# Universidad de las Ciencias Informáticas


# **Título:** Diseño del Módulo Centro Coordinador Provincial

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

**Autoras:** Yarielys Hernández Fonticiella
Marisel Rivera Alarcón

**Tutores:** Msc. Karina Pérez Teruel
Ing. Luis Mariano Reyna Soler

**Asesores:** Ing. Lourdes Escalona Peral
Ing. Jorge Carlos Yero

**Ciudad de La Habana, Junio 2008**
**"Año 50 de la Revolución"**

DECLARACIÓN DE AUTORÍA

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

Para que así conste firmamos la presente a los 27 días del mes de Junio del año 2008.

**Yarielys Hernández Fonticiella**

_____

Firma de la Autora

**Marisel Rivera Alarcón**

_____

Firma de la Autora

**Msc. Karina Pérez Teruel**

_____

Firma de la Tutora

**Ing. Luis Mariano Reyna Soler**

_____

Firma del Tutor

*"A menos que creamos en nosotros mismos, nadie lo hará; este es el consejo que conduce a la realización y al éxito."*

*Héctor A. García*

# Agradecimientos

*Sentir es la vida verdadera, pensar es una utopía, agradecer es demostrar a alguien, cuan importante ha sido y cuanto de especial hubo en su obra, quisiera expresar mi eterna gratitud a:*

- *Fidel, por ser luz y esperanza para el mundo, por haber puesto en nuestras manos la tecnología necesaria para ser verdaderos hombres de Ciencia y Futuro.*
- *Mi familia: padres, hermana, sobrinos, por su amor, dedicación y confianza.*
- *Anthony, por haber convertido cada deseo en flor, cada dolor en estrella, cada lágrima en sonrisa, cada momento en AMOR.*
- *Mis abuelos, tíos, primos, por confiar en mí.*
- *Lurdita, mi asesora, por su paciencia y esmerada dedicación.*
- *Karina, mi tutora, por guiar mis pasos.*
- *Mariano, mi tutor, por su constancia.*
- *Yany, por haberme dado el mejor regalo del mundo: su AMISTAD.*
- *Yasmín Caridad, Andy y Miguel, por ser un excelente equipo de trabajo.*
- *Maritrini, por su perseverancia, voluntad y entrega en la realización de este proyecto.*
- *Todos mis amigos, por demostrarme que la vida es bella, que soy imprescindible, que existo.*
- *En fin, a todos los que creyeron en esta investigación y nos apoyaron para la realización de la misma.*

*A todos, Muchas gracias, Yarielys.*

- *A mis padres Miriam y Urbano, por su amor incondicional, su confianza y su paciencia, pero sobre todo por creer en mí y apoyarme en la lucha por lograr mi mayor sueño.*
- *A toda mi familia por depositar toda su confianza en mí.*
- *En especial mis a hermanos, a Fernan por apoyarme siempre y estar a mi lado todo este tiempo.*
- *A la Revolución, y a Fidel que ha sido un ejemplo durante mi vida, a la UCI.*
- *A mis tutores y asesores por su gran ayuda.*
- *A Yari por ayudarme, guiarme y estar siempre conmigo en los buenos y en los malos momentos. Yariti muchas gracias por todo.*
- *A mi compañeros de grupo por su apoyo durante estos años de la carrera.*
- *A todos los que aportaron su granito de arena en esta tesis.*
- *En fin a todas las personas que de una forma u otra han tenido que ver conmigo y que se han ganado un lugar en mi corazón, los tengo presente a todos y para ustedes también va dedicado este trabajo de tesis que es la culminación de todos estos años de esfuerzo y dedicación .*

*Marisel*

# Dedicatoria

- *A la Revolución.*
- *A mis padres, hermana, sobrinos…*
- *A Anthony.*
- *A toda mi familia.*
- *A Yani, Yusi, Mayi.*
- *A Lurdita y a Karina.*
- *A Hereida.*
- *A mis amigos.*
- *A todos los que de forma directa o indirecta intervienen en el proceso de informatización del país y en especial en el Sistema Nacional de Salud, con el noble propósito de llevar adelante la obra de la Revolución.*

*Yarielys.*

- *A mi madre adorada que merece todas las alegrías que yo pueda darle.*
- *Al padre que me ha apoyado desde mi niñez y que siempre está conmigo en las buenas y las malas.*
- *A mis hermanos, especialmente a Mila y Pipo.*
- *A Fernandito, que ha sido la guía en estos tantos años, y algo maravilloso que me ha pasado en la vida.*
- *A mi abuela querida y a tía que tanto me quieren: Erenia y Nidia.*
- *A Ray, Mailen y Eliercito.*
- *Al Comandante Fidel, que hizo esta maravillosa universidad para que jóvenes de todo el país pudiéramos estudiar en ella.*

*Marisel*

**Resumen**

Actualmente en los Centros Coordinadores Provinciales de Emergencias Médicas de Cuba, las demandas de emergencias, urgencias y no urgencias médicas son procesadas de forma manual. Razón por la cual, con el presente trabajo se diseñó una aplicación web para facilitar el proceso de gestión de información de las demandas en los Centros Coordinadores Provinciales de Emergencias Médicas de Cuba.

Para diseñar el sistema informático se utilizó como metodología de desarrollo de software: RUP, como lenguaje de modelado: UML y como herramienta de modelado: Visual Paradigm. Durante el desarrollo de la investigación se obtuvieron artefactos tales como: Modelo de Negocio, Modelo del Sistema y Modelo de Diseño.

Este sistema informatizará todos los procesos que actualmente tienen lugar en un Centro Coordinador Provincial de Emergencia Médica. Permitirá reducir el tiempo de respuesta del servicio que se brinda a la población ante un caso de emergencia, urgencia o no urgencia médica; y brindar un servicio de mayor calidad. Además, posibilitando que el país no realice grandes gastos en la adquisición de un software para este sector, ya que los que existen no se adaptan a las características del Sistema Nacional de Salud.

**Palabras claves:** demanda, software, requisitos, diseño.

# Índice

**Introducción**

La humanidad avanza vertiginosamente en la modernización y aplicación de nuevas tecnologías en el campo de las Ciencias Informáticas, logrando un alto grado de desarrollo que hoy está presente en la mayoría de las actividades que desarrolla el ser humano: en la construcción, la industria, el comercio, la medicina, la educación, las finanzas, la investigación, la comunicación, etc. La informática avanza a un acelerado ritmo de desarrollo por lo que conocer esta tecnología y utilizarla no constituye un privilegio, sino una necesidad, constituyendo un factor determinante en los niveles de eficiencia y competitividad, tanto a nivel empresarial como personal.

Con el riguroso bloqueo (Económico, Financiero y Comercial), impuesto por los Estados Unidos a nuestro país desde los primeros momentos del triunfo de la Revolución y con su recrudecimiento constante, la economía cubana ha sido afectada en todos los sectores de la producción y los servicios. A raíz de esta situación, en el país el campo de las nuevas tecnologías, que fueron surgiendo en el mundo desarrollado, experimentó un gran estancamiento. Cuba ha invertido grandes recursos en estos últimos años, según sus posibilidades económicas, con el objetivo de revertir esta situación.

En la actualidad, el estado cubano realiza grandes esfuerzos para llevar las Ciencias Informáticas a todos los sectores de la producción y los servicios. Con el objetivo de ampliar la industria informática, como una opción más en el desarrollo económico, se comienzan a crear proyectos informáticos que facilitan las labores en el sistema empresarial y la atención social a la población; reduciendo el gasto de materiales, los costo de inversión, incrementando la eficiencia económica, la competitividad, las oportunidades de almacenamiento de la información y disminuyendo considerablemente el tiempo de gestión y obtención de datos importantes para realizar análisis estadísticos determinados.

En el campo de la informática, la Universidad de las Ciencias Informáticas (UCI) avanza como líder en esta tarea de informatizar los diferentes sectores de la sociedad cubana, dentro de los cuales uno de los más importantes es el sector de la salud.

Dentro del Sistema Nacional de Salud (SNS) se inserta el Sistema Integrado de Urgencias Médicas (SIUM) y su misión es organizar la atención de urgencias y emergencias médicas desde la comunidad, consultorios, policlínicos, coordinaciones de ambulancias de urgencias y emergencias, hasta el Sistema de Emergencia y Terapia Hospitalaria, que se establece mediante un proceso de evaluación y decisión médica, a través de los diferentes eslabones del Sistema Nacional de Salud. (1),

El SIUM a su vez tiene una estructura organizacional, la cual está formada por el Centro de Emergencia Médica Nacional, los Centros Coordinadores Provinciales de Emergencia Médica (CCPEM), las Bases Municipales de Ambulancias, el Centro de Operaciones Provincial y el Centro de Operaciones Nacional.

En cada una de las provincias del país existe un Centro Coordinador Provincial de Emergencia Médica; en el que diariamente se lleva a cabo un proceso de atención al paciente que define dentro de su flujo de actividades fundamentales, las que se enumeran a continuación:

- Recepción de llamadas de emergencias, urgencias y no urgencias.

- Recolección de la información necesaria para confeccionar una solicitud de demanda de los servicios que brinda el CCPEM, y toma de decisión acerca del hospital o institución de destino del paciente y del tipo de ambulancia para su traslado.

- Coordinación del transporte sanitario que trasladará al paciente a través de un mensaje por la radio si se encuentra fuera de la Base de Ambulancias o a través de una llamada telefónica a la base en caso de que el móvil se encuentre estacionado en la misma.

- Control del móvil que se encuentra ejecutando la demanda, registrando una serie de claves que indican la situación o posición del mismo.

- Confección de reportes que muestran como es el funcionamiento diario del CCPEM, mediante un conjunto de datos del paciente y estadísticas.

Actualmente, en los CCPEM, las demandas de emergencia y urgencia médica de la población, así como los casos de no urgencia, son procesadas de forma manual, lo cual influye en el tiempo de respuesta al servicio que solicita la población. A raíz de esto se manifiestan dificultades como: pérdidas de demandas, demoras en la ejecución de las mismas, quejas de la población por el incumplimiento en la recogida de pacientes, falta de control de los móviles que viajan sin la debida autorización del Centro Coordinador. De igual forma diariamente se elaboran una serie de reportes cuya realización manual se hace engorrosa al tener que buscar la información necesaria para su confección en grandes cantidades de documentos que han sido previamente archivados.

A partir del análisis de la situación problémica se plantea como **problema científico**: ¿cómo facilitar la gestión de información de las demandas en los Centros Coordinadores Provinciales de Emergencia Médica?

Este problema se centra en el **objeto de estudio** referente al proceso de gestión de información en el Sistema Integrado de Urgencias Médicas de Cuba.

Por lo que se define como **objetivo general** del trabajo: diseñar una aplicación Web que facilite el proceso de gestión de información de las demandas en los Centros Coordinadores Provinciales de Emergencias Médicas de Cuba.

De forma tal que el **campo de acción** que abarca está relacionado con la gestión de información de las demandas en los Centros Coordinadores Provinciales de Emergencia Médica de Cuba.

Para el desarrollo del trabajo se proponen las siguientes **tareas investigativas**:

1. Definir el proceso de gestión de demandas en los Centros Coordinadores Provinciales de Emergencias Médicas de Cuba.
2. Analizar el estado del arte de sistemas informáticos similares.
3. Proponer un modelo para la Ingeniería de Requisitos del proyecto.
4. Analizar la propuesta de arquitectura definida por el MINSAP y el área temática, para diseñar el Módulo Centro Coordinador Provincial.
5. Seleccionar metodología, herramientas y el lenguaje de modelado a utilizar.
6. Modelar las condiciones actuales que rigen el proceso de gestión de las demandas en los CCPEM.
7. Analizar las necesidades de funcionamiento del sistema informático, describiendo la Especificación de Requisitos del Software.
8. Obtener el Modelo de Sistema del Módulo Centro Coordinador Provincial.
9. Seleccionar métricas para evaluar los resultados obtenidos en el flujo de trabajo: Requerimientos.
10. Analizar los resultados obtenidos a partir de las métricas aplicadas.
11. Obtener el Modelo de Diseño del Módulo Centro Coordinador Provincial.

El trabajo de diploma está compuesto por tres capítulos:

Capítulo 1. Fundamentación teórica. En este capítulo se analizan conceptos y el proceso de gestión de información de las demandas en los sistemas relacionados con emergencias médicas, urgencias y no urgencias en el mundo y en Cuba. Además se propone un modelo para la Ingeniería de Requisitos (IR) del proyecto, se seleccionan: metodología, herramienta y lenguaje de modelado a utilizar para el diseño del software.

Capítulo 2. Características del sistema. Aborda los procesos que tienen lugar en los Centros Coordinadores Provinciales de Emergencias Médicas de Cuba. Se modela el negocio y a partir de las funcionalidades identificadas, se realiza la propuesta del sistema. Además se aplican métricas para evaluar los resultados alcanzados durante obtención, análisis y descripción de requisitos del sistema a diseñar.

Capítulo 3. Diseño del sistema. En él se muestran los patrones que se emplearán en el diseño de la aplicación. Se elaboran además un grupo de artefactos, tales como: diagramas de clases y secuencia, los cuales complementan el diseño del producto a desarrollar, teniendo en cuenta la metodología, herramienta y lenguaje de modelado seleccionados para la ingeniería del software.

## Capítulo 1. Fundamentación teórica

### Introducción

"Cuando un software de computadora se desarrolla con éxito ---cuando satisface las necesidades de las personas que lo utilizan; cuando funciona impecablemente durante mucho tiempo; cuando es fácil de modificar o incluso es más fácil de utilizar--- puede cambiar todas las cosas y de hecho las cambia para mejor. Ahora bien, cuando un software de computadora falla ---cuando los usuarios no se quedan satisfechos, cuando es propenso a errores; cuando es difícil de cambiar e incluso más difícil de utilizar--- pueden ocurrir y de hecho ocurren verdaderos desastres. Todos queremos desarrollar un software que haga bien las cosas, evitando que esas cosas malas merodeen por las sombras de los esfuerzos fracasados. Para tener éxito al diseñar y construir un software necesitaremos disciplina. Es decir, necesitaremos un enfoque de ingeniería". (2)

Contribuir al desarrollo de un software de computadora con éxito, analizando y poniendo en práctica la disciplina Ingeniería de Software, es la meta primordial del presente trabajo.

Específicamente en este capítulo se analizan sistemas informáticos similares, existentes tanto a nivel internacional como nacional, se propone un modelo para la Ingeniería de Requisitos, además se hace referencia a la arquitectura definida por el MINSAP (Ministerio de Salud Pública) para el desarrollo del software. También, a partir de una valoración crítica se seleccionan: metodología, herramienta, lenguaje de modelado a utilizar y se hace una breve valoración del lenguaje de programación a emplear en la implementación del software.

### 1.1 Sistemas informáticos existentes a nivel internacional

Con el paso del tiempo se han desarrollado diversas soluciones informáticas a nivel nacional e internacional para la gestión de atención a pacientes mediante el uso del transporte sanitario, o sea, de la ambulancia.

#### 1.1.1 GamXXI. Central Coordinadora y Gestión de Ambulancias

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (3)

GamXXI, es uno de los productos de la empresa especializada en el desarrollo de soluciones informáticas de gestión y contabilidad en entornos visuales, Professional Soft, ubicada en Sabadell (Barcelona). Es una aplicación que contempla todas las necesidades de gestión de las empresas de ambulancias.

Este software permite:

- Control de datos de pacientes, servicios, programaciones mutuas, aseguradoras, hospitales, tratamientos, ambulancias, diagnósticos, tipos de traslado, personal, motivos de anulación, etc.
- Ayudas para la gestión rápida en la aplicación como direcciones predeterminadas, poblaciones, kilómetros automáticos según origen-destino, etc.
- Tratamientos diferenciados por los tipos de servicios:
    - Normales (Ida y/o vuelta), Urgentes (Vía pública), Siniestros (Accidentes empresa), Colectivos (Ruta) y Programados (Se repiten periódicamente).
- Accesos restringidos según permisos del usuario.
- Botones de acceso rápido según usuario.
- Más de 50 listados, todos exportables a Excel, txt, e-mail.
- Agenda para teléfonos de interés y personal de la empresa.
- Para empresas de un gran volumen de datos, existe la opción de SQL Server de Microsoft.

El sistema cuenta con los siguientes módulos:

1- Centro Coordinador

Módulo de adjudicación y finalización de servicios a través de la PC.

2- Facturación

Módulo para facturar todos los servicios que se realicen en la empresa, tanto privados como públicos.

3- Telecomunicaciones

Sistema para la transmisión informatizada de datos entre central y vehículos.

4- Turnos

Generación automática del calendario anual según convenio.

5- Web

Recepción de servicios por medio de una página Web, según usuarios autorizados.

6- GPS

Localización de vehículos por satélite.

7- Vehículos

Módulo para el control exhaustivo de cada vehículo.

8- Taller

Módulo para el control del taller interno y externo.

### 1.2.2 Ambuwin CS, sistema dedicado al Sector del Transporte Sanitario

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (4)

El sistema Ambuwin CS, dedicado al Sector del Transporte Sanitario, es una aplicación basada en cliente/servidor, que permite la gestión completa de las empresas dedicadas al Sector del Transporte Sanitario. Sus principales características son: fácil utilización, manejo de servicios en tiempo real, sistema de alertas horarias, sistema de alertas de servicios urgentes, estatus de vehículos, control de pacientes, control de colegiados, control de vehículos (servicios, itv, reparaciones, seguros, equipamiento, etc.), facturación, presupuestos, gestores de informes, sistema de control de accesos a la aplicación, etc.

Dentro de las principales características del sistema se pueden mencionar:

- *Servicios:* control exhaustivo de servicios a realizar en tiempo real, tipos de servicios, orígenes y destinos, estatus, horarios, generación automática de sesiones a realizar (programados), colegiado que realiza el servicio, tipo de ambulancia que lo realiza, bloqueos y desbloqueos de servicios, anulación, pendientes, etc.

- *Urgencias:* sistema de alertas para servicios urgentes.

- *Vehículos:* control de gastos, consumos, fechas de caducidad itv, seguros, extintores, certificados sanitarios, datos de compra, etc.

- *Conductores:* datos generales, datos académicos, control de cursos realizados, tipos de carnets.

- *Ayudantes:* igual que conductores.

*Colegiados:* datos generales y seguimiento de gastos.

- *Aseguradoras y entidades:* Datos generales, tipos de facturación, generador de listas de precios personalizable.

- *Otras bases de datos:* Tipos de servicios, servicios médicos, tipos de vehículos, etc.

- *Otras prestaciones:* Sistema de alerta horaria para servicios personalizable, rangos provinciales de Kms. etc.

- Facturación y control de almacén, compras, control de cobros, pagos, seguimiento de clientes.

### 1.2.3 Gestión TTS2000. Software para la gestión del transporte sanitario

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (5)

El software Gestión TTS2000, es un programa específico destinado al Sector del Transporte Sanitario, para la realización de una gestión eficaz de todas las tareas que este requiere. Permite una organización automática del transporte sanitario optimizando los recursos de los que se dispone. Además este programa permite la generación de facturación y estadísticas, aspectos muy importantes en el transporte sanitario. También puede ser enlazado con el programa de contabilidad de la empresa en cuestión. Ofrece la posibilidad de llevar un control de flota exhaustivo, teniendo controlado en todo momento los vehículos.

Gestión TTS2000 permite coordinar tanto el trabajo, la administración, como el servicio privado, realizar facturación, estadísticas, estructuración automática de servicios, control de personal, control de flota, división de trabajo por zonas y empresas. El funcionamiento de este software se ejecuta a través de ficheros que permiten tener un control absoluto del trabajo, tales como:

- Clientes
- Tarifas
- Provincias
- Poblaciones
- Callejero
- Terapias
- Enfermos
- Operador

- Móviles
- Conductores
- Zonas
- Empresas
- Festivos
- Control de servicios
- Inicio / fin trabajo
- Verificación de partes
- Diario
- Incidencias
- Facturación
- Control vehículos
- Utilidades

No existe la menor duda de que los sistemas anteriormente descritos son muy eficientes y poseen un importante número de funcionalidades, que posibilitan que la red de atención a las emergencias y urgencias de cualquier país que los utilice tenga una alta calidad en los servicios que brinda al ser humano. Sin embargo, dichos sistemas implementan una funcionalidad, que constituye la base fundamental, para brindar un servicio al paciente y para nada coincide o se adapta al sistema de salud cubano y es, precisamente, la *facturación.*

A diferencia del modelo capitalista reflejado en estos productos, la salud en Cuba es gratuita y no es preciso abonar determinada cantidad de dinero para que el paciente sea atendido. Además, no se puede dejar de considerar que todos los modelos analizados son aplicaciones de software propietario que poseen limitaciones o valores (alto costo en el mercado internacional), de ahí la necesidad de su análisis para determinar cuál de ellos o qué elementos específicos de cada uno puede ser utilizado en la elaboración de la solución al problema planteado.

## 1.2 Sistemas informáticos existentes a nivel nacional

En aras de informatizar todos los sectores de la sociedad cubana y dentro de ellos el de la salud, se han ido creando, gradualmente en el país, soluciones informáticas que favorecen la prestación de un servicio de mayor calidad en la medicina cubana.

### 1.2.1 SICUM. Sistema Informático para el Control de la Urgencia Médica

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (6)

El sistema informático SICUM está diseñado para controlar de forma automatizada la atención al paciente dentro de la Red Nacional del Sistema Integrado de Urgencias Médicas (SIUM). Además, evalúa la eficiencia en los servicios de urgencia, dentro de la red existente en el país. Existen dos versiones de este: una utilizando Paradox y otra versión en plataforma Cliente/Servidor confeccionada en Borland Delphi con gestor de base de datos: SQL Server.

El Sistema brinda información acerca de:

1- Características del traslado del paciente
   - Solicitud o demanda de rescate de un paciente
   - Información sobre las ambulancias y su recorrido
   - Características del personal autorizado para viajar
   - Características del paciente:

   - Estado al comienzo y fin del traslado
   - Problemas de salud que presenta
   - Accionar del personal de rescate
   - Parámetros vitales del paciente
   - Determinación y cálculo del estado del paciente (Escala de Glasgow y Trauma de Score)

2- Consultas en cualquier punto de la red de urgencia (Consultorios de urgencia, policlínicos)
   - Médico que atiende
   - Características de la consulta
   - Diagnóstico y conducta médica para cada diagnóstico
   - Seguimiento del estado del paciente durante el ingreso en sala de observaciones (medicamentos suministrados, análisis de laboratorios, etc.)

3- Reportes
   - Estadísticas para medir el funcionamiento de la atención primaria.
   - Estadísticas que muestran el desempeño y efectividad de la red del SIUM como sistema.
   - Generación dinámica de reportes personalizados por el usuario.

Como parte del proceso de informatización que se lleva en el sector de la salud, juega un papel fundamental la informatización de los CCPEM. La aplicación SICUM, desarrollada para el control de

la urgencia médica en Cuba, no cumple con todas funcionalidades que actualmente requiere un CCPEM.

Dicha aplicación nunca fue desplegada y por ende, tampoco se le dio soporte, ni se implementaron versiones superiores, que corrigieran las posibles deficiencias e incrementaran las funcionalidades del software. Razón por la cual, desde el año 2001 hasta la fecha han surgido nuevos procesos a informatizar, los cuales no están comprendidos en la primera y única versión implementada del software SICUM.

El sistema anteriormente descrito no cumple, además, con los requisitos de integración plasmados en la estrategia de informatización de la salud en Cuba. Entonces, es interés de los directivos del Centro Nacional de Urgencias Médicas, el desarrollo de una nueva aplicación con tecnología de avanzada que gestione las urgencias médicas en Cuba; donde se tenga en cuenta, informatizar el proceso actual que rige el funcionamiento de los Centros Coordinadores Provinciales de Emergencia Médica.

Dicho sistema debe ser una aplicación web, implementada en PHP 5, que utilice como gestor de base de datos: MySQL, y que además incluya la integración al SiSalud (Sistema de Información para la Salud), a través de los componentes que brinden información necesaria para dicho software.

## 1.3 Modelos existentes para la Ingeniería de Requisitos

La correcta especificación de requisitos determina en gran medida el éxito de un proyecto de desarrollo de software. El proceso de desarrollo debe estar dirigido por los requisitos del cliente. La Ingeniería de Requisitos (IR) es la disciplina de Ingeniería del Software dedicada a mejorar la captura, análisis, especificación, validación y gestión de requisitos del software. Para esto, la IR estudia y define métodos, técnicas, notaciones y herramientas para garantizar una adecuada definición de los requisitos.

### 1.3.1 Modelo Integrado de Madurez de Capacidades (CMMI)

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (7)

En el mes de enero del año 2002, el Instituto de Ingeniería del Software de la Universidad Carnegie Mellon (SEI), publicó su primera versión del Modelo Integrado de Madurez de Capacidades (CMMI), desarrollado para la mejora o evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de software.

Este modelo describe seis áreas de procesos ingenieriles. Las dos primeras abordan las actividades de la IR: Desarrollo de Requisitos, Gestión de Requisitos, Solución Técnica, Integración del Producto, Verificación y Validación. Para las áreas de proceso: Desarrollo de Requisitos y Gestión de Requisitos, CMMI establece los Objetivos y Prácticas Específicos. A continuación son explicadas y ejemplificadas cada una de las prácticas por objetivo.

**Objetivos y Prácticas Específicas para: Desarrollo de Requisitos. (Nivel de Madurez 3)**

*OE 1 Obtener requisitos del cliente*

PE 1.1 Obtener Necesidades

PE 1.2 Elaborar Requisitos del Cliente

*OE 2 Obtener Requisitos del Producto*

PE 2.1 Establecer Requisitos del Producto y de los Componentes del Producto

PE 2.2 Ubicar Requisitos de Componentes del Producto

PE 2.3 Identificar Requisitos de Interfaz

*OE 3 Analizar y Validar Requisitos*

PE 3.1 Establecer Conceptos Operacionales y Escenarios

PE 3.2 Establecer Definición de Funcionalidades Requeridas

PE 3.3 Analizar Requisitos

PE 3.4 Analizar Requisitos para Alcanzar Equilibrio

PE 3.5 Validar Requisitos

**Objetivos y Prácticas Específicas para: Gestión de Requisitos. (Nivel de Madurez 2)**

*OE 1 Gestionar Requisitos*

PE 1.1 Obtener un Entendimiento de los Requisitos

PE 1.2 Obtener Compromiso con los Requisitos

PE 1.3 Gestionar Cambios en los Requisitos

PE 1.4 Mantener Trazabilidad Bidireccional de los Requisitos

PE 1.5 Identificar Inconsistencias entre el Trabajo del Proyecto y los Requisitos.

### 1.3.2 El modelo de Pohl

El modelo de Pohl es un modelo iterativo, en el que se describen cuatro etapas que pueden verse en la figura1.1, aunque el orden de las actividades puede ser cualquiera, se asume una secuencia en la que los requisitos son elicitados, a continuación son negociados entre los participantes se integran con el resto de la documentación y posteriormente se validan. (8)


Figura 1.1: Modelo de actividades de Ingeniería de Requisitos de Pohl.

Este modelo presenta cuatro características principales que se explican a continuación: (9)

*Elicitación de requisitos:* el objetivo de la elicitación es mostrar el conocimiento oculto sobre las necesidades de clientes y usuarios y el sistema a desarrollar, para que sean de fácil comprensión para todos los participantes en el problema. En este modelo, durante el desarrollo de las actividades, es necesario identificar a las fuentes de información, conocer lo mejor posible el dominio del problema, reutilizar especificaciones de requisitos similares en la medida de lo posible y utilizar las técnicas habituales de elicitación como son las entrevistas, casos de uso, cuestionarios, prototipos, etc.

*Negociación de requisitos:* tiene como objetivo lograr acuerdos entre todos los participantes sobre los requisitos elicitados, avanzando en la dimensión de acuerdo del proceso. Para ello hay que tener en cuenta cuatro factores:

- hacer explícitos los conflictos y evitar los conflictos emocionales entre los participantes, de forma que quede claro qué es lo que se negocia y que dicha negociación no se vea afectada por motivos personales
- hacer explícitos para cada conflicto las alternativas, las argumentaciones y las razones subyacentes que los provocan, de forma que la negociación pueda basarse en las raíces del conflicto
- asegurarse de que se toman las decisiones correctas, de forma que la mayoría de los participantes estén de acuerdo en los resultados de la negociación y no se sientan desplazados del proceso
- asegurarse de involucrar a las personas adecuadas en el momento adecuado, para evitar tener que volver a replantear las negociaciones, porque alguno de los participantes afectados no participó en las negociaciones oportunas.

*Especificación /documentación de requisitos:* el objetivo es claro, documentar los requisitos elicitados y negociados. Propone utilizar las notaciones que sean necesarias para que todos los participantes la entiendan. De este modo, según su propuesta, se avanza en la dimensión de formalidad del proceso.

*Validación/Verificación de requisitos.* El objetivo es comprobar que los requisitos documentados corresponden a las necesidades de los clientes y usuarios (validación) y comprobar que la

especificación cumple los criterios de calidad oportunos (verificación).

### 1.3.3 El modelo espiral

El modelo espiral asume una naturaleza iterativa del proceso y la dificultad de establecer un punto de terminación del mismo dado que los requisitos nunca llegarán a ser perfectos.

Además de las actividades que se muestran en la figura 1.2, al proceso se adjudica una cuarta etapa dedicada a la gestión de requisitos, que se realiza durante todo el proceso y se encarga de gestionar la obtención incremental de los requisitos y los necesarios cambios a los que están sujetos. (10)

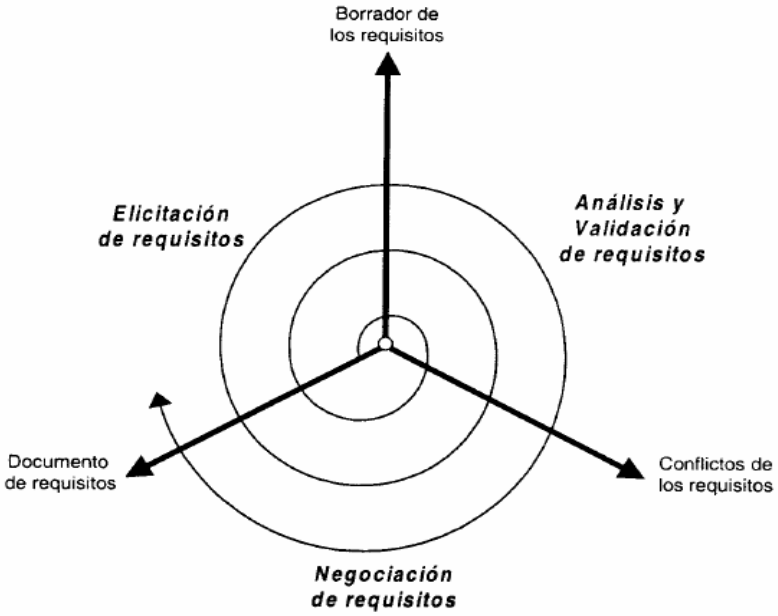


Figura 1.2: Modelo espiral

*Elicitación de requisitos:* En esta actividad, se consultan las diferentes fuentes de información como clientes, usuarios, expertos en dominio, etc, con el propósito de entender el dominio del problema y establecer los requisitos del sistema a desarrollar, obviamente, no es factible que los requisitos elicitados no sean completos y que se expresen en forma vaga o no estructurada.

*Análisis y validación de requisitos:* Los requisitos elicitados se integran y analizan, lo que suele provocar la identificación de requisitos que faltan, inconsistencias y conflictos entre los requisitos. Aunque se denomina a esta actividad análisis y validación de requisitos, sin embargo no se hace referencia a ningún tipo de actividad de validación tal como se entiende el término en esta tesis, es decir, la confirmación por parte de clientes y usuarios de que los requisitos reflejan realmente sus necesidades y especifican el sistema que ellos desean.

*Negociación de requisitos:* Su objetivo es solucionar todos los conflictos que se hayan presentado durante el análisis, llegando a acuerdos entre los participantes en el proceso. Normalmente hay que elicitar nuevamente. (11)

### 1.3.4 Modelo de Referencia para la Ingeniería de Requisitos en proyectos de Bioinformática (INeR)

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (12)

Este modelo es propuesto por Karina Pérez Teruel, profesora de la Universidad de las Ciencias Informáticas (UCI) en su tesis de maestría. La autora propone el Modelo de Referencia para la Ingeniería de Requisitos en proyectos de Bioinformática (INeR) debido a que en una encuesta realizada, se obtuvo como resultado, que los modelos de IR tradicionales, usados en este tipo de proyecto, atentaban contra el cumplimiento de los objetivos de esta disciplina. Este hecho provoca que se excedan el tiempo y presupuesto asignados. Además en la bibliografía consultada, no se encontraron adaptaciones de la disciplina de IR a las necesidades de la Bioinformática.

Este modelo esta compuesto fundamentalmente por:

- *Procesos:* Marco de procesos que engloban las actividades relacionadas con requisitos en el proceso de desarrollo de software.
- *Técnicas:* Conjunto de técnicas adecuadas al tipo de proyecto en cuestión usadas con el objetivo de aportar eficiencia a los procesos del modelo.
- *Herramientas:* Selección de herramientas que permiten automatizar o semi-automatizar los procesos tomando en cuenta las técnicas específicas del modelo.

Lo cual se muestra en la siguiente figura:

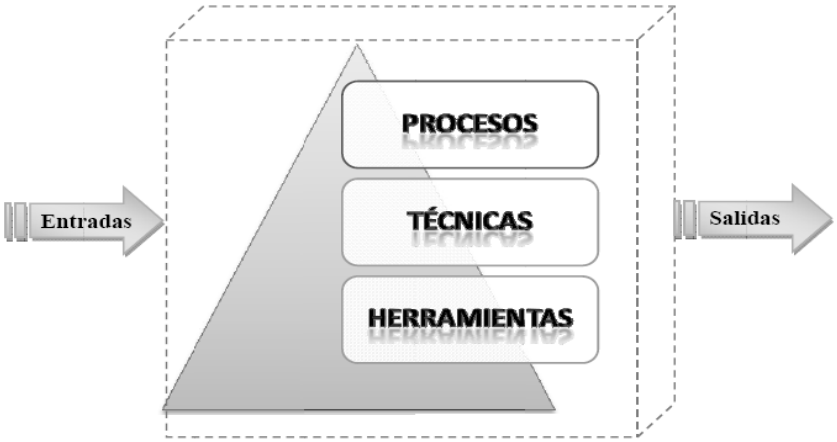


Figura 1.3: Modelo INeR.

El modelo define un conjunto de objetivos, entradas y salidas tales como:

*Objetivos:*

- Establecer y mantener el acuerdo entre los involucrados del proyecto referente a los requisitos funcionales y no funcionales del sistema.
- Proporcionar a los desarrolladores de un mejor entendimiento de los requisitos del sistema.
- Definir las fronteras del sistema.
- Proporcionar una base para estimaciones y planificación de costo y tiempo de desarrollo del sistema.

*Entradas*

- Objetivos de la organización
- Modelo del Negocio (opcional)
- Solicitudes de cambios en requisitos
- Información organizacional relacionada al proyecto
- Políticas organizacionales relacionadas al proyecto

*Salidas*

- Plan de Gestión de Requisitos
- Listado de requisitos y sus atributos
- Especificación de Requisitos
- Estado de los Requisitos

*Los procesos que define el modelo INeR son:*

1. Inicio del Proyecto (IP)
2. Obtención de Requisitos (OR)
3. Análisis de Requisitos (AR)
4. Descripción de Requisitos (DR)
5. Gestión de Requisitos (GR)

Los flujos de trabajo definidos para el marco de procesos de INeR, tienen como punto inicial, el comienzo de un proyecto de desarrollo de software bioinformático. Se agrupan los cuatro primeros procesos en una categoría denominada: Desarrollo de Requisitos, debido a que estos procesos están enfocados a establecer lo que debe hacer el sistema. El proceso de gestión de requisitos va dirigido a evolucionar los requisitos desarrollados conforme evoluciona el proyecto. Las salidas de los procesos de desarrollo de requisitos constituyen entradas fundamentales para el resto de los flujos de trabajo ingenieriles. El proceso de gestión de requisitos se mantiene a lo largo del ciclo de vida del proyecto.

De los modelos antes expuestos para la IR, se concluye que el más apropiado para aplicar en el desarrollo de la solución al problema planteado, es el Modelo de Referencia para la Ingeniería de Requisitos en proyectos de Bioinformática (INeR), pues se ajusta mejor a las características específicas del proyecto. Tanto CMMI, como el modelo de Pohl y el modelo espiral son muy generales, pues de hecho, son modelos que se aplican a nivel internacional, a una gran variedad de proyectos.

### 1.4 Arquitectura

La Arquitectura de Software no es más que un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un

sistema de información. La misma establece los fundamentos para que analistas, diseñadores, programadores, trabajen en una línea común que permita alcanzar los objetivos del sistema de información, cubriendo todas las necesidades.

El desarrollo del software para el Centro Nacional de Urgencias Médicas (CNUM) estará guiado por una arquitectura basada en componentes y orientada a servicios. Esta arquitectura permitirá que el sistema informático esté compuesto por tres módulos o aplicaciones diferentes tales como: Módulo de Emergencia Médica Nacional, Módulo Centro Coordinador Provincial y el Módulo Operaciones, los cuales se integrarán a través de diversos servicios web que brindarán y/o consumirán cada uno de ellos.

Además, dichas aplicaciones se pueden encontrar en un mismo nodo (computadora) o distribuidas en nodos diferentes. A su vez, estos módulos se integrarán a varios componentes de SiSalud (Sistema de Información para la Salud) tales como: SAAA (Servicio de Autenticación, Autorización y Auditoria), RPS (Registro de Personal de la Salud), RUS (Registro de Unidades de Salud), RU (Registro de Ubicación) y RC (Registro de Ciudadanos), consumiendo algunos de los servicios que brindan estos registros según las necesidades de la aplicación.

**1.5 Metodologías ágiles de desarrollo de software**

Los procesos ágiles de desarrollo de software, conocidos anteriormente como metodologías livianas, intentan evitar los tortuosos y burocráticos caminos de las metodologías tradicionales enfocándose en la gente y los resultados.

Estas metodologías están enfocadas al individuo y a las interacciones del equipo de desarrollo sobre el proceso y las herramientas. Desarrollan software que funciona pero no generan una buena documentación.

Las metodologías ágiles están basadas en heurísticas provenientes de prácticas de producción de código, preparadas para cambios durante el proyecto y el cliente es parte del equipo de desarrollo. Los grupos de trabajo están diseñados con poco personal trabajando todos en el mismo sitio, generan pocos artefactos, poseen pocos roles, además de no hacer énfasis en la arquitectura del software. Entre ellas se encuentra: eXtreme Programing (XP), Crystal Methods, Feature Driven

Development (FDD).

### 1.5.1 eXtreme Programming (XP)

Es la más destacada de los procesos ágiles de desarrollo de software. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto, es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software. (13)

Las características fundamentales del método son: (14)

- Desarrollo iterativo e incremental, pequeñas mejoras, unas tras otras.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión.
- Programación en parejas, las tareas de desarrollo se llevan a cabo por dos personas en un mismo puesto. Se supone que la calidad del código escrito de esta manera es mayor (el código es revisado y discutido mientras se escribe), esto es más importante que la posible pérdida de productividad inmediata.
- Frecuente interacción del equipo de programación con el cliente o usuario, un representante del cliente trabaja junto al equipo de desarrollo.
- Corrección de todos los errores antes de añadir una nueva funcionalidad. Obliga a realizar entregas frecuentes.
- Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- Propiedad del código compartida, en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas

de regresión garantizan que los posibles errores sean detectados.

- Simplicidad en el código, es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario.

La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

### 1.5.2 Crystal Methods

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (15)

Las metodologías Crystal fueron creadas por el antropólogo de proyectos Alistair Cockburn, esta familia dispone de un código de color para marcar la complejidad de una metodología: cuanto más oscuro un color, más pesado es el método. Cuanto más crítico es un sistema, más rigor se requiere. El código cromático se aplica a una forma tabular elaborada por Cockburn que se usa en muchos metodologías ágiles (MAs) para situar el rango de complejidad al cual se aplica una metodología.

Los métodos se llaman Crystal evocando las facetas de una gema: cada faceta es otra versión del proceso y todas se sitúan en torno a un núcleo idéntico. Existen cuatro variantes de metodologías: Crystal Clear (Claro como el cristal) para equipos de 8 o menos integrantes; amarillo para 8 a 20; naranja para 20 a 50; rojo para 50 a 100. La más exhaustivamente documentada es esta.

Crystal Clear (CC) puede ser usado en proyectos pequeños. Como casi todos los otros métodos, consiste en valores, técnicas y procesos. Está basado en siete propiedades fundamentales: Entrega frecuente, comunicación osmótica, mejora reflexiva, seguridad personal, fácil acceso a usuarios expertos, foco y ambiente técnico con prueba automatizada.

No requiere ninguna estrategia o técnica, pero es conveniente tener unas cuantas a mano para empezar. Entre las más comunes a las metodologías ágiles (MAs) se pueden mencionar: victoria temprana y rearquitectura incremental. La primera, basada en buscar pequeños triunfos iniciales y no aspirar a una gran victoria tardía, plantea además, que no es conveniente utilizar la técnica de lo

peor primero de XP. La preferencia de Cockburn es lo más fácil primero, lo más difícil segundo. La segunda, prevé que no es conveniente interrumpir el desarrollo para corregir la arquitectura, más bien la arquitectura debe evolucionar en etapas, manteniendo el sistema en ejecución mientras ella se modifica.

Una de las técnicas que propone y de la cual se obtiene una comparación con XP es la programación lado a lado, muchas personas sienten que la programación en pares de XP involucra una presión excesiva; la versión de Crystal Clear establece proximidad, pero cada quien se aboca a su trabajo, prestando un ojo a lo que hace su compañero, quien tiene su propia máquina.

La mayoría de los modelos de proceso o metodologías propuestos entre 1970 y 2000 se describían como secuencias de pasos. Aún cuando se recomendaran iteraciones e incrementos, los modelos parecían dictar un proceso en cascada, por más que los autores aseguraran que no era así. El problema con estos procesos es que realmente están describiendo un flujo de trabajo requerido, un grafo de dependencia: el equipo no puede entregar un sistema hasta que esté integrado y corra. No puede integrar y verificar hasta que el código no esté escrito y corriendo. Y no puede diseñar y escribir el código hasta que se le dice cuáles son los requerimientos. Un grafo de dependencia se interpreta necesariamente en ese sentido, aunque no haya sido la intención original.

En lugar de esta interpretación lineal, CC enfatiza el proceso como un conjunto de ciclos anidados. En la mayoría de los proyectos se perciben siete ciclos: el proyecto, el ciclo de entrega de una unidad, la iteración (requiere múltiples entregas por proyecto pero no muchas iteraciones por entrega), la semana laboral, el período de integración (de 30 minutos a tres días), el día de trabajo, el episodio de desarrollo de una sección de código (de pocos minutos a pocas horas). A pesar que no contempla el desarrollo de software propiamente dicho, involucra unos veinte productos de trabajo o artefactos.

Los métodos no prescriben las prácticas de desarrollo, las herramientas o los productos que pueden usarse, pudiendo combinarse con otros métodos como Scrum, XP y Microsoft Solutions Framework. En un comentario que hace el creador de esta metodología expresa que cuando imaginó Crystal Clear pensaba proporcionar un método ligero; comparado con XP, sin embargo, este resulta muy pesado. Es más fácil de aprender e implementar; a pesar de su jerga chocante XP es más disciplinado; pero si un equipo ligero puede tolerar sus rigores, lo mejor será que se mude a XP.

### 1.5.3 Feature Driven Development (FDD)

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (16)

Otro método ágil es FDD, es iterativo y adaptativo. A diferencia de otras MAs, no cubre todo el ciclo de vida sino sólo las fases de diseño y construcción y se considera adecuado para proyectos grandes y de misión crítica.

A su vez esta metodología no requiere un modelo específico de proceso y se complementa con otras metodologías. Enfatiza cuestiones de calidad y define claramente entregas tangibles y formas de evaluación del progreso.

Se utilizó por primera vez en grandes aplicaciones bancarias a fines de la década de 1990. Los autores sugieren su uso para proyectos nuevos o actualizaciones de sistemas existentes, y recomiendan adoptarlo en forma gradual. Aunque no hay evidencia amplia que documente sus éxitos, las grandes consultoras suelen recomendarlo incluso para delicados proyectos de misión crítica.

Los principios de esta metodología son pocos y simples:

- Se requiere un sistema para construir sistemas si se pretende escalar a proyectos grandes.
- Un proceso simple y bien definido trabaja mejor.
- Los pasos de un proceso deben ser lógicos y su mérito inmediatamente obvio para cada miembro del equipo.
- Vanagloriarse del proceso puede impedir el trabajo real.
- Los buenos procesos van hasta el fondo del asunto, de modo que los miembros del equipo se puedan concentrar en los resultados.
- Los ciclos cortos, iterativos, orientados por rasgos son mejores.

## 1.6 Metodologías tradicionales de desarrollo de software

Las metodologías tradicionales de desarrollo de software, están basadas en normas provenientes de estándares, seguidos por el entorno de desarrollo, con cierta resistencia a los cambios. El proceso es mucho más controlado, con numerosas políticas o normas, existe un contrato prefijado y

el cliente interactúa con el equipo de desarrollo mediante reuniones, los grupos de trabajos son grandes y posiblemente distribuidos. Generan gran cantidad de artefactos y roles. La arquitectura del software es esencial y se expresa mediante modelos.

### 1.6.1 Microsoft Solution Framework (MSF)

Microsoft Solution Framework es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. Se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas. (17)

Esta metodología tiene las siguientes características: es adaptable, parecido a un compás, usado en cualquier parte como un mapa. Es escalable, puede organizar equipos tan pequeños entre 3 ó 4 personas, así como también, proyectos que requieren 50 personas o más. Es flexible, puede ser utilizada en el ambiente de desarrollo de cualquier cliente. Es una tecnología agnóstica, porque puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología. (18)

MSF se compone de varios modelos encargados de planificar las diferentes partes implicadas en el desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el Modelo de Aplicación.

### 1.6.2 Rational Unified Process (RUP)

El Proceso Racional Unificado (Rational Unified Process), es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP es en realidad un refinamiento realizado por Rational Software del más genérico proceso unificado.

Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso, incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso). Esta

metodología divide el proceso de desarrollo en ciclos, teniendo un producto al culminar cada ciclo, estos se dividen en fases que finalizan con un hito, donde se debe tomar una decisión importante, las fases son: (19)

Inicio: Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.

Elaboración: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.

Construcción: Se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario.

Transición: Se implementa el producto en el cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados.

RUP puede ser visto como una metodología ágil, pues es tan configurable como se desee.

Las autoras consideran que lo más importante antes de elegir la metodología que se va a usar para la implementación de un software, es determinar el alcance que tendrá y luego de ahí analizar la que más se acomoda a la aplicación.

Haciendo un pequeño resumen se puede concluir que la metodología RUP es la más adaptable al proyecto debido a que está definida para proyectos de largo plazo o en otras palabras de grandes dimensiones, no así XP, pues se recomienda para proyectos de corto plazo. RUP es una metodología que propone y exige el uso de artefactos en cada iteración, característica que le permite al software alcanzar un grado de certificación en el desarrollo del mismo.

Crystal no se adapta como metodología a utilizar, pues ha sido catalogada por su propio creador como una metodología pesada, aunque fácil de aprender e implementar, no es disciplinada, siendo está unas de las principales característica que presenta RUP.

El Proceso Unificado de Software define seis flujos básicos de desarrollo y tres de apoyo, con los

cuales se cubre todo el ciclo de vida del software, FDD, sin embargo es un método ágil, iterativo y adaptativo pero no cumple con esta condición debido a que se centra sólo en las fases de diseño y construcción, por lo que es considerado adecuado para proyectos de misión crítica.

Por otra parte, MSF tiene como desventajas que es un modelo prescriptivo por lo que solicita demasiada documentación en sus fases, el análisis de riesgos es necesario, pero si se hace muy exhaustivo puede demorar o hasta frenar el avance del proyecto, además, al estar basado en tecnología Microsoft, trata de obligar a usar herramientas de Microsoft, aunque es posible no usar esa tecnología, pero ello produce más complejidad en el proyecto.

Por todo lo antes expuesto las autoras consideran que la metodología más apropiada para utilizar es RUP, siendo esta además, con la que más familiarizadas se encuentran, lo cual facilita su puesta en práctica.

Se ha decidido combinar la metodología RUP con el modelo INeR para la Ingeniería de Requisitos debido a la importancia y al papel que juega en el desarrollo de un producto de calidad, la captura, análisis, especificación, validación y gestión de requisitos del software. Es bien sabido que RUP propone una fase completa para los Requerimientos, pero aún así las autoras consideran que es fundamental aplicar el modelo INeR para fortalecer y mejorar los resultados de la disciplina de requisitos y por tanto de los restantes procesos ingenieriles.

### 1.7 UML: Lenguaje Unificado de Modelamiento

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (20)

UML es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

Este lenguaje capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. Un sistema se modela como una colección de objetos discretos que interactúan para realizar un trabajo que finalmente beneficia a un usuario externo. A su vez, pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un

acercamiento estándar.

No es un lenguaje de programación. Las herramientas pueden ofrecer generadores de código de UML para una gran variedad de lenguaje de programación, así como construir modelos mediante ingeniería inversa a partir de programas existentes. Es un lenguaje de propósito general para el modelado orientado a objetos. Es también un lenguaje de modelado visual que permite una abstracción del sistema y sus componentes.

Hace algunos años, existían diversos métodos y técnicas Orientadas a Objetos, con muchos aspectos en común pero utilizando distintas notaciones, se presentaban inconvenientes para el aprendizaje, aplicación, construcción y uso de herramientas, etc., además de pugnas entre enfoques, lo que generó la creación del UML como estándar para el modelado de sistemas de software principalmente, pero con posibilidades de ser aplicado a todo tipo de proyectos.

Tiene como objetivos fundamentales:

- Ser un lenguaje de modelado de propósito general que puede ser usado por todos los modeladores.
- No tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática.
- No pretende ser un método de desarrollo completo. No incluye un proceso de desarrollo paso a paso. Incluye todos los conceptos que se consideran necesarios para utilizar un proceso moderno iterativo, basado en construir una sólida arquitectura para resolver requisitos dirigidos por casos de uso.
- Ser tan simple como sea posible pero manteniendo la capacidad de modelar toda la gama de sistemas que se necesita construir. UML necesita ser lo suficientemente expresivo para manejar todos los conceptos que se originan en un sistema moderno, tales como la concurrencia y distribución, así como también los mecanismos de la ingeniería de software, como son el encapsulamiento y componentes.
- Debe ser un lenguaje universal, como cualquier lenguaje de propósito general.
- Imponer un estándar mundial.

### 1.8 Herramienta de modelado: Visual Paradigm

La herramienta de desarrollo Visual Paradigm, se integra al IDE (Integrated Development Environment, Entorno de Desarrollo Integrado) de Eclipse. Fue diseñada para desarrollar software con OOP (Object Oriented Programming, Programación Orientada a Objetos), busca reducir la duración del ciclo de desarrollo brindando ayuda a arquitectos, analistas, diseñadores y desarrolladores.

Visual Paradigm es un producto distinguido que facilita la organización de los diagramas, su misión es diseñar, integrar y desplegar las aplicaciones de la empresa y sus bases de datos subyacentes. La herramienta ayuda al equipo de desarrollo de software a agilizar el modelado del software, aumentando al máximo y acelerando el trabajo en equipo y las contribuciones individuales.

La herramienta CASE Visual Paradigm posibilita:

- Generación de código (PHP).
- Entorno de creación de diagramas para UML.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Análisis de textos.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad en múltiples plataformas.

Además de todas facilidades para el modelado de software, que brinda la herramienta Visual Paradigm, la cuales fueron reflejadas en las características mencionadas anteriormente, resulta importante destacar que esta herramienta está registrada en el Documento de Arquitectura de la Facultad 7; documento que rige y estandariza la arquitectura de todos los proyectos de la facultad, siendo este otro elemento que fundamenta su selección para modelar los artefactos que

se confeccionán para dar solución al problema planteado.

### 1.8 Lenguaje de programación: PHP (Hypertext Preprocessor)

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (21)

El lenguaje de programación Hypertext Preprocessor, es un lenguaje interpretado de alto nivel, embebido en páginas HTML y ejecutado en el servidor. Una de sus características más potentes, es su soporte para gran cantidad de bases de datos, entre las que se pueden mencionar: InterBase, mSQL, MySQL, Oracle, Informix, PostgreSQL, entre otras.

También ofrece la integración con varias bibliotecas externas, que brindan al desarrollador la posibilidad de realizar cualquier tarea, desde generar documentos en pdf (Portable Document Format) hasta analizar código XML (extensible Markup Language) y también para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando la librería GTK+. Es software libre, lo que implica menos costes y servidores más baratos que otras alternativas. Es muy rápido y su integración con la base de datos MySQL y el servidor Apache, le permite constituirse como una de las alternativas más atractivas del mercado.

Es multiplataforma, funciona tanto para Unix (con Apache) como para Windows (con Microsoft Internet Information Server) de forma que el código que se haya creado para una de ellas, no tiene porqué modificarse al pasar a la otra. Su sintaxis está inspirada en C, ligeramente modificada para adaptarlo al entorno en el que trabaja, de modo que si se está familiarizado con esta sintaxis, le resultará muy fácil aprender PHP. Su librería estándar es realmente amplia, lo que permite reducir los llamados "costes ocultos", uno de los principales defectos de ASP (Active Server Pages).

Tiene una de las comunidades más grandes en Internet, con lo que no es complicado encontrar ayuda, documentación, artículos, noticias, y más recursos. Ofrece una solución simple y universal para las paginaciones dinámicas de la Web de fácil programación. Su diseño elegante lo hace perceptiblemente más fácil de mantener y ponerse al día, a diferencia con el código de otros lenguajes.

Debido a su amplia popularidad esta perfectamente soportado por una gran comunidad de desarrolladores. Como producto de código abierto, goza de la ayuda de un gran grupo de

programadores, permitiendo que los fallos de funcionamiento se encuentren y se reparen rápidamente. El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar sus capacidades.

Su funcionamiento se puede describir a través de los pasos siguientes:

- Escribir en las páginas HTML pero con el código PHP dentro.
- Guardar la página en el servidor Web.
- Un navegador solicita una página al servidor.
- El servidor interpreta el código PHP.
- El servidor envía el resultado del conjunto de código HTML y el resultado del código PHP que también es HTML.

En ningún caso se envía código PHP al navegador, por lo que todas las operaciones realizadas son transparentes al usuario, el código PHP es ejecutado en el servidor y el resultado es enviado al navegador. El resultado es normalmente una página HTML. Por lo que al usuario le parecerá que está visitando una página HTML que cualquier navegador puede interpretar.

Al ser un lenguaje que se ejecuta en el servidor, no es necesario que el navegador lo soporte, es independiente del navegador, pero sin embargo para que sus páginas PHP funcionen, el servidor donde están alojadas debe soportar PHP. Además se encuentra libre en el mercado y se puede acceder a él por medio de Internet.

Constituye una petición del cliente, el desarrollo del producto empleado el lenguaje de programación PHP, además, luego de analizar las características y ventajas que posee, expuestas anteriormente, se considera que es el más apropiado para la implementación del software que se pretende realizar.

**Conclusiones**

Existen varios sistemas a escala internacional que automatizan las diversas funcionalidades del sistema de emergencias y urgencias de un país, pero ninguno se adapta a las características del Sistema Integrado de Urgencias Médicas de Cuba y el software existente a nivel nacional no implementa dentro de sus funcionalidades, todas los procesos que actualmente tienen lugar en un CCPEM. Es preciso entonces, realizar el diseño de una nueva aplicación. Para la ingeniería de

requisitos del proyecto se aplicará el Modelo de Referencia para la Ingeniería de Requisitos en proyectos de Bioinformática (INeR), combinándose con la metodología RUP. Como lenguaje de modelado se utilizará: UML y como herramienta de modelado visual: Visual Paradigm.

## Capítulo 2. Características del sistema

### Introducción

En el presente capítulo se retoman el problema y la situación problémica desde una perspectiva diferente, o sea, profundizando en aspectos tales como: objetivos estratégicos de la organización, procesos de negocio que los soportan, flujo actual de los mismos. Se profundiza además en el objeto de informatización, obteniendo finalmente los requerimientos tanto funcionales como no funcionales con los que debe cumplir el software. También se plasman los resultados de desarrollar los primeros flujos de trabajo que propone RUP, tales como: Modelamiento del Negocio y Requerimientos. Además, se hace énfasis en este último flujo de trabajo aplicando el Modelo de Referencia para la Ingeniería de Requisitos en Proyectos de Bioinformática (INeR).

### 2.1 Objetivos de la organización

El Sistema Integrado de Urgencias Médicas (SIUM), se inserta dentro del Sistema Nacional de Salud y tiene como misión organizar la atención de urgencias, emergencias y no urgencias médicas desde la comunidad, consultorios, policlínicos, coordinaciones de ambulancias de urgencia y emergencia hasta el Sistema de Emergencia y Terapia Hospitalaria. (22)

El Centro Coordinador de Emergencia Médica Nacional, el Centro Coordinador Provincial de Emergencia Médica, las Bases Municipales de Ambulancias, el Centro de Operaciones Nacional y el Centro de Operaciones Provincial, son las piezas funcionales que componen al SIUM y que garantizan día a día la atención a las demandas de urgencias, emergencias y no urgencias médicas de la población.

El presente trabajo tiene como objeto de estudio el proceso de gestión de información en el SIUM y el campo de acción abarca lo relacionado a la gestión de información de las demandas en los Centros Coordinadores Provinciales de Emergencias Médicas de Cuba.

Los CCPEM tienen como objetivo fundamental coordinar y controlar el servicio de atención especializada que se le brinda a la población ante un caso de urgencia, emergencia o no urgencia médica; con el objetivo de evitar fallecimientos, así como elevar la calidad de vida del pueblo cubano,

enalteciendo a la vez el prestigio del SNS.

**2.2 Flujo actual de los Procesos**

En un Centro Coordinador Provincial de Emergencia Médica, diariamente tienen lugar una serie de actividades, las cuales suceden en un orden jerárquico, garantizando, el buen funcionamiento y la respuesta al servicio que solicita la población.

Dentro de los procesos fundamentales que han sido identificados en el entorno donde se enmarca el problema se encuentran:

- Procesar demanda
- Archivar certificados
- Confeccionar listado de demandas NU
- Procesar demanda NU
- Controlar móvil de NU
- Controlar trabajo diario

A continuación se ofrecen detalles acerca de cada uno de estos procesos.

Diariamente en un CCPEM, se recepcionan llamadas a través de las cuales una persona solicita atención especializada, en el menor tiempo posible, a un caso de emergencia o urgencia de la población. La llamada realizada a través del número 104 es recepcionada por una enfermera coordinadora, esta realiza un protocolo de preguntas con el objetivo de discriminar la demanda, o sea, determinar si es o no, un caso de emergencia o urgencia médica. En caso de ser una urgencia, recoge una serie de datos y determina el tipo de ambulancia y la institución de destino a la cual será llevado el paciente.

Posterior a ello entrega la solicitud de demanda (datos del paciente recepcionados en una tirilla de papel) al controlador encargado de la región o municipio a la cual pertenece el paciente. El controlador consulta las ambulancias disponibles por base y determina la ambulancia específica que trasladará el caso. Luego comunica los datos de la demanda al chofer de la ambulancia o al expedidor de la base, en la cual se encuentra el móvil, para que la demanda sea ejecutada. Una vez que la

ambulancia comienza a ejecutar la demanda, el controlador le da seguimiento a la misma controlando al móvil a través de una serie de claves, que son comunicadas por planta.
Este conjunto de actividades, con una secuencia lógica, que han sido descritas anteriormente fue identificado como: ***procesar demanda***.

En los centros coordinadores actualmente se atienden también casos no urgentes. Los cuales son asumidos por un personal específico el cual ha sido contratado para esta función y cuentan con la profesión de enfermería.

En la no urgencia (NUa) se recepcionan certificados de personas que, por su padecimiento, no pueden trasladarse a una institución de salud a recibir un servicio médico. De dichos certificados se escogen un conjunto de datos significativos y se confecciona una ficha (llamada certificado por oficio) de cada paciente, la cual es archivada en orden alfabético y tiene además un estado que representa si el certificado está activo o inactivo. El técnico de la NUa es el encargado de ***archivar el certificado***.

Las demandas no urgentes son procesadas a partir de la ***confección del listado de demandas no urgentes***. Este proceso consiste en adicionar al listado de demandas no urgentes, los pacientes que no pueden trasladarse por sus propios medios hacia una institución de salud a recibir atención médica periódicamente y que con anterioridad han entregado en la no urgencia un certificado, que ha sido archivado, como constancia de que presentan un padecimiento que se los impide. El listado se confecciona por municipios o regiones y teniendo en cuenta aspectos tales como: el diagnóstico del paciente, la hora en que debe ser recogido, ya sea en una dirección particular o en una institución, la posición en que debe ser trasladado, o sea, si puede ir sentado o acostado.

Una vez adicionados estos casos que presentan certificado al listado, se pueden seguir adicionando casos de la población y de instituciones médicas que llamen a la NUa solicitando el traslado de un paciente. Esta solicitud se realiza de forma telefónica y es atendida por enfermeras coordinadoras de la NUa, que van ubicando las demandas NU en los horarios que aún quedan disponibles. Una vez que son las 4:00 pm, culmina el horario de recepción de demandas y comienzan a llamar por teléfono a las bases de ambulancias para informar el listado de demandas no urgentes del día siguiente. Luego entregan este listado al controlador y este se encarga de agrupar los pacientes que serán trasladados al unísono como parte de la ejecución de una misma demanda. Este proceso fue identificado como: ***procesar demandas NU.***

Es interés del CCPEM la prestación de un buen servicio. Es por ello que, con el fin de que el transporte que brinda servicio a la población sea utilizado exclusivamente para las cuestiones laborales para las que ha sido destinado, el mismo es controlado rigurosamente por los controladores tanto de la urgencia como de la no urgencia.

Diariamente los controladores de la no urgencia, cuando inician su jornada de trabajo, reciben el listado de demandas no urgentes del día y a partir de ese momento comienzan a dar seguimiento a las ambulancias, que van ejecutando las demandas no urgentes. Los choferes de las ambulancias en cuanto inician la ejecución de una demanda, se comunican por planta con los controladores a través de una serie de claves, las cuales permiten ***controlar al móvil de no urgencias***. El controlador registra la hora en que cada ambulancia pasa por las diferentes claves durante la ejecución de una determinada demanda.

Con el objetivo de ***controlar el trabajo diario*** que se realiza en un CCPEM, el jefe del CCPEM diariamente pide al jefe de guardia una serie de informes tales como: el Parte Diario, la Hoja de cargo, y las Enfermedades Trazadoras. El jefe de guardia revisa las demandas procesadas por la emergencia y urgencia, y confecciona los reportes del día para entregarlos al jefe del CCPEM.

**2.3 Objeto de informatización**

Al analizar el flujo actual de los procesos descritos anteriormente las autoras han determinado el objeto a informatizar, que será el hilo conductor en el diseño de una aplicación web que facilite el proceso de gestión de información de las demandas en los Centros Coordinadores Provinciales de Emergencias Médicas de Cuba.

Se pretende que el sistema gestione las demandas de emergencias y urgencias, o sea, que brinde la posibilidad de Insertar, Listar y Modificar demandas de emergencia y urgencia. También permitirá gestionar los certificados que son recepcionados en la NUa, permitiendo: Insertar, Buscar, Modificar y Visualizar certificados. A su vez se gestionarán las demandas NU facilitando que estas puedan ser: Insertadas, Buscadas, Modificadas y Visualizadas.

De igual forma se gestionarán nomencladores tales como: la clave, el problema médico, la clasificación del problema médico y el estado de conciencia, los cuales facilitarán la entrada de información al

sistema y a su vez harán que este sea más configurable, permitiendo Insertar, Listar, Modificar y Eliminar cada uno de ellos. También se gestionarán los usuarios con acceso al sistema, implementando las siguientes funcionalidades: Autenticar, Registrar, Listar, Modificar, Eliminar, Buscar usuario del Registro de Ciudadanos, Listar usuarios con el rol de controlador y Asignar un municipio a un controlador.

Como parte del resumen del trabajo diario en un CCPEM diariamente se van a generar cuatro reportes, tales como: el Parte Diario, la Hoja de Cargo, las Enfermedades Trazadoras y la Entrega de Guardia de la No Urgencia. El sistema además, se integrará al SiSalud al consumir servicios de varios componentes del mismo como: el SAAA, el RPS, el RUS, el RU y el RC, contribuyendo de esta forma a que exista una mayor integración entre las diferentes aplicaciones que han sido creadas para el Sistema Nacional de Salud.

### 2.4. Modelo de Negocio

El primer flujo de trabajo que propone RUP para el desarrollo de un sistema informático es el Modelamiento del Negocio. El cual tiene como objetivos fundamentales:

- Comprender la estructura y la dinámica de la organización en la cual se va a implantar un sistema.
- Comprender los problemas actuales de la organización e identificar las mejoras potenciales.
- Asegurar que los consumidores, usuarios finales y desarrolladores tengan un entendimiento común de la organización.
- Derivar los requerimientos del sistema que va a soportar la organización.

Para lograr esos objetivos el proceso de modelamiento posibilita obtener una visión de la organización, que permite definir los procesos, roles y responsabilidades de la organización en los modelos de casos de uso del negocio y de objetos. (23)

Si los procesos del negocio pueden ser fácilmente identificados, entonces se procede a la realización del Modelo de Negocio, pero si ocurre lo contrario, o sea, que los procesos no pueden ser reconocidos con claridad entonces se hace un Modelo de Dominio, el cual recoge los principales conceptos que se identifican en el entorno donde se enmarca el problema. En el caso del campo de acción de este

trabajo los procesos han sido fácilmente identificados por lo que se realizará el Modelo de Negocio para posteriormente pasar al flujo de trabajo: Requerimientos.

### 2.4.1 Actores del Negocio

Un actor del negocio es cualquier individuo, grupo, entidad, organización, máquina o sistema de información externos; con los que el negocio interactúa. Lo que se modela como actor es el rol que se juega cuando se interactúa con el negocio para beneficiarse de sus resultados. (24) En la figura 2.1 se puede apreciar la descripción de los actores del negocio.

| Actor | Descripción |
|---|---|
| Demandante | Se refiere a la persona que hace la llamada al Centro Coordinador para solicitar ayuda, puede ser un médico, el paciente o algún familiar del mismo y es el principal beneficiado cuando se procesa una demanda de emergencia , de urgencia o no urgente, así como cuando se archiva un certificado. |
| Jefe_del_Centro_Coordinador | Se refiere al director o Jefe del CCPEM y es el principal beneficiado cuando se controla el trabajo diario del CCPEM. |
| Enfermera_Coordinadora_NU | Se refiere a la Enfermera _coordinadora de la no urgencia y es la principal beneficiada de la confección del listado de demandas no urgentes. |
| CCPEM | Se refiere al Centro Coordinador Provincial de Emergencia Médica como institución estatal, siendo el principal beneficiado del control del móvil de la NUa. |

Figura 2.1: Actores del Negocio

### 2.4.2 Trabajadores del Negocio

Un trabajador del negocio es la abstracción de una persona o sistema software que representa un rol desempeñado en las realizaciones de CUN. Un trabajador del negocio colabora con otros trabajadores, es notificado de los eventos del negocio y manipula las entidades del negocio para realizar sus responsabilidades. En la figura 2.2 se puede apreciar la descripción de los trabajadores del negocio.

| Trabajador | Descripción |
|---|---|
| Enfermera_coordinadora | Es la enfermera que atiende las demandas de Emergencia y Urgencia por el número telefónico: 104 y asigna la institución y el tipo de móvil (ambulancia) a las mismas. |
| Enfermera_coordinadora_NU | Es la enfermera encargada de recepcionar las llamadas de la parte de la no urgencia y de procesar las demandas no urgentes que son recibidas durante el horario establecido para ello. |
| Controlador | Es la persona que ejecuta la demanda, o sea, recibe la demanda que envía la enfermera_coordinadora, decide la ambulancia específica que realizará el traslado y lo comunica por planta al chofer de la ambulancia si esta se encuentra en la calle, pero si la ambulancia está en la base, se comunica con el expedidor de la misma para que de salida al móvil y ejecute la demanda. Posterior a ello, da seguimiento a la ambulancia a través de un conjunto de claves. |
| Controlador_NU | Se comunica por planta con la ambulancia para darle seguimiento. |
| Técnico_NU | Es el encargado de recepcionar los certificados y archivar, en tarjetas ordenadas alfabéticamente, los datos más importantes de los mismos. Además confecciona el listado de demandas no urgentes. |
| Jefe_de_Guardia | Controla el trabajo del Centro Coordinador a través de los reportes diarios que confecciona y además presta ayuda a las coordinadoras en caso de que surja algún problema que ellas no sepan darle solución. |

Figura 2.2: Trabajadores del Negocio

### 2.4.3 Diagrama de Casos de Uso del Negocio

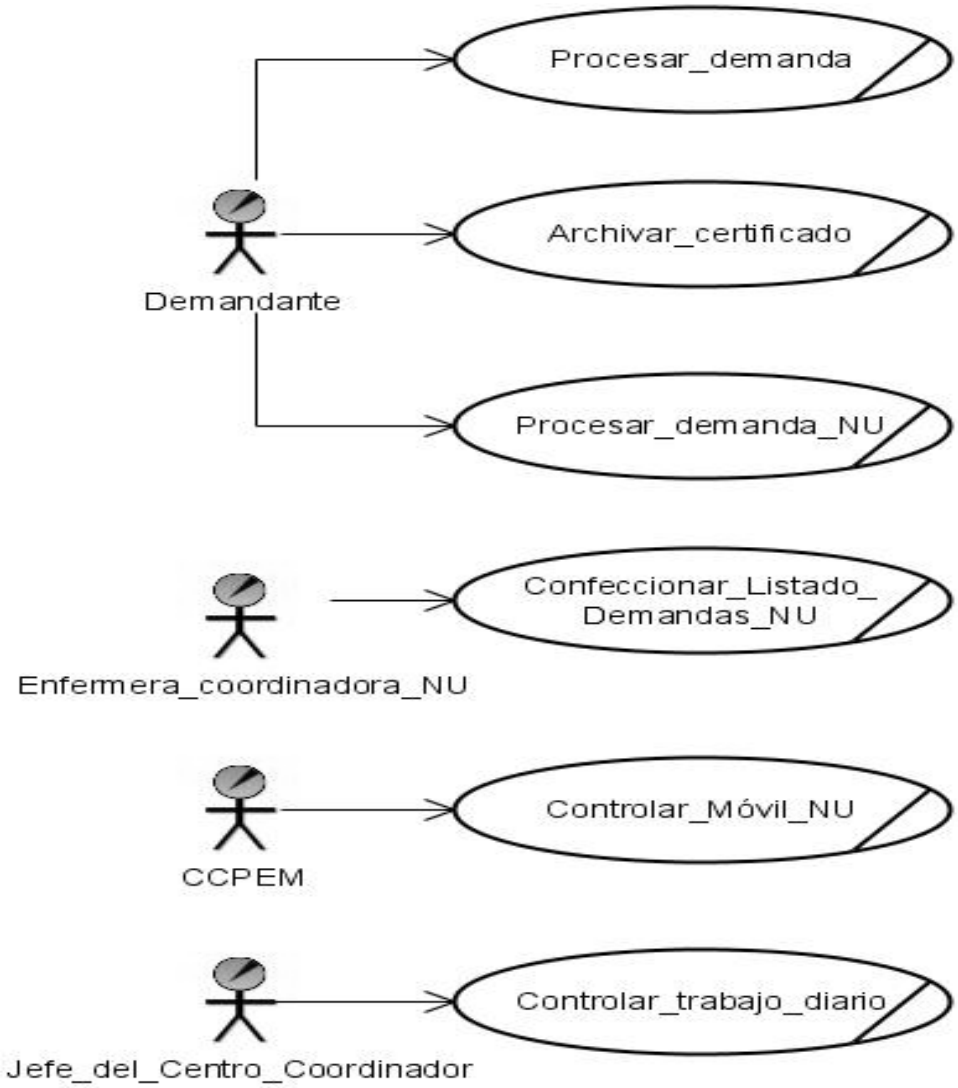


Figura 2.3: DCU del Negocio.

### 2.4.4 Diagramas de actividades

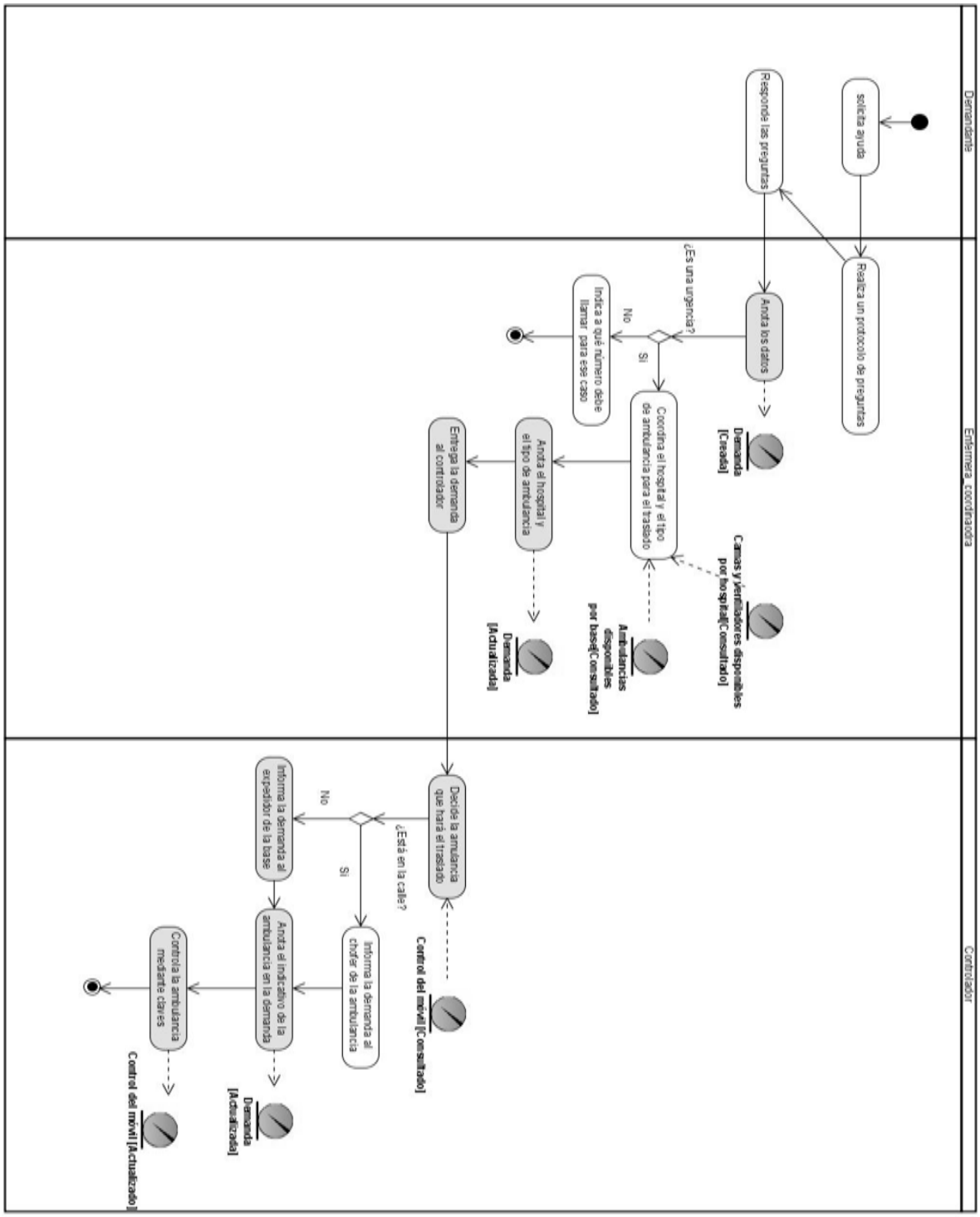


Figura 2.4: Diagrama de actividad CU <Procesar demanda>

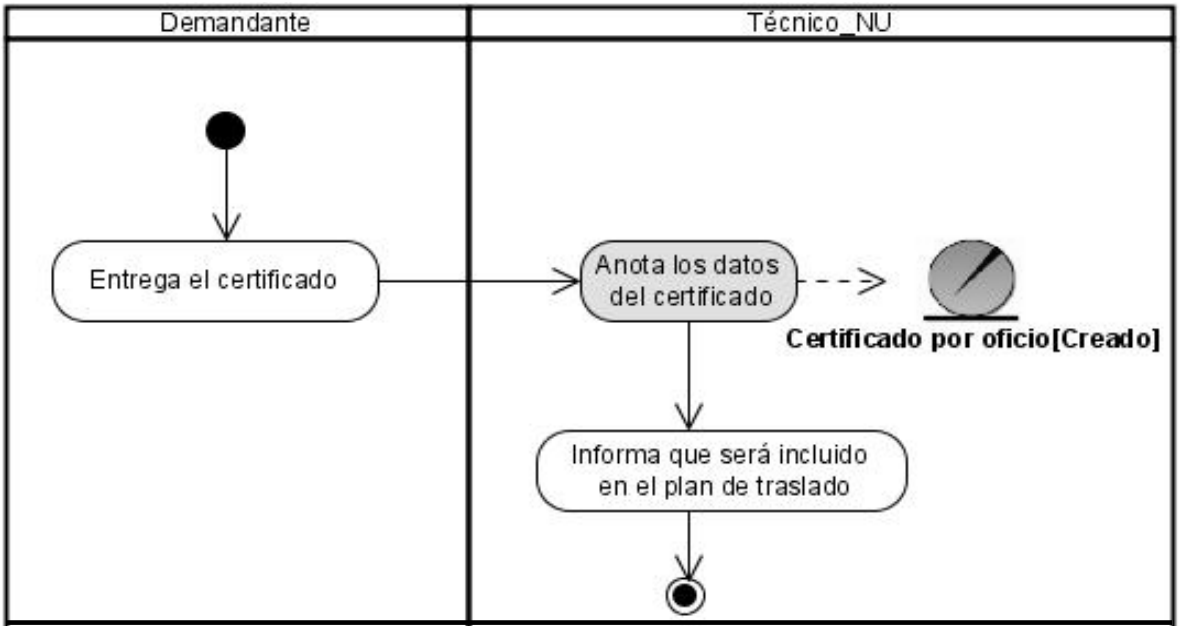


Figura 2.5: Diagrama de actividad CU <Archivar certificado>

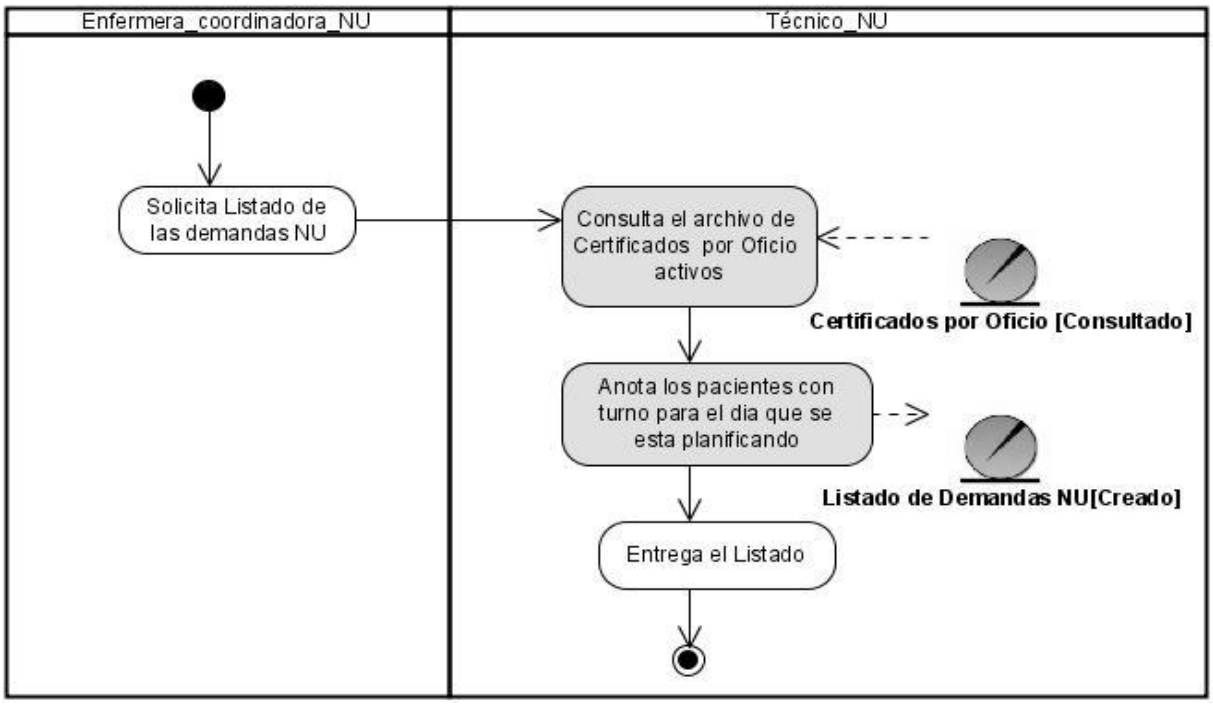


Figura 2.6: Diagrama de actividad CU <Confeccionar Listado de Demandas NU>

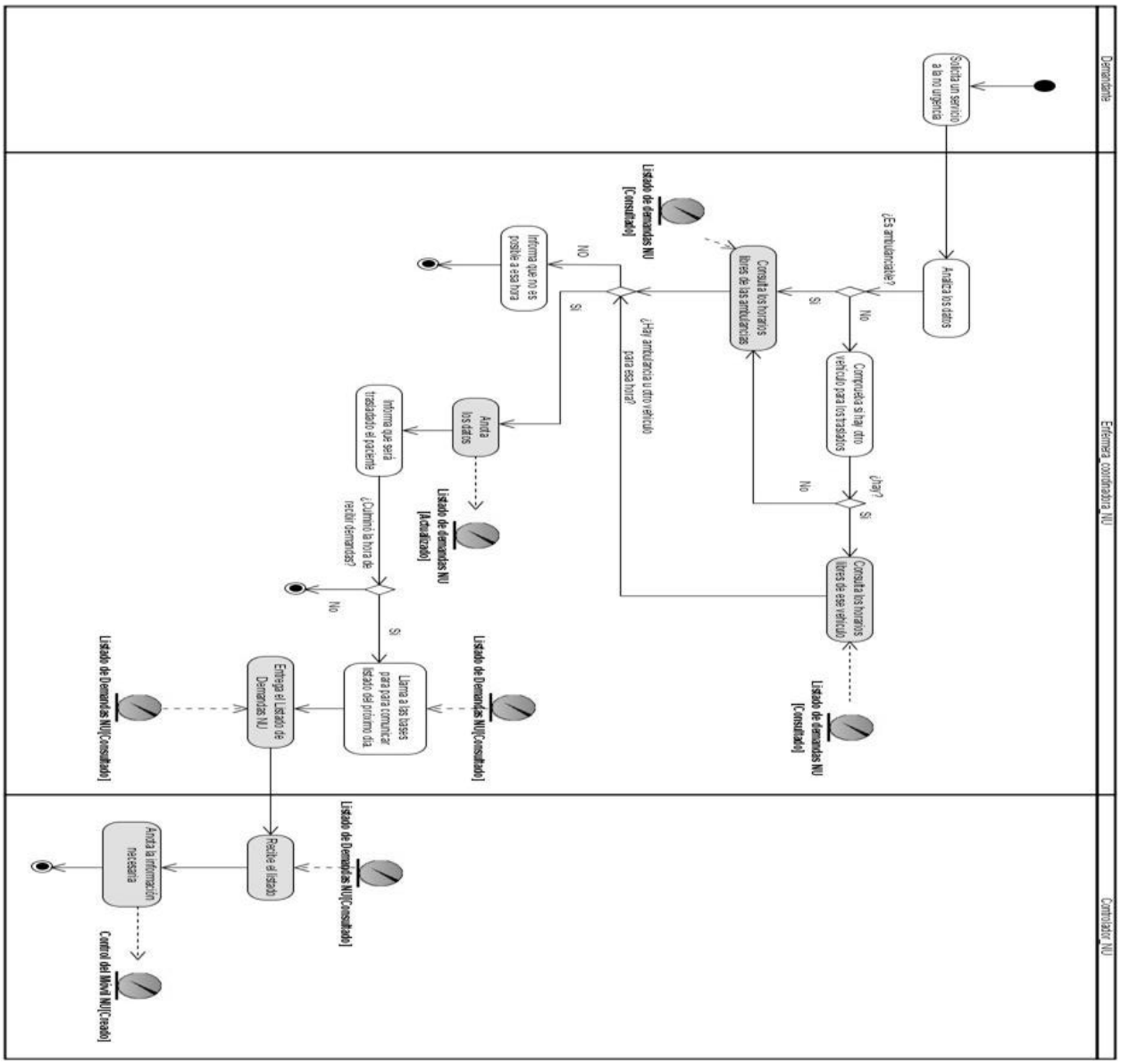


Figura 2.7: Diagrama de actividad CU <Procesar demanda NU>

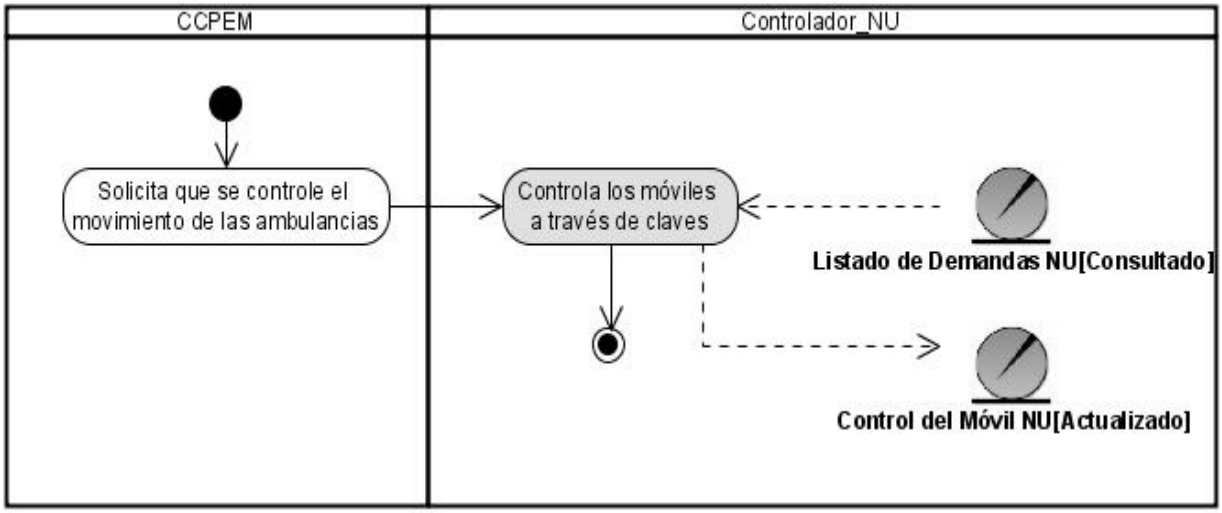


Figura 2.8: Diagrama de actividad CU <Controlar Móvil de NU>

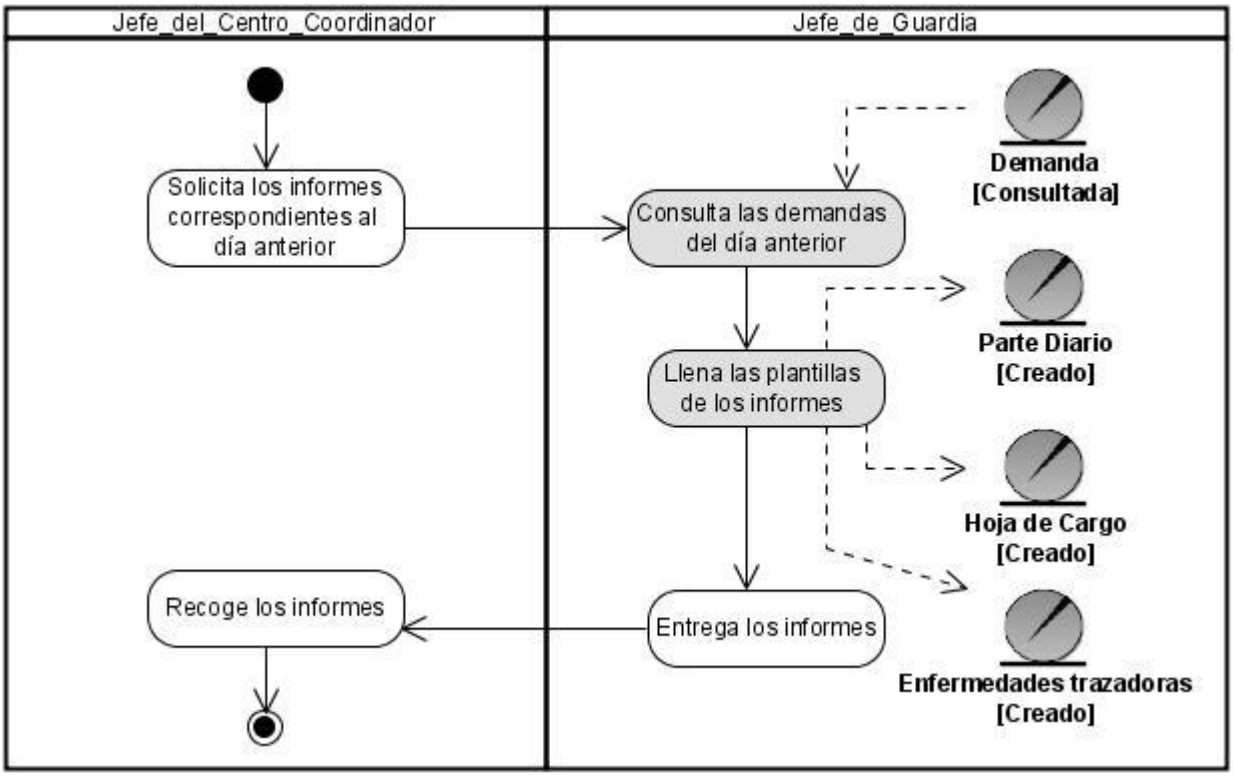


Figura 2.9: Diagrama de actividad CU <Controlar trabajo Diario>

### 2.4.5 Modelo de Objetos del Negocio

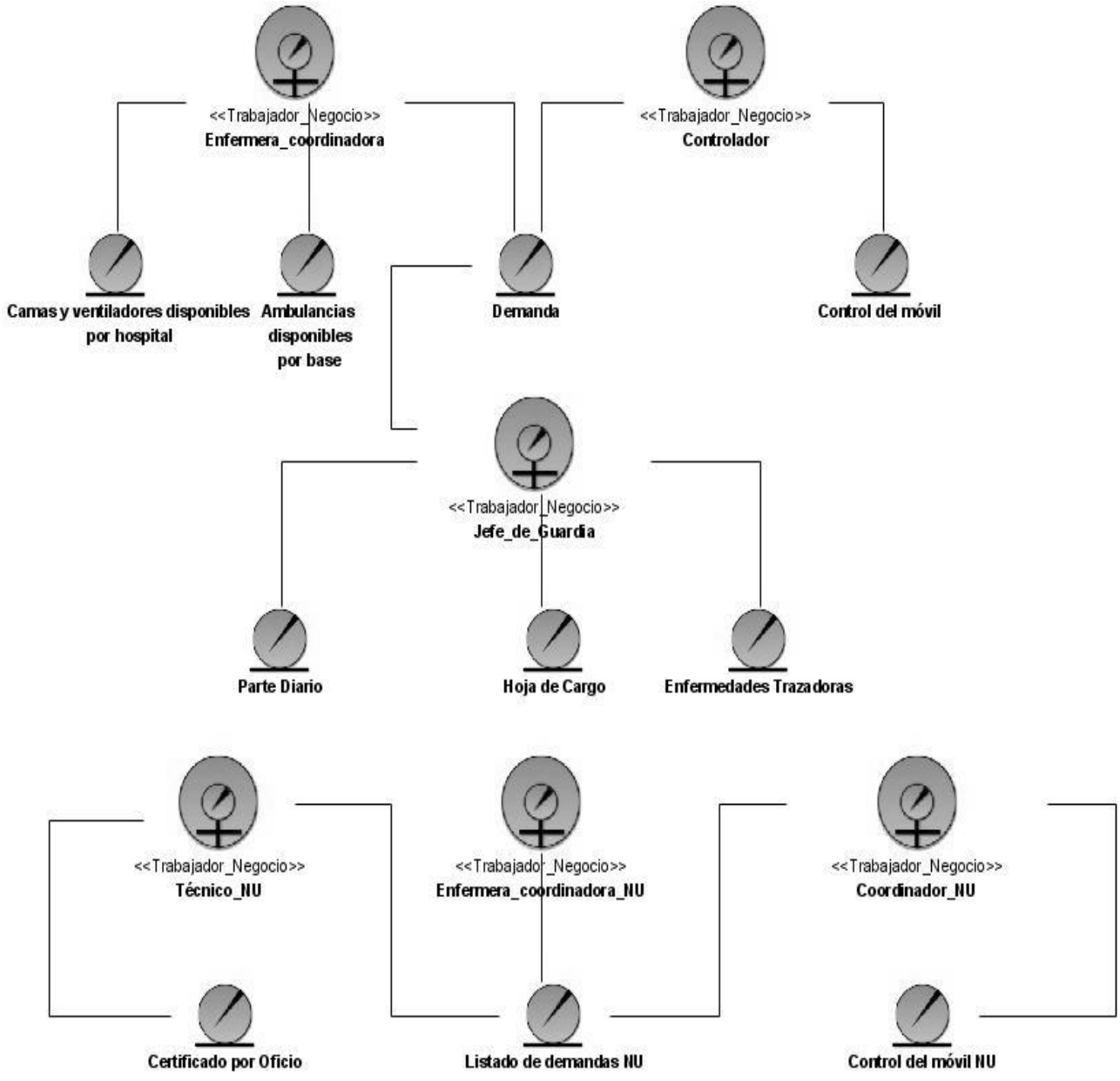


Figura 2.10: Modelo de Objetos del Negocio

### 2.5 Resultados de la aplicación de INeR para la Ingeniería de Requisitos del software

Además del flujo de trabajo ingenieril que propone RUP para la obtención, el análisis, especificación y validación de los requisitos, las autoras han decidido aplicar el Modelo de Referencia para la Ingeniería de Requisitos en Proyectos de Bioinformática, con el objetivo de garantizar requisitos más precisos y de mayor calidad y que de esta forma los desarrolladores puedan tener un mejor entendimiento de los requisitos del sistema.

El modelo INeR propone cinco procesos como se muestra en la figura 2.11

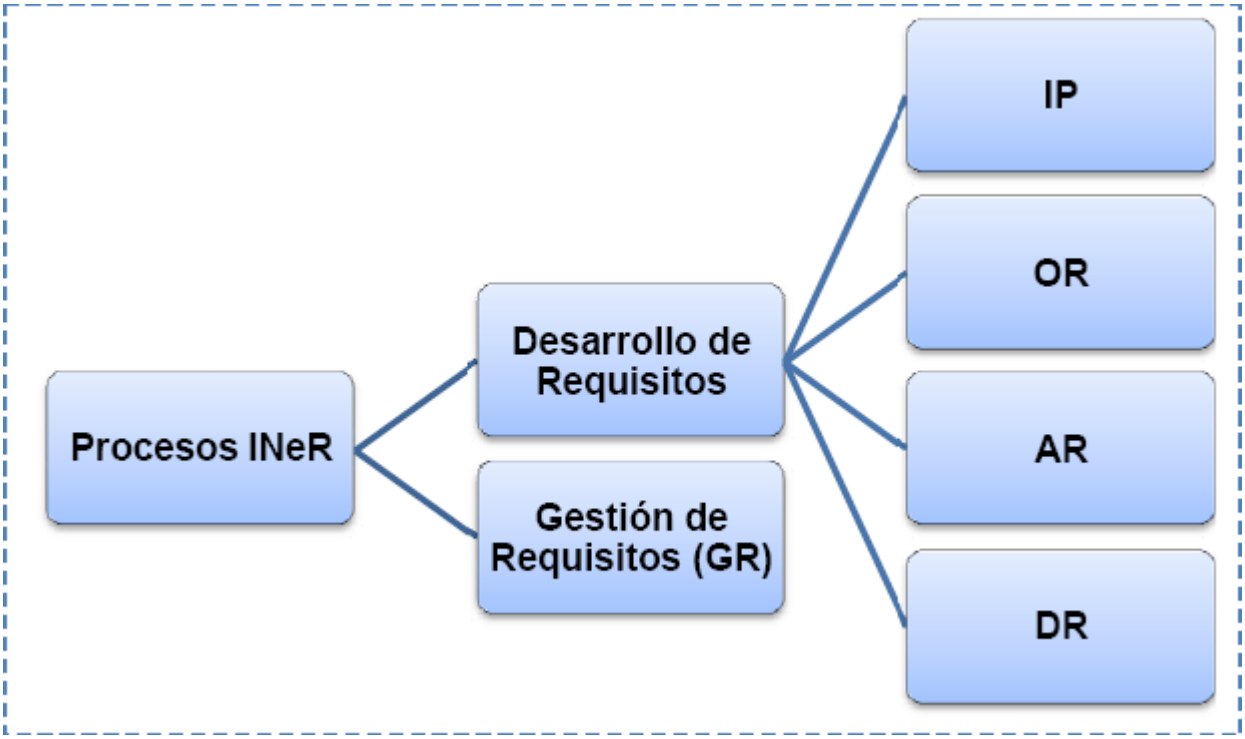


Figura 2.11: Procesos de INeR.

En el presente trabajo solamente se aplicarán tres procesos de los propuestos por el modelo: OR, AR y DR, que están agrupados en la categoría: Desarrollo de Requisitos. Se obvia el primer proceso que sería: IP, debido a que cuando surge la idea de aplicar este modelo al proyecto en cuestión ya este había pasado la fase de Inicio del Proyecto y fue imposible entonces comenzar desde el principio a aplicar INeR.

A continuación se describe el resultado de aplicar los procesos mencionados.

### 2.5.1 La obtención de los requisitos (OR)

Este proceso tiene como *objetivo*: obtener del cliente definición de sus necesidades.

Los *roles* que propone son:

- Líder de proyecto
- Ingeniero de requisitos
- Analistas

Las autoras fundieron el rol de ingeniero de requisitos con el de analistas, desempeñando ambas los dos roles al unísono.

Las *entradas* para la OR fueron: listado inicial de los involucrados en el proyecto, así como la primera versión del Glosario de Términos, el listado inicial de requisitos y prototipos básicos de una parte de lo que sería el futuro sistema.

Como parte de las actividades de la OR: se realizó un taller de requisitos con los involucrados en el proyecto, donde se efectuaron cuatro reuniones con el objetivo de ayudar a los clientes y usuarios a formular problemas y explorar posibles soluciones, involucrándolos y haciéndolos sentirse partícipes del desarrollo. Para la obtención de requisitos se aplicaron además entrevistas a los clientes y usuarios finales facilitando las condiciones para una buena comunicación y entendimiento de las partes involucradas en el proyecto.

Se aplicó también la técnica del aprendiz, técnica fundamentada en la idea del maestro y el aprendiz y basada en la observación del trabajo real. El aprendiz se sienta con el maestro a aprender por medio de la observación, haciendo preguntas y también realizando algún trabajo bajo la supervisión del maestro. (25) En este caso las autoras desempeñaron el papel de aprendiz y como maestro los empleados del Centro Coordinador Provincial de Emergencia Médica de Ciudad Habana.

Posterior a la obtención de requisitos se confeccionó un listado con los requisitos funcionales que fueron identificados hasta ese momento y se priorizaron, no ocurriendo lo mismo con los requisitos no

funcionales, los cuales serían obtenidos más adelante a partir de la decisión conjunta del equipo de desarrollo. Además se agregaron nuevos conceptos al Glosario de Términos y se refinó el Documento Visión.

Finalmente como *salidas* de este proceso quedaron formalmente definidos el Documento de Requisitos y el Glosario de Términos.

### 2.5.2 Análisis de los requisitos (AR)

Este proceso tiene como *objetivo*: convertir los requisitos obtenidos de los clientes/usuarios en requisitos reales.

Los *roles* que propone son:

- Ingeniero de requisitos
- Clientes
- Arquitecto

En este proceso solamente intervinieron dos roles, el Ingeniero de requisitos y los clientes.

Las *entradas* para el AR fueron: el Documento de Requisitos obtenido durante el proceso anterior y el listado de los involucrados en el proyecto.

Como *actividades* del proceso AR: se seleccionó un equipo para el análisis conjunto de los requisitos. El equipo estuvo integrado por los ingenieros de requisitos, en este caso este rol fue desempeñado por las autoras, los implementadores, el jefe de proyecto, así como profesores vinculados al proyecto. Se trabajó en conjunto además para garantizar que los requisitos representaran las necesidades reales del cliente y para verificar la correctitud de los mismos. Posterior a ello los ingenieros de requisitos documentaron cada uno de los requisitos con sus atributos y revisaron la prioridad de los mismos.

Como *salida* de este proceso se obtuvo el Documento de Requisitos actualizado a partir del análisis realizado.

### 2.5.3 Descripción de requisitos (DR)

Este proceso tiene como *objetivo*: obtener una descripción detallada de los requisitos que sirva como base para las actividades de ingeniería subsecuentes y validar los artefactos obtenidos técnicamente y desde el punto de vista del cliente.

Los *roles* que propone son:

- Analistas
- Especificadores
- Clientes

Las autoras fundieron el rol de especificadores con el de analistas, desempeñando ambas los dos roles al unísono, también intervinieron en este proceso los clientes.

Las *entradas* para la DR fueron: el Modelo de Negocio, desarrollado por las analistas, en este caso las autoras, el Documento de Requisitos actualizado, así como información relacionada al proyecto, ejemplo: el Manual del SIUM, facilitada por los clientes.

Como *actividades* del proceso DR: se analizaron los documentos elaborados referentes a los requisitos. Para modelarlos se utilizó la técnica de casos de uso, porque los requisitos representaban interacción con humanos y sistemas externos. No se usó la técnica del análisis estructurado porque los requisitos no representaban manejo masivo de datos, de igual forma tampoco se usó la técnica de pseudocódigo porque los requisitos no representaban operaciones de alta complejidad algorítmica. Se evaluó la calidad, integridad y consistencia de cada requisito y se validaron los modelos obtenidos con los involucrados, detectándose algunos errores que fueron corregidos por las analistas.

Finalmente como *salida* de este proceso se obtuvo el Documento de Especificación de Requisitos de Software.

## 2.6 Propuesta de sistema

A continuación se hará una descripción breve de las principales características del sistema que se va a desarrollar, lo cual podrá ser constatado a través del listado de los requisitos del software.

### 2.6.1 Especificación de los requisitos de software

Los requisitos de software son especificaciones de las características, cualidades, capacidades y comportamiento de un sistema, descritos de manera entendible para el equipo de desarrollo, clientes y usuarios y en función de la satisfacción de sus necesidades.

Constituyen la base para la medición de la calidad del producto final. (26) Los requisitos pueden ser clasificados en funcionales y no funcionales.

#### 2.6.1.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir.

En el Expediente de Proyecto se puede encontrar información detallada de cada requisito funcional, específicamente, en la plantilla: Especificación de Requisitos. A continuación se muestra la lista de los requisitos funcionales que fueron identificados:

**RF 1** Gestionar demanda

**RF 1.1** Insertar demanda

**RF 1.2** Listar demanda

**RF 1.3** Modificar demanda

**RF 2** Gestionar certificado

**RF 2.1** Insertar certificado

**RF 2.2** Buscar certificado

**RF 2.3** Modificar certificado

**RF 2.4** Visualizar certificado

**RF 3** Gestionar demanda NU

**RF 3.1** Insertar demanda NU

**RF 3.2** Buscar demanda NU

**RF 3.3** Modificar demanda NU

**RF 3.4** Visualizar demanda NU

**RF 4** Generar reporte Hoja de Cargo

**RF 5** Generar reporte Parte Diario

**RF 6** Generar reporte Enfermedades Trazadoras

**RF 7** Generar reporte Entrega de Guardia NU

**RF 8** Gestionar clave

**RF 8.1** Insertar clave

**RF 8.2** Modificar clave

**RF 8.3** Listar clave

**RF 8.4** Eliminar clave

**RF 9** Gestionar estado_conciencia

**RF 9.1** Insertar estado_conciencia

**RF 9.2** Modificar estado_conciencia

**RF 9.3** Listar estado_conciencia

**RF 9.4** Eliminar estado_conciencia

**RF 10** Gestionar clasificación_problema

**RF 10.1** Insertar clasificación _problema

**RF 10.2** Modificar clasificación_problema

**RF 10.3** Listar clasificación_problema

**RF 10.4** Eliminar clasificación_problema

**RF 11** Gestionar problema médico

**RF 11.1** Insertar problema médico

**RF 11.2** Modificar problema médico

**RF 11.3** Listar problema médico

**RF 11.4** Eliminar problema médico

**RF 12** Gestionar usuario

**RF 12.1** Autenticar usuario

**RF 12.2** Buscar usuario_ciudadano

**RF 12.3** Registrar usuario

**RF 12.4** Listar usuarios

**RF 12.5** Modificar usuario

**RF 12.6** Eliminar usuario

**RF 12.6** Listar controlador

**RF 12.7** Asignar municipio_controlador

**2.6.1.2 Requisitos no funcionales**

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. A continuación la lista de los requerimientos no funcionales que fueron identificados.

**Apariencia o Interfaz Externa:**

**RNF 1** La interfaz del sistema no contiene numerosas imágenes para evitar demoras en la respuesta de cualquier acción del usuario. La misma será sencilla, amigable e intuitiva, de fácil navegación por parte del usuario. Estará diseñada para una óptima visualización siendo adaptable a cualquier resolución.

**RNF 2** Los componentes del sistema serán desarrollados siguiendo el principio de Alta Cohesión y Bajo Acoplamiento, y los que sean reutilizables en los diferentes módulos del sistema serán desarrollados como Servicios Web XML que interactuarán con otros componentes a través de SOAP.

**Usabilidad:**

**RNF 3** La Aplicación Web facilita la interacción usuario–sistema con el objetivo de evitar rechazo en el uso de la misma. Mediante mensajes guíará al usuario en las diferentes acciones que realice. El usuario deberá poseer conocimientos básicos de computación y estar familiarizado con el negocio del Centro Coordinador Provincial de Emergencia Médica.

**Portabilidad:**

**RNF 4** Permite que el sistema se ejecute sobre el sistema operativo Windows XP y Linux.

**Soporte:**

**RNF 5** La Aplicación Web contará con una Ayuda donde el usuario podrá suplir las dudas que se le puedan presentar durante la utilización de la misma. El usuario del módulo deberá recibir un adiestramiento previo en la utilización del sistema, con el fin de que pueda explotar las prestaciones del sistema sin contratiempos ocasionados por la falta de preparación técnica.

**Seguridad:**

- **Confidencialidad:**

**RNF 6.1** La información que brinda el sistema estará protegida contra el acceso de usuarios no autorizados. Solamente los administradores del sistema podrán realizar cambios en la configuración y en la información.

- **Disponibilidad:**

**RNF 6.2** El sistema estará accesible cada vez que los usuarios del mismo lo requieran

**Software:**

- **Software en el Servidor**

**RNF 7.1** El servidor debe contar con sistema operativo Linux/Debian 4 Etch, con el servidor web Appserver.

- **Software en el Cliente**

**RNF 7.2** Para utilizar la Aplicación Web será necesaria una computadora con el Sistema Operativo Windows o Linux en cualquiera de sus versiones, recomendándose Windows XP o superior y Ubuntu 7.10 ó Superior. Además se podrá acceder desde cualquier Navegador Web recomendándose Internet Explorer 6 ó superior y Firefox 2.0 ó superior.

**Hardware:**

**RNF 8** Para garantizar el correcto funcionamiento de la aplicación se necesitan como requerimientos mínimos una computadora con un procesador Pentium II o superior, una memoria RAM de 512 MB o más, un Disco Duro de 10 GB o más y una tarjeta de red a 128 Mbps o más.

**Restricciones en el diseño y la implementación:**

**RNF 9.1** Se utilizarán los patrones de diseño establecidos.

**RNF 9.2** La lógica de presentación constituirá una capa independiente de la lógica de negocio, centrando su función en la interfaz de usuario y validaciones simples de la entrada de datos.

**RNF 9.3** Para el análisis y el diseño del sistema deberá ser utilizada la metodología RUP, usando el lenguaje de modelado UML y como herramienta para llevarlo a cabo el Visual Paradigm for UML v 3.0.

**RNF 9.4** Para la implementación se utilizará como lenguaje de programación PHP 5 y como herramienta de desarrollo el Zend Studio v 5.5.0.

### 2.6.2 Actores del Sistema

Cada trabajador del negocio (inclusive si fuera un sistema ya existente) que tiene actividades a automatizar es un candidato a actor del sistema. Si algún actor del negocio va a interactuar con el sistema, entonces también será un actor del sistema. Los actores del sistema: no son parte del mismo, pueden intercambiar información, pueden ser un recipiente pasivo de información, pueden representar el rol que juega una o varias personas, un equipo o un sistema automatizado. (27)

A continuación la figura 2.12 muestra la justificación de los actores del sistema.

| Actor | Descripción |
|---|---|
| Enfermera Coordinadora | Es la encargada de Insertar una demanda de urgencia o emergencia en el sistema. |
| Enfermera Coordinadora de NU | Es la encargada de Gestionar (Insertar, Buscar, Modificar y Visualizar) la información de los Certificados. Además Inserta en el sistema las demandas NU, es una generalización del rol Controlador_NU por lo tanto, también realiza sus funciones. |
| Controlador | Es el encargado de Gestionar (Listar y Modificar) las demandas de urgencia o emergencia en el sistema. |
| Controlador de NU | Es el encargado de Gestionar (Modificar y Visualizar), así como de Buscar, demandas NU en el sistema. |
| Jefe de Guardia | Es el encargado de Generar en el sistema los reportes estadísticos tales como: el Parte Diario, las Enfermedades Trazadoras, la Hoja de Cargo y la Entrega de Guardia de la no urgencia. |
| Administrador | Es el encargado de Gestionar (Insertar, Modificar, Listar y Eliminar) la información de los nomencladores: clave, estado_conciencia, problema |

| | |
|---|---|
| | médico y clasificación_problema. |
| Usuario | Es el encargado de Autenticar un usuario en el sistema. (Es la generalización de los actores: Enfermera_coordinadora, Controlador Enfermera_coordinadora_NU, Controlador_NU, Jefe de Guardia y Administrador) |
| Registro de Unidades de Salud (RUS) | Es el encargado de brindar un listado de los nombres de las Instituciones de Salud dado un municipio. |
| Registro de Personal de la Salud (RPS) | Es el encargado de brindar información acerca del Registro Profesional del personal de la salud (Verificar veracidad del RP dado). |
| Registro de Ubicación (RU) | Es el encargado de brindar un listado de todas las provincias del país para escoger una y de ahí mostrar sus correspondientes municipios. |
| Servicio de Autenticación, Autorización y Auditoria (SAAA) | Es el encargado de garantizar y gestionar la autenticación de los usuarios en el sistema. |
| Registro de Ciudadano (RC) | Es el encargado de brindar un conjunto de datos de los usuarios para que estos puedan ser registrados o adicionados al grupo de usuarios que tienen acceso y permisos en el sistema. |
| Operaciones | Es el encargado de brindar la disponibilidad de ambulancias con la que cuenta una Base de Ambulancias en particular. |

Figura 2.12: Actores del Sistema

### 2.6.3 Estructuración del sistema en paquetes

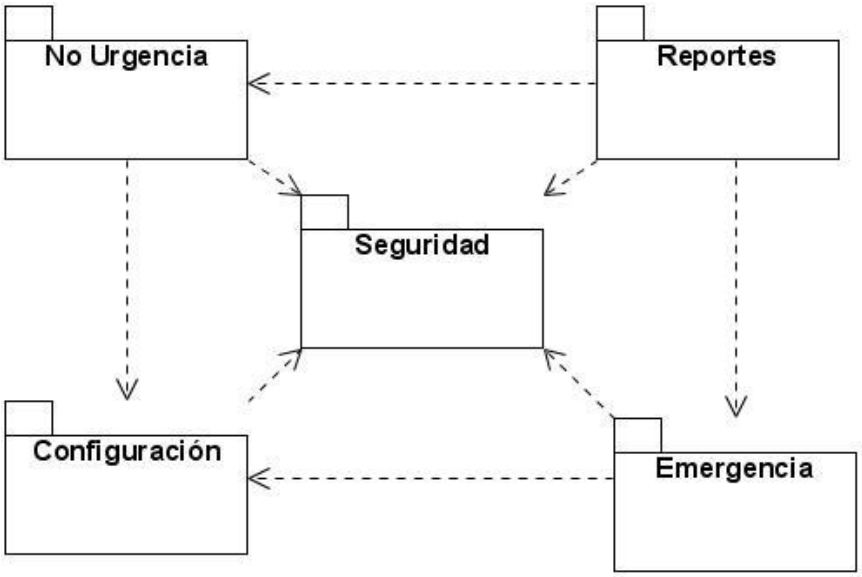


Figura 2.13: Estructuración del sistema en paquetes.

### 2.6.4 Diagramas de CU del sistema

A continuación se presentan los diagramas de casos de uso del sistema organizados por paquete.

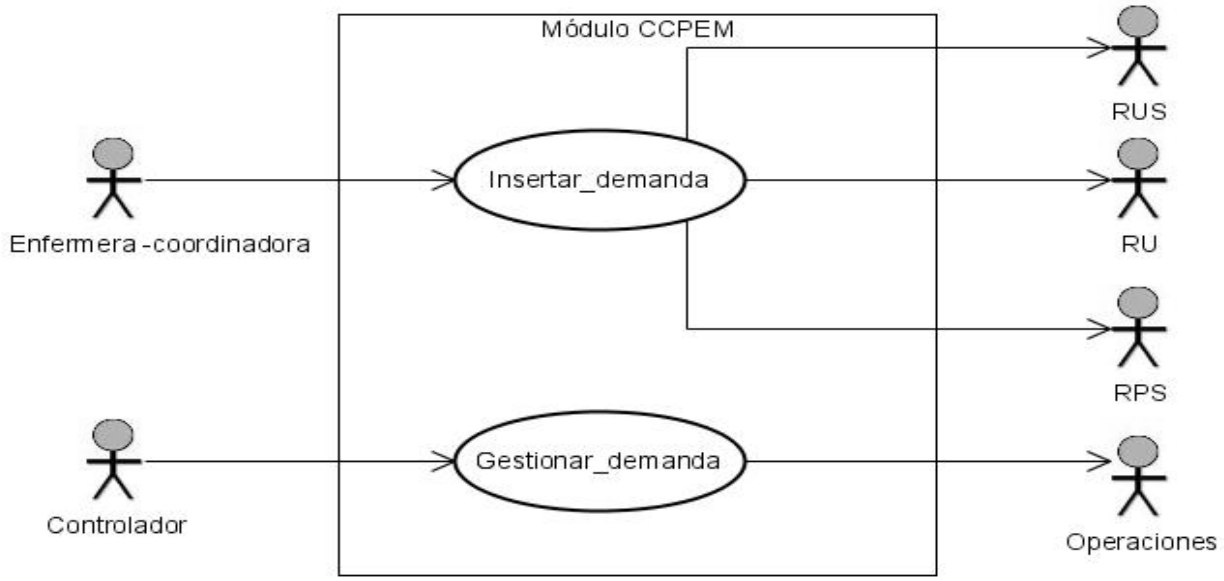


Figura 2.14: DCUS_Paquete_Emergencia

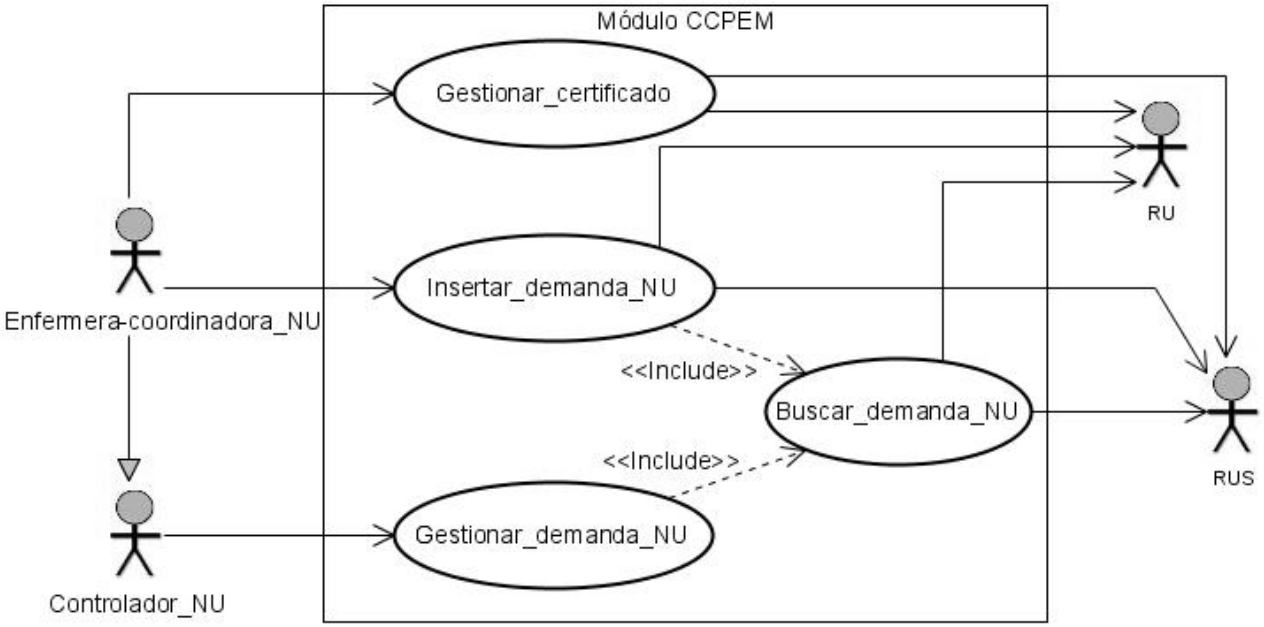


Figura 2.15: DCUS_Paquete_No_Urgencia

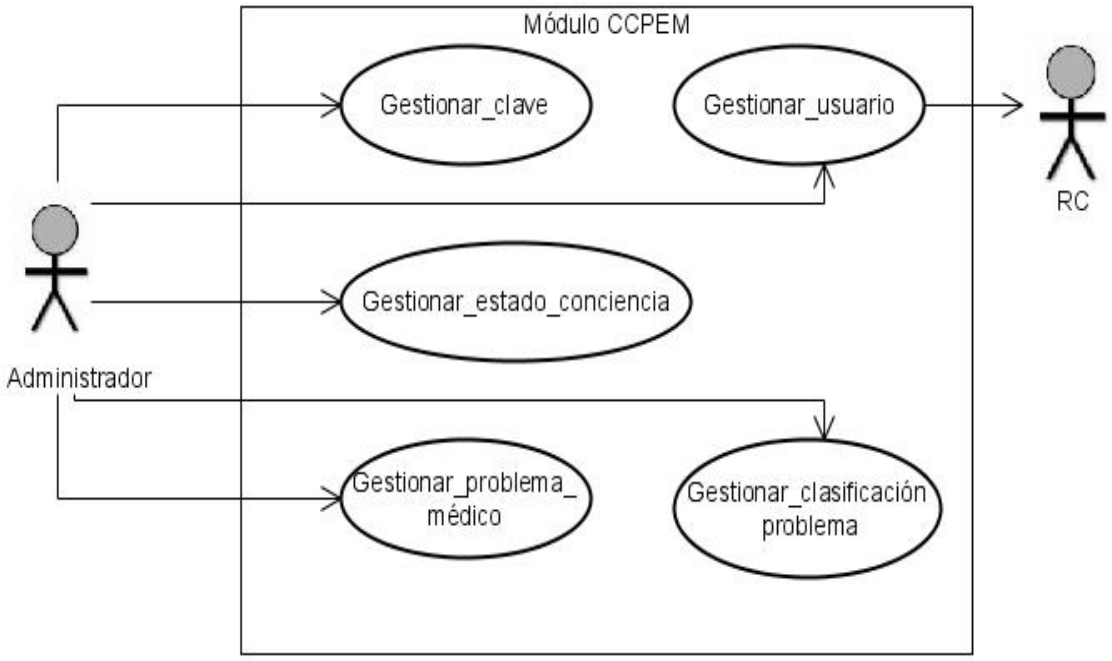


Figura 2.16: DCUS_Paquete_Configuración

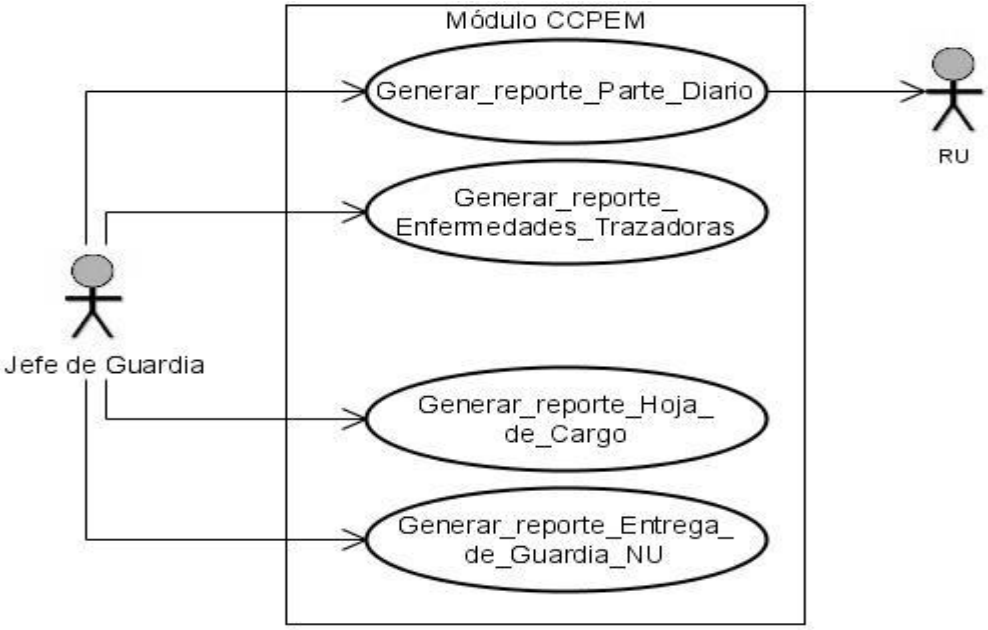


Figura 2.17: DCUS_Paquete_Reportes

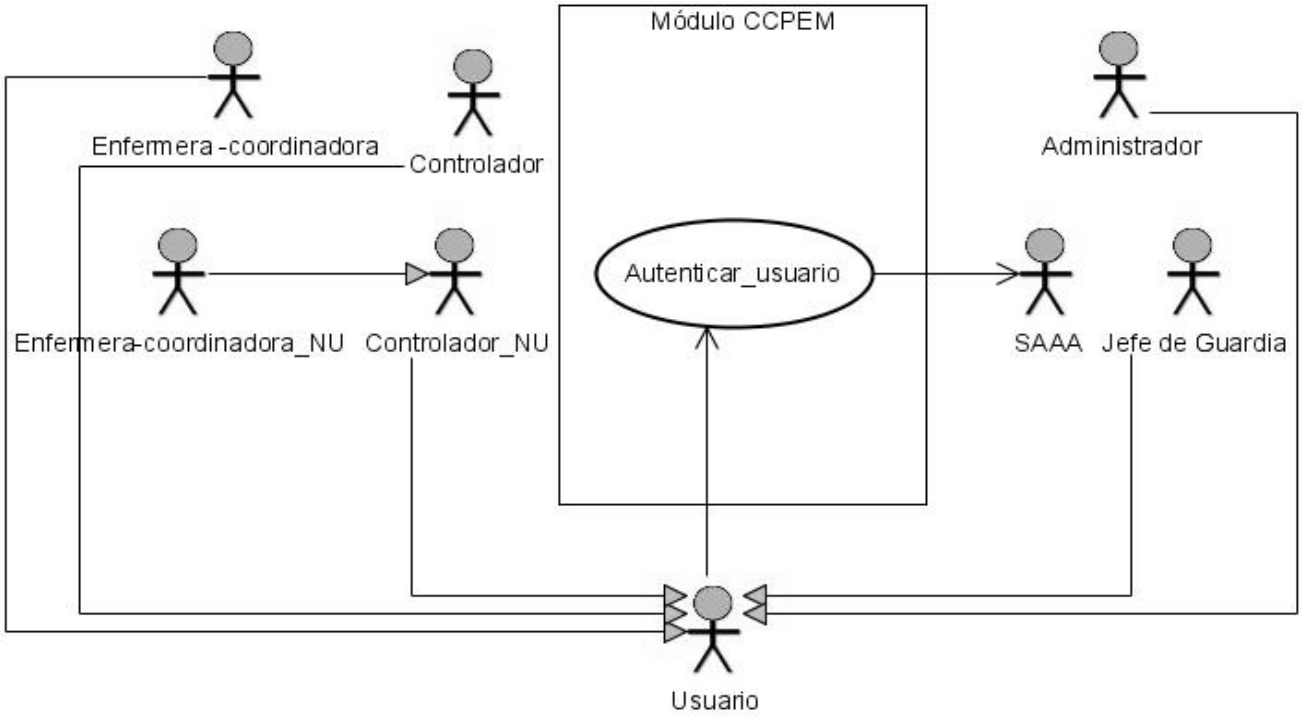


Figura 2.18: DCUS_Paquete_Seguridad

## 2.7 Métricas aplicadas

La medición (necesidad de obtener datos objetivos que ayuden a mejorar la calidad) es muy común en el mundo de la ingeniería. Las métricas nos ayudan a entender tanto el proceso técnico que se utiliza para desarrollar un producto, como el propio producto, así como el proceso para intentar mejorarlo, el producto se mide para intentar aumentar su calidad.

Se define calidad como la concordancia con los requisitos funcionales y de rendimiento, explícitamente establecidos, los estándares de desarrollo explícitamente documentados y las características implícitas que se esperan de todo software desarrollado profesionalmente. (28)

• Los requisitos del software son la base de las medidas de la calidad. La falta de concordancia con los requisitos es una falta de calidad.

• Unos estándares específicos definen un conjunto de criterios de desarrollo que guían la manera en que se hace la ingeniería del software. Si no se siguen los criterios, habrá seguramente poca calidad.

• Existe un conjunto de requisitos implícitos que a menudo no se nombran (por ejemplo la facilidad de mantenimiento); si el software cumple con sus requisitos explícitos pero falla en los implícitos, la calidad del software no será fiable.

### 2.7.1 Métricas aplicadas en el Modelo de Casos de Uso del Sistema

Con el fin de tener cifras certeras que midan el grado de correctitud, completitud, consistencia y complejidad se aplica un modelo de métricas al Modelo de Casos de Uso del Sistema. Para que así quede un Modelo de Casos de Uso de Sistema que cumpla con esta serie de requisitos importantes para los usuarios y desarrolladores. Los atributos antes mencionados contienen una serie de factores los cuales tienen una métrica asociada que da un valor cuantitativo del umbral de calidad que tiene dicho modelo, dichos atributos son:

Completitud: Grado en que se ha logrado detallar todos los casos de uso relevantes.

Consistencia: Grado en que los casos de uso del sistema describen las interacciones adecuadas entre el usuario y el sistema.

Correctitud: Grado en que las interacciones actor/sistema soportan adecuadamente el proceso del negocio.

Complejidad: Grado de calidad en la presentación de los elementos que describen el contexto y la claridad del sistema.

En la Figura.2.19 se pueden observar las preguntas que se hacen para medir cada uno de estos atributos.

| | |
|---|---|
| **Completitud** | 1. ¿Han sido involucradas todas las áreas funcionales a las cuales apoyará el sistema?<br>2. ¿Han sido definidos todos los roles relevantes de usuario encargados de generar/ modificar o consultar información?<br>3. ¿Están definidos todos los requisitos que justifican la funcionalidad de los casos de uso?<br>4. ¿Se presenta una descripción detallada (descripción extendida esencial) de todos los casos de uso del sistema?<br>5. ¿Están todas las acciones del flujo de eventos redactadas en función del responsable? |
| **Consistencia** | 6. ¿El nombre dado a los casos de uso es una expresión verbal que describe alguna funcionalidad relevante en el contexto del usuario?<br>7. ¿Representan los casos de uso una interacción observable por un actor?<br>8. ¿Está adecuadamente redactado (en el lenguaje del usuario) el flujo de eventos?<br>9. ¿La descripción del flujo de eventos se inicia con la descripción de una acción externa originada por un actor o por una condición interna del sistema claramente identificable?<br>10. ¿Existe una adecuada separación entre el flujo básico de eventos y los flujos alternos? |
| **Correctitud** | 11. ¿Existe para cada caso de uso por lo menos un usuario responsable?<br>12. ¿Representa el caso de uso requisitos comprensibles por el usuario?<br>13. ¿Las interacciones definidas introducen mejoras al proceso actual? |

| | |
|---|---|
| Complejidad | 14. ¿Los elementos dentro del diagrama de casos de uso están adecuadamente ubicados de manera que facilitan su interpretación? |

Figura 2.19: Métrica para evaluar el Modelo de Casos de Uso del Sistema.

**2.7.1.1 Resultados de la métrica aplicada al DCUS por el Laboratorio de Calidad de la Facultad 7**

En la revisión efectuada por el laboratorio de Calidad de la Facultad 7 los resultados arrojados fueron los siguientes:

El Modelo de CU del Sistema fue evaluado por los atributos de completitud, consistencia, complejidad y correctitud devengando los siguientes resultados: el atributo de completitud se cumple en un 66%, de nueve CU revisados tres estaban incompletos. La consistencia se cumple en un 80% debido a dificultades presentadas en dos descripciones de CU en las que faltaba introducir la indicación de cuando inicia el CU.

La correctitud se cumple en un 100% ya que existe, para cada caso de uso, por lo menos un usuario responsable. Representan los casos de uso requisitos comprensibles por el usuario y además las interacciones definidas introducen mejoras al proceso actual. La complejidad se evaluó con 100% debido a que se cumple la métrica como está establecida. El Laboratorio de Calidad emite estos resultados con las facultades que se le confieren.

**2.7.2 Métricas para el levantamiento de requisitos**

• Cubrimiento de los requisitos por casos de uso

$$\#\,\boldsymbol{Requisitos} \geq \mathbf{2} * \#\,\boldsymbol{CUN}$$

Esta métrica permite validar que cada uno de los requisitos se encuentre enmarcado en alguno de los casos de uso del negocio, debido a que los mismos son funcionalidades del sistema y a la vez describen por sí solo algún proceso del negocio. Lo que quiere decir que esta métrica tiene como objetivo garantizar que los requisitos del cliente no se vayan del margen del negocio planteado. No puede haber requisitos que no se encuentren dentro de los CU del negocio.

#### 2.7.2.1 Resultados de la métrica aplicada al levantamiento de requisitos por el Laboratorio de Calidad de la Facultad 7

$$\#\ Requisitos \geq 2\ * \#\ CUN$$

**39** > **12**

Al aplicar la métrica para evaluar el levantamiento de requisitos, se comprobó que cada uno de los requisitos se encontraba dentro de la funcionalidad de un Caso de Uso al menos, obteniéndose un resultado de un 100% de cumplimiento de la métrica.

## 2.8 Análisis de los resultados arrojados por las métricas aplicadas

Una vez aplicadas las métricas abordadas en los epígrafes anteriores, para evaluar la calidad con la que se llevo a cabo el Flujo de Trabajo: Requerimientos, las autoras percibieron que aún existían algunos errores en las descripciones textuales de los casos de uso del sistema. Posterior a dicha revisión fueron revisadas y refinadas, teniendo en cuenta las sugerencia realizadas por el Laboratorio de Calidad de la Facultad 7.

## Conclusiones

Con el desarrollo de este capítulo se logró identificar los principales procesos que rigen el negocio en un Centro Coordinador Provincial de Emergencia Médica. Lo cual facilitó el entendimiento, a las analistas y en este caso, a las autoras, del entorno en el cual se desarrolla la problemática a resolver. A su vez fueron identificadas posibles mejoras al proceso de gestión de información de las demandas en un CCPEM.

Con la aplicación del modelo INeR para la ingeniería de requisitos del proyecto se obtuvieron resultados satisfactorios, pues permitió desarrollar el flujo de trabajo de Requirimientos con más profundidad y calidad, generando un Documento de Especificación de Requisitos del Software consistente y con requerimientos bien definidos y fundamentados. Todo lo antes expuesto tributa a un mejor trabajo en los siguientes flujos de trabajo ingenieriles y además posibilita que los desarrolladores tengan una mejor visión de los requerimientos con los que debe cumplir el sistema software.

## Capítulo 3. Diseño del sistema

### Introducción

Para desarrollar una aplicación, es necesario contar con descripciones detalladas, con un alto nivel de la solución lógica y saber como satisfacer los requisitos y las restricciones. El diseño pone de relieve una solución lógica: como el sistema cumple con los requisitos. Los diseños se implementan en software y en hardware.

A partir del análisis de los requisitos identificados en el capítulo anterior, en el siguiente capítulo se muestra la propuesta de diseño para la implementación de un software que facilite la gestión de información de las demandas, en un Centro Coordinador Provincial de Emergencia Médica. Dicha propuesta está respaldada por los diferentes artefactos que propone RUP para el diseño de un sistema software, que es en este caso, una aplicación web.

### 3.1 Modelo de diseño

El modelo de diseño es un modelo de objetos que describe la realización de los CU, y sirve como una abstracción del modelo de implementación y el código fuente. Es usado como una entrada inicial en las actividades de implementación y prueba. Es abarcador y está compuesto por artefactos que engloban todas las clases del diseño, subsistemas, paquetes, colaboraciones, y las relaciones entre ellos.

### 3.2 Patrones de Diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

#### 3.2.1 Patrón Modelo-Vista-Controlador (MVC)

Para la realización de este epígrafe se ha utilizado como referencia bibliográfica: (29)

El patrón MVC es un patrón de diseño de software en el cual todo el proceso está dividido en tres capas, típicamente estas capas son: el Modelo, la Vista y el Controlador. El Modelo incorpora la capa

de dominio y persistencia, es la encargada de guardar los datos en un medio persistente (ya sea una base de datos, un archivo de texto, XML, registro, etc.). En el modelo es donde se hace el levantamiento de todos los objetos que el sistema debe utilizar, es el proveedor de los recursos.

La *Vista* se encarga de presentar la interfaz al usuario, en sistemas web, es típicamente la página HTML y el código que provee de datos dinámicos a la página, aunque pueden existir otro tipo de vistas. En la vista solo se deben de hacer operaciones simples, como ifs, ciclos, formateo, etc.

El *Controlador* es el que escucha los cambios en la vista y se los envía al modelo, el cual le regresa los datos a la vista, es un ciclo donde cada acción del usuario causa que se inicie un nuevo ciclo.

El patrón MVC se ve frecuentemente en aplicaciones web. Aunque se pueden encontrar diferentes implementaciones de MVC, en la variante seleccionada no existe ninguna comunicación entre el Modelo y la Vista y esta última recibe los datos a mostrar a través del Controlador, como se muestra en la figura 3.1.

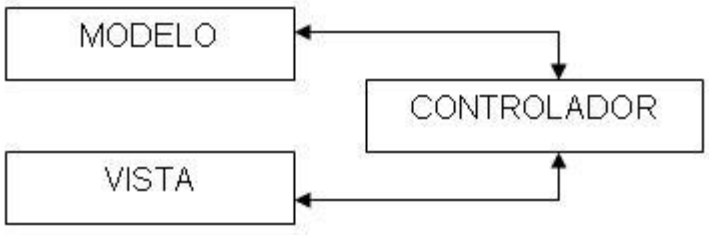


Figura 3.1: Variante del patrón MVC

Al aplicar dicho patrón, las clases que fueron creadas para diseñar el software se incluyen dentro de la tres categorías propuestas por el mismo, tales como: clase Vista, clase Modelo y clase Control. De esta forma se garantiza que la clase Modelo no tenga acoplamiento, ni visibilidad directa respecto a la clase Vista y los datos de la aplicación y de la funcionalidad se conservan en la clase Modelo, no en la Vista. La clase Controlador procesa los eventos (peticiones) al sistema y redirecciona a la clase Modelo y Vista, tanto el procesamiento como la visualización de resultados, respectivamente.

Algunos de los principales beneficios que brinda aplicar el patrón MVC son:

- Menor acoplamiento.

Desacopla las vistas de los modelos.

- Desacopla los modelos de la forma en que se muestran e ingresan los datos.

- Mayor cohesión.
    - Cada elemento del patrón esta altamente especializado en su tarea (la vista en mostrar datos al usuario, el controlador en las entradas y el modelo en su objetivo de negocio).
- Las vistas proveen mayor flexibilidad y agilidad.
    - Se pueden crear múltiples vistas de un modelo.
    - Las vistas pueden anidarse.
    - Se puede cambiar el modo en que una vista responde al usuario sin cambiar su representación visual.
    - Se pueden sincronizar las vistas.
    - Las vistas pueden concentrarse en diferentes aspectos del modelo.
- Más claridad de diseño.
- Facilita el mantenimiento.
- Mayor escalabilidad.

### 3.2.2 Patrones Grasp

En diseño orientado a objetos, GRASP son patrones generales de software para asignación de responsabilidades, es el acrónimo de "General Responsibility Assignment Software Patterns" . Aunque se considera que más que patrones propiamente dichos, son una serie de "principios y estilos" de aplicación recomendable en la creación de software. (30)

A continuación se enuncian cinco de los principales patrones Grasp: (31)

**Experto**

¿Quién asumirá la responsabilidad en el caso general?

Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

**Creador**

¿Quién crea?

Asignar a la clase B la responsabilidad de crear una instancia de clase A, si se cumple una de las siguientes condiciones:

1. B contiene A

2. B agrega A

3. B tiene los datos de inicialización de A

4. B registra A

5. B utiliza A muy de cerca.

**Controlador**

¿Quién administra un evento del sistema?

Asignar la responsabilidad de administrar un mensaje de eventos del sistema a una clase que represente una de las siguientes opciones:

1. El negocio o la organización global (un controlador de fachada).

2. El "sistema" global (un controlador de fachada).

3. Un ser animado del dominio que realice el trabajo (un controlador de papeles).

4. Una clase artificial (Fabricación Pura) que represente el caso de uso (un controlador de casos de uso).

**Bajo acoplamiento**

¿Cómo dar soporte a poca dependencia y a una mayor reutilización?

Asignar las responsabilidades de modo que se mantenga bajo acoplamiento.

**Alta cohesión**

¿Cómo mantener controlable la complejidad?

Asignar las responsabilidades de modo que se mantenga una alta cohesión.

Los patrones antes mencionados también fueron aplicados durante el diseño del software. A cada clase le fueron asignadas las tareas que podían realizar según la información que poseían, poniéndose de manifiesto el patrón *Experto*. Lo cual trae como beneficio que se conserve el encapsulamiento, ya

que los objetos se valen de su propia información para hacer lo que se les pide. Esto soporta un *bajo acoplamiento*, lo que favorece el hecho de tener sistemas más robustos y de fácil mantenimiento. El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clase "sencillas" y más cohesivas que son más fáciles de comprender y de mantener. Así se brinda soporte a una *alta cohesión*.

Además se crearon las instancias de otras clases en correspondencia con la responsabilidad dada; poniéndose de manifiesto el patrón *Creador*.

El patrón *Controlador* también estuvo presente al definirse clases controladoras, que sirven como intermediarias entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado.

El diseño obtenido cumple con los patrones de *Bajo acoplamiento* y *Alta cohesión* permitiendo la colaboración entre los elementos del diseño (clases), sin verse afectados la reutilización de los mismos y el entendimiento de estos cuando se encuentran aislados.

### 3.3 Diagramas de clases del diseño

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces, en una aplicación. Normalmente contiene la siguiente información:

- clases, asociaciones y atributos
- interfaces, con sus operaciones y constantes
- métodos
- información sobre los tipos de los atributos
- navegabilidad
- dependencias

Un diagrama de este tipo contiene las definiciones de las entidades del software en vez de conceptos del mundo real. (32)

A continuación, figura 3.2, 3.3 y 3.4, una muestra de diagramas de clase del diseño del software.

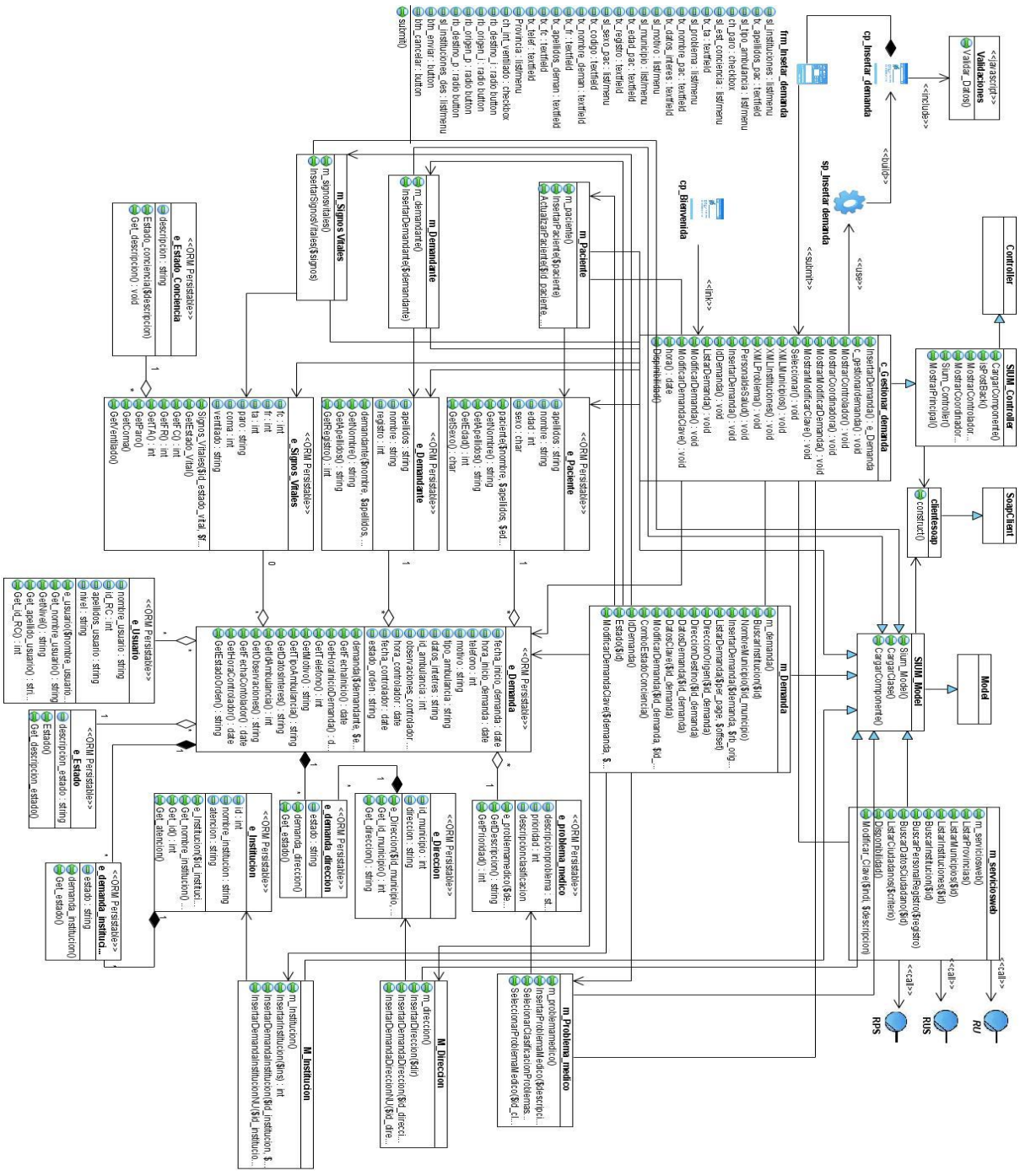

Figura 3.2: DCD_Insertar_demanda

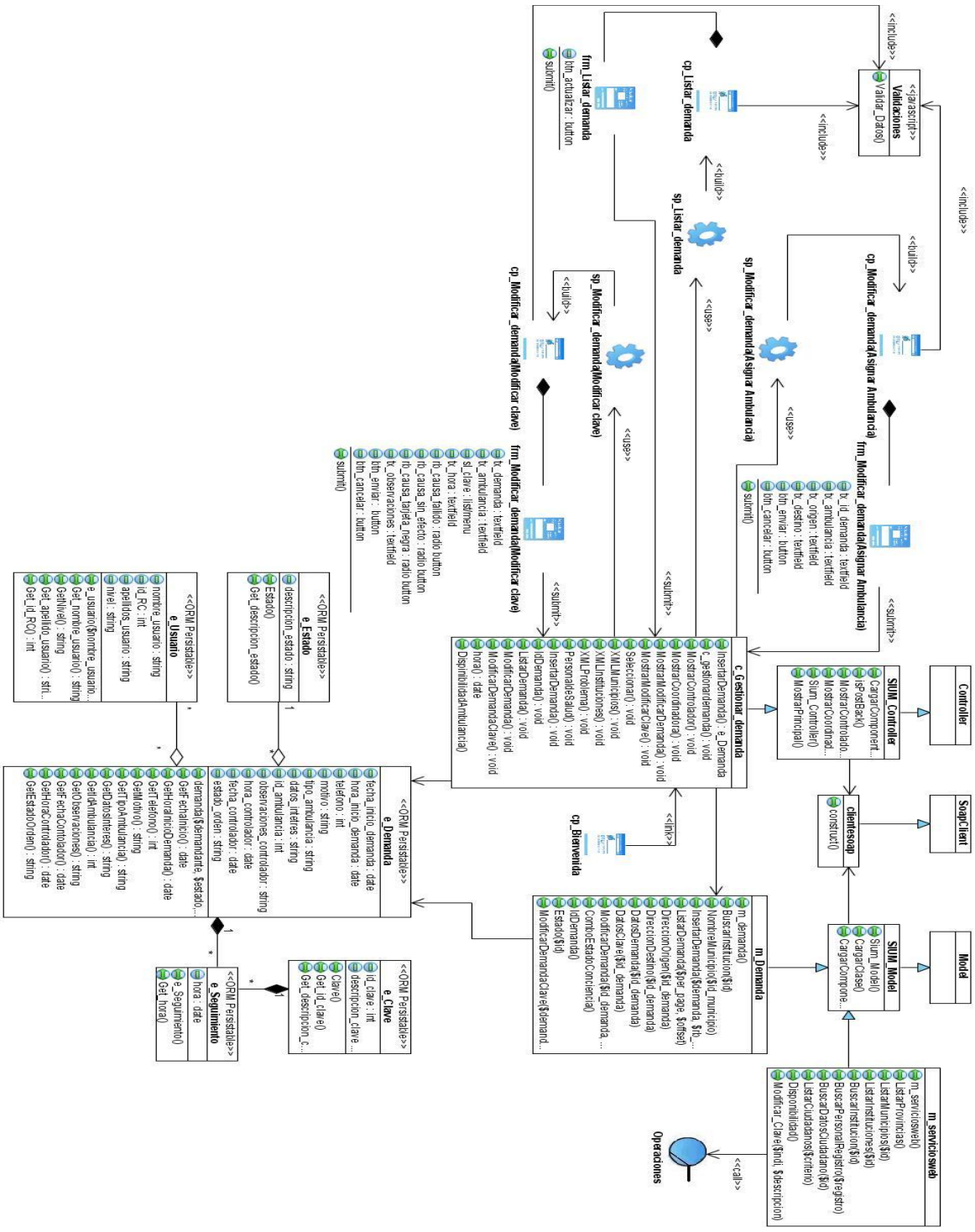


Figura 3.3: DCD_Gestionar_demanda

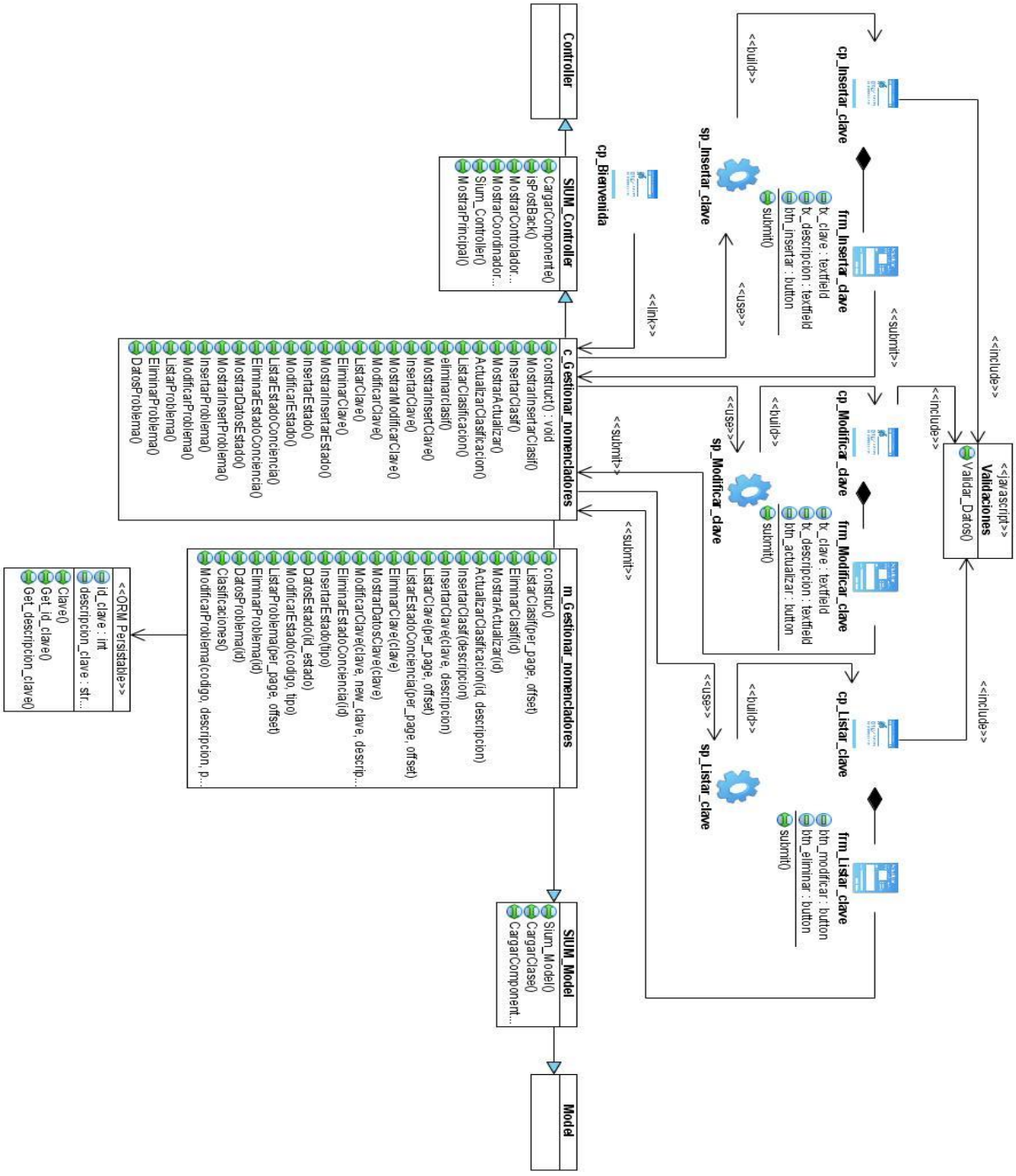


Figura 3.4: DCD_Gestionar_clave

### 3.4 Diagramas de interacción

Un diagrama de interacción explica gráficamente las interacciones existentes entre las instancias (y las clases) del modelo de estas. El punto de partida de las interacciones es el cumplimiento de las poscondiciones de los contratos de operación.

El UML define dos tipos de estos diagramas; ambos sirven para expresar interacciones semejantes o idénticas de mensaje:

1. Diagramas de colaboración.
2. Diagramas de secuencia. (33)

Los diagramas de colaboración describen las interacciones entre los objetos en un formato de grafo o red, como se aprecia en la figura 3.5

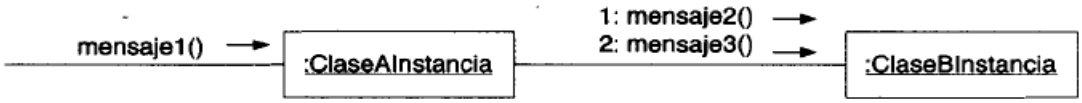


Figura 3.5 Diagrama de colaboración.

Los diagramas de secuencia describen las interacciones en una especie de formato de cerca o muro. (31) Ver la figura 3.6

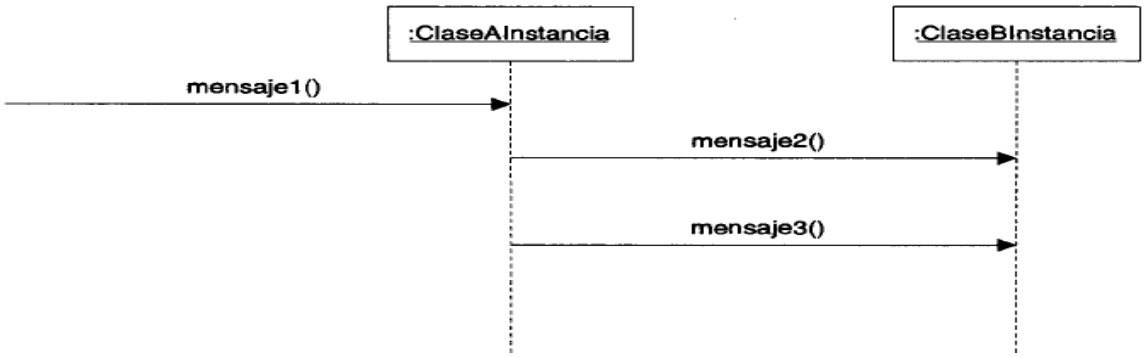


Figura 3.6 Diagrama de secuencia.

Las interacciones entre los objetos y las clases, de la solución propuesta como resultado del diseño del software, fueron modeladas a través de diagramas de secuencia. A continuación se muestran algunos ejemplos, desde la figura 3.7 hasta la figura 3.14.

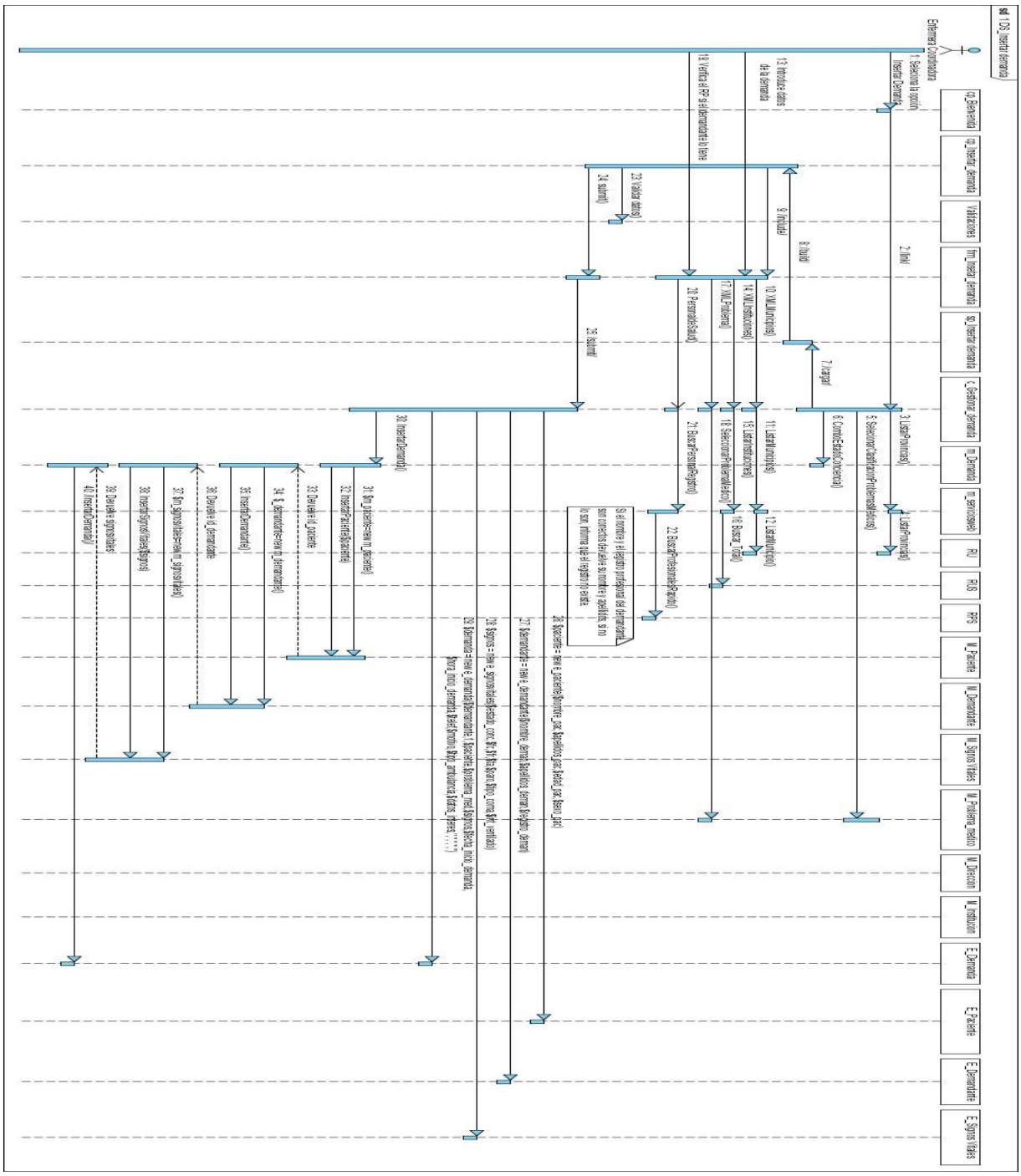

Figura 3.7 DS_Insertar_demanda

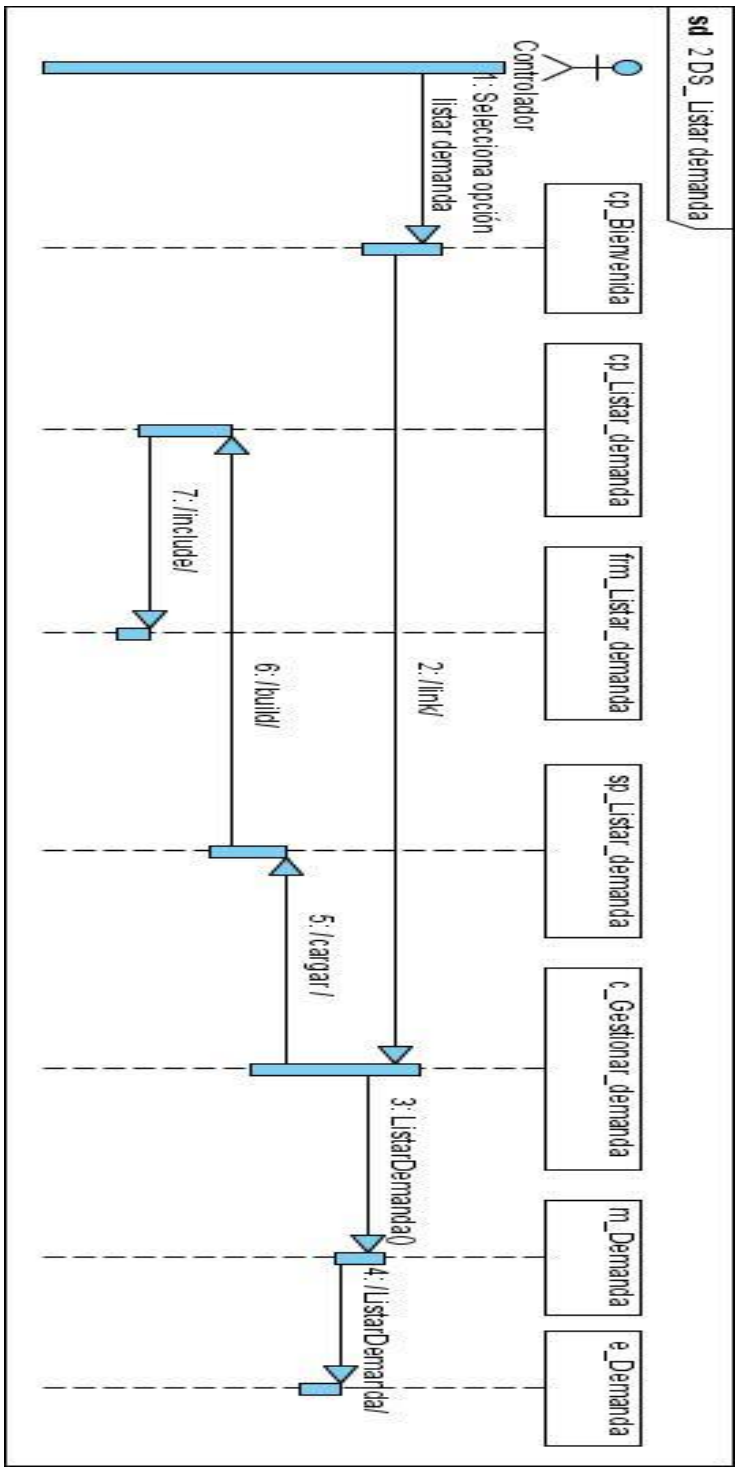


Figura 3.8 DS_Listar_demanda

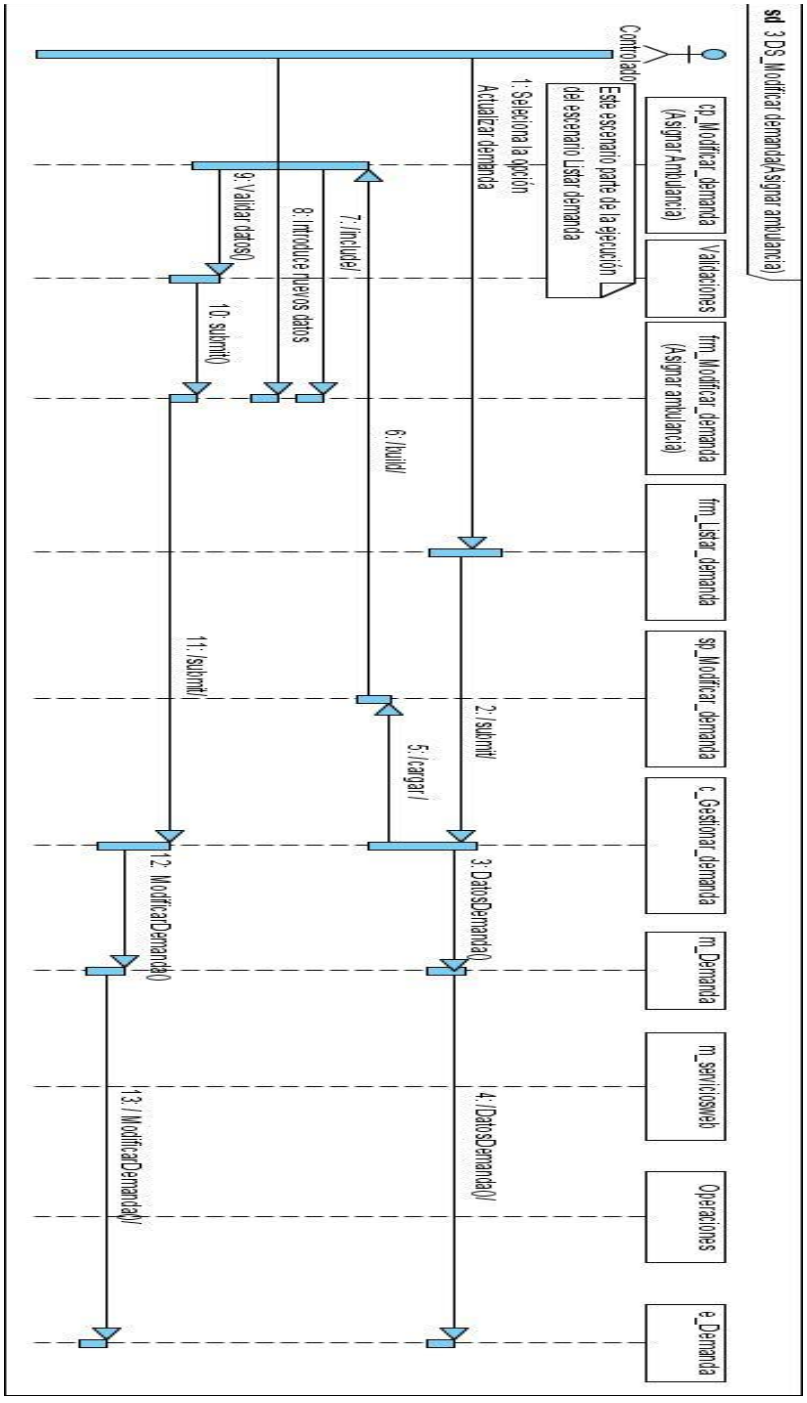


Figura 3.9 DS_Modificar_demanda(Asignar_ambulancia)

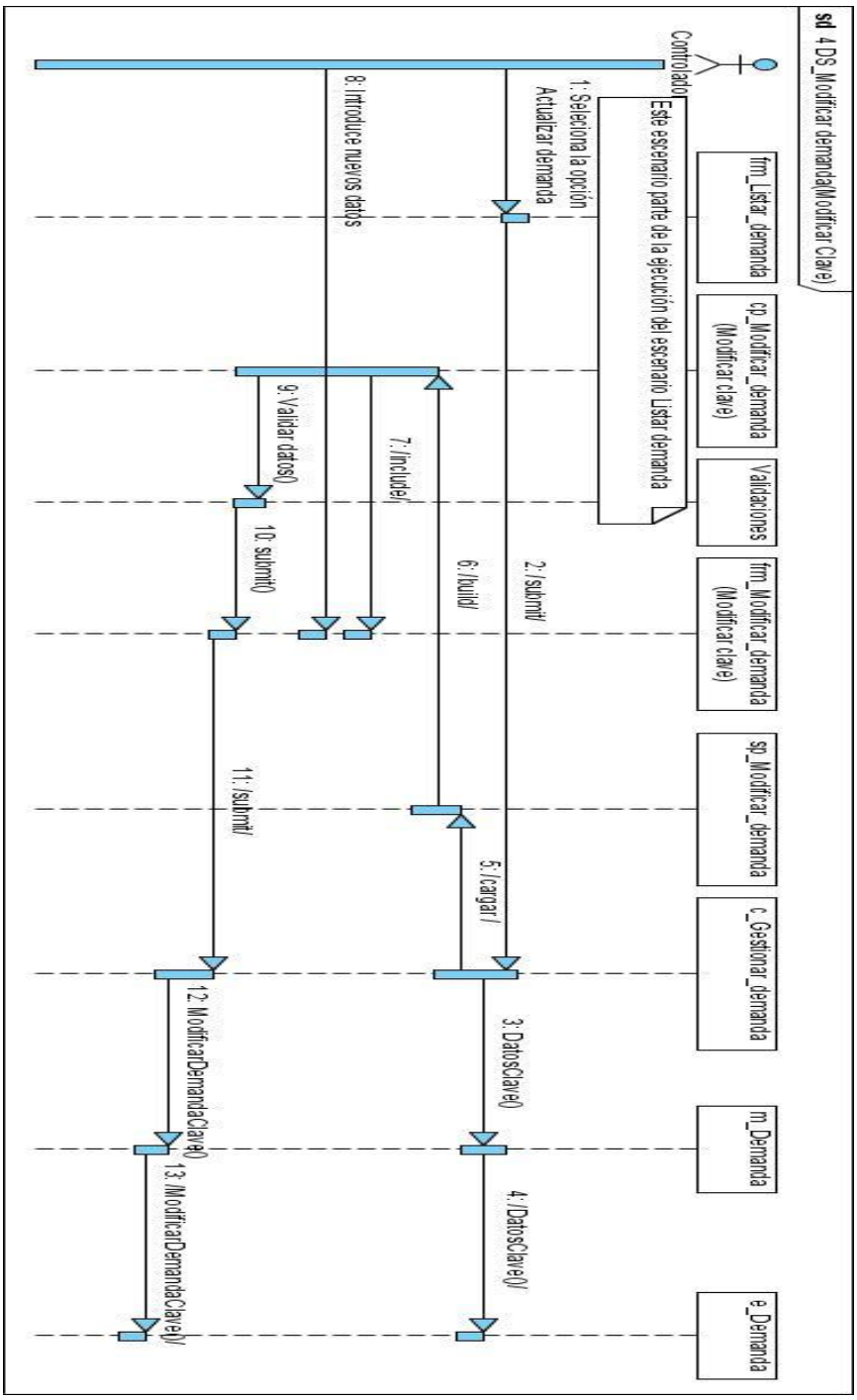


Figura 3.10 DS_Modificar_demanda(Modificar_clave)

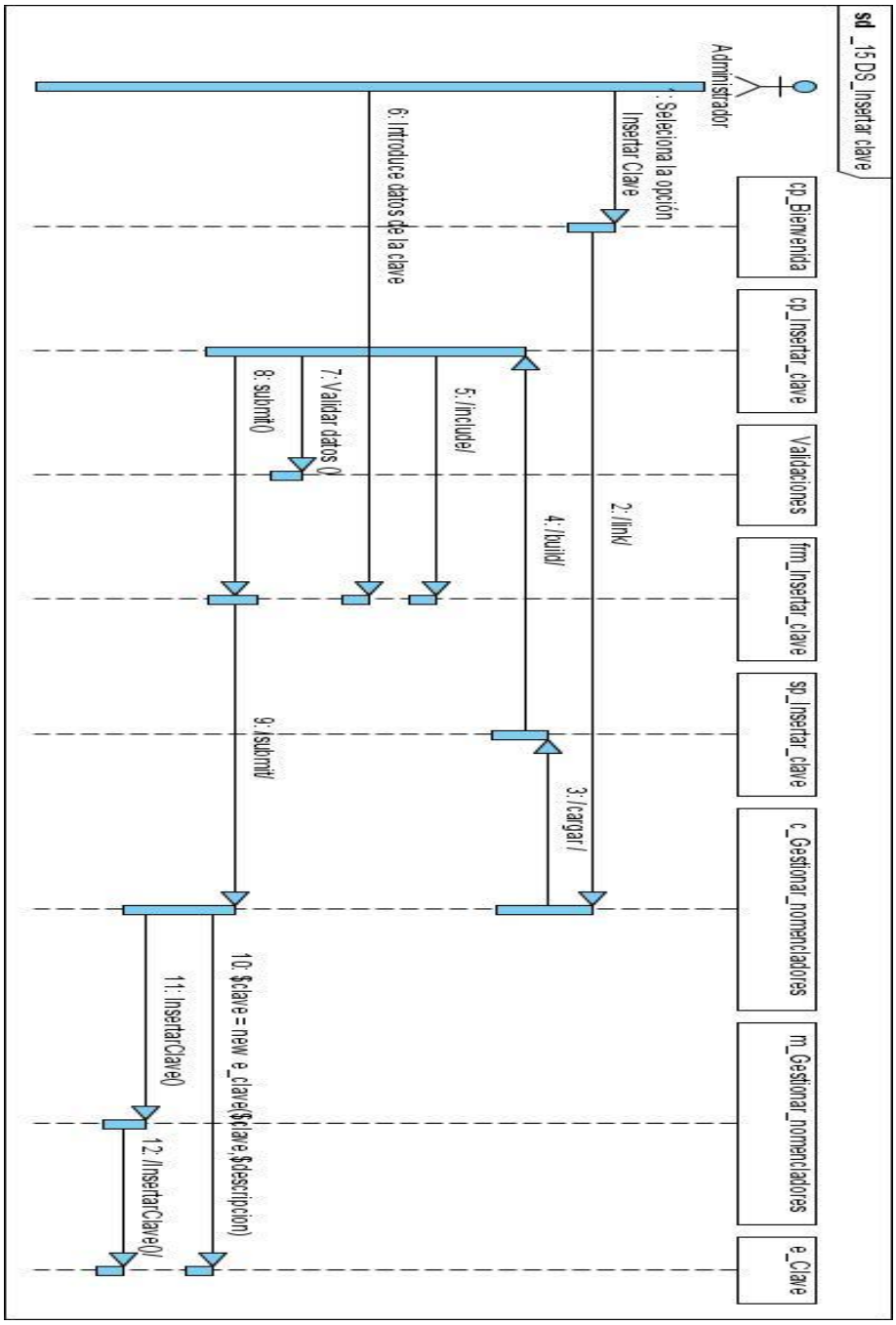


Figura 3.11 DS_Insertar_clave

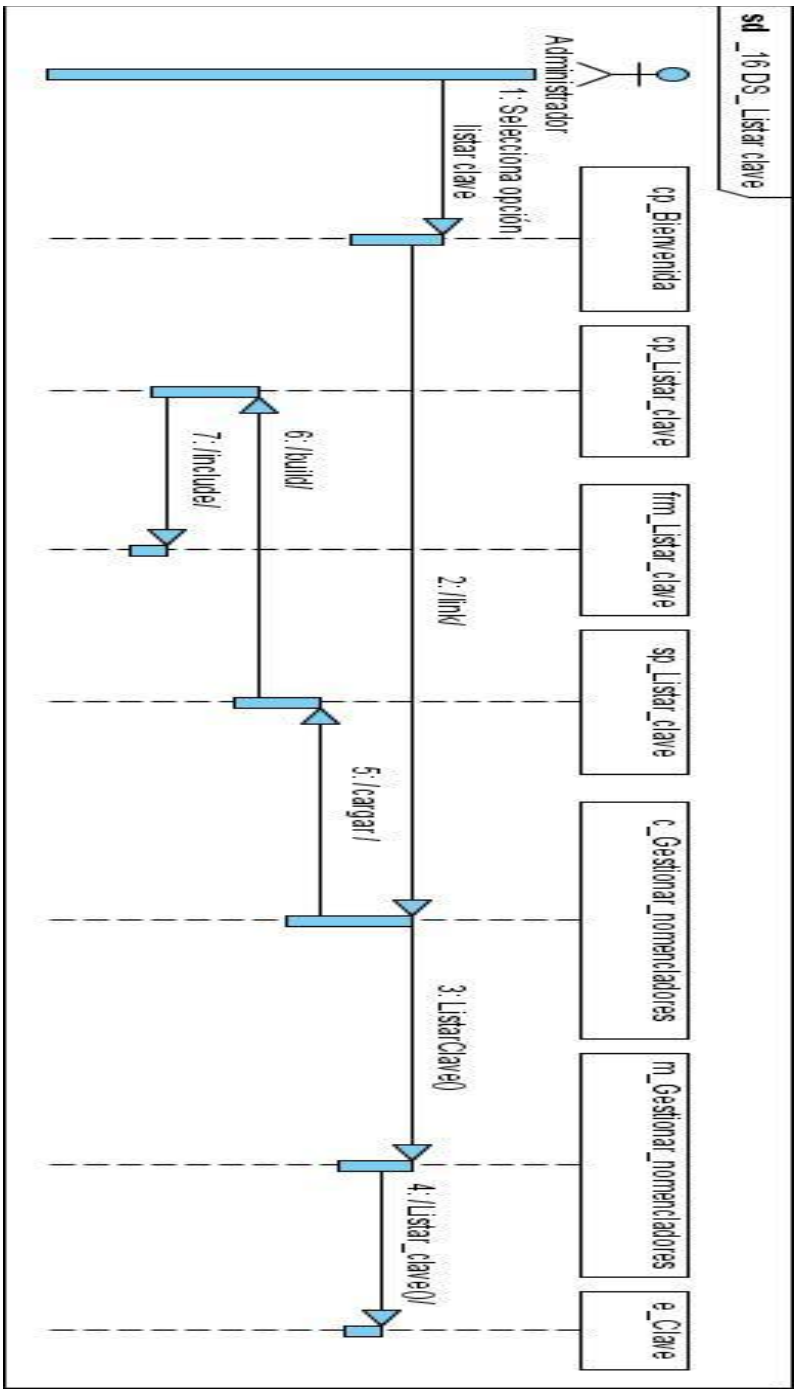


Figura 3.12 DS_Listar_clave

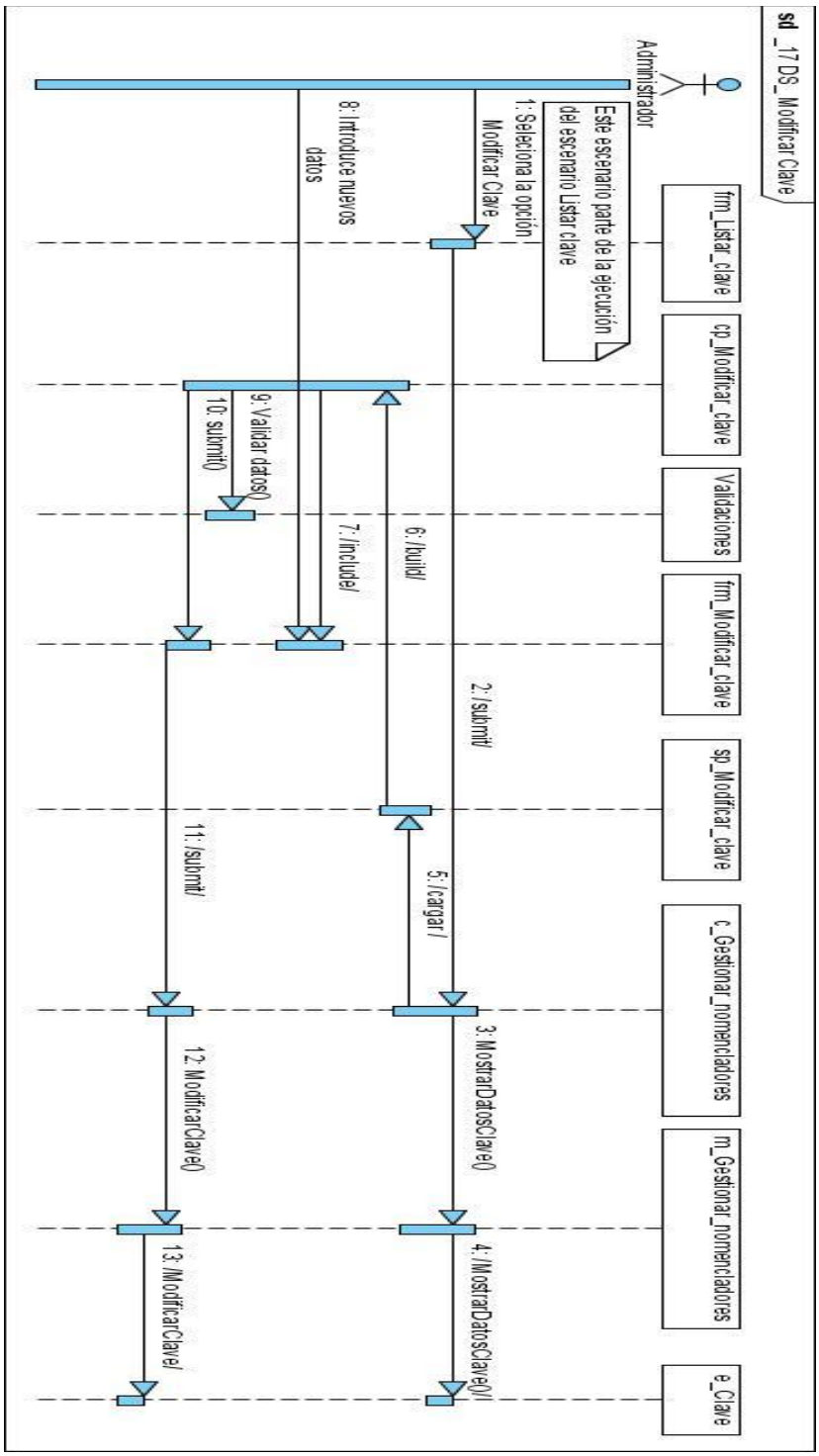


Figura 3.13 DS_Modificar_clave

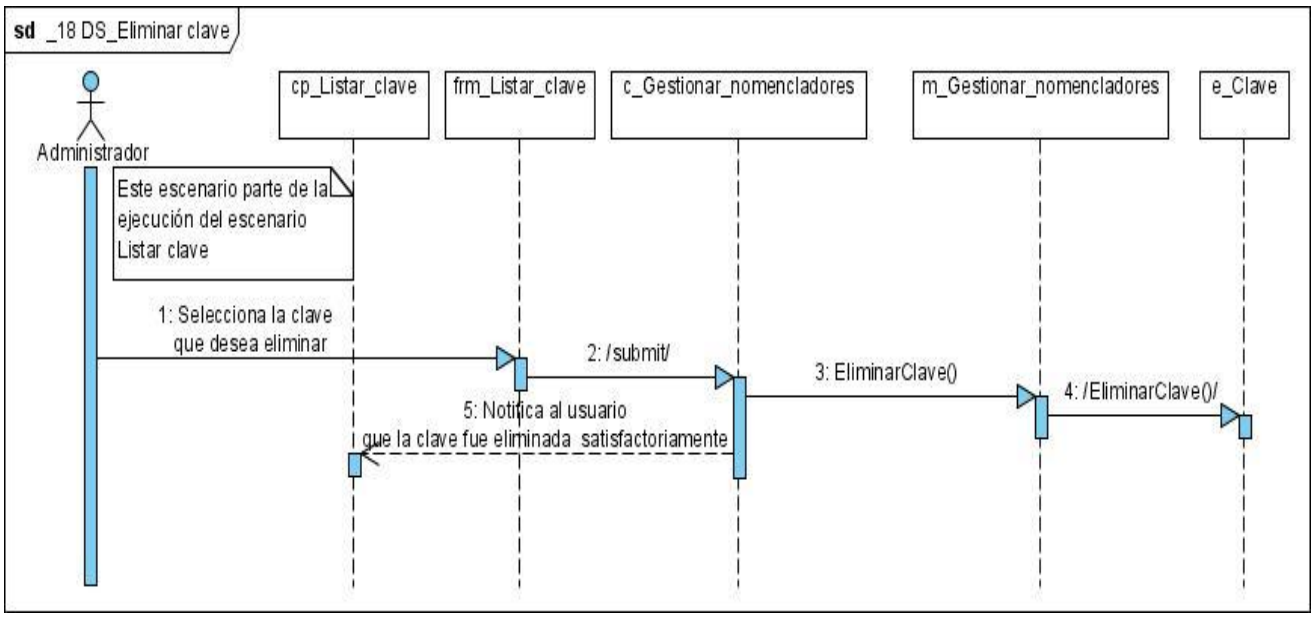


Figura 3.14 DS_Eliminar_clave

**Conclusiones**

En este capítulo se seleccionaron un conjunto de patrones de diseño; los cuales, fueron aplicados al diseño del software que dará cumplimiento al objetivo específico planteado. Para diseñar las clases de la solución informática en cuestión, se pusieron en práctica, el patrón Modelo-Vista-Controlador y un grupo de los patrones de asignación de responsabilidades: GRASP, alegando que, un buen diseño de clases es aquel en el que son aplicados muchos patrones, que en pocas palabras, “un patrón es un par problema/solución con nombre que se puede aplicar en nuevos contextos, con consejos acerca de cómo aplicarlo en nuevas situaciones y discusiones sobre sus compromisos”. (34)

En fin, se obtuvo un Modelo de Diseño consecuente tanto con los requisitos funcionales, como con los no funcionales, con los cuales debía cumplir la aplicación web a modelar; obteniéndose artefactos tales como: diagramas de clases y de secuencia, los cuales se espera que sirvan como entrada principal para implementación del software, brindando un alto cúmulo de información a los desarrolladores del mismo.

**Conclusiones**

- Se definió el proceso de gestión de demandas en los CCPEM de Cuba.
- Se determinó que los sistemas similares existentes a nivel internacional y nacional, no cubren las necesidades actuales de informatización de los CCPEM.
- Se aplicó el modelo más idóneo para la ingeniería de requisitos del proyecto, que resultó ser, el Modelo de Referencia para los Proyectos de Bioinformática (INeR).
- Fue puesta en práctica en el desarrollo de la solución, la propuesta de arquitectura definida por el MINSAP y el área temática.
- Se utilizó la metodología de desarrollo: RUP, la herramienta de modelado: Visual Paradigm y el lenguaje de modelado: UML en el desarrollo de la solución.
- Se obtuvo un Modelo de Negocio congruente con la realidad del funcionamiento diario de los CCPEM.
- Se redactó el Documento de Especificación de Requisitos del Software ajustado a las necesidades de funcionamiento del sistema informático a desarrollar.
- Se obtuvo el Modelo de Sistema del Módulo Centro Coordinador Provincial evidenciando la integración de los usuarios con el sistema.
- Se aplicaron un conjunto de métricas para evaluar los resultados obtenidos en el Flujo de Trabajo de Requerimientos.
- Se analizaron los resultados obtenidos a partir de las métricas aplicadas, corrigiéndose los errores detectados.
- Se obtuvo el Modelo de Diseño del Módulo Centro Coordinador Provincial, aplicando el patrón Modelo-Vista-Controlador y patrones Grasp, con el objetivo de que sirva, como guía fundamental, a los desarrolladores en la implementación del software.

**Recomendaciones**

- Desarrollar futuras versiones en las que se tenga en cuenta, cómo garantizar el bajo acoplamiento entre el software y los diferentes componentes que utiliza al integrarse al SiSalud.
- Integrar el software con otros componentes que gestionen la disponibilidad de camas y ventiladores de las terapias de emergencias, la localización de las ambulancias a través de GPS, así como integrarlo con el Módulo Centro Coordinador de Emergencia Médica Nacional.
- Diseñar nuevas funcionalidades que aumenten el alcance y calidad del software.

## Referencias bibliográficas

1. **Acosta, Álvaro Sosa.** *Urgencia, Emergencias, Terapias y Ambulancias. Proyecto de resolución de urgencias. Informe del SIUM.* Habana : s.n., 2005.

2. **Pressman, Roger S.** *Ingeniería del Software: Un Enfoque Práctico.* [Digital] s.l. : MC Graw Hill, 2002.

3. [En línea] [Citado el: 22 de Noviembre de 2007.] http://www.professional-soft.com/index.asp.

4. [En línea] [Citado el: 22 de Noviembre de 2007.] http://www.ambuwin.com/.

5. [En línea] [Citado el: 22 de Noviembre de 2007.] http://www.seinco.es/software/gestion-tts2000.asp.

6. **Lauzurica, Belkis Díaz.** *Sistema Informático para el Control de la Urgencia Médica (SICUM).* La Habana : s.n., 2001.

7. **Teruel, Karina Pérez.** *Modelo de Referencia para la Ingeniería de Requisitos en proyectos de Bioinformática (INeR). Tesis de Maestría.* 2007.

8. [En línea] [Citado el: 26 de Noviembre de 2007.] http://www.info-ab.uclm.es/asignaturas/42530/pdf/M1tema3.pdf.

9. **Lilianne Cantillo Romero, Karelys Mesa Pérez.** *Ingeniería de requisitos: elicitación, análisis y negociación, y especificación. Trabajo de Diploma.* [Digital] La Habana : s.n., 2007.

10. *Ídem a Referencia 8.*

11. *Ídem a Referencia 9.*

12. *Ídem a Referencia 7.*

13. [En línea] [Citado el: 10 de Enero de 2008.] http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html.

14. *Ídem a Referencia 13.*

15. [En línea] [Citado el: 10 de Enero de 2008.] http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/heterodox.asp#8.

16. *Ídem a Referencia 13.*

17. *Ídem a Referencia 13.*

18. *Ídem a Referencia 13.*

19. **Ivar Jacobson, Grady Booch, James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* España : s.n., 2000.

20. **Addison Wesley Ed, James Rumbaugh, Ivar Jacobson y Grady Booch.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* 2000. ISBN: 84-7829-037-0.

21. [En línea] [Citado el: 10 de Enero de 2008.] http://libertonia.escomposlinux.org/story/2004/7/15/115328/134.

22. *Ídem a Referencia 1.*

23. *Conf_2_Modelacion_Negocio.* [Digital] Habana : UCI.

24. *Ídem a Referencia 22.*

25. *Ídem a Referencia 7.*

26. *Ídem a Referencia 7.*

27. *Conferencia_3_estudiantes.* [Digital] Habana : UCI.

28. *Ídem a Referencia 2.*

29. [En línea] [Citado el: 20 de Mayo de 2008.] http://web2development.blogspot.com/2007/05/patron-mvc.html.

30. **Larman, Craig.** *UML y Patrones. Una introducción al análisis y diseño orientado a objeto y al proceso unificado. 2da. Edición.*

31. —. *UML y Patrones. Introducción al análisis y diseño orientado a objetos. 1ra. Edición.* México : Prentice Hall, 1999.

32. *Ídem a Referencia 30.*

33. *Ídem a Referencia 30.*

34. *Ídem a Referencia 29.*

35. **Pública, Ministro de Salud.** *Reglamento del Servicio Médico de Ambulancias y Normas Operacionales.* La Habana : s.n., 2007.

36. **Ramírez, Anairis Alvarez.** *Propuesta automatizada de control administrativo de la sección de transporte del Sistema Integrado de Urgencias Médicas. .* Habana : s.n., 2007.

**Bibliografía**


1. [En línea] [Citado el: 22 de Noviembre de 2007.] http://www.professional-soft.com/index.asp.
2. [En línea] [Citado el: 22 de Noviembre de 2007.] http://www.ambuwin.com/.
3. [En línea] [Citado el: 22 de Noviembre de 2007.] http://www.seinco.es/software/gestion-tts2000.asp.
4. [En línea] [Citado el: 10 de Enero de 2008.] http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html.
5. [En línea] [Citado el: 10 de Enero de 2008.] http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/heterodox.asp#8.
6. [En línea] [Citado el: 10 de Enero de 2008.] http://libertonia.escomposlinux.org/story/2004/7/15/115328/134.
7. [En línea] [Citado el: 10 de Enero de 2008.] www.ciberteca.net/webmaster/php.
8. [En línea] [Citado el: 22 de Noviembre de 2007.] http://www.inergis.com/noticias/noticia022.html.
9. [En línea] [Citado el: 22 de Noviembre de 2007.] http://www.dne.sld.cu/minsap/.
10. [En línea] [Citado el: 13 de Enero de 2008.] https://pid.dsic.upv.es/C9/TIR%20-%20T%C3%A9cnicas%20Avanzadas%20en%20In/default.aspx.
11. [En línea] [Citado el: 19 de Marzo de 2008.] http://148.202.148.5/cursos/cc321/fundamentos/unidad2/tema2_1.html.
12. [En línea] [Citado el: 28 de Marzo de 2008.] http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.
13. [En línea] [Citado el: 29 de Marzo de 2008.] http://www.mcc.unam.mx/~cursos/Algoritmos/javaDC99-2/patrones.html.
14. [En línea] [Citado el: 20 de Mayo de 2008.] http://web2development.blogspot.com/2007/05/patron-mvc.html.
15. [En línea] [Citado el: 6 de Junio de 2008.] http://jorgesaavedra.wordpress.com/2006/08/17/patrones-grasp-craig-larman/.
16. [En línea] [Citado el: 26 de Noviembre de 2007.] http://www.info-ab.uclm.es/asignaturas/42530/pdf/M1tema3.pdf.

17. [En línea] [Citado el: 12 de Febrero de 2008.] http://juazammo.blogspot.com/2007/12/rup-en-proyectos.html.

18. [En línea] [Citado el: 11 de Febrero de 2008.] http://www.itbuilder.com.mx/blogs/fabiola.soto/post/RUP.aspx.

19. [En línea] [Citado el: 20 de Mayo de 2008.] http://www.calidaddelsoftware.com/modules.php?name=News&file=article&sid=277.

20. **Acosta, Álvaro Sosa. 2005.** *Urgencia, Emergencias, Terapias y Ambulancias. Proyecto de resolución de urgencias. Informe del SIUM.* Habana : s.n., 2005.

21. **Addison Wesley Ed, James Rumbaugh, Ivar Jacobson y Grady Booch. 2000.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* 2000. ISBN: 84-7829-037-0.

22. **B. Bernárdez, A. Durán, M. Toro. 2004.** *Una propuesta para la verificación de requisitos basada en métricas.* Sevilla : s.n., 2004.

23. **Barreiro, Enrique.** *tema 2 - ingeniería de requerimientos.* Vigo : s.n.

24. **Cockburn, A. 2001.** *Writing Effective Use Cases.* 2001.

25. *Conf_2_Modelacion_Negocio_*. [Digital] Habana : UCI.

26. *Conferencia_3_estudiantes.* [Digital] Habana : UCI.

27. **Hans-Erik Eriksson, Magnus Penker. 2000.** *Business Modeling with UML: Business Patterns at Work.* 2000.

28. **Ivar Jacobson, Grady Booch, James Rumbaugh. 2000.** *El Proceso Unificado de Desarrollo de Software.* España : s.n., 2000.

29. **José H. Canós, Patricio Letelier yMª Carmen Penadés.** [En línea] [Citado el: 10 de Enero de 2008.] http://www.willydev.net/descargas/prev/TodoAgil.Pdf.

30. **Larman, Craig. 1999.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos. 1ra. Edición.* México : Prentice Hall, 1999.

31. **—.** *UML y Patrones. Una introducción al análisis y diseño orientado a objeto y al proceso unificado. 2da. Edición.*

32. **Lauzurica, Belkis Díaz. 2001.** *Sistema Informático para el Control de la Urgencia Médica (SICUM).* La Habana : s.n., 2001.

33. **Lilianne Cantillo Romero, Karelys Mesa Pérez. 2007.** *Ingeniería de requisitos: elicitación, análisis y negociación, y especificación. Trabajo de Diploma.* [Digital] La Habana : s.n., 2007.

34. **Pohl, K. 1997.** *Requirements Engineering: An Overview. Encyclopedia of Computer Science and Technology.* 1997.

35. **Pressman, Roger S. 2002.** *Ingeniería del Software: Un Enfoque Práctico.* [Digital] s.l. : MC Graw Hill, 2002.

36. **Pública, Ministro de Salud. 2007.** *Reglamento del Servicio Médico de Ambulancias y Normas Operacionales.* La Habana : s.n., 2007.

37. **Ramírez, Anairis Alvarez. 2007.** *Propuesta automatizada de control administrativo de la sección de transporte del Sistema Integrado de Urgencias Médicas. .* Habana : s.n., 2007.

38. **Rolando Armas Andrade, Arturo Chamorro Gómez, Maite Montes Beobide, José Antonio. 2007.** *Desde ISO 9001 hacia CMMI, pasos para la mejora de los procesos y métricas.* Madrid : s.n., 2007.

39. **Schmuller, Joseph. 2000.** *Aprendiendo UML en 24 horas.* México : s.n., 2000.

40. **Teruel, Karina Pérez. 2007.** *Modelo de Referencia para la Ingeniería de Requisitos en proyectos de Bioinformática (INeR). Tesis de Maestría.* 2007.

**Anexos**

**Anexo 1. Descripciones textuales de los CU del Negocio.**

A continuación se muestran las descripciones textuales de los casos de uso del negocio; para obtener información más detallada, remitirse al Expediente de Proyecto.

| | |
|---|---|
| **Caso de Uso:** | Procesar_demanda |
| **Actores:** | Demandante |
| **Trabajadores:** | Enfermera_coordinadora, Controlador |
| **Resumen:** | El caso de uso se inicia cuando un demandante realiza una llamada haciendo una solicitud de emergencia médica, la Enfermera_coordinadora recibe la llamada, le asigna tipo de ambulancia e institución, le entrega esa información al Controlador y este se lo comunica al chofer de la ambulancia en cuestión, por planta o mediante el expedidor. La ambulancia traslada al paciente, finalizando así el CU. |
| **Precondiciones:** | Haya ocurrido una llamada telefónica solicitando ayuda especializada. |
| **Pos condiciones:** | Demanda ejecutada o atendida. |
| **Mejoras** | La base podrá saber constantemente el estado de las ambulancias. |
| **Prioridad** | Crítico |

Tabla A.1: CUN_ Procesar Demanda

| | |
|---|---|
| **Caso de Uso:** | Archivar_certificado |
| **Actores:** | Demandante |
| **Trabajadores:** | Técnico_NU |
| **Resumen:** | El caso de uso se inicia cuando un demandante, que puede ser una persona de la población, presenta un certificado en la no urgencia del CCPEM. El Técnico_NU anota datos fundamentales como: la frecuencia de las consultas y la institución de la salud a la que debe ser trasladado el paciente, archiva el certificado e informa que el paciente será incluido en el listado de demandas no urgentes a trasladar, finalizando así el CU. |
| **Precondiciones:** | -- |
| **Pos condiciones** | Certificado archivado. |

| | |
|---|---|
| **Mejoras** | Los certificados se pueden gestionar (insertar, buscar, modificar y Visualizar) |
| **Prioridad** | Crítico |

Tabla A.2: CUN_ Archivar certificado

| | |
|---|---|
| **Caso de Uso:** | Procesar_demanda_NU |
| **Actores:** | Demandante |
| **Trabajadores:** | Enfermera_coordinadora _NU, Controlador_NU |
| **Resumen:** | El caso de uso se inicia cuando la Enfermera_coordinadora_NU, recibe una llamada de una persona, ya sea de la población o un médico desde un hospital, solicitando un servicio de traslado para un paciente y lo anota en el listado de demandas no urgentes del día siguiente si está libre el horario solicitado. Luego de las 4:00 pm se comunica con las bases de ambulancias para informarles los casos que le corresponden a cada una para el día siguiente y entrega el listado el listado de demandas no urgentes al Controlador_NU, finalizando así el CU |
| **Precondiciones:** | Se tienen que haber ejecutado los casos de uso: Archivar certificado y Confeccionar Listado de Demandas NU |
| **Pos condiciones** | Listado de demandas NU actualizado y completado |
| **Mejoras** | -- |
| **Prioridad** | Crítico |

Tabla A.3: CUN_ Procesar_demanda_NU

| | |
|---|---|
| **Caso de Uso:** | Confeccionar_Listado_Demandas_NU |
| **Actores:** | Enfermera_coordinadora_NU |
| **Trabajadores:** | Técnico_NU |
| **Resumen:** | El caso de uso se inicia cuando la Enfermera_coordinadora_NU solicita conocer el Listado de las demandas NU, para poder incluir las nuevas solicitudes de servicio de la población, para lo cual solicita al técnico dicho listado. El técnico confecciona el listado incorporando los casos que constituyen certificados por oficio y así finaliza el CU. |

| | |
|---|---|
| **Precondiciones:** | Se deben haber recepcionado con dos o más días de antelación los Certificados de los pacientes que tienen turno para el día que se está planificando. |
| **Pos condiciones** | Listado de demandas no urgentes creado. |
| **Mejoras** | -- |
| **Prioridad** | Crítico |

Tabla A.4: CUN_ Confeccionar_Listado_Demandas_NU

| | |
|---|---|
| **Caso de Uso:** | Controlar_Móvil_NU |
| **Actores:** | Centro Coordinador Provincial de Emergencia Médica (CCPEM) |
| **Trabajadores:** | Controlador_NU |
| **Resumen:** | El caso de uso se inicia cuando el CCPEM necesita controlar el movimiento de las ambulancias mientras ejecutan una demanda no urgente. Todo este control se lleva a cabo, a través de claves que se registran constantemente y que informan el estado en el que se encuentra el móvil en todo momento, finalizando así el CU. |
| **Precondiciones:** | Se le debe haber entregado al controlador el Listado de las Demandas NU para hacer durante el día, así como el documento Control del Móvil de NU |
| **Pos condiciones** | Control diario de los móviles. |
| **Mejoras** | -- |
| **Prioridad** | Crítico |

Tabla A.5: CUN_ Controlar_Móvil_NU

| | |
|---|---|
| **Caso de Uso:** | Controlar_trabajo_diario |
| **Actores:** | Jefe_del_Centro_Coordinador |
| **Trabajadores:** | Jefe_de_Guardia |

| | |
|---|---|
| **Resumen:** | El caso de uso se inicia cuando el Jefe del Centro Coordinador necesita tener el control de las actividades desarrolladas durante el día en el Centro Coordinador; para lo cual solicita al Jefe de guardia la información pertinente, el cual confecciona la información a partir de las demandas archivadas y obtiene el reporte diario, el listado de las enfermedades trazadoras, así como la hoja de cargo, correspondientes a su día de guardia, finalizando así el CU. |
| **Precondiciones:** | Se debe haber registrado las demandas del día. |
| **Pos condiciones** | Confección de reportes estadísticos |
| **Mejoras** | Confeccionar Reporte de entrega de guardia de la no urgencia. |
| **Prioridad** | Auxiliar |

Tabla A.6: CUN_ Controlar_trabajo_diario

**Anexo2. Descripciones textuales de los CU del Sistema.**

A continuación se muestran las descripciones textuales de los casos de uso del sistema; para obtener información más detallada, remitirse al Expediente de Proyecto.

| | |
|---|---|
| **Caso de Uso:** | Insertar_demanda |
| **Actores:** | Enfermera_coordinadora |
| **Resumen:** | Este CU se inicia cuando la Enfermera_coordinadora selecciona la opción Insertar Demanda, una vez que ha recibido una llamada informando una emergencia médica, luego, se muestra un una pantalla con todos los datos que componen una demanda, introduce los datos de la demanda, finalizando así el CU. |
| **Precondiciones:** | Antes de iniciarse el CU la Enfermera Coordinadora debe haberse autenticado en el sistema. |

| | |
|---|---|
| **Referencias** | RF 1.1 |
| **Prioridad** | Crítico |
| **Poscondiciones** | Demanda registrada en el sistema |

Tabla A.7: CUS_ Insertar_demanda

| | |
|---|---|
| **Caso de Uso:** | Gestionar_demanda |
| **Actores:** | Controlador |
| **Resumen:** | Este CU se inicia cuando el Controlador selecciona la opción Listar o Modificar los datos de una demanda. Se muestra una lista con las demandas que le corresponden al controlador, según los municipios que este atiende y da la posibilidad de modificar una demanda que sea seleccionada, mostrando una pantalla con los datos a modificar. El Controlador ejecuta la acción deseada y finaliza así el CU. |
| **Precondiciones:** | 1- Antes de iniciarse el CU el Controlador debe haberse autenticado en el sistema.<br>2- Para que se ejecuten los escenarios Modificar_demanda (Asignar_Ambulancia) y Modificar_Demanda (Modificar_clave) tiene que haberse ejecutado el escenario Listar_demanda.<br>3- Para que se ejecute el escenario Modificar_demanda (Modificar_clave) tiene que haberse ejecutado el escenario Modificar_Demanda (Asignar_Ambulancia) |
| **Referencias** | RF1.2, RF 1.3 |
| **Prioridad** | Crítico |
| **Poscondiciones:** | Demandas listadas y actualizadas. |

Tabla A.8: CUS_Gestionar_demanda

| | |
|---|---|
| **Caso de Uso:** | Gestionar_certificado |

| | |
|---|---|
| **Actores:** | Enfermera_coordinadora_NU |
| **Resumen:** | Este CU lo inicia la Enfermera_coordinadora_NU cuando selecciona la opción insertar, modificar, buscar o visualizar los datos de un certificado, mostrándose la pantalla correspondiente a cada una de estas funcionalidades. La Enfermera_coordinadora_NU ejecuta la acción deseada y finaliza así el CU. |
| **Precondiciones** | 1- Antes de iniciarse el CU la Enfermera Coordinadora NU debe haberse autenticado en el sistema.<br>2- Para que se ejecuten los escenarios Modificar_certificado y Visualizar_certificado tiene que haberse ejecutado el escenario Buscar_certificado. |
| **Referencias** | RF 2.1, RF 2.2, RF 2.3 y RF 2.4 |
| **Prioridad** | Crítico |
| **Poscondiciones** | Certificado insertado, buscado, modificado y eliminado |

Tabla A.9: CUS_ Gestionar_certificado

| | |
|---|---|
| **Caso de Uso:** | Insertar_demanda_NU |
| **Actores:** | Enfermera_coordinadora_NU |
| **Resumen:** | Este CU lo inicia la Enfermera_coordinadora_NU cuando selecciona la opción Insertar los datos de una demanda no urgente, mostrándose la pantalla correspondiente a esta funcionalidad. La Enfermera_coordinadora_NU ejecuta la acción deseada y finaliza así el CU. |
| **Precondiciones:** | 1- Antes de iniciarse el CU la Enfermera Coordinadora NU debe haberse autenticado en el sistema.<br>2- Este CU se ejecuta a partir de la ejecución del CU incluido Buscar demanda NU. |

| | |
|---|---|
| **Referencias** | RF3.1, CU incluido Buscar_demanda_NU. |
| **Prioridad** | Crítico |
| **Poscondiciones** | Demanda_NU registrada en el sistema. |

Tabla A.10: CUS_ Insertar_demanda_NU

| | |
|---|---|
| **Caso de Uso:** | Buscar_demanda_NU |
| **Actores:** | Controlador_NU (Generaliza a la Enfermera_coordinadora_NU) |
| **Resumen:** | Este CU lo inicia el Controlador_NU cuando selecciona la opción buscar una o un grupo de demandas no urgentes según los criterios de búsqueda especificados. El Controlador_NU introduce los parámetros de búsqueda y finaliza así el CU. |
| **Precondiciones:** | 1- Antes de iniciarse el CU el Controlador_NU debe haberse autenticado en el sistema. |
| **Referencias** | RF 3.2 |
| **Prioridad** | Crítico |
| **Poscondiciones** | Listado de demanda(s) NU. |

Tabla A.11: CUS_ Buscar_demanda_NU

| | |
|---|---|
| **Caso de Uso:** | Gestionar_demanda_NU |
| **Actores:** | Controlador_NU (Generaliza a la Enfermera_coordinadora_NU) |
| **Resumen:** | Este CU lo inicia el Controlador_NU cuando selecciona la opción modificar o visualizar las demandas no urgentes, mostrando la pantalla correspondiente a cada una de estas funcionalidades. El Controlador_NU ejecuta la acción deseada y finaliza así el CU. |
| **Precondiciones:** | 1- Antes de iniciarse el CU el Controlador NU debe haberse autenticado en el sistema.<br>2- Este CU se ejecuta a partir de la ejecución del CU incluido Buscar |

| | |
|---|---|
| | demanda NU. |
| **Referencias** | RF 3.3 y RF 3.4, CU incluido Buscar_demanda_NU. |
| **Prioridad** | Crítico |
| **Poscondiciones** | Demanda_NU actualizada y visualizada |

Tabla A.12: CUS_ Gestionar_demanda_NU

| | |
|---|---|
| **Caso de Uso:** | Gestionar_clave |
| **Actores:** | Administrador |
| **Resumen:** | Este CU lo inicia el Administrador cuando selecciona la opción Insertar, Modificar, Listar o Eliminar las claves, mostrándose la pantalla correspondiente a cada una de estas funcionalidades. El Administrador ejecuta la acción deseada y finaliza así el CU. |
| **Precondiciones:** | 1- Antes de iniciarse el Administrador del sistema debe haberse autenticado en el sistema.<br>2- La ejecución de los escenarios Modificar y Visualizar clave parten de la ejecución del escenario Listar clave. |
| **Referencias** | RF 8.1, RF 8.2, RF 8.3 y RF 8.4 |
| **Prioridad** | Crítico |
| **Poscondiciones** | Clave insertada, listada, modificada o eliminada. |

Tabla A.13: CUS_Gestionar_clave

| | |
|---|---|
| **Caso de Uso:** | Gestionar_estado_conciencia |
| **Actores:** | Administrador |
| **Resumen:** | Este CU lo inicia el Administrador cuando selecciona la opción Insertar, Modificar, Listar o Eliminar estados de conciencia, mostrándose la pantallas correspondiente a cada una de estas funcionalidades. El Administrador |

| | |
|---|---|
| | ejecuta la acción deseada y finaliza así el CU. |
| **Precondiciones:** | 1- Antes de iniciarse el Administrador del sistema debe haberse autenticado en el sistema.<br>2- La ejecución de los escenarios Modificar y Visualizar estado de conciencia parten de la ejecución del escenario Listar_estado_conciencia. |
| **Referencias** | RF9.1, RF9.2, RF9.3 y RF9.4. |
| **Prioridad** | Crítico |
| **Poscondiciones** | Estado_conciencia insertado, listado, modificado o eliminado. |

Tabla A.14: CUS_ Gestionar_estado_conciencia

| | |
|---|---|
| **Caso de Uso:** | Gestionar_clasificación_problema |
| **Actores:** | Administrador |
| **Resumen:** | Este CU lo inicia el Administrador cuando selecciona la opción Insertar, Modificar, Listar o Eliminar una clasificación de un problema, mostrando la pantalla correspondiente a cada una de estas funcionalidades. El Administrador ejecuta la acción deseada y finaliza así el CU. |
| **Precondiciones:** | 1- Antes de iniciarse el Administrador del sistema debe haberse autenticado en el sistema.<br>2- La ejecución de los escenarios Modificar y Visualizar clasificación de un problema parten de la ejecución del escenario Listar_clasificación_problema. |
| **Referencias** | RF 10.1, RF 10.2, RF 10.3 y RF 10.4 |
| **Prioridad** | Crítico |
| **Poscondiciones** | Clasificación_problema insertada, listada, modificada o eliminada |

Tabla A.15: CUS_ Gestionar_clasificación_problema

| **Caso de Uso:** | Gestionar_problema_médico |
|---|---|
| **Actores:** | Administrador |
| **Resumen:** | Este CU lo inicia el Administrador cuando selecciona la opción Insertar, Modificar, Listar o Eliminar un problema médico, mostrando la pantalla correspondiente a cada una de estas funcionalidades. El Administrador ejecuta la acción deseada y finaliza así el CU. |
| **Precondiciones:** | 1- Antes de iniciarse el Administrador del sistema debe haberse autenticado en el sistema.<br>2- La ejecución de los escenarios Modificar y Visualizar un problema médico parten de la ejecución del escenario Listar_problema_medico. |
| **Referencias** | RF 11.1, RF 11.2, RF 11.3 y RF 11.4 |
| **Prioridad** | Crítico |
| **Poscondiciones** | Problema médico insertado, listado, modificado o eliminado |

Tabla A.16: CUS_ Gestionar_problema_médico

**Glosario de Términos**

Para la realización del glosario de términos se ha utilizado como referencia bibliográfica: (35), para definir el significado de los vocablos o expresiones relacionados con la atención urgente o no urgente al paciente.

**Ambulancias Grandes o de Emergencia Médica:** son las Ambulancias Intensivas o de Apoyo Vital Avanzado. Están ubicadas en las Bases de cada Municipio y al servicio de este. Estratégicamente existirá alguna desplegada en un Área Intensiva Municipal, Policlínico de Urgencias, Hospitales y en la Unidad Neonatal; para las Ambulancias Intensivas Neonatales, el objetivo es disminuir tiempo de atención emergente.

**Ambulancias de Urgencias:** son para el traslado de pacientes urgentes con requerimientos de ambulancias que no constituyen una Emergencia Médica, excepto cuando por situaciones especiales se autoriza por El Puesto de Dirección de Ambulancias Provincial. Estas son Ambulancias ubicadas en las Bases de Ambulancia de cada Municipio y al servicio de este, y estratégicamente existirá alguna desplegada en un Área Intensiva Municipal, Policlínico de Urgencias, Hospitales u otra área distante, el objetivo es disminuir el tiempo de atención para la urgencia.

**Ambulancias de Transporte Sanitario No Urgente:** son para el traslado de pacientes con patologías que no son Urgencias y que por sus características específicas necesitan ser transportados en Ambulancia. Estas están ubicadas en las Bases de Ambulancia de cada Municipio.

**Aplicación (Sistema):** sistema que ofrece a un usuario final un conjunto coherente de casos de uso.

**Aplicación Web:** es una aplicación informática que los usuarios utilizan accediendo a un servidor Web a través de un navegador o browser. Estas son muy son populares debido a la habilidad para actualizar y mantener la información manipulada sin distribuir e instalar el software en miles de potenciales clientes.

**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 componen el sistema. La misma se interesa no solo por la estructura y el comportamiento, sino también por las restricciones y

compromisos de uso, funcionalidad, 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 secuencias de acciones, incluyendo variaciones, que un sistema lleva a cabo y que conduce a un resultado observable de interés para un actor determinado.

**Demanda:** ante un caso de emergencia, urgencia o no urgencia, se solicita atención especializada al Centro Coordinador Provincial de Emergencia Médica.

**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.

**Diseño:** un proceso que utiliza los productos del análisis para producir una especificación para implementar un sistema. Una descripción lógica de cómo trabajará un sistema.

**Diseño orientado a objetos:** la especificación de una solución software lógica en función de objetos software, como clases, atributos, métodos y colaboraciones.

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

**Emergencia:** enfermos o lesionados con peligro vital inmediato real o potencial. Su traslado requiere de Apoyo Vital Avanzado, después de las primeras medidas de estabilización.

**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

**Móvil:** se refiere a una ambulancia, es un vehículo que los servicios médicos utilizan para responder las llamadas de emergencia. Por eso, una ambulancia siempre está provista de las tecnologías médicas para la supervisión de la condición del paciente, el diagnóstico y el tratamiento. (36)

**Paciente Ambulanciable:** son aquellos pacientes que por su afección no puedan deambular; así como los que no deban deambular por criterio médico de su patología. Pueden considerarse excepciones por el tipo de Valencia Social de la demanda y se exponen los siguientes criterios de Paciente Ambulanciable:

a) Aquel que por su patología necesita de cuidados, vigilancia o terapéutica durante el traslado.
b) Aquel que por su patología tiene una limitación para mantener el tono muscular o tiene alguna forma de inmovilización, extricación, órtesis, u otros dispositivos que anulen la posibilidad de viajar sentado en alguna forma de transporte público.
c) Aquel que por la urgencia de su traslado, a pesar de no necesitar lo definido en el ítem anterior, no pueden esperar por la disponibilidad de otra forma de transporte.
d) Enfermos con afecciones mentales agudas o que por su estado volitivo puedan implicar agresividad o molestias.
e) De forma excepcional, y bajo solicitudes colegiadas, aquellos enfermos que por su status jurídico no sea recomendable llevar en transporte no sanitario.

**Pacientes no ambulanciables*:*** que requieren transporte especial pero no ambulancias. Es decir que pueden ser trasladados en ómnibus o taxis destinados para el traslado de pacientes. Ejemplo Exámenes o consultas médicas entre hospitales o desde lugares distantes, etcétera. Para esto se debe utilizar otro tipo de transporte sanitario que no sea ambulancias.

**Paquete:** mecanismo de propósito general para organizar elementos en grupos.

**Proyecto:** esfuerzo de un equipo de desarrollo para llevar un sistema a lo largo de un ciclo de vida.

**Servicio:** unidad de software que encapsula alguna funcionalidad de negocio y proporciona estas a otros servicios a través de interfaces públicas bien definidas.

**Servicio Web:** es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

**Servicio No Urgente de Ambulancia:** es el que se le brinda a un paciente que sin tener una urgencia requiere de una ambulancia, ejemplos: un fracturado de extremidades o cadera a consulta, un paciente con limitaciones físicas a radioterapia, etc.

**Software:** conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica, 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.

**Urgencia de primera prioridad:** enfermos o lesionados que no poseen un riesgo vital inmediato pero si puede llegar a ser de manera mediata si no se le presta la atención requerida. Algunos de estos pacientes pueden atenderse totalmente en la red de urgencia en atención primaria ya sea en el servicio de urgencia local o en el principal, necesitando traslado bajo Apoyo Vital Básico cuando así corresponda. Otros requieren ser trasladados a un hospital en igual condición.

**Urgencia de segunda prioridad:** son las urgencias que no tienen riesgo vital, ni inmediato, ni mediato para el paciente, las mismas deben tener solución en toda la red de urgencias en atención primaria de salud. No requieren ambulancias excepto cuando hay falta de capacidad resolutiva de alguna especialidad. En tal situación debe evaluarse su posible traslado en vehículo ligero, basado en las condiciones específicas de cada lugar y momento: situación del paciente, especialidad necesaria, distancia y posibilidades de vehículo ligero en la unidad, servicio de taxi o aquellos que por indicación del Gobierno local pudieran encontrarse en prestación de servicio o apoyo, etc.

**Urgencias Sentidas o No Emergencias:** son aquellos pacientes que son urgencia sólo por su apreciación o la de su familiar. Todos pueden atenderse en la red de urgencia en atención primaria y orientarse técnicamente al programa de medicina familiar.

**Vehículos ligeros de apoyo y Servicio de taxi:** en la red de urgencia de la Atención Primaria es imprescindible la existencia de una piquera de taxi al servicio de los pacientes y/o vehículo estatal de guardia como apoyo para darle respuesta a las urgencias de segunda prioridad, las cuales muchas veces se trasladan en transporte sanitario a pesar de no ser ambulanciables (ver concepto de pacienteambulanciable). Esto debe tener solución a nivel de cada territorio.