Universidad de las Ciencias Informáticas

Facultad 7

Sistema para la Gestión y Análisis de Información Estadística en la salud pública cubana: Subsistema Servidor de Datos

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autores: Jorge Rodríguez del Toro

Alejandro Montes de Oca Samoano

Tutores: Ing. Norge Martínez Almaguer

Lic. Yasel Couce Sardiñas

Ciudad de La Habana, Junio de 2009

"Año del 50 aniversario del triunfo de la Revolución"

Declaración de Autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos al Grupo de Desarrollo del Área Temática de Sistemas de Apoyo a la Salud de la Facultad 7 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Jorge Del Toro Rodríguez

_____

Alejandro Montes de Oca Somoano

_____

Ing. Norge Martínez Almaguer

_____

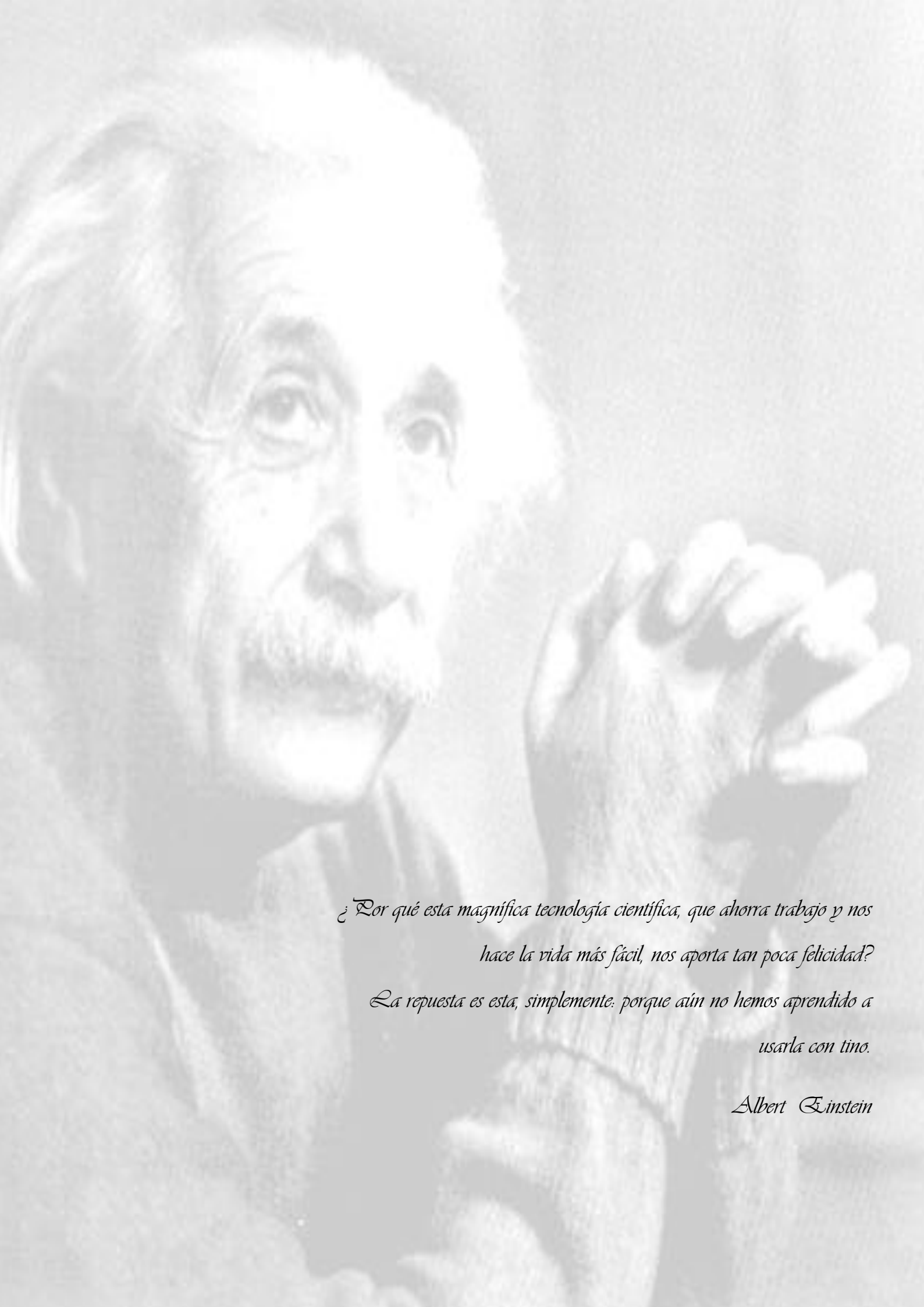
Lic. Yasel Couce Sardiñas

_____

Datos de Contacto

Ing. Norge Martínez Almaguer (nmartinez@uci.cu). Graduado de Ingeniero en Ciencias Informáticas en el año 2007 en la Universidad de las Ciencias Informáticas de Cuba. Imparte la asignatura Práctica Profesional 1, en la Facultad 7 de la propia Universidad, con la categoría docente Profesor Instructor. Actualmente se desempeña como Jefe del Área Temática, Sistemas de Apoyo a la Salud.

Lic. Yasel Couce Sardiñas (yaselc@uci.cu): Graduado de Licenciado en Ciencias de la Computación en el año 2005 en la Universidad Central de Las Villas. Posee Categoría Docente de Profesor Asistente. Ha impartido las asignaturas de Sistema Operativo y Seguridad Informática en la Facultad 7 durante los cursos 2005 – 2006, 2006 – 2007 y 2007 – 2008. Actualmente se desempeña como profesor de práctica profesional.



¿ Por qué esta magnífica tecnología científica, que ahorra trabajo y nos hace la vida más fácil, nos aporta tan poca felicidad?

La repuesta es esta, simplemente: porque aún no hemos aprendido a usarla con tino.

Albert Einstein

De Alejandro:

A mami y papi, por su apoyo incondicional y por ser los mejores padres que la vida me pudo dar.

A mi hermana Rocío por estar presente en mi vida y ser mi mayor alegría.

A mi novia Aylin, por hacer de cada día una nueva ilusión.

A mi abuela Silvia por sembrar en mí su amor para toda la vida.

A mi abuela Olga por su amor y preocupación en mi formación como profesional.

A mis tías Tere, Zoraida y Ángela por ser tías y abuelas.

A mi tía Silvita porque aun estando lejos no ha dejado de preocuparse por mi futuro.

Al primo Pedrito por haber estado siempre que lo necesité.

A mí querida tía nana por haberme acogido en su casa como a un hijo.

A mí tío Oscar por hacer de los momentos dificiles, conversaciones placenteras que quedaron en mí para toda la vida.

A mí suegra Anita por su apoyo y preocupación en mi realización como profesional y por haberme dado la posibilidad de conocer el amor.

Al piquete del cuarto, por haber hecho de los días en la universidad una estancia maravillosa que nunca olvidaré.

A mi familia en general, por ser la mejor familia que una persona puede soñar.

De Jorge:

A Dios por haberme permitido existir y poder realizar este sueño.

A todas aquellas personas que durante la realización de este trabajo me han apoyado de una forma u otra cuando lo he necesitado.

En especial a mis padres por darme fuerzas para seguir adelante y porque son los que me han convertido en lo que hoy soy.

A toda mi familia por sus ánimos y esperanzas.

A esa gran persona que ha influido en mi vida desde que nací y no pudo ver la realización de este sueño, a mi abuela Vina.

A todas esas personas que me han visto y hecho crecer.

A mis compañeros de estudio, a mis amigos, que han compartido conmigo toda esta etapa, a mis hermanos de cinco años.

A la profe Idelsis por haber dado su apoyo y habernos ayudado tanto en cada paso de nuestra carrera.

A todos,

Muchas Gracias.

De Alejandro:

A mi madre por haber hecho suya cada alegría o tristeza y cada logro o fracaso en mi vida como estudiante.

A mi hermana, para inspirarla a alcanzar sus metas.

A mi abuela Olga, que la vida no le dio la oportunidad de estar conmigo este día, donde quiera que estés llegue a ti este triunfo.

A mi abuela Silvia por haberme enseñado tantas cosas en tan poco tiempo.

A mi novia por su paciencia, abnegación y por haber esperado por mi estos años.

Todas estas personas mencionadas anteriormente han influido sobre manera en mi formación como profesional, pero quisiera dedicar este trabajo (mami no te pongas celosa) a una persona que en los últimos meses a vivido momentos muy difíciles y aun así no ha dejado de ejercer por un segundo su función como padre y amigo y ha sabido encontrar la forma de alejarnos de preocupaciones tanto a mi hermana como a mi .Esa persona con sus regaños y gritos me ha logrado convertir en lo que soy hoy, un hombre de bien. Es por eso y por muchas otras cosas que no me alcanzarían las hojas de este documento para plasmarlas que yo dedico el presente

Trabajo de Diploma

A mi Padre.

.

De Jorge:

A mis padres por haber esperado tanto este momento y haberme convertido en el hombre que soy.

A mi hermana que espero siga siendo grande y sienta la satisfacción de llegar hasta donde yo he llegado.

A mis tíos, a mis primos, a la gente del barrio, a Riverí.

A Aldo y Estrella, a Aldito, a Alida, a Osniel, a Orlando y a todos los que con su fuerza, su persona y sus consejos han hecho que me mantenga firme.

A los que me enseñaron mucho y los que también aprendieron de mí.

A todos aquellos que conozco y que no me es posible mencionarlos por que no alcanzaría un libro.

A mis enemigos también por que sin ellos este momento no hubiera sido posible.

A todas las personas que amo en esta vida.

Resumen

El presente Trabajo de Diploma tiene como objetivo la creación de un software que gestione el almacenamiento de la información del Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana. Para el desarrollo de la solución software propuesta se analizaron las tendencias actuales en Cuba y el mundo para el desarrollo de aplicaciones de este tipo y al mismo tiempo se tienen en cuenta experiencias anteriores las cuales sirvieron como punto de partida para la realización de este sistema.

Se utilizó como herramienta CASE (Computer-Aided Software Engineering), Enterprise Architect versión 7.0, herramienta basada en UML, que permite crear los diagramas que se obtienen como parte de la documentación del sistema. Como lenguaje de programación se utilizo Csharp y ambiente de desarrollo integrado se utilizo Visual Estudio 2008 y SharpDevelop y para la gestión de la base de datos el EMS PostgreSQL Manager 2007 para PostgreSQL.

A partir del uso de esta aplicación se facilitará el almacenamiento de la información estadística en dependencia de las necesidades de cada una de las instituciones de salud. Evita la dependencia total de la red, ya que existen servicios que permiten guardar la información estadística en dispositivos de almacenamiento externo, para luego ser guardados en el servidor; logrando así la portabilidad de la misma (información estadística) de un lugar a otro.

Índice

Introducción

A medida que han pasado los años, la Revolución Cubana ha sido partícipe de la ola de avances tecnológicos y científicos que han revolucionado al mundo (para bien en su gran medida) en las últimas décadas. Como Revolución al fin, esta no ha estado ausente a estos cambios y se ha dado a la tarea de informatizar la sociedad, con vista a integrarse plenamente a la infraestructura global de la información haciendo uso de las Tecnologías de la Información y las Comunicaciones (TIC). La informatización de la sociedad se define en Cuba como el proceso de utilización ordenada y masiva de las TIC para satisfacer las necesidades de información y conocimiento de todas las personas y esferas de la sociedad. [1]

Uno de los sectores en que se trabaja es el Ministerio de Salud Pública (MINSAP). Esta prioridad tiene como objetivo prestar mejores servicios y aumentar con ello el bienestar de la sociedad cubana. Como parte de esta informatización se están desarrollando múltiples proyectos donde intervienen además del MINSAP, el Ministerio de la Informática y las Comunicaciones (MIC), la facultad 7 de la Universidad de las Ciencias Informáticas y otros organismos de la Administración Central.

Durante los últimos años han sido varios los proyectos encaminados a desarrollar la informatización de los servicios de la salud en el país, pero a pesar de los esfuerzos no se han logrado alcanzar los objetivos, producto a la carencia de recursos, inapropiada definición de proyectos y falta de integración entre las organizaciones y entidades inmersas en esta importante tarea, además de que las personas involucradas en su desarrollo no han contado con la tecnología necesaria para su ejecución.

Las nuevas soluciones informáticas que se desarrollan permitirán ir alcanzando por etapas la informatización de la Salud Pública Cubana, al contar con la integración de los datos generados por cada paciente a los distintos niveles de atención sanitaria. Estas soluciones junto al personal requerido en las Unidades de Salud (US) permitirán perfeccionar la calidad asistencial ofrecida a la sociedad, facilitar las funciones del personal de la salud y colaborar con la gestión administrativa, asistencial, docente y de investigación.

Las estadísticas de la salud se definen como la información numérica, cuantificable que se utiliza para conocer el estado de salud de la población con la finalidad de planificar, evaluar y controlar programas y acciones que realiza el Sistema Nacional de Salud (SNS), es que se hace necesario mantener un dominio de la Información Estadística de la Salud en el país. [2]

En la actualidad el SNS cuenta con el Sistema de Información Estadística Complementario (SIE-C Salud). El mismo es capaz de generar grandes volúmenes de datos desde un nivel a otro a través de una estructura piramidal de departamentos de estadísticas que van desde la Unidad de Salud hasta la Dirección Nacional de Estadísticas del SNS. El SIE-C Salud está constituido por más de 130 subsistemas de información, en los que se recoge la información estadística del sector en las Unidades de Salud y se consolida en los niveles superiores.

Actualmente el almacenamiento de todos sus datos se hace de forma manual en muchas de las unidades, y en otras con sistemas no estandarizados o de tecnologías propietarias. Esto provoca lentitud en la entrega de la información que se requiere, y en el control exacto de los datos estadísticos que se necesitan periódicamente en el país, que en ocasiones son inexactos. Además es más susceptible a perder información por fenómenos naturales, que el material se deteriore o se cometan errores de cálculos en los procesos que se llevan a cabo en estas instituciones.

De los 130 subsistemas del SIE-C Salud existen 5 que han sido implementados en la Facultad 7 de la UCI, pero no han sido explotados aún en el Sistema Nacional de Salud. Esto se debe en gran medida a la alta dependencia de conectividad que poseen, propiciado por el consumo de información desde sistemas externos como el Registro de Unidades de Salud, el Registro del Ciudadano, el Registro de Ubicación y el SAAA. Este último de gran importancia para el acceso de los usuarios a los subsistemas antes mencionados. Además en el SIE-C Salud surgen frecuentemente nuevos subsistemas o modificaciones a los existentes, lo que trae consigo y aparejado con la arquitectura seguida, que sea necesario iniciar nuevos ciclos de desarrollo y como consecuencia demoras en satisfacer las necesidades del Sistema Nacional de Salud.

Por todo lo anteriormente expresado se plantea que el **problema a resolver** radica en como garantizar el almacenamiento de los modelos de flujo y la información captada por el Sistema de Gestión y Análisis de Información Estadística de la Salud Pública.

Para darle solución al mismo se plantea como **Objetivo General** de la investigación: Desarrollar un subsistema que gestione el almacenamiento de la información del Sistema para la Gestión y Análisis de Información Estadística en la Salud Pública Cubana.

El **Objeto de estudio** se centra en el proceso de automatización del Sistema de Información Estadístico Complementario de Salud. El **Campo de acción** se enmarca en el almacenamiento de la Información Estadística de la Dirección Nacional de Estadísticas del Sistema Nacional de Salud.

Para dar cumplimiento al objetivo planteado se han propuesto las siguientes **Tareas:**

- Evaluar las tendencias y el estado actual en el almacenamiento de información estadística en varios sectores de Cuba y el resto del mundo.
- Analizar el procesamiento de los datos estadísticos en las unidades del Sistema Nacional de Salud.
- Asimilar las tendencias actuales en el desarrollo de aplicaciones para el almacenamiento de información estadística en Cuba y el mundo.
- Realizar las pruebas de compatibilidad con el Frameworks Mono y su ejecución en un entorno basado en el sistema operativo GNU/LINUX.
- Implementar la primera versión estable del subsistema.
- Generar la documentación necesaria para describir los procesos tanto del desarrollo como del funcionamiento de dicho subsistema.

En la investigación, para la realización de las tareas se ha utilizado como método empírico la entrevista para la obtención y elaboración de los datos y el conocimiento de los hechos fundamentales que caracterizan la situación actual, tales como: datos de entradas, tipos de reportes, restricciones de los procesos, entre otros.

Los métodos teóricos tienen una especial importancia en el proceso de la investigación, utilizándose en la construcción y desarrollo de la teoría científica y en el enfoque general para abordar los problemas de la ciencia. En este caso, mediante el análisis y la síntesis, se logró una mejor comprensión del fenómeno, dividiéndolo en partes para su solución e integrándolas para obtener un resultado final.

El documento se estructura en cuatro capítulos, como se muestra a continuación:

CAPÍTULO 1: Fundamentación teórica: Se describe la estructura y organización de conceptos fundamentales asociados al dominio del problema. Se expone un estado del arte del tema tratado, tanto a nivel nacional como internacional. Se explican y justifican, las tendencias, tecnologías y herramientas en las que se apoya la solución al problema.

CAPÍTULO 2: Características del sistema: Se abordan las características del subsistema Servidor de Datos. Se define el objeto de automatización y una propuesta del sistema. Se desarrollan los artefactos correspondientes a los flujos de trabajo Modelo de Negocio y Requerimientos.

CAPÍTULO 3: Análisis y Diseño del sistema: Se centra en la modelación detallada y la construcción de la estructura de la aplicación, obedeciendo a la arquitectura definida por el Área Temática para su posterior solución. En este capítulo se definen la estructura y los elementos del diseño, se muestran los diagramas de clases del análisis y del diseño de los casos de uso así como el modelo de datos. Finalmente, se obtiene el modelo de diseño, el cual constituye una base para la futura implementación.

CAPÍTULO 4: Implementación: Se describe la implementación del sistema propuesto a través del diagrama de componentes y la distribución del mismo mediante el diagrama de despliegue. Se realiza la descripción de los componentes del sistema.

Capítulo 1: Fundamentación Teórica

El presente capítulo tiene como objetivo abordar los elementos que fundamentan la base teórica conceptual para el desarrollo del Subsistema Servidor de Datos. Se hace alusión además a los antecedentes en Cuba y en el SNS, estos se toman como punto de partida para la concepción de la solución en cuestión. Además se realiza un análisis de las técnicas, tecnologías, metodologías, herramientas de software y lenguajes de programación sobre los que se llevará a cabo el proceso de desarrollo.

1.1 Sistema Nacional de Salud en Cuba.

Desde los primeros momentos del triunfo de la Revolución, y como uno de los puntos a cumplir del programa del Moncada, la Salud Pública constituye un elemento que distinguiría el proceso revolucionario. La garantía de atención médica gratuita a toda la población se convirtió en uno de los paradigmas sociales fundamentales.

Se comenzó a trabajar por llevar la acción del trabajador de la salud a los lugares más apartados y en esa dedicación surge el SNS, que es como concepto el conjunto de formas y métodos que sirven de base para la organización de la atención a la salud en un país determinado, designándose al Ministerio de Salud Pública (MINSAP) como su organismo rector y encargado de dirigir, ejecutar y controlar la aplicación de la política del Estado y del Gobierno en cuanto a la Salud Pública y el desarrollo de las Ciencias Médicas se refiere.

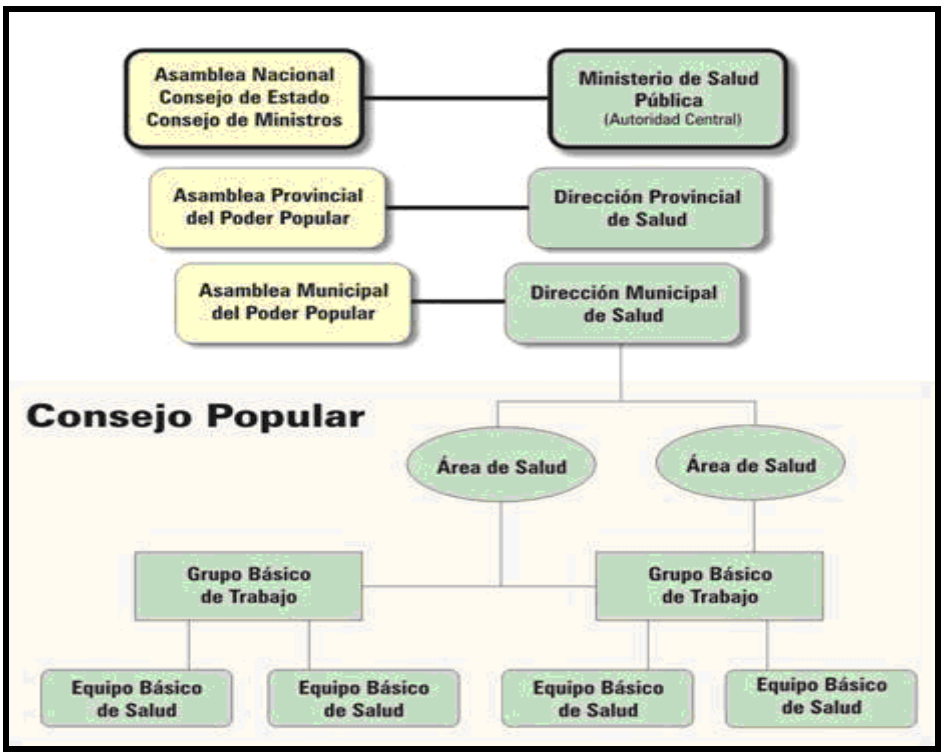


Figura 1: Esquema de la estructura organizativa del MINSAP

A partir de 1960 comenzaron a realizarse importantes reformas con el objetivo de lograr un mayor desarrollo e igualdad en la salud publica cubana. Dentro de estas reformas, las que más impacto causaron a la población sin duda fueron el surgimiento del servicio de hospitales rurales, los primeros pasos para el fortalecimiento de la atención primaria y los policlínicos integrales, así como en la década del 80 el Programa del Médico y la Enfermera de la Familia.

A raíz de estas medidas y estrategias antes mencionadas, se han logrado los resultados hasta hoy vistos en esta importante esfera, los cuales han sido posibles gracias a la Revolución, sus dirigentes y el empeño de todos los cubanos que de una forma u otra han aportado su granito de arena para que la medicina cubana se encuentre hoy entre las mejores de América.

1.2 Conceptos asociados al dominio del problema.

1.2.1 Estadísticas.

"La estadística es la ciencia que trata la recolección, clasificación y presentación de los hechos sujetos a una apreciación numérica como base a la explicación, descripción y comparación de los fenómenos".

Es transversal a una amplia variedad de disciplinas, desde la física hasta las ciencias sociales, desde las ciencias de la salud hasta el control de calidad, y es usada para la toma de decisiones en áreas de negocios e instituciones gubernamentales [3].

La Estadística se divide en dos ramas:

- La estadística descriptiva, que se dedica a los métodos de recolección, descripción, visualización y resumen de datos originados a partir de los fenómenos en estudio. Los datos pueden ser resumidos numérica o gráficamente.
- La inferencia estadística, que se dedica a la generación de los modelos, inferencias y predicciones asociadas a los fenómenos en cuestión teniendo en cuenta la aleatoriedad de las observaciones. Se usa para modelar patrones en los datos y extraer inferencias acerca de la población bajo estudio. Estas inferencias pueden tomar la forma de respuestas a preguntas si/no (prueba de hipótesis), estimaciones de características numéricas (estimación), pronósticos de futuras observaciones, descripciones de asociación (correlación) o modelamiento de relaciones entre variables (análisis de regresión).

Ambas ramas (descriptiva e inferencial) comprenden la estadística aplicada. Hay también una disciplina llamada estadística matemática, la cual se refiere a las bases teóricas de la materia.

1.2.2 Sistema de Información Estadística.

Los Sistemas de Información Estadística definen un conjunto de funciones o componentes interrelacionados que forman un todo, es decir, obtienen, procesan, almacenan y distribuyen información para apoyar la toma de decisiones y el control en una organización.

Además brindan información veraz, oportuna, relevante, exacta, útil y periódica. Tienen como objetivo fundamental lograr periódicamente la medición del cumplimiento de los planes, así como ofrecer toda aquella información estadística que posibilite el análisis integral de la evolución socioeconómica que experimenta un país.

1.2.3 Sistema de Información Estadístico Complementario de Salud (SIE_C Salud).

El Sistema de Información Estadístico Complementario de Salud (SIE_C Salud) es un sistema diverso y voluminoso por los componentes que lo integran. El mismo se expresa por la carga estadística,

número de variables, alcance geográfico, áreas del conocimiento e indicadores de salida que provee. Es uno de los mayores sistemas estadísticos complementarios del país, con un alto costo de ejecución y que no toda la información que se produce se utiliza ya que muchos de los indicadores permanecen almacenados.

El SIE_C Salud cubano es ramal, de alcance nacional y mixto, ya que contiene subsistemas cuyo soporte técnico es manual solamente o manual y automatizado. El mismo utiliza fundamentalmente el método de registro continuo y la aplicación de encuestas por muestreo, posee cobertura nacional y Departamentos de Estadística y Registros Médicos en todas las unidades e instituciones de salud.

La certificación de los eventos regulados para su captación y tratamiento estadístico posterior, es universal, y el diseño y funcionamiento de los sistemas de información estadísticos que soportan la actividad estadística en salud, se basan en los conceptos y atributos de la calidad de la información estadística tales como, factibilidad, confiabilidad, pertinencia y racionalidad. [4]

1.3 Sistemas automatizados vinculados al problema.

1.3.1 Nacional.

En años anteriores en la Universidad de las Ciencias Informáticas (UCI) se desarrollaron cinco aplicaciones correspondientes a los subsistemas Actividades de Cirugía y otras atenciones y servicios, Consulta Externa, Urgencias Médicas, Infección Intrahospitalaria y Certificados Médicos por invalidez temporal a trabajadores de los ciento treinta subsistemas con que cuenta la Dirección Nacional de Estadística.

Los subsistemas creados manejan grandes volúmenes de datos desde el nivel de unidad de salud hasta el nivel nacional, pero uno de sus principales inconvenientes es el tiempo empleado en la realización de los mismos. Unido a esto, el manejo de los datos es de forma diferente entre ellos, pues hay diferencias entre sus bases de datos y no hay un patrón estándar para la creación de las mismas, lo que trae como consecuencia desincronización en la información generada por cada uno de estos módulos y falta de integridad.

Desventajas de estos sistemas:

- Dependencia total de conectividad con sistemas externos que brindan información necesaria para el funcionamiento de la aplicación.

Este sistema surge por la necesidad de automatizar el proceso de recolección de la información estadística. Además las soluciones software ya existentes no cumplen con las expectativas, pues son pocos los modelos automatizados y los que lo están no se han estandarizado.

Otro de los sistemas automatizados encontrados en la esfera nacional fue el Sistema de Información Geográfica para la Gestión de la Estadística de Salud en Cuba. El mismo fue creado en el Instituto Superior Politécnico “José Antonio Echeverría” (CUJAE) con el objetivo de facilitar la gestión de la estadística de salud. El mismo permite cartografiar y hacer diferentes tipos de análisis de importantes indicadores de salud como la natalidad, mortalidad, demográficos, recursos y servicios. SIG-ESAC se crea principalmente para la cartografía bioestadística, aunque se le añaden algunas herramientas para estudios epidemiológicos.

1.3.2 Internacional.

INEbase

Es el sistema que utiliza el Instituto Nacional de Estadísticas de España (INE) para el almacenamiento de la información estadística, contiene toda la información que el INE produce en formatos electrónicos.

La organización primaria de la información sigue la clasificación temática del Inventario de Operaciones Estadísticas de la Administración General del Estado (IOE). La unidad básica de INEbase es la operación estadística, definida como el conjunto de actividades que conducen a la obtención de resultados estadísticos sobre un determinado sector o tema a partir de datos recogidos de forma individualizada. [5]

PC Axis

Software gratuito desarrollado por el Instituto de Estadística Sueco que permite describir y explotar grandes cantidades de datos estadísticos. En PC-Axis es posible organizar tablas, establecer bases de datos estadísticas locales, exportar tablas a una amplia variedad de formatos (Excel, dbase, HTML, etc.) y elaborar gráficos. PC-Axis se utiliza para visualizar las publicaciones que EUSTAT difunde en CD-ROM y los archivos incluidos en el apartado de Banco de Datos que usted puede descargar de forma gratuita. Programación para poder utilizarlo y permite que el usuario invierta su tiempo en determinar “qué información desea y no cómo la puede obtener”. Mediante MS-Excel el analista puede

crear sus propios indicadores, sin embargo, este tipo de operación también pueden realizarse desde el propio cubo de información. [6]

DevInfo

Es una poderosa base de datos que se utiliza para recopilar y difundir información sobre el desarrollo humano. El conjunto de programas de soporte electrónico ha evolucionado después de una década de innovaciones en los sistemas de bases de datos que apoyan la toma de decisiones con conocimiento de causa y promueven la utilización de datos para fomentar el desarrollo humano. El proyecto DevInfo es una iniciativa interinstitucional gestionada por UNICEF en nombre del sistema de las Naciones Unidas.

Los usuarios más comunes de DevInfo son los equipos de país de las Naciones Unidas, las oficinas nacionales de estadística, los ministerios de planificación y los planificadores de distrito. También son usuarios frecuentes los miembros de los medios de comunicación (para informar y registrar datos de desarrollo humano), las instituciones educativas (para analizar datos y ayudar a los estudiantes a obtener acceso a los datos), así como los administradores de DevInfo (especialmente para adaptar el sistema o añadir datos por medio de módulos avanzados de administración de la base de datos). [7]

1.5 Los Sistemas de Almacenamiento de Datos (Gestión Documental).

Durante siglos, la gestión documental en las organizaciones fue el dominio exclusivo de administradores, archiveros y bibliotecarios, cuyas herramientas manuales básicas eran los libros de registro, las carpetas, cajas y estanterías en que se guardaban los documentos de papel, unido a una larga lista de técnicas de recuperación de información mediante sistemas de codificación y clasificación. Más recientemente se fueron sumando a ellos los informáticos, que son cada vez más necesarios debido a la complejidad y nivel de sofisticación que van alcanzando los sistemas computacionales de apoyo a las actividad administrativa.

El uso del computador en la gestión documental se inicia en la práctica a partir de las grandes bibliotecas nacionales anglófonas, la Biblioteca del Congreso de los Estados Unidos de América y la British Library, que en los años 60 del siglo XX crean programas de bases de datos conocidos como MARC (Machine Readable Cataloguing) o Catalogación leíble por computador. Poco después se

comienza también a usar registros computarizados para inventariar documentación administrativa en soporte papel. [8]

Cuando el uso de las tecnologías de información y comunicación se hizo común en la administración pública y privada, con el inicio de las bases de datos y la aparición de los procesadores de textos y otras aplicaciones ofimáticas, y sobre todo con la llegada del correo electrónico, surgió la necesidad de capturar y conservar también documentos que nacen, viven y mueren en formato electrónico. Conseguir esto representó un nuevo salto en la complejidad y exigencias a los sistemas informatizados y en la forma de pensar de los administradores y archiveros.

En la actualidad, coexisten en el mundo los más diversos sistemas de gestión documental: desde el simple registro manual de la correspondencia que entra y sale, hasta los más sofisticados sistemas informáticos que manejan no sólo la documentación administrativa propiamente tal, venga ella en papel o en formato electrónico, sino que además controlan los flujos de trabajo del proceso de tramitación de los expedientes, capturan información desde bases de datos de producción, contabilidad y otros, enlazan con el contenido de archivos, bibliotecas, centros de documentación y permiten realizar búsquedas sofisticadas y recuperar información de cualquier lugar.

1.5.1 Propósito general de los sistemas de almacenamiento de datos.

El propósito general de los sistemas de almacenamientos de datos es el conjunto de normas, técnicas y prácticas usadas para administrar el flujo de documentos de todo tipo en una organización. Además permitir la recuperación de información desde ellos, determinar el tiempo que los documentos deben guardarse, eliminar los que ya no se utilicen y asegurar la conservación indefinida de los más valiosos, aplicando principios de racionalización y economía.

1.5.2 Ventajas y Desventajas de los sistemas de almacenamiento de datos.

Ventajas:

Gestión y control efectivo: De una forma sencilla, la organización tiene acceso instantáneo a toda la documentación necesaria para su actividad de negocio, con las ventajas añadidas de la eliminación de

desplazamientos, reducción de tiempo de consultas y tareas de archivo, ahorro de espacio físico, resolución del problema de localización de documentos.

Uso racional de los recursos: La gestión documental facilita que la información se comparta y se aproveche de forma más eficiente y como un recurso colectivo. Como consecuencia, se reducen drásticamente situaciones como la duplicidad de documentos archivados, fotocopias innecesarias, dobles grabaciones de datos, etc. Seguridad y fiabilidad Información, documentos, etc. de gran valor para la organización pueden custodiarse en locales de alta seguridad, garantizando su perfecto estado de conservación mientras que, para el uso diario, se dispone de su réplica electrónica.

Desventajas:

Típicamente, es necesario disponer de una o más personas que administren la base de datos, en la misma forma en que suele ser necesario, en instalaciones de cierto porte, disponer de una o más personas que administren los sistemas operativos. Esto puede llegar a incrementar los costos de operación en una empresa. Sin embargo hay que balancear este aspecto con la calidad y confiabilidad del sistema que se obtiene.

- Si se tienen muy pocos datos que son usados por un único usuario por vez y no hay que realizar consultas complejas sobre los datos, entonces es posible que sea mejor usar una planilla de cálculo.
- Complejidad: los Sistemas de Gestión de Base de Datos son software muy complejos y las personas que vayan a usarlo deben tener conocimiento de las funcionalidades del mismo para poder aprovecharlo al máximo.
- Tamaño: la complejidad y la gran cantidad de funciones que tienen hacen que sea un software de gran tamaño, que requiere de gran cantidad de memoria para ejecutarse.
- Coste del hardware adicional: los requisitos de hardware para correr un Sistema Gestor de Base de Datos por lo general son relativamente altos, por lo que estos equipos pueden llegar a costar gran cantidad de dinero.

1.6 Los sistemas de almacenamiento de datos en Cuba.

En Cuba, en la actualidad, no existen grandes Sistemas de Almacenamiento de Datos, los software que gestionan datos en la nación no son de gran complejidad y por lo general están basados en

software propietarios. Con la informatización de la sociedad cubana se espera que al menos las fundamentales ramas de la economía cubana, y otros sectores como la Salud Pública superen estas barreras.

1.7 Tendencias, tecnologías y metodologías actuales a considerar.

En la actualidad, muchas entidades han sido beneficiadas por el cambio en los mecanismos y procedimientos tradicionales de manejo de la información, debido al avance de las nuevas tecnologías. Con el tiempo se han perfeccionado la velocidad de procesamiento, la capacidad de almacenamiento masivo y la posibilidad de interconexión.

En este epígrafe se realiza un análisis detallado de los principales conceptos y tecnologías que pueden ser adecuados para el desarrollo del sistema. Se describe la arquitectura a utilizar, así como la metodología para el análisis y diseño del sistema, teniendo en cuenta las facilidades que puede aportar al trabajo. También se hace un estudio de los lenguajes de programación y del sistema de Gestión de Bases de datos (SGBD) que se empleará.

1.7.1 Lenguajes, protocolos y estándares abiertos.

Csharp (C#)

C# es un lenguaje de programación que toma las mejores características de lenguajes preexistentes como Visual Basic, Java o C++ y las combina en uno solo. El hecho de ser relativamente reciente no implica que sea inmaduro, pues Microsoft ha escrito la mayor parte de la BCL usándolo, por lo que su compilador es el más depurado y optimizado de los incluidos en el *.NET Framework SDK* [10]

Entre sus principales características destacan:

- **Sencillez:** C# elimina muchos elementos que otros lenguajes incluyen y que son innecesarios en .NET. Por ejemplo:
  - ✓ El código escrito en C# es auto contenido, lo que significa que no necesita de ficheros adicionales al propio fuente tales como ficheros de cabecera.

- ✓ El tamaño de los tipos de datos básicos es fijo e independiente del compilador, sistema operativo o máquina para quienes se compile (no como en C++), lo que facilita la portabilidad del código.
- ✓ No se incluyen elementos poco útiles de lenguajes como C++ tales como macros, herencia múltiple o la necesidad de un operador diferente del punto (**.**) acceder a miembros de espacios de nombres (**::**)

- **Modernidad:** C# incorpora en el propio lenguaje elementos que a lo largo de los años ha ido demostrándose son útiles para el desarrollo de aplicaciones y que en otros lenguajes como Java o C++ hay que simular, como un tipo básico decimal que permita realizar operaciones de alta precisión con reales de 128 bits (muy útil en el mundo financiero), la inclusión de una instrucción foreach que permita recorrer colecciones con facilidad y es ampliable a tipos definidos por el usuario, la inclusión de un tipo básico string para representar cadenas o la distinción de un tipo bool específico para representar valores lógicos.

- **Orientación a objetos:** C# es un lenguaje orientado a objetos, pero a diferencia de otros lenguajes como C++ el C# es más puro pues no admiten ni funciones ni variables globales sino que todo el código y datos han de definirse dentro de definiciones de tipos de datos, lo que reduce problemas por conflictos de nombres y facilita la legibilidad del código.

- **Orientación a componentes:** La propia sintaxis de C# incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular mediante construcciones más o menos complejas. Es decir, la sintaxis de C# permite definir cómodamente propiedades, eventos o atributos.
- **Gestión automática de memoria:** Como ya se comentó, todo lenguaje de .NET tiene a su disposición el recolector de basura. Esto tiene el efecto en el lenguaje de que no es necesario incluir instrucciones de destrucción de objetos. Sin embargo, dado que la destrucción de los objetos a través del recolector de basura es indeterminista y sólo se realiza cuando éste se active -ya sea por falta de memoria, finalización de la aplicación o solicitud explícita en el fuente-, C# también proporciona un mecanismo de liberación de recursos determinista a través de la instrucción using.

XML (Extensible Markup Language)

Fue desarrollado por el World Wide Web Consortium (W3C). No es más que un conjunto de reglas para definir etiquetas semánticas que organizan un documento en diferentes partes. Es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados. Tiene otras aplicaciones entre las que destaca su uso como estándar para el intercambio de datos entre diversas aplicaciones o software con lenguajes privados como en el caso del SOAP. [11]

XML se ha convertido en un objeto de culto para los desarrolladores que prefieren la transparencia de datos y la compatibilidad entre aplicaciones, pues es una manera simple a través de la cual representar datos. Entre las ventajas que provee XML, la flexibilidad que ofrece es altamente útil para diferentes propósitos. La información se obtiene de manera estructurada y descriptivamente visual y la información ya no es un conjunto de datos sin sentido que solo entendía el sistema para el cual es enviada. Ahora el desarrollador puede definir su propia estructura de datos, a manera similar como se hacia en las BD antiguas, pero con la certeza que otras personas y otros sistemas puedan "entender" su propio esquema de definición de datos (aquí radica la importancia de XML).

SOAP (Single Object Access Protocol)

Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Este deriva de un protocolo creado por David Winer en 1998, llamado XML-RPC. SOAP fue creado por Microsoft, IBM entre otros y está actualmente bajo el auspicio de la W3C. Los mensajes SOAP son independientes de cualquier sistema operativo y protocolo, y pueden ser transportados usando una variedad de protocolos de Internet, incluyendo HTTP, SMTP y MIME. Es uno de los protocolos utilizados en los servicios Web. [13]

WSDL (Web Services Descripcion Language)

WSDL describe la interfaz pública a los servicios Web. Está basado en XML y describe la forma de comunicación, es decir, los requisitos del protocolo y los formatos de los mensajes necesarios para interactuar con los servicios listados en su catálogo. Las operaciones y mensajes que soporta se describen en abstracto y se ligan después al protocolo concreto de red y al formato del mensaje. [12]

1.7.2 Servicios Web

Los Servicios Web son sistemas de comunicación entre diferentes servidores, a través de la red, basados en mensajes que cumplen un estándar (SOAP) basado en XML. Son aplicaciones independientes de la plataforma que pueden ser fácilmente publicadas, localizadas e invocadas. [9]

Un Servicio Web en vez de obtener solicitudes desde el navegador y retornar páginas Web como respuesta, lo que hace es recibir solicitudes a través de un mensaje formateado en XML desde una aplicación, realiza una tarea y devuelve un mensaje de respuesta también formateado en XML.

1.7.3 Arquitectura de Software

Una arquitectura software viene determinada por las diferentes instancias de cada tipo de componentes y conectores que la componen, y por una serie de enlaces específicos que definen la unión de todas ellas formando una estructura. A esta estructura se le da el nombre de *configuración*, y suele considerarse insertada en una *jerarquía*, pues toda entidad software, independientemente de su granularidad, dispone de una estructura que puede ser descrita mediante una arquitectura software.

En general, la arquitectura software nace como una herramienta de alto nivel para cubrir distintos objetivos, entre los que se destacan:

- Comprender y manejar la estructura de las aplicaciones complejas.
- Reutilizar dicha estructura (o partes de ella) para resolver problemas similares.
- Planificar la evolución de la aplicación, identificando sus partes mutables e inmutables, así como los costes de los posibles cambios.
- Permitir el estudio de alguna propiedad específica del dominio.
- Analizar la corrección de la aplicación, y su grado de cumplimiento respecto a los requisitos iníciales (prestaciones o fiabilidad).

Arquitectura Multinivel

El objetivo primordial es la separación de la lógica de negocios de la lógica de diseño. Un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario. La ventaja principal de esta arquitectura es que el desarrollo se puede llevar a cabo en varios niveles . En caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado.

Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles, de forma que basta con conocer la API que existe entre niveles. En el diseño de sistemas informáticos actual se suele usar mucho la arquitectura multinivel. En dichas arquitecturas a cada nivel se le confía una misión simple, lo que permite el diseño de arquitecturas escalables (que pueden ampliarse con facilidad en caso de que las necesidades aumenten).El diseño más utilizado actualmente es el diseño en tres niveles (o en tres capas). [14]

1.7.4 Sistema de Gestión de Bases de Datos

PostgreSQL 8.2

PostgreSQL es un poderoso sistema de bases de datos relacional que cuenta ya con más de 15 años de desarrollo activo y una arquitectura probada, la cual ha ganado una fuerte reputación en cuanto a confiabilidad, integridad de los datos y precisión.

Funciona en la mayoría de los sistemas operativos (Linux, UNIX y Windows). Es completamente conforme al estándar ACID27 (Atomicidad, Consistencia, Aislamiento y Durabilidad). Este estándar es considerado la propiedad clave de los sistemas de gestión de bases de datos de procesamiento de transacciones.

Posee soporte pleno de llaves foráneas, uniones, vistas, triggers y funciones en varios lenguajes. Incluye la mayoría de los tipos de datos de SQL92 y SQL99 y soporta el almacenamiento de objetos binarios de gran tamaño, incluyendo imágenes, sonido y video. Posee interfaces de programación nativas para C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otras y muy buena documentación.

PostgreSQL posee características sofisticadas como Control de Concurrencia Multi-Versiones (MVCC28), recuperación en puntos de tiempo, espacios de tablas, replicación asíncrona, transacciones anidadas, copias de respaldo en caliente, planificador-optimizador de consultas, entre otras. Es altamente escalable, tanto en la cantidad de datos que puede manipular como la cantidad de usuarios concurrentes que soporta. [15]

SQLite 3.5

Es un sistema de gestión de bases de datos relacional compatible con ACID, y que está contenida en una relativamente pequeña (~500KB) biblioteca en C. SQLite es un proyecto de dominio público creado por D. Richard Hipp.

A diferencia de los sistemas de gestión de base de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de ello, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones.

Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un solo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción. [16]

1.7.5 Desarrollo basado en RUP

RUP (Rational Unified Process)

La metodología de desarrollo de software RUP es muy generalizada y estudiada; brinda una información detallada de las etapas, ciclos, flujos de trabajo y artefactos en el desarrollo de un software. Esta metodología se ajusta muy bien a los modelos orientados a objetos, posibilita el modelado de software siguiendo este paradigma. RUP enfrenta el problema desde sus inicios y hace un seguimiento a través de las distintas vistas hasta que se logra una estabilización en el software desarrollado. También norma qué artefactos se generan durante el proceso y define qué, cuándo y quién debe realizar una tarea especificada. Organiza el desarrollo para poder hacer planificaciones y definir hitos que posibiliten controlar el progreso del proceso de desarrollo. [17]

UML 2.0 (Unified Modeling Language)

Lenguaje de Modelado Unificado es una representación gráfica que posibilita especificar, construir, visualizar y documentar las partes o artefactos (información que se utiliza o produce mediante un proceso de software). Pueden ser artefactos: un modelo, una descripción que comprende el desarrollo de software que se basen en el enfoque Orientado a Objetos, utilizándose también en el diseño Web.

En las versiones previas del UML, se hacía un fuerte hincapié en que UML no era un lenguaje de programación. Un modelo creado mediante UML no podía ejecutarse. En el UML 2.0, esta asunción cambió de manera drástica y se modificó el lenguaje, de manera tal que permitiera capturar mucho más comportamiento. De esta forma, se permitió la creación de herramientas que soporten la automatización y generación de código ejecutable, a partir de modelos UML. [18]

1.7.6 Herramientas

En vista a los epígrafes antes mencionados, al creciente desarrollo de las tecnologías en la actualidad y analizando también el auge de las aplicaciones libres (Software Libre) como el comienzo de una nueva etapa en el desarrollo de la Informática y las Comunicaciones, se seleccionaron las siguientes herramientas para la construcción del sistema que dará solución al problema planteado:

SharpDevelop

Es un entorno de desarrollo integrado libre para los lenguajes de programación C#, Visual Basic .NET y Boo. Es usado típicamente por aquellos programadores de los citados lenguajes, que no desean o no pueden usar el entorno de desarrollo de Microsoft, el Microsoft Visual Studio. Hay disponible un puerto para Mono/Gtk#, llamado MonoDevelop, el cual funciona en otros sistemas operativos. [19]

Para el completado automático de código, la aplicación incorpora sus propios pases. La versión 1.1 de la aplicación puede importar proyectos de Visual Studio .NET. La versión 2.0 ya es capaz de editarlos directamente.

Entre sus principales características se encuentra:

- Incorpora un diseñador de formas de Windows.
- Completado de código. Soporta el uso de la combinación de teclas Ctrl + Espacio.

- Depurador incorporado.
- Herramientas para "Ir a Definición", "Encontrar referencias" y "renombrado".
- Títulos para títulos y para depuración.

Enterprise Architect

La herramienta de modelado Enterprise Architect, es la herramienta potente y flexible para la plataforma de Windows. Enterprise Architect provee el límite competitivo para el desarrollo de software, administración de proyecto, administración de requerimientos y análisis de negocio y entre sus características más importantes figuran: [20]

- Última especificación UML 2.0
- Importación/Exportación XMI 2.0
- Nuevo motor de Reporte HTML
- Permite transformar elementos simples en objetivos complejos.
- Perfiles y soporte de Tecnologías
- Pruebas, rastreo de recursos, mantenimiento

PostgreSQL Manager

PostgreSQL Maestro es la herramienta de administración GUI de Windows para manejo, control y desarrollo de PostgreSQL. Ello le permite hacer todas las operaciones de base de datos fácil y rápidamente. Usando maestro de PostgreSQL se puede crear, editar, copiar, y dejar caer toda la base de datos objeta tales como esquemas, tablas, vistas, funciones, campos, reglas, sucesiones, lenguajes, operadores, etc.; construye las consultas visualmente, ejecutándolas y verificando sintaxis de SQL. [21]

SQLite Manager 3.5.4

SQLite Manager es un gestor de bases de datos SQLite, con una interfaz muy clara, dividida en pestañas para los elementos de diseño, administración y la elaboración de instrucciones SQL, con la posibilidad de crear y navegar por las tablas, índices y vistas, insertar, eliminar y editar las tablas, ejecutar sentencias SQL, etc. Dispone de un completo sistema de generación de informes, exportables en una gran variedad de formatos, incluyendo HTML, CSV y XML. En definitiva, SQLiteManager ofrece

una forma más amena para navegar entre los objetos de las bases de datos, gestionar las mismas y construir instrucciones SQL, junto a una tabla completa con los resultados de las peticiones formuladas. [22]

ProFTPd

ProFTPd es un servidor FTP que se promociona a sí mismo como un "Software servidor FTP altamente configurable con licencia GPL (Licencia Publica General) y con una amplia documentación. ProFTPd usa un único fichero de configuración "/etc/proftpd.conf". El fichero de configuración es muy similar al que tiene Apache. Puede ser fácilmente configurado como múltiples servidores FTP virtuales, y tiene capacidades para ser enjaulado dependiendo del sistema de archivos que haya por debajo. Puede ejecutarse con un demonio propio o como un servicio más de inetd. Es capaz de trabajar sobre IPv6.Su diseño es modular, lo que permite escribir extensiones como cifrado SSL/TLS, RADIUS, LDAP o SQL como módulos.

En este capítulo se abordaron aspectos fundamentales del Sistema Nacional de Salud en Cuba. Así como los requisitos y premisas definidos para la informatización de la salud en el país, con el objetivo de analizar el objeto de estudio definido. Se profundizó en los antecedentes históricos e importancia de la información estadística en Cuba. Se estudiaron sistemas vinculados al campo de acción en el ámbito nacional e internacional y se realizó un análisis de las tecnologías, lenguajes y herramientas para dar cumplimiento al proceso de desarrollo.

Capítulo 2: Características del Sistema.

En este capítulo se realiza la descripción de las características del sistema. Para poder entender mejor el contexto en que se ubica, se definen conceptos agrupados en un Modelo de Dominio. Se enumeran además los requisitos funcionales y no funcionales que debe tener el sistema propuesto, lo que permite hacer una concepción general del trabajo. Así como, identificar mediante un Diagrama de Casos de Uso, las relaciones de los actores y las secuencias de acciones que se realizan. Se realiza además la descripción de cada uno de los casos de usos definidos, lo que facilita una mejor interpretación de los mismos.

2.1 Descripción del Sistema

Se propone un sistema que cuenta con tres subsistemas que se describen a continuación:

Subsistema Editor de Plantillas

Partiendo de que toda la información estadística es captada mediante los Modelos de flujo, los cuales responden a las necesidades de información de cada uno de los subsistemas de información del SIE-C de Salud y que son creados por un conjunto de especialista de la rama, y ellos definen el formato y las reglas bajo la cuales será asentada la información captada, surge el desarrollo del Subsistema Editor de Plantillas que responde a esta necesidad. El mismo tiene como objetivo automatizar la gestión y edición de los modelos de flujo para la captura de la información estadística en el sector de la salud.”

Subsistema Servidor de Datos

El Subsistema Servidor de Datos se encarga del manejo y la persistencia de la información estadística. Es un componente que además permanecerá distribuido entre el centro de datos, y las estaciones de trabajo, siendo en estas últimas donde se almacenará una copia de la información perteneciente al nivel en que se encuentren trabajando. Ofrece un conjunto de interfaces mediante Servicios Web, que permitirán comunicación con otros sistemas que constituyen las fuentes primarias de información y además brindara funcionalidades auxiliares para la captura de información a partir de ficheros almacenados en soporte extraíble (Memorias, CD, DVD, Disquetes).

Subsistema Captura y Reparo de Información

El Subsistema Captura y Reparo de Información tiene un conjunto importante de funciones dentro de su negocio. El mismo está encargado de la captura, reparo y validación de la información a partir de los modelos de flujo, así como de establecer comunicación estrecha con el componente servidor de datos, tanto para persistir la información como para obtener actualizaciones que les sean realizadas a los modelos de flujo.

2.2 Modelo de Dominio

Permite de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio del sistema en desarrollo, puesto que los procesos del negocio no son visibles. Esto ayuda a los usuarios, clientes, desarrolladores e interesados, a utilizar un vocabulario común para poder entender el contexto en que se emplaza el sistema. Para capturar correctamente los requisitos y poder construir un sistema que cumpla con las metas previstas, se necesita tener un firme conocimiento del funcionamiento del objeto de estudio. Este modelo va a contribuir posteriormente a identificar algunas clases que se utilizarán en el sistema.

2.2.1 Conceptos Fundamentales

Los conceptos son un elemento importante en la realización del Modelo de Dominio, ya que permiten de manera visual representar las principales características que se manejan en el dominio del sistema en desarrollo. Luego de un análisis profundo del problema que se plantea, se han derivado como conceptos fundamentales los siguientes:

Plantilla: La clase plantilla no es más que el modelo de flujo que contiene información, datos y reglas de validación. Estas se generan como un XML.

Esquema XML: La clase Esquema XML describe un fichero de texto con formato XML que contiene la estructura de la plantilla.

Servidor de Datos: Es la clase controladora del modelo, pues es la que provee a todas las otras clases de la información y el acceso a los datos que necesiten.

WebServices: Es la encargada de brindar los servicios de conexión y acceso al servidor de datos mediante una interfaz WDSL.

Registro: Es la que se encarga de registrar la información de los modelos de flujos por cada entidad, en un periodo determinado.

Reporte: Se encarga de generar los reportes solicitados

Consolidado: Es la que se encarga de generar en un documento PDF o Excel (Modelo Consolidado) los reportes obtenidos en la clase Reportes.

Editor de Plantillas: Esta clase no se especifica pues no es utilizada para la realización de este trabajo. Solo está puesta en el Modelo de Dominio para una visión general del sistema. Representa un componente externo al subsistema.

Captura y Reparo de la Información: Esta clase no se especifica pues no es utilizada para la realización de este trabajo. Solo esta puesta en el Modelo de Dominio para una visión general del sistema. Representa un componente externo al subsistema.

2.2.2 Modelo del Dominio.

El modelo del dominio se representa en forma de diagrama de clases, donde figuran los principales conceptos y roles del sistema analizado. A continuación se muestra el mismo:

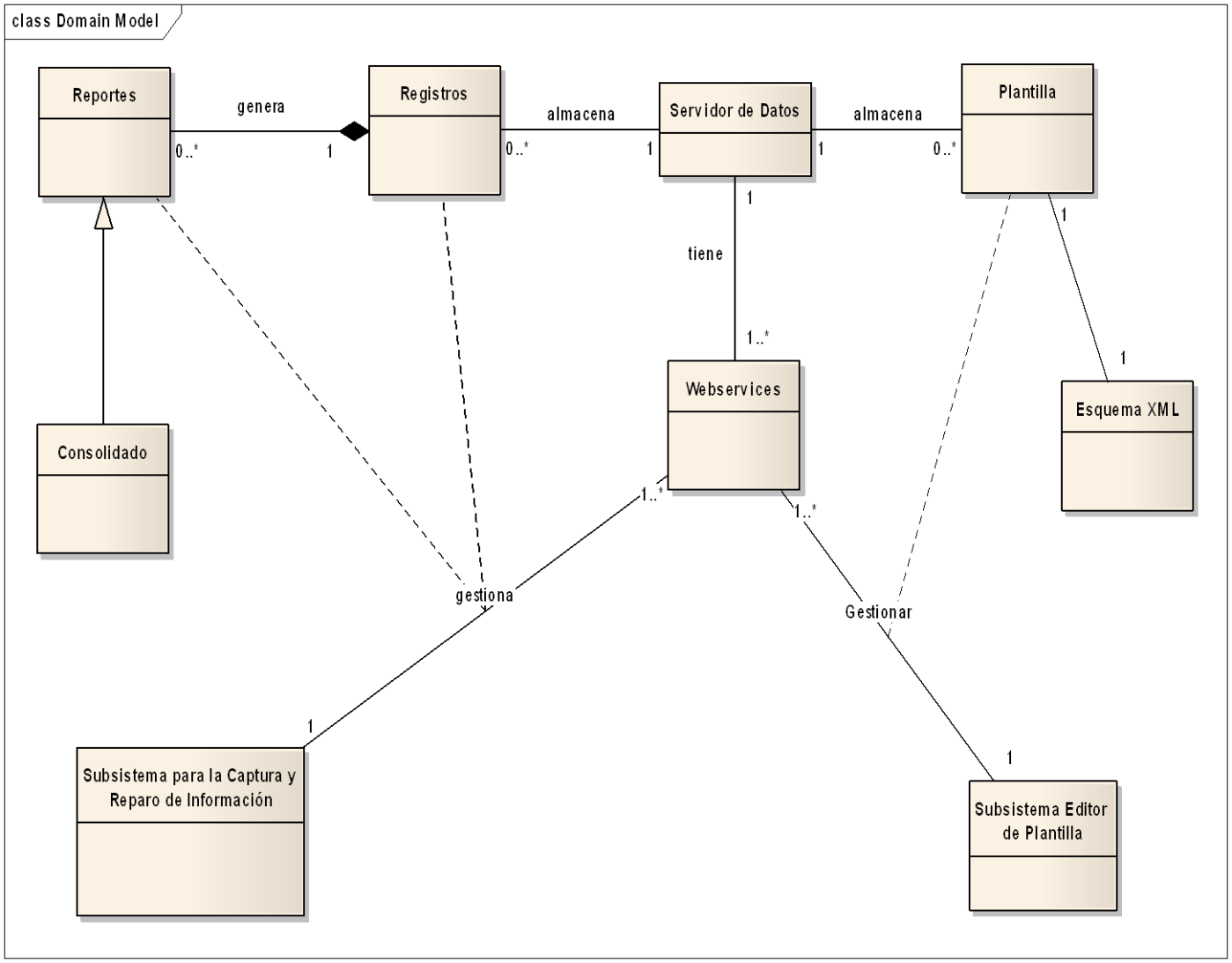


Figura 2: Modelo de Dominio.

2.3 Levantamiento de Requisitos

El flujo de trabajo modelado del negocio da una visión de qué es necesario hacer para dar respuesta a las solicitudes del cliente, lo cual se logra definiendo los procesos más importantes, obteniendo así un modelo de dominio. El modelado del negocio brinda una vía natural para determinar los requerimientos que debe tener el Subsistema Servidor de Datos.

2.3.1 Requisitos Funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Se mantienen invariables sin importar con qué propiedades o cualidades se relacionen. Con su definición se busca establecer un común entendimiento con el cliente.

A continuación se enumeran los principales requisitos funcionales que debe cumplir el Subsistema Servidor de Datos para la satisfacción del cliente.

N °	Requisitos Funcionales
RF1	Brindar servicio web para insertar plantilla.
RF2	Verificar credenciales.
RF3	Verificar archivos.
RF4	Insertar periodicidad de la plantilla.
RF5	Guardar fecha de actualización de la plantilla. (En este caso coincide con la de creación)
RF6	Listar registros existentes en la BD.
RF7	Brindar servicio web para actualizar plantilla.
RF8	Brindar servicio web para listar plantillas.
RF9	Brindar servicio web para bloquear plantilla.
RF10	Brindar servicio web para desbloquear plantilla (hacerla pública).

RF11	Brindar servicio web para insertar registro.
RF12	Insertar nombre de la plantilla.
RF13	Insertar fecha de creación de la plantilla.
RF14	Insertar periodo.
RF15	Insertar identificador de la unidad.
RF16	Insertar fecha de creado.
RF17	Listar plantillas existentes en la BD.
RF18	Brindar servicio web para actualizar registro.
RF19	Brindar servicio web para buscar registro por identificador.
RF20	Brindar servicio web para buscar registro por periodo, unidad y plantilla.
RF21	Brindar servicio web para buscar registro por periodo, unidad y plantilla.
RF22	Brindar servicio web para buscar registro por unidad (listado).
RF23	Brindar servicio web para buscar registro por plantilla (listado).
RF24	Construir reporte.
RF25	Verificar nivel de acceso.
RF26	Verificar nivel de validación.
RF27	Insertar nivel de validación.
RF28	Cambiar nivel de validación del registro.
RF29	Validar registro.
RF30	Devolver permisos (toquen).
RF31	Insertar registro (importar).

RF32	Actualizar registro (sobrescribir).
RF33	Insertar plantilla (importar).
RF34	Actualizar plantilla (sobrescribir).
RF35	Insertar usuario de la BD
RF36	Insertar contraseña de la BD
RF37	Insertar url donde van a estar los ficheros XML.
RF38	Insertar usuario de acceso al ftp (actualizar en el directorio las plantillas).
RF39	Insertar contraseña de acceso al ftp.
RF40	Actualizar plantillas en el directorio.
RF41	Enviar credenciales al Componente de Seguridad.
RF42	Construir consolidado.
RF43	Solicita credenciales.
RF44	Guardar fecha de actualización del registro.
RF45	Administrar Configuración
RF46	Buscar Registros
RF47	Buscar Plantillas

2.3.2 Requisitos no Funcionales

Los requerimientos no funcionales son propiedades o cualidades que el sistema a desarrollar debe tener. Definiendo propiedades o características que hacen al producto atractivo, usable, rápido y confiable. Estos requerimientos no funcionales (RNF) son fundamentales en el éxito del producto y normalmente están vinculados a requerimientos funcionales, es decir, una vez que se conozca lo que el sistema debe hacer, se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser.

Existen múltiples categorías para clasificar a los requerimientos no funcionales, siendo las siguientes las más representativas dentro del conjunto de los mismos.

N °	Categoría	Subcategoría	Requisitos no Funcionales
RNF1	**Hardware**		El subsistema deberá correr en un ordenador Pentium 4 o superior.
RNF2	**Hardware**		El ordenador donde se ejecute el sistema contará con 2 GB de memoria RAM como mínimo.
RNF3	**Hardware**		El ordenador donde se ejecute la aplicación contará con 1 TB de disco duro y un Backus para la salva de la información.
RNF4	**Rendimiento**		El sistema debe estar poco cargado y distribuido, para posibilitar respuestas más rápidas.
RNF5	**Software**		El servidor debe tener PostgreSQL 8.2 o superior, servidor HTTP que soporte ASP.NET y MONO 1.9 o superior. Debe disponer de un servidor ProFTPd 2.4 virtual para la descarga de plantillas.

RNF6	**Soporte**		El personal que trabaja con el subsistema debe contar con el nivel técnico requerido mediante adiestramiento de servicio.
RNF7	**Usabilidad**		El sistema debe garantizar un acceso rápido y fácil, podrá ser usado por cualquier usuario del Sistema Nacional de Salud (SNS).
RNF8	**Portabilidad**		Permitir que el subsistema se ejecute sobre Sistema Operativo Linux y Windows Server.
RNF9	**Seguridad**		Disponer de un mecanismo de seguridad basado en el Componente de Seguridad del Área Temática Sistema de Apoyo a la Salud (SAS).
RNF10	**Seguridad**	**Disponibilidad**	La seguridad no implicará lentitud o retraso en la repuesta dada por el subsistema, por lo que se debe minimizar y reducir el tiempo de respuesta, así como optimizar el código.
RNF11	**Eficiencia**		El sistema deberá ser rápido ante las solicitudes de los usuarios en el procesamiento de la información, el tiempo de respuesta deberá ser menos de 5 segundos.
RNF12	**Estándares Aplicables**		Para la implementación del subsistema se deberán seguir los estándares de codificación definidos por el área temática.
RNF13	**Documentación y Ayuda en Línea**		Se dispondrá de la documentación del sistema realizada con RUP y de un Manual de Usuario. Además cuenta con el Manual de Instalación que indicará los pasos a seguir para instalar y configurar el subsistema.

2.4 Modelo del Sistema.

El modelo de casos de uso permite llegar a un entendimiento común sobre cómo utilizar el Subsistema Servidor de Datos, entre los clientes, usuarios y desarrolladores.

2.4.1 Definición de los actores del sistema.

A continuación se definen y describen los actores del sistema. Estos actores pueden estar representados por un usuario en concreto (una persona u otro sistema externo) que interactúe con el sistema en desarrollo.

Actor	Descripción
Administrador del Sistema	Es el encargado de definir los privilegios de acceso a los usuarios del sistema.
Subsistema editor de plantilla	Sistema externo que se encarga de crear y modificar las plantillas.
Subsistema para la captura y reparo de la información	Sistema externo que se encarga del manejo de la información a través de registros. Este brinda la opción de generar reportes.
Componente de Seguridad	Sistema externo que a través de un usuario y una contraseña determinada, devuelve una serie de datos pertenecientes al usuario y los permisos que tiene para acceder a la información.

2.4.2 Diagrama de casos de uso del sistema.

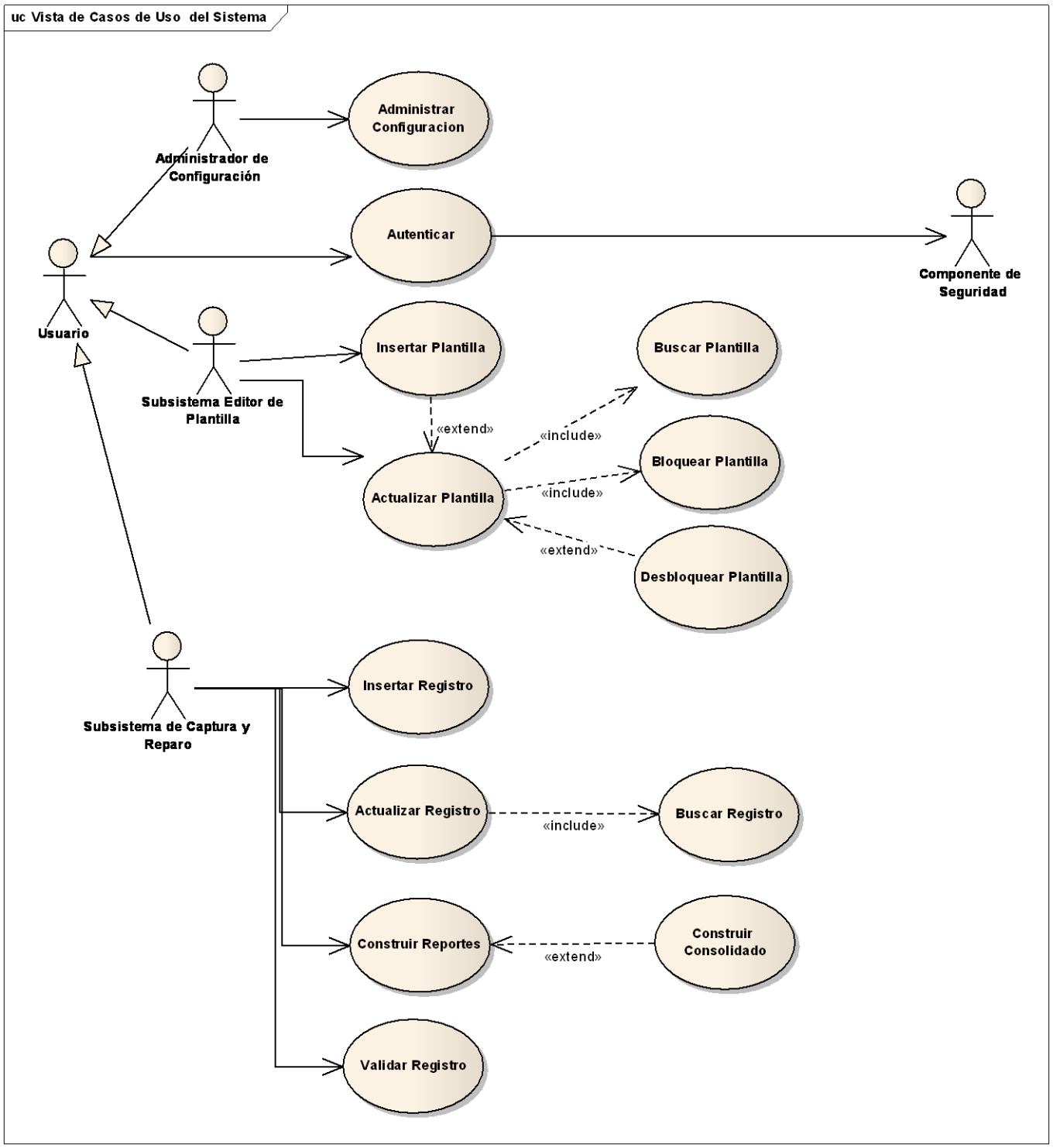


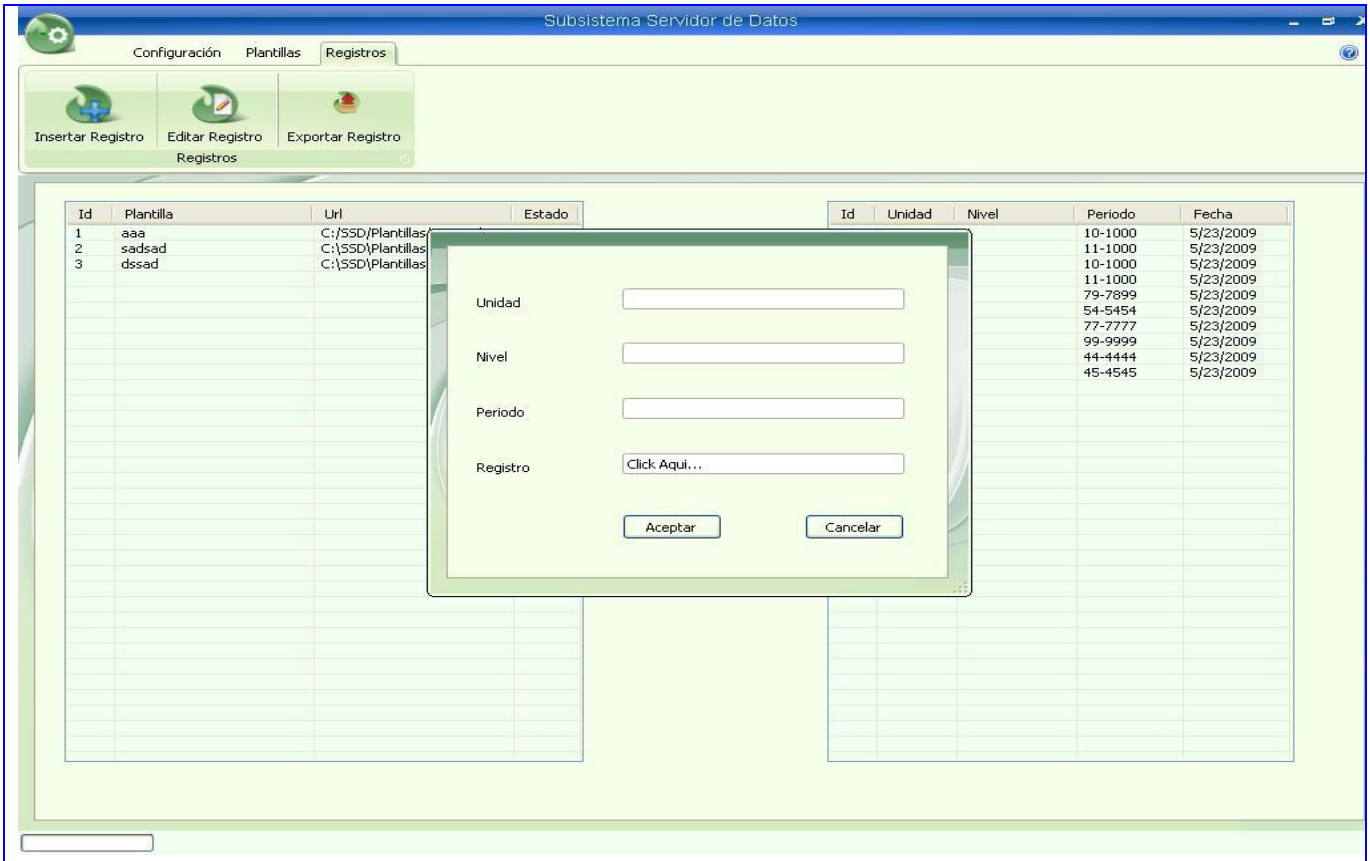
Figura 3: Diagrama de casos de uso del sistema.

2.4.3 Descripción textual de los casos de uso del sistema.

2.4.3.1 Caso de uso Administrar Configuración.

Caso de Uso:	**Administrar Configuración**
Actor	**Administrador de Configuración**
Resumen:	El caso de uso se inicia cuando el Administrador de Configuración solicita configurar el sistema. El caso de uso termina cuando el sistema notifica la validez de los cambios realizados.
Precondiciones:	**El usuario tiene que estar autenticado.**
Referencias	**RF31, RF32, RF33, RF34, RF35, RF36, RF37, RF38, RF39, RF40, RF14, RF15, RF16, RF17, RF6, RF27**
Prioridad	**Alta**

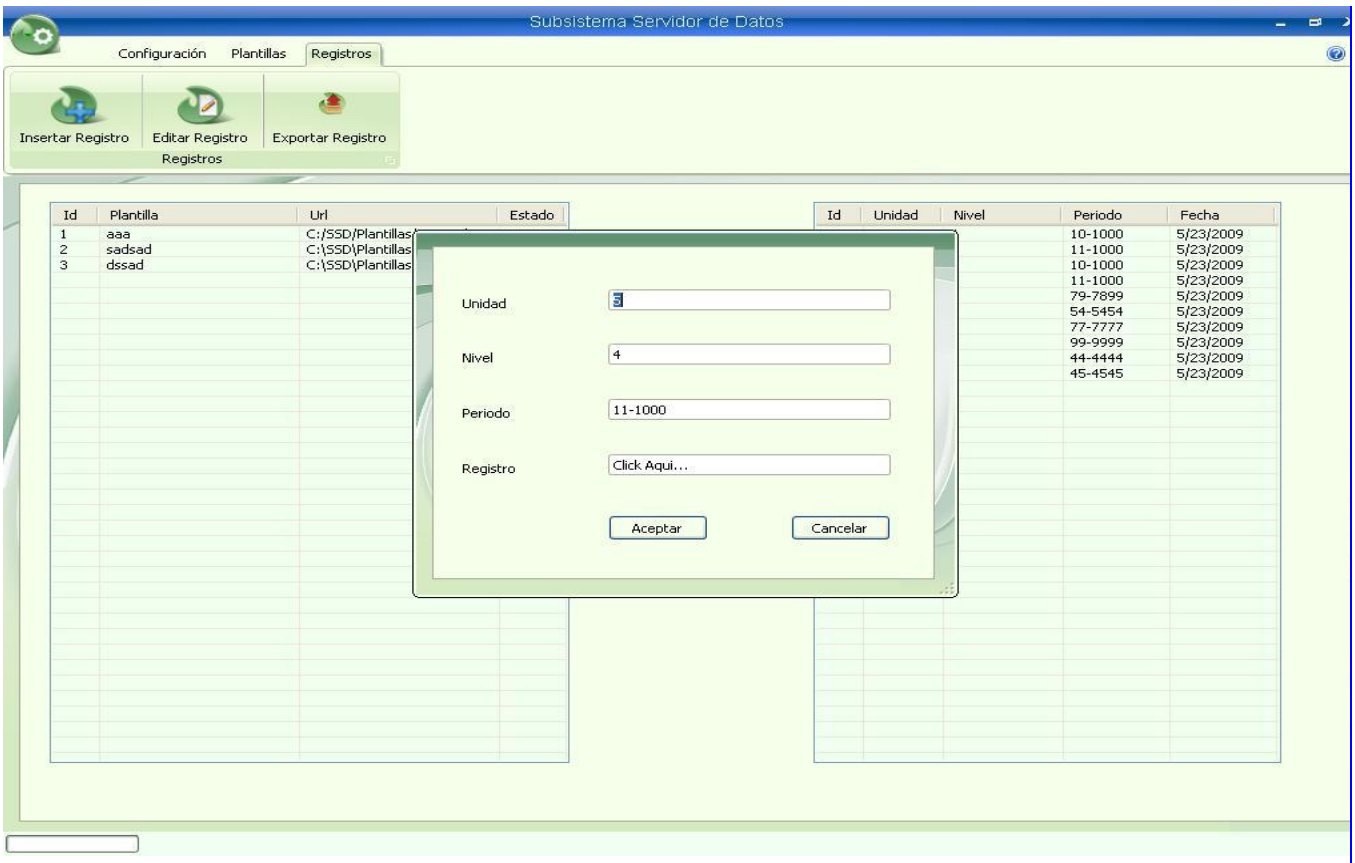
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Administrador de Configuración solicita configurar el Sistema.	2 El sistema le muestra una pantalla con las opciones de configuración que presenta.
Sección ”Insertar Registro”	
3 El Administrador de Configuración selecciona la opción Insertar Registro en el menú Registros.	4 El sistema muestra una pantalla para insertar los datos propios del registro ().
Prototipo de Interfaz	



Sección " Editar Registro"

3 El Administrador de Configuración selecciona la opción Editar Registro en el menú Registros.	4 El sistema muestra una pantalla para que el actor introduzca una serie de datos propios del registro y luego este sea buscado en el Servidor de Datos (Ver CU Buscar Registro). 5 Luego de encontrado el registro el actor realiza las modificaciones que quiera sobre dicho registro.

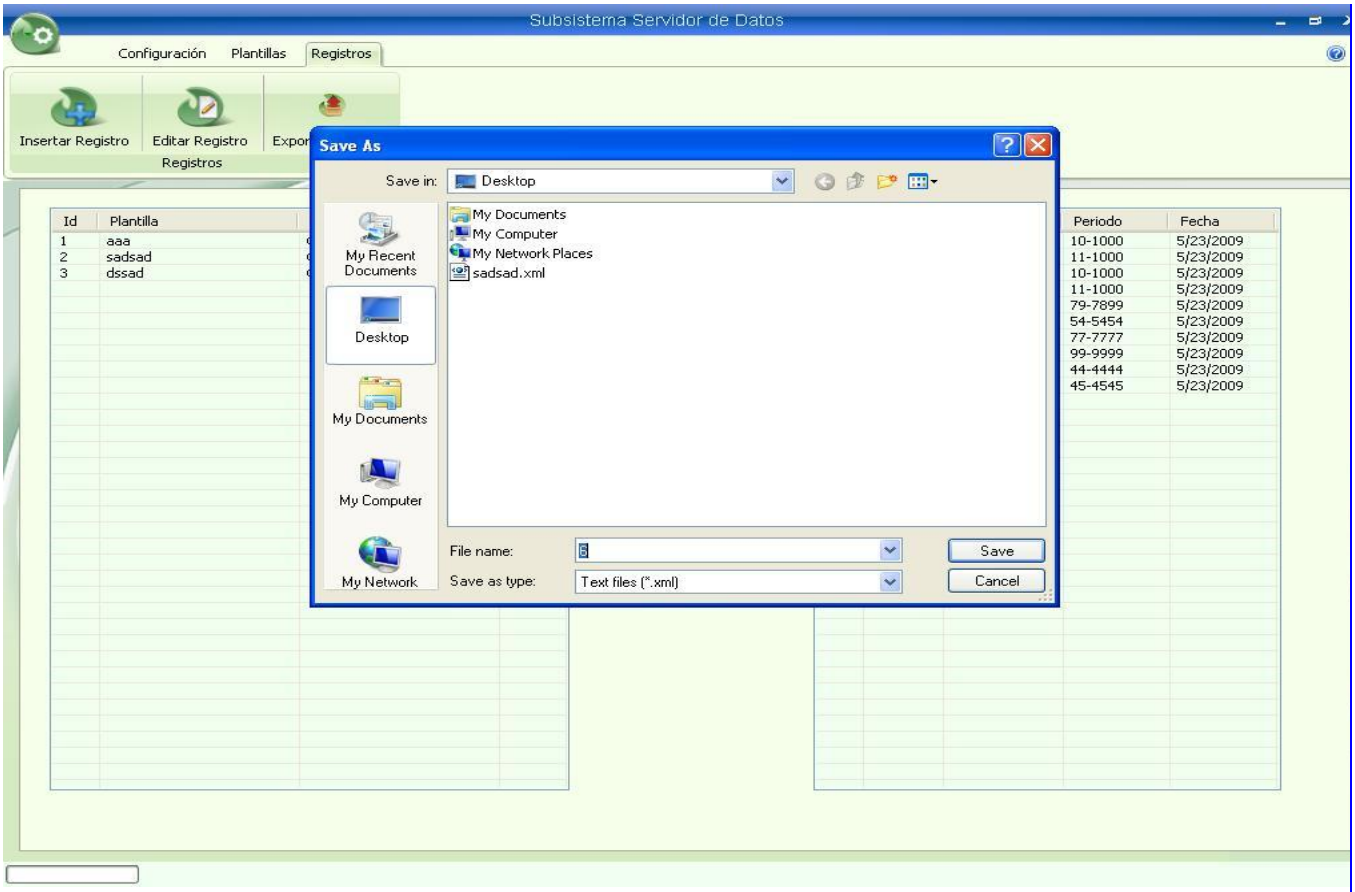
Prototipo de Interfaz



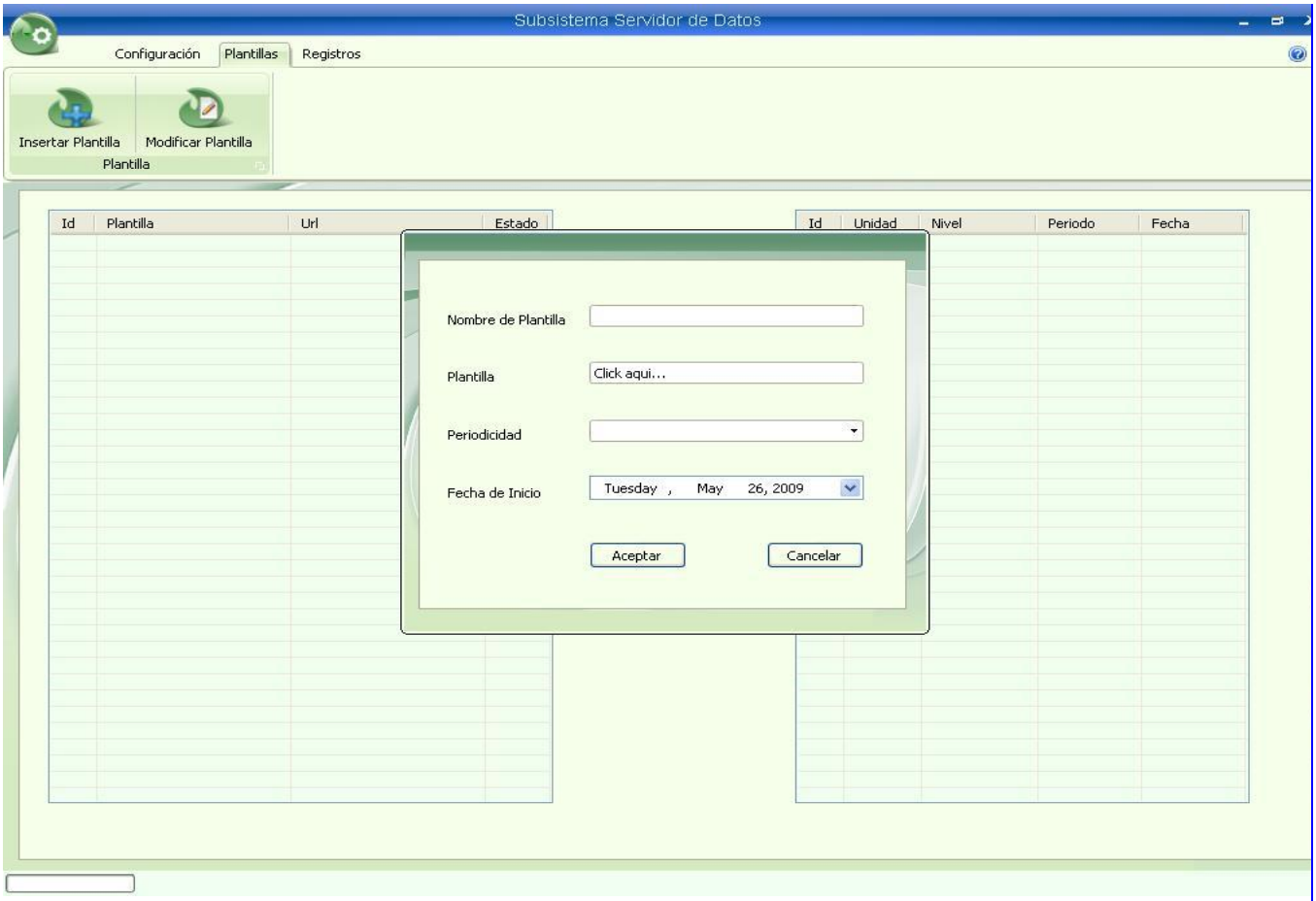
Sección " Exportar Registro"

3 El Administrador de Configuración selecciona la opción Exportar Registro en el menú Registros.	4 El sistema muestra una venta para que seleccione en que lugar de su ordenador desea exportar el registro.

Prototipo de Interfaz



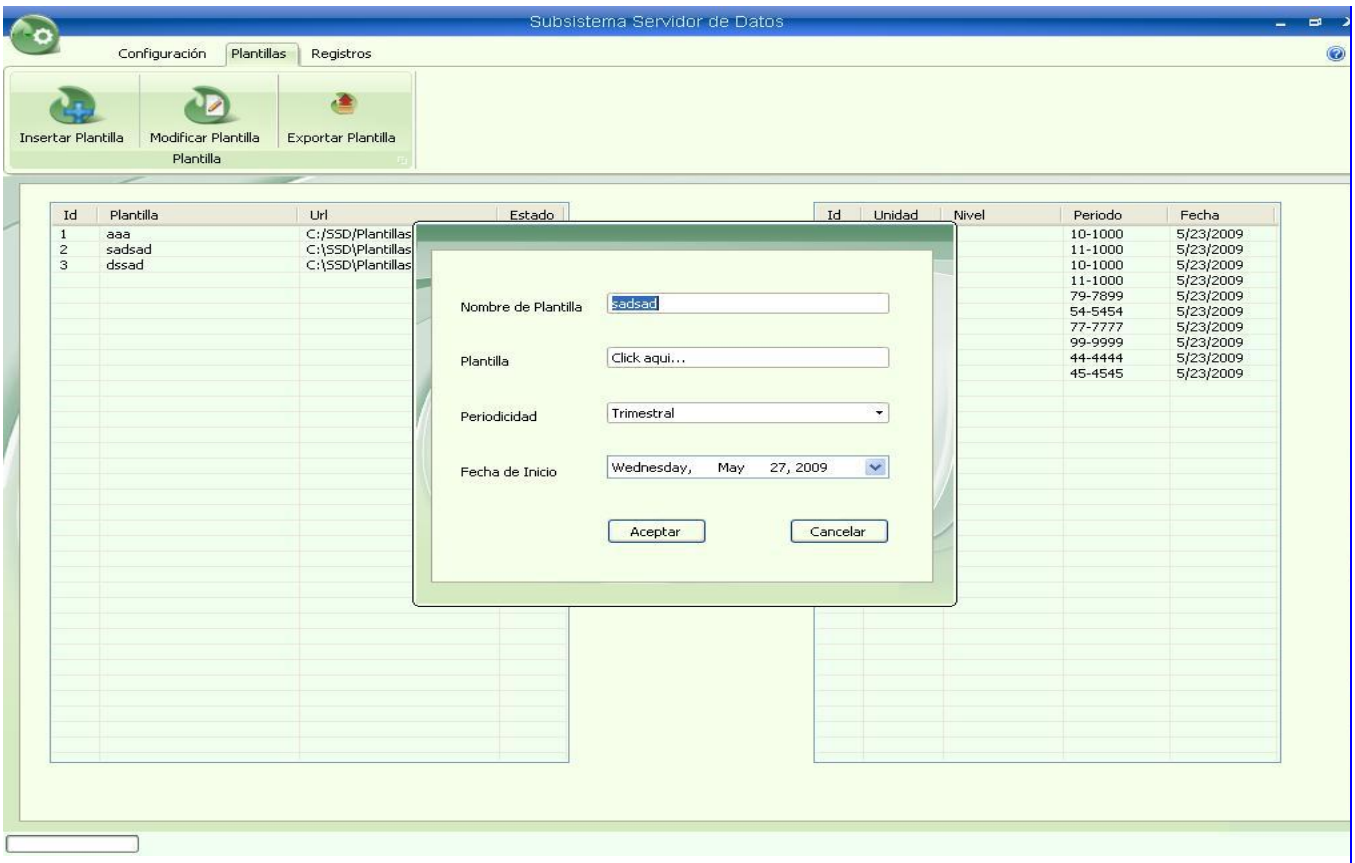
Sección “ Insertar Plantilla”	
3 El Administrador de Configuración selecciona la opción Insertar Plantilla en el menú Plantillas.	4 El sistema muestra una pantalla para insertar los datos propios de la plantilla ().
Prototipo de Interfaz	



Sección " Modificar Plantilla"

3 El Administrador de Configuración selecciona la opción Modificar Plantilla en el menú Plantillas.	4 El sistema muestra una pantalla para introducir una serie de datos referentes a la plantilla que desea buscar (Ver CU Buscar Plantilla). 5 Luego de encontrada la plantilla el actor puede modificar la misma.

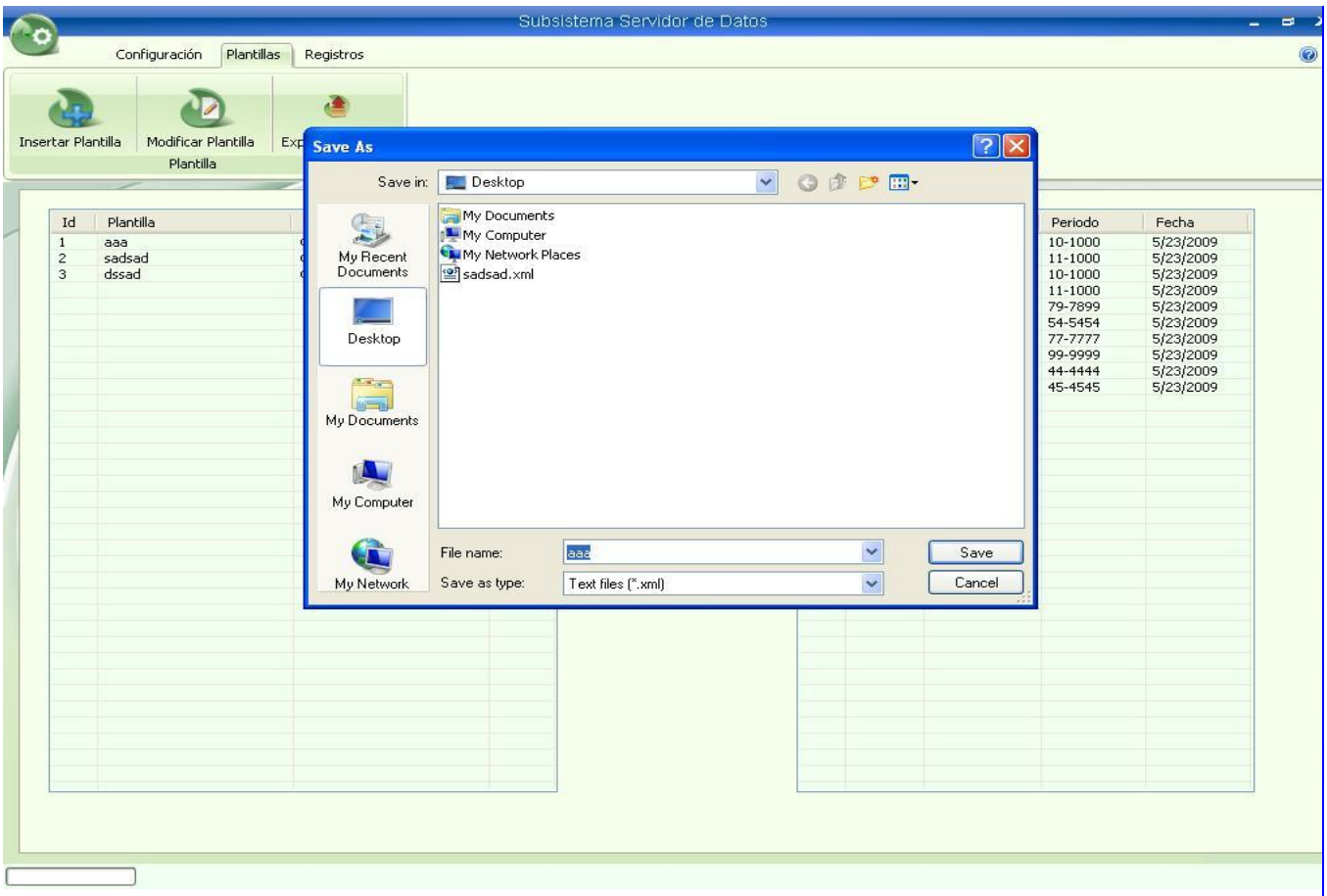
Prototipo de Interfaz



Sección " Exportar Plantilla"

3 El Administrador de Configuración selecciona la opción Exportar Plantilla en el menú Plantillas.	4 El sistema muestra una venta para que seleccione en que lugar de su ordenador desea exportar la plantilla.

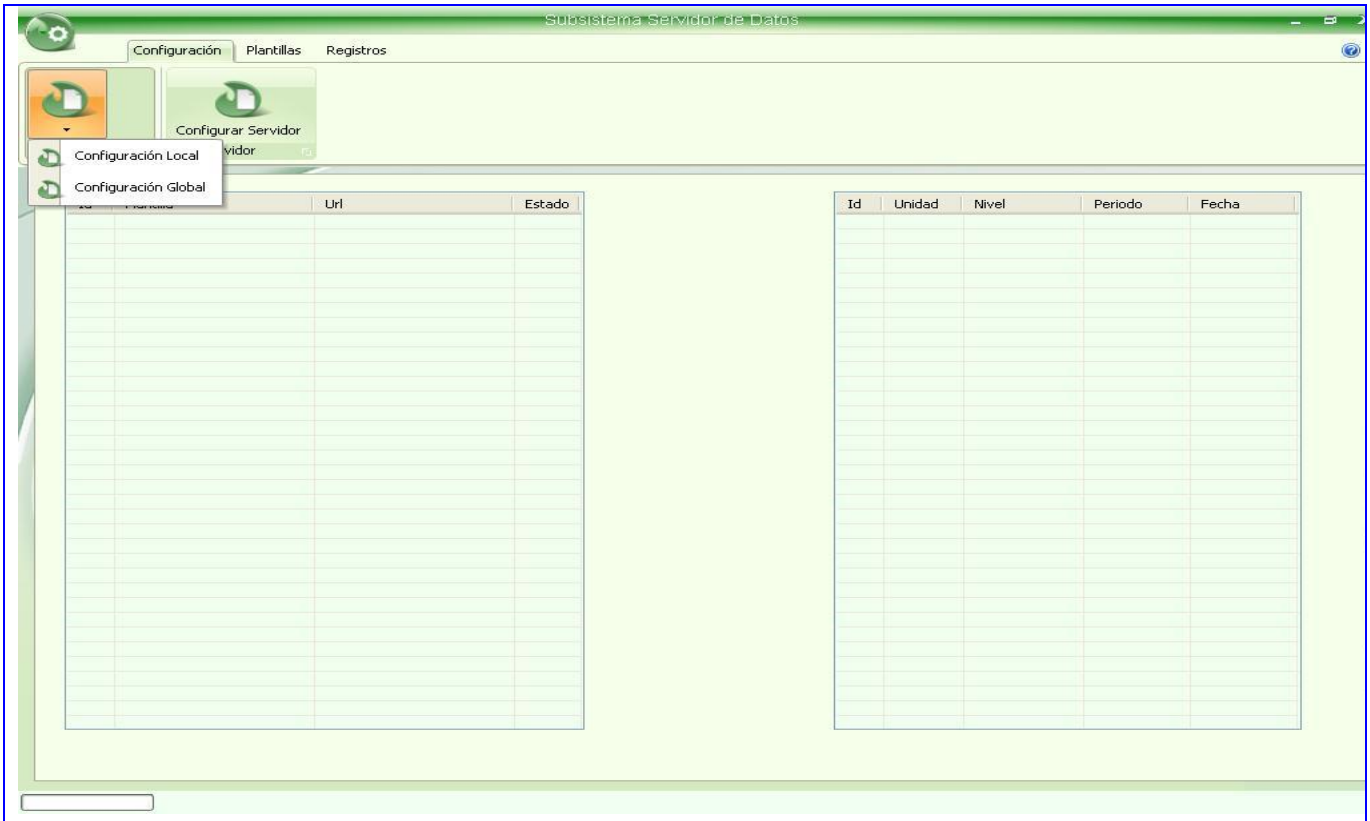
Prototipo de Interfaz



Sección "Acceso a Datos"

3 El Administrador de Configuración selecciona la opción Acceso a Datos en el menú Configuración.	4 El sistema muestra una pestaña, la cual va a brindar la posibilidad de configurar local o globalmente el servidor.

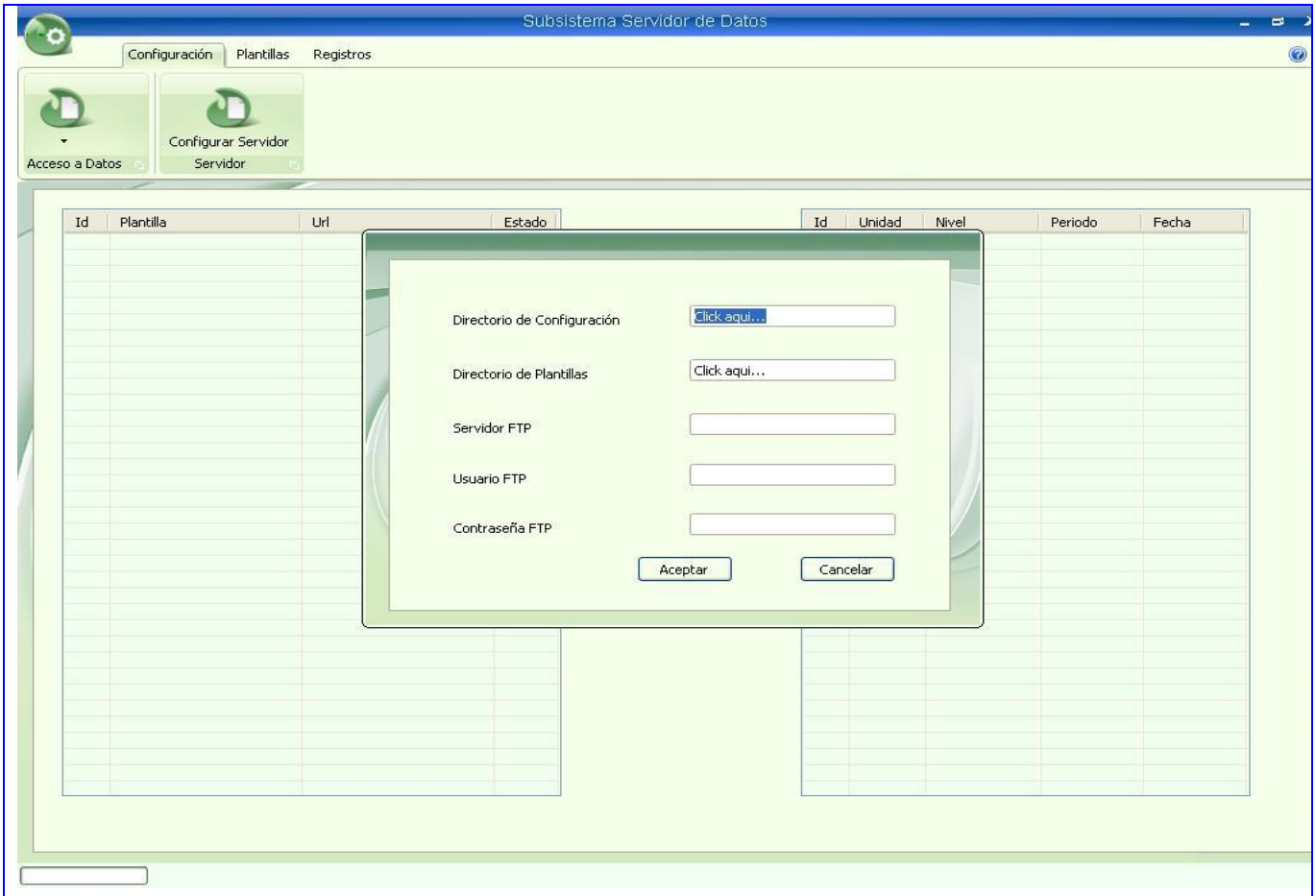
Prototipo de Interfaz



Sección "Configurar Servidor"

3 El Administrador de Configuración selecciona la opción Configurar Servidor en el menú Configuración.	4 El sistema muestra una pantalla, a través de la cual se llenarán una serie de datos para la posterior configuración del servidor.

Prototipo de Interfaz



5 El Administrador de Configuración configura el sistema.	6 El sistema devuelve un mensaje con el éxito de los cambios llevados a cabo.
Pos condiciones	**La configuración del sistema ha sido administrada.**

2.4.3.2 Caso de uso Autenticar.

Caso de Uso:	**Autenticar**
Actores:	**Usuario** **Componente de Seguridad**
Resumen:	El caso de uso se inicia cuando el usuario desea acceder al sistema, para ello debe autenticarse utilizando el componente de seguridad del Área Temática donde debe estar previamente registrado. Finaliza cuando el usuario se autentifique correctamente y el sistema muestre las credenciales del mismo.
Precondiciones:	**El Componente de Seguridad debe estar disponible para realizar el proceso de autenticación.**
Referencias	**RF30, RF41**
Prioridad	**Alta**

Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
1. El usuario inserta credenciales.	2. El sistema le envía al CS la contraseña introducida por el usuario.
	3. El CS devuelve al sistema información sobre ese usuario.
	4. El sistema devuelve las credenciales en caso de tener los permisos, y si no los tiene envía un mensaje de retorno.
Pos condiciones	**El usuario es autenticado y tiene acceso a trabajar en el sistema según el rol que desempeña.**

2.4.3.3 Caso de uso Insertar Planilla.

Caso de Uso:	**Insertar Plantilla**
Actores:	**Subsistema Editor de Plantilla**
Resumen:	El caso de uso se inicia cuando el Subsistema Editor de Plantilla desea insertar una plantilla. El caso de uso finaliza cuando es insertada la misma.
Precondiciones:	**El usuario tiene que estar autenticado.**
Referencias	**RF1, RF2, RF3, RF4, RF5, RF12, RF13**
Prioridad	**Alta**

Flujo Normal de Eventos	
Insertar nueva Plantilla	
Acción del Actor	**Respuesta del Sistema**
1. El Subsistema editor de plantilla solicita Insertar plantilla.	2. Ver CUS_Autenticar 3. El sistema le da la opción de insertar plantilla (Brinda servicios web).
4. El Subsistema editor de plantilla inserta el nombre de la plantilla. 5. El Subsistema editor de plantilla inserta la periodicidad de la plantilla. 6. El Subsistema editor de plantilla inserta la fecha de creación de la plantilla.	7. El sistema guarda la fecha de actualización de la plantilla (Coincide con la de creación en este caso). 8. El sistema notifica que la plantilla fue insertada o no.
Pos condiciones	**La plantilla es insertada en el Servidor de Datos**

2.4.3.4 Caso de uso Actualizar Plantilla.

Caso de Uso:	**Actualizar Plantilla**
Actores:	**Subsistema Editor de Plantilla**
Resumen:	El caso de uso se inicia cuando el actor (Subsistema editor de plantilla) desea buscar una plantilla para actualizarla o modificarla. Esta se puede actualizar debido a cambios que se hicieron o porque la que existía tenía errores. Finaliza cuando se modifica o actualiza la plantilla.
Precondiciones:	**El usuario tiene que estar autenticado.**
Referencias	**RF2, RF3, RF5, RF7**
Prioridad	**Alta**

Flujo Normal de Eventos	
Actualizar Plantilla	
Acción del Actor	**Respuesta del Sistema**
1. El Subsistema editor de plantilla solicita Actualizar plantilla.	2. Ver CUS_Autenticar
	3. El sistema le da la opción de actualizar una plantilla (Brinda servicios web). 4. El sistema busca la información que desea modificar (Plantilla). (Ver CUS_ Buscar plantilla). Si es por error ver **Flujo Alterno 2.**
	5. El sistema bloquea la plantilla (Ver CUS_Bloquear plantilla)
6. El Subsistema editor de plantilla modifica la plantilla.	7. El sistema guarda la fecha de actualización de la plantilla. 8. El sistema desbloquea la plantilla para que esta sea publicada nuevamente.(ver CUS_Desbloquear plantilla)

Flujo Alterno 2	
Acción del Actor	**Respuesta del Sistema**
	4. El sistema busca la información que desea modificar (Plantilla). (Ver CUS_ Buscar plantilla).
	5. El sistema bloquea la plantilla (Ver CU bloquear plantilla).
6. El actor inserta la plantilla (Ver CUS_Insertar Plantilla).	7 El sistema guarda la fecha de actualización de la plantilla (En este caso coincide con la de creación).
Pos condiciones	**Se modifica la plantilla y se actualiza la base de datos.**

2.4.3.5 Caso de uso Buscar Plantilla

Caso de Uso:	**Buscar Plantilla**
Actores:	**Subsistema Editor de Plantilla**
Resumen:	El caso de uso se inicia cuando el usuario necesita actualizar una plantilla determinada. Finaliza el caso de uso cuando se accede a la información solicitada.
Precondiciones:	**El usuario tiene que estar autenticado.**
Referencias	**RF8**
Prioridad	**Alta**
Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra la lista de plantillas

	(Brinda servicios web).
2. Selecciona la plantilla deseada.	3. Muestra el contenido de la plantilla.
Pos condiciones	**El sistema muestra la plantilla seleccionada.**

2.4.3.6 Caso de uso Bloquear Plantilla.

Caso de Uso:	**Bloquear Plantilla**
Actores:	**Subsistema Editor de Plantilla**
Resumen:	El caso de uso inicia cuando el Subsistema editor de plantilla solicita actualizar una plantilla ya sea a la hora de actualizarla o porque alguna de estas está dañada.
Precondiciones:	**El usuario tiene que estar autenticado.**
Referencias	**RF9**
Prioridad	**Alta**

Flujo Normal de Eventos	
Bloquear	
Acción del Actor	**Respuesta del Sistema**
1. El SEP especifica que tipo de bloqueo quiere realizar.	2. El sistema bloquea parcialmente la plantilla. Si el sistema selecciona bloqueo por error. **Ver Flujo alterno1**

Flujo Alterno 1	
Acción del Actor	**Respuesta del Sistema**
	3. El sistema bloquea totalmente la plantilla.
Pos condiciones	**La plantilla es bloqueada**

2.4.3.7 Caso de uso Desbloquear Plantilla.

Caso de Uso:	**Desbloquear Plantilla**
Actores:	**Subsistema editor de plantilla**
Resumen:	El caso de uso inicia cuando el Subsistema editor de plantilla solicita actualizar una plantilla ya sea a la hora de actualizarla o porque alguna de estas está dañada.
Precondiciones:	**El usuario tiene que estar autenticado.**
Referencias	**RF10**
Prioridad	**Alta**

Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
1. El Subsistema editor de plantilla solicita desbloquear una plantilla.	2. El sistema brinda la posibilidad de desbloquear una plantilla (Servicios web).
3. El Subsistema editor de plantilla desbloquea la plantilla (pública), permitiendo que todos los que tienen permisos puedan acceder a ella.	
Pos condiciones	**La plantilla es desbloqueada**

2.4.3.8 Caso de uso Insertar Registro.

Caso de Uso:	**Insertar Registro**
Actores:	**Subsistema para la Captura y Reparo**
Resumen:	El caso de uso se inicia cuando el Subsistema para la captura y reparo desea

	insertar un registro. El caso de uso finaliza cuando es insertado el registro.
Precondiciones:	**El usuario tiene que estar autenticado.**
Referencias	**RF11, RF2, RF3, RF14, RF15, RF16, RF27, RF44**
Prioridad	**Alta**

Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
1 El Subsistema para la captura y reparo solicita insertar un registro.	2. Ver CUS_Autenticar
	3. El sistema le da la opción de insertar un registro.
4. El Subsistema para la captura y reparo inserta el periodo, el identificador de la unidad, la fecha de creado y el nivel de validación.	5. El sistema guarda la fecha de actualización del registro (Coincide con la fecha de creación en este caso). 6. El sistema notifica que el registro fue insertado satisfactoriamente o que no pudo ser insertado.
Pos condiciones	**El registro es insertado en el Servidor de Datos**

2.4.3.9 Caso de uso Actualizar Registro.

Caso de Uso:	**Actualizar Registro**
Actores:	**Subsistema para la Captura y Reparo**
Resumen:	El caso de uso se inicia cuando el Subsistema para la captura y reparo solicita actualizar un registro. Finaliza cuando se modifican los datos.
Precondiciones:	**El usuario debe autenticarse para poder actualizar el registro.**

Referencias	**RF18, RF2, RF3, RF44, RF24**
Prioridad	**Alta**
Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
1. El Subsistema para la captura y reparo solicita Actualizar registro.	2. Ver CUS_Autenticar
	3. El sistema le da la opción de actualizar una plantilla (Brinda servicios web). 4. El sistema busca la información que desea modificar (Plantilla). (Ver CUS_ Buscar plantilla).
4. El Subsistema para la captura y reparo modifica la información.	5. El sistema guarda fecha de actualización del registro. 6. El sistema muestra el registro con la información actualizada.
Pos condiciones	**Se modifica el registro y a su vez actualiza la Base de Datos.**

2.4.3.10 Caso de uso Buscar Registro.

Caso de Uso:	**Buscar Registro**
Actores:	**Subsistema para la captura y reparo**
Resumen:	El caso de uso se inicia cuando el Subsistema para la captura y reparo necesita modificar o actualizar un registro. Finaliza cuando se accede a la información solicitada.
Precondiciones:	**El usuario debe autenticarse para poder actualizar el registro.**
Referencias	**RF19, RF20, RF21, RF22, RF23**

Prioridad	**Alta**
Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
1. El Subsistema para la captura y reparo solicita buscar un registro.	2. El sistema muestra el formulario con criterios de búsqueda.
3. El Subsistema para la captura y reparo introduce los criterios de búsqueda.	4. Muestra la información. Si no muestra ver **Flujos Alternos 1**
Flujos Alternos 1	
Acción del Actor	**Respuesta del Sistema**
	3. Muestra un mensaje informando que la información no esta disponible
Pos condiciones	**El registro es encontrado y actualizado.**

2.4.3.11 Caso de uso Construir Reporte.

Caso de Uso:	**Construir Reportes**
Actores:	**Subsistema para la captura y reparo**
Resumen:	El caso de uso se inicia cuando el subsistema para la captura y reparo desea construir un reporte de los registros ya existentes. El caso de uso finaliza cuando el reporte es construido.
Precondiciones:	**El usuario debe estar autenticado**
Referencias	**RF24**

Prioridad	**Alta**
Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
1. El Subsistema para la captura y reparo solicita al sistema construir un reporte.	2. Ver CUS_Autenticar
	3. El sistema verifica si existe algún registro. Si no existe ningún registro ver **(Flujo Alterno 2).**
	4. Se construye el reporte y lo envía al actor.
Flujo Alterno 2	
Acción del Actor	**Respuesta del Sistema**
	3. El sistema devuelve un mensaje de retorno especificando que no existe ningún registro.
Pos condiciones	**El reporte es construido.**

2.4.3.12 Caso de uso Construir Consolidado.

Caso de Uso:	**Construir Consolidado**
Actores:	**Subsistema para la Captura y Reparo**
Resumen:	El caso de uso se inicia cuando el Subsistema para la captura y reparo solicita elaborar el modelo consolidado de las unidades de salud en un período determinado. El caso de uso finaliza cuando el consolidado es construido.
Precondiciones:	**El usuario debe estar autenticado**
Referencias	**RF25, RF26, RF42**
Prioridad	**Alta**

Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
1. Solicita se elabore el modelo consolidado de las unidades de salud.	2. Ver CUS_Autenticar.
	3. El sistema verifica si existe el registro o los registros de los cuales se quiere generar el modelo consolidado.Si no existe el registro ver **(Flujo Alterno 1).**
	4 El sistema verifica el nivel de acceso. 5 El sistema elabora y envía el consolidado al nivel superior.
Flujo Alterno 1	
Acción del Actor	**Respuesta del Sistema**
	4. El sistema devuelve un mensaje de retorno con los problemas que han ocurrido.
Pos condiciones	**Es construido el modelo consolidado de las unidades de salud.**

2.4.3.13 Caso de uso Validar Registro.

Caso de Uso:	**Validar Registro**
Actores:	**Registro de Unidad de Salud**
Resumen:	El caso de uso inicia cuando el Registro de Unidad de Salud procede a examinar el modelo que muestra el sistema. En caso de encontrar un dato fuera de los valores normales y no se especifica el por qué o si el modelo presenta otro problema, podrá rechazarlo. Finaliza el caso de uso cuando se

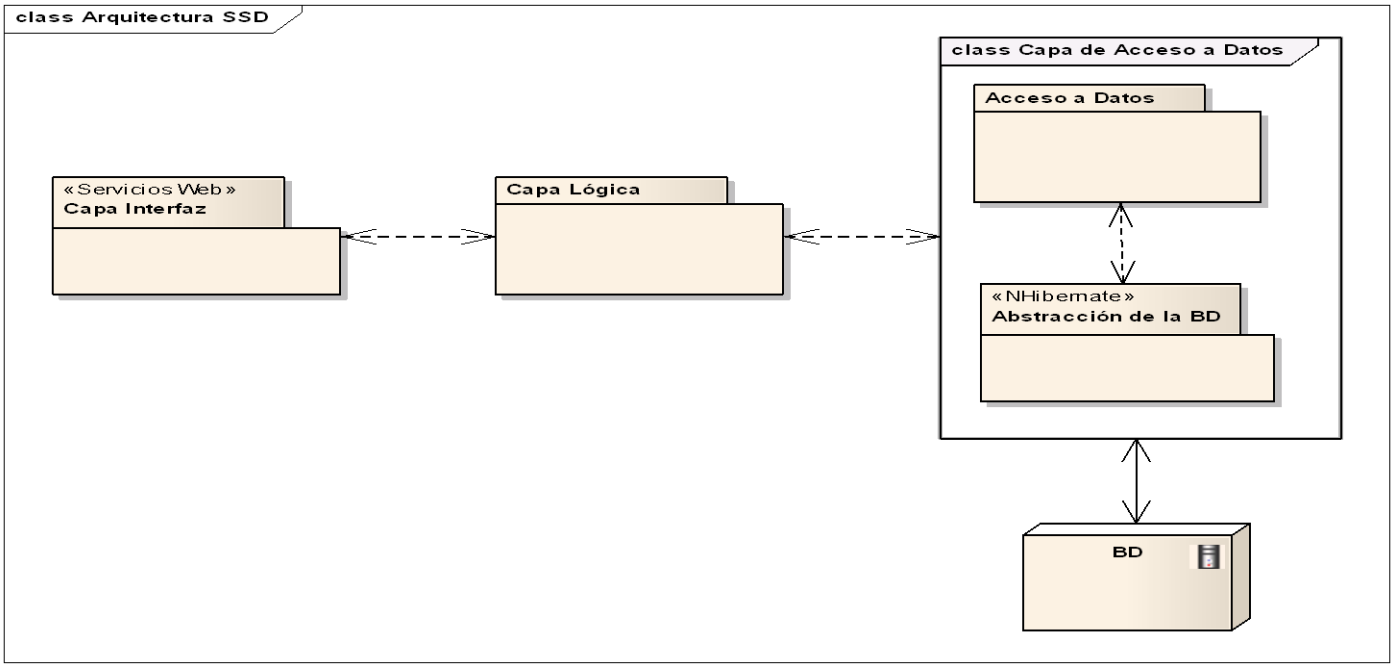
	rechaza o acepta la información.
Precondiciones:	**El usuario debe estar autenticado.**
Referencias	**RF25, RF28, RF29**
Prioridad	**Critica**
Flujo Normal de Eventos	
Sección “”	
Acción del Actor	**Respuesta del Sistema**
1. El Registro de Unidades de Salud te brinda la opción Validar registro.	2 El sistema muestra el formulario con la información a validar.
3. El Registro de Unidades de Salud acepta o rechaza los datos.	4 El sistema actualiza la base de datos.
Pos condiciones	**Actualización de la Base de Datos.**

En el capítulo se determinaron los conceptos y roles que intervienen en la gestión de la información, así como los requisitos funcionales y no funcionales, a partir de entrevistas con los clientes. Fueron descritos los flujos de eventos asociados a los casos de uso definidos. Finalmente, quedaron sentadas las bases para el diseño e implementación del sistema propuesto.

Capitulo3: Diseño del Sistema

En este capítulo se describen las clases del diseño y las relaciones entre ellas para cada uno de los casos de uso con que cuenta el subsistema. También se ofrece en diagrama con las entidades fundamentales del modelo de datos y una breve descripción de las tablas que conforman la base de datos.

3.1 Estructura del diseño



Capa de presentación:

La capa de presentación está desarrollada con la utilización de XML WebServices**.** Esta es la capa que conoce cuáles funciones de la capa de negocio responden a cualquiera de los eventos desencadenados por el usuario posibilitando así un correcto intercambio entre el sistema y cliente.

Herramientas utilizadas en la Capa de Presentación:

Diseño	SharpDevelop 1.0

Capa de negocio:

Establece la comunicación entre la capa de presentación y la capa de acceso a datos, encargada de recibir y responder cada petición de los usuarios. Los ficheros que la conforman reciben las solicitudes de los clientes, se comunican con la capa de acceso a datos, actualizando o recuperando información y emitiendo una respuesta. Constituye la parte del sistema donde se establecen todas las reglas de negocio que deben cumplirse. La capa de negocio esta implementada con la utilización de mono Frameworks, que es multiplataforma y además posibilita la implementación del sistema con la utilización de librerías, de un marco de trabajo y desarrollos más rápidos.

Herramientas utilizadas en la capa de negocio:

Implementación	SharpDevelop 1.0

Capa de acceso a datos:

La capa de acceso a datos está implementada con la utilización de la librería NHibernate, que es un motor de persistencia, que permite tratar las tablas de la base de datos como si fueran objetos y además abstrae al sistema de la utilización de un Gestor de Base de Datos.

Herramientas utilizadas en la capa de acceso a datos:

Implementación	SharpDevelop 1.0
Librería	NHibernate

Capa de datos:

El servidor central utiliza como Gestor de Base de Datos PostgreSQL 8.3.1, que realiza todo el almacenamiento de datos, recibe solicitudes de almacenamiento o recuperación de información desde la capa de acceso a datos. En el caso de las PC clientes se utiliza PostgreSQL 8.3.1 cuando se necesita más de una PC en una unidad de salud. Cuando se necesita una única PC en una unidad de salud se utiliza SQLite.

Herramientas utilizadas en la capa de datos:

Gestores de BD	PostgreSQL 8.3.1 y SQLite

3.2 Modelo de diseño

El Modelo de Diseño es otro de los tantos modelos que se tienen en cuenta en el ciclo de vida del desarrollo de un software, consumiendo una elevada cantidad de esfuerzo para su realización. Este modelo es de vital importancia para los especialistas y desarrolladores ya que pone al relieve una solución lógica. Además define e identifica las consecuencias del ambiente de implementación. Es un modelo físico, no genérico y específico para una implementación, más formal, dinámico, que contiene un sinnúmero de estereotipos físicos sobre las clases en dependencia del lenguaje utilizado y ha sido creado principalmente como programación visual en procesos iterativos e incrementales.

En concreto, el diseño tiene el propósito de formar la base para la implementación, desarrollar la arquitectura, adaptar el diseño para que sea consistente con el entorno de implementación, dándole soporte a todos los requisitos, incluyendo los no funcionales y las restricciones que se le suponen, así como descomponer los trabajos a desarrollar en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo, teniendo en cuenta la posible concurrencia.

3.2.1 Diagramas de clases del diseño.

El diagrama de clase es el diagrama principal del diseño para un sistema, elaborado para satisfacer los detalles concretos de la implementación. En él se muestra la estructura estática del sistema, es donde se pueden ver las relaciones entre todas las clases, además de los atributos y operaciones de cada una de las mismas.

Los diagramas de clases del diseño describen gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación. Normalmente contienen la información de: las clases, asociaciones y atributos, interfaces, con sus operaciones y constantes, métodos, información sobre los tipos de los atributos, navegabilidad y dependencia. A diferencia del modelo conceptual, un diagrama de este tipo contiene las definiciones de las entidades del software en vez de conceptos del mundo real. Seguidamente se muestra algunas de las representaciones de los mismos:

Se realizó un diagrama de diseño para cada realización de caso de uso que se va a implementar, los mismos se muestran a continuación:

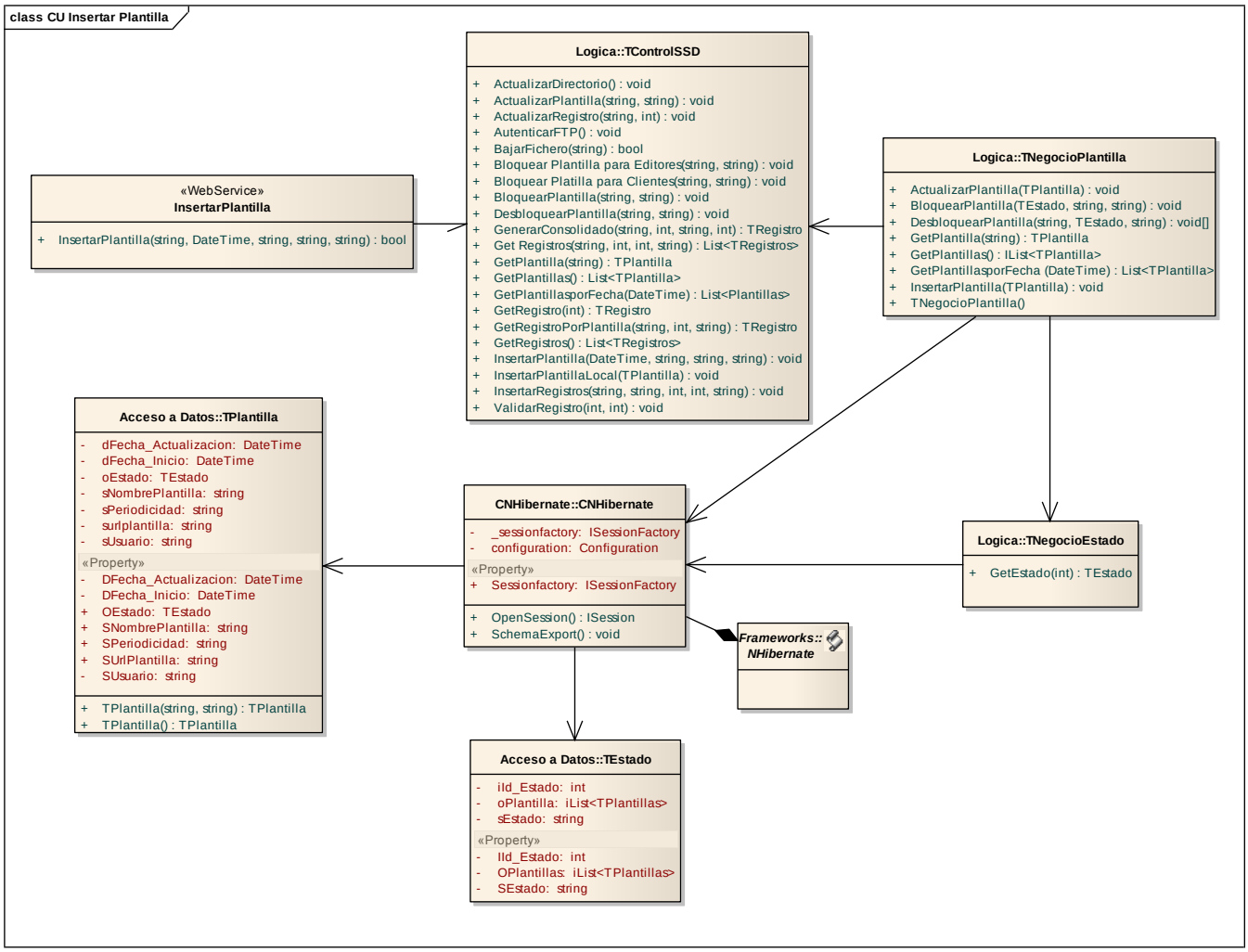


Figura 4: Diagrama de clases del diseño para el Caso de Uso Insertar Plantilla.

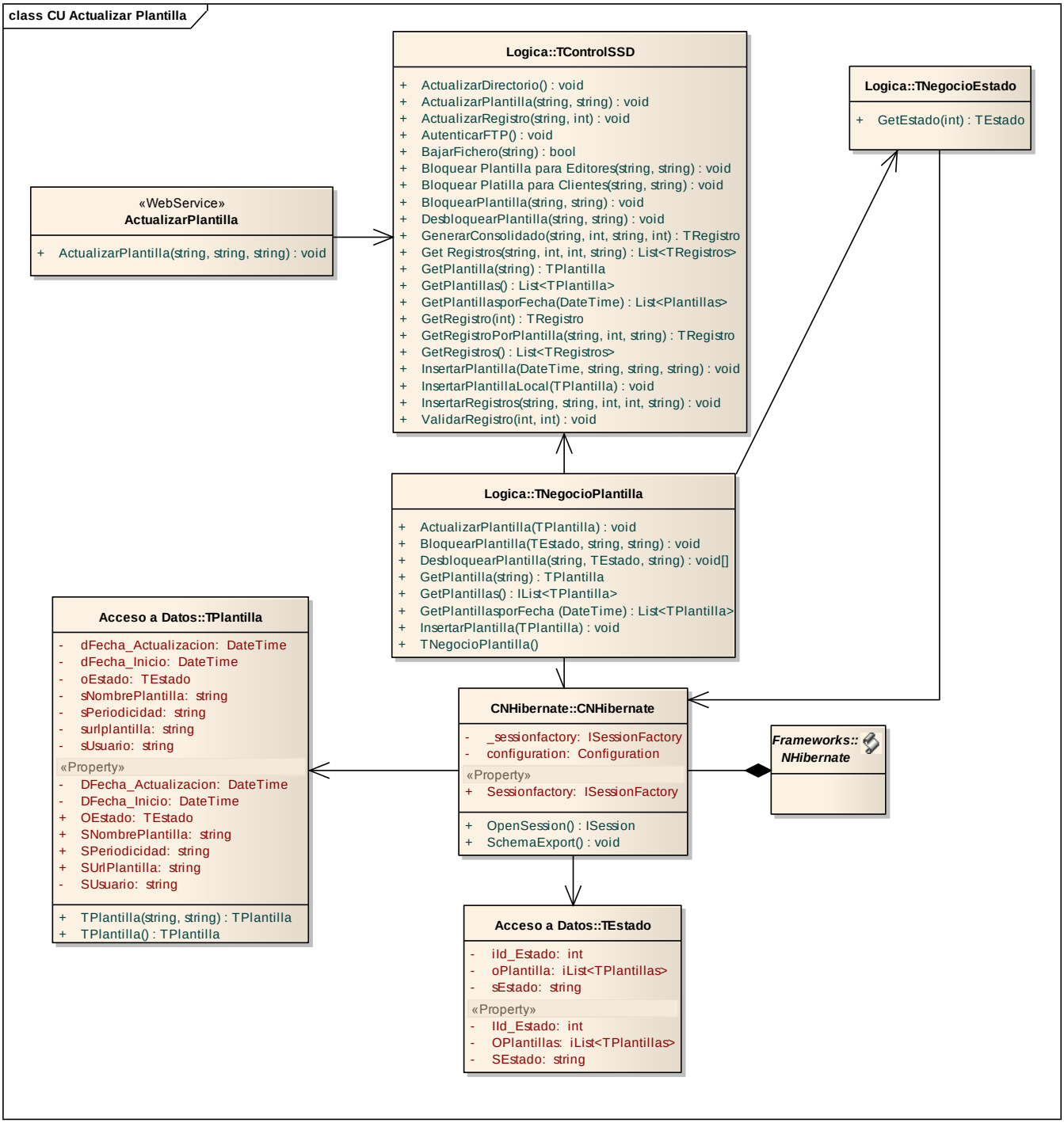


Figura 5: Diagrama de clases del diseño para el Caso de Uso Actualizar Plantilla.

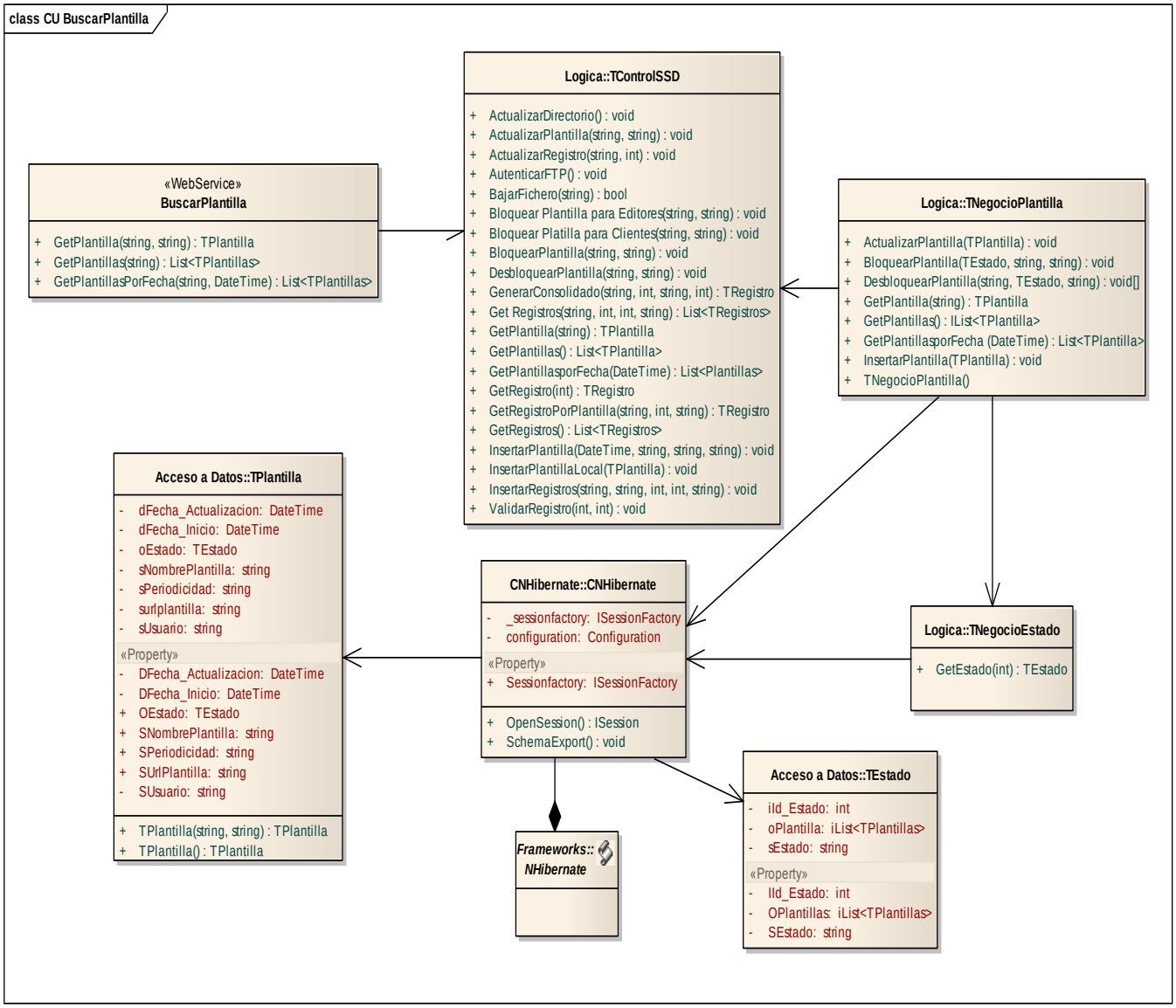


Figura 6: Diagrama de clases del diseño para el Caso de Uso Buscar Plantilla.

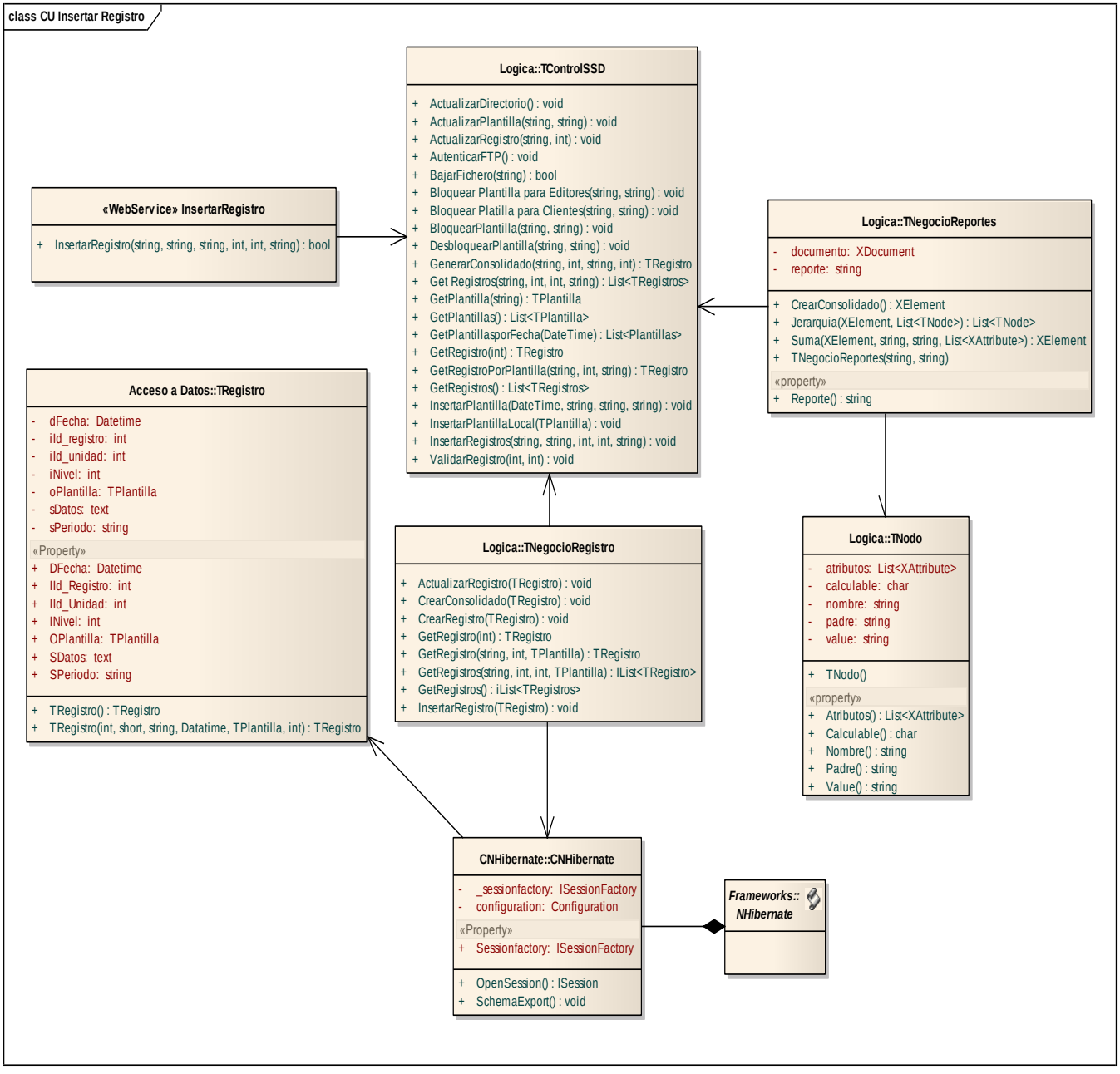


Figura 7: Diagrama de clases del diseño para el Caso de Uso Insertar Registro.

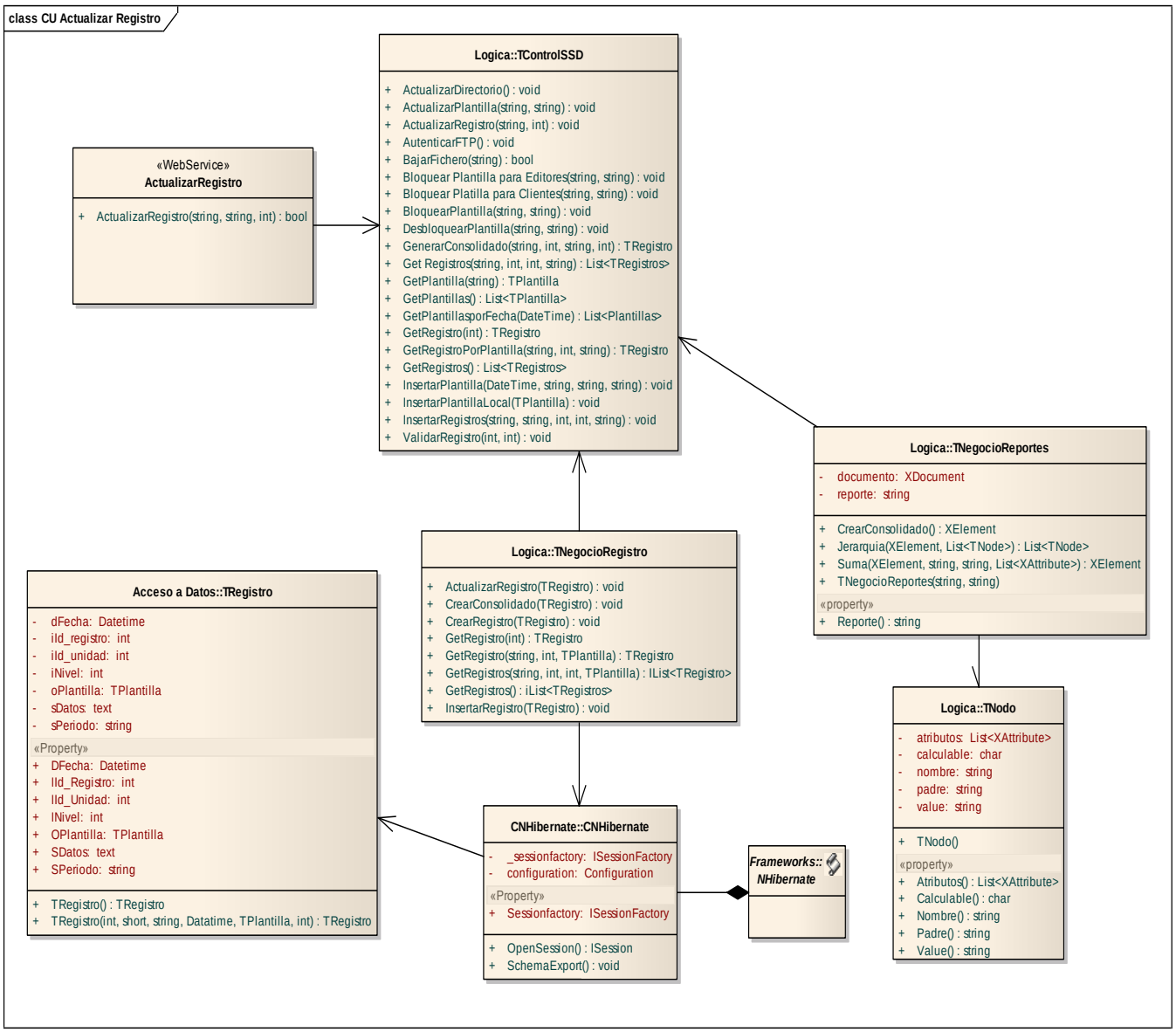


Figura 8: Diagrama de clases del diseño para el Caso de Uso Actualizar Registro.

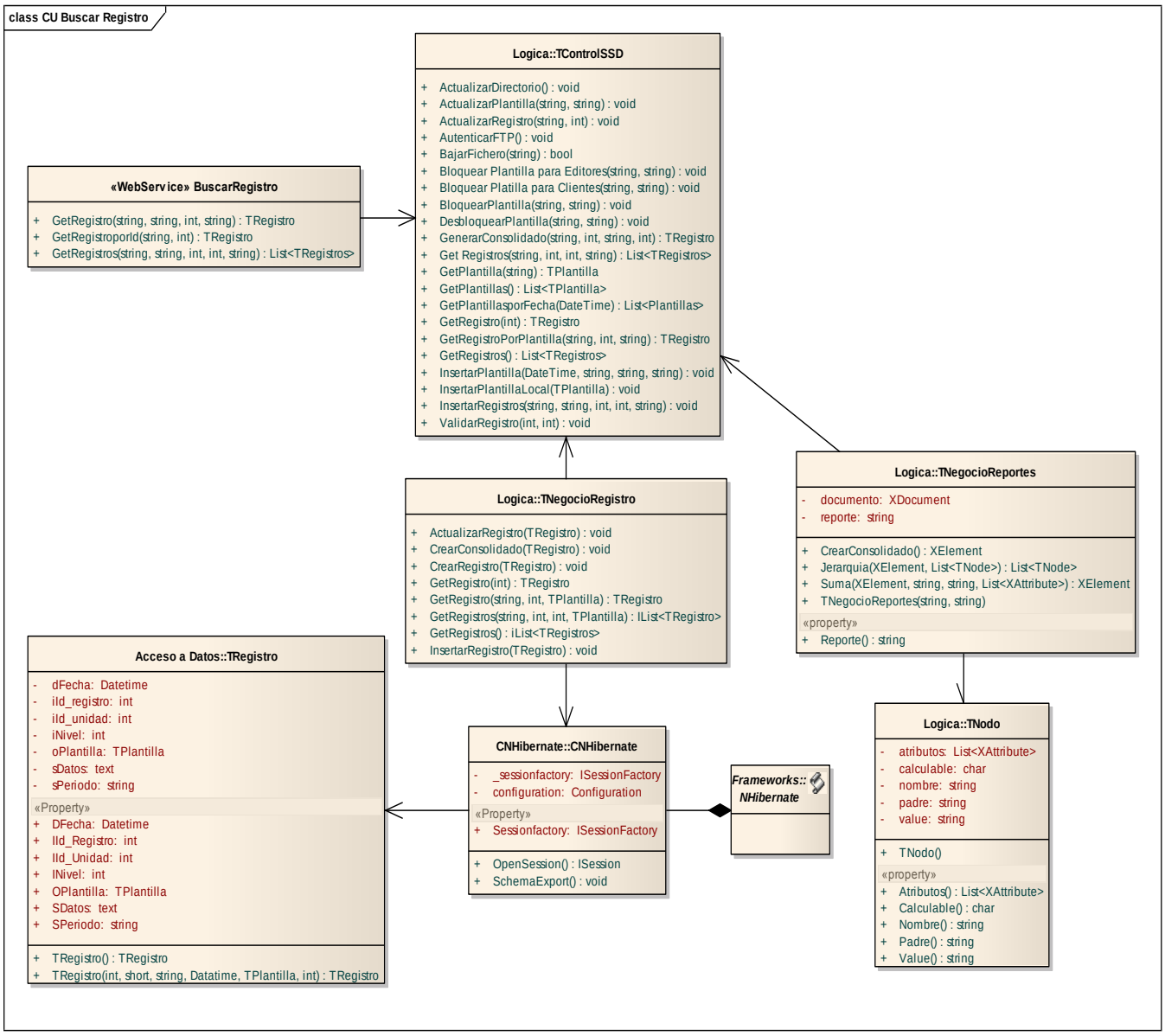


Figura 9: Diagrama de clases del diseño para el Caso de Uso Buscar Registro.

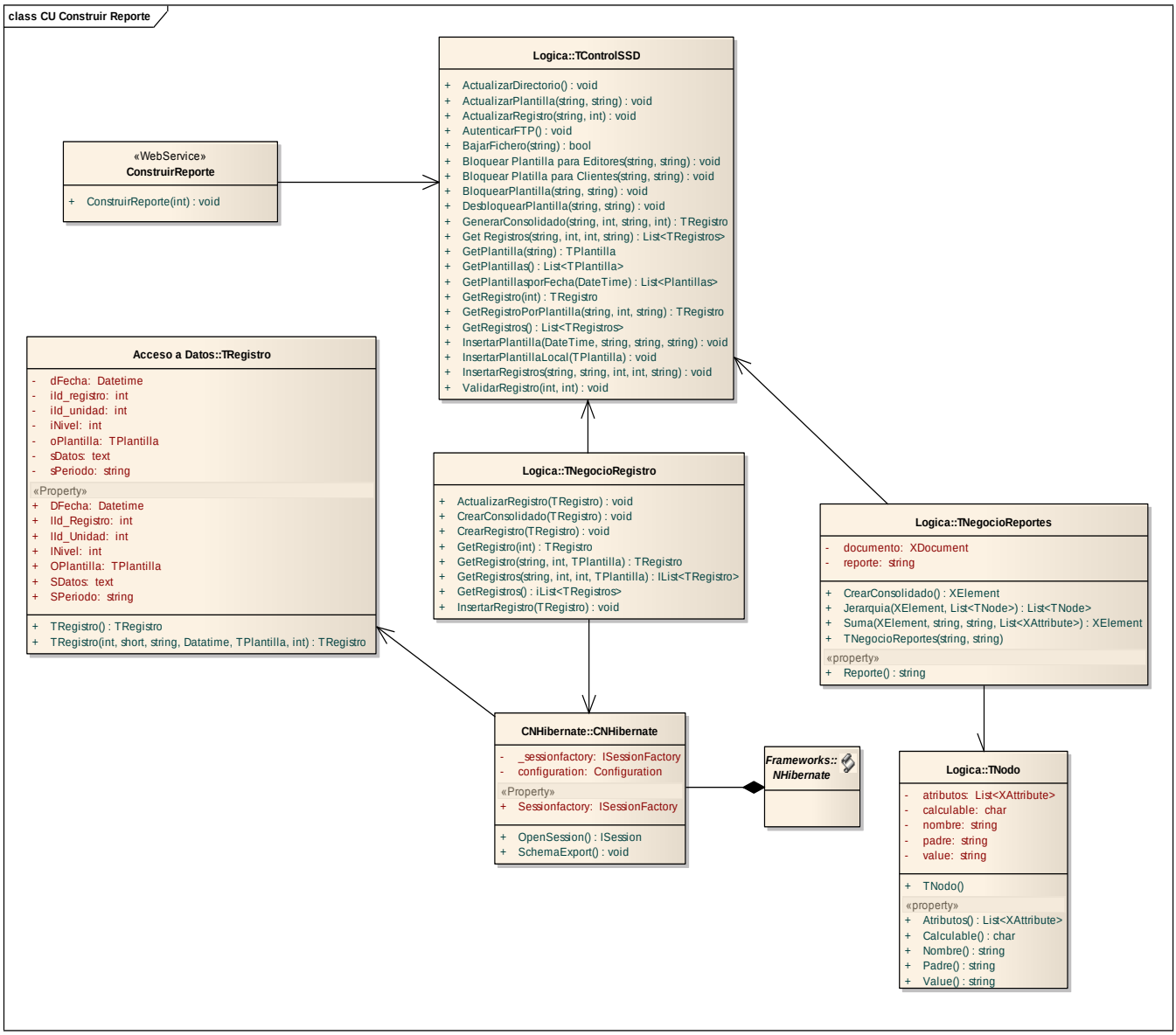


Figura 10: Diagrama de clases del diseño para el Caso de Uso Construir Reporte.

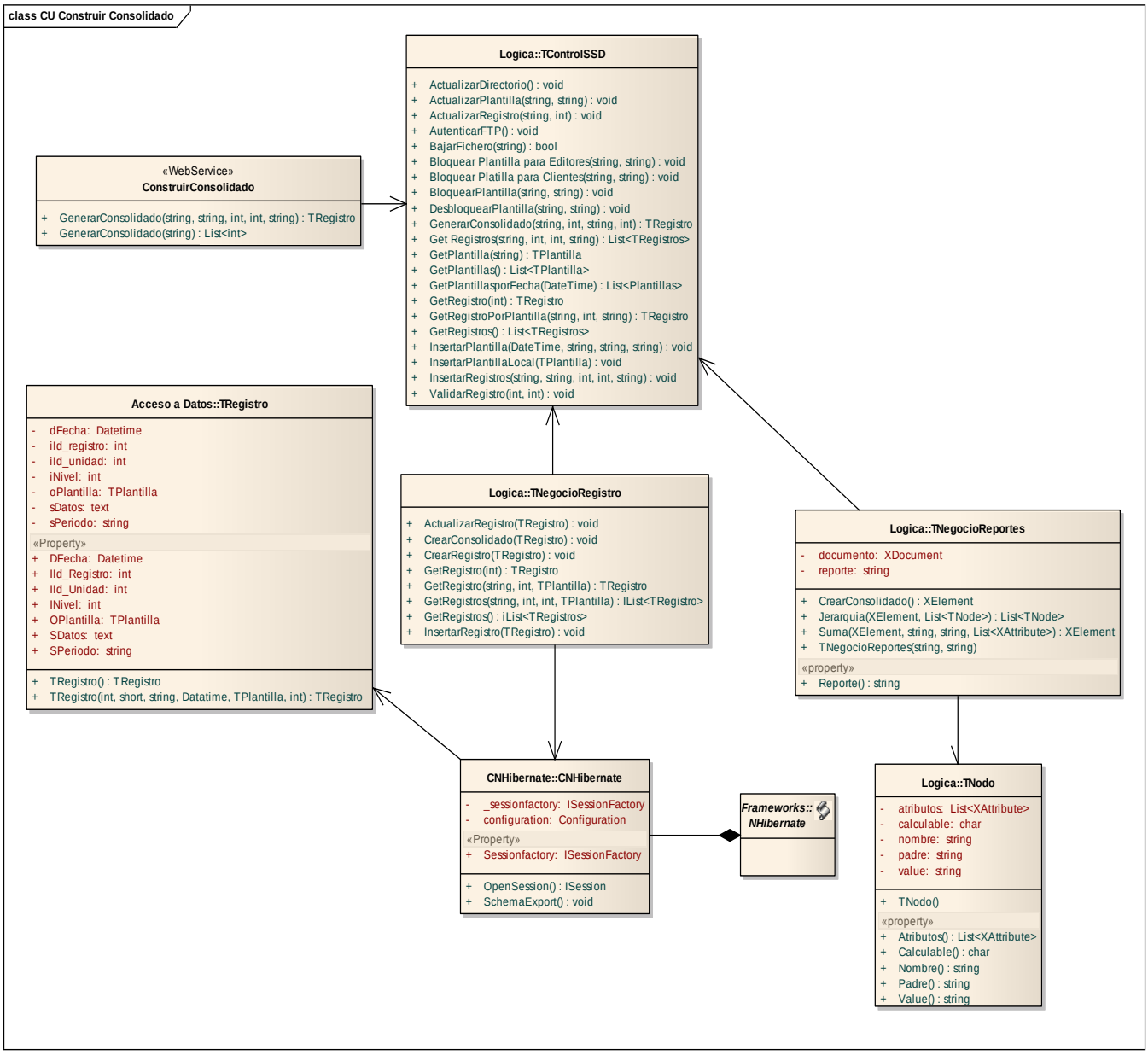


Figura 11: Diagrama de clases del diseño para el Caso de Uso Construir Consolidado.

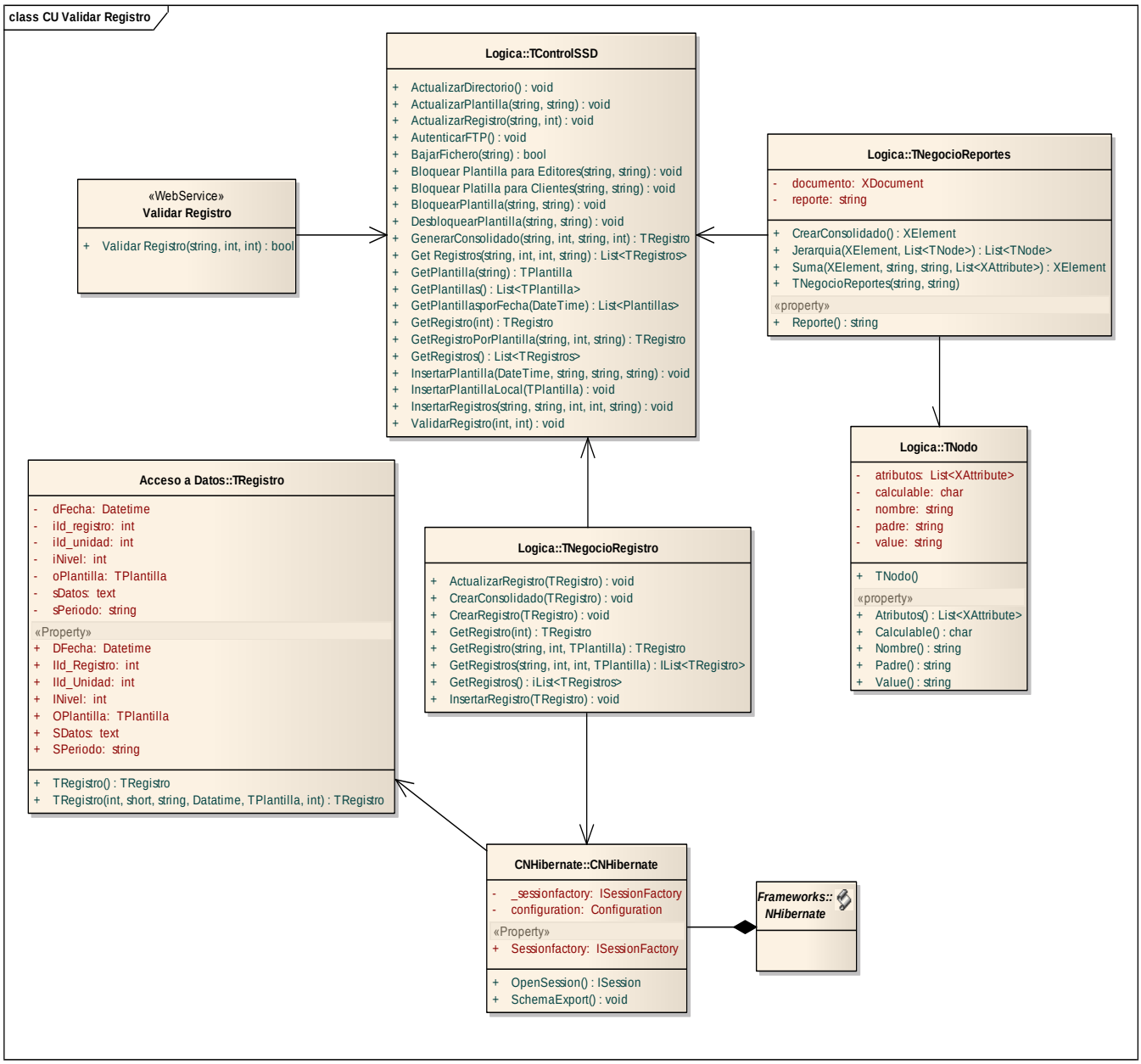


Figura 12: Diagrama de clases del diseño para el Caso de Uso Validar Registro.

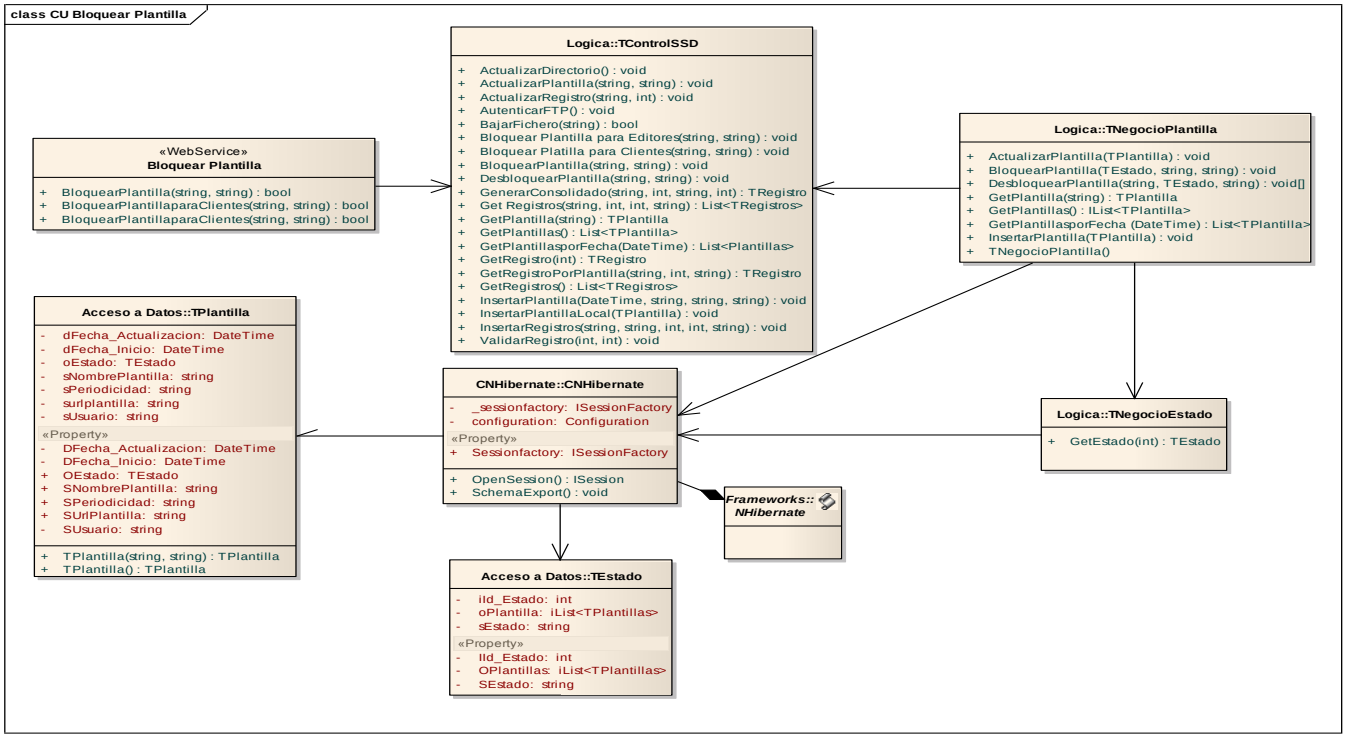


Figura 13: Diagrama de clases del diseño para el Caso de Uso Bloquear Plantilla.

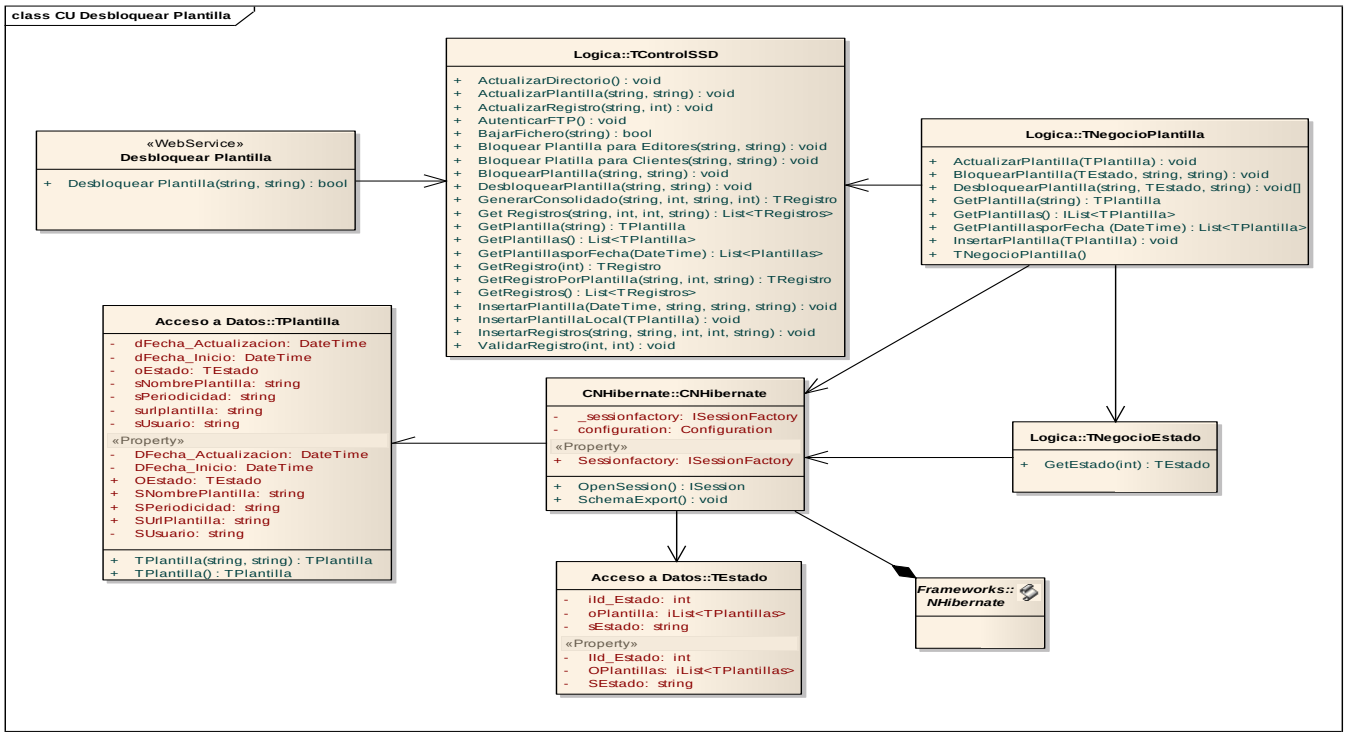


Figura 14: Diagrama de clases del diseño para el Caso de Uso Desbloquear Plantilla.

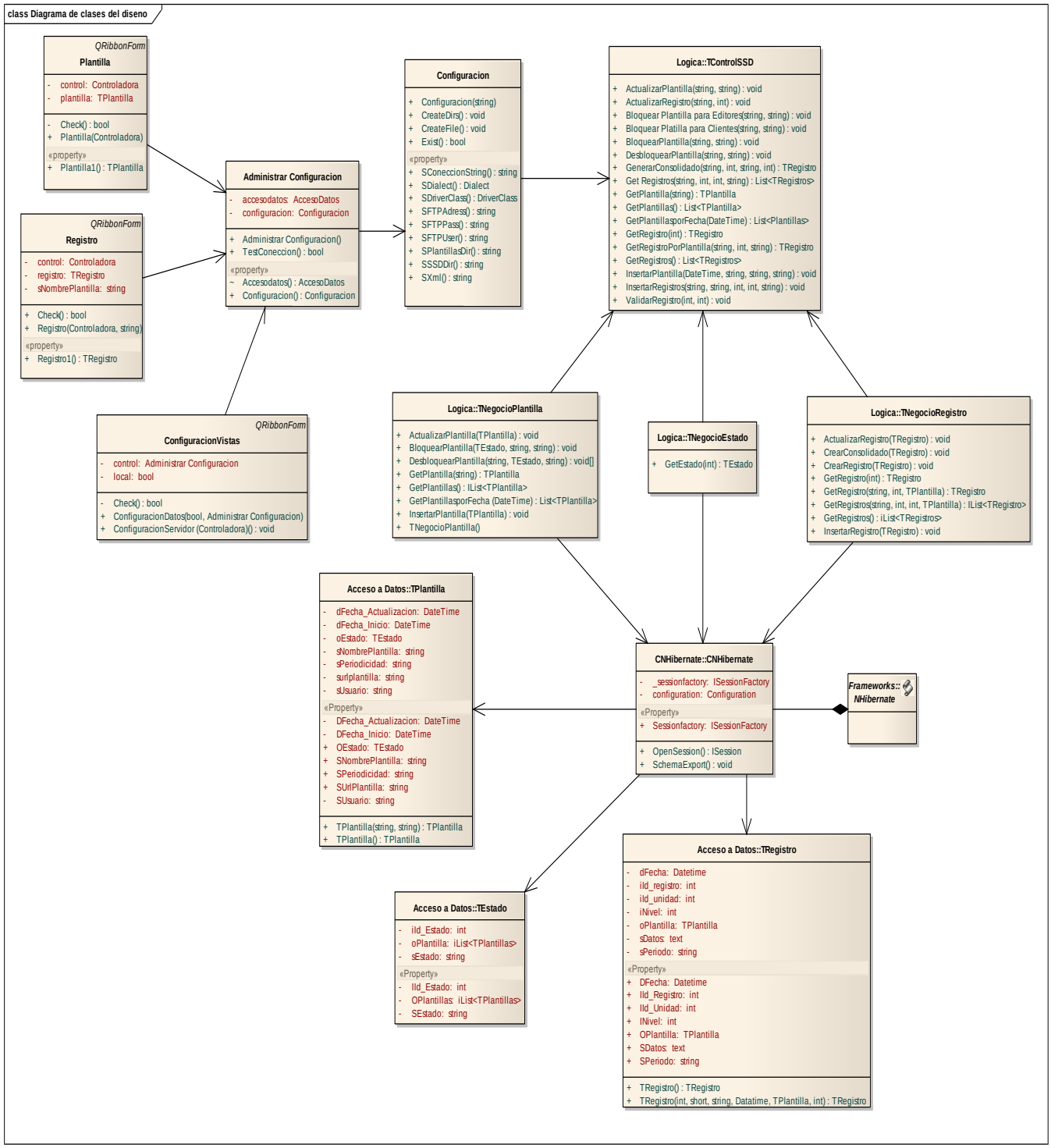


Figura 15: Diagrama de clases del diseño para el Caso de Uso Administrar Configuración.

3.2.2 Descripción de las Clases.

Nombre: TPlantilla
Acceso a Datos
Descripción: Es la clase que contiene los datos de los diferentes Esquemas y a la cual se accede cada vez que se necesite obtener determinada información referente a los Esquemas.

Nombre: TRegistro
Capa: Acceso a datos
Descripción: Es la clase que contiene los datos de los diferentes Registros. A la misma se accede cada vez que se necesite obtener determinada información referente a los Registros.

Nombre: TEstado
Capa: Acceso a datos
Descripción: Es la clase que contiene el estado de cada una de las diferentes Plantilla. A la misma se accede cada vez que se necesite obtener el estado en que se encuentra una plantilla determinada.

Nombre: CNhibernate
Capa: Acceso a datos
Descripción: Clase que encapsula la conexión al gestor de base de datos a través de NHibernate y abre una sesión por la cual se realizan las transacciones a la base de datos.

Nombre: Frameworks:: NHibernate
Capa: Acceso a datos
Descripción: Es una clase generada por la librería NHibernate para el mapeo de objetos relacionales y como su nombre lo indica, tiene como función principal mapear los objetos desde una aplicación .Net a una base de datos Relacional.

Nombre: TNegocioPlantilla
Capa: Lógica o de negocio
Descripción: La clase TNegocioPlantilla establece la comunicación entre las diferentes clases que se relacionan con las Plantillas, también son las encargadas de recibir y responder cada petición de los usuarios.

Nombre: TNegocioRegistro
Capa: Lógica o de negocio
Descripción: La clase TNegocioRegistro establece la comunicación entre las diferentes clases que se relacionan con los Registros. Estas también son las encargadas de recibir y responder cada petición de los usuarios.

Nombre: TNegocioEstado
Capa: Lógica o de negocio
Descripción: La clase TNegocioEstado es la encargada de establecer la comunicación entre las plantillas y el estado en que se encuentran estas.

Nombre: Insertar Plantilla
Capa: Interfaz
Descripción: Clase que mediante WebServices le brinda la posibilidad al usuario (Subsistema Editor de plantilla) de insertar una plantilla en el Servidor de Datos.

Nombre: Actualizar Plantilla
Capa: Interfaz
Descripción: Clase que mediante WebServices le brinda la posibilidad al usuario (Subsistema Editor de plantilla) de Actualizar una Plantilla, ya sea porque la misma cambio o porque se encuentra dañada.

Nombre: Buscar Plantilla
Capa: Interfaz
Descripción: Clase que mediante WebServices le brinda la posibilidad al usuario (Subsistema Editor de plantilla) de Buscar una Plantilla en el Servidor de Datos.

Nombre: Bloquear Plantilla
Capa: Interfaz
Descripción: Clase que mediante WebServices le brinda la posibilidad al usuario (Subsistema Editor de plantilla) de Bloquear una plantilla mientras trabaja con ella, para que otro usuario que tenga acceso a las mismas no pueda verla ni tenga acceso a ella.

Nombre: Desbloquear Plantilla
Capa: Interfaz
Descripción: Clase que mediante WebServices le brinda la posibilidad al usuario (Subsistema Editor de plantilla) de Desbloquear una plantilla después de haberla Bloqueado.

Nombre: Insertar Registro
Capa: Interfaz
Descripción: Clase que mediante WebServices le brinda la posibilidad al usuario (Subsistema para la Captura y Reparo) de insertar un Registro en el Servidor de Datos.

Nombre: Actualizar Registro
Capa: Interfaz
Descripción: Clase que mediante WebServices le brinda la posibilidad al usuario (Subsistema para la Captura y Reparo) de Actualizar un Registro.

Nombre: Buscar Registro
Capa: Interfaz
Descripción: Clase que mediante WebServices le brinda la posibilidad al usuario (Subsistema para la Captura y Reparo) de Buscar un Registro en el Servidor de Datos.

Nombre: Crear Reporte
Capa: Interfaz
Descripción: Clase que permite crear reportes de los diferentes registros creados por el Subsistema para la Captura y Reparo.

Nombre: Construir Consolidado
Capa: Interfaz
Descripción: Clase que permite generar un modelo consolidado de cada uno de los registros que se hayan creado o que se quieran crear. El modelo consolidado es un tipo de reporte.

Nombre: Validar Registro
Capa: Interfaz
Descripción: Clase que permite validar los registro que se vayan creando o ya hayan sido creados, para que no contengan errores y cumplan con las medidas que se le exigen a cada uno de ellos (Registros).

Nombre: Plantilla
Capa: Interfaz
Descripción: Clase que muestra las funcionalidades que puede realizar el administrador de configuración con las plantillas.

Nombre: Registro
Capa: Interfaz
Descripción: Clase que muestra las funcionalidades que puede realizar el administrador de configuración con los registros.

Nombre: ConfiguracionVistas
Capa: Interfaz
Descripción: Clase que muestra las posibles configuraciones que puede realizar el administrador de configuración sobre el servidor de datos.

Nombre: Administrar Configuración
Capa: Lógica o de Negocio
Descripción: Clase que controla todo lo referente a la administración de configuración del servidor. Esta hace llamada a métodos que se encuentran en la clase Configuración.

Nombre: Configuración
Capa: Lógica o de Negocio
Descripción: Clase que crea la configuración que se le dará al servidor de datos.

Nombre: TNodo
Capa: Lógica o de Negocio
Descripción: Clase en la que se declaran atributos que serán usados en la clase TNReportes para conocer la jerarquía de un documento XML y posteriormente crear el modelo consolidado.

Nombre: TNegocioReportes
Capa: Lógica o de Negocio
Descripción: La clase TNegocioReportes es la clase que contiene y maneja toda la información referente a los reportes creados sobre los diferentes registros existentes en la base de datos. Su función fundamental es generar los modelos consolidados y algún otro reporte especifico que se quiera hacer.

3.3 Diagramas de interacción.

En ellos, se expresan las interacciones que existen entre las clases de un caso de uso determinado, especificándolas en una secuencia lógica. Además se especifican mediante qué métodos se relacionan las clases, los parámetros de entrada que necesitan y las respuestas que se obtienen de cada interacción. Esto facilita el entendimiento del funcionamiento del sistema y el versionado del mismo posteriormente. Los diagramas de interacción pertenecientes a los casos de usos

3.4 Diseño de la Base de Datos.

3.4.1 Modelo de Datos.

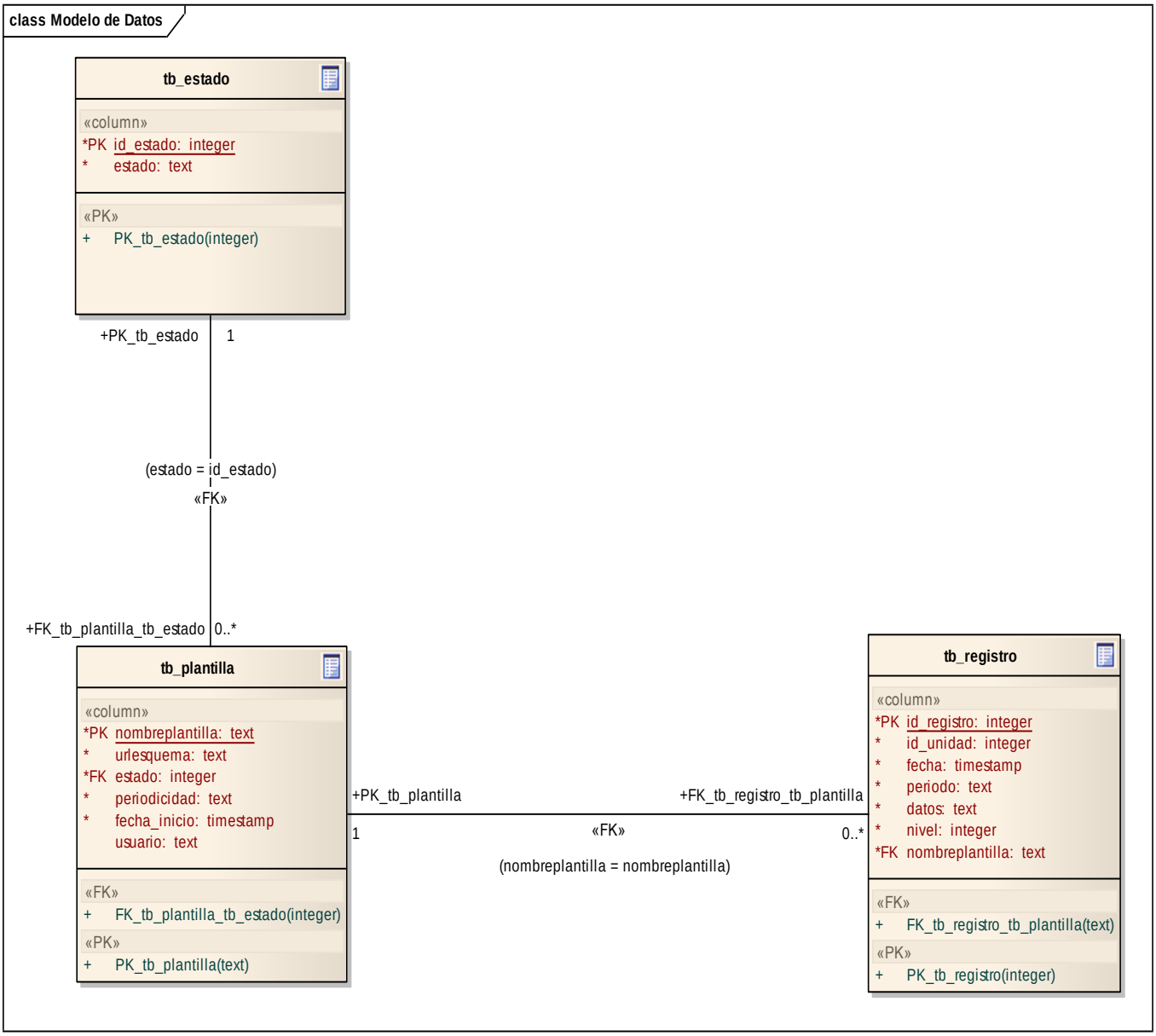


Figura 16: Modelo de datos.

3.4.2 Descripción de las tablas.

Nombre: tb_plantilla		
Descripcion: Contiene los datos principales de cada una de las diferentes plantillas que se generan, como son el nombre, la dirección, el estado en que se encuentra y la fecha en que fue creada la plantilla.		
Atributos	**Tipo**	**Descripción**
nombreplantilla	**text**	Nombre que tendrán cada una de las plantillas. Este atributo será el identificador de las plantillas.
urlesquema	**text**	Contiene la dirección donde se encuentra la plantilla.
estado	**int**	Atributo que contiene el estado en que se encuentra la plantilla. Esta pueda estar bloqueada, pública, bloqueada para editores y bloqueada para clientes.
periodicidad	**text**	Atributo que guarda la periodicidad con que se actualiza una plantilla.
fecha de inicio	**timestamp**	Atributo que guarda la fecha en que fue insertada la plantilla por primera vez en la base de datos.
usuario	**text**	Atributo que contiene el usuario que ha bloqueado la plantilla en algún momento.

Nombre: tb_registro		
Descripcion: Contiene los datos principales de cada uno de los registros que se crean, como son el id, la unidad a la que pertenece, la fecha en que fue confeccionado, el periodo, los datos, el nivel al que pertenece y el esquema al cual se le aplicó dicho registro.		
Atributos	**Tipo**	**Descripción**
id_registro	**int**	Número de identificación que va a tener cada uno de los registros.
Id_unidad	**int**	Número de identificación de la unidad a la que pertenece el registro.
fecha	**timestamp**	Fecha en la que fue creado el registro.
periodo	**varchar**	Período en el cual fue creado el registro.
datos	**text**	Datos relacionados con los registros.
nivel	**int**	Nivel al que pertenece el registro.
nombreplantilla	**int**	Atributo que contiene la plantilla a partir del cual se crea o modifica un registro determinado.

Nombre: tb_estado		
Descripcion: Tabla que contiene el estado en que se encuentran cada una de las plantillas existentes en la base de datos.		
Atributos	**Tipo**	**Descripción**
id_estado	**int**	Número de identificacion que va a tener el estado de cada una de las plantillas existentes en la base de datos.
estado	**text**	Representa el estado en que se encuentra un aplantilla en el servidor de datos. Estas pueden estar bloqueadas, públicas, bloqueadas por editores y bloqueadas por clientes.

En este capítulo se relacionaron las descripciones de los casos de uso fundamentales del sistema. Así como de los diagramas de clases e interacción (secuencia) para la representación gráfica de cada uno de ellos (casos de uso). Se muestra también como queda conformado el modelo de datos, se especifican en cada caso las características de las tablas que lo conforman y la responsabilidad que tiene cada una de ellas. Los atributos que tienen los tipos de datos que guardan así como las relaciones y dependencias que existen entre ellas.

Capítulo 4: Implementación

Uno de los flujos que define RUP como metodología de desarrollo es el de implementación, en el cual se materializa la creación del analista, se refina la arquitectura y se termina con un sistema ejecutable. Sus propósitos son: definir la organización del código, implementar las clases y objetos en forma de componentes, probar los componentes desarrollados e integrar los mismos en un sistema ejecutable. Su principal resultado es el modelo de implementación.

En este capítulo se dedica a la implementación, en el mismo se hará una breve explicación sobre la integración de este módulo con otros sistemas y descripción de los métodos y agentes más importantes, los estándares de diseño, codificación y tratamiento de errores. También se definen y se representan los diagramas de despliegue y componente, este último de forma general.

4.1 Modelo de Implementación.

El modelo de implementación es una colección de componentes y los subsistemas que los contienen. Estos componentes incluyen: ficheros ejecutables, ficheros de código fuente, y otros tipos de ficheros necesarios para la implantación y despliegue del sistema. Describe también como se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y como dependen de los componentes unos de otros.

El modelo de implementación define una jerarquía, tal y como se muestra en la siguiente figura:

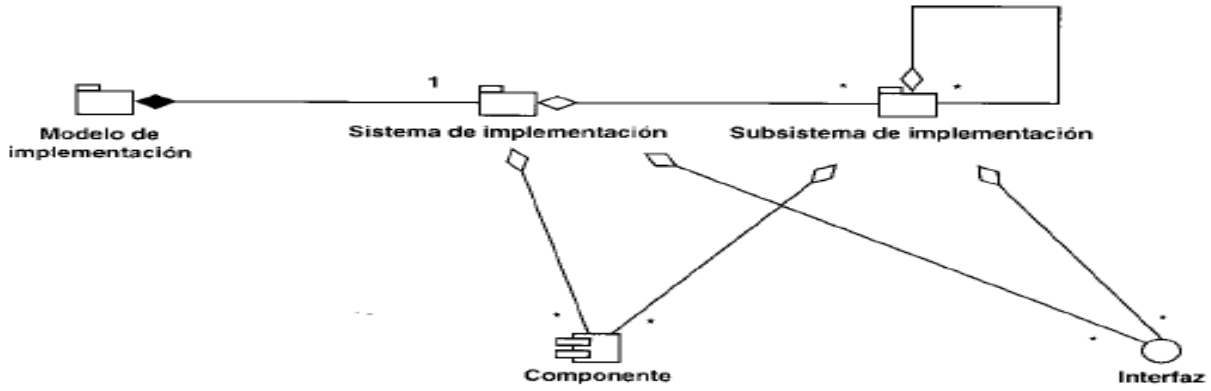


Figura 17: Jerarquía del Modelo de Implementación.

4.1.1 Diagrama de Componentes.

Los diagramas de componentes modelan la estructura del software en un conjunto de componentes, agrupados en ocasiones por paquetes. Muestran la organización y las dependencias lógicas entre un conjunto de paquetes y componentes de software, siendo éstos últimos componentes fuentes, binarios o ejecutables. Los componentes de software presentan tipo, que indica si son útiles en tiempo de compilación, enlace o ejecución.

Además estos diagramas de componentes cubren la vista de la implementación estática y se relacionan con los diagramas de clases ya que un componente suele tener una o más clases, interfaces o colaboraciones, dejando así una traza con el modelo de diseño.

A continuación se muestra una vista general del modelo de implementación expresado en diagramas de componentes.

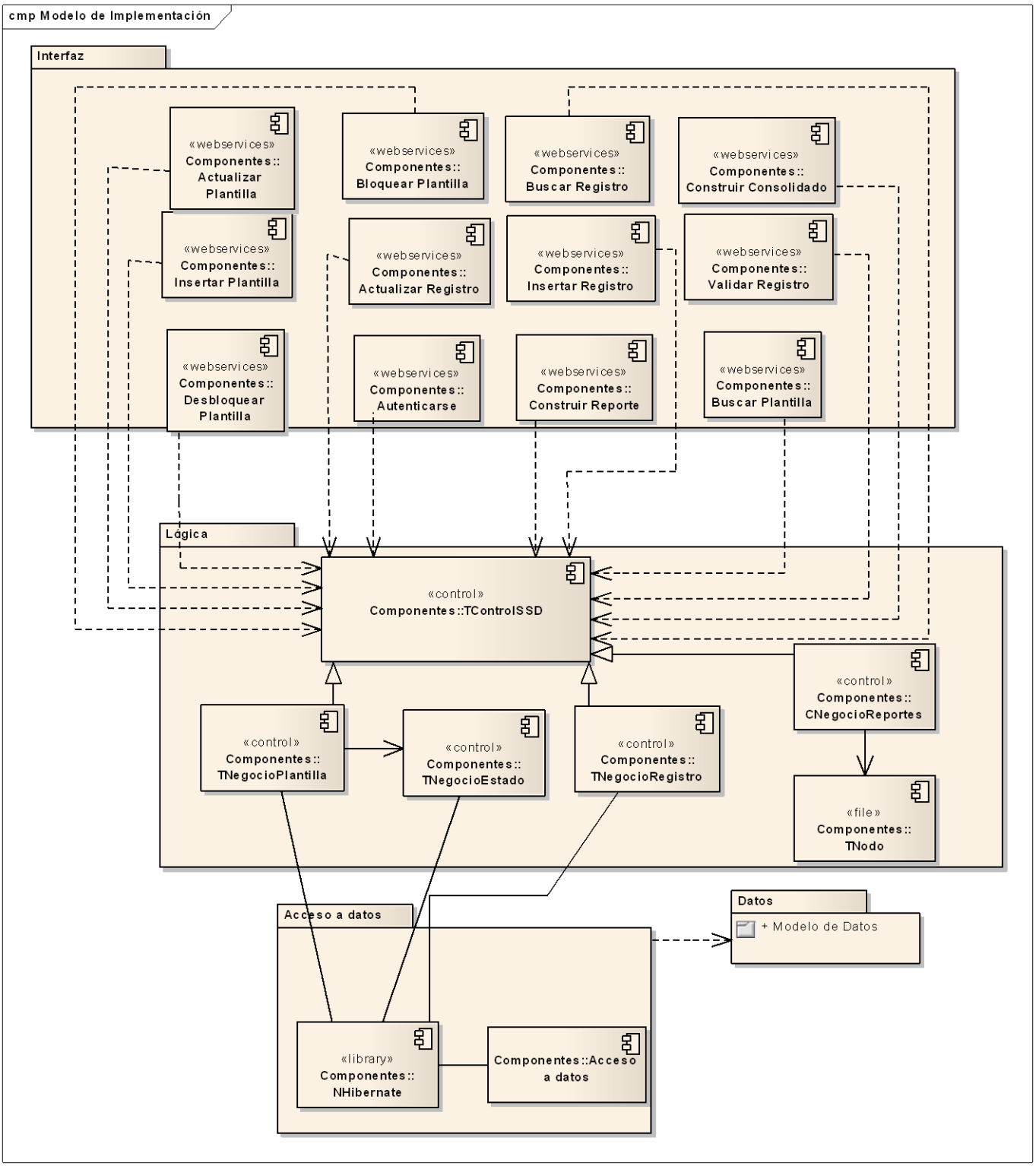


Figura 18: Diagrama de Componentes.

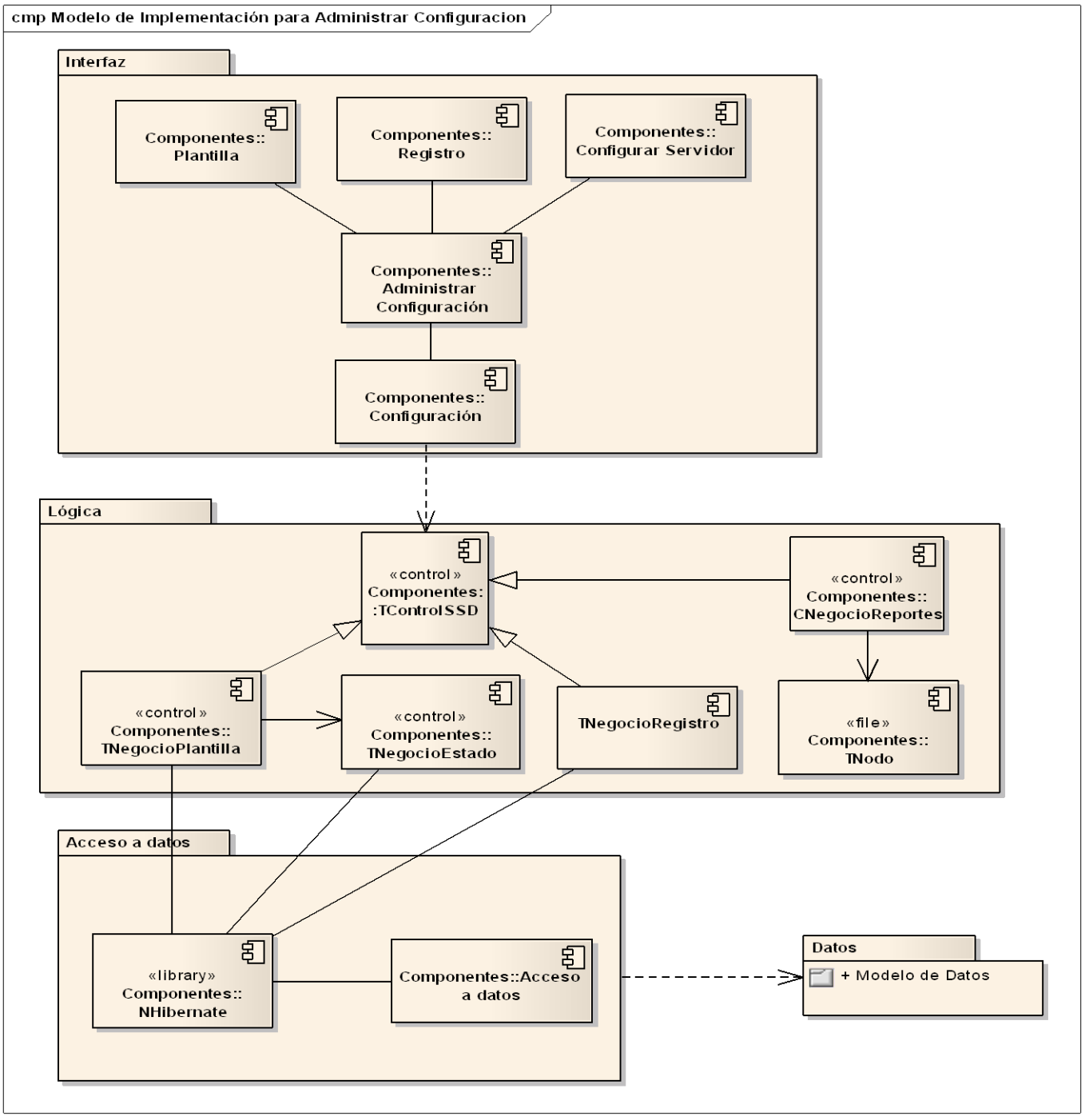


Figura 19: Diagrama de Componentes para la administración de configuración.

4.1.2 Diagrama de Despliegue

Este diagrama es un grafo de nodos unidos por conexiones de comunicación. Un nodo puede contener instancias de componentes software, objetos, procesos. En general, los nodos representan objetos físicos con recursos computacionales como procesadores y periféricos; pueden mostrarse como una clase (una familia de procesadores) o una instancia, por lo que su nombre sigue la misma sintaxis establecida para clases y objetos. Las conexiones son asociaciones de comunicación 96 entre los nodos, y se etiquetan con un estereotipo que identifica el protocolo de comunicación o la red utilizada. Seguidamente se representara este diagrama en sus 2 variantes.

Variante1: Subsistema Editor de Plantilla desplegado en una sola PC.

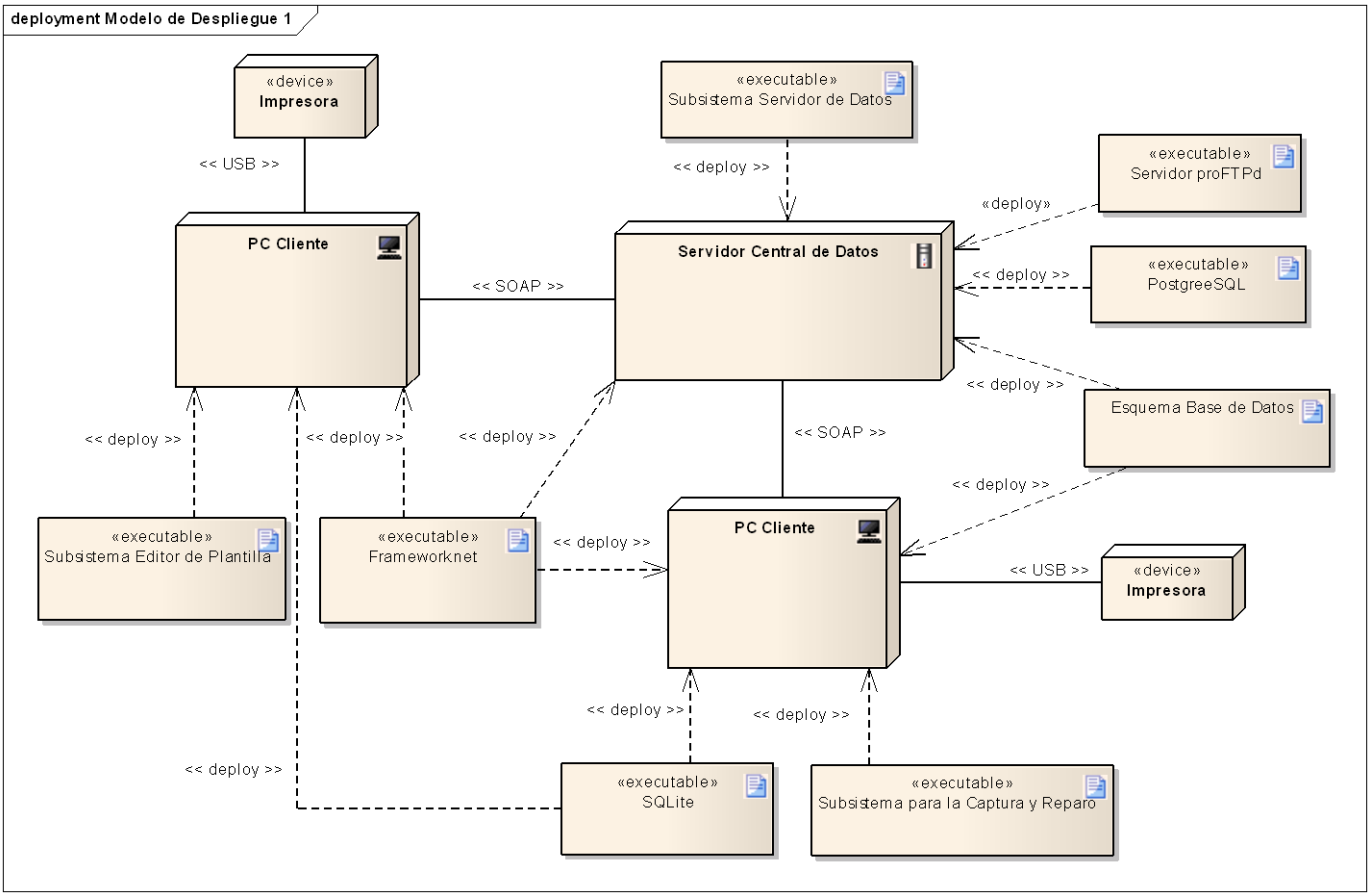


Figura 21: Diagrama de Despliegue variante 1.

Variante2: Subsistema Editor de Plantilla desplegado en varias PC.

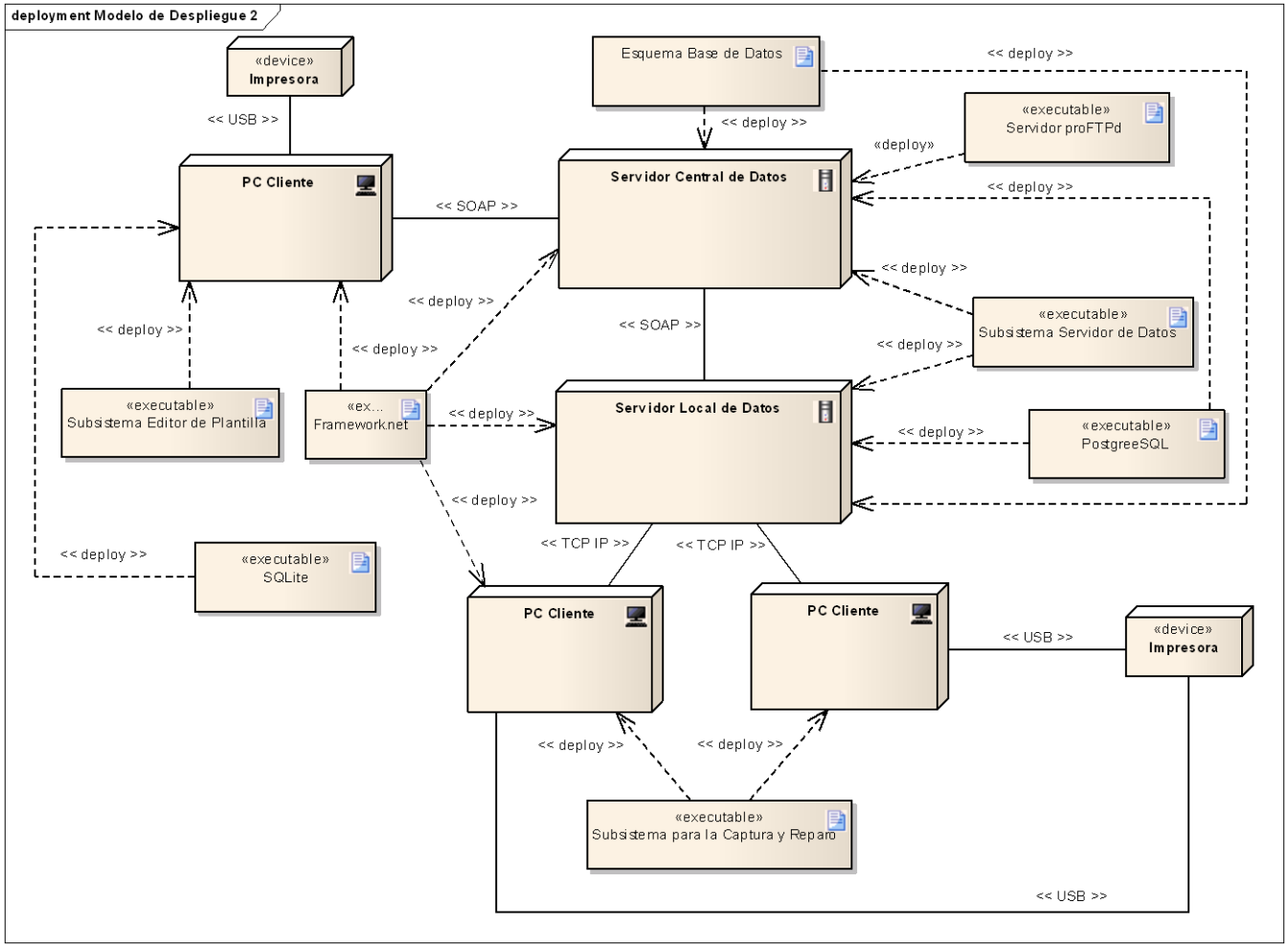


Figura 22: Diagrama de Despliegue variante 2.

4.2 Estándar de Codificación.

Actualmente se encuentran estándares de codificación para la mayoría de los lenguajes existentes. El uso de los mismos, partiendo de las convenciones definidas, permite una mejor comunicación entre los programadores creando las condiciones para la reusabilidad y el mantenimiento de los sistemas. Para definir el estilo de codificación a seguir en la aplicación se utilizó el estándar de codificación desarrollado por el Área Temática.

Variables: Las variables tendrán un prefijo para el tipo de datos en minúscula. El nombre que se les dará a las mismas debe comenzar con la primera letra en minúscula, la cual identificará el tipo de datos al que se refiere (ver figura 18), en caso de que sea un nombre compuesto se empleará notación CamellCasing**

Ejemplo: sNombrePaciente

Clases y Objetos: Deben nombrarse de forma estándar para todas las aplicaciones. La primera letra de las clases será una T mayúscula y se utilizará la notación CamellCasing en caso de que sea un nombre compuesto. Los atributos comenzarán con minúscula, lo cual estará en correspondencia al tipo de datos al que se refiere y las funciones comenzarán con mayúscula utilizando verbos que denoten la acción, estos últimos empleando siempre la notación PascalCasing. El nombre empleado para las clases, objetos, atributos y funciones debe permitir que con sólo leerlo se conozca el propósito de los mismos.

Bases de Datos, Tablas, Esquemas y Campos: Los nombres de las Bases de Datos deben comenzar con el prefijo bd, a continuación underscoard y luego el nombre completamente en minúscula, en caso de que sea un nombre compuesto se empleará notación C. El nombre a emplear para las tablas debe comenzar con el prefijo tb seguido de underscoard y luego debe escribirse todas las letras en minúscula, en caso de que sea un nombre compuesto se utilizará underscoard para separarlo. Todos los campos identificadores van a comenzar con el identificador id seguido de underscoard y posteriormente el nombre del campo y los campos que no son identificadores se nombrarán reflejando claramente que representan para la entidad. En caso de los campos que su nombre cubre más de una palabra se colocará un underscoard entre las palabras.

Tabla 1.1: Nombrado de variables según tipo.		
Tipo Datos	**Prefijo**	**Ejemplo**
Int	i	iCantPacientes
flota	f	fPesoPaciente
double	d	dPesoCarro
bool	b	bPacienteActivo
string	s	sNombrePaciente
char	c	cLetra
De tipo enum	ev	evSexo
byte	b	bCantDiasPaciente
sbyte	sb	sbEdadPaciente
short	sh	shVariableShort
ushort	us	usVariableUshort
long	l	lVariableLong
ulong	ul	ulVariableUlong
decimal	dc	dcVariableDecimal
Objetos	o	oPacienteHistorico
Objetos de tipo Struct	st	stUnaStruct

Figura 23: Nombrado de variables según tipo.

Los controles seguirán el siguiente tratamiento:

Tabla 1.2: Nomenclatura de los controles visuales.		
Control	**Prefijo**	**Ejemplo**
Botón	btn	btnAceptar
Etiqueta	lbl	lblNombre
Lista/Menú	mn	mnPrincipal
Campo de Texto	txt	txtFecha
Botón de Opción	bpt	optSexo
Casilla de Verificación	chx	chxBorrar
Casilla de Selección	cbx	cbxSexo

Figura 24: Nomenclatura de los controles visuales.

En este capítulo se mostraron los resultados de la etapa de construcción del sistema. En el proceso de implementación se cumplieron los principios de diseño y estándares de interfaz, elementos que ayudan a una mejor comunicación con el usuario, así como los estándares de implementación. Como culminación al diseño se presentó el modelo de implementación describiendo la distribución física del sistema y sus componentes.

Conclusiones

Con la investigación se realizó un estudio para creación del Subsistema Servidor de Datos y dar así cumplimiento a los objetivos planteados. A continuación se enumeran los principales resultados:

- Las aplicaciones informáticas existentes dedicadas al procesamiento de información estadística, son herramientas potentes para realizar análisis estadísticos, pero en su generalidad son aplicaciones dependientes de conectividad y las que no lo son, han sido desarrolladas sobre tecnologías propietarias. Las características con las que cuentan las aplicaciones existentes no son adaptables a las exigencias del MINSAP.

- Después de analizar las posibles herramientas y tecnologías para el desarrollo del sistema, se decidió implementar una aplicación de escritorio basada en una arquitectura en capas, utilizando Csharp como lenguaje de programación y Visual Studio y Charp Develop como IDE (Entorno de Desarrollo Integrado), sobre Microsoft .Net Framework. Como gestor de base de dato ligero SQLite y Postgres como gestor de base de datos, así como XML Servicios Web para la intercomunicación entre los distintos subsistemas.

- Durante el desarrollo de la aplicación se realizaron pruebas de compatibilidad y funcionamiento con el Framework Mono lo que permitirá la migración a entornos vasados en el sistema operativo GNU/Linux.

- Se generaron los artefactos relacionados con la definición del subsistema, establecidos por la adaptación de la metodología de desarrollo RUP realizada por el Área Temática.

- Se llevó a cabo un estudio del almacenamiento de la información en formato XML, lo cual trajo como resultado su aplicación en el desarrollo del subsistema Servidor de Datos y un importante ahorro de espacio en la base de datos.

- Se desarrolló una aplicación informática que permite almacenar los datos estadísticos y los modelos de flujo del Sistema para la Gestión y Análisis de Información Estadística.

Beneficios

La utilización del sistema desarrollado permite:

- Flujo estándar de la información estadística en todo el país.
- Edición de los modelos de flujo en dependencia de las necesidades. Esto proporciona autonomía a los usuarios para establecer sus definiciones, sin necesidad de volver a iniciar un nuevo ciclo de desarrollo.
- Acceso real a la información desde todos los lugares en el momento en que se necesite. Esto es debido a que el almacenamiento de la información será de forma distribuida, por lo que en cada uno de los niveles se almacenará localmente un agregado de la información primaria que se necesita en dicho nivel para ejecutar sus funciones en todo momento.
- Posibilidad de obtener información desagregada desde el Servidor de Datos central.
- Posibilidad de mejorar sustancialmente los procesos de Captación, Flujo y Procesamiento de la información estadística en el Ministerio de Salud Pública.
- Se establece una experiencia inicial para el desarrollo de nuevos productos y la mejora continua de las primeras versiones.

Recomendaciones

Al equipo de desarrollo:

- Continuar con el desarrollo del subsistema actual con la incorporación de nuevas funcionalidades lo que le permitiría un mayor tiempo de vida útil como software.
- Establecer estrategias de integración con sistemas que se desarrollan para el Sistema Nacional de Salud, evitando una doble captación de los datos.
- Incorporar funcionalidades que permitan a los usuarios finales obtener la información que necesite en base a los datos captados, definir rangos válidos para los datos captados, así como incorporar facilidades que mejoren la usabilidad del subsistema.
- Implementar una estrategia de sincronización automática que permita mantener sincronizados los datos desde el subsistema servidor de datos.
- Desarrollar un mecanismo de representación de datos que permita una mejor extracción de los mismos.
- Brindar capacitación al personal que utilizará el subsistema, sobre su funcionamiento y prestaciones.
- Incorporar la integración con herramientas para la generación de reportes y análisis de datos.
- Implementar un conjunto de interfaces web que proporcionen acceso a la información residente en el servidor central, dando la posibilidad de que los usuarios puedan disponer de la información siempre que tenga acceso a la red.
- Explotar las facilidades del SGBD para el procesamiento de la información en formato XML, lo cual garantizaría una mayor rapidez y eficiencia en la respuesta del sistema.

Referencias Bibliográficas

1. **Martínez Almaguer, Norge y González Marrero, Karen.** Sistema de Información Estadística Complementario de Salud. Módulo Consulta Externa. Ciudad de La Habana : s.n., 2007.

2. Idem a la referencia 1.

3. **Gómez Luis, Geraldo y Cesar Cantú, Pedro**. www.respyn.uanl.mx. www.respyn.uanl.mx. [En línea] 2003. [Citado el: 15 de 1 de 2009.] http://www.respyn.uanl.mx/iv/1/ensayos/bioestadistica.html..

4. **Gran Alvarez, Mirian A, y otros**. Sistema de Información Estadística de Salud Cubano. Sistema de Información Estadística de Salud Cubano. [En línea] 2003. [Citado el: 1 de 2 de 2009.] http://www.dne.sld.cu/Libro/capitulo1/capitulo1.htm .

5. [En línea] [Citado el: 28 de 1 de 2009.] http://www.ine.es/inebmenu/queesinebase.htm.

6. [En línea] [Citado el: 29 de 1 de 2009.] 6. http://www.ine.es/prodyser/pcaxis/pcaxis.htm.

7. [En línea] [Citado el: 3 de 2 de 2009.] http://www.unicef.org/spanish/statistics/index_24300.html.

8. [En línea] [Citado el: 2 de 2 de 2009.] http://es.wikipedia.org/wiki/Gesti%C3%B3n_documental.

9. [En línea] [Citado el: 4 de 2 de 2009.] http://cv1.cpd.ua.es/ws/default.asp .

10. **Seco González, Jose Antonio**. El Lenguage de programación C#. El Lenguage de programación C#. [En línea] [Citado el: 5 de 2 de 2009.] http://programacion.com/tutorial/csharp/3/.

11. [En línea] [Citado el: 4 de 2 de 2009.] http://uddi.uci.cu/files/arquitectura.2007.5.9.pdf.

12. Idem a la referencia 11. Idem a la referencia 11. [En línea] [Citado el: 5 de 2 de 2009.]

13. Idem a la referencia 11. Idem a la referencia 11. [En línea] [Citado el: 6 de 2 de 2009.]

14. Idem a la referencia 11. Idem a la referencia 11. [En línea] [Citado el: 7 de 2 de 2009.]

15. **Yero Durañona, Yanosky y Rodríguez González, Lazaro**. Servidor de Imágenes Médicas. Ciudad de La Habana : s.n., 2008.

16. Idem a la referencia 15.

17. [En línea] [Citado el: 28 de 1 de 2009.] http://www.epidataconsulting.com/tikiwiki/tiki-read_article.php?articleId=31.

18. Idem referencia 15.

19. Idem referencia 15.

20. [En línea] [Citado el: 26 de 1 de 2009.] http://www.abcdatos.com/programas/programa/z1770.html .

Bibliografía

1. **Martínez Almaguer, Norge y González Marrero, Karen**. Sistema de Información Estadística Complementario de Salud. Módulo Consulta Externa. Ciudad de La Habana : s.n., 2007.

2. **Gómez Luis, Geraldo y Cesar Cantú, Pedro**. www.respyn.uanl.mx. www.respyn.uanl.mx. [En línea] 2003. [Citado el: 15 de 1 de 2009.] http://www.respyn.uanl.mx/iv/1/ensayos/bioestadistica.html..

3. **Gran Alvarez, Mirian A, y otros**. Sistema de Información Estadística de Salud Cubano. Sistema de Información Estadistíca de Salud Cubano. [En línea] 2003. [Citado el: 1 de 2 de 2009.] http://www.dne.sld.cu/Libro/capitulo1/capitulo1.htm .

4. [En línea] [Citado el: 28 de 1 de 2009.] http://www.ine.es/inebmenu/queesinebase.htm.

5. [En línea] [Citado el: 29 de 1 de 2009.] 6. http://www.ine.es/prodyser/pcaxis/pcaxis.htm.

6. [En línea] [Citado el: 3 de 2 de 2009.] http://www.unicef.org/spanish/statistics/index_24300.html.

7. [En línea] [Citado el: 2 de 2 de 2009.] http://es.wikipedia.org/wiki/Gesti%C3%B3n_documental.

8. [En línea] [Citado el: 4 de 2 de 2009.] http://cv1.cpd.ua.es/ws/default.asp .

9. **Seco González, Jose Antonio**. El Lenguage de programación C#. El Lenguage de programación C#. [En línea] [Citado el: 5 de 2 de 2009.] http://programacion.com/tutorial/csharp/3/.

10. [En línea] [Citado el: 4 de 2 de 2009.] http://uddi.uci.cu/files/arquitectura.2007.5.9.pdf.

11. **Yero Durañona, Yanosky y Rodríguez González, Lázaro**. Servidor de Imágenes Médicas. Ciudad de La Habana : s.n., 2008.

12. [En línea] [Citado el: 28 de 1 de 2009.] http://www.epidataconsulting.com/tikiwiki/tiki-read_article.php?articleId=31.

13. [En línea] [Citado el: 26 de 1 de 2009.] http://www.abcdatos.com/programas/programa/z1770.html .

14. **Cerami, Ethan**. Web Service Essencial. s.l. : ORelly, 2002. ISBN: 0-596-00224-6.

15. **Jacobson, Ivan, Booch, Brady y Rumbaugh, Jame**. El Proceso Unificado de Desarrollo de Software. Volumen 1 . La Habana : Felix Varela, 2004.

16. —. El Proceso Unificado de Desarrollo de Software. Volumen 2. La Habana : Felix Varela, 2004.

17. **Larman, Craig**. UML y Patrones. La Habana : Felix Varela, 2004.

18. **Pressman, Roger**. Ingenieria de Software un enfoque práctico. Parte 1. La Habana : Felix Varela, 2005.

19. —. Ingeniería de Software un enfoque práctico. Parte 2. La Habana : Felix Varela, 2005.

20. **Karel Fernández Cedeño, Tranys Rafael Ortega Valenzuela.** Sistema de Información Estadística Complementario de Salud. Módulo: Actividades de Cirugía y otras Atenciones y Servicios. 2008.

21. Guía de Usuario de Enterprise Architect. [Online] [Citado: Enero 27,2009] Disponible en http://www.sparxsystems.com.ar/EAUserGuide/ea.html.

22. Modelo de Dominio. [Online] [Citado: Enero 29,2009] Disponible en http://iie.fing.edu.uy/ense/asign/desasoft/practico/hoja8/ejemplos_clase2.pdf.

23. Programación por capas [Online] [Citado: Febrero 1,2009] Disponible en http://oasis.cisc-ug.org/letzhune/cisc/tutoriales/tercero/Programacion%20por%20capas.doc.

24. Servicios Web en C# .Net [Online] [Citado: Enero 15: 2009] Disponible en http://www.locualo.net/programacion/servicios-web-net/00000025.aspx.

Anexos

Anexo1. Diagramas de Secuencia de los casos de uso significativos.

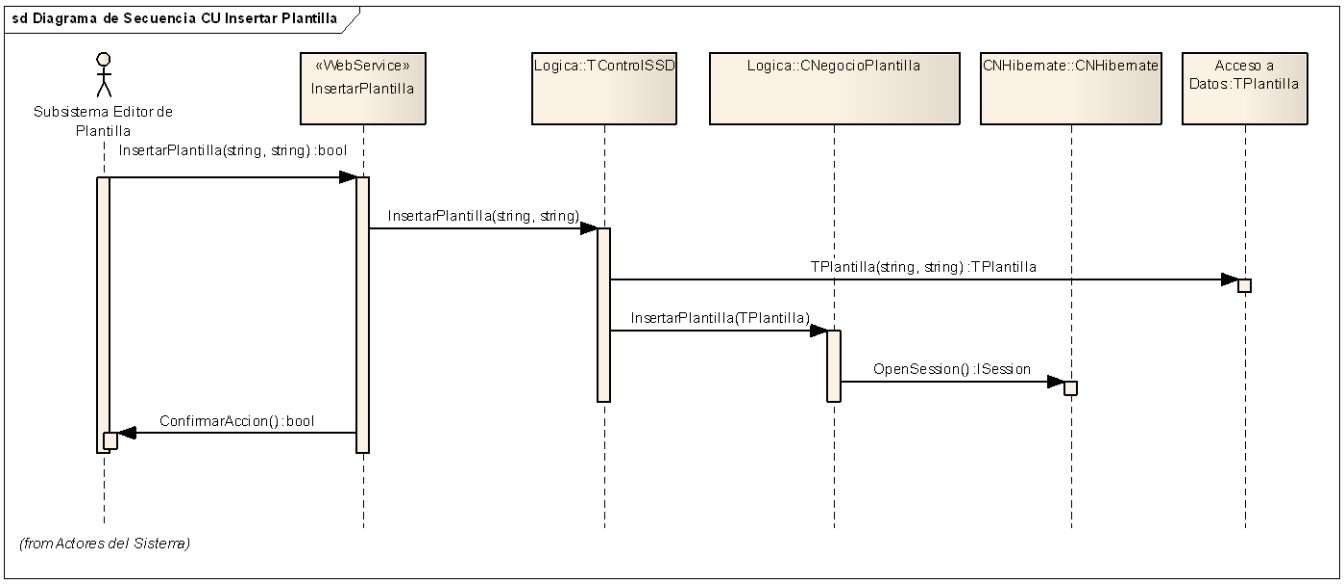


Figura 25: Diagrama de secuencia caso de uso Insertar Plantilla.

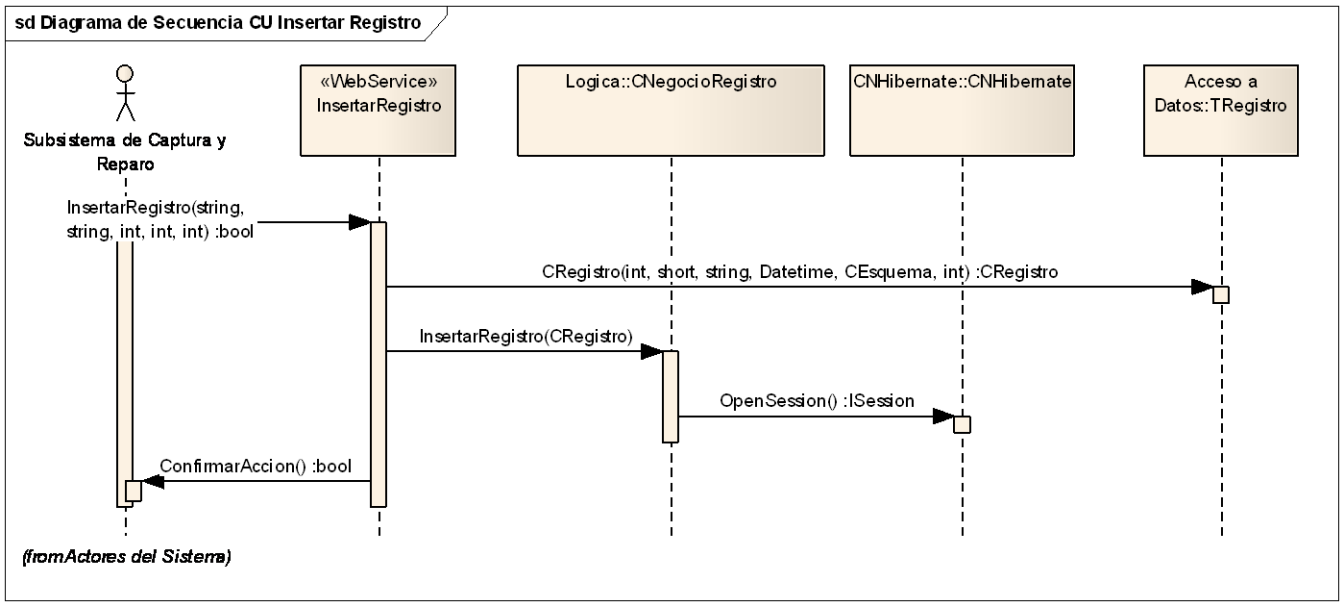


Figura 26: Diagrama de secuencia caso de uso Insertar Registro.

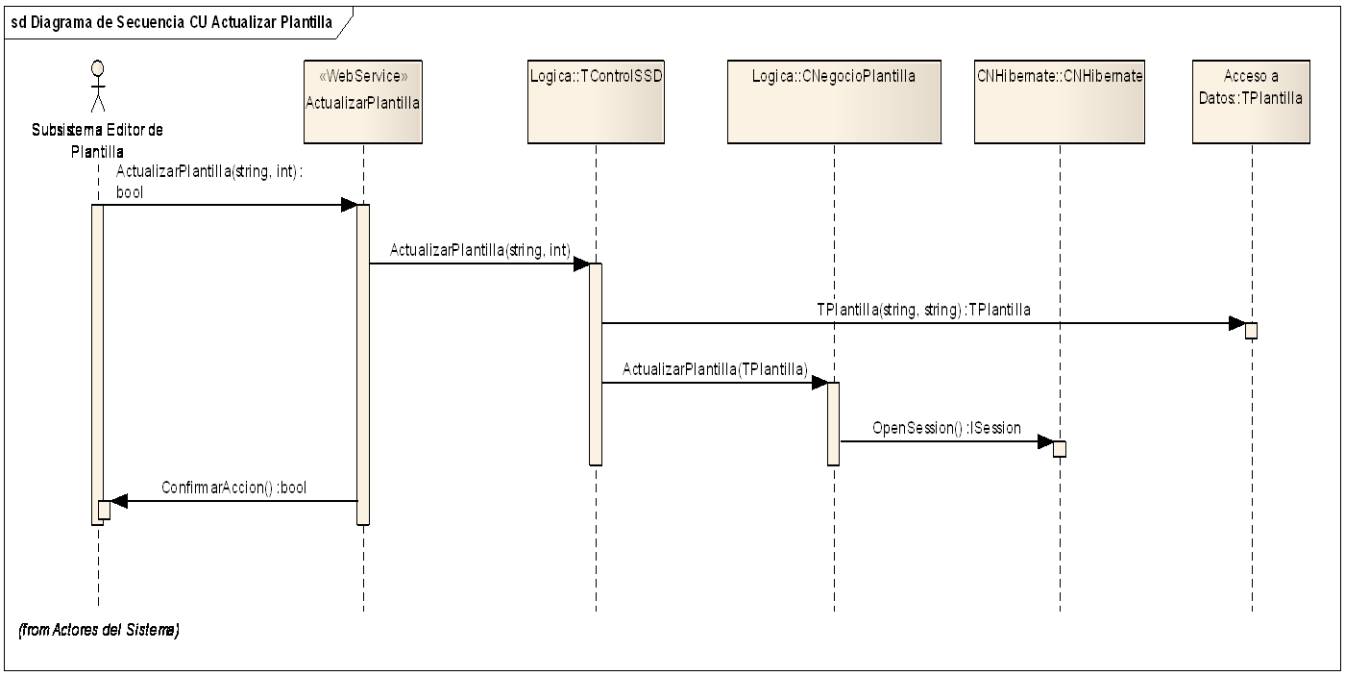


Figura 27: Diagrama de secuencia caso de uso Actualizar Plantilla.

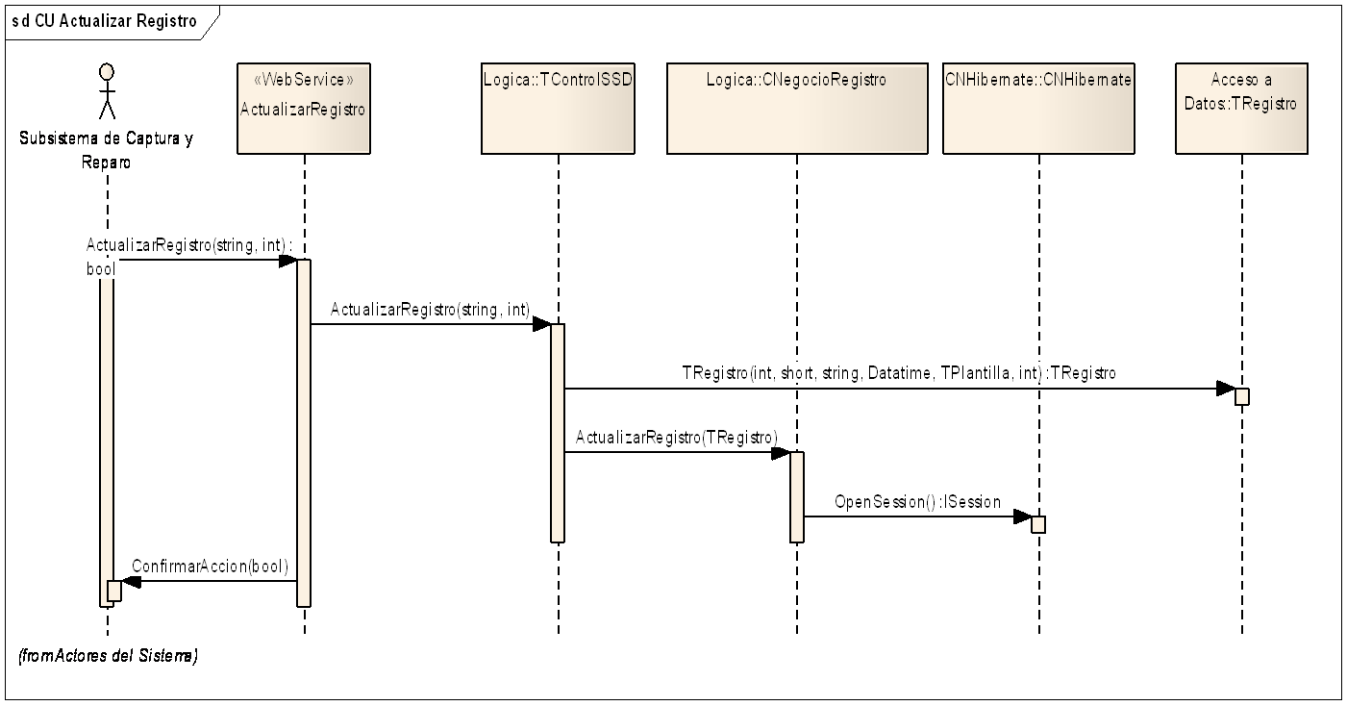


Figura 28: Diagrama de secuencia caso de uso Actualizar Registro.

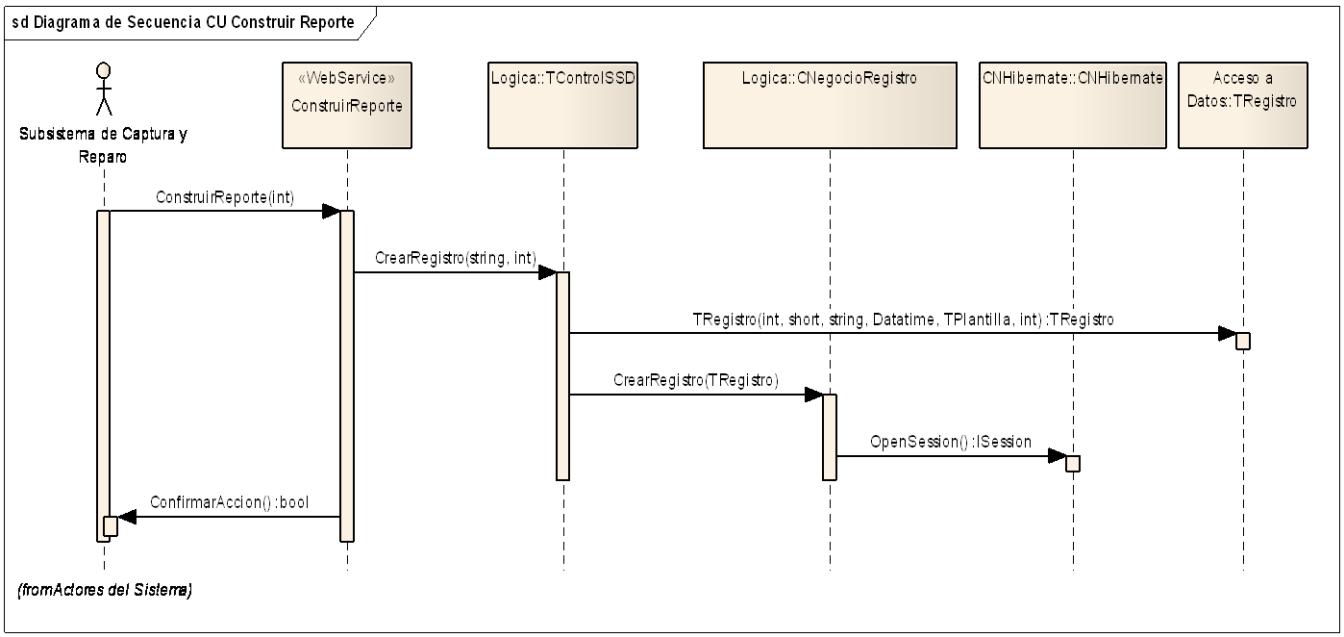


Figura 29: Diagrama de secuencia caso de uso Construir Reporte.

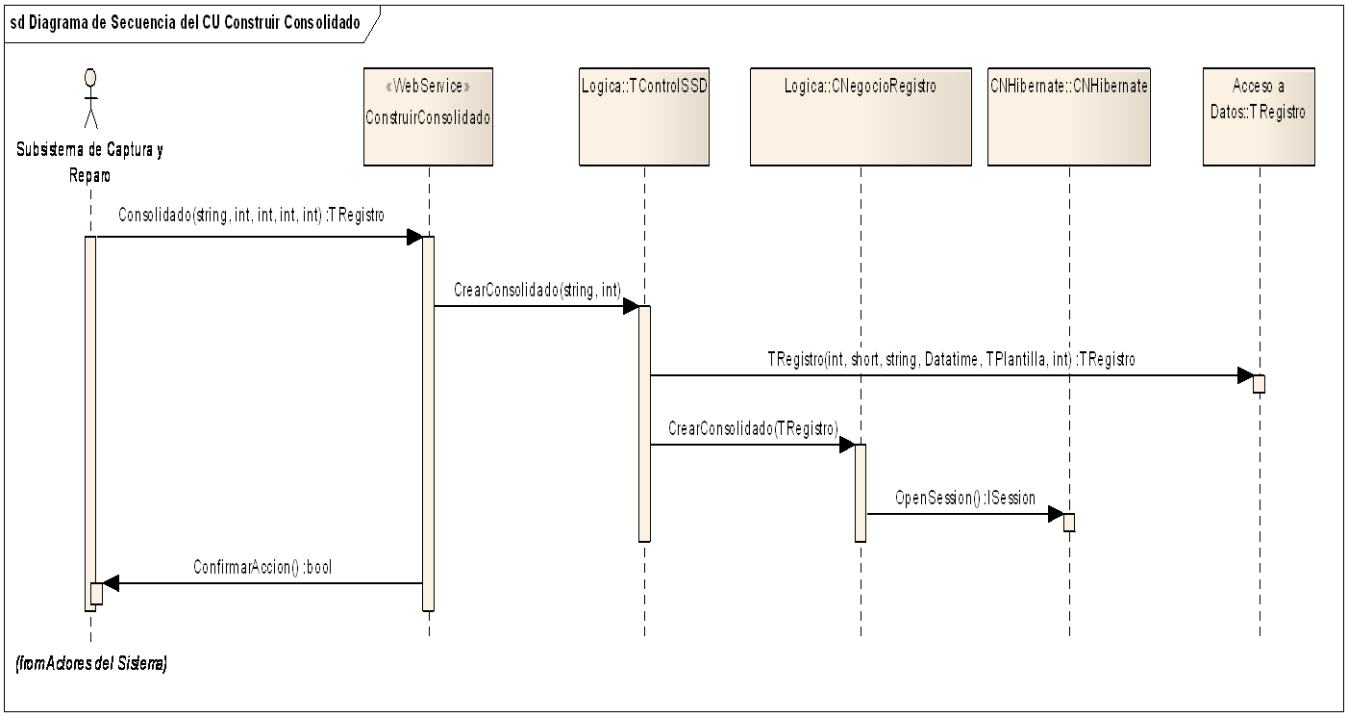


Figura 30: Diagrama de secuencia caso de uso Construir Consolidado.

Glosario de Términos

ACID: Atomicidad, Consistencia, Aislamiento y Durabilidad. Conjunto de características necesarias para que una serie de instrucciones puedan ser consideradas como una transacción.

Arquitectura: Conjunto de decisiones significativas acerca de la organización de un sistema de software, la selección de los elementos estructurales a partir de los cuales se compone el sistema. La misma se interesa no solo por la estructura y el comportamiento, sino también por las restricciones y compromisos de uso, funcionalidades, funcionamiento, flexibilidad al cambio, reutilización, comprensión, economía y tecnología, así como por aspectos estéticos.

Base de Datos: Es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso.

Caso de uso: Descripción de un conjunto de secuencia de acciones, incluyendo variaciones, que un sistema lleva a cabo y que conduce a un resultado observable de interés para un actor determinado.

Componente: Parte física y reemplazable de un sistema que se ajusta y proporciona la realización de un conjunto de interfaces.

Dependencia: Relación semántica entre dos elementos, en la cual un cambio en uno puede afectar al otro.

Diagrama: Presentación gráfica de un conjunto de elementos y sus relaciones.

Dominio: Área de conocimiento o actividad caracterizada por un conjunto de conceptos y terminologías comprendidos por los practicantes de ese dominio.

Editor de Plantillas: Subsistema externo que se encargará de generar en formato ligero todo lo relacionado con los modelos de flujo, incluyendo las reglas de validación.

Enterprise Architect: EA es una herramienta progresiva que cubre todos los aspectos del ciclo de desarrollo, proporcionando una trazabilidad completa desde la fase inicial del diseño a través del despliegue y mantenimiento. También provee soporte para pruebas, mantenimiento y control de cambio. Puede modelar procesos de negocio, sitios web, interfaces de usuario, redes, configuraciones

de hardware, mensajes y más. Tiene las características que precisa para diseñar y administrar su desarrollo e implementación.

Inetd: Es un demonio presente en la mayoría de sistemas tipo Unix, conocido como el "Súper Servidor de Internet", ya que gestiona las conexiones de varios demonios. La ejecución de una única instancia de inetd reduce la carga del sistema, en comparación con lo que significaría ejecutar cada uno de los demonios que gestiona, de forma individual.

Informática: Es la disciplina que estudia el tratamiento automático de la información utilizando dispositivos electrónicos y sistemas computacionales.

Informatizar: Proceso de aplicar sistemas o equipos informáticos al tratamiento de la información.

Internet: Es un método de interconexión de redes de computadoras implementado en un conjunto de protocolos denominado TCP/IP y garantiza que redes físicas heterogéneas funcionen como una red (lógica) única.

Paquete: Mecanismo de propósito general para organizar elementos en grupos.

Prototipo: Maqueta visual funcional o no de la futura aplicación. Este puede ser una imagen o una aplicación software que simule funcionalidades del software.

Plantilla o Modelo de flujo: Modelo o planilla en la cual se encuentran los indicadores a captar por parte del sistema, cada uno se corresponde con un subsistema de información de los que conforman el sistema de información estadístico complementario de salud. Estos modelos son creados y modificados por parte de la Dirección Nacional de Registros Médicos y Estadística Sanitaria en dependencia de las necesidades de información del propio Ministerio de Salud o de otras instituciones que utilizan esta información con el objetivo de trazar estrategias y evaluar procedimientos.

Servicio: Unidad de software que encapsula algunas funcionalidades del negocio y proporciona estas a otros servicios a través de interfaces publicas bien definidas.

Servicio Web: Colección de protocolos y estándares que se utilizan para intercambiar datos entre aplicaciones.

SOAP (Protocolo de acceso simple a objetos): Estándar que define como dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. SOAP es uno de los protocolos usados en los servicios web.

Software: Conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea especifica, en contraposición a los componentes físicos del sistema.

Software Libre: Es el software que una vez obtenido puede ser usado, copiado, estudiado, modificado y redistribuido libremente.

Subsistema: Agrupación de elementos, de los que algunos constituyen una especificación del comportamiento ofrecido por los elementos contenidos.

Subsistema Servidor de Datos: El Módulo Servidor de Datos, subsistema que se encarga del manejo de la persistencia de la información, tanto de los datos estadísticos como de la información de los modelos de flujo. Se encarga además del manejo de entradas y salidas desde los servidores centrales, es un componente que además permanecerá distribuido entre el centro de datos, y las estaciones de trabajo, lugar este último donde se almacenará una copia de la información perteneciente a dicho nivel. Brindará un conjunto de interfaces mediante Servicios Web, que permitirán comunicación con otros sistemas que constituyen las fuentes primarias de información, además brinda interfaces Web para la captura de información a partir de ficheros almacenados en soporte extraíble (Memorias, CD, DVD, Disquetes).

UDDI (Universal descripción, descubrimiento e integración): Formato XML que se utiliza para describir servicios web.