Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Desarrollo del Módulo de Gestión de Recursos Materiales del Sistema Integral de Gestión Administrativa de la Facultad 7

Autores: Yanet Alba Morales

Alexander Rodríguez López

Tutores: Ing. Ricardo Collada de la Rosa

Ing. Néstor Llanes Guerra

Ciudad de La Habana, julio de 2010

"Año 52 de la Revolución"

Frase

´´ La educación empieza con la vida, y no acaba sino con la muerte´´

José Martí

Datos de contacto

Ing. Ricardo Collada de la Rosa. Graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en el 2008. Instructor recién graduado en adiestramiento. Durante su trabajo como profesor ha impartido la asignatura de Segundo Perfil.

En la vinculación con la producción pertenece al Departamento de Sistemas Especializados en Medicina del Centro Especializado en Soluciones de Informática Médica (CESIM) y específicamente se desempeña como Arquitecto Principal del CESIM.

Correo electrónico: rcollada@uci.cu

Ing. Néstor Llanes Guerra Graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en el 2007. Profesor Instructor. Durante su trabajo como profesor ha impartido las asignaturas de Máquinas Computadoras I, Máquinas Computadoras II, Teleinformática I y Teleinformática II; desempeñándose como Jefe de las cuatro asignaturas.

En la vinculación con la producción pertenece al Departamento de Sistemas Especializados en Medicina del Centro Especializado en Soluciones de Informática Médica (CESIM) y específicamente trabaja en el desarrollo del proyecto Alas Rehabilitación donde se desempeña como Líder de Proyecto.

Correo electrónico: nllanes@uci.cu

Resumen

En el mundo de hoy llevar el control de recursos materiales se hace cada vez más difícil. Dada esta situación, se impone la necesidad de combatir el derroche y el descontrol sobre estos recursos, así como de implementar mecanismos efectivos para su cuidado y reparación.

Debido a todo este problema existente, la Universidad de las Ciencias Informáticas se ha trazado la meta de implementar una aplicación que permita viabilizar la gestión de Recursos Materiales del Sistema Integral de Gestión Administrativa de la Facultad 7.

Para poner en práctica este proyecto fueron utilizadas un conjunto de herramientas y técnicas como son: UML 2.0, PostgreSQL 8.3, Python, Enterprise Arquitect, Django y Wing IDE.

Con la instalación y puesta en práctica del módulo de Gestión de Recursos Materiales se automatizará el control de todos los recursos materiales de la facultad, de esta manera no existirá pérdida de la información, se garantiza la integridad de la información y la gestión de la información sería más eficiente.

Índice

FRASE ---------- II
DATOS DE CONTACTO ---------- III
RESUMEN ----------IV
ÍNDICE ----------V
INTRODUCCIÓN ----------1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA ----------5
1.1 ANTECEDENTES, TENDENCIAS DE RECURSOS MATERIALES Y SISTEMAS AUTOMATIZADOS O SERVICIOS DE RECURSOS MATERIALES EXISTENTES ---------- 5
1.2 HERRAMIENTAS Y TECNOLOGÍAS UTILIZADAS. ---------- 6
1.3 PYTHON ---------- 7
1.4 DJANGO ---------- 8
1.5 POSTGRESQL ---------- 8
1.6 ENTERPRISE ARQUITECT (EA) ----------10
1.7 UML2.0 ----------11
1.8 WING IDE 3.2 ----------12
1.9 METODOLOGÍA DE DESARROLLO ----------12
CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA ---------- 14
2.1 OBJETO DE AUTOMATIZACIÓN ----------14
2.1.1 Descripción del objeto de estudio ----------14
2.1.2 Modelo del Negocio ----------14
2.2 Especificación de los requisitos de *software*. ----------17
2.2.1 Requerimientos Funcionales: ----------18
2.2.2 Requerimientos no Funcionales: ----------18
2.3 Definición de los casos de uso y actores del sistema. ----------20
2.3.1 Actores del Sistema. ----------20
2.3.2 Casos de uso del sistema. ----------20
2.3.3 Diagrama de casos de uso del sistema. ----------23
2.3.4 Casos de uso expandidos. ----------23
CAPÍTULO 3. ANÁLISIS Y DISEÑO DEL SISTEMA ---------- 29
3.1 Descripción de la arquitectura. Fundamentación. ----------29
3.2 Análisis. ----------29
Modelo de análisis: ----------29
Diagramas de clases de análisis: ----------29
3.3 Descripción de las clases: ----------32
Clases Interfaz: ----------32
CI_GenerarReporte: Permite acceder a los reportes. ----------32

3.4 Diseño. ---- **33**
Modelo del diseño: ---- **33**
3.5 Diagramas de interacción ---- **33**
Modelado mediante estereotipos Web: ---- 35
CAPÍTULO 4: IMPLEMENTACIÓN ---- **40**
4.1 Modelo de datos ---- **40**
4.2 Descripción de las tablas. ---- **41**
4.3 Diagramas de componentes. ---- **44**
4.4 Diagramas de despliegue. ---- **50**
4.5 Tratamiento de errores. ---- **50**
4.6 Seguridad. ---- **51**
4.6.1 Autenticación. ---- 52
4.6.2 Autorización. ---- 52
CONCLUSIONES ---- **54**
RECOMENDACIONES ---- **55**
REFERENCIAS BIBLIOGRÁFICAS ---- **56**
BIBLIOGRAFÍA ---- **58**
ANEXOS ---- **60**
Anexo 1. Diagramas de clases del análisis. ---- **60**
Anexo 2. Diagramas de Interacción del diseño. ---- **61**
GLOSARIO DE TÉRMINOS ---- **64**

Introducción

En la actualidad en el mundo la industria del *software* ha evolucionado grandemente hasta convertirse en un renglón importante en la economía. La mayoría de los países que han alcanzado un alto nivel de desarrollo en esta rama son del “primer mundo”, ya que poseen gran cantidad de capital para invertir en recursos materiales y humanos. Mientras que en los países subdesarrollados, existe un déficit presupuestario que pueda respaldar las condiciones que se requieren para desarrollar una buena calidad en la industria del *software* y un bajo nivel de preparación a los profesionales.

En la economía de la India se ha producido una revolución industrial estimulada por la presencia cada vez mayor de las multinacionales, el aumento de las operaciones de las empresas nacionales y la ampliación del mercado interno. A pesar de que es un país muy pobre, el desarrollo de la informática la ha llevado a postularse como una de las próximas potencias mundiales. Aún con sus condiciones de pobreza, ocupa el primer lugar en exportación de *software* y servicios informáticos (mantenimiento, soporte, infraestructura, diseño, consultoría); también es líder en el *ranking* de países con mayor número de ingenieros calificados y se sitúa como el tercer país con mayor reserva de mano de obra tecnológica altamente competitiva.

Debido a la crítica situación que ha existido con el bloqueo impuesto por el gobierno de los Estados Unidos de América, la economía cubana se ha visto frenada en muchos frentes. Es por ello que siguiendo una idea del Comandante en Jefe de buscar nuevas vías para mejorar la economía, se crearon los Joven Club de Computación y Electrónica que fueron una iniciativa para llevar la informática a los más apartados lugares del país y de esta forma eliminar la brecha digital que se crea entre las ciudades y las zonas rurales. Por lo cual es asequible para casi la totalidad de la población cubana y se garantiza de esta manera la obtención de los conocimientos más básicos relacionados con la informática y la electrónica.

Además, se sigue trabajando en el desarrollo de programas para los equipos médicos y multimedia sobre los más variados temas. Se creó el Ministerio de Informática y las Comunicaciones, institución que se encargaría de enfocar de manera positiva el desarrollo de la informática.

Una de las más importantes fue la creación de la Universidad de las Ciencias Informáticas (UCI), inaugurada por el propio Comandante Fidel Castro Ruz hace 8 años, en la cual se gradúan anualmente aproximadamente 1000 ingenieros informáticos con el objetivo de aportar en materia de exportación y sustitución de importaciones a la economía del país y la sociedad en general.

La rama de la informática tiene como principal fuente de materias primas, los recursos humanos. Por ello, países como la India pueden ubicarse en posiciones privilegiadas en este sector en el mundo. Cuba ha apostado también por convertirse en una potencia en materia de informática y para ello cuenta con planes de capacitación de técnicos y profesionales de este sector. Esto no quita que se deben realizar inversiones en materia de recursos materiales, las cuales con el avance constante de la tecnología se vuelven cada día más cuantiosas.

Esto sin contar las dificultades y obstáculos que debe sortear el país para realizarlas debido fundamentalmente al bloqueo económico antes mencionado. Dada esta situación se impone la necesidad de combatir el derroche y el descontrol sobre estos recursos, así como de implementar mecanismos efectivos para su cuidado y reparación.

En la Universidad de las Ciencias Informáticas existe una gran variedad de recursos materiales de diversos tipos y que abarcan inclusive componentes de *hardware* de las computadoras. Cuando se lleva un control a este nivel de detalle, la información tiende a estar desactualizada, ya que muchos medios de este tipo se mueven o dañan con frecuencia, y el debido registro del reemplazo o movimiento de un medio se convierte en una tarea de recurrencia alta.

Por otra parte, en la UCI se lleva el control de los recursos materiales mediante varios sistemas que utilizan las diversas direcciones o áreas responsables de ciertos recursos. Estos sistemas manejan la información por sí mismos lo cual trae problemas de redundancia y sincronización de la información.
De esta forma, se vuelve complicado el tema de la asignación de recursos y la toma de decisiones respecto a los bienes materiales de la universidad. Siendo esta la **situación problémica.**

De lo anteriormente expuesto surge como **problema científico** de la presente investigación: ¿Cómo viabilizar el proceso de gestión de la información relacionada

con los procesos de control, inventario y seguimiento de los recursos materiales en la facultad 7 de la Universidad de las Ciencias Informáticas?

Para dar solución a este problema el **objeto de estudio** está dado por: El proceso de gestión de la información administrativa en la Universidad de las Ciencias Informáticas. Enmarcándose en el **campo de acción**: El proceso de gestión de la información relacionada con el proceso de control, inventario y seguimiento de los recursos materiales en la Facultad 7 de la Universidad de las Ciencias Informáticas.

Como **objetivo general** para resolver el problema planteado se propone: Desarrollar un sistema informático que facilite el proceso de gestión de la información relacionada con los procesos de control, inventario y seguimiento de los recursos materiales en la Facultad 7 de la Universidad de las Ciencias Informáticas.

Para dar cumplimiento al objetivo planteado, se realizarán las siguientes **tareas de la investigación**:

1. Analizar los sistemas de inventario y control de recursos materiales existentes en la nación e internacional.
2. Realizar un levantamiento de los procesos y requisitos de un sistema de gestión de los recursos materiales en la universidad.
3. Asimilar la metodología, lenguaje de programación, plataforma y Entorno Integrado de Desarrollo (IDE) definidos en la arquitectura del sistema para el desarrollo de la herramienta.
4. Diseñar un sistema que permita realizar el control y seguimiento de los recursos materiales.
5. Diseñar la base de datos que sustentará el sistema informático.
6. Implementar el sistema diseñado.
7. Realizar pruebas de caja negra a la solución desarrollada.

El presente trabajo está estructurado en cuatro capítulos que se describen a continuación:

Capítulo 1: Fundamentación teórica: se realiza un estudio detallado del estado del arte en el ámbito internacional, nacional y en nuestra Universidad. Se abordan las tendencias, tecnologías, herramientas, metodología y *software* actuales, además de la fundamentación del uso de la metodología y tecnología escogidas para la realización de este trabajo.

Capítulo 2: Características del sistema: Identificación de los procesos del negocio, también se describen los casos de uso, actores y trabajadores del mismo. Se

especifican las funcionalidades que el sistema una vez realizado debe ser capaz de efectuar. Además, se definen y detallan los casos de uso y actores del sistema.

Capítulo 3: Diseño del sistema: Se muestran los diagramas de clases del análisis para cada caso de uso del sistema. Igualmente se muestran algunos diagramas de las clases del diseño, junto con los correspondientes diagramas de interacción. Se refleja también el diagrama de despliegue y las restricciones que se tuvieron en cuenta para el diseño.

Capítulo 4: Implementación: Recoge aspectos que definen la implementación a partir del diagrama de despliegue y el diagrama de componentes además se determinan las pruebas a realizar en el sistema para verificar su integridad y si se ajusta a los requerimientos planteados por el cliente y se plantean posibles mejoras al sistema.

Capítulo 1: Fundamentación Teórica

El presente capítulo tiene como objetivo abordar los diferentes elementos que brindan la base teórica conceptual para el desarrollo de este trabajo. Se exponen además los antecedentes y características de los sistemas para la gestión de los recursos materiales, tomando esto como punto de partida para la concepción de la solución en cuestión. Además, se hace un análisis de las tecnologías, metodologías y herramientas de *software* empleadas en el proceso de desarrollo.

1.1 ANTECEDENTES, TENDENCIAS DE RECURSOS MATERIALES Y SISTEMAS AUTOMATIZADOS O SERVICIOS DE RECURSOS MATERIALES EXISTENTES

Llevar el control de los recursos materiales de cualquier institución es de gran importancia, ya que de esta manera se garantiza de manera óptima que se preserven estos recursos dándole la máxima seguridad al control, las políticas administrativas, la confidencialidad y la exactitud de las anotaciones de los documentos primarios. En resumen, ayuda a proteger los recursos contra el fraude, el desperdicio y el uso inadecuado, obteniendo eficacia, eficiencia y una buena economía.

En el mundo este control lo realizan diferentes empresas o de manera general cualquier institución, las que establecen a través de leyes, directivas y regulaciones el cumplimiento de su misión. Por ejemplo, en Guatemala se creó un sistema computarizado para el manejo de inventario y control de pedidos en una industria de elaboración de perfume. Este sistema busca optimizar los recursos con que una persona cuenta al trabajar, reduciendo tiempo y costos. (1)

El sistema permitirá obtener información de una manera rápida y exacta al tener definido cómo se controlarán físicamente los productos en la bodega y partiendo de ello para su fácil localización en el sistema. Se pretende que en la estructura del sistema se pueda consultar tanto la cantidad existente de cierto producto, como su ubicación física en los diferentes estantes a través del código. (2)

En Cuba, en la Provincia de Ciego de Ávila se creó un *software* para la Organización Nacional de Bufetes Colectivos (ONBC), el cual tiene como principal objetivo llevar el control de todos los recursos como bombillos, materiales de oficina entre otros. Este

sistema es propio de la organización, es un sistema obsoleto y está montado sobre la base de MS-DOS.

En Cuba se creó un *software* llamado Sistema Económico para las unidades de Salud (MEBUS), este tiene como principal objetivo llevar el control de todos los medios básicos de todas las unidades de salud, este incluye información de los submayores de los medios básicos, brinda la depreciación por cada medios y da la posibilidad de revalorizar el medio cuya depreciación haya caducado. Además, este sistema contiene una base de datos en la cual se plasman todos los materiales a los cuales se les haya dado de baja durante 5 años. (3)

Ninguno de estos sistemas puede ser utilizado pues no cumplen con los requisitos que se necesitan para la realización de esta aplicación, ya que están implementados en otros lenguajes de programación, son propios de su organización, uno de ellos está montado sobre una base de MS-DOS y es un sistema obsoleto.

1.2 HERRAMIENTAS Y TECNOLOGÍAS UTILIZADAS.

La Universidad de las Ciencias Informáticas (UCI), como productora de *software* tiene en consideración varias tecnologías, las cuales están distribuidas en los diversos proyectos de acuerdo con las necesidades de los mismos, de los clientes o de la propia universidad, de esta manera, se deben realizar las características de cada una para seleccionar la que satisfaga las necesidades específicas de un *software* determinado.

La facultad 7 de la Universidad de las Ciencias Informáticas, tiene como una de sus funciones la realización de un *software* para gestionar todos los recursos materiales. Existen varios lenguajes de programación como por ejemplo C y C++, *Visual Basic*, pascal, java, prolog, Python; pero se decidió utilizar Python. Como sistema gestor de base de datos se trabajará con el PostgreSQL, para modelar se utilizara Enterprise Arquitect. Y para modelar el negocio se realizará con UML 2.0.

Para llevar a cabo el desarrollo del *software* con las tecnologías anteriormente expuestas, como entorno de desarrollo (IDE) se escogió la herramienta Wing IDE. Con la finalidad de llevar a la práctica el *software*, enmarcado en la facultad 7 de la UCI, el cual es el eje central de este trabajo de diploma, se utilizarán las herramientas y

tecnologías mencionadas anteriormente, las que serán abordadas con más profundidad a continuación.

1.3 PYTHON

Python es un lenguaje dinámico de alto nivel. Es simple y ordenado además de potente y flexible. Está disponible para casi todas las plataformas. Muchos programas *opensource* están escritos en él. (4)

En la actualidad Python se desarrolla como un proyecto de código abierto. Los usuarios del mismo consideran a éste mucho más limpio y elegante para programar. Permite dividir el programa en módulos reutilizables desde otros programas. Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas (o como ejemplos para empezar a aprender a trabajar con él). (5)

Python se utiliza como lenguaje de programación interpretado, lo que ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar. El intérprete se puede utilizar de modo interactivo, lo que facilita experimentar con características del lenguaje, escribir programas desechables o probar funciones durante el desarrollo del programa. (6)

Python es un lenguaje de programación multiparadigma. Esto significa que más que forzar a los programadores a adoptar un estilo particular de programación, permite varios estilos: programación orientada a objetos, programación estructurada y programación funcional. Otros muchos paradigmas más están soportados mediante el uso de extensiones. (7)

Python usa tipo de dato dinámico y *reference counting* (Recuento de referencia) para el manejo de memoria. Una característica importante de Python es la resolución dinámica de nombres, lo que enlaza un método y un nombre de variable durante la ejecución del programa (también llamada ligadura dinámica de métodos). (8)

Ventajas

- Desarrollo más rápido: Se puede escribir un programa, salvarlo y ejecutarlo. En un lenguaje compilado tienes que pasar por los pasos de compilar y ligar el *software*, lo cual puede ser un proceso lento.

- Multiplataforma: El mismo código funciona en cualquier arquitectura, la única condición es que disponga del intérprete del lenguaje. No es necesario compilar el código una vez para cada arquitectura.

Inconvenientes

- Lentitud: Los programas interpretados son más lentos que los compilados. Sin embargo, los programas interpretados suelen ser cortos, en los que la diferencia es inapreciable.

1.4 DJANGO

Django es un *framework* para el desarrollo de aplicaciones Web basado en el lenguaje de programación python y que sigue el patrón de diseño MVC.

Sigue la arquitectura del "modelo-vista-controlador" (MVC). Dicho de forma sencilla, esto es una manera de desarrollar *software* que el código para definir y acceder a los datos (el modelo) esté separado de la lógica de negocio (el controlador), que a su vez está separada de la interfaz de usuario (la vista). (9)

Características básicas:

- Lenguaje Python.
- Desarrollo muy rápido.
- Similar a Ruby on Rails.
- Busca el perfeccionismo.
- Patrón MVC.
- Amplia comunidad.

1.5 POSTGRESQL

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) de código abierto. Comenzó como un proyecto denominado Ingres en la Universidad Berkeley de California. Ingres fue más tarde desarrollado comercialmente por la Relational Technologies/Ingres Corporation. (10)

En 1986 otro equipo dirigido por Michael Stonebraker de Berkeley continuó el desarrollo del código de Ingres para crear un sistema de bases de datos objeto-relacionales llamado Postgres. En 1996, debido a un nuevo esfuerzo de código abierto y a la incrementada funcionalidad del software, Postgres fue renombrado a PostgreSQL, tras un breve período como Postgres95. (11)

El proyecto PostgreSQL sigue actualmente un activo proceso de desarrollo en el mundo, gracias a un equipo de desarrolladores y contribuidores de código abierto. PostgreSQL posee muchas características que normalmente sólo se encontraban en las bases de datos comerciales tales como DB2 u Oracle. La siguiente es una breve lista de algunas de esas características: (12)

- DBMS Objeto-Relacional: PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multa-versión, soporte multa-usuario, optimización de consultas, herencia, y *arrays*.
- Altamente Extensible: PostgreSQL soporta operadores funcionales, métodos de acceso y tipos de datos definidos por el usuario.
- Soporte_SQL_Comprensivo: Soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.
- Integridad Referencial: PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.
- API Flexible: La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, PERL, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, y Pike.
- Lenguajes Procedurales: PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar PERL, Python, o TCL como lenguaje procedural embebido.
- MVCC: MVCC, o Control de Concurrencia Multi-Versión (Multi-Versión Concurrency Control), es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. MVCC está considerado mejor que el bloqueo a nivel de fila porque un lector nunca es bloqueado por un escritor. En su lugar, PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos. PostgreSQL es capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles.
- Cliente/Servidor: PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar

procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

- Write Ahead Logging (WAL): La característica de PostgreSQL conocida como “Write Ahead Logging” incrementa la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del cual se puede restaurar la base de datos.

1.6 ENTERPRISE ARQUITECT (EA)

Enterprise Architect combina el poder de la última especificación UML 2.1 con alto rendimiento, interfaz intuitiva, para traer modelado avanzado al escritorio, y para el equipo completo de desarrollo e implementación. EA puede ser utilizado por un equipo entero, incluyendo analistas, evaluadores, administradores de proyectos, personal del control de calidad, equipo de desarrollo y más, por una fracción del costo de algunos productos competitivos. (13)

Enterprise Architect es una herramienta comprensible de diseño y análisis UML, cubriendo el desarrollo de *software* desde el paso de los requerimientos a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento. Es Multi-usuario, basada en Windows, diseñada para ayudar a construir *software* robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad. (14)

Además, soporta generación e ingeniería inversa de código fuente para muchos lenguajes populares, incluyendo C++, C#, Java, VB.Net, *Visual Basic* y PHP. También hay Add-ins gratis para Corba y Python disponibles. Con un editor de código fuente con "resaltador de sintaxis" incorporado, esta herramienta le permite navegar y explorar su modelo de código fuente en el mismo ambiente. (15)

Para aquellos que trabajan en Eclipse o Visual Studio.Net, Sparx Systems también vende puentes livianos para estas IDE's, permitiéndole modelar en este y saltar directamente al código fuente en su editor preferido. Las plantillas de generación de código que permiten personalizar el código fuente generado a las especificaciones de su compañía. (16)

EA le ayuda a visualizar sus aplicaciones soportando ingeniería inversa de un amplio rango de lenguajes de desarrollo de *software* y esquemas de repositorios de base de datos. Ingresar framework completos desde código fuente o archivos Java .jar o aún ensambladores binarios .Net Importando framework y librerías de código, se puede maximizar la re-utilización y entendimiento de su inversión existente. (17)

Soporta transformaciones de Arquitectura avanzada dirigida por Modelos (MDA) usando plantillas de transformaciones de desarrollo y fáciles de usar. Con transformaciones incorporadas para DDL, C#, Java, EJB y XSD, Ud. puede rápidamente desarrollar soluciones complejas desde los simples "modelos independientes de plataforma" (MIP) que son el objetivo en "modelos específicos de plataforma" (MEP). (18)

1.7 UML2.0

UML2.0 (Unified Modeling Language) por sus siglas en inglés, el Lenguaje Unificado de Modelado es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object *Management* Group) (19)

Las características más generales de UML2.0 son:

- Tecnología de orientación a objetos.
- Viabilidad en la corrección de errores.
- Desarrollo incremental e iterativo.
- Participación del cliente en todas las etapas del proyecto.

Es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de *software*. Además, entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de *software* reutilizable. (20)

El UML prescribe un conjunto de notaciones y diagramas estándares para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Mientras que ha habido muchas notaciones y métodos usados para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación. En resumen UML es el resultado de la unión de tres metodologías, Booch, OMT, y OOSE. (21)

Estas han tenido una aplicación extensa en el campo de la POO (Programación orientada a objetos), tienen su historia, y han sido aplicadas en una gran variedad de industrias y problemas, por lo que pueden ser clasificadas como maduras. (22)

Entre las ventajas que brinda el UML2.0 se encontraron:

- Mejora el nivel de comunicación formal.
- Se desarrollan los procesos y los productos con una mayor fiabilidad y calidad.
- Se define, organiza y comparte conocimiento.
- El esfuerzo de especificación es más eficiente.
- El impacto de las decisiones sobre un proceso o producto es más visible.

1.8 WING IDE 3.2

Es un entorno de desarrollo totalmente integrado para Python, muy veloz y dotado de potentes herramientas de edición y depuración de código, así como buenas aptitudes de búsqueda optimizadas, lo que permite reducir en gran medida las probabilidades de errores y hace más fácil la navegación a través del código Python. Wing IDE Professional es una alta calidad del medio ambiente de desarrollo integrado (IDE) para el tiempo de las velocidades de Python, desarrolla y mejora la calidad de código con un montón de inteligencia de gran alcance del código, depuración y las características del editor.

Wing IDE y Python para el desarrollo rápido de la multiplataforma de escritorio y aplicaciones web, integración de aplicaciones empresariales, pruebas de software, y las secuencias de comandos de aplicaciones. A diferencia de muchas tecnologías competidoras, Wing IDE y Python le permiten concentrarse en la construcción de la aplicación de la funcionalidad específica.

1.9 METODOLOGÍA DE DESARROLLO

El Rational Unified Process (RUP) o Proceso Unificado de Desarrollo, es un proceso de amplio marco que ofrece las mejores prácticas para el *software* y los sistemas de entrega, la aplicación eficaz y la gestión de proyectos. Constituye una de las metodologías estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objeto. (23)

RUP es un proceso de desarrollo de *software*, que se clasifica como un proceso pesado, se basa en casos de uso para describir lo que se espera del software y está

muy orientado a la arquitectura del sistema, documentándose lo mejor posible, utilizando UML (Unified Modeling Language) como herramienta principal de modelado. Se caracteriza por ser iterativo e incremental. (24)

Define artefactos (que son los productos tangibles del proceso ejemplo: código fuente), trabajadores o roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso), actividades (tarea que tiene un propósito claro, es realizada por un trabajador) y flujo de actividades (secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.) (25)

En el presente capítulo se exponen los antecedentes y características de los sistemas para la gestión de los recursos materiales. Además, se hace un análisis de las tecnologías, metodologías y herramientas de *software* empleadas en el proceso de desarrollo.

Capítulo 2: Características del sistema

El presente capítulo tiene una vital importancia para el desarrollo del sistema, puesto que se realiza una identificación de los procesos del negocio, se describen los casos de uso, actores y trabajadores del mismo. Además, se especifican las funcionalidades que el sistema una vez realizado debe ser capaz de efectuar, se definen y detallan los casos de uso y actores del sistema.

2.1 OBJETO DE AUTOMATIZACIÓN

2.1.1 Descripción del objeto de estudio

EL objeto de estudio del presente trabajo es el proceso de gestión de la información relacionada con el control, inventario y seguimiento de los recursos materiales. Cuando este trabajo se hace de forma manual se vuelve un mecanismo muy lento y engorroso. Para que no exista pérdida de ninguna información y el trabajo sea más rápido, debe existir un sistema *online* donde diariamente se realicen todos los reportes y todas las tareas que tengan que ver con los recursos de cada centro.

De esta forma, se ayuda a mejorar de manera óptima que se preserven los mismos dándole la máxima seguridad, control, las políticas administrativas, la confidencialidad y la exactitud de las anotaciones de los documentos primarios. En resumen, se ayuda a protegerlos contra el fraude, el desperdicio y el uso inadecuado.

Propuesta del sistema para resolver esta problemática

Desarrollar un sistema informático que facilite el proceso de gestión de la información relacionada con los procesos de control, inventario y seguimiento de los recursos materiales en la Facultad 7 de la Universidad de las Ciencias Informáticas.

2.1.2 Modelo del Negocio

Un modelo de negocio (también llamado diseño de negocio) es el mecanismo por el cual un negocio trata de generar ingresos y beneficios. Es un resumen de cómo una compañía planifica servir a sus clientes. Implica tanto el concepto de estrategia como el de implementación.

Identificación de roles del entorno del negocio.

Actor del Negocio: Cualquier individuo, grupo u organización externa; con los que el negocio interactúa.

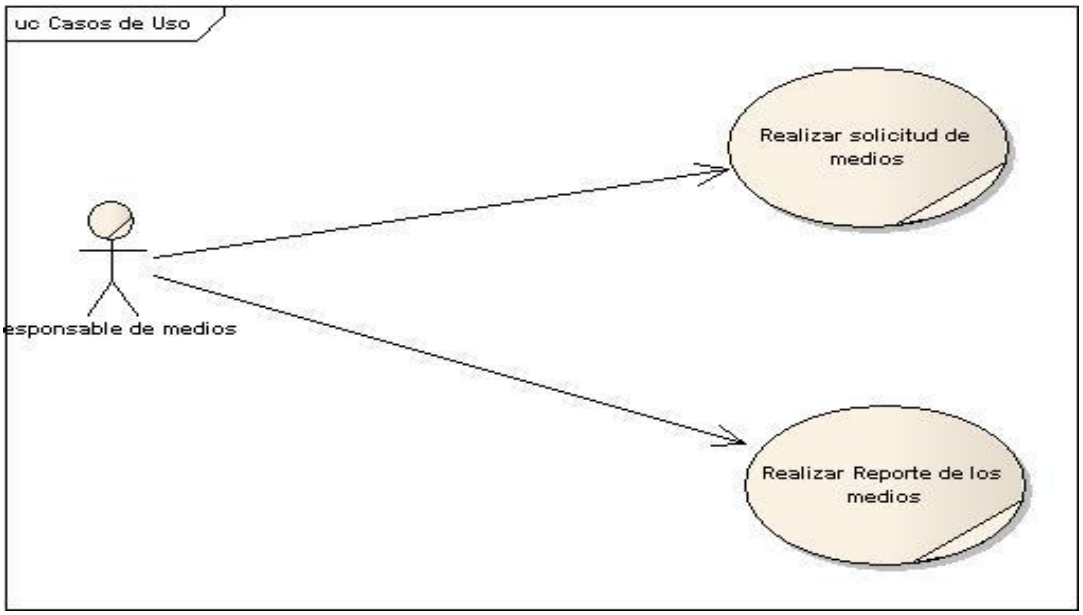
Usuario	Justificación
Responsable de Medios	Es el responsable de hacer reportes diarios del estado de los medios y de realizar solicitud de medios.

Trabajador del Negocio: Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.

Trabajador	Justificación
Administrador de Medios	Es el encargado de revisar los reportes, verificar de qué tipo es el daño y controlar los medios.

Caso de Uso: Fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores.

Diagrama de Caso de Uso del Negocio: representa gráficamente a los procesos del negocio y su interacción con los actores del negocio.



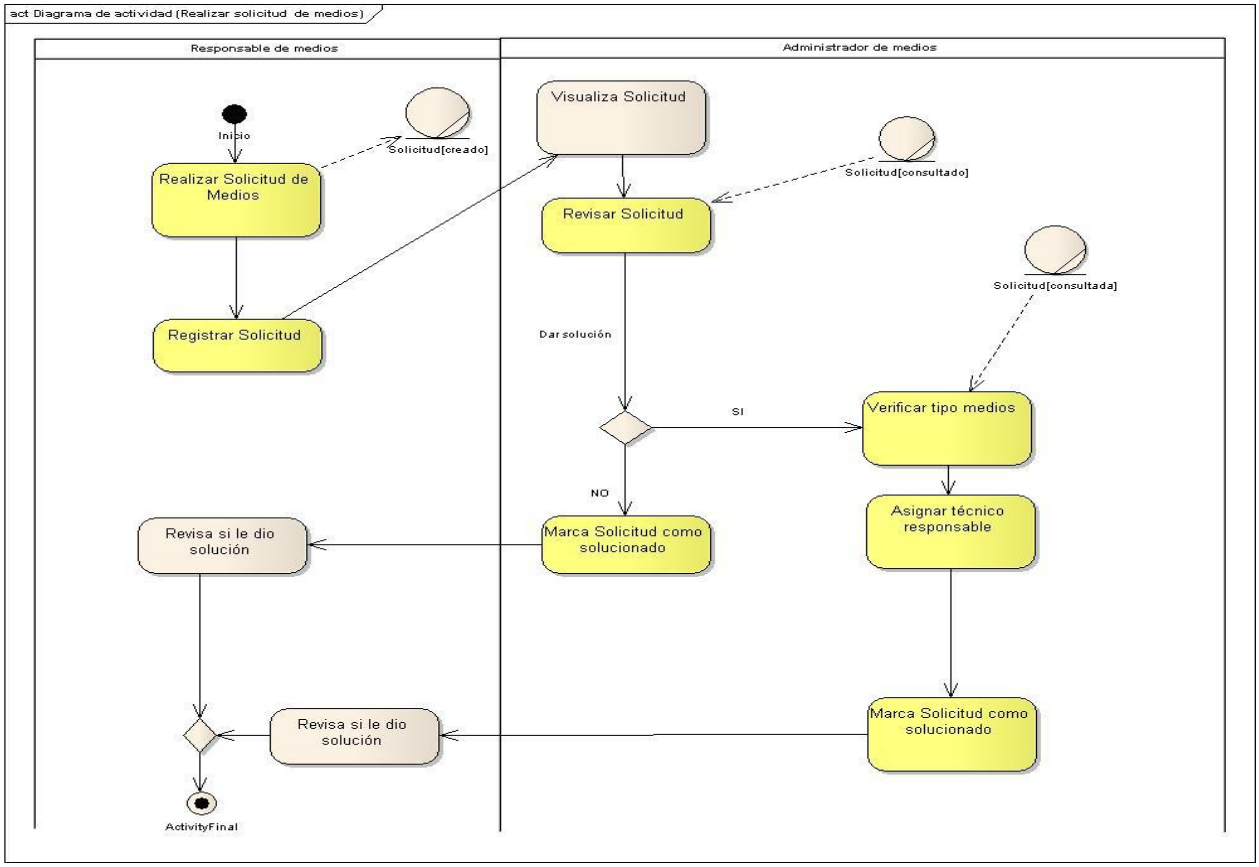
Caso de Uso:	Realizar reportes de medios.
Actores:	Responsable de medios.

Trabajadores:	Administrador de medios.
Resumen: El caso de uso se inicia cuando el Responsable de cada aula, laboratorio o departamento registra un reporte diario del estado en que se encuentran todos los medios.	

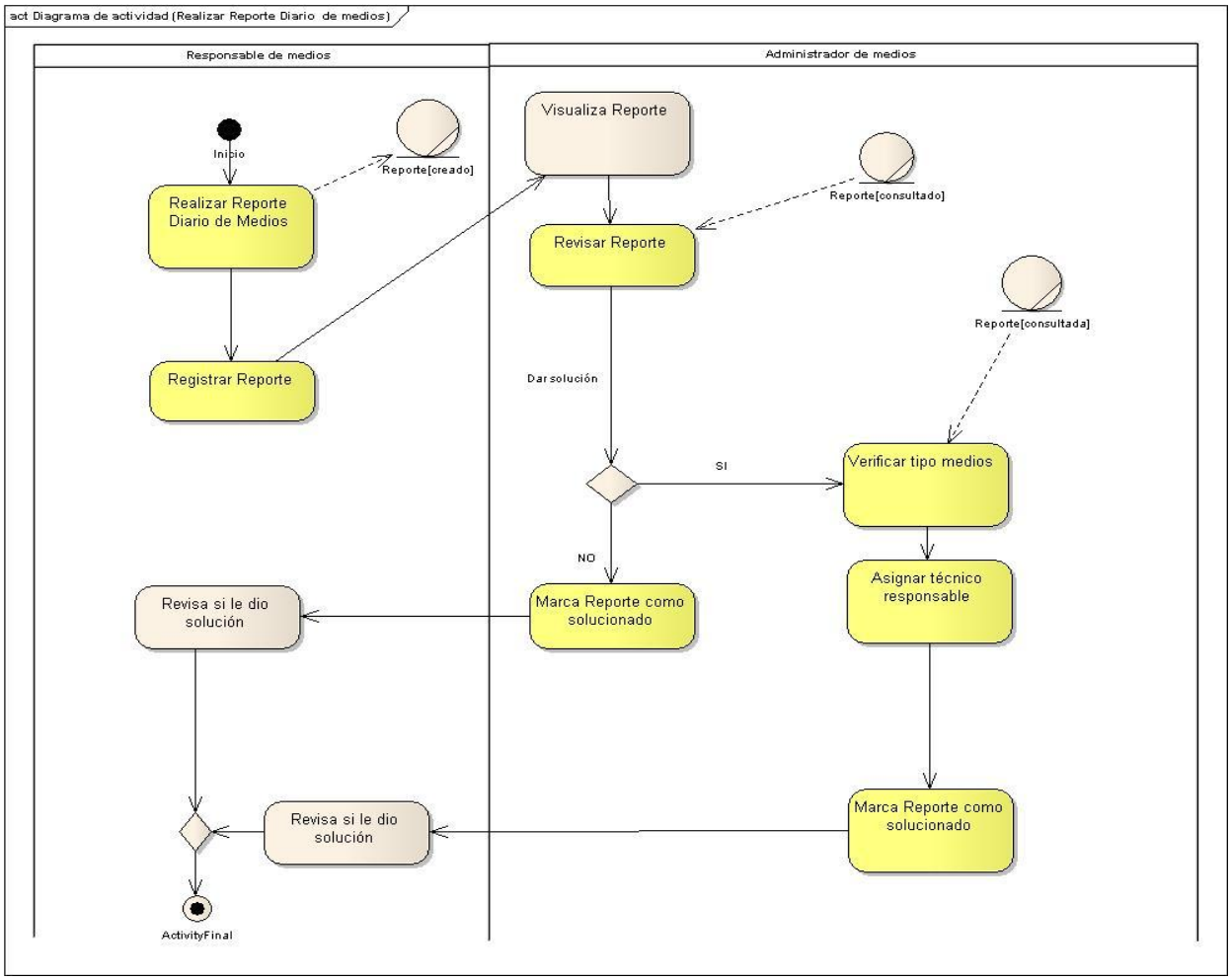
Caso de Uso:	Realizar Solicitudes de medios.
Actores:	Responsable de medios.
Trabajadores:	Administrador de medios.
Resumen: El caso de uso se inicia cuando el Responsable de cada aula, laboratorio o departamento registra una solicitud si existe alguna incidencia durante el día de pérdida o daño de algún medio.	

Diagrama de Actividades: contiene estados en 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.

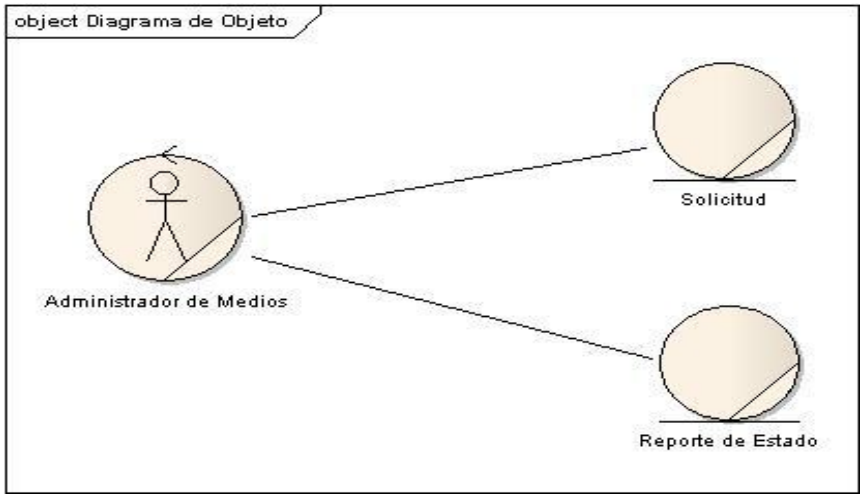
Realizar solicitud de los medios.



Realizar reportes de los medios.



Modelo de Objetos: muestra la participación de los trabajadores y entidades del negocio y la relación entre ellos.



2.2 Especificación de los requisitos de *software*.

Requisitos funcionales: Se basa en los casos de uso. Describen de qué forma el usuario va a utilizar el sistema. Cada usuario requiere de varios casos de uso. Los analistas proponen cómo será la interfaz del sistema esbozando varias versiones para que el usuario decida. Característica requerida del sistema que expresa una capacidad de acción del mismo; una funcionalidad; generalmente expresada en una declaración en forma verbal. [26]

2.2.1 Requerimientos Funcionales:

- RF1: Registrar reporte diario.
- RF2: Registrar solicitud.
- RF3: Generar reporte.
- RF4: Generar solicitud.
- RF5: Gestionar los reportes diarios.
- RF6: Gestionar las solicitudes.
- RF7: Configurar el sistema.

Requisitos no funcionales: Especifica las propiedades del sistema que tienen que ver con rendimiento, velocidad, uso de memoria, plataforma. Fiabilidad: tiempo de respuesta media, defectos por miles de líneas de código. Imponen condiciones a requisitos funcionales. [27]

Llamamos requisito no funcional a todas las exigencias de cualidades que se imponen al proyecto: exigencias de usar un cierto lenguaje de programación o plataforma tecnológica, por ejemplo, un requisito no funcional es una característica ya sea del sistema, del proyecto o del servicio de soporte, que es requerida junto con la especificación del sistema pero que como se dijo anteriormente, no se satisface añadiendo código, sino cumpliendo con esta como si de una restricción se tratara.

2.2.2 Requerimientos no Funcionales:

- **Usabilidad:** El sistema debe ser diseñado de forma tal que los usuarios que hagan uso del mismo obtengan los conocimientos necesarios en el menor tiempo posible para la explotación de sus funcionalidades.
- **Seguridad:** Es el requerimiento más complejo y difícil de garantizar, a su vez permitirá garantizar que la aplicación será utilizada correctamente por cada usuario según sus niveles permitidos.
    - **Confidencialidad:** Se debe tratar el manejo de permisos de forma que solamente se acceda a la información autorizada de acuerdo con los niveles

de permisos que debe tener cada usuario del sistema. De esta forma se garantiza que la información no sea expuesta a personal indebido.

- **Integridad:** Se debe de tratar el manejo de la información de forma tal que la información no sea modificada por personal ajeno evitando de esta forma alteraciones en los resultados planteados en la documentación.
- **Disponibilidad:** Se deberá garantizar el acceso pleno a cada usuario con facultades para el uso de la aplicación las 24 horas del día

- **Documentación en línea y ayuda a los usuarios:** El sistema debe estar provisto de ayudas dinámicas y documentos (manuales de usuarios o tutoriales) que permitan el uso eficiente del sistema y el respaldo a las dudas en su uso.

- **Soporte:** Debe presentar un diseño que permita la realización de pruebas y la explotación de la aplicación de forma eficiente. Se incluye la posibilidad de brindar asistencia técnica que permita la solución de problemas en tiempo real de ejecución que garantice la solución de fallas que pueda presentar. Debe ser garantizado su adaptabilidad y compatibilidad con los distintos Sistemas Operativos.

  Como garantía del producto se especificará los requerimientos de legalidad que permitan la autoría del propietario, el derecho de autor y todos los requerimientos legales que se establezcan entre las partes involucradas.

- **Interfaz:** El diseño de la interfaz visual debe ser atractivo para los usuarios que interactuaran con la aplicación de forma que les facilite el entendimiento de las funcionalidades que el mismo brinda además de poseer colores amigables y refrescantes para una mejor interacción entre usuario y la aplicación.

  La interfaz constará de un grupo de botones, editores para textos, desplegables e informaciones que permitirán al usuario un fácil manejo de la aplicación y una realización rápida de las actividades. Toda la información referente a los textos, mensajes y opciones serán mostrados en idioma español.

  El diseño de la interfaz permitirá mostrar mensajes para la guía de usuarios en caso de errores en entradas inválidas de los datos o de confirmación de realización de actividades. La interfaz permitirá además la posibilidad de obtener informaciones de salidas en diferentes formatos con posibilidad de ser impresos.

- ***Software*:** Para la elaboración de la aplicación se utilizará una computadora Pentium 4 a 3.0 GHz de CPU y 513 MB de RAM, como sistema operativo Windows XP con *Service* Pack 3. En caso crítico puede ser utilizado Ubuntu como sistema operativo para el desarrollo de la aplicación.

- ***Hardware:***

  En el cliente:

  - Procesador Intel Pentium III o superior.
  - 2 Gb de memoria RAM.
  - Monitor tipo VGA o superior.
  - Tarjeta de Red.

  En el servidor:

  - Procesador Intel Pentium IV o superior.
  - 512 Mb de memoria RAM.
  - Disco duro de 80 Gb o más.
  - Monitor tipo VGA o superior.
  - Tarjeta de Red.

2.3 Definición de los casos de uso y actores del sistema.

2.3.1 Actores del Sistema.

Actores	**Justificación**
Responsable de los medios.	Representa a un usuario que desempeña el rol de responsable del local al cual esté asignado, es el encargado de registrar solicitudes y reportes de medios.
Administrador de medios.	Representa a un usuario el cual tiene como responsabilidad llevar el control de todas las solicitudes y reportes, darle solución a cada una de estas, así como asignándole un técnico responsable que le dé solución.

2.3.2 Casos de uso del sistema.

Casos de uso del sistema: Generar reporte, Generar solicitud, Gestionar reporte diario, Gestionar solicitudes, Registrar reporte diario, Registrar solicitud, Configurar sistema.

CU-1	Registrar reporte diario.
Actor	Responsable de los medios.
Descripción	Este caso de uso permite al responsable de los medios realizar un reporte diario del local al cual esté asignado.

Referencia	RF1.

CU-2	Registrar solicitudes.
Actor	Responsable de los medios.
Descripción	Este caso de uso permite al responsable de los medios dejar registrado las solicitudes que se realizan de los locales al cual esté asignado, si durante el día existe alguna incidencia de pérdida o daño de algún medio.
Referencia	RF2.

CU-3	Generar reporte diario.
Actor	Responsable de los medios.
Descripción	Este caso de uso es el que posibilita la obtención de todos los reportes que necesita el usuario. Los reportes pueden ser por pérdida o deterioro de algún recurso del local al cual esté asignado este usuario.
Referencia	RF3.

CU-4	Generar solicitud.
Actor	Responsable de los medios.
Descripción	Este caso de uso es el que posibilita la obtención de todas las solicitudes que necesita el usuario. Las solicitudes pueden ser por pérdida o deterioro de algún recurso del local al cual esté asignado este usuario.
Referencia	RF4.

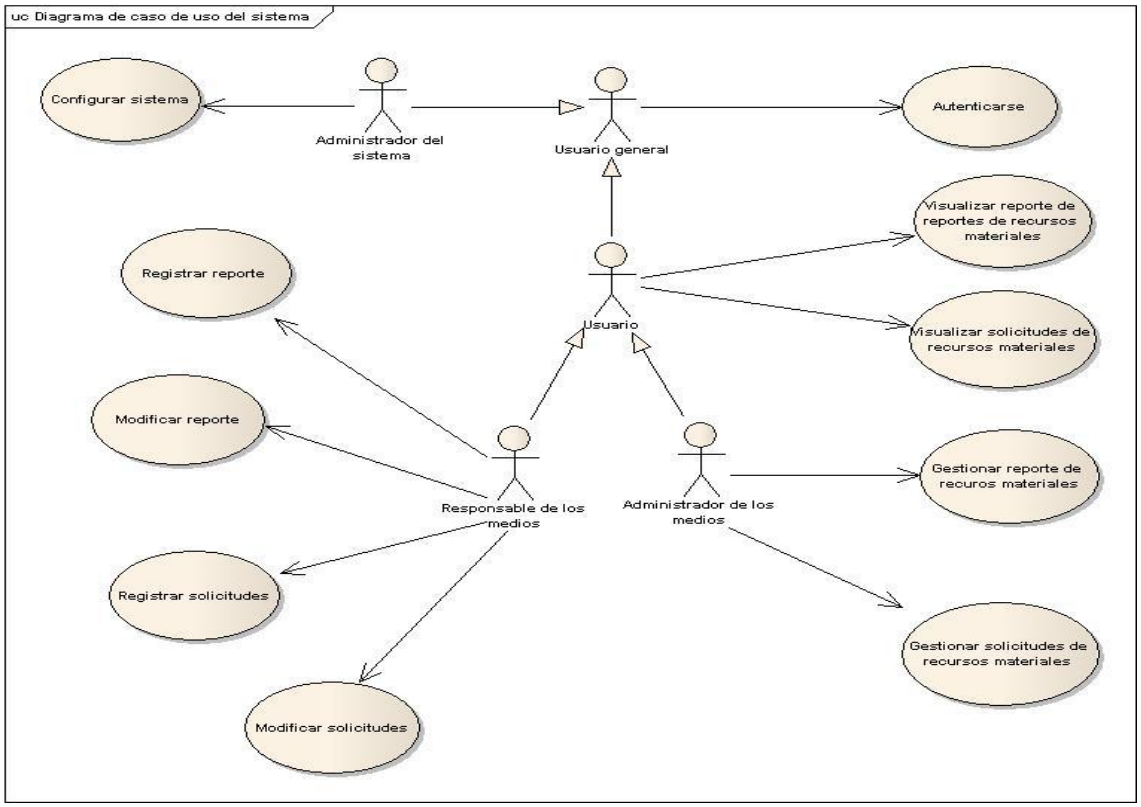
CU-5	Gestionar reporte diario.
Actor	Administrador de medios
Descripción	Este caso de uso permite que el administrador de medios pueda gestionar todos los reportes realizados por el responsable de los medios, una vez que revise los reportes hechos, verifica si le puede dar solución o no. Si puede darle solución envía a un técnico responsable para que le dé solución.

Referencia	RF5.

CU-6	Gestionar solicitud.
Actor	Administrador de medios.
Descripción	Este caso de uso permite que el administrador de medios pueda gestionar todas las solicitudes realizadas por el responsable de los medios, una vez que revise las solicitudes hechas, verifica si le puede dar solución o no. Si puede darle solución envía a un técnico responsable.
Referencia	RF6.

CU-7	Configurar sistema.
Actor	Administrador del sistema.
Descripción	Este caso de uso permite que el administrador pueda configurar el sistema, de manera que cuando el usuario acceda al mismo tenga acceso a la información que el pueda utilizar en dependencia del rol que desempeñe como administrador de los medios o responsable de los medios.
Referencia	RF7.

2.3.3 Diagrama de casos de uso del sistema.



2.3.4 Casos de uso expandidos.

Caso de uso		
CU-1	Registrar reporte diario.	
Propósito	Permitir notificar el estado de los recursos materiales.	
Actores: Responsable de los medios.		
Resumen: El caso de uso se inicia cuando el responsable de los medios, una vez realizado el reporte, accede a la aplicación para dejar registrado y documentado el estado de todos los medios.		
Referencias	RF1	
Precondiciones	El Responsable de los medios tiene que haberse autenticado.	
Poscondiciones	Se archiva el reporte realizado.	
Acción del actor		**Respuesta del sistema**
1. El responsable de los medios selecciona en la página principal la opción de Registrar reporte.		2. El sistema muestra la ventana para realizar el reporte.
3. El usuario entra los datos.		4. Guarda los datos del reporte en la aplicación.

Flujo alternativo	
Acción del actor	**Respuesta del sistema**
	5. No se pudo registrar el reporte, se muestra nuevamente la ¨ Página Principal ¨.

Caso de uso		
CU-2	Registrar solicitud	
Propósito	Permitir notificar el estado de los recursos materiales.	
Actores: Responsable de los medios.		
Resumen: El caso de uso se inicia cuando el responsable de los medios, una vez realizada la solicitud, accede a la aplicación para dejar registrado y documentado el estado de todos los medios.		
Referencias	RF2	
Precondiciones	El Responsable de los medios tiene que haberse autenticado.	
Poscondiciones	Se archiva la solicitud realizada.	
Acción del actor		**Respuesta del sistema**
1. El responsable de los medios selecciona en la página principal la opción de Registrar solicitud.		2. El sistema muestra la ventana para registrar la solicitud.
3. El usuario registra la solicitud.		4. Guarda los datos de la solicitud
Flujo alternativo		
Acción del actor		**Respuesta del sistema**
		5. No se pudo registrar la solicitud, se muestra nuevamente la ¨ Página Principal ¨.

Caso de uso	
CU-3	Generar reporte
Propósito	Permitir ver todos los reportes realizados por el administrador que sería el encargado de darle solución a los problemas existentes.
Actores: Responsable de los medios y el administrador de los medios.	
Resumen: El caso de uso inicia cuando el usuario, en dependencia del rol que desempeña, y del nivel de acceso a la información que tenga, solicite un reporte y el sistema muestra los reportes que desea ver.	
Referencias	RF3

Precondiciones	El responsable de los medios o el administrador de los medios tienen que haberse autenticado.
Poscondiciones	Se recibe el reporte solicitado.

Acción del actor	**Respuesta del sistema**
1. El usuario selecciona la opción ´´Generar Reportes´´	2. Muestra todos los reportes
3. El usuario selecciona el reporte que desea, y presiona la tecla aceptar o cancelar.	4. El sistema le muestra el reporte seleccionado.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
	5. No se genera el reporte y se muestra la pantalla ¨ Principal ¨.

Caso de uso	
CU-4	Generar solicitud
Propósito	Permitir ver todas las solicitudes realizadas por el administrador que sería el encargado de darle solución a los problemas existentes.
Actores: Responsable de los medios y el administrador de los medios.	
Resumen: El caso de uso inicia cuando el usuario, en dependencia del rol que desempeña, y del nivel de acceso a la información que tenga solicite una solicitud y el sistema muestra las solicitudes que desea ver.	
Referencias	RF4
Precondiciones	El responsable de los medios o el administrador de los medios tienen que haberse autenticado.
Poscondiciones	Se recibe la solicitud solicitada.

Acción del actor	**Respuesta del sistema**
1. El usuario selecciona la opción ´´Generar Solicitud´´.	2. Muestra todas las solicitudes que estén registrados
3. El usuario selecciona la solicitud que desea, y presiona la tecla aceptar o cancelar.	4. El sistema le muestra la solicitud seleccionada.
Flujo alternativo	

Acción del actor	Respuesta del sistema
	5. No se genera la solicitud y se muestra la pantalla ¨ Principal ¨.

Caso de uso		
CU-5	Gestionar reporte diario	
Propósito	Darle solución a todos los reportes diarios que se hayan registrado.	
Actores: Administrador de los medios		
Resumen: El caso de uso se inicia cuando el administrador de los medios entra a la aplicación para revisar los reportes diarios que se realizan acerca del estado de los recursos materiales de todos los locales de la facultad 7, y de esta manera tratar de solucionarlo buscando un técnico responsable que le de solución al mismo. Si le puede o no dar solución enviará mediante el correo la respuesta.		
Referencias	RF5	
Precondiciones	El administrador de los medios tiene que haberse autenticado.	
Poscondiciones	Queda almacenado el reporte al que se le dio solución o no.	
Acción del actor		**Respuesta del sistema**
1. El administrador de los medios selecciona del menú la opción deseada. Gestionar Reportes		2. El sistema le permite: Ver todos los reportes del día.
3. El administrador de lo medios selecciona el reporte que analizará.		4. El sistema le muestra el reporte seleccionado.
5. El administrador de los medios le trata de dar solución al problema que tenga el reporte, así como asignándole un técnico responsable. Si le puede dar solución o no marca la opción de si le dio o no solución.		6. El sistema guarda la respuesta de si le dio o no solución, donde la misma sería marcar una opción que tendrá acceso solamente el administrador de los medios, solucionada o no solucionado.
Flujo alternativo		
Acción del actor		**Respuesta del sistema**
		5. No se pudo ver los reportes hechos y se muestra la ¨ Página Principal ¨ para volver a intentarlo.

Caso de uso	
CU-6	Gestionar solicitud
Propósito	Darle solución a todas las solicitudes registradas.
Actores: Administrador de los medios	
Resumen: El caso de uso se inicia cuando el administrador de los medios entra a la aplicación para revisar si se ha realizado alguna solicitud de alguna incidencia que haya ocurrido durante el día, y de esta manera tratar de solucionarlo buscando un técnico responsable que le dé solución a la misma. Si le puede o no dar solución enviará mediante el correo la respuesta.	
Referencias	RF6
Precondiciones	El administrador de los medios tiene que haberse autenticado.
Poscondiciones	Queda almacenado la solicitud al que se le dio solución o no.

Acción del actor	**Respuesta del sistema**
1. El administrador de los medios selecciona del menú gestionar solicitud.	2. El sistema le permite: Ver todas las solicitudes del día.
3. El administrador de los medios selecciona la solicitud que analizará.	4. El sistema le muestra la solicitud seleccionada.
5. El administrador de los medios le trata de dar solución a la solicitud, así como asignándole un técnico responsable que le dé solución a la misma.	6. El sistema guarda la respuesta de si le dio o no solución, donde la misma sería marcar una opción que tendrá acceso solamente el administrador de los medios, solucionada o no solucionado.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
	5. No se pudo ver las solicitudes hechas y se muestra la ¨ Página Principal ¨ para volver a intentarlo.

Caso de uso	
CU-7	Configurar sistema

Propósito	Permitir que el administrador asigne a cada responsable y administrador de los medios cual sería su responsabilidad. De esta manera, una vez que el usuario se autentica, tiene acceso solo a la información correspondiente al rol que desempeña.
Actores: Administrador del sistema.	
Resumen: El caso de uso se inicia cuando el administrador del sistema accede a la aplicación para hacer las asignaciones correspondientes. El sistema le brinda la posibilidad de asignarles al responsable y el administrador de los medios la responsabilidad de cada uno.	
Referencias	RF7
Precondiciones	El sistema tiene que estar habilitado.
Poscondiciones	El sistema quedaría configurado.

Acción del actor	**Respuesta del sistema**
1. El administrador del sistema accede a la aplicación y llena los datos de configuración.	2. El sistema los almacena en la BD.

Flujo alternativo	
Acción del actor	**Respuesta del sistema**
	No se pudo hacer cambios en el sistema por algún error de la aplicación, volver a la ´´Pagina Principal´´ para volver a intentarlo.

CAPÍTULO 3. Análisis y Diseño del Sistema

El presente capítulo tiene una gran importancia ya que partiendo del buen análisis y diseño de una aplicación, se logra un desarrollo más rápido y ordenado. En este capítulo se modelará el análisis con los diagramas de clase del análisis. De esta forma, el sistema propuesto será más entendible. Luego se realiza el modelado del diseño para que de esta forma el mismo dé cumplimiento a los requisitos funcionales y no funcionales. Además, se realizan los diagramas de clases y la descripción de las mismas, todo esto mediante la realización de los casos de usos definidos en el capítulo anterior. Se describen los principios del diseño y los diagramas de interacción.

3.1 Descripción de la arquitectura. Fundamentación.

La Arquitectura de un *Software* indica la estructura, funcionamiento e interacción entre las partes del *software*. Es un nivel de diseño que se centra principalmente en aspectos más allá de la estructura de datos de la computación.

El Modelo Vista Controlador (MVC) es un estilo de arquitectura del *software* que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. El modelo es el sistema de gestión de base de datos, la lógica del negocio y el controlador son los responsables de recibir los eventos de entrada desde la vista.

3.2 Análisis.

Modelo de análisis:

El análisis es una parte importante en el desarrollo de un proyecto ya que gracias a él se obtiene una visión más clara y detallada del sistema. Con el análisis se comienzan las actividades del diseño y este se basa en los requisitos funcionales. A pesar de que durante el modelado del análisis se hayan refinado los requisitos.

Diagramas de clases de análisis:

El diagrama de clases del análisis se realiza para cada caso de uso del sistema y muestra las clases participantes en dicho caso de uso y sus relaciones. En los diagramas de clases se identifican tres tipos de clases: Interfaz (Modelan la interacción entre el sistema y sus actores), Controladora (Coordinan la realización de uno o unos pocos casos de uso coordinando las actividades de los objetos que implementan la

funcionalidad del caso de uso.) y Entidad (Modelan información que posee larga vida y que es a menudo persistente).

- **Modelo:** Esta es la representación específica de la información con la cual el sistema opera. En resumen, el modelo se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas. El sistema también puede operar con más datos no relativos a la presentación, haciendo uso integrado de otras lógicas de negocio y de datos afines con el sistema modelado.
- **Vista:** Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- **Controlador:** Este responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y probablemente a la vista.

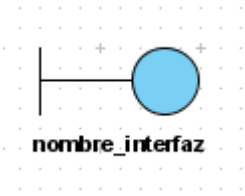
Representación	Nombre	Características
	Interfaz	Se utiliza para modelar la interacción entre el sistema y sus actores. Representan ventanas, formularios, paneles, interfaces de comunicación.
	Entidad	Se utiliza para modelar información que posee una vida larga y que es a menudo persistente.
	Control	Representan coordinación, secuencia, transacciones y control de los objetos y se usan para encapsular el control de un caso de uso en concreto.

Diagrama de clases del análisis Registrar Solicitud.

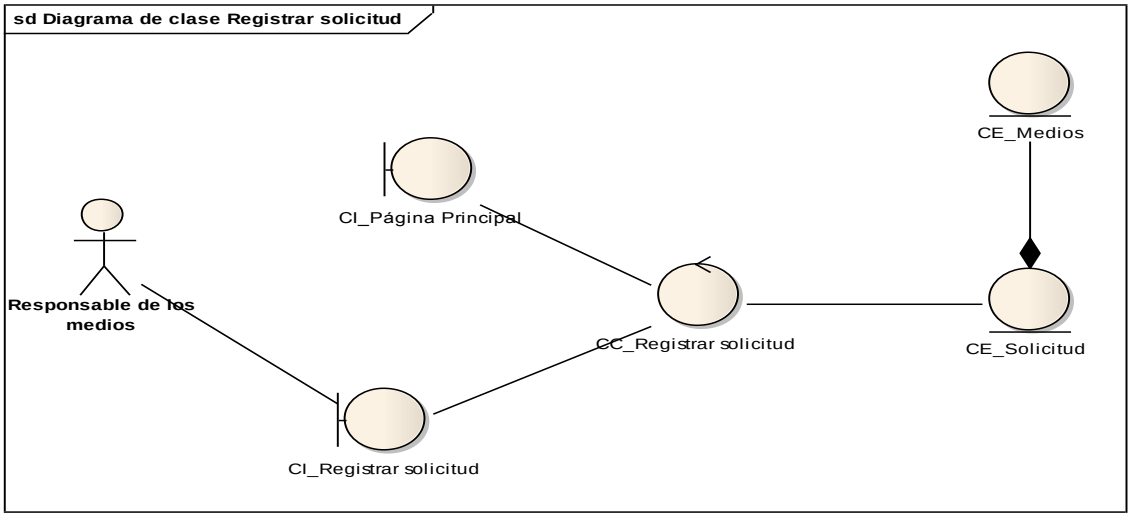


Diagrama de clases del análisis Registrar Reporte Diario.

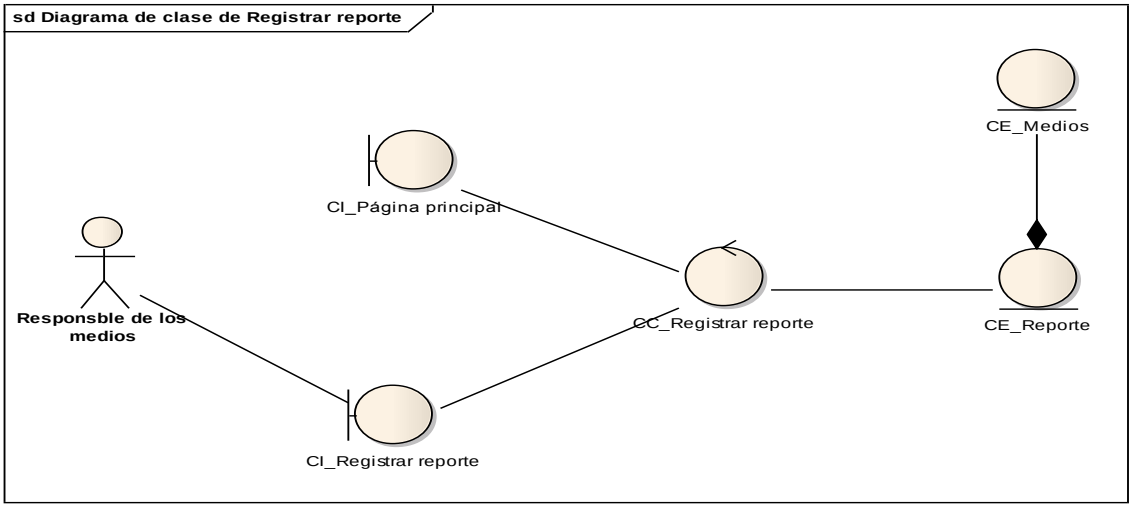


Diagrama de clases del análisis Gestionar Solicitud.

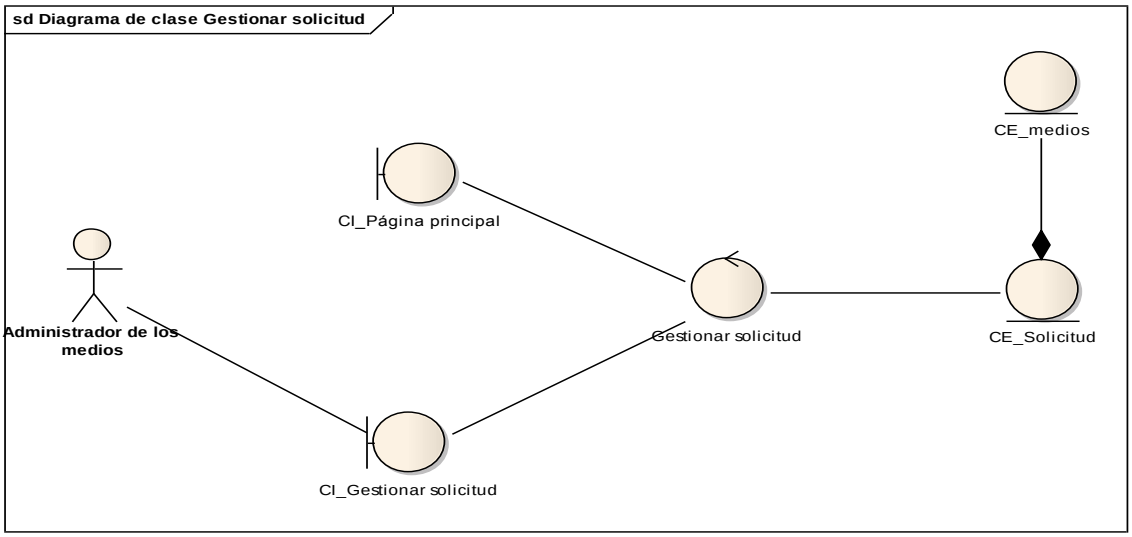
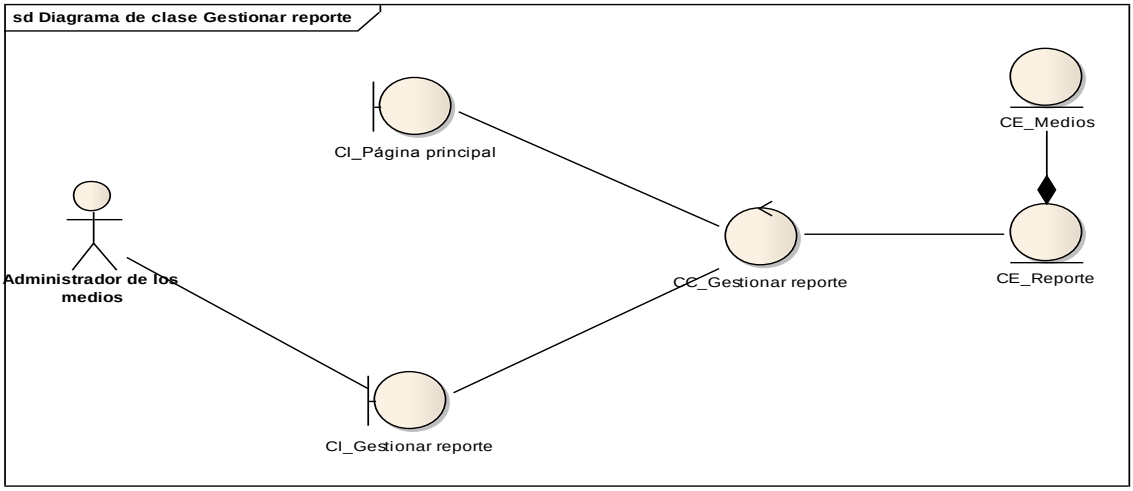


Diagrama de clases del análisis Gestionar Reporte Diario.



3.3 Descripción de las clases:

Clases Interfaz:

CI_GenerarReporte: Permite acceder a los reportes.

CI_RegistrarSolicitud: Permite entrar los datos para enviar la solicitud.

CI_RegistrarReporte: Permite entrar los datos para enviar el reporte.

CI_GestionarSolicitud: Permite ver las solicitudes realizadas y que el administrador de los medios marque la opción de si le dio solución o no (solucionado o no solucionado).

CI_GenerarSolicitud: Permite acceder a las solicitudes.

CI_ConfigurarSistema: Permite introducir datos a guardar en la aplicación, para su configuración.

CI_GestionarReporte: Permite ver los reportes realizados y que el administrador de los medios marque la opción de si le dio solución o no (solucionado o no solucionado).

Clases Controladoras:

CC_GenerarReporte: Coordina las actividades de generación de reportes, según el tipo de reporte que se solicite.

CC_RegistrarSolicitud: Coordina las actividades de las solicitudes que se realicen.

CC_RegistrarReporte: Coordina las actividades de los reportes que se realicen.

CC_GestionarSolicitud: Coordina las actividades de ver las solicitudes realizadas y gestionarlas.

CC_GenerarSolicitud: Coordina las actividades de generar solicitudes, según el tipo de solicitud que se solicite.

CC_ConfigurarSistema: Coordina las actividades de configurar el sistema.

CC_GestionarReporte: Coordina las actividades de ver los reportes realizados y gestionarlos.

Clases Entidad:
CE_Reporte: Contiene la información sobre los reportes realizados.
CE_Medios: Contiene la información sobre los medios que existen en cada local.
CE_Solicitud: Contiene la información sobre las solicitudes realizadas.

3.4 Diseño.

Modelo del diseño:

El diseño es un refinamiento que toma en cuenta los requerimientos no funcionales, por lo cual se centra en cómo el sistema cumple sus objetivos, traduce los requerimientos funcionales y no funcionales en una representación del software, constituyendo el primer paso para el desarrollo de cualquier sistema.

Un Modelo de Diseño es una abstracción del Modelo de Implementación y su código fuente, el cual se emplea para representar y documentar su diseño. Es usado como entrada esencial en las actividades relacionadas a la implementación. Representa a los casos de uso en el dominio de la solución. [28]

Para el modelado del diseño se realizó un diagrama de clases por cada caso de uso y un diagrama de secuencia por cada escenario del caso de uso. Se utilizó la arquitectura definida: el patrón Modelo Vista Controlador.

3.5 Diagramas de interacción

Los diagramas de secuencia y los diagramas de colaboración (llamados diagramas de interacción) son dos de los cinco tipos de diagramas UML que se utilizan para modelar los aspectos dinámicos de los sistemas. Un diagrama de interacción muestra una interacción, que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes; un diagrama de colaboración es un diagrama de interacción que destaca la organización estructural de los objetos que envían y reciben mensajes. [29]

Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema. La mayoría de las veces, esto implica modelar instancias concretas o

prototípicas de clases, interfaces, componentes y nodos, junto con los mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento. Los diagramas de interacción no son sólo importantes para modelar los aspectos dinámicos de un sistema, sino también para construir sistemas ejecutables por medio de ingeniería directa e inversa. [30]

Un diagrama de interacción es básicamente una proyección de los elementos de una interacción. La semántica del contexto de una interacción, los objetos y roles, enlaces, mensajes y secuenciación se aplican a los diagramas de interacción. Al igual que los demás diagramas, los diagramas de interacción pueden contener notas y restricciones. [31]

Diagrama de Interacción del diseño Gestionar Solicitud.

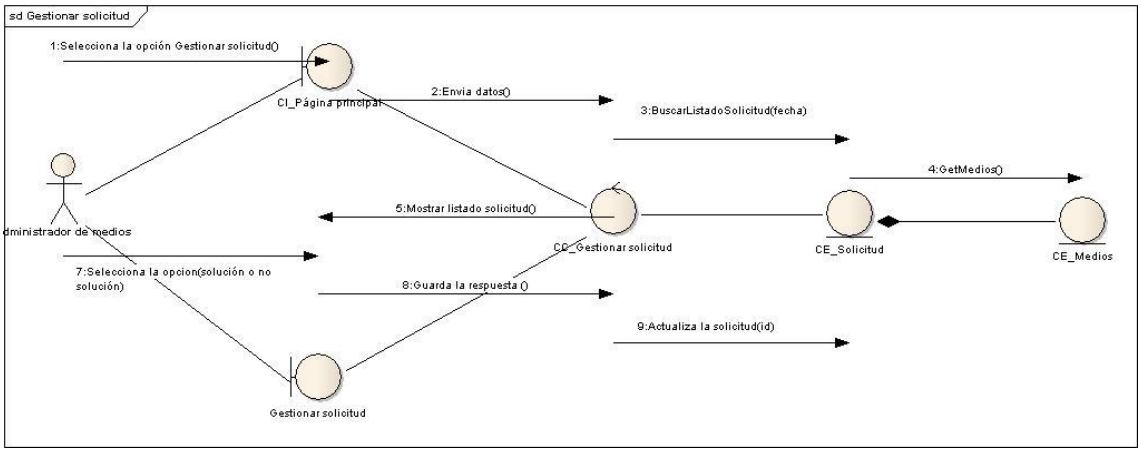


Diagrama de Interacción del diseño Registrar Reporte.

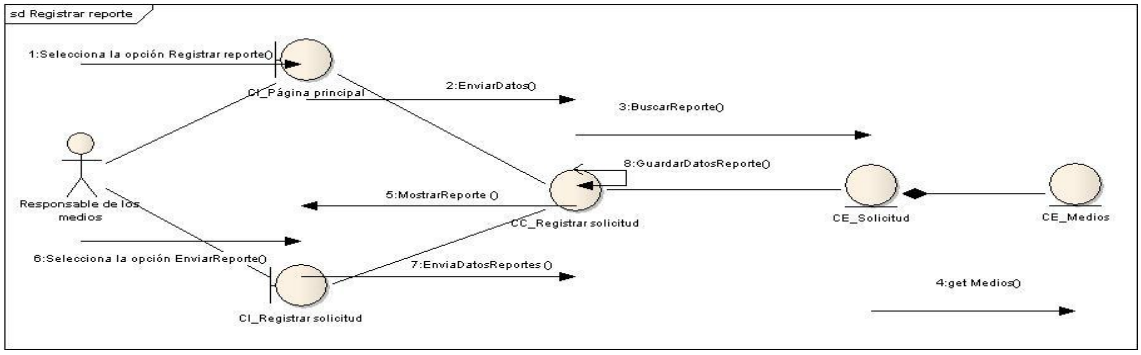


Diagrama de Interacción del diseño Gestionar Reporte.

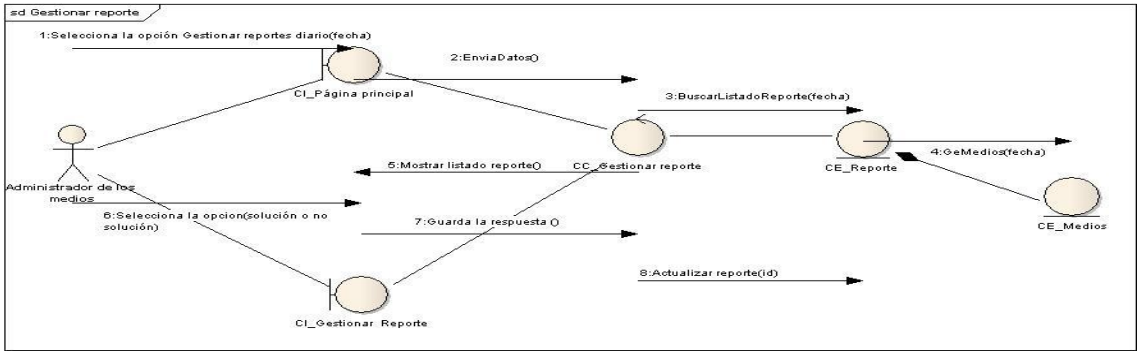
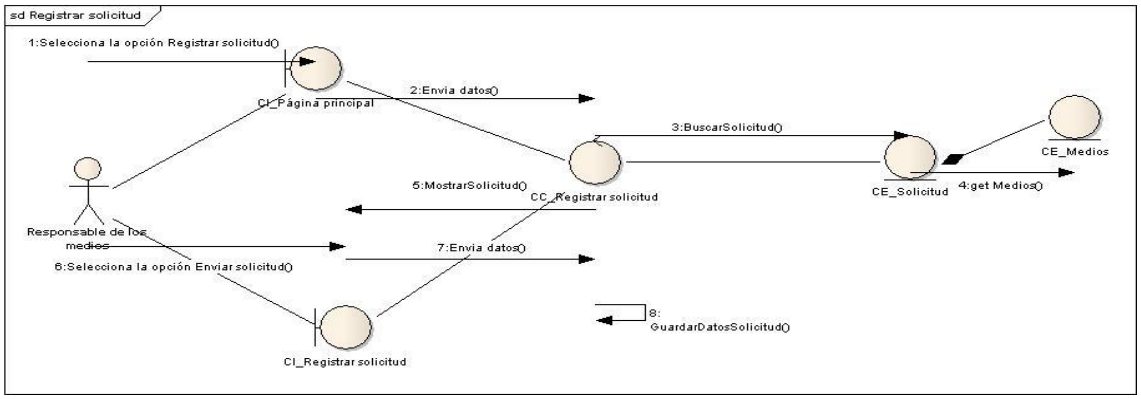


Diagrama de Interacción del diseño Registrar Solicitud.



Modelado mediante estereotipos Web:

Para la realización de los Diagramas de Clases del Diseño se utilizó la extensión de UML. La misma presenta como elementos significativos a tres clases UML: Server Page, Client Page y Form empleadas para el código servidor, código cliente y formularios.

El código servidor se encarga de construir el resultado XHTML que conforma el código cliente (<<build>>). Los formularios envían sus datos al código servidor a través de un (<<submit>>), la relación entre la clase empleada para el código cliente y la clase para el formulario es de agregación. Entre páginas clientes pueden existir vínculos (<<link>>) o redireccionamientos (<<redirect>>). Una página cliente es construida por una sola página servidora. Puede completar su funcionamiento utilizando la relación de inclusión (<<include>>).

Estereotipos	Hasta Desde	Client Page	Form	Server Page
	**Página Cliente (CP)**	<<link>>, <<redirect>>	aggregation	<<link>> <<redirect>>
	**Formulario (Form)**	aggregated by		<<submit>>

	**Página Servidora (SP)**	<<build>>, <<redirect>>		<<redirect>> <<include>>

Diagrama de clases del diseño Gestionar Reporte.

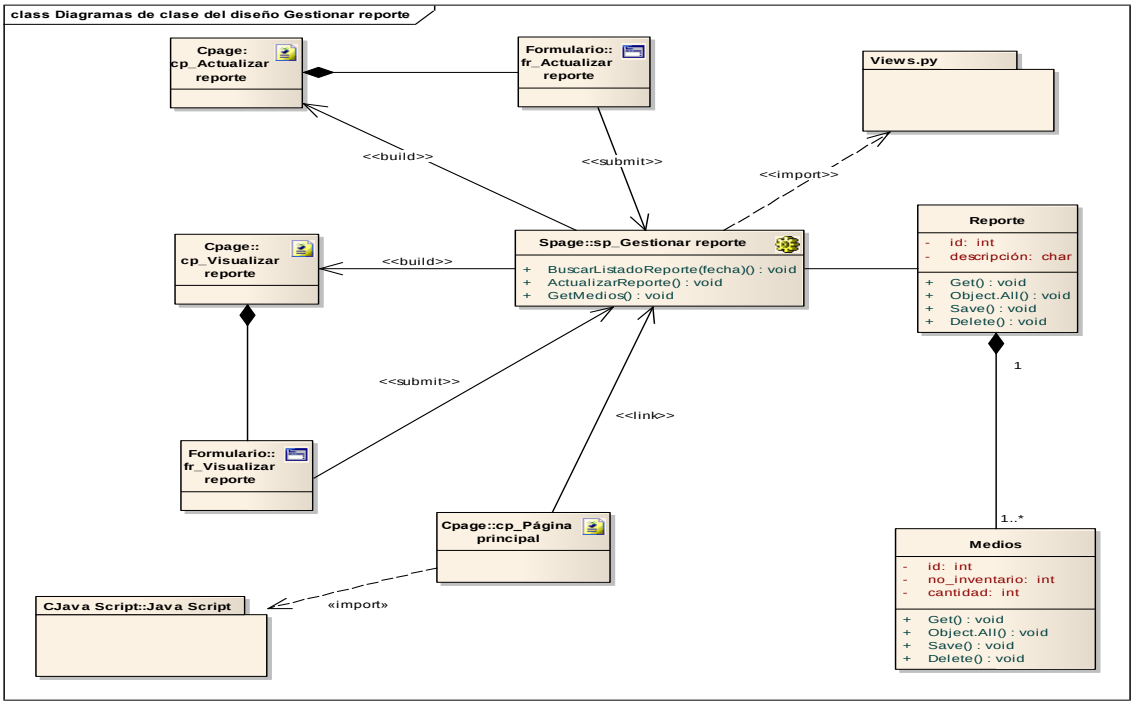


Diagrama de clases del diseño Registrar Reporte.

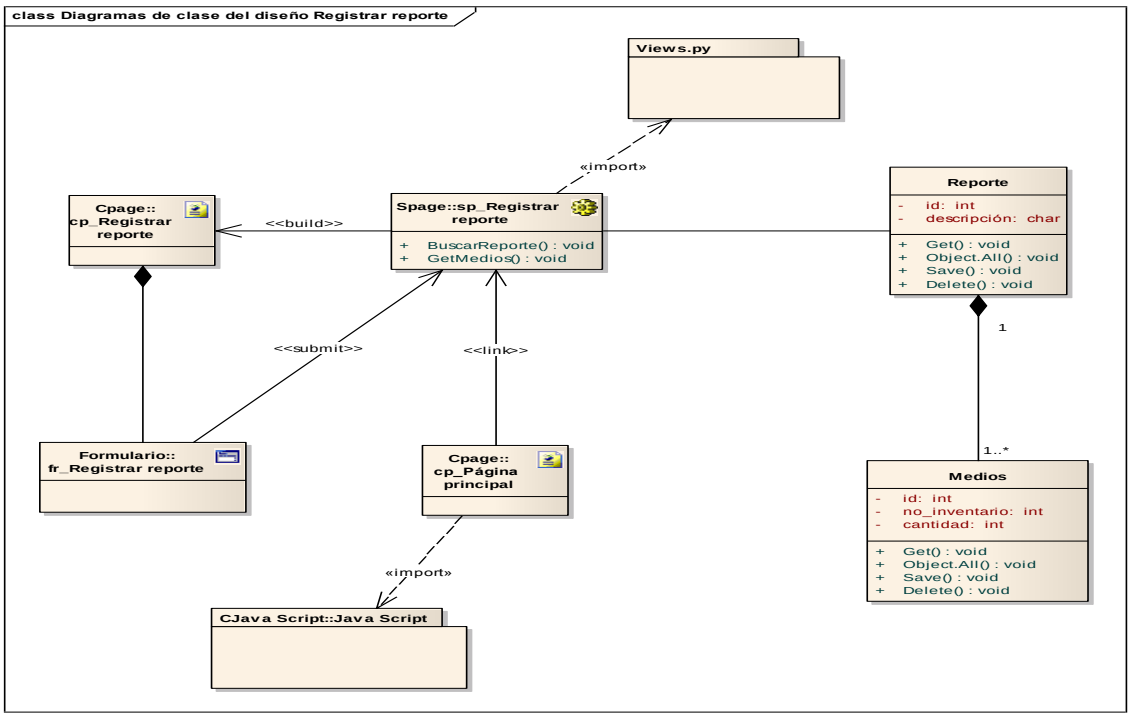


Diagrama de clases del diseño Registrar Solicitud.

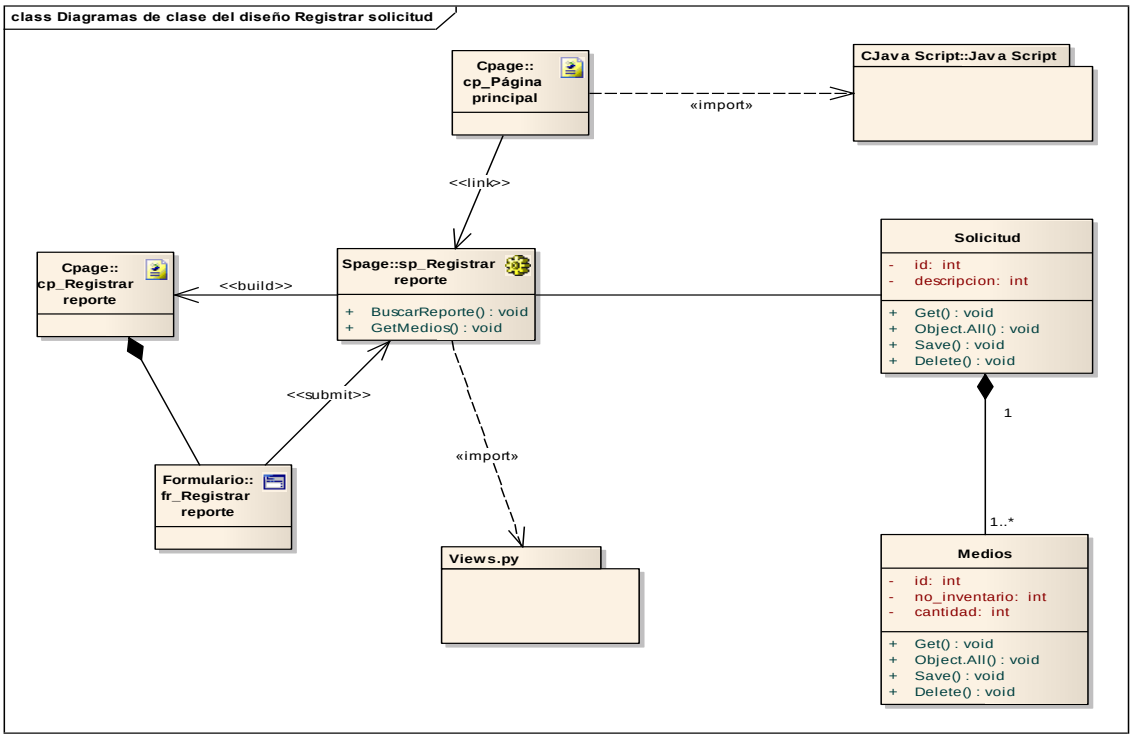
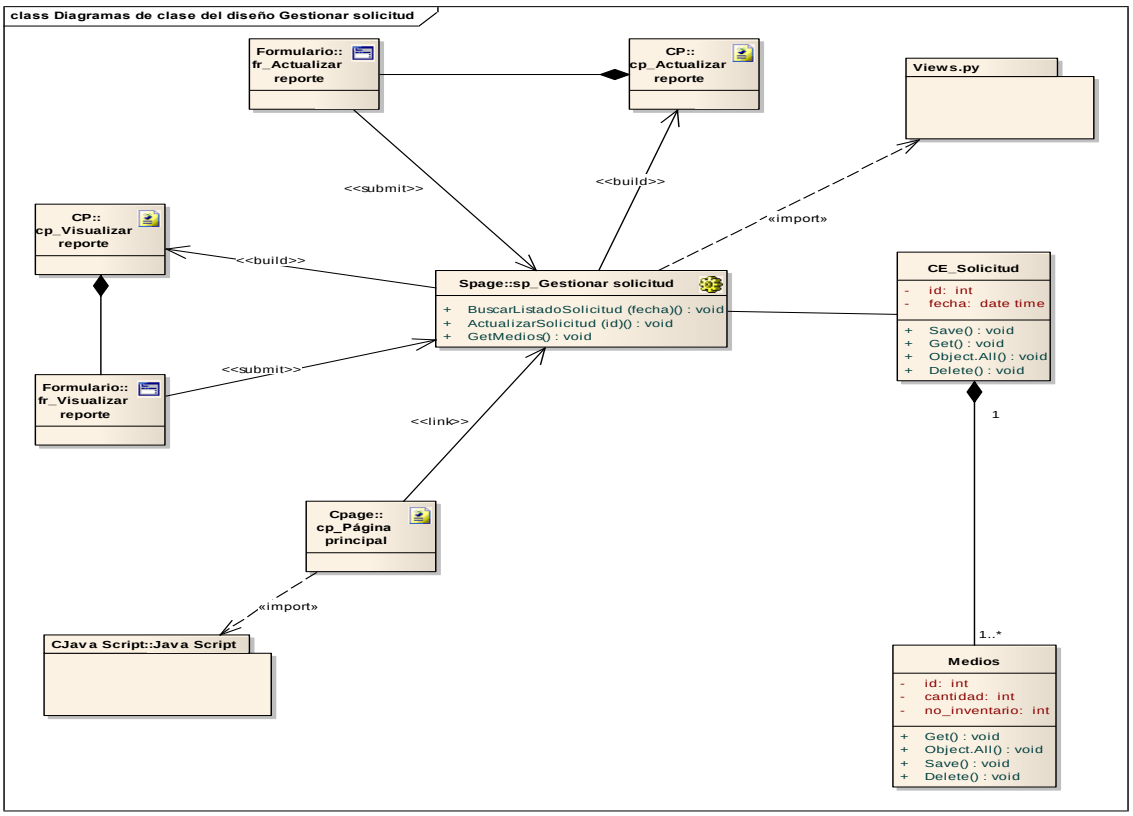


Diagrama de clases del diseño Gestionar Solicitud.



La unión entre capa de presentación y capa de negocio conocido en el paradigma de la programación por capa representaría la integración entre Vista y su correspondiente Controlador de eventos y acceso a datos, MVC no pretende discriminar entre capa de

negocio de capa y presentación pero si pretende separar la capa visual gráfica de su correspondiente programación y acceso a datos, algo que mejora el desarrollo y mantenimiento de la Vista y el Controlador en paralelo, ya que ambos cumplen ciclos de vida muy distintos entre sí. [32]

Los framework para el desarrollo web actuales tienden a implementar el patrón MVC, Django, TurboGears y Pylons. Por lo que insisten en lo importante de entender bien este patrón para poder aprovechar mejor todo lo bueno que brindan estas excelentes herramientas.

En este sistema se utilizará el framework de desarrollo web de código abierto, escrito en Python, que cumple en cierta medida el paradigma del Modelo Vista Controlador. La meta fundamental de Django es facilitar la creación de sitios web complejos. Pone énfasis en el reuso, la conectividad y extensibilidad de componentes, del desarrollo rápido. Python es usado en todas las partes del Framework, incluso en configuraciones, archivos y en los modelos de datos.

También proporciona una aplicación incorporada para administrar los contenidos, que puede incluirse como parte de cualquier página hecha con él mismo y que puede administrar varias páginas hechas con este framework a partir de una misma instalación; la aplicación administrativa permite la creación, actualización y eliminación de objetos de contenido, llevando un registro de todas las acciones realizadas sobre cada uno, y proporciona una interfaz para administrar los usuarios y los grupos de usuarios (incluyendo una asignación detallada de permisos).

Además aparenta implementar el patrón MVC, pero el controlador es llamado vista y la vista template. Este se asemeja mucho a la implementación del patrón MVC, para Django la Vista describe “qué” datos serán presentados y no “cómo” se verán los mismos. Aquí es donde entran en juego los templates, que describen “cómo los datos son presentados”.

Se dice que el “controller” de un MVC clásico está representado por el propio Framework. Es decir, el sistema que envía una petición a la vista correspondiente, de acuerdo con la configuración de URL de Django (archivo de configuración). En el caso de querer hacer una correspondencia, entonces se diría que éste es un Framework “MTV”: modelo, template, vista.

El patrón de diseño Modelo Vista Controlador (MVC) describe una forma, muy utilizada en el Web, de organizar el código de una aplicación separando los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. [33]

Modelo: Componente encargado del acceso a datos.

Vista: Define la interfaz de usuario, HTML+CSS, enviados en el navegador.

Controlador: Responde a eventos y modifica la vista y el modelo.

La principal ventaja de esta separación reside en la facilidad para realizar cambios en la aplicación puesto que:

- Cuando se realiza un cambio de bases de datos, programación o interfaz de usuario solo tocaremos uno de los componentes.
- Se puede modificar uno de los componentes sin conocer como funcionan los otros.

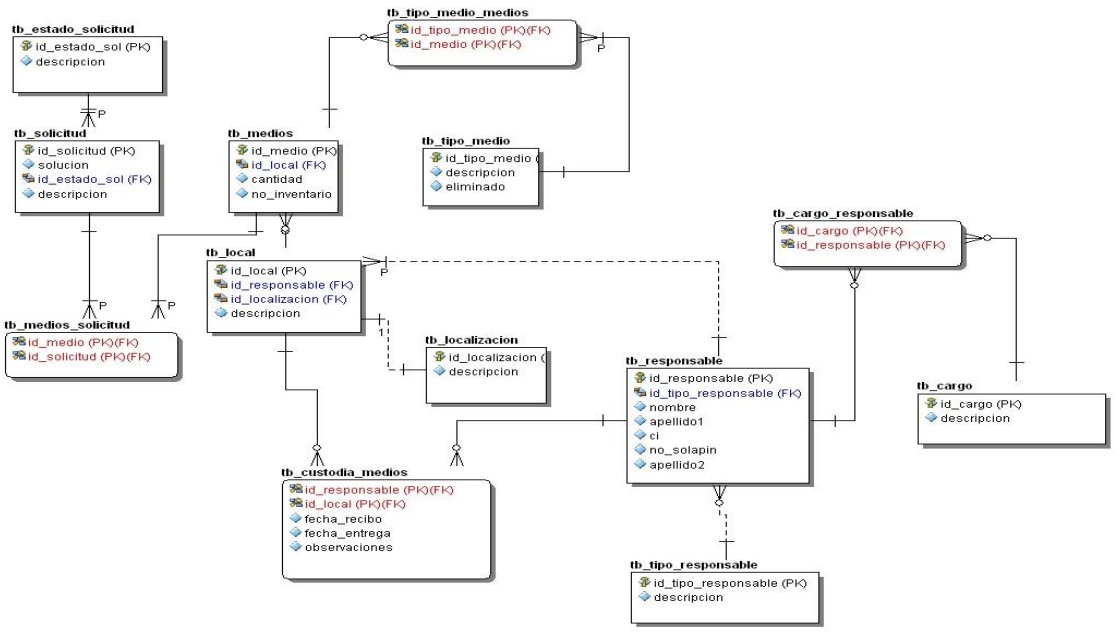
Capítulo 4: Implementación

La etapa de implementación comienza con el resultado de la etapa de Diseño y se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, *scripts*, ficheros de código binario, ejecutables y similares. El objetivo principal de la etapa de implementación es desarrollar la arquitectura y el sistema en conjunto. De forma más específica, los propósitos de la Implementación son: [34]

- Definir la organización del código.
- Planificar las integraciones de sistema necesarias en cada iteración.
- Implementar las clases y subsistemas encontrados durante el Diseño.

En el ciclo de vida del *software* la etapa de Implementación es el centro durante las iteraciones de construcción, aunque también se lleva a cabo el trabajo de implementación durante la fase de elaboración, para crear la línea base ejecutable de la arquitectura, y durante la fase de transición, para tratar defectos tardíos. Ya que el modelo de implementación denota la implementación actual del sistema en términos de componentes y subsistemas de implementación, es natural mantener el modelo de implementación a lo largo de todo el ciclo de vida del *software*. [35]

4.1 Modelo de datos



4.2 Descripción de las tablas.

Nombre: tb_estado_solicitud		
Descripción: Esta tabla contiene los atributos del estado de la solicitud que realiza un responsable de los medios.		
Atributo	**Tipo**	**Descripción**
id_estado_sol	int	Representa el identificador del estado de la solicitud.
descripción	varchar	Representa la clasificación del estado de la solicitud realizada por el responsable de los medios.

Nombre: tb_solicitud		
Descripción: Esta tabla contiene los atributos de la solicitud que realiza un responsable de los medios.		
Atributo	**Tipo**	**Descripción**
id_solicitud	int	Representa el identificador de la solicitud
solución	varchar(500)	Representa la descripción de la solicitud.
id_estado_sol	int	Representa el identificador del estado de la solicitud.
descripción	varchar(500)	Representa la clasificación de la solicitud realizada por el responsable de los medios.

Nombre: tb_medios_solicitud		
Descripción: Esta tabla contiene los atributos identificación de las tablas de medios y solicitud.		
Atributo	**Tipo**	**Descripción**
id_medio	int	Representa el identificador del medio.
id_solicitud	int	Representa el identificador de la solicitud.

Nombre: tb_medios		
Descripción: Esta tabla contiene los atributos del medio.		
Atributo	**Tipo**	**Descripción**
id_medios	int	Representa el identificador del medio.
id_local	int	Representa el identificador del local.

cantidad	int	Representa la cantidad de medios que existe en un local.
no_inventario	varchar(20)	Representa el número de inventario del medio.

Nombre: tb_local		
Descripción: Esta tabla contiene los atributos del local		
Atributo	**Tipo**	**Descripción**
id_local	int	Representa el identificador del local.
id_responsable	int	Representa el identificador del responsable.
id_localizacion	int	Representa el identificador de localización
descripción	varchar(100)	Representa el número de inventario del medio.

Nombre: tb_tipo_medio_medios		
Descripción: Esta tabla contiene los atributos de identificación de las tablas tipo de medio y medio.		
Atributo	**Tipo**	**Descripción**
id_tipo_medio	int	Representa el identificador del tipo de medio.
id_medio	int	Representa el identificador de medios.

Nombre: tb_tipo_medio		
Descripción: Esta tabla contiene los atributos de tipo de medio		
Atributo	**Tipo**	**Descripción**
id_tipo_medio	int	Representa el identificador del tipo de medio.
descripción	char (10)	Representa la descripción del tipo de medio que es.
eliminado	boolean	Representa la descripción de si se encuentra eliminado.

Nombre: tb_localización
Descripción: Esta tabla contiene los atributos de localización.

Atributo	Tipo	Descripción
id_localización	int	Representa el identificador de localización.
descripción	varchar (100)	Representa la descripción de la localización de cada medio.

Nombre: tb_custodia_medios		
Descripción: Esta tabla contiene los atributos de la custodia de los medios.		
Atributo	**Tipo**	**Descripción**
id_responsable	int	Representa el identificador del responsable.
id_local	int	Representa el identificador del local.
fecha_recibo	timestamp	Representa la fecha de recibo del local asignado para la guardia.
fecha_entrega	timestamp	Representa la fecha de entrega del local asignado para la guardia.
observaciones	varchar(1000)	Representa la descripción de los daños ocurridos o no en el horario de la guardia.

Nombre: tb_cargo_responsable		
Descripción: Esta tabla contiene los atributos de identificación de las tablas “tipo de cargo” y responsable.		
Atributo	**Tipo**	**Descripción**
id_cargo	int	Representa el identificador del cargo.
id_responsable	int	Representa el identificador del responsable.

Nombre: tb_responsable		
Descripción: Esta tabla contiene los atributos identificación de las tablas responsable y tipo_responsable.		
Atributo	**Tipo**	**Descripción**
id_responsable	int	Representa el identificador del responsable.
id_tipo_responsable	int	Representa el identificador del tipo de responsable.
nombre	varchar(50)	Representa el nombre del responsable.
apellido1	varchar(50)	Representa el primer apellido del responsable.

ci	char(11)	Representa el carnet de identidad del responsable.
no_solapin	varchar(10)	Representa el número del solapín del responsable.
apellido2	varchar(50)	Representa el segundo apellido del responsable.

Nombre: tb_cargo		
Descripción: Esta tabla contiene los atributos del cargo		
Atributo	**Tipo**	**Descripción**
id_cargo	int	Representa el identificador del cargo.
descripción	varchar (50)	Representa la descripción del cargo de cada responsable.

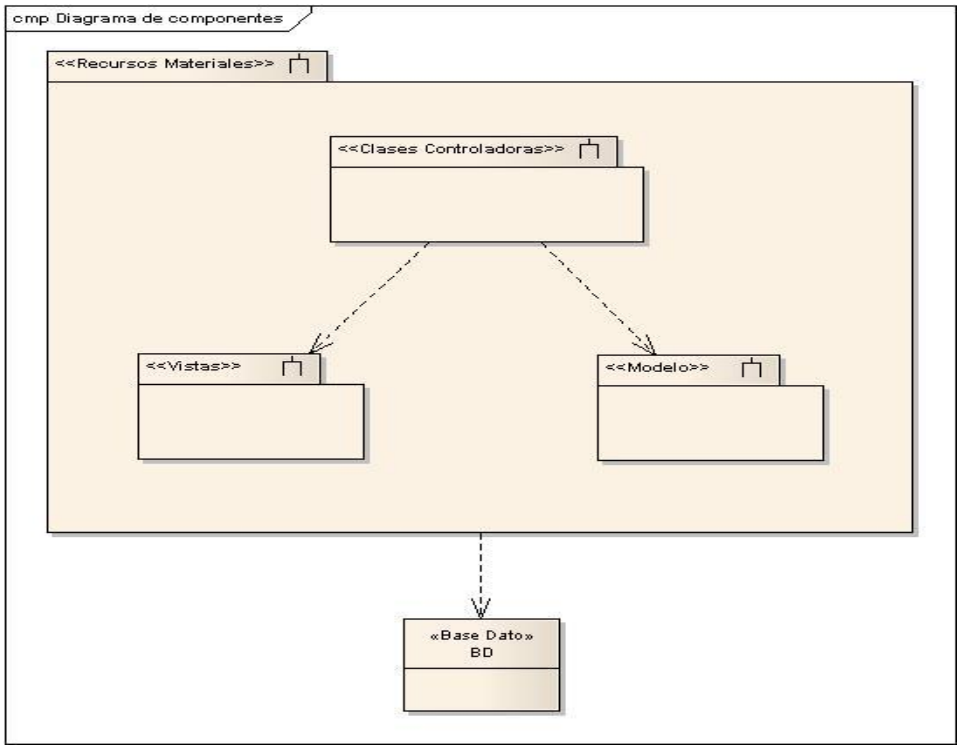
Nombre: tb_tipo_responsable		
Descripción: Esta tabla contiene los atributos del tipo de responsable.		
Atributo	**Tipo**	**Descripción**
id_localización	int	Representa el identificador de localización.
descripción	varchar (50)	Representa la descripción del tipo de responsable.

4.3 Diagramas de componentes.

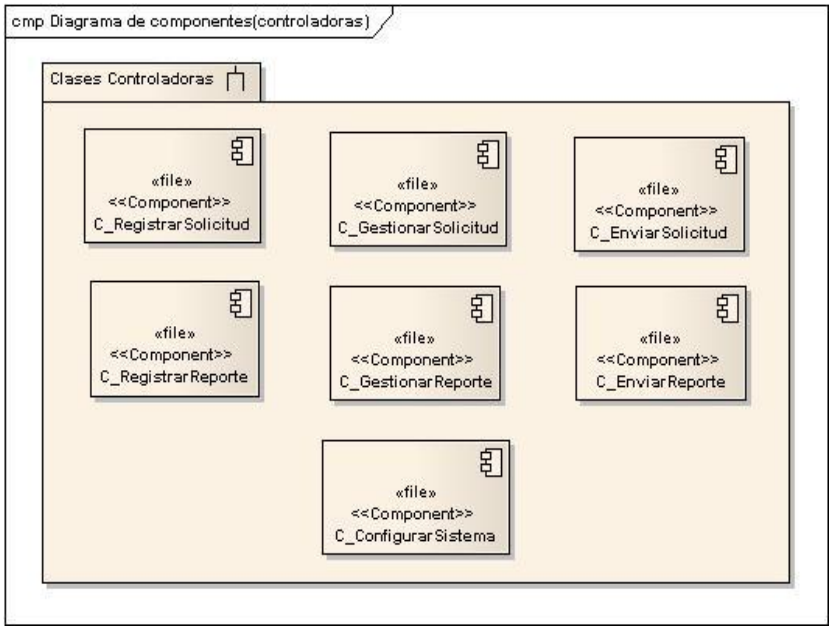
Un Diagrama de Componentes es un esquema o diagrama que muestra las interacciones y relaciones de los componentes de un modelo. Componente a una clase de uso específico, que puede ser implementada desde un entorno de desarrollo, ya sea de código binario, fuente o ejecutable; dichos componentes poseen tipo, que indican si pueden ser útiles en tiempo de compilación, enlace y ejecución. [36]

Dentro del diagrama de componentes los elementos de modelado serán componentes y paquetes. En cuanto a los componentes, solo aparecen tipos de componentes, ya que las instancias específicas de cada tipo se encuentran en el diagrama de despliegue. Un diagrama de Componentes tiene un nivel más alto de abstracción que cualquier otro diagrama de clases, un componente se implementa por una o más clases en tiempo de ejecución.

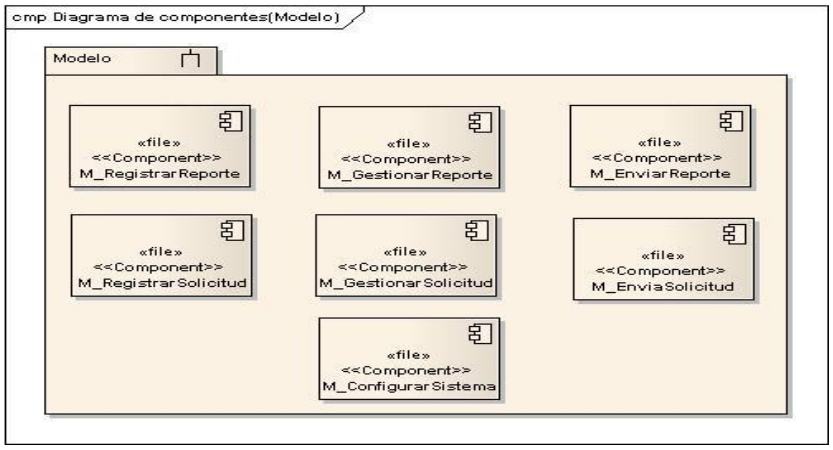
Los componentes se utilizan para modelar la vista estática de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente, se realizan por partes. Cada diagrama describe un apartado del sistema.



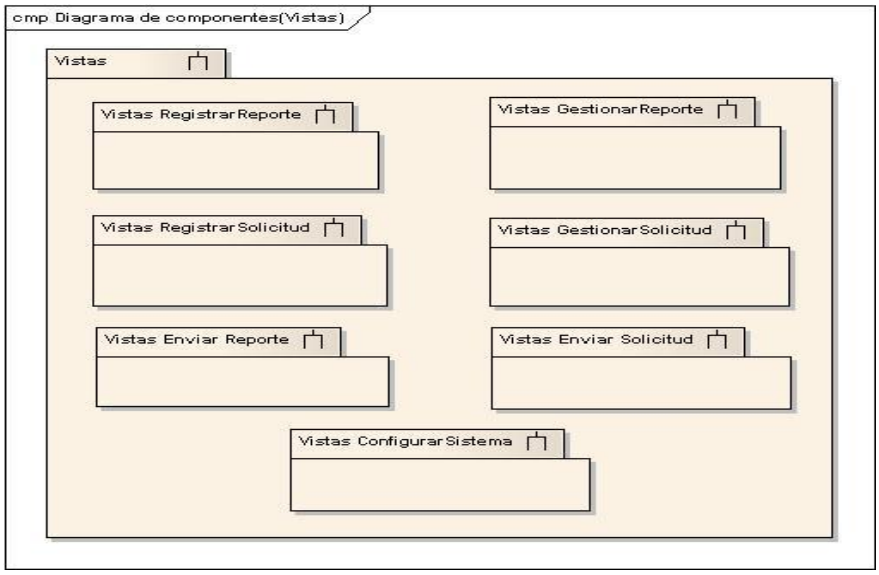
Subsistema Controladoras.



Subsistema modelo.

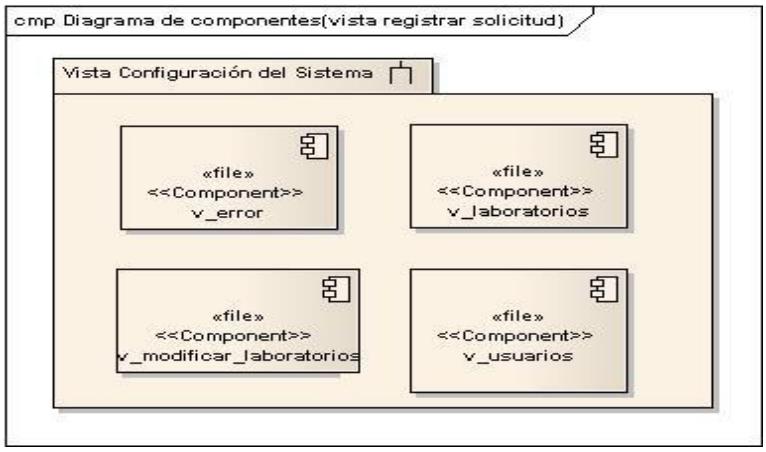


Subsistema Vistas.

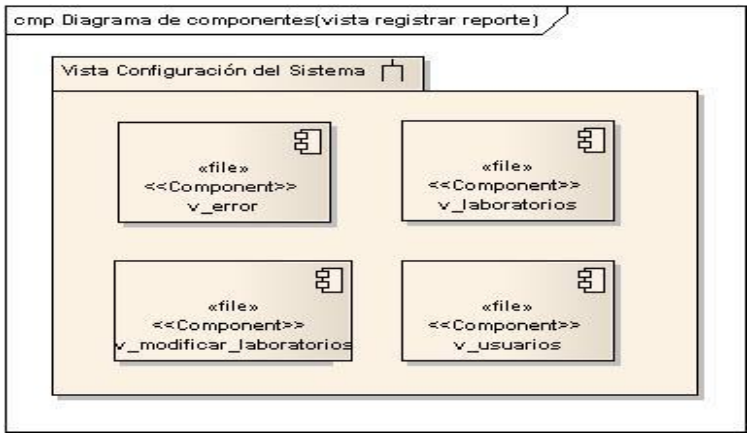


A continuación se describen detalladamente cada uno de estos subsistemas que componen las vistas del *software*.

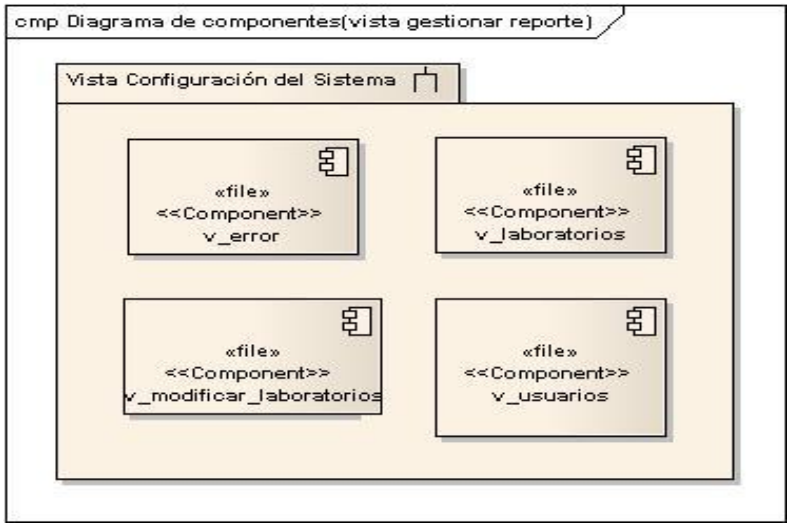
Vista registrar solicitud.



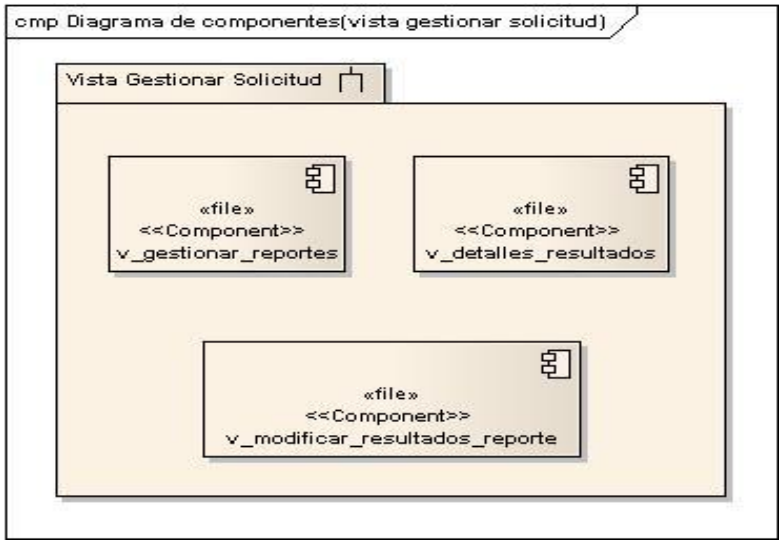
Vista registrar reporte.



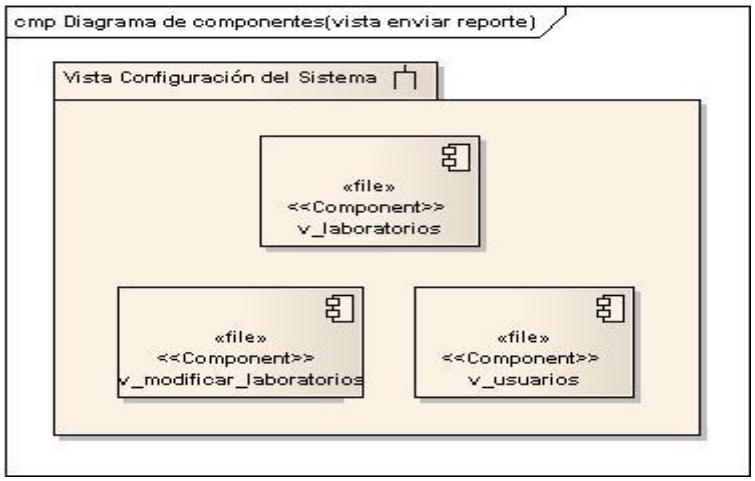
Vista gestionar reporte.



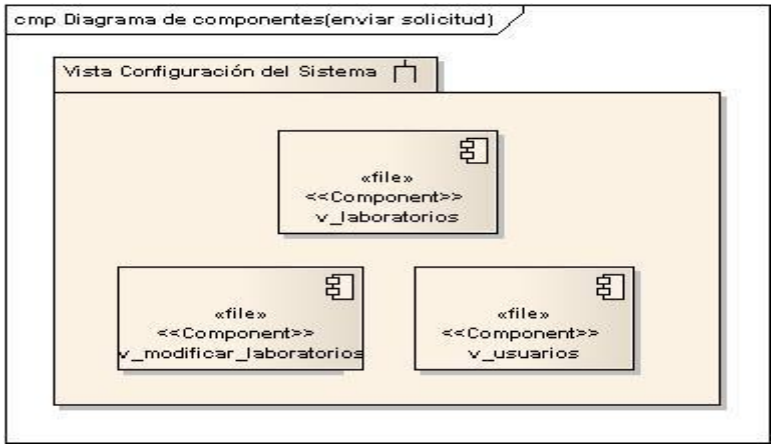
Vista gestionar solicitud.



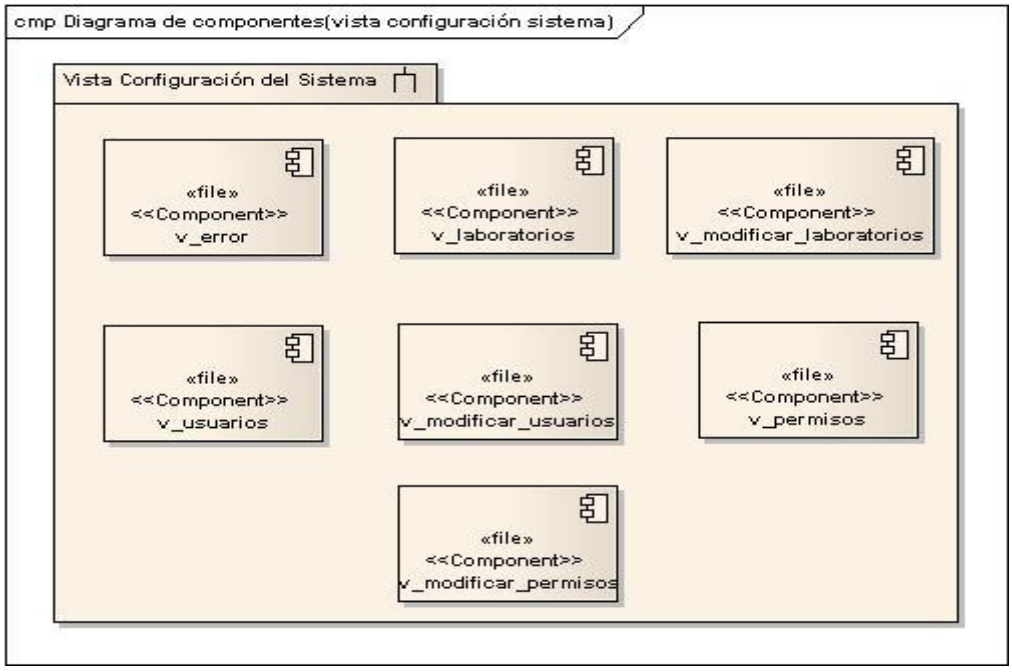
Vista enviar reporte.



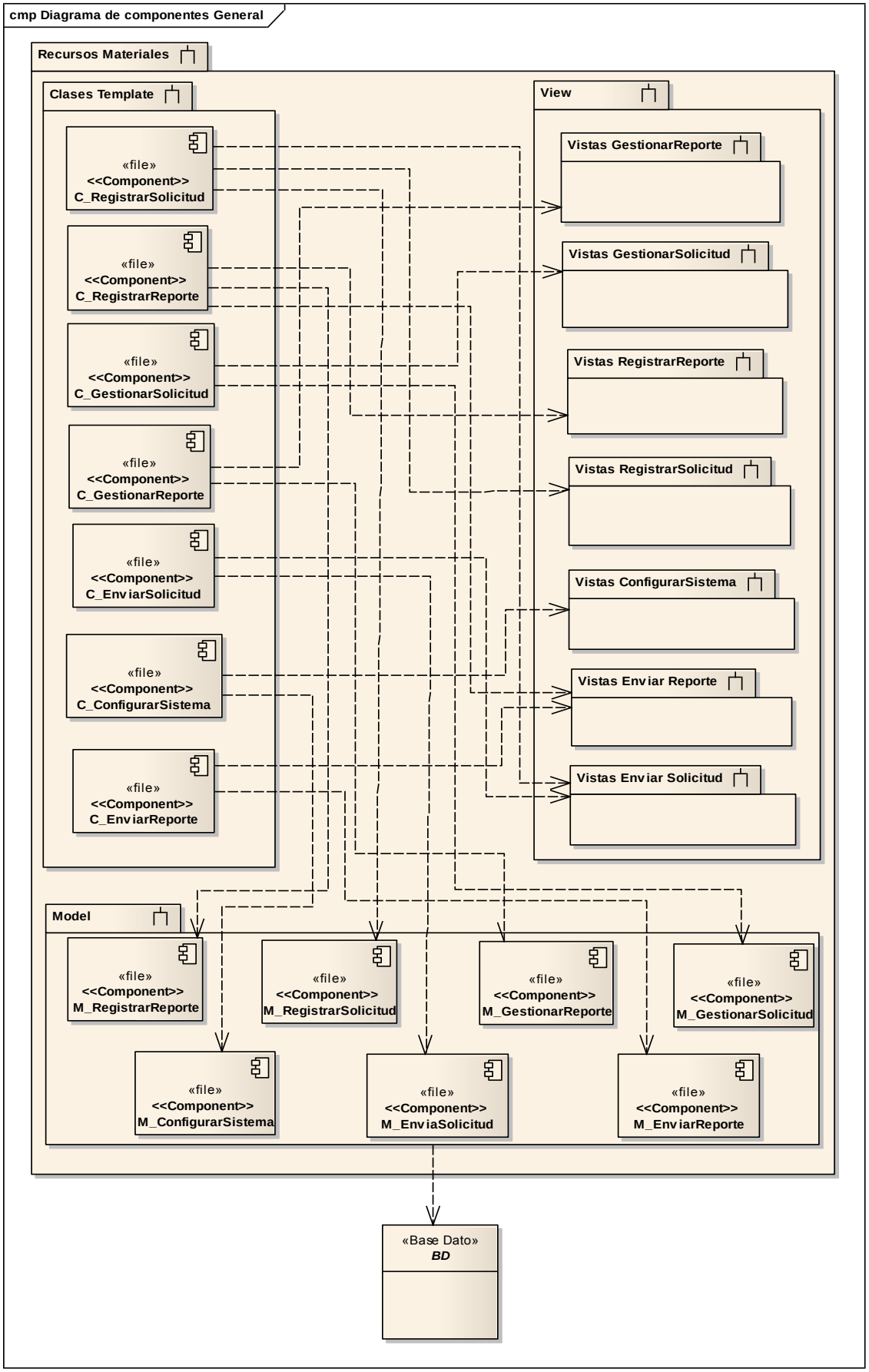
Vista enviar solicitud.



Vista configuración sistema.



Relación entre los componentes

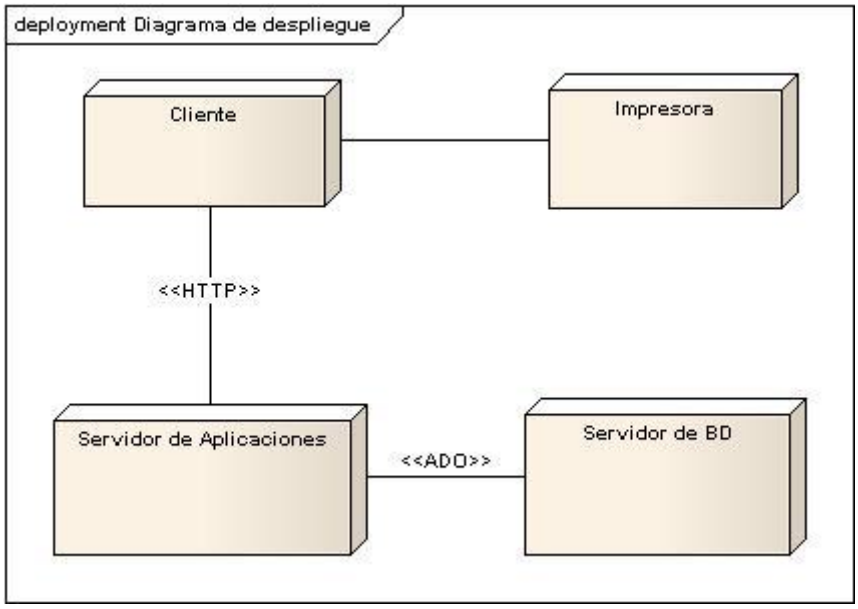


4.4 Diagramas de despliegue.

Un diagrama de Despliegue muestra cómo y dónde se desplegará el sistema. Las máquinas físicas y los procesadores se representan como nodos, y la construcción interna puede ser representada por nodos o artefactos embebidos. Como los artefactos se ubican en los nodos para modelar el despliegue del sistema, la ubicación es guiada por el uso de las especificaciones de despliegue. [37]

Los Diagramas de Despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución, en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria.

Un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación. Un nodo puede contener instancias de componentes. En general un nodo será una unidad de computación de algún tipo. Las instancias de componentes *software* pueden estar unidas por relaciones de dependencia, posiblemente a interfaces.



4.5 Tratamiento de errores.

Todos los lenguajes de programación modernos de alto nivel, y más. Los antiguos, se han incorporado en las construcciones y los métodos para el manejo de errores o excepciones. Podría significar que el programa se bloquea, o podría significar que en

el programa se van a producir resultados sin sentido. Una vez que se comience a usar las redes, servidores de texto, bases de datos, o incluso archivos construidos por cualquier persona, está abierta a errores. Y las posibilidades de convertirse en interminable. Si se desea, se puede esperar hasta entonces si se producen errores en las soluciones de parche. [38]

Un error de información y procesamiento a través de excepciones es una de las principales características de Python. Un programador del mismo puede lanzar una excepción en cualquier parte de un programa.

Básicamente hay tres tipos de errores diferentes: [39]

- Sistemáticos: Estos errores vienen, como su nombre indica, por sistema. Puede que empleemos una regla mal graduada en que cada centímetro mida en realidad 13 milímetros, o puede que nos hayamos olvidado de sumar el diámetro de una bola a la hora de marcar la distancia entre la bola y un punto. Estos errores hay que intentar evitarlos y en caso de cometerlos, darse cuenta a tiempo. Generalmente el valor verdadero de la magnitud a medir no se encuentra en la región de los datos tomados.
- Estadísticos: Estos errores vienen dados por motivos muy diversos. En este caso el valor verdadero de la magnitud a medir está en la región de los datos tomados.
- Incertidumbres: Estos errores son causados por la precisión del aparato que empleamos para medir, que puede ser menor o igual a las fluctuaciones estadísticas de la medida.

4.6 Seguridad.

Garantizar que los recursos informáticos de una compañía estén disponibles para cumplir sus propósitos, es decir, que no estén dañados o alterados por circunstancias o factores externos, es una definición útil para conocer lo que implica el concepto de seguridad informática. En términos generales, la seguridad puede entenderse como aquellas reglas técnicas destinadas a prevenir, proteger y resguardar lo que es considerado como robo, pérdida o daño, ya sea de manera personal o grupal. En este sentido, es la información el elemento principal de proteger, resguardar y recuperar dentro de las redes empresariales. [40]

De manera general la seguridad de los sistemas informáticos se concentra en garantizar el derecho a acceder a datos y recursos del sistema configurando los mecanismos de autentificación y control que aseguran que los usuarios de estos recursos sólo posean los derechos que se les han otorgado. Los mecanismos de seguridad pueden causar inconvenientes a los usuarios.

Debido a todo esto es que la seguridad informática debe estudiarse de modo que no evite que los usuarios hagan uso de todo lo que necesiten y así puedan utilizar los sistemas de información en forma segura. Para una buena seguridad del *software* se puso en práctica un mecanismo de seguridad que sería la autenticación y autorización, para de esta manera evitar que los usuarios no autorizados accedan a los datos que se manejan en el sistema.

4.6.1 Autenticación.

La autenticación consiste en la confirmación de la identidad de un usuario; es decir, la garantía para cada una de las partes de que su interlocutor es realmente quien dice ser. Un control de acceso permite (por ejemplo gracias a una contraseña codificada) garantizar el acceso a recursos únicamente a las personas autorizadas. [41]

Con la autenticación lo que se pretende no es que se pueda leer el mensaje por personas ajenas, sino que se pueda asegurar la procedencia del mismo. Todo esto no es nuevo ya que lo hemos estado utilizando desde hace mucho tiempo.

En mensajería usual (empresas como SEUR, WRW, correos) se pone la firma en cada envío. En las transferencias bancarias es necesaria la firma de la persona que tiene la cuenta bancaria. Y por supuesto, en materia de nuevas tecnologías también se puede ver como ejemplo muy cercano de la actualidad el acceso a sistemas Unix, en los que cada cuenta de usuario necesita un login y una contraseña.

4.6.2 Autorización.

El control de acceso muchas veces llamado autorización, es el cómo una aplicación Web permite el acceso ha contenido y funciones a algunos usuarios y a otros no. Estas verificaciones son desarrolladas después de la autenticación, y gobiernan lo que pueden hacer los usuarios autorizados. El control de acceso suena como un problema simple pero es difícil de implementar correctamente.

El modelo de control de acceso de una aplicación Web tiene un lazo estrecho con el contenido y funciones que el sitio provee. Además, los usuarios pueden caer dentro de un número de grupos o roles con diferentes capacidades o privilegios. Los desarrolladores frecuentemente menosprecian la dificultad de implementar un mecanismo de control de acceso confiable.

Conclusiones

Un sistema de control de recursos es una herramienta poderosa en manos de la administración. Sobre todo cuando se utiliza para preservar y controlar bienes que han sido costosos y de difícil adquisición, como es el caso de la Universidad de las Ciencias Informáticas.

Como lenguaje de programación se definió Python como base del desarrollo, como framework Django, unidos al gestor de bases de datos PostgreSQL, estos constituyen herramientas potentes, eficaces y seguras para el desarrollo de aplicaciones web. Así mismo, las herramientas y tecnologías seleccionadas para el desarrollo del sistema están bajo licencia de *software* libre, lo cual aporta independencia tecnológica a la solución, lo cual permite la implementan buenas prácticas y contienen recursos y herramientas que facilitan y organizan el buen desarrollo del *software*.

La realización del estudio del estado del arte permitió establecer la línea base de las tendencias actuales para estos tipos de sistemas informáticos, con lo cual se logró además tener una mejor comprensión de los objetivos de implementación.

El sistema fue implementado siguiendo buenas prácticas que hoy han cobrado auge en los sistemas de gestión sobre la web. El patrón “Modelo Vista Controlador” constituye un paradigma organizacional y funcional del código fuente y las acciones del sistema con grandes ventajas para el *software*. Es importante señalar que la obtención de los diagramas y artefactos establecidos por la metodología RUP, contribuyen a la comprensión y sobre todo a la mantenibilidad de la aplicación en el tiempo.

Con la puesta en práctica de este sistema, la facultad 7 de la UCI contará con un control más exacto del estado y la cantidad de recursos materiales con que cuenta. Esto garantizará a su vez la mejora en la toma de decisiones a la hora de adquirir o reparar estos medios.

Recomendaciones

Se recomienda:

- Cambiar el diseño actual por uno que sea más atractivo para el usuario.
- Extender las funcionalidades del sistema en la residencia de estudiantes, implementando funcionalidades como: gestionar reportes de la residencia de estudiantes de la facultad.

Referencias bibliográficas

[1]. Universidad de San Carlos de Guatemala. [En línea] [Citado el: 9 de febrero de 2010.] http://biblioteca.usac.edu.gt/tesis/08/08_5871.pdf

[2] Ídem a la referencia 1

[3]. Centro de Cálculo Provincial de Salud [En línea] [Citado el: 9 de febrero de 2010.] http://bvs.sld.cu/revistas/san/vol7_1_03/san54103.pdf [4]. Universidad de Murcia. [En línea] [Citado el: 9 de febrero de 2010.] http://www.um.es/f-bellasartes/sgic/pa/pa_bbaa_06_grm.pdf

[4]. Tutorial de Python. [En línea] [Citado el: 9 de febrero de 2010.] http://lateral.netmanagers.com.ar/static/tutorial-5.pdf.

[5].Ídem a la referencia 4

[6] Ídem a la referencia 4

[7] Ídem a la referencia 4

[8] Ídem a la referencia 4

[9]. Tutorial de Django. [En línea] [Citado el: 9 de febrero de 2010.] http://davidasorey.net/static/django-tutorial/

[10] PostgreSQL. [En línea] [Citado el: 9 de febrero de 2010.] http://www.guia-ubuntu.org/index.php?title=PostgreSQL.

[11] Ídem a la referencia 10

[12] Ídem a la referencia 10

[13] Enterprise Arquitect. [En línea] [Citado el: 9 de febrero de 2010.] http://www.sparxsystems.com.ar/products/ea.html

[14] Ídem a la referencia 13

[15] Ídem a la referencia 13

[16] Ídem a la referencia 13

[17] Ídem a la referencia 13

[18] Ídem a la referencia 13

[19] Leguaje Unificado de Modelado. [En línea] [Citado el: 9 de febrero de 2010.] http://elvex.ugr.es/decsai/java/pdf/3E-UML.pdf

[20] Ídem a la referencia 19

[21] Ídem a la referencia 19

[22] Ídem a la referencia 19

[23] RUP. [En línea] [Citado el: 9 de febrero de 2010.] www.slideshare.net/dersteppenwolf/la-ingeniera-de-software-y-rup.

[24] Ídem a la referencia 23

[25] Ídem a la referencia 23
[26] Requisitos Funcionales y no Funcionales. [En línea] [Citado el: 30 de febrero de 2010.] http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/
[27] Ídem a la referencia 26
[28] Ministerio del Poder Popular para las Telecomunicaciones y la Informática. MeRinde. Modelo de Diseño [En línea] [Citado el: 30 de febrero de 2010.] http://merinde.rinde.gob.ve/index.php?option=com_content&task=view&id=92&Itemid=296
[29] Ingeniería de Software II.¨Continuación del FT Análisis de Diseño.¨ Conferencia # 1.s.l.: UCI, 2009 - 2010.
[30] Ídem a la referencia 29
[31] Ídem a la referencia 29
[32] Patrón MVC. [En línea] [Citado el: 30 de febrero de 2010.] http://librosweb.es/symfony_1_0/capitulo2/el_patron_mvc.html
[33] Patrón MVC. [En línea] [Citado el: 30 de febrero de 2010.] http://www.programacionweb.net/articulos/articulo/?num=505
[34] Etapa /Implementación. [En línea] [Citado el: 20 de abril de 2010.] http://lsi.ugr.es/~arroyo/inndoc/doc/implementacion/implementacion_d.php
[35] Ídem a la referencia 38
[36] Componente. [En línea] [Citado el: 20 de abril de 2010.] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.
[37] Despliegue. [En línea] [Citado el: 20 de abril de 2010.] virtual.usalesiana.edu.bo/web/practica/archiv/despliegue.doc
[38] Tratamiento de Errores. [En línea] [Citado el: 20 de abril de 2010.] http://www.wdvl.com/Authoring/python/error_handling/phillip_watts07232009.html.
[39] Tipos de Errores. [En línea] [Citado el: 20 de abril de 2010.] http://www.lawebdefisica.com/apuntsfis/errores/.
[40]. Seguridad Informática. [En línea] [Citado el: 20 de abril de 2010.] http://www.inegi.gob.mx/inegi/contenidos/espanol/ciberhabitat/museo/cerquita/redes/seguridad/intro.htm
[41] Autenticación. [En línea] [Citado el: 20 de abril de 2010.] http://html.rincondelvago.com/redes-y-autenticacion-de-firmas-digitales.html

Bibliografía

1. Universidad de Alcalá. [En línea] [Citado el: 9 de febrero de 2010.] http://www.uah.es/empresariales/facultad/documentos/sistema_calidad/procesos_apoyo/PA-04.pdf.

2. Universidad de Murcia. [En línea] [Citado el: 9 de febrero de 2010.] http://www.um.es/f-bellasartes/sgic/pa/pa_bbaa_06_grm.pdf.

3. Universidad de Jaén. [En línea] [Citado el: 9 de febrero de 2010.] http://www.ujaen.es/centros/facsoc/SGIC/PROC.%20DE%20APOYO/PA06grm.pdf.

4. Área de Enfermería.[En línea] [Citado el: 9 de febrero de 2010.] HYPERLINK http://www.madrid.org/cs/Satellite?cid=1142436193437&language=es&pagename=HospitalRamonCajal%2FPage%2FHRYC_contenidoFinal

5. Tutorial de Python. [En línea] [Citado el: 9 de febrero de 2010.] http://lateral.netmanagers.com.ar/static/tutorial-5.pdf.

6. Tutorial de Django. [En línea] [Citado el: 9 de febrero de 2010.] http://davidasorey.net/static/django-tutorial/

7. PostgreSQL. [En línea] [Citado el: 9 de febrero de 2010.] http://www.guia-ubuntu.org/index.php?title=PostgreSQL.

8. Enterprise Arquitect. [En línea] [Citado el: 9 de febrero de 2010.] http://www.sparxsystems.com.ar/products/ea.html

9. Leguaje Unificado de Modelado. [En línea] [Citado el: 9 de febrero de 2010.] http://elvex.ugr.es/decsai/java/pdf/3E-UML.pdf

10. RUP. [En línea] [Citado el: 9 de febrero de 2010.] www.slideshare.net/dersteppenwolf/la-ingeniera-de-software-y-rup

11. Requisitos Funcionales y no Funcionales. [En línea] [Citado el: 30 de febrero de 2010.] http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/

12. Patrón MVC. [En línea] [Citado el: 30 de febrero de 2010.] http://www.programacionweb.net/articulos/articulo/?num=505

13. Ministerio del Poder Popular para las Telecomunicaciones y la Informática. MeRinde. Modelo de Diseño [En línea] [Citado el: 30 de febrero de 2010.]http://merinde.rinde.gob.ve/index.php?option=com_content&task=view&id=92&Itemid=296

14. Ingeniería de *Software* II.¨Continuación del FT Análisis de Diseño.¨ Conferencia # 1.s.l.: UCI, 2009 - 2010.

15. http://lsi.ugr.es/~arroyo/inndoc/doc/implementacion/implementacion_d.php

16. http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf

17. http://virtual.usalesiana.edu.bo/web/practica/archiv/despliegue.doc
18. http://www.wdvl.com/Authoring/python/error_handling/phillip_watts07232009.html.
19. http://www.lawebdefisica.com/apuntsfis/errores/.
20.http://www.inegi.gob.mx/inegi/contenidos/espanol/ciberhabitat/museo/cerquita/redes/seguridad/intro.htm.
21. http://html.rincondelvago.com/redes-y-autenticacion-de-firmas-digitales.html

Anexos

Anexo 1. Diagramas de clases del análisis.

Diagrama de clases del análisis Generar Reporte.

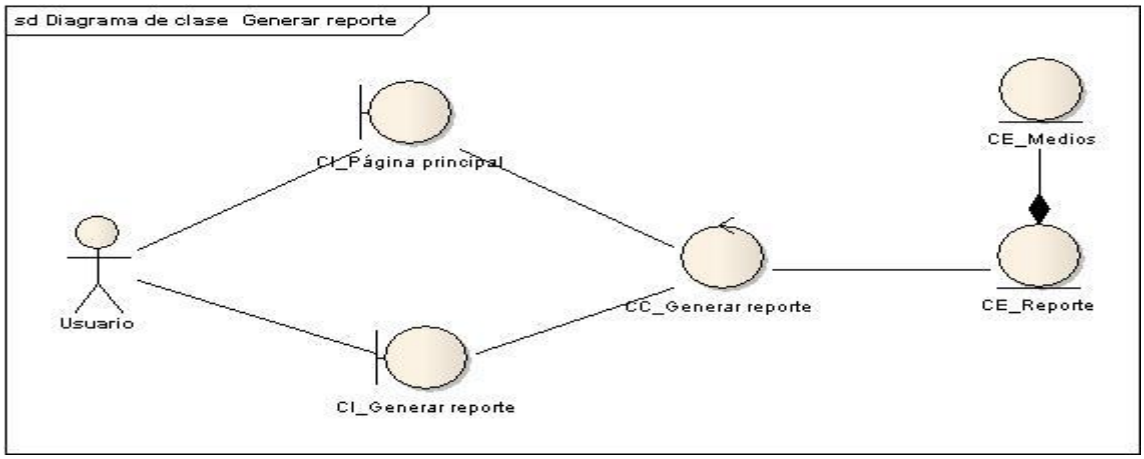


Diagrama de clases del análisis Generar Solicitud.

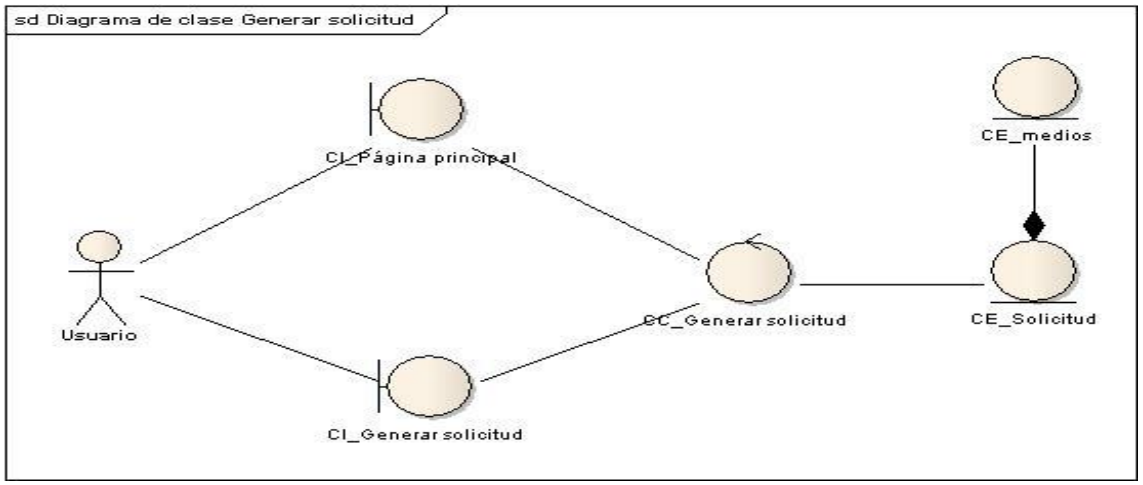
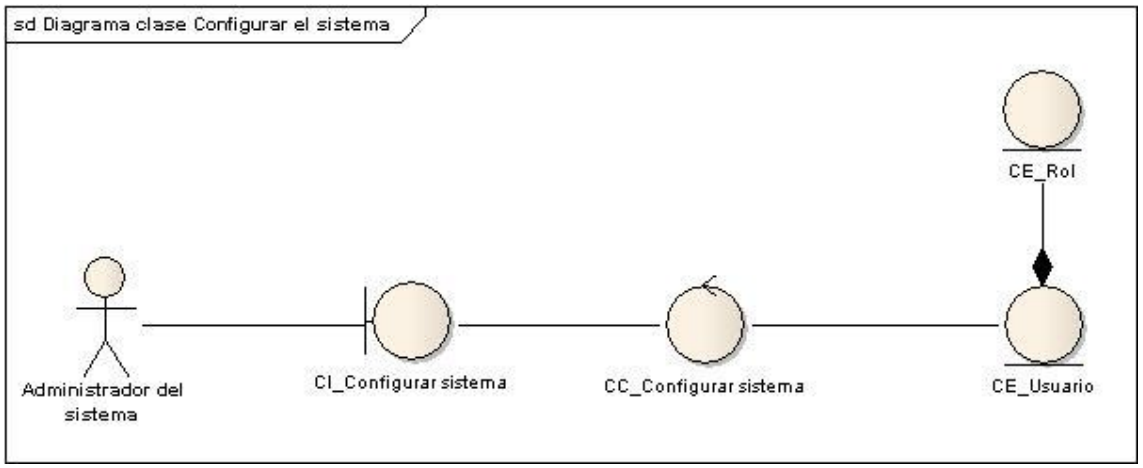


Diagrama de clases del análisis Configurar Sistema.



Anexo 2. Diagramas de Interacción del diseño.

Diagrama de Interacción del diseño Generar Solicitud.

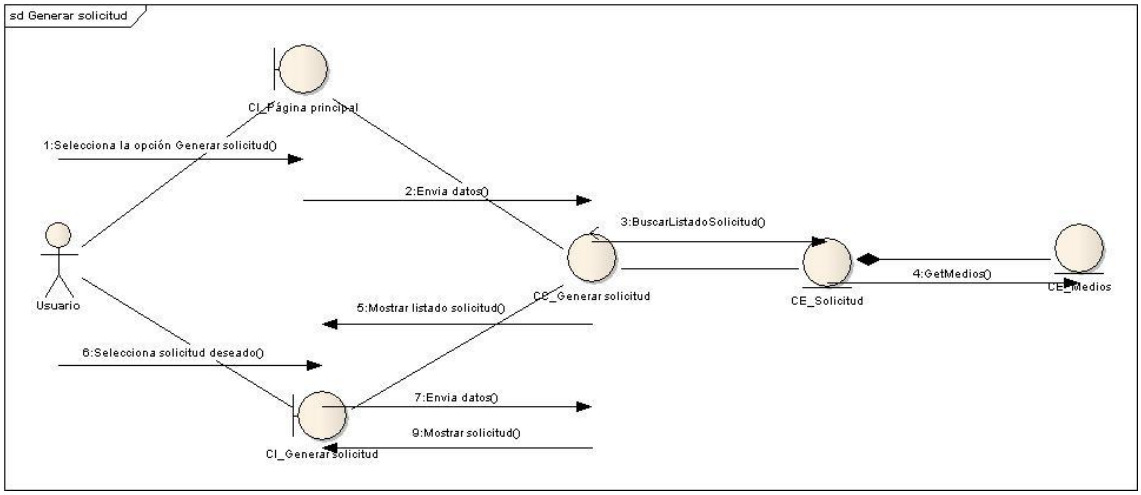


Diagrama de Interacción del diseño Generar Reporte.

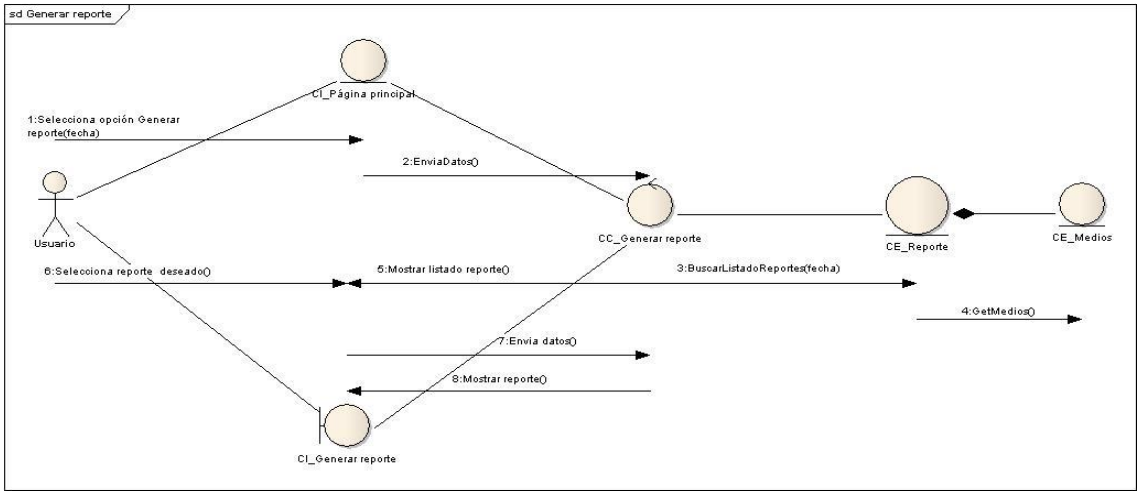


Diagrama de Interacción del diseño Configurar Sistema.

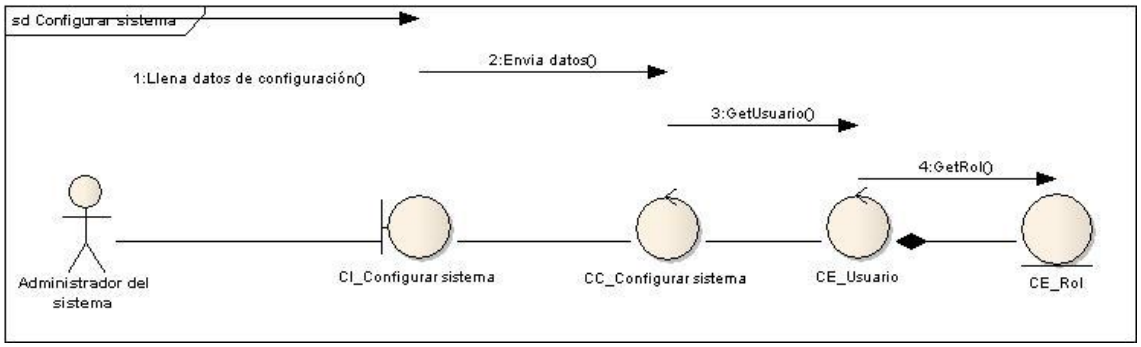


Diagrama de clases del diseño Configurar Sistema.

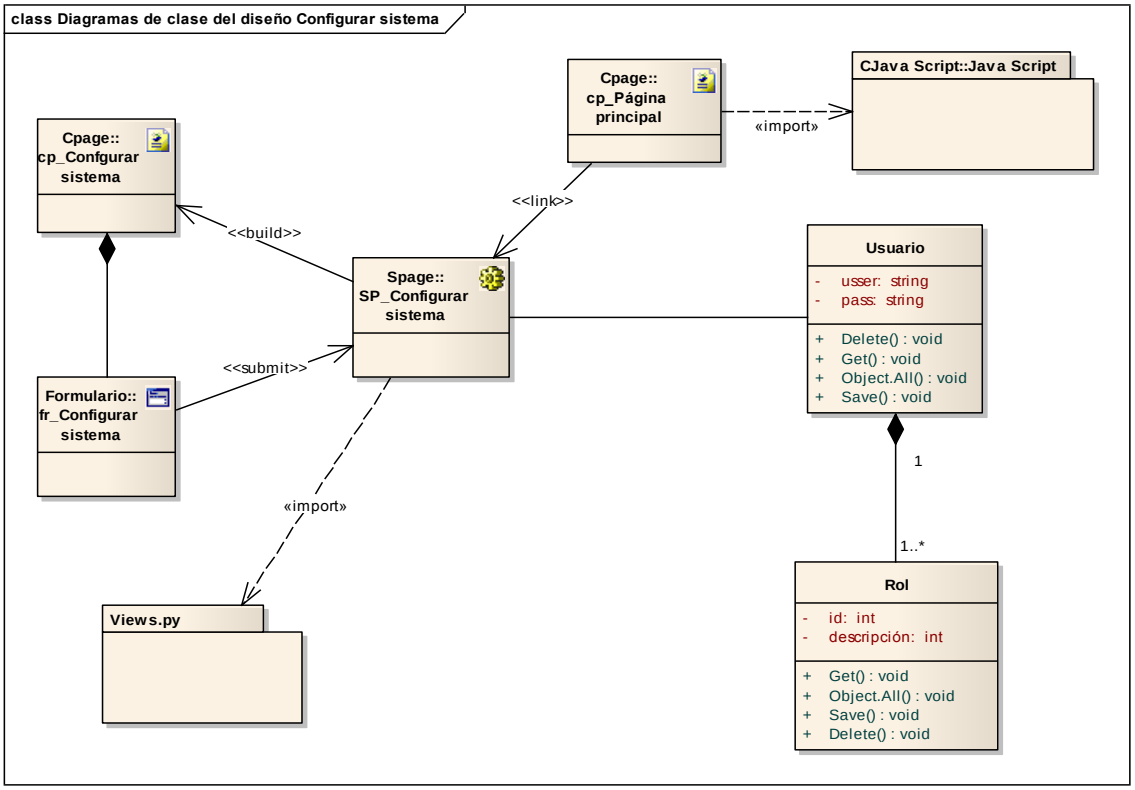


Diagrama de clases del diseño Generar Solicitud.

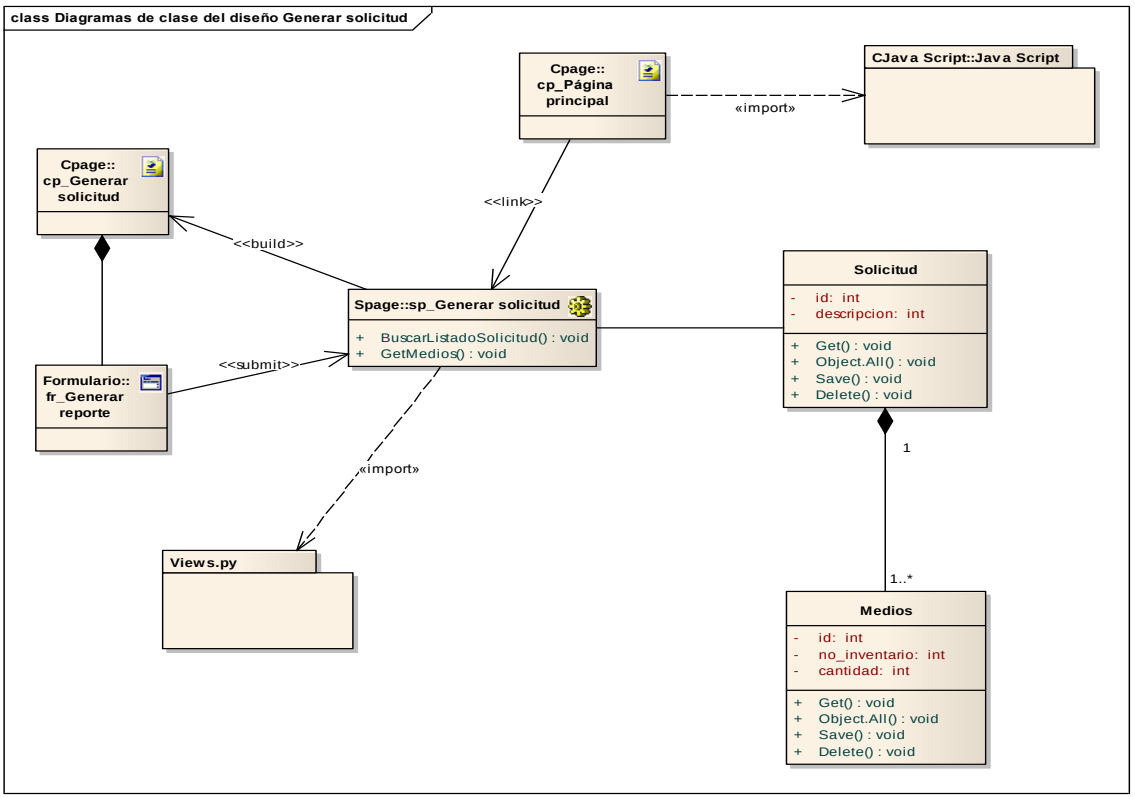
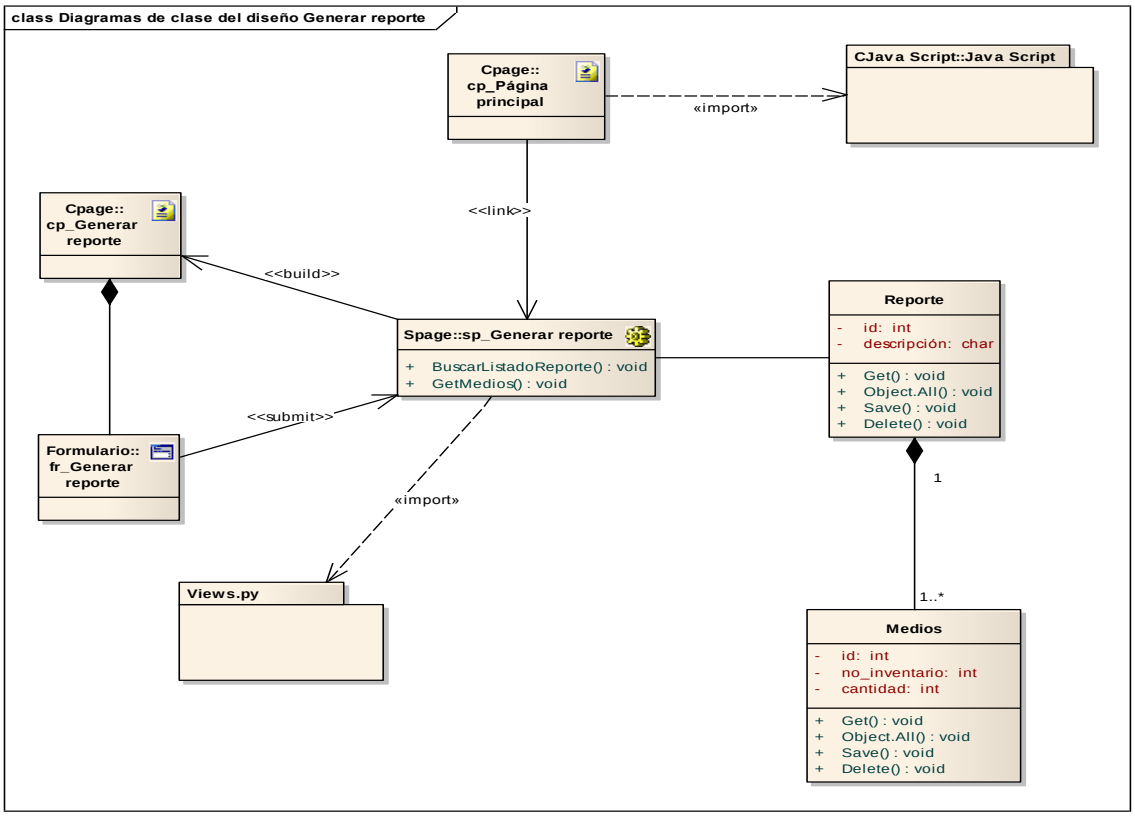


Diagrama de clases del diseño Generar Reporte.



Glosario de términos

Actor del Negocio: Cualquier individuo, grupo, organización externa; con los que el negocio interactúa.

C#: Es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA e ISO.

C: Lenguaje de programación creado en 1972 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL.

C++: Lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C. Se puede decir que C++ es un lenguaje que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos.

Caso de Uso: Fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores.

Diagrama de Caso de Uso del Negocio: Representa gráficamente a los procesos del negocio y su interacción con los actores del negocio.

Diagrama de Actividades: Contiene estados en que puede hallarse 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.

Framework: Es una estructura de soporte definida en la cual un proyecto de *software* puede ser organizado y desarrollado.

Gestión de recursos materiales: La gestión de los recursos materiales es una definición práctica, es una acumulación de bienes bajo ciertos controles y propósitos. La gestión de recursos materiales como noción, nace junto con la noción de propiedad privada, y puede remontarse a las primeras sociedades humanas, en donde podemos encontrar la noción del almacenamiento y acumulación de bienes como alimentos, granos, animales y subproductos.

Modelo de Objetos: Muestra la participación de los trabajadores y entidades del negocio y la relación entre ellos.

Recursos materiales: Los recursos materiales son los bienes tangibles que la organización puede utilizar para el logro de sus objetivos. En los recursos materiales podemos encontrar los siguientes elementos: maquinarias, inmuebles, insumos, productos terminados, elementos de oficina, instrumentos y herramientas.

Trabajador del Negocio: Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo. Ellos realizan las actividades y son propietarios de elementos.

UCI: Universidad de las Ciencias Informáticas.

Windows: Sistema Operativo de Microsoft.