 CU Realizar seguimiento a las revisiones técnicas.

CU Modificar estado de las revisiones técnicas

Caso de Uso:	Modificar estado de las revisiones técnicas
Actores:	Revisor Técnico
Resumen:	El actor se identifica en el sistema, el actor selecciona la opción Modificar estado de revisión y el sistema muestra las solicitudes que han sido asignadas, el actor selecciona una de ellas y procede a modificar el estado en que se encuentran estas.
Precondiciones:	El actor tiene que estar autenticado en el sistema.
Referencias	RF 5.
Prioridad	Media

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El actor selecciona la opción Realizar seguimiento.	2 Muestra un submenú en el cual se puede visualizar la opción Modificar estado de revisión.
3 El actor selecciona la opción Modificar estado de revisión.	4 Muestra las revisiones que se encuentran asignadas.
5 El actor selecciona la revisión a la que desee modificar el estado de revisión.	6 El sistema muestra los datos de la revisión entre ellas, el estado en el que se encuentra dicha revisión, con la

	posibilidad de cambiar dicho estado.
7 El actor modifica el estado de la revisión seleccionada.	8 El sistema guarda los datos y envía notificaciones al revisor encargado de la revisión y al solicitante de la revisión, y muestra además el siguiente mensaje: Revisión modificada satisfactoriamente.

Tabla 17 CU Modificar estado de las revisiones técnicas.

CU Anular Dictamen técnico.

Caso de Uso:	Anular Dictamen técnico
Actores:	Gerencia del GATS
Resumen:	El actor se identifica en el sistema, el actor selecciona la opción Anular Dictamen técnico y el sistema muestra las revisiones que han sido terminadas recientemente, el actor selecciona una y procede a anular el dictamen técnico emitido para esta solicitud.
Precondiciones:	El actor tiene que estar autenticado en el sistema.
Referencias	RF 7.
Prioridad	Media
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**

1 El actor selecciona la opción Anular Dictamen técnico.	2 Muestra las revisiones que han sido terminadas recientemente.
3 El actor selecciona la revisión a la que desee anular el dictamen técnico.	4 El sistema muestra los datos de la revisión, incluido un botón para acceder al dictamen técnico generado y otro para anular este dictamen.
5 El actor anula el dictamen a través del botón correspondiente.	6 El sistema guarda los datos y envía notificaciones al revisor encargado de la revisión y al solicitante de la revisión, y muestra además el siguiente mensaje: Dictamen anulado satisfactoriamente.

Tabla 18 CU Anular Dictamen técnico.

Anexo B

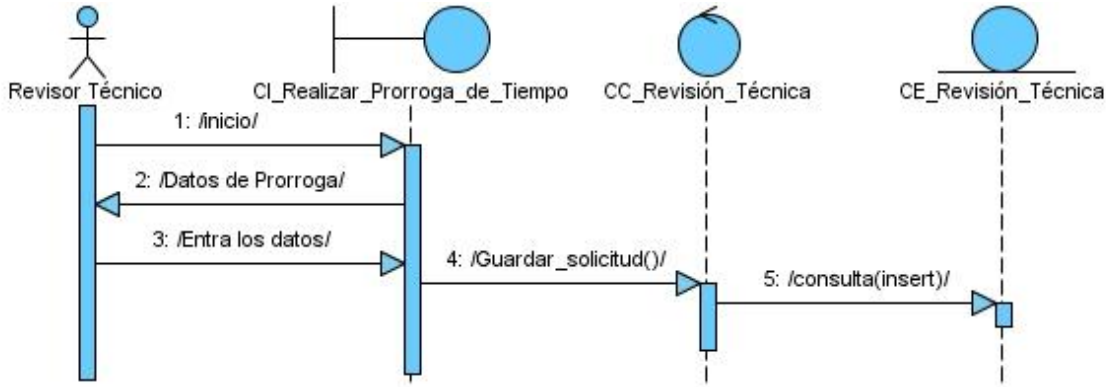


Figura 35 CU Realizar Prórroga de tiempo

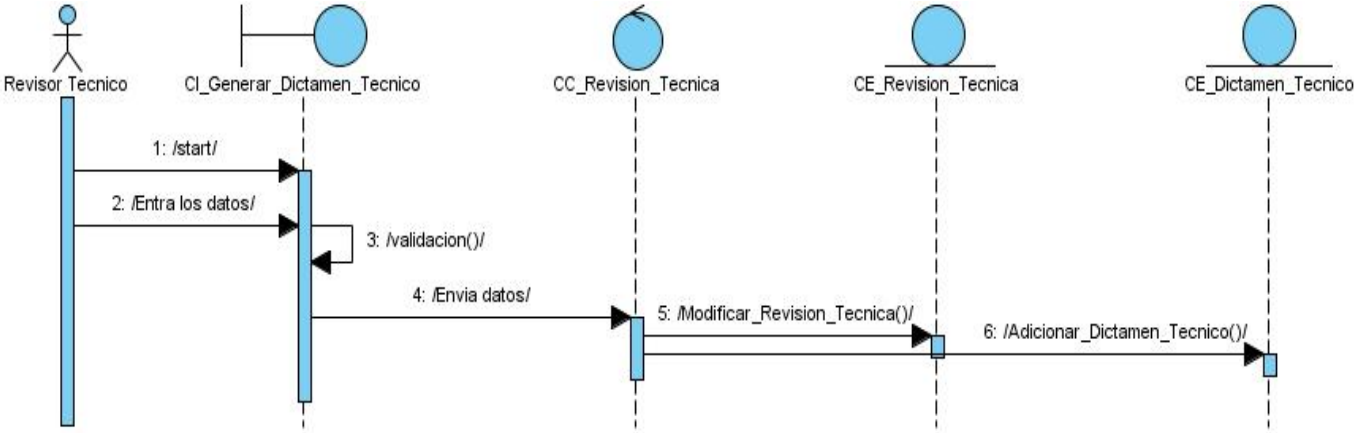


Figura 36 Generar Dictamen Técnico

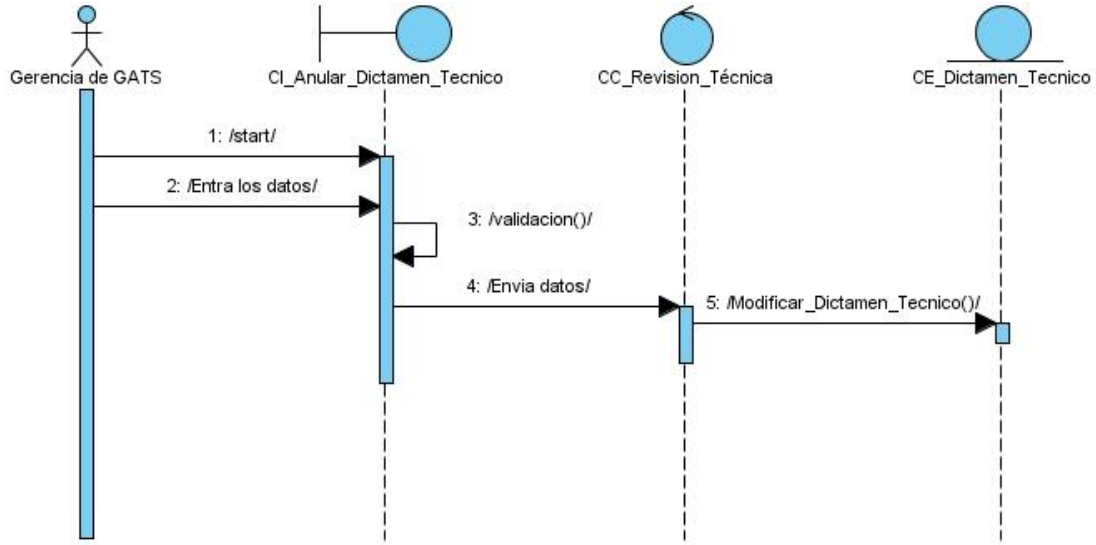


Figura 37 CU Anular Dictamen Técnico

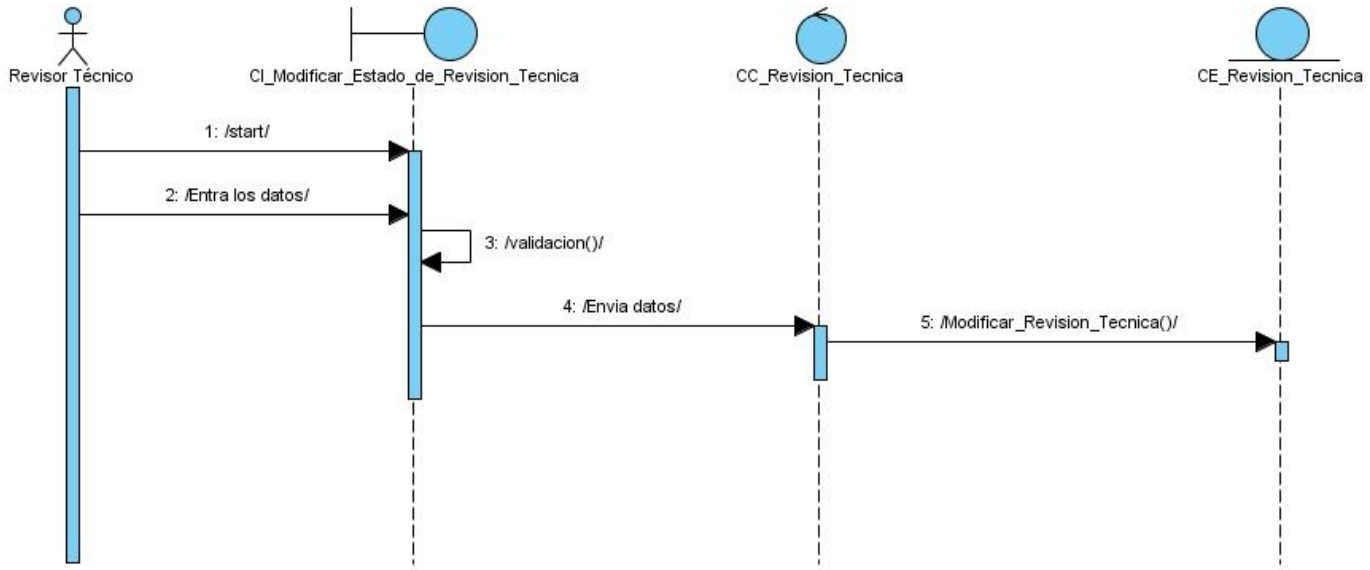


Figura 38 CU Modificar Estado de Revisión Técnica

Anexo C

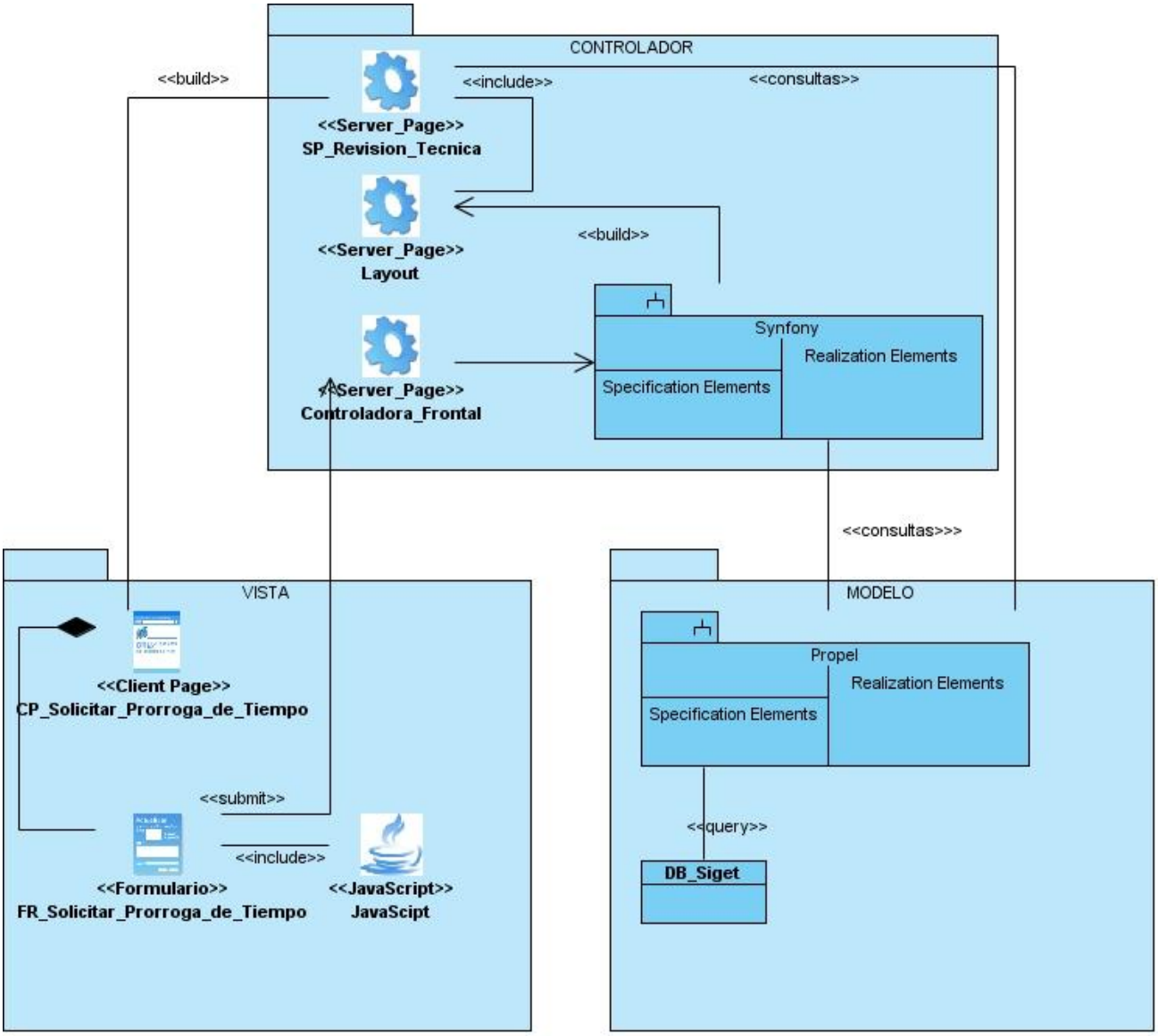


Figura 39 Diagrama de clase del diseño CU Solicitar Prorroga de Tiempo

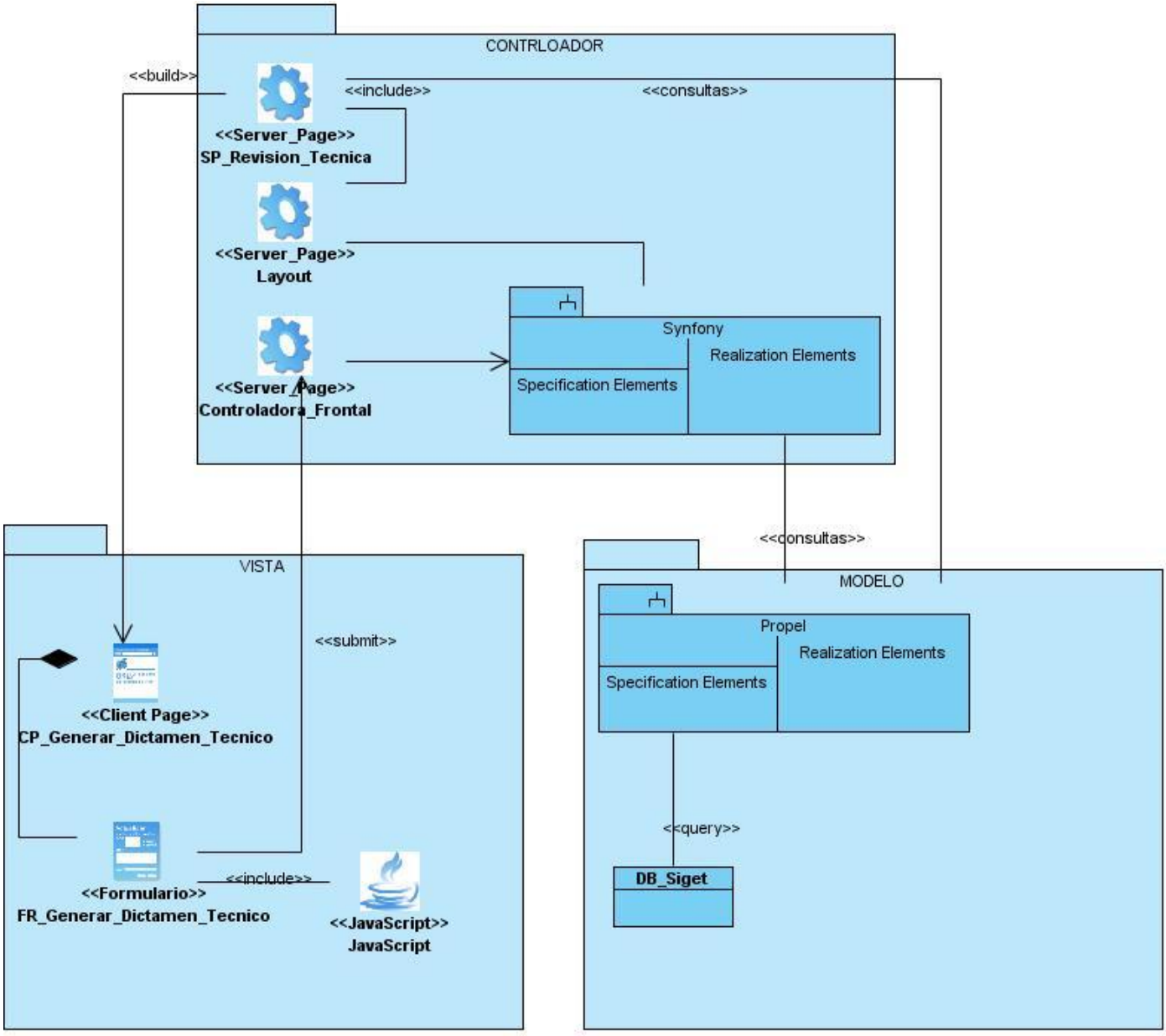


Figura 40 Diagrama de clase del diseño CU Generar Dictamen Técnico

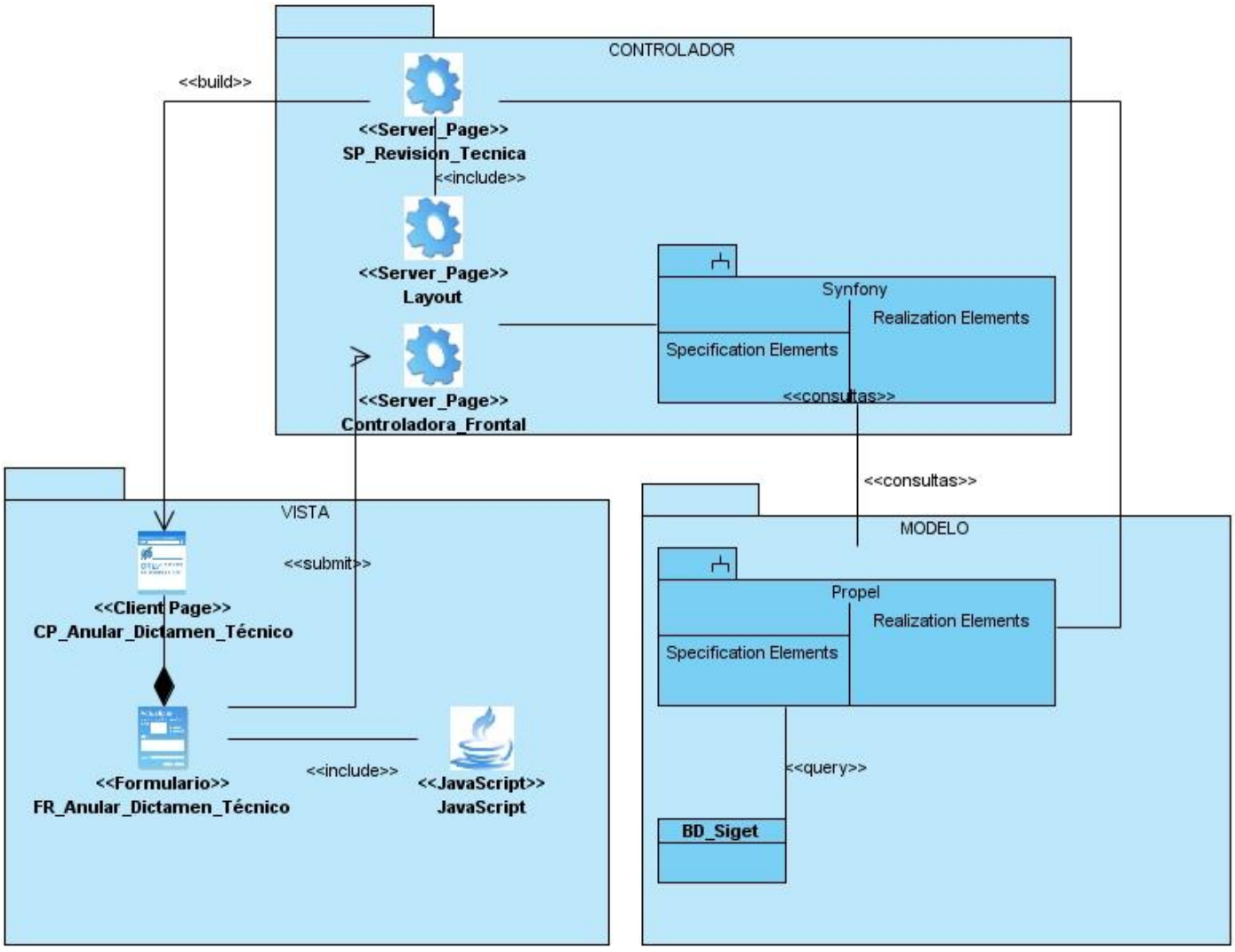


Figura 41 Diagrama de clase del diseño CU Anular Dictamen Técnico

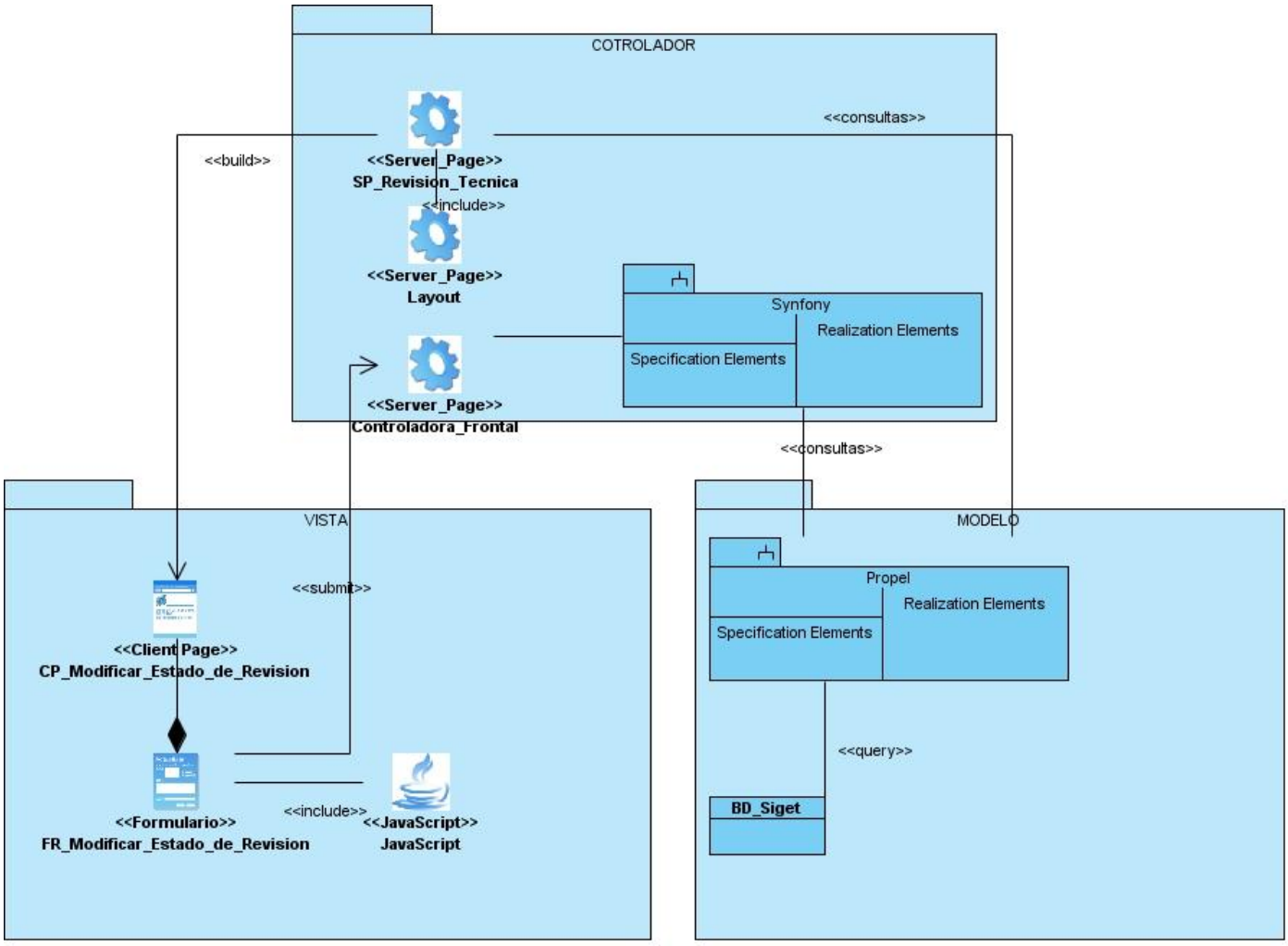


Figura 42 Diagrama de clase del diseño CU Modificar Estado de Revisión Técnica

Anexo D

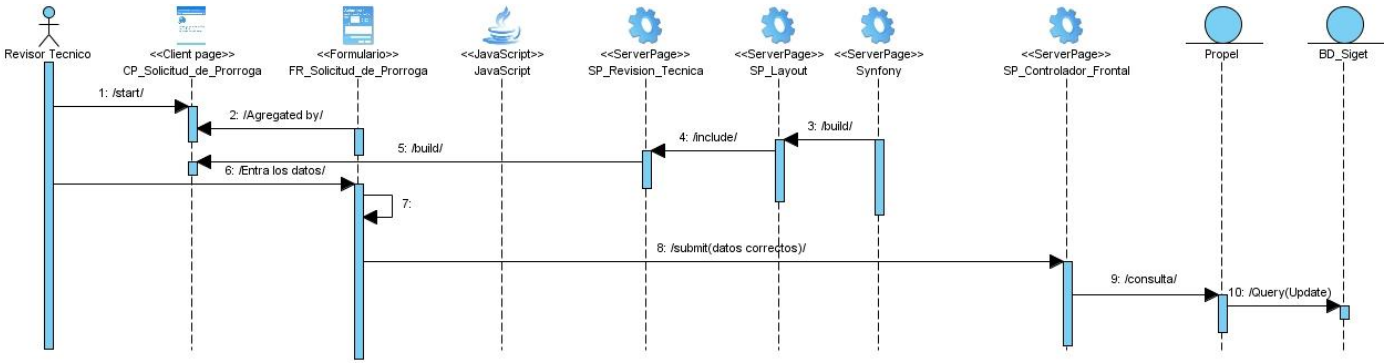


Figura 43 Diagrama de secuencia CU Solicitar Prorroga de Tiempo

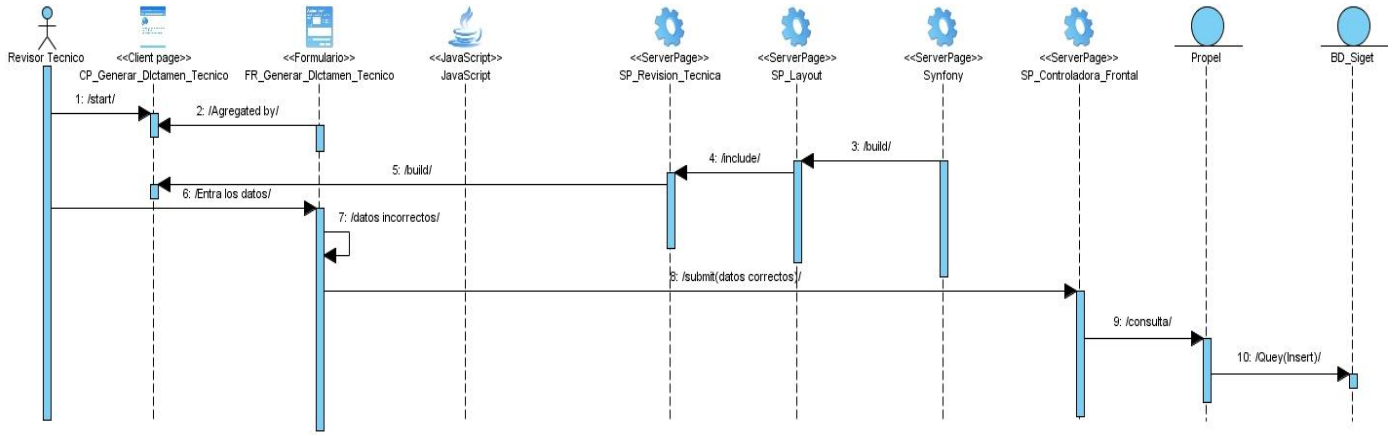


Figura 44 Diagrama de secuencia CU Generar Dictamen Técnico

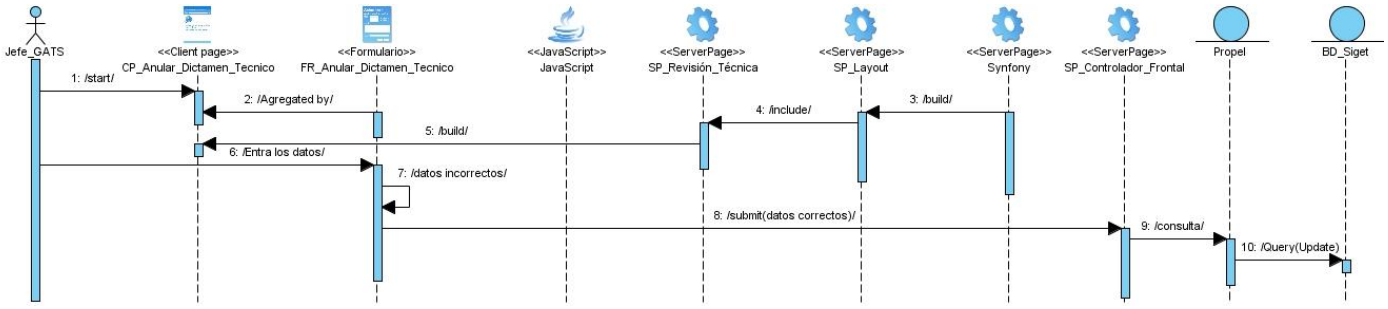


Figura 45 Diagrama de secuencia CU Anular Dictamen Técnico

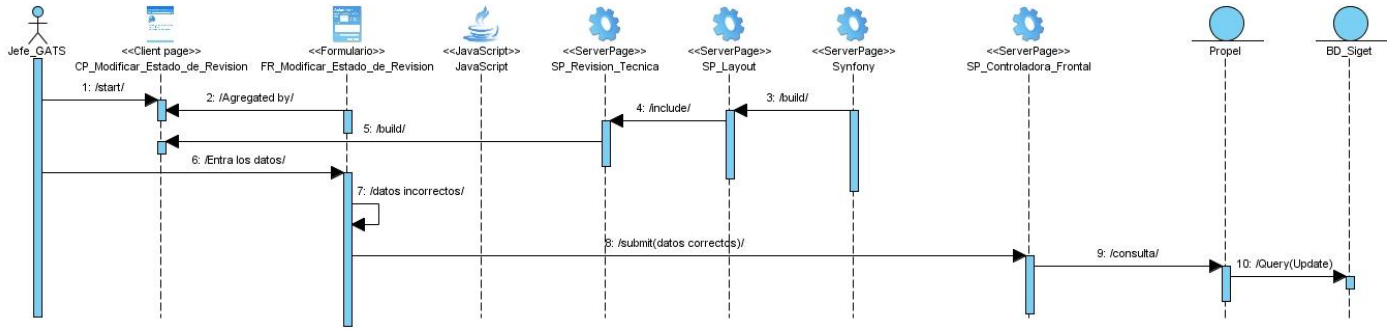


Figura 46 Diagrama de secuencia CU Modificar estado de Revisión Técnica