



Universidad Popular Autónoma del Estado de Puebla

Centro Interdisciplinario de Posgrados

Investigación y Consultoría

Doctorado en Tecnologías de la Información
y Análisis de decisiones.

“Una estrategia para el modelado de negocios en la Web”

Tesis Doctoral

Autor: Pedro Vargas García

Asesor: Dr. Abraham Sánchez López



UPAEP – Secretaría General

Dirección General de Apoyos Académicos

Dirección del Centro de Recursos para el Aprendizaje y la Investigación.

Biblioteca Central - **Karol Wojtyla**

Tesis Digitales Restricciones de uso:

DERECHOS RESERVADOS ©

PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de textos, imágenes, gráficas, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente de donde la obtuvo mencionando el autor o autores involucrados en el documento.

Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

DEDICATORIA

A mis Padres

Pedro Vargas Reyes (†)

Carolina García Bello

Por quienes fui en un principio

A mi Esposa

Yolanda Cardoso Lucero

Por quien soy ahora

A mis Hijos

Pedro Sebastián Vargas Cardoso

Natalia Carolina Vargas Cardoso

Carlos Emilio Vargas Cardoso

Por quienes seré

AGRADECIMIENTOS

Quiero manifestar mi agradecimiento a mis maestros en el estudio del doctorado, en particular a mi asesor de tesis, el **Dr. Abraham Sánchez López**, no solo por su incansable labor de dirección, si no por sus continuas enseñanzas y apoyo en la elaboración de este trabajo. También quiero expresar mi gratitud para los compañeros de trabajo de la Universidad Politécnica de Puebla por su respaldo personal y amistad incondicional. Finalmente agradezco el apoyo sincero de mi familia y amigos. De corazón muchas gracias a todos.

CONTENIDO

ÍNDICE DE FIGURAS.	7
ÍNDICE DE TABLAS	9
Capítulo 1	11
1.1 Introducción.	11
1.2 Objetivo general.	13
1.3 Objetivos específicos.	13
1.4 Justificación.	14
1.4.1 <i>Conveniencia.</i>	14
1.4.2 <i>Relevancia Social.</i>	14
1.4.3 <i>Implicaciones prácticas.</i>	15
1.5 Viabilidad de la investigación.	15
1.6 Alcances y limitaciones.	15
1.6.1 <i>Alcances.</i>	15
1.6.2 <i>Limitaciones.</i>	16
1.7 Resultados esperados.	16
1.8 Organización del estudio.	16
Capítulo 2	19
2.1 Introducción al modelado de negocios.	19
2.2 Definición de procesos de negocio.	21
2.3 Requisitos de un lenguaje de modelado.	21
2.4 Metodologías de modelado.	22
2.5 Perspectivas de un proceso de negocio.	23
2.6 Lenguajes de definición de procesos de negocio.	24
2.6.1 <i>Modelos basados en redes de Petri.</i>	24
2.6.2 <i>Modelos basados en UML.</i>	26
2.6.3 <i>Otros lenguajes y formalismos.</i>	26
2.7 Conclusión.	27
Capítulo 3	18
3.1 Introducción a la arquitectura, especificación y diseño de sistemas orientados a servicios.	29
3.2 Antecedentes.	29

3.3	Sistemas de información Par a Par.	31
3.4	Alianzas.	33
3.5	Confederaciones de Software.	35
3.5.1	<i>E-Gobierno: confederación vía integración.</i>	35
3.5.2	<i>Sistemas de Manufactura: Descomposición e Integración.</i>	36
3.5.3	Mejora en Middleware de las confederaciones.	38
3.5.4	Procesos de negocios en las confederaciones.	40
3.6	Conclusión.	41
Capítulo 4		43
4.1	Introducción al modelado Web.	43
4.2	Atributos de los sistemas y aplicaciones basados en la Web.	44
4.3	Estratos de la ingeniería de aplicaciones Web.	46
4.3.1	<i>Proceso.</i>	47
4.3.2	<i>Métodos.</i>	48
4.4	Herramientas y tecnología.	49
4.5	El proceso de Ingeniería Web.	49
4.5.1	<i>Definición del marco de trabajo.</i>	50
4.5.2	<i>Refinamiento del marco de trabajo.</i>	52
4.6	Mejores prácticas en Ingeniería Web.	53
4.7	Un mecanismo formal de extensión a UML.	53
4.7.1	<i>Modelando.</i>	53
4.7.2	<i>Arquitectura de aplicaciones Web.</i>	56
4.7.3	<i>Modelando páginas Web.</i>	58
4.7.4	<i>Formularios.</i>	62
4.7.5	<i>Marcos.</i>	64
4.8	Metodología WAE para el desarrollo de las aplicaciones WEB con UML.	66
4.9	Estereotipos.	66
4.9.1	<i>Servidor Scripting.</i>	70
4.9.2	<i>Cliente Scripting.</i>	70
4.10	Conclusión.	70
Capítulo 5		72
5.1	Introducción a la propuesta metodológica.	72

5.2	Propuesta de modelado.	73
5.3	Modelado de Negocios.	75
5.3.1	<i>Identificar los procesos de negocio.</i>	75
5.3.2	<i>Identificar los roles del entorno de negocio.</i>	76
5.3.3	<i>Describir los casos de uso del negocio.</i>	78
5.3.4	<i>Especificación de las reglas de negocios.</i>	82
5.4	Modelado de requisitos: modelos conceptual y de casos de uso iniciales.	84
5.4.1	<i>Obtención del modelo inicial de casos de uso del sistema.</i>	85
5.4.2	<i>Obtención del modelo conceptual inicial.</i>	88
5.4.3	<i>Representación de la reglas del negocio.</i>	90
5.4.4	<i>Identificación de requisitos no funcionales.</i>	91
5.5	Modelado Web.	91
Capítulo 6		104
6.1	Introducción al modelo centrado en arquitectura de procesos.	104
6.2	Antecedentes.	105
6.3	Método exhaustivo.	108
6.4	Filosofía de diseño.	108
6.4.1	<i>Principios Empresariales.</i>	108
6.4.2	<i>Principios Técnicos.</i>	109
6.4.3	<i>Principios de solución.</i>	109
6.4.4	<i>Principios de los datos.</i>	110
6.4.5	<i>Principios de servicio.</i>	110
6.5	Planteamiento general.	111
6.5.1	<i>Modelo centrado en arquitectura de procesos (MAP).</i>	114
6.5.2	<i>Meta arquitectura.</i>	115
6.5.3	<i>Arquitectura conceptual.</i>	116
6.5.4	<i>Arquitectura lógica.</i>	117
6.5.5	<i>Arquitectura física.</i>	118
6.5.6	<i>Despliegue de la arquitectura.</i>	118
6.5.7	<i>Arquitectura de gestión /administración.</i>	119
6.5.8	<i>Arquitectura de la información.</i>	120
6.5.9	<i>Aspecto de la arquitectura.</i>	121

6.5.10	<i>Arquitectura de componentes.</i>	122
6.6	Conclusión.	126
Capítulo 7		128
7.1	Resultados.	128
7.1.1	Modelo de negocio.	131
7.1.2	Modelo de requerimientos.	133
7.1.3	Modelo Web.	134
7.2	Conclusiones.	135
7.3	Trabajo Futuro - Análisis navegacional del sistema.	143
REFERENCIAS.		150

ÍNDICE DE FIGURAS.

Figura 2.1.	Proceso de gestión de reclamaciones modelado con redes de Petri.	25
Figura 3.1.	Rol Central de la Arquitectura del Sistema	30
Figura 3.2	Arquitectura de un Sistema Orientado a Servicios.	33
Figura 3.3	Interfaz del administrador del sistema de manufactura.	37
Figura 3.4	Tres servicios interrelacionados con un portal G y sus ligas de comunicación	39
Figura 3.5	Sistema con portales front-end.	39
Figura 4.1	Estratos de la Ingeniería de Software.	47
Figura 4.2	Proceso de Ingeniería Web.	52
Figura 4.3	Asociación entre clases estereotipadas.	55
Figura 4.4	Colaboración del cliente.	61
Figura 4.5	Colaboración del servidor	61
Figura 4.6	Forms presentado al servidor de páginas	63
Figura 4.7	Ejemplo de frames.	65
Figura 5.1	Diagrama de trazabilidad entre el modelo de negocios, de lo requisitos del sistema y la Web.	74
Figura 5.2	Diagrama de caso de uso del negocio.	77
Figura 5.3	Proceso para atender consultas, correspondiente al objetivo estratégico 1.	77
Figura 5.4	Proceso de atención de emergencia según el objetivo estratégico 2.	78
Figura 5.5	Proceso de Suscripción de nuevos clientes del objetivo estratégico 3.	78
Figura 5.6	Descripción del caso de uso Suscribir.	79

Figura 5.7 Diagrama de roles para el caso de uso de Atención de emergencia.	80
Figura 5.8 Diagrama de secuencia para el caso de uso de Atención de emergencia	81
Figura 5.9 Diagrama de proceso.	82
Figura 5.10 Extracto del glosario: objetos de información y actividades.	84
Figura 5.11 Diagrama inicial de caso de uso del sistema.	85
Figura 5.12 Descripción del caso de uso del sistema	86
Figura 5.13 Modelo conceptual inicial para el caso de uso del negocio solicitud de atención.	89
Figura 5.14 Caso de uso Acceso.	92
Figura 5.15 Caso de uso Cuentas de Usuario	93
Figura 5.16 Diagrama de colaboración Acceso_sistema.	94
Figura 5.17 Diagrama Web Acceso	94
Figura 5.18 Caso de uso Altas	95
Figura 5.19. Diagrama de actividades “Altas”.	96
Figura 5.20 Diagrama Web “Actualizar”.	96
Figura 5.21 Caso de uso Consultas.	97
Figura 5.22 Diagrama de colaboración Consultas.	98
Figura 5.23 Diagrama Web Consultas.	98
Figura 5.24 Casos de uso Estados	99
Figura 5.25 Diagrama Web Estados.	100
Figura 5.26 Diagrama de Colaboración principal	101
Figura 5.27 Diagrama Web Servicios	101
Figura 5.28 Caso de uso suscribir.	102
Figura 5.29 Diagrama de caso de uso Suscribir.	102
Figura 5.30 Diagrama Web Suscribir.	103
Figura 6.1. Dimensiones RSVP de MAP.	112
Figura 6.2 Niveles de arquitectura MAP	114
Figura 6.3 Meta Arquitectura en MAP	115
Figura 6.4 Arquitectura Conceptual en MAP.	116
Figura 6.5 Arquitectura Lógica de MAP	117
Figura 6.6. Arquitectura Física en MAP	118
Figura 6.7 Despliegue de la arquitectura en MAP	119
Figura 6.8 Gestión de la Arquitectura en MAP	120

Figura 6.9 Arquitectura de la Información en MAP	121
Figura 6.10 Aspecto de la Arquitectura en MAP.	122
Figura 6.11 Arquitectura de Componentes en MAP.	123
Figura 7.1 Infraestructura de comunicaciones para el sistema TRMCS.	131

ÍNDICE DE TABLAS

Tabla 2-1. Principales conceptos relacionados con procesos de negocio.	20
Tabla 4-1 Elementos de la página del servidor	66
Tabla 4-2. Elementos de la página del cliente	66
Tabla 4-3. Elementos del Formulario	67
Tabla 4-4 Elementos del Frame Set	67
Tabla 4-5 Elementos de Target	68
Tabla 4-6. Elementos de JavaScript	68
Tabla 6-1. Tarjeta de Referencia del Modelo centrado en Arquitectura de Procesos	124
Tabla 7-1 Comparativa con la Meta arquitectura	136
Tabla 7-2 Comparativa Arquitectura conceptual.	137
Tabla 7-3 Comparativa Arquitectura lógica.	137
Tabla 7-4 Comparativa Arquitectura lógica.	138
Tabla 7-5 Comparativa Arquitectura de despliegue.	139
Tabla 7-6 Comparativa gestión/administración de la arquitectura.	140
Tabla 7-7 Comparativa Arquitectura de la información.	140
Tabla 7-8 Comparativa aspectos de la arquitectura.	141
Tabla 7-9 Comparativa arquitectura de los componentes.	142
Tabla 7-10 Actividades futuras para la elaboración del modelo navegacional del sistema.	147

Resumen

El presente trabajo de investigación tiene como objetivo proponer una estrategia que permita elaborar el modelo de un sistema de negocios que funciona en la Web, partiendo de los objetivos de negocio de la organización y culminando en la construcción del modelo en la Web del sistema, integrando las herramientas de la ingeniería de software existentes, proponiendo las adecuaciones pertinentes a los desarrollos previos y ocupando las reglas de especificación formal en aquellos aspectos que sea necesario.

Teniendo como punto de partida determinar los procesos del negocio de la organización bajo estudio, y de describir sus flujos de trabajo mediante diagramas de actividades, los casos de uso se identifican y se estructuran a partir de las actividades de cada proceso, mientras que los conceptos del modelo conceptual se obtienen a partir de los datos que fluyen entre las actividades. Las reglas del negocio se identifican e incluyen en un glosario, como parte de la especificación de datos y actividades. El modelado conceptual y el de casos de uso se utilizan en paralelo, lo cual facilita la identificación y especificación de casos de uso adecuados.

En la segunda parte del trabajo de investigación se ocupó una estrategia centrada en arquitectura de procesos como un referente comparativo para verificar la funcionalidad y pertinencia de la estrategia propuesta. Dicha estrategia propone como una alternativa para enfrentar de manera eficaz la complejidad del diseño de arquitectura y la administración de la arquitectura de procesos y el ciclo de vida del desarrollo de los sistemas de información en un paradigma orientados a servicios.

Con base en la elaboración de los modelos correspondientes a las estrategias antes mencionadas, se elaboró un comparativo entre ambas con el propósito de identificar fortalezas y debilidades de cada una de ellas.

Capítulo 1

1.1 Introducción.

Según la definición de Pressman (2005), la Ingeniería del Software es una disciplina o área de la computación que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelvan temas de todo tipo. Esta definición permite incluir dentro de la ingeniería del software a un gran número de áreas diversas de la computación, desde desarrollos de sistemas operativos a la construcción de compiladores o los nuevos desarrollos de sistemas en la Web.

La aplicación de la ingeniería del software permite obtener software de modo rentable y de calidad, asegurando que los resultados se adecuen a las necesidades reales del usuario. Esta ventajosa característica de la ingeniería del software ha provocado que, cuando los sistemas evolucionaron al entorno de la hipermedia y de la Web, los grupos de investigación encaminaron sus trabajos a encontrar métodos y técnicas enmarcados dentro de la ingeniería del software que se adaptaran a las nuevas características de estos entornos. Este fue el comienzo de lo que hoy se denomina Ingeniería Web (Desphande et al. 2002).

Desde que, con la propuesta de HDM hecha por Garzotto et al. (1993), se comenzó a hablar de diferentes métodos para sistemas hipermedia, se sientan las bases para el nacimiento de la ingeniería Web. En los primeros trabajos se analizó porqué la nueva tendencia de desarrollo hipermedia y Web requerían de nuevas técnicas de desarrollo, diferentes a las propuestas de los métodos clásicos de la ingeniería de software.

Analizando los estudios de Lang (2002) y Baresi et al. (2001) se puede concluir que las diferencias principales que obligan a la inclusión de nuevos modelos y adaptación de los existentes son:

1. La complejidad de la estructura hipermedia: el complejo sistema navegacional que se establece debido a la existencia de hiperenlaces e hipervínculos causa, en muchas

ocasiones, que el usuario pueda perderse en el hiperespacio, si el sistema no se construye basado en alguna técnica que garantice la calidad del mismo (Shubin & Meehan, 1997).

2. La colaboración interdisciplinar: la gran cantidad y diversidad de roles que deben tomar parte en el desarrollo de sistemas en la Web complican su desarrollo en comparación con los sistemas clásicos (Koch 2001). Roles como los analistas, los diseñadores gráficos, los expertos en marketing, los usuarios finales, entre otros, deben trabajar con técnicas que sean amigables para que cada uno de ellos pueda elaborar bien su trabajo.
3. La visión externa de la Web: mientras que en los sistemas clásicos el número de usuarios finales estaba normalmente definido y cerrado o al menos controlado a un grupo de usuarios concreto, en los sistemas Web la visión cambia. El número de usuarios no es sólo ilimitado, si no que, además, es totalmente impredecible en la mayoría de los casos. Los roles de los usuarios finales que interactúan con el sistema es abierto y hay que adaptar el desarrollo para que se puedan satisfacer las necesidades de cualquier usuario anónimo sin llegar a conocer su perfil.
4. El mantenimiento: al ser más abiertos, los sistemas Web requieren de un mantenimiento aumentativo, adaptativo, así como correctivo, más crítico que en los sistemas clásicos, con el añadido de que, además, los sistemas Web normalmente deben funcionar las veinticuatro horas del día los trescientos sesenta y cinco días del año, lo que implica disponer de menos tiempo para las tareas de mantenimiento.

Aunque existen más diferencias, éstas son las más destacadas en todos los estudios comparativos.

Aunado a lo anterior, las tecnologías de gestión de procesos de negocio sufrieron en los 90's una revolución en la forma de coordinar las actividades internas de las organizaciones. En esta década, se comenzaron a utilizar las tecnologías en entornos orientados a los negocios, como una herramienta para facilitar las colaboraciones entre organizaciones. Actualmente se aplican para realizar la composición de servicios en entornos diseñados conforme a la computación orientada a servicios.

Una aplicación Web orientada a procesos de negocio es un programa de computadora cuya principal característica es su muy alto nivel de abstracción, por lo que el modelado del proceso de

negocios es susceptible de tener errores principalmente por el alto grado de concurrencia, la computación distribuida y la dependencia de entidades externas que las caracterizan.

Actualmente una de las herramientas de ingeniería de software de mayor uso es la construcción de modelos usando el Lenguaje de Modelado Unificado (UML). La mayor parte de los modelos que se desarrollan con UML son guiados por casos de uso. Aunque el éxito de los casos de uso se suele justificar con el hecho de que constituyen una técnica simple e intuitiva. Cockburn (1997) y Korson (1998 y 1999), señalan las dificultades que existen para la obtención y la especificación de casos de uso verdaderamente útiles, y la falta de consenso sobre cómo organizarlos y manejarlos.

El ámbito de esta tesis se enmarca dentro de la ingeniería del software en general, y se propone, con base en un análisis de las herramientas de modelado existentes, proponer una estrategia que permita modelar un sistema de negocios en la Web y verificar que el sistema construido con base en este modelo tenga un comportamiento conforme a sus especificaciones en cuanto a procesos de negocios, sistema de software y comportamiento en la Web.

1.2 Objetivo general.

El objetivo de esta investigación es proponer una estrategia que permita elaborar el modelo de un sistema de negocios que funciona en la Web, partiendo de los objetivos de negocio de la organización y culminando en la construcción del modelo en la Web del sistema, integrando las herramientas de la ingeniería de software existentes, proponiendo las adecuaciones pertinentes a los desarrollos previos y ocupando las reglas de especificación formal en aquellos aspectos que sea necesario.

1.3 Objetivos específicos.

1. Realizar un estudio del estado del arte sobre uso y aplicación de las herramientas de ingeniería de software para el modelado de negocios, sistemas y Web, en particular del Lenguaje de Modelado Unificado (UML) con especial énfasis en la verificación de requisitos funcionales, de procesos de negocio y composiciones de servicios Web. Analizar las ventajas y dificultades que presentan las soluciones existentes.

2. Proponer una estrategia para elaborar el modelo de un sistema de negocios en la Web, con base en lo investigado en el objetivo anterior.
3. Realizar el modelado de un caso de estudio ocupando la estrategia propuesta, con el propósito de verificar funcionalidad y pertinencia de la misma.
4. Realizar el modelado del mismo caso de estudio, ocupando algún método actual ya propuesto en la literatura para dicho propósito.
5. Comparar el proceso de construcción del modelo usando la estrategia propuesta respecto al proceso de modelado mediante algún método existente, con el propósito de validar ventajas y dificultades en ambos. Con base en lo anterior proponer una serie de extensiones y/o adecuaciones a la estrategia propuesta para eficientar la construcción y especificación del modelo de negocios en la Web.

1.4 Justificación.

La justificación de la investigación se ha dividido en tres aspectos: conveniencia, relevancia social e implicaciones prácticas, cada uno de los cuales se tratará por separado.

1.4.1 Conveniencia.

Proponer una estrategia de modelado para construir sistemas de negocios orientados a la Web a partir de las herramientas existentes y proponiendo la integración y adecuaciones de las mismas, representa una aportación a la ingeniería de Software, sobre todo tomando en cuenta que los trabajos desarrollados hasta este momento no resuelven de manera integral el modelado de sistemas de negocios en la Web.

1.4.2 Relevancia Social.

Si la estrategia propuesta resulta adecuada para el modelado de negocios en la Web, permitirá que los futuros desarrollos en Internet se puedan llevar cabo de manera más rápida, sean más robustos y mucho más confiables, abriendo la posibilidad de que cada vez un número mayor de negocios, empresas y organizaciones vean en la Internet un área de oportunidad para expandir sus actividades comerciales.

1.4.3 Implicaciones prácticas.

La estrategia propuesta, al estar basada en las herramientas existentes, permitirá que todos aquellos ingenieros encargados de modelar sistemas, que ya manejan dichas herramientas, puedan adoptar de manera sencilla ésta estrategia, facilitándoles el desarrollo de nuevos y mejores sistemas de computadoras destinados a su uso en la Web.

1.5 Viabilidad de la investigación.

Para realizar la investigación se cuenta con la infraestructura necesaria en equipo de cómputo, software y de acceso a Internet como respaldo tecnológico al llevarla a cabo. En cuanto al recurso humano, además del titular y encargado del desarrollo de la investigación, se cuenta con la colaboración, asesoría y supervisión del Dr. Abraham Sánchez López, quien, además de contar con experiencia en el campo de la investigación, hará las funciones de asesor durante el desarrollo del presente proyecto.

Se cuenta además con las suscripciones a la ACM y a la IEEE, así como el acceso a material bibliográfico digital de reciente publicación, lo que permite consultar los trabajos publicados en las últimas fechas, en el área en que se desarrolla la investigación, se cuenta además con un número considerable de libros para consulta.

Con base en lo anterior se considera que la realización de la presente investigación es viable en los términos requeridos para la elaboración de una propuesta metodológica innovadora dentro del área de desarrollo de la misma.

1.6 Alcances y limitaciones.

1.6.1 Alcances.

El presente proyecto de investigación tiene como meta la elaboración de una propuesta para el modelado de aplicaciones de negocios en la Web. La estrategia propuesta se construirá integrando algunas de las herramientas de ingeniería de software existentes en la actualidad para el modelado de sistemas en general y deberá considerar alternativas de solución para aquellos problemas identificados en los trabajos relacionados. Como parte de la misma, se elaborará un

documento que contenga, el estudio y análisis del estado del arte de las herramientas de ingeniería de software actuales en el modelado de sistemas, la descripción detallada de la estrategia propuesta, el modelado de un caso de estudio ocupando dicha estrategia, y un comparativo entre la estrategia propuesta y los métodos existentes actualmente.

1.6.2 Limitaciones.

Por cuestiones de tiempo los temas que conformen el estudio del estado del arte, no tendrán mayor profundidad que la requerida para dar sustento a la presente investigación. La estrategia propuesta se aplicará al modelado de un caso de estudio particular, aplicando una validación formal únicamente en aquellos casos en que sea necesario.

1.7 Resultados esperados.

Los resultados esperados de la presente investigación son:

- o Un estudio del estado del arte de los trabajos relacionados con el modelado de sistemas de negocios y composiciones de servicios Web.
- o El análisis de los problemas encontrados en las soluciones existentes.
- o Estudio de los lenguajes de definición de procesos de negocio más utilizados, de las necesidades que existen para llevar a cabo el modelado de negocios en la Web, de las técnicas y herramientas de verificación para determinar cuales resultan más adecuadas para las necesidades planteadas en el ámbito de los procesos de negocio.
- o Una estrategia para llevar a cabo el desarrollo de aplicaciones de negocios en la Web.
- o Un comparativo entre la estrategia propuesta y alguna de las herramientas ocupadas en la actualidad.

1.8 Organización del estudio.

El contenido de la investigación se encuentra organizado en varias secciones o capítulos, cuyo contenido se describe brevemente a continuación.

Capítulo 1: Introducción.

Contiene la introducción a la investigación, los objetivos, la justificación, los alcances y limitaciones y los resultados esperados de la investigación. Además de que contiene la descripción del contenido de los demás capítulos que conforman el documento.

Capítulo 2: Modelado de negocios.

En esta sección se hace una introducción a la necesidad de modelar los sistemas de negocios, como un mecanismo para plantear y administrar su desarrollo.

Capítulo 3: Herramientas actuales para el modelado de sistemas.

En esta sección se presenta un estudio de la metodología de modelado de sistemas, se analizan las características de cada método, las ventajas y dificultades que se tiene al aplicar cada uno de ellos.

Capítulo 4: Modelado Web.

Contiene una introducción al modelado Web, se revisan además las metodologías de modelado para aplicaciones Web.

Capítulo 5: Estrategia propuesta y su aplicación a un caso de estudio.

En esta sección se describe el caso de estudio a utilizarse en el modelado de negocios, se muestran también los diagramas principales y se describe el mecanismo seguido para obtenerlos.

Capítulo 6: Modelado centrado en la arquitectura.

En este capítulo se muestra el modelado del caso de estudio mediante el uso de la metodología centrada en la arquitectura, se hace un estudio comparativo de este modelo y el proceso descrito en el capítulo anterior.

Capítulo 7: Resultados y conclusiones.

En esta sección se realiza el análisis de las ventajas y deficiencias encontradas al modelar el caso de estudio usando la estrategia propuesta y el modelado orientado a la arquitectura, se plantean

las alternativas pertinentes para resolver los problemas encontrados. Se proponen trabajos posteriores sobre la misma línea de trabajo.

Anexos.

Contiene todos los documentos ocupados para dar sustento a la presente investigación, tal es el modelo construido del caso de estudio, entre otros.

Referencias.

Contiene las referencias a autores y libros ocupadas en la investigación, así como el conjunto de material como libros, artículo y páginas Web consultadas durante el desarrollo de la investigación

Capítulo 2

2.1 Introducción al modelado de negocios.

La gestión de procesos de negocio comprende un conjunto de tecnologías orientadas a dar soporte a la ejecución de la lógica de negocio de las organizaciones. La recepción de un pedido por parte de un distribuidor de libros implica, por ejemplo, recopilar los datos del cliente y del pedido, la comprobación de existencias en almacén, pedido a otro distribuidor o editorial si fuese necesario, o incluso su importación desde el extranjero, cobro del pedido y envío del mismo al cliente. Estas tareas deben coordinarse de acuerdo a una determinada lógica, que se representa como un *proceso de negocios*.

En la tabla 1 se muestran las definiciones de los principales conceptos relacionados con el ámbito de la gestión de procesos de negocio. Los procesos de negocio definen la forma en que las organizaciones llevan a cabo sus propósitos. Fundamentalmente, un proceso de negocio se caracteriza por: contener actividades que tienen un propósito común; llevarse a cabo colaborativamente por un grupo; sobrepasar, a menudo, las fronteras de una organización; ser iniciado como reacción a eventos originados en el exterior de la organización.

En 1995 Ould identifica tres tipos básicos de procesos de negocios en una organización: procesos centrales, procesos de soporte y procesos de gestión. Los primeros son los que directamente intentan aportar valor a los clientes. Los segundos dan soporte a las necesidades de entidades internas a la organización para indirectamente, aportar valor a los clientes. Los últimos gestionan los tipos de procesos anteriores y se encargan de la planificación al nivel de negocios.

Un *workflow* es un proceso de negocio operativo, esto es, un proceso de negocio que puede ser ejecutado automáticamente. Un sistema de gestión de *workflow* (WFMS) es el sistema de *software* que proporciona el soporte necesario para ejecutar y controlar *workflows*. El concepto de gestión de *workflow* (WFM) está, como se puede apreciar en las definiciones, muy centrado en la ejecución de los procesos. En los últimos tres o cuatro años se ha encontrado esta

aproximación demasiado restrictiva, y se acuñó un nuevo término: *Business Process Management* (BPM) como un concepto mas amplio que WFM, y su principal diferencia, según lo describen Aalst, Hofstede y Weske (2003), está en que hace especial énfasis en las fases de diagnóstico de procesos (análisis de procesos para identificar problemas o aspectos que pueden ser mejorados) y definición de procesos (diseño o rediseño de procesos), aspectos tradicionalmente desatendidos por el WFM.

Dado que esta investigación se centra en las tecnologías para la definición y análisis de procesos de negocio, en los siguientes apartados se describen estas facetas.

Tabla 2-1. Principales conceptos relacionados con procesos de negocio.

Proceso de negocio	Un conjunto de uno o más procedimientos o actividades que colectivamente realizan un objetivo de negocio o una meta, normalmente en el contexto de una estructura organizativa que define unos papeles funcionales y relacionales
Workflow	La automatización de un proceso de negocio, completo o en parte, durante la cual se transfieren documentos, información y tareas de un participante a otro para que actúen sobre ellos, de acuerdo a un conjunto de reglas procedurales.
Sistema de gestión de workflow	Un sistema que define, crea y gestiona la ejecución de workflows mediante la utilización de software, ejecutado en uno o mas motores de workflow, que es capaz de interpretar la definición del proceso, interactuar con los participantes del workflow y, cuando es necesario, invocar a otros sistemas de información y aplicaciones.
Definición de un proceso	Representación de un proceso de negocio de forma que soporte manipulación automática, como modelado o ejecución por un sistema de gestión de workflow. La definición de un proceso consiste en una red de actividades y sus relaciones, criterios para indicar el inicio y finalización del proceso, e información acerca de las actividades individuales, tales como sus participantes, aplicaciones y datos asociados, etc.
Gestión de procesos de negocios	Técnicas de soporte para procesos de negocio mediante la utilización de métodos, técnicas y software para diseñar, llevar a cabo, controlar y analizar procesos operativos involucrando a personas, organizaciones, aplicaciones, documentos y otras fuentes de información.

Fuente: Workflow Management Coallition. (1999) y Aalst et al. (2003)

2.2 Definición de procesos de negocio.

Una definición de un proceso es, esencialmente, un modelo del mismo inteligible y procesable por aplicaciones *software*. Estos modelos se representan mediante *lenguajes de definición de procesos de negocio*. Desde principios de los 90's se han definido y utilizado una gran cantidad de lenguajes de definición de procesos de negocio. Aunque ha habido intentos para establecer algunos lenguajes como estándar, ninguno de ellos se ha consolidado.

Dependiendo del uso que se le pretenda dar a la definición de un proceso, el modelo debe recoger distintas características del mismo. Esto influye en gran manera en la elección del método de modelado y del lenguaje de definición. Ould (1995) clasifica en tres categorías los motivos por los cuales se modela un proceso de negocio:

- I. Para describirlo: definirlo, comunicarlo, negociarlo, etc. Para comunicar a un empleado como debe realizar su trabajo, y como éste se relaciona con el trabajo de sus compañeros y los objetivos de la empresa.
- II. Para analizarlo: implica un análisis, cuantitativo o cualitativo, de un proceso para, decidir cambios en la planificación de actividades, cambios de asignación de tareas a empleados, incremento o reducción del grado de paralelismo, entre otros.
- III. Para ejecutarlo: a partir de un modelo (o especificación) formal del proceso, conseguir que un sistema *software* (e.g., un WFMS) coordine su ejecución, esto es, gestione automáticamente las tareas necesarias para que los procesos puedan ser llevados a cabo.

2.3 Requisitos de un lenguaje de modelado.

A pesar de que los requisitos exigibles a una técnica de modelado de procesos depende de la utilización que se pretenda dar a las especificaciones, en 1995 Ould enuncia una serie de requisitos deseables en la mayoría de los casos:

- Debe permitir el modelado de situaciones complejas. Los procesos de negocio no se pueden modelar, en general, solo con estructuras jerárquicas (árboles).
- Los modelos no pueden ser ambiguos. Para ello, resulta conveniente que posean una semántica y sintaxis formalmente definidas.
- La notación empleada debe ser fácilmente asimilable por personas. Desde este punto de vista, los lenguajes gráficos resultan más adecuados.

- Debe permitir capturar tanto los aspectos funcionales del proceso como los no funcionales.
- Debe permitir la incorporación fácil de los patrones que mas frecuentemente aparecen en los procesos, tales como planificación, delegación de responsabilidades, contratos, actividades periódicas, etc. Por ello, debe también permitir la reutilización de modelos o partes de modelos.
- Se debe centrar en el proceso, no en los datos.
- Debe proporcionar la posibilidad de ser visualizado desde distintos puntos de vista o niveles de abstracción.

2.4 Metodologías de modelado.

Cada lenguaje de definición de procesos está basado en un modelo. Existen, principalmente, dos metodologías de modelado de procesos (Georgakapoulos, Hornick y Sheth 1995):

- **Las metodologías basadas en comunicación** surgen del trabajo de “*Conversation for Action Model*” (Winograd y Flores, 1987). Estas metodologías asumen que el objetivo de la reingeniería de un proceso es incrementar la satisfacción del cliente. Representa cada acción como un ciclo con cuatro fases de comunicación entre un cliente y un intérprete: preparación, negociación, realización y aceptación. En la fase de preparación el cliente solicita la realización de una acción, o el intérprete se ofrece para realizar una acción. En la fase de negociación ambos acuerdan los términos relativos a la acción y definen los términos de satisfacción. En la fase de realización se lleva a cabo la acción de acuerdo a los términos anteriores. En la fase de aceptación el cliente expresa su satisfacción (o insatisfacción) con la acción realizada. Cada una de las fases de un ciclo se puede unir a otros ciclos, de tal forma que un intérprete de un ciclo puede actuar como cliente de otros. De esta forma se puede ver un proceso de negocio como una red social en la que un grupo de personas, realizando distintos papeles, alcanzan el objetivo del proceso. Georgakapoulos et al. (1995) identifican las principales limitaciones de estas metodologías. Por una parte, dado que están totalmente centradas en la satisfacción del cliente, no resultan apropiadas para situaciones cuyo objetivo de modelado sea otro, como, por ejemplo, minimizar el gasto de material de un proceso de

producción. Por otra parte, las definiciones de procesos no permiten la ejecución automática en WFMSs.

- **Las metodologías basadas en actividad** se centran en el modelado del trabajo. En lugar de modelar compromisos entre personas, se centran en modelar el conjunto de tareas necesarias para llevar a cabo el proceso. Una gran mayoría de los modelos utilizados tanto comercialmente como a nivel de investigación siguen estas metodologías.

2.5 Perspectivas de un proceso de negocio.

Habitualmente, un proceso de negocio está compuesto por la unión de aspectos de distintos ámbitos o perspectivas. Es común identificar las siguientes cinco perspectivas (Jablonski y Bussler, 1996; Verbeek, Basten y Aalst, 2001): control de flujo o proceso, recursos u organización, datos o información, actividades o tareas o funciones y operación o aplicación.

- En la *perspectiva de control de flujo* se define la coordinación de actividades, esto es, qué actividades deben ser ejecutadas y en qué orden. Se suelen utilizar primitivas de ejecución secuencial o en paralelo, bifurcación condicional, unión de ramas alternativas, iteración y sincronización, entre otras.
- En la *perspectiva de recursos* se definen los actores (máquinas o humanos) necesarios para la ejecución del proceso. Los actores humanos, también llamados participantes, suelen estar agrupados según su papel o papeles en la organización.
- En la *perspectiva de datos* se gestionan datos de control y de producción. Los datos de control se introducen para controlar la ejecución del proceso (e.g., para tomar decisiones en puntos de bifurcación condicional). Los datos de producción se corresponden con las entidades del proceso (e.g., documentos, formularios, tablas, etc.)
- En la *perspectiva de actividades* se describen las características de cada actividad del proceso (descripción, duración estimada, plazos, prioridad, clases de actores necesarias, etc.)
- Cada actividad puede estar compuesta por más de una acción elemental. En la *perspectiva de operación* se describen las acciones elementales, que normalmente consisten en la ejecución de una aplicación con unos determinados parámetros.

2.6 Lenguajes de definición de procesos de negocio.

Desde los inicios de la tecnología de *workflow* y procesos de negocio, se han desarrollado una gran cantidad de lenguajes de definición de procesos y, a pesar de algunos intentos, no se ha conseguido establecer ningún lenguaje como estándar comúnmente aceptado. En este apartado se describen brevemente algunos de los más utilizados.

Desde el punto de vista del análisis de procesos de negocio, es importante que el modelo utilizado dé lugar a definiciones de procesos no ambiguas. Para ello, muchos de estos modelos se construyen sobre una semántica formal, que relaciona una definición de un proceso con una estructura matemática.

2.6.1 Modelos basados en redes de Petri.

Las redes de Petri, y algunas de sus extensiones, son formalismos muy utilizados para el modelado de procesos de negocio. Existen diferentes trabajos que describen y analizan en detalle este formalismo, sus aplicaciones y distintas variantes (Peterson, 1981; Murata, 1989; Reisig y Rozemberg, 1998).

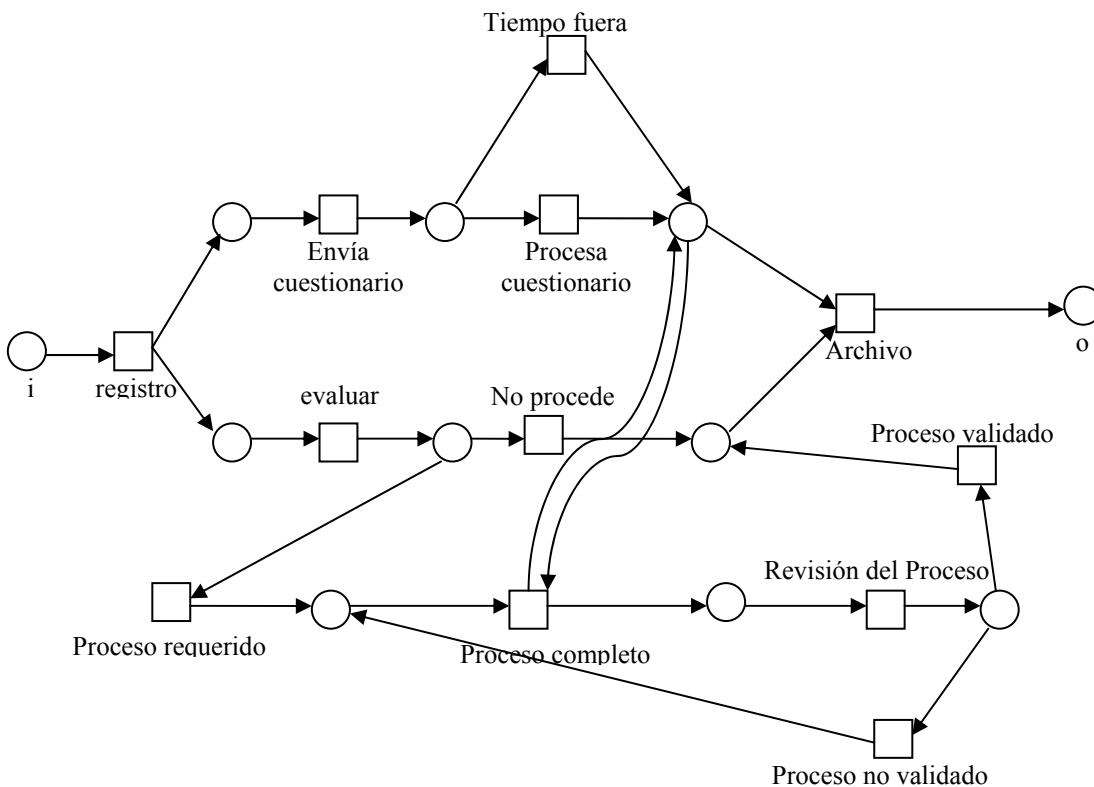
Algunas de las variantes de redes de Petri utilizadas para el modelado de procesos de negocio son: *Workflow Nets* (Aalst, 1998), *Information Control Nets* (Ellis y Nutt, 1993), *INCOME/WF* (Oberweis, Schätzle, Stucky, Weitz, y Zimmermann, 1997), *FunSoft Nets* (Deiters y Gruhn, 1998) y *MILANO WFMS* (Agostini y Michellis, 2000). Salimifard y Wright en 2001 presentan con detalle otras aplicaciones de redes de Petri para el modelado de procesos de negocio. En el trabajo de Aalst (1998) se justifica la utilización de redes de Petri para modelar procesos de negocio, debido a que tienen las siguientes propiedades:

- o **Semántica formal:** las definiciones de procesos son claras y precisas porque tanto las redes de Petri como sus principales extensiones están matemáticamente formalizadas.
- o **Representación gráfica:** son intuitivas y fáciles de leer.
- o **Expresividad:** permite modelar las primitivas mas utilizadas en procesos de negocio.
- o **Propiedades bien conocidas:** se han estudiado en profundidad sus propiedades.

- o **Análisis:** se han desarrollado una gran cantidad de algoritmos de análisis para redes de Petri, que permiten comprobar propiedades de viveza, seguridad, medidas de prestaciones, etc.

Los trabajos más profundos y destacables de aplicación de redes de Petri a procesos de negocio han sido desarrollados por Aalst. Entre mediados y finales de los 90, este autor desarrolla el formalismo *Workflow Nets*, basado en redes de Petri de *alto nivel* (formalismo que añade *color*, *tiempo* y *jerarquía* al modelo clásico de redes de Petri). En la Figura 2.1 se muestra un ejemplo sencillo de un proceso modelado mediante redes de Petri.

Figura 2.1. Proceso de gestión de reclamaciones modelado con redes de Petri.



Fuente: YAWL: Yet Another Workflow Lenguaje.

Veerbeek, Basten y Aalst (2001) usando la herramienta Woflan, proponen el uso de algoritmos de verificación de procesos de negocio implementados como *Workflow Nets*. En trabajos más recientes (Aalst y Hofstede, 2003; Aalst, Hofstede, Kiepuszewski y Barros 2003) se justifica,

mediante un análisis basado en patrones de *workflow*, que los lenguajes basados en redes de Petri modelan bien patrones basados en estado, pero no determinados patrones más complejos (instancias múltiples, sincronizaciones complejas, retrocesos no locales, etc.) Por ello, Aalst y Hofstede (2003) han propuesto un nuevo lenguaje, llamado YAWL (*Yet Another Workflow Language*), diseñado a partir del formalismo de redes de Petri de alto nivel, pero capaz de modelar también dichos patrones complejos.

2.6.2 Modelos basados en UML.

En numerosos trabajos se ha investigado la utilización de UML para el modelado de procesos de negocio. Los diagramas de actividad son el componente de UML más empleado con esta finalidad. Es mas, Eshuis (2002) justifica que el paradigma de orientación a objetos no resulta adecuado para el modelado de procesos de negocio, y que por tanto UML tampoco, exceptuando los diagramas de actividad, que no se basan en este paradigma.

El trabajo mas profundo utilizando diagramas de actividad es la tesis doctoral de Eshuis (2002). En este trabajo se analizan distintas semánticas formales propuestas en trabajos previos para diagramas de actividad, se justifica que ninguna de ellas resulta adecuada para el modelado de procesos de negocio, se proponen dos semánticas formales (Eshuis y Wieringa, 2001a, 2001b) y se presenta una herramienta de verificación funcional, aplicando *model checking* a dichas semánticas (Eshuis y Wieringa, 2002). Dumas y Hofstede (2001) analizan la capacidad de los diagramas de actividad de UML para representar distintos patrones de *workflow*.

2.6.3 Otros lenguajes y formalismos.

Además de los lenguajes basados en redes de Petri y diagramas de actividad de UML, se han desarrollado otros lenguajes y formalismos para el modelado de procesos de negocio. En este apartado se mencionan los más destacados:

- La Workflow Management Coalition en 2002, especifica un lenguaje de intercambio de procesos de *workflow* (XPDL), con el objetivo de ser un lenguaje común que se pueda exportar o importar en cualquier herramienta de *workflow*. Se define como una aplicación

de XML. Forma parte de la interfaz 1 (*intercambio de procesos de negocio*) de la arquitectura de referencia del WfMC (Hollingsworth, 1995).

- Ould en 1995 define un lenguaje grafico, centrado en el modelado de los papeles de los distintos tipos de actores y en la colaboración entre ellos. No se define formalmente su semántica.
- *Event-driven Process Chains* (EPC) forman parte de ARIS (Scheer, 1999). No tienen una semántica formal, aunque están inspiradas en redes de Petri.
- *Architectural Modelling Box for Enterprise Redesign* (AMBER) es un lenguaje de modelado de procesos de negocio (Eertink, Janssen, Luttighuis, Teeuw y Vissers, 1999). En el trabajo de Janssen, Mateescu, Mauw y Springintveld (1998) se define una semántica formal en Promela que permite verificar procesos de negocio con Spin.
- En el proyecto Mentor (Muth, Wodtke, Weissenfels, Weikum, y Kotz-Dittrich, 1998) se utiliza STATEMATE (Harel et al. 1990), un entorno de modelado de sistemas de tiempo real, para modelar procesos de negocio. STATEMATE está basado en *statecharts* (Harel, 1987), un formalismo relativamente semejante a los diagramas de actividad de UML, pero con distinta semántica y capacidad expresiva.
- Recientemente se propone la utilización de modelos basados en π -calculus para el modelado de procesos de negocio (Milner, 1999). Aalst en 2004 defiende justificadamente la utilización de redes de Petri frente a π -calculus.
- BPEL4WS y BPML, dos lenguajes de definición de procesos de negocio con notación XML, orientados a la utilización de servicios Web como mecanismo de comunicación.

2.7 Conclusión.

Hasta ahora existen varios trabajos que han abordado el problema del modelado de procesos de negocio. Cada uno de ellos esta centrado en distintas técnicas y herramientas. Después de realizar

un análisis exhaustivo de los trabajos más relevantes, se han identificado varias carencias en estas propuestas:

- **No se tiene en cuenta la perspectiva de datos:** Todos los trabajos presentados están orientados a la perspectiva de control de flujo. Los principales formalismos empleados hasta ahora, como redes de Petri, statechart o diagramas de actividad no modelan, al menos de forma sencilla, el concepto de variable. Sin embargo, nuevos lenguajes como BPEL4WS y BPML sí modelan la perspectiva de datos, incorporando variables como parte fundamental del lenguaje. Por tanto, aunque la perspectiva de control de flujo es, sin duda, la más importante en un proceso de negocio desde el punto de vista de funcionalidad, es necesario tener en cuenta también la perspectiva de datos.
- **Dificultad para expresar especificaciones:** Desde el punto de vista de la definición de las especificaciones, es decir, el conjunto de propiedades que el diseñador del proceso espera que sean ciertas en el proceso diseñado, identificamos dos carencias fundamentalmente: rigidez para definir especificaciones y necesidad de utilizar el formalismo final en lugar del formalismo de modelado. La rigidez se da en aquellos sistemas que sólo permiten verificar un conjunto de propiedades genéricas, y no permiten que el diseñador defina propiedades específicas para un determinado proceso.
- **Lenguajes de definición y formalismos de verificación específicos:** Los trabajos analizados presentan soluciones al problema de la verificación de procesos de negocio que son específicas para lenguajes de definición de procesos concretos y que emplean formalismos y herramientas de verificación concretos.

Teniendo en cuenta estos hechos, consideramos que las principales ventajas de un entorno genérico, modular y extensible serían las siguientes:

- Una solución genérica tendría un rango de aplicación considerablemente más amplio que una solución ligada a un lenguaje de definición de procesos concreto.
- Dado un lenguaje de definición concreto, sería considerablemente más sencillo integrarlo en un entorno genérico ya existente que desarrollar un entorno completo para el mismo.
- Un entorno genérico, que integre formalismos y herramientas, permite seleccionar en cada momento el que resulte más adecuado para el tipo de proceso o de propiedades a modelar.

Capítulo 3

3.1 Introducción a la arquitectura, especificación y diseño de sistemas orientados a servicios.

Los sistemas de software orientado a servicios se están convirtiendo en el principal paradigma de la ingeniería de software, en este capítulo se analizan los elementos fundamentales de la especificación de requisitos, así como la relación existente entre los requisitos de especificación y la arquitectura de los sistemas orientados a servicios. Esto es necesario para entender que la orientación a servicios no está limitada a servicios Web y a Internet únicamente. Se muestra que existen diferentes variantes de sistemas de software orientados a servicio, con diferentes dominios de aplicación, diferentes propiedades de usuarios, diferentes procesos de desarrollo y diferentes propiedades de ingeniería de software. Se presentan además las ventajas de usar los sistemas orientados a servicios. Se muestran las condiciones que deben de cumplir las interfases orientadas al usuario de servicios, la aplicación de la filosofía par a par, y la combinación de diferentes tecnologías de comunicación entre servicios (inclusive aparentemente obsoletos), y la autonomía de servicios. Estas comisiones implican excelentes propiedades de higiene de software para los sistemas orientados a servicios. Finalmente se muestra, como la orientación a servicios promete abrir el camino del software como un producto de ingeniería

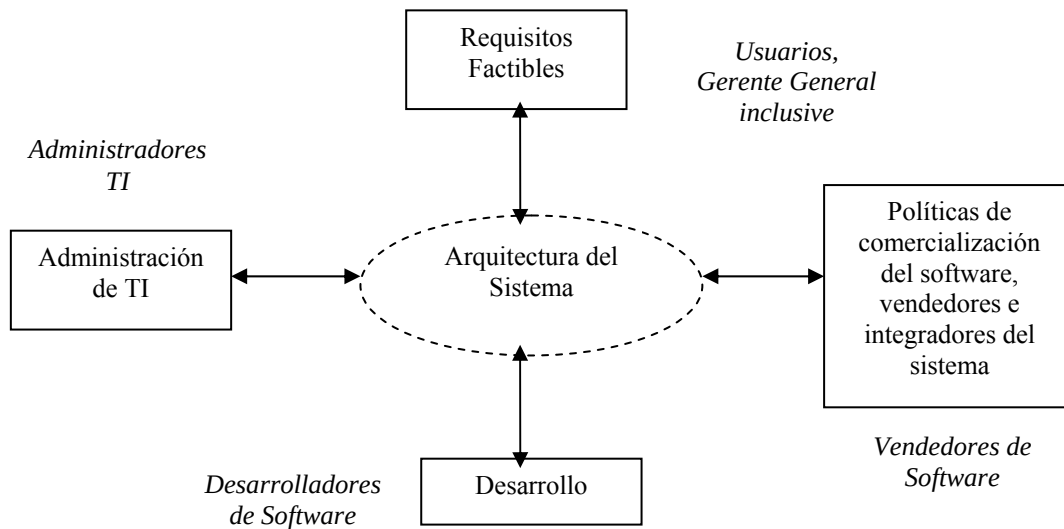
3.2 Antecedentes.

La orientación a servicios (SO) se está convirtiendo en el tema central de la Ingeniería de Software. Hay un crecimiento explosivo en el número de conferencias, productos, artículos y debates para utilizar los principios de la SO y las arquitecturas orientadas a servicios (SOA). Los sistemas de software orientado servicios (SOSS) son de diferente tipo dependiendo de las funciones que el sistema provee, por ejemplo si es para comercio electrónico o para una empresa internacional, y la forma en que el sistema se ha desarrollado. La propiedad común de los SOSS es que sus componentes se comportan como sistemas de servicio masivo en vida real. Los SOSS deben ser redes virtuales par a par (p2p) de componentes autónomos (servicios). Los servicios

pueden tener varias propiedades; necesitan ser necesariamente servicios Web en el sentido W3C (2002) y no necesitan usar protocolos de comunicación estándar como se muestra en Barry y asociados (2003) y Datz (2004).

Se sabe o conoce entonces que las propiedades de la Ingeniería de Software, así como las propiedades de la orientación a usuarios de algunos SOSSS depende fuertemente de las propiedades de las interfases de servicio y que las interfases de usuario del sistema se pueden implementar como un servicio específico (pares en la red). Todas estas cuestiones están relacionadas con la arquitectura de los sistemas. Las propiedades de la arquitectura influyen en el posible conjunto de funciones, en el desarrollo (especialmente en la especificación de requisitos), en las técnicas de desarrollo viable (por ejemplo las ágiles), en las normas y políticas de administración de Tecnologías de la Información (TI), las estrategias de comercialización del software y en la integración del sistema (Ver figura 3.1). Las funciones posibles de un SOSS incluyen las funciones importantes para los usuarios de la alta administración.

Figura 3.1. Rol Central de la Arquitectura del Sistema



Fuente: Elaboración propia.

Las funciones posibles de algunos grandes sistemas dependen de esta arquitectura. La decisión acerca de que arquitectura será usada en muchas ocasiones debe formularse en cada uno de los estados del ciclo de vida del sistema. Por otro lado, la estructura, las técnicas y contenido de la especificación de requisitos, está influenciada por las propiedades de la arquitectura del sistema y

los detalles de su implementación. Se mostrarán que los SOSS pueden usar una combinación de diferentes técnicas de desarrollo durante la historia del software (por ejemplo, paso de mensajes, orientación a objetos, bases de datos comunes, y en algunos casos sistemas orientados a lotes). Todos esos enfoques pueden ser citados en la especificación del SOSS. La SO es un nuevo paradigma para mucha gente que desarrolla software. Esto implica algunos problemas con su aplicación.

3.3 Sistemas de información Par a Par.

Los grandes sistemas de información pueden desarrollarse como una red de componentes de servicios autónomos pobremente acoplados (posiblemente sistemas de información), integrados usando el principio par a par (P2PIS). Los cambios en la arquitectura puede estar acompañados por cambios en la especificación de requisitos, los cuales se reflejan en la estructura de orientación a servicios del sistema.

La especificación de P2PIS comienza con la especificación de la interfaz de usuario del sistema (portal) y con la especificación de los servicios. La especificación de los servicios comienza por la definición de sus interfases. Esta última puede estar acompañada por la especificación de los servicios de la infraestructura (formato de mensajes, protocolos de comunicación, servicios de comunicación en general). Los servicios en P2PIS pueden ser el desarrollo de nuevas aplicaciones, sistemas encapsulados y heredados, o productos de terceras partes. P2PIS habilita un nuevo tipo de requisitos (por ejemplo, los requisitos con los cuales se puedan soportar organizaciones descentralizadas y flexibles de una empresa global, como lo muestra Král & emlika (2003)) y hacer factibles las propiedades de Ingeniería de Software como reusabilidad, flexibilidad, apertura, mantenimiento, el uso de herencia y de productos de tercera parte o la reducción del desarrollo de costos y duración. La experiencia muestra que dichos sistemas pueden ser extremadamente estables (Král, 1995).

Existen dos variantes principales de P2PIS. La primera se usa en comercio electrónico donde el servicio inicia con una comunicación que verifica primero la comunicación entre partes. Las partes pueden ofrecer sus interfases (típicamente especificadas en WSDL). Este esquema implica

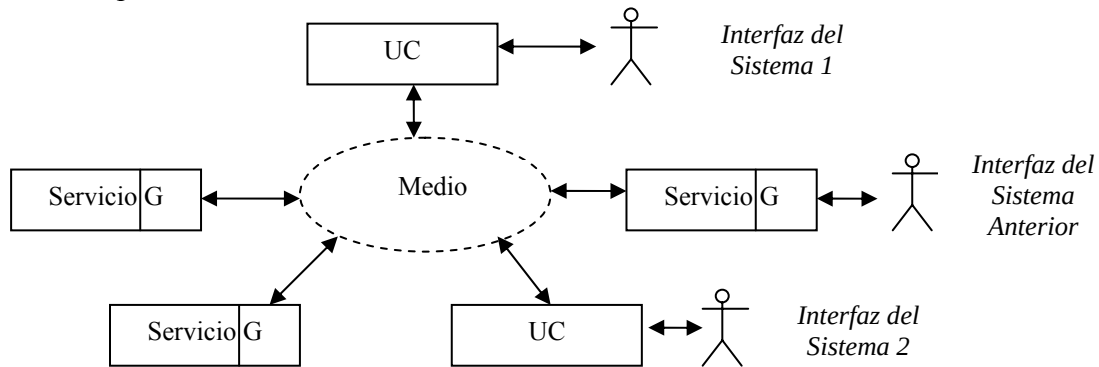
el uso de Internet y estándares internacionales como SOAP. A estos sistemas se les llama alianzas de software (*alianzas*).

Los sistemas formados por un conjunto de servicios estables conocidos, los cuales están en comunicación permanente se les llaman confederaciones de software (*confederaciones*). Las confederaciones ocurren con frecuencia. Por ejemplo:

- Sistemas de información de empresas internacionales que tienen una estructura de red de unidades organizacionales autónomas (divisiones). Los sistemas de información están formados por una red de sistemas de información de las divisiones y por algún componente adicional que sirve de enlace (por ejemplo, portales). Esta arquitectura simplifica la integración de nuevas divisiones así como la creación de nuevas empresas que podrían ser de ventas, el outsourcing de algunas divisiones o dividir las empresas en otras más pequeñas.
- Sistemas de información para servicios electrónicos de gobierno (e-gobierno) los cuales construyen una red de sistemas de información de oficinas particulares (Král & emlika, 2001).
- Múltiples términos de colaboración entre sistemas de información y las empresas, y los sistemas de formación de estas y sus socios de negocio se necesitan para administrar la cadena de suministro (SCM) como lo mencionan Lawson, Kiong & Hunter (1999) y para administrar las relaciones de cliente (CRM) como lo muestra Dyché, (2002).
- Asociaciones abiertas de organizaciones del cuidado de la salud (psicólogos, hospitales, laboratorios, bases de datos de servicios de salud, entre otras) forman un sistema de información que intenta simplificar, expandir y hacer más rápido el cuidado y la atención médica.
- Sistemas de control de proceso (software de tiempo real) soportan por ejemplo el cómputo integrado a la manufactura. Dichos sistemas fueron los primeros que tuvieron las propiedades de sistemas orientados a servicios. Estos probaron en un principio las ventajas de la orientación servicios.

Si un sistema *S* tiene la arquitectura P2P, tendrá la estructura de par/servicios para colaborar. Los servicios pueden estar equipados por compuertas de conexión al entorno (*G*). El sistema *S* muy probablemente está equipado con una interfaz de usuario o portal (*UC*). Pueden existir múltiples portales. Las alianzas no necesitan tener algún portal como se muestra en la figura 3.2.

Figura 3.2 Arquitectura de un Sistema Orientado a Servicios.



Fuente: Elaboración propia

Las propiedades de P2PIS dependen sustancialmente de las propiedades que provean las interfases mediante las compuertas y por las funciones del medio. Las propiedades más importantes de estas interfases es que tanto pueden variar estas. Las interfases estables incrementan la estabilidad de la P2PIS, reducen el costo de mantenimiento y desarrollo, y ocultan los detalles de implementación y la filosofía de los componentes. Esto demuestra que las compuertas no necesitan transformar sus componentes en servicios Web y que el medio en las confederaciones no necesita estar basado en Internet. Por otro lado, los servicios Web y los servicios orientados a Internet que necesitan las alianzas usarán los estándares de red y las herramientas de precondition de comercio electrónico para la comunicación entre socios desconocidos antes de que la comunicación comience. Los SOSS está usualmente contruidos partiendo de la especificación de las interfases y los componentes que proveen los servicios están integrados principalmente como cajas negras.

3.4 Alianzas.

Un tema crucial en el desarrollo de alianzas es la estandarización de los protocolos de comunicación. Desde un punto de vista técnico, el diálogo entre socios (pares) en P2PIS puede ser fácilmente automatizado de manera completa, pero los acuerdos de negocio, y tanto las razones de negocio y legislativas, son usualmente controladas (supervisadas) personalmente por los representantes de comunicación de las partes.

El diálogo es conducido o por una persona de negocios (iniciador) que busca socios de negocios. La persona de negocios aplica su conocimiento acerca del historial de colaboración con sus

socios y tomando como base la situación actual (esto es, la credibilidad) en el mercado evalúa al posible socio. El socio evaluado de manera similar, evalúa a la empresa del iniciador. Los socios tienen que verificar, según lo anterior para que cerrar un contrato. Es necesario supervisar el control sobre las operaciones de negocio. Los documentos de negocios producidos en este sentido representan la confirmación formal de las personas. Se puede tener esto para P2PIS en general, pero es especialmente necesario dentro de las alianzas.

En la práctica esto establece la cooperación en las alianzas mediante los servicios Web en el sentido de W3C. El diálogo entre socios de negocio inicia usando los estándares: Descripción Universal, Descubrimiento e Integración (UDDI) y el Lenguaje de Definición de Servicios Web (WSDL), y continúa con el Protocolo Simple de Acceso a Objetos (SOAP).

La coordinación entre los procesos de negocios de muchos socios puede ser una tarea complicada. La solución óptima para la supervisión de las etapas de los procesos de negocio es la piedra angular de la especificación de requisitos de las alianzas. De esta manera algunas confederaciones consiguen un buen funcionamiento. Los procesos de negocios se desarrollan usando herramientas de desarrollo como .NET o J2EE.

La principal ventaja de las alianzas es su flexibilidad, generalidad, y soluciones estandarizadas. Sus desventajas son los problemas con la eficiencia, estabilidad, el tamaño de los estándares usados, y problemas con la integración de sistemas heredados y productos de tercera parte.

Un problema profundo es que SOAP no es amigable al usuario ya que está basado en llamadas a procedimientos remotos (RPC) lo que cierran el dominio de conocimiento de los programadores (por ejemplo orientación a objetos) y no al dominio del problema de usuario. Esto aumenta los problemas a la especificación de requisitos usando WSDL y UDDI. UDDI es un servicio centralizado. Los servicios centralizados no son buenos en los esquemas P2P. Esto lo confirma la experiencia con sistemas UDDI. SOAP, al igual que los lenguajes orientados a objetos, requieren llamadas a muchos métodos, y por lo tanto, muchos mensajes en cada uno de los diálogos simples. Esto causa problemas de eficiencia, problemas con el prototipado y entendimiento de la comunicación por los humanos.

Los sistemas que usan RPC (por ejemplo, SOAP) se sitúan mejor en la operación de negocios que en el análisis del proceso de negocios, usualmente basados en una fila de datos común.

Las alianzas se sitúan en tareas operativas dentro de los mercados globales. Las decisiones importantes están soportadas por el análisis de mercado y el historial de cooperación entre socios. Esto implica involucramiento de usuarios, análisis masivo de datos, y herramientas específicas para esto. Estas herramientas no están soportadas en el esquema RCP/SOAP.

Una ventaja importante de las alianzas es que los componentes de comunicación de software pueden ser desarrollados con el esquema de desarrollo de servicios Web/SOAP de manera individual como programas de servicio, por ejemplo, terminales.

Esto puede provocar que los casos de negocio (procesos) fallen por alguna razón. En este caso, la razón de la falla y las responsabilidades de la gente puede detectarse mediante el análisis de los mensajes enviados durante el proceso de negocios correspondiente. El análisis puede usarse como evidencia en este sentido. Los mensajes se almacenan en espacio de memoria a los cuales no se tiene acceso por los usuarios, expertos en economía y por abogados los cuales están en el esquema orientado a usuarios. Esto no aclara cuales mensajes en SOAP forman parte de los requisitos. Lo cual indica que la representación de un mensaje y qué tanto se puede leer, pueda estar disponible.

3.5 Confederaciones de Software.

3.5.1 *E-Gobierno: confederación vía integración.*

Algunas confederaciones típicas de software (SWC) están relacionadas con el ejemplo de gobierno electrónico. La maquinaria de e-gobierno -sistemas de información del estado (SIS) es uno de los más grandes sistemas de software que se han construido cualquier país. La lista de los requisitos más importantes de los SIS es:

- Los SIS proporcionan servicios a ciudadanos, empresas oficinas estatales y municipales. Los SIS puede estar disponible para comunicarse con sistemas de información de compañías privadas y ciudadanos. Los SIS tiene un subsistema complejo (portal) que provee la interfaz para los ciudadanos. Tal interfaz debe ser flexible en cuanto a funcionalidad, dependiendo de los perfiles de grupos específicos de ciudadanos y de las

oficinas estatales que los usa. Pueden tener uno o más portales de interfaz provistos por una interfaz integrada definida en una estructura interna del sistema que sea transparente.

- Los SIS pueden soportar la colaboración de la mayoría de las oficinas del estado y municipales. Por ejemplo la colaboración durante la investigación del robo de autos o la verificación de documentos.
- Cada SIS debe reflejar los cambios frecuentes en las leyes y la estructura de la administración estatal.
- Los SIS pueden usar herramientas autónomas ofrecidas por productos de tercera parte para filtrado de datos, minería y análisis. Se pueden agregar nuevas herramientas en el futuro.

Dado que muchos sistemas son usados por oficinas individualmente, esto dificulta la reescritura del sistema en el tiempo. Los sistemas existentes podrán integrarse sin un cambio sustancial en sus funciones. Los sistemas pueden fácilmente integrarse en SIS sin una reconstrucción sustancial, ésta es una razón importante dentro de los requisitos. Ninguna oficina tomará la responsabilidad si algún subsistema no trabaja correctamente. Esto puede reducir la resistencia del usuario y las políticas provocadas por su prehensión sobre la posición y el poder.

Es altamente deseable que los sistemas de información de varias oficinas se encuentren en un nivel local (en una oficina en particular), usándose sin cambios sustanciales. Lo cual implica que las interfases que constituyen estos sistemas de información autónomos tendrán que usar el conocimiento del dominio tanto de la mayoría como en particular. Podemos ver que si éstos ofrecen ventajas sustanciales en ingeniería de software ofrecerán también muchos beneficios para el usuario. Una apropiada especificación y diseño del software de una confederación de software, incrementa el dinamismo de la estructura del sistema y la apertura del mismo.

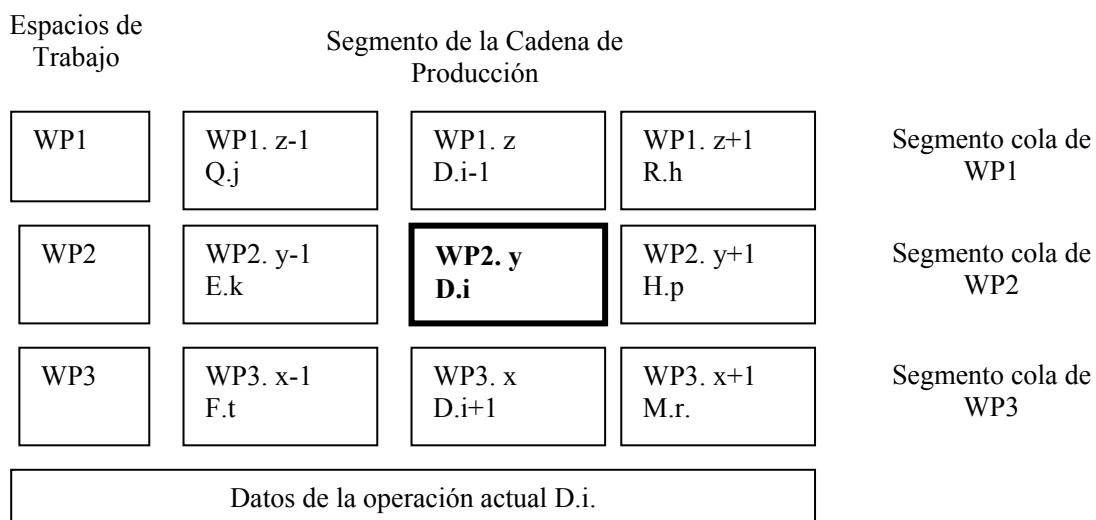
3.5.2 *Sistemas de Manufactura: Descomposición e Integración.*

Los sistemas de apoyo a la administración electrónica (e-gobierno) son sistemas desarrollados principalmente a través de la integración de sistemas actuales, posiblemente equipados con portales apropiados y transformados a fin de que puedan trabajar como servicios (pares en un sistema P2P). Algunos SOSS son, sin embargo, desarrollados desde cero a través descomposición del sistema en servicios. A continuación, los servicios, las interfaces de usuario, y el medio se

desarrollan e integran. Esto es típico del software de sistemas de tiempo real, por ejemplo, en la industria manufacturera. Tales sistemas han demostrado muchas ventajas en la orientación a servicios.

La Figura 3.3 muestra la interfaz del gestor de un sistema flexible de manufactura para la producción de partes de máquinas-herramientas (Král, 1995). La fabricación de las partes se define por secuencias lineales de operaciones de fabricación. Es posible una generalización de los flujos de trabajo más complejas (por ejemplo, el montaje), pero se convierte en la interfaz más compleja. El administrador del taller elige la operación central (actual) *D.i*, y el sistema muestra las operaciones anterior y siguiente las operaciones en la secuencia tecnológica *D* en la cola en el lugar de trabajo. Podemos notar que el administrador considera que la interfaz soporta las actividades estándares con las que se encuentra familiarizado. El administrador puede añadir o modificar la secuencia tecnológica y reorganizar la cola.

Figura 3.3 Interfaz del administrador del sistema de manufactura.



Fuente: Elaboración propia

Los datos requeridos también pueden ser llenados por un planificador de la empresa. Si el programador de datos es correcto, no se requieren acciones por parte del administrador. A este tipo de procesos de negocios se les llama reconstrucción de procesos de negocio (BPR), el primer software. Este debe utilizarse tan a menudo como sea posible.

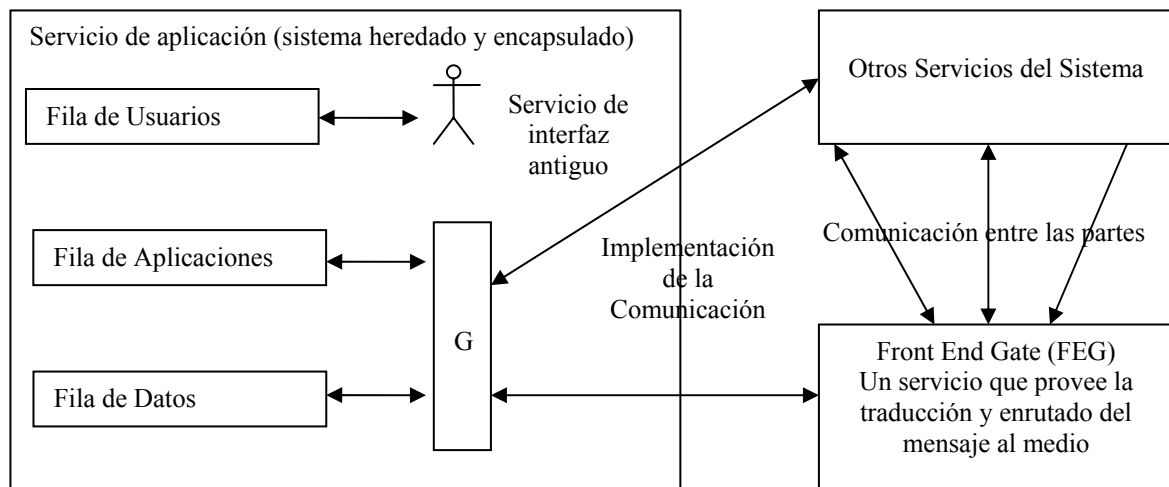
La interfaz del administrador fue orientada a datos y generada a partir de una base de datos. Otras partes del sistema se comunican a través de comandos (por ejemplo, almacenar el producto P en el almacén WP), pueden utilizarse acciones diferentes (no sólo RPC). Es importante que la estructura de los servicios de software refleje la estructura (y las interfaces) de los servicios en la vida real.

Estas propiedades del sistema son las razones por las cuales el sistema de fabricación flexible (FMS), como una isla de automatización, fue utilizado con mucho éxito por más de veinte años en una empresa que tuvo varios sistemas sucesivos de información empresarial. El FMS fue utilizado sin ningún cambio y casi sin ningún tipo de mantenimiento. Esto gracias al hecho de que la interfaz corresponde a la intuición del administrador, el cual utiliza su conocimiento, y se apoya en sus habilidades. Para generalizar, la interfaz de servicio (EAI) debe apoyar la intuición y el conocimiento de los usuarios y debe ser orientada al problema del usuario. El usuario debe tener la oportunidad de influir en el diseño de la interfaz. Las interfaces orientadas al usuario, ofrecen la posibilidad de simular e incluso sustituir los servicios (componentes) que aún no exista mediante la comunicación a través de portales. Decimos entonces que esas interfaces son funcionales al usuario. Esto mejora sustancialmente las herramientas de prototipado.

3.5.3 Mejora en Middleware de las confederaciones.

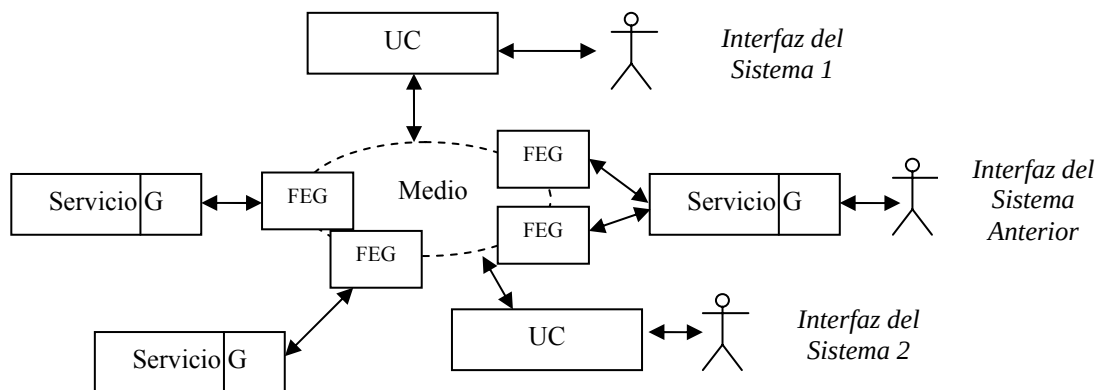
La decisión de utilizar estándares o formatos de mensajes propietarios se debe basar en un correcto análisis (orientado a servicios). Las normas en su forma actual son difíciles de usar. Puede ser razonable para empresas dinámicas que eligen una solución propietaria de formatos de mensajes. No es necesario que sea muy difícil adaptarse a soluciones propietarias para estabilizarse en normas futuras utilizando las herramientas como XSLT y PHP. Usando las herramientas como XSLT y PHP, podemos crear nuevos tipos de servicios llamados portales front-end gate (FEG). FEG es un servicio utilizado como una parte front-end gate de G, de un servicio S (como se muestra en las figura 3.4 y 3.5) o como un router. El FEG transforma los formatos de entrada y salida del mensaje S a una forma aceptable por las partes y S oculta a las propiedades del portal G, para no revelar los detalles de implementación de S. El problema es que, según la experiencia, XSLT es muy ineficaz e inestable actualmente para tareas complejas.

Figura 3.4 Tres servicios interrelacionados con un portal G y sus ligas de comunicación



Fuente: Elaboración propia

Figura 3.5 Sistema con portales front-end.



Fuente: Elaboración propia.

Nuestros dos ejemplos muestran las principales situaciones extremas en el desarrollo de una confederación. En e-gobierno, el sistema está construido principalmente a través de la integración de los sistemas existentes. En este caso, podemos definir sólo las interfaces. En el caso de la industria manufacturera, la etapa de integración es precedida por la de descomposición.

En este caso, deben especificarse los límites de los servicios. Deben ser intuitivamente "similares" a los límites de los servicios del mundo. Un tema específico en las confederaciones es que deben aplicarse también a menudo las actitudes / filosofías conocidas de la historia de desarrollo del software. El Middleware puede ser implementado a través de Internet o a través de

una base de datos común con los factores desencadenantes. CORBA puede utilizarse en algunas situaciones. Todas las actitudes deben ser en la práctica combinadas, dependiendo del tipo de la funcionalidad (por ejemplo, los comandos en el plano operativo y el análisis de datos sobre el nivel de gestión), se pueden aplicar diferentes filosofías y sus combinaciones en las confederaciones. Esto aumenta la complejidad del desarrollo.

3.5.4 Procesos de negocios en las confederaciones.

Los procesos de negocio en las confederaciones (BPC) deben ser compuestos por pasos función/acción de pares ofrecidos por sus interfaces. Se ha visto que los servicios de negocios normalmente deben ser supervisados por parte de los usuarios (los propietarios de los procesos) o debe existir esa posibilidad. También es necesario de ofrecer a los usuarios una herramienta para definir los procesos de negocio. Ambos requisitos implican que los lenguajes entendidos por computadoras deben basarse en el dominio de conocimiento del usuario de lenguajes y nociones. Deben ser por lo tanto orientadas al usuario.

Los datos que definen los procesos de negocio son los mejores para almacenar en la interfase de usuario del sistema (portales) y los pasos de negocio son supervisados a través de un portal o por un servicio de carácter temporal S controlando algún proceso particular. Una solución extrema es que no hay definición de datos de los procesos de negocio. El análisis de los datos de negocio (a través, por ejemplo, de OLAP) debe basarse sobre una perspectiva orientada a datos. En este caso, los usuarios (por lo general, los directores) deben tener un acceso transparente a los datos de cada nivel de los componentes. Este es un problema muy complicado (Lenzerini, 2001).

Las nociones y los lenguajes que especifiquen las acciones de los componentes (por ejemplo, de acuerdo a los libros) en un sentido orientado al usuario son notablemente estables. Lo mismo puede, por lo tanto, ocurrir para los lenguajes de composición de interfaces. Es posible que la interfaz de un componente no tenga que variar si la aplicación del componente cambia. La estabilidad de las interfaces es importante para la comunicación de las partes de un componente dado y para la estabilidad de SOSS. También tiene ventajas sustanciales de ingeniería de software como (modificabilidad oportunidades de contratación externa, la reducción del número de los mensajes, la integración de productos de diversos los proveedores, apertura, entre otros).

3.6 Conclusión.

Las funciones importantes, especialmente aquellas de apoyo a la gerencia general de la empresa, dependen de la orientación de la aplicación de servicio. Esto implica que el sistema debe tener una arquitectura adecuada de red de servicios autónomos par a par orientados a la interfaz de usuario y eventualmente, al desarrollo de nuevas infraestructuras de servicios. Por lo tanto, la filosofía SO influye en la especificación de requisitos sustancialmente, más que otras filosofías. Este punto aún no se entiende lo suficiente, las arquitecturas son muy difíciles de cambiar, por lo tanto se deben elegir al principio de la especificación de requisitos. Por el contrario, los requisitos deben reflejar las características de la arquitectura.

Las interfaces orientadas al usuario de los servicios simplifican la colaboración entre usuarios y desarrolladores, además de que permiten el desarrollo aprovechando la posibilidad de que los servicios se pueden simular mediante la interfaz de usuario de los servicios (portales). Simplifica el diseño de los procesos de negocio y mejora la ingeniería de software de las propiedades del sistema. Se pueden diseñar buenas interfaces fácilmente si los servicios reflejan los servicios de la vida real.

Las interfaces orientadas al usuario pueden servir como especificación de los servicios. Esto puede lograrse, si la colaboración entre usuarios y desarrolladores se ajusta y los desarrolladores puedan comprender los ámbitos de conocimiento del usuario. Esto debe entrenarse. Además permiten el uso de diferentes tecnologías de aplicación en los distintos servicios y agilidad en las formas de desarrollo en proyectos de gran envergadura.

La arquitectura del sistema debidamente elegida influye y a menudo determina los instrumentos y procesos del sistema de desarrollo. Es importante considerar que no hay buenas herramientas para el modelado SOSS. El principal obstáculo es la incapacidad o falta de voluntad de muchos desarrolladores de la aplicación de SO lo que tiene como consecuencia de que cualquier cambio de paradigma implica un proceso a largo plazo. La solución puede estar en la contratación de personas que estén capacitadas en desarrollos orientados a servicios, por ejemplo, desarrolladores de sistemas de software de tiempo real.

De esta manera, los cambios en las tareas de administración de TI deberán de facilitar los acuerdos de detalles de la arquitectura del sistema. La administración de TI puede ser menos dependiente de los proveedores de software, ya tiene una mayor libertad en cuanto a lo que hay que comprar, a quién y en que desarrollo para lograr una ventaja competitiva. Las propiedades buenas de la ingeniería de software SOSS, es que simplifica muchas tareas de administración de TI (externalización selectiva, control de proceso de desarrollo, modificaciones y mantenimiento, etc.). SO requiere cambios en las estrategias de marketing y en los métodos de los proveedores de software e integradores de sistemas.

Los desarrolladores SOSS a menudo deben aplicar datos orientados a objetos y técnicas, e incluso integrar aplicaciones por lote. Además deben de ser capaces de comprender y utilizar los conocimientos de los usuarios de todos los niveles de la jerarquía de la organización. No es fácil, y debe tenerse en cuenta en la formación de los expertos en software, por lo general demasiado cerrados al conocimiento detallado que poseen en el paradigma orientado a la computadora

El paradigma SO es una filosofía que influye en toda la industria de software y en la práctica promete abrir el camino del software de calidad en otras ramas de la industria.

Capítulo 4

4.1 Introducción al modelado Web.

La World Wide Web y el Internet son posiblemente los desarrollos más importantes en la historia de la computación. Estas tecnologías se han convertido en parte integral de la sociedad del siglo XXI. El crecimiento acelerado de la tecnología tiene su origen en el siglo pasado, con los primeros días del software. Eran tiempos en que los programadores a menudo se incorporaban al desarrollo de sistemas en conjunto sin un esquema para hacerlo, en ocasiones para bien y en otras para mal. La actitud prevaleciente al igual que en los inicios de la Ingeniería de Software parecía ser incorporarse al campo de desarrollo si importar los métodos ocupados para hacerlo. Lo anterior conduce a una cuestión clave: es necesario aplicar principios, conceptos y métodos de ingeniería de software al desarrollo Web, pero mediante un enfoque diferente.

En ausencia de un proceso disciplinado dirigido al desarrollar sistemas basados en la Web, existe una creciente preocupación por enfrentar problemas serios en el desarrollo, despliegue y mantenimiento exitoso de dichos sistemas. En esencia la metodología que actualmente se está creando puede conducir a una estrategia enmarañada conforme avanzamos en este sentido. Esto implica una serie de aplicaciones Web mal desarrolladas y con muy altas probabilidades de fracaso. Lo que es peor, conforme los sistemas Web crecen en complejidad, una falla en uno de ellos puede propagar amplios problemas por muchos medios. Cuando esto ocurra, la confianza en Internet será sacudida, lo que provocaría una regulación gubernamental innecesaria y mal concebida, teniendo como consecuencia daños irreparables a tecnologías únicas.

Para evitar una Web enmarañada y tener éxito en el desarrollo y aplicación de sistemas Web complejos y a gran escala, existe una apremiante necesidad de enfoques disciplinarios, nuevos métodos y herramientas con que desarrollar, desplegar y evaluar los sistemas y aplicaciones Web. Tales enfoques y técnicas deben considerar las características especiales de los demás medios, los ambientes y escenarios operativos, y la multiplicidad de perfiles de usuario, lo cual agrega desafíos adicionales al desarrollo de aplicaciones Web.

La ingeniería Web aplica sólidos principios científicos, de ingeniería y administración, y enfoques disciplinados y sistemáticos para el desarrollo, despliegue y mantenimiento exitoso de los sistemas y aplicaciones basadas en la Web de alta calidad (Muruguesan 1999).

4.2 Atributos de los sistemas y aplicaciones basados en la Web.

En un principio los sitios consistían en un conjunto de archivos de hipertexto ligados que presentaban la información mediante texto y gráficos limitados. Conforme el tiempo ha pasado el HTML se incrementó al desarrollar herramientas que permitieron a los ingenieros Web ofrecer capacidad de cálculo junto con información. Nacieron los sistemas y aplicaciones basados en Web. En la actualidad, las aplicaciones Web han evolucionado a sofisticadas herramientas de computación que no sólo proporcionan funciones por sí mismas al usuario final, sino que también se han integrado con bases de datos corporativas y aplicaciones de negocios.

Existe discrepancia en cuanto a por qué las aplicaciones Web son diferentes a muchas otras categorías de software. Powell resume las diferencias principales cuando establece que los sistemas basados en la Web involucran una mezcla entre publicación impresa y desarrollo del software, entre marketing e informática, entre comunicaciones internas y relaciones externas, y entre arte y tecnología (Powell 1998). En la mayoría de las aplicaciones Web se encuentran los siguientes atributos:

- **Intensidad de red.** Una aplicación reside en una red y debe satisfacer las necesidades de una variada comunidad de clientes. Una aplicación puede residir en Internet y permitir una comunicación mundial abierta. Alternativamente una aplicación puede colocarse en una intranet con lo cual los servicios serían únicamente dentro de una organización.
- **Concurrencia.** Un gran número de usuarios pueden tener acceso a la aplicación Web al mismo tiempo. En muchos casos, los patrones de uso entre usuarios finales varían enormemente.
- **Carga impredecible.** El número de usuarios de la aplicación Web puede variar en órdenes de magnitud día con día.
- **Desempeño.** Si un usuario de la aplicación Web tiene que esperar demasiado para ingresar y efectuar el procesamiento de datos del lado del servidor, o para formatear y desplegar los datos del lado del cliente, puede decidir irse a cualquier otra parte.

- **Disponibilidad.** Aunque la expectativa de una disponibilidad total es poco razonable, los usuarios de las aplicaciones más populares demandan acceso sobre una base de 24 horas diarias, los siete días de la semana, los 365 días del año. Los usuarios del otro lado del mundo pueden demandar acceso cuando las aplicaciones de software en América puede estar fuera de línea por mantenimiento.
- **Gobernada por datos.** La función primordial de muchas aplicaciones Web es usar intermediarios para presentar contenidos de texto, gráficos, audio y video al usuario final. Además se utilizan para tener acceso a la información que existe en bases de datos que originalmente no eran parte integral del ambiente basado en la Web, por ejemplo: comercio electrónico o aplicaciones financieras.
- **Sensibilidad al contenido.** La calidad y naturaleza estética del contenido sigue siendo un factor determinante de la calidad de la aplicación Web.
- **Evolución continua.** A diferencia del software convencional, que evoluciona a lo largo de una serie de liberaciones planeadas y espaciadas cronológicamente, las aplicaciones Web evolucionan de manera continua no es raro que algunas aplicaciones Web actualicen su contenido sobre una agenda minuto a minuto o que el contenido sea calculado de manera independiente para cada solicitud.
- **Inmediatez.** Aunque la inmediatez o necesidad apremiante de poner software en el mercado lo más rápido posible, es una característica de muchos dominios de aplicación, las aplicaciones Web con frecuencia muestran un tiempo de comercialización que puede ser cuestión de días o semanas. Los ingenieros Web deben aplicar métodos de planeación, análisis, diseño, implementación y puesta a prueba que han sido adaptados a los cortos tiempos requeridos para el desarrollo de aplicaciones Web.
- **Seguridad.** Dado que las aplicaciones Web están disponibles mediante acceso a la red, es difícil, si no imposible, limitar la población de usuarios finales que pueden tener acceso a la aplicación. Con la finalidad de proteger el contenido confidencial y ofrecer modos seguros para la transmisión de datos, se pueden implementar fuertes medidas de seguridad a lo largo de la infraestructura que sustenta una aplicación Web y dentro de la misma aplicación.
- **Estética.** Una parte innegable de la apariencia de la aplicación Web es su presentación y la disposición de sus elementos. Cuando una aplicación se diseña para comercializar o

vender productos o ideas, la estética puede tener tanto que ver con el éxito como el diseño técnico.

Los atributos anteriores se aplican a todas las aplicaciones Web, pero con diferentes grados de influencia. En el desarrollo de ingeniería Web es posible encontrar las siguientes categorías de aplicaciones:

- o Informativo: se proporcionan contenido de sólo lectura con navegación y enlaces simples.
- o Descarga: un usuario descarga información del servidor apropiado.
- o Personalizable: el usuario personaliza el contenido según sus necesidades específicas.
- o Interacción: la comunicación entre una comunidad de usuarios ocurre por medio de cuartos de charla, tablero de anuncios o mensajería instantánea.
- o Entrada del usuario: la entrada con base en formularios es el principal mecanismo para las necesidades de comunicación.
- o Orientada a transacciones: el usuario hace la solicitud que ejecuta la aplicación Web.
- o Orientada a servicios: la aplicación proporciona un servicio al usuario; por ejemplo lo asesora con el cálculo del pago de una hipoteca.
- o Portal: la aplicación canaliza al usuario hacia otro contenido o servicio Web fuera del dominio del portal de la aplicación.
- o Acceso a una base de datos: el usuario consulta una gran base de datos y extrae información.
- o Almacén de datos: el usuario consulta una colección de grandes bases de datos y extrae información.

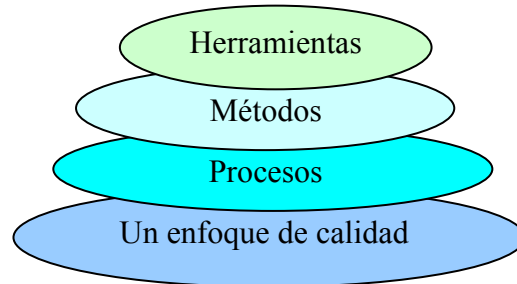
Los atributos y las categorías de aplicación antes mencionadas representan importantes hechos de vida para los ingenieros Web. La clave consiste en desarrollar aplicaciones dentro de las restricciones que imponen dichos atributos y aún así obtener una aplicación Web exitosa.

4.3 Estratos de la ingeniería de aplicaciones Web.

El desarrollo de sistemas y aplicaciones basados en la Web incorpora modelos de procesos especializados, métodos de ingeniería de software adaptados a las características del desarrollo de aplicaciones Web y un conjunto de importantes tecnologías habilitadoras. Los procesos, métodos

y herramientas proporcionan un enfoque de estratos de la ingeniería Web que es conceptualmente idéntico a los estratos de la ingeniería de software que se muestra en la figura 4.1.

Figura 4.1 Estratos de la Ingeniería de Software.



Fuente: Elaboración propia

4.3.1 **Proceso.**

Los modelos de procesos de ingeniería Web adoptan la filosofía del desarrollo ágil. El desarrollo ágil enfatiza un enfoque de desarrollo riguroso que incorpora ciclos de desarrollo rápidos. Aoyama en 1998 describe la motivación para el enfoque ágil en la siguiente forma:

Internet cambió la prioridad principal del desarrollo de software de que a cuánto. El reducido tiempo para el mercado se ha convertido en el límite competitivo por el que luchan las compañías líderes. En consecuencia, reducir el ciclo de desarrollo es ahora una de las misiones más importantes del ingeniero de software.

Aún cuando los ciclos rápidos de tiempo se imponen en el desarrollo, es necesario reconocer que para resolver un problema aún requiere analizarse, desarrollar un diseño, la implementación debe proceder en forma incremental y se debe contar con un enfoque sistematizado de pruebas. Sin embargo, dichas actividades del marco de trabajo se definen dentro de un proceso que:

1. adopte el cambio,
2. aliente la creatividad e independencia del equipo de desarrollo y fortalezca la interacción con los accionistas de la aplicación Web,
3. construya sistemas que utilicen pequeños equipos de desarrollo, y
4. subraye desarrollo evolutivo o incremental mediante el uso de cortos ciclos de desarrollo (McDonald y Welland 2001).

4.3.2 *Métodos.*

El panorama de los métodos de ingeniería Web abarca un conjunto de labores técnicas que permiten al ingeniero Web comprender, caracterizar y luego construir una aplicación Web de alta calidad. Los métodos de ingeniería Web se pueden categorizar de la siguiente manera:

Métodos de comunicación: definen el enfoque con que se facilita la comunicación entre ingenieros Web y los demás participantes de la aplicación Web, como los usuarios finales, clientes de negocio, expertos en problemas de dominio, diseñadores de contenido, líderes de equipo, gestores de proyecto. Las técnicas de comunicación son particularmente importantes durante la recolección de requisitos y siempre que sea evaluado un incremento en la aplicación Web.

Método de análisis de requisitos: proporcionan una base para comprender el contenido que entregará una aplicación Web, la función que proporcionará al usuario final y los modos de interacción que cada clase de usuario requerirá mientras ocurre la navegación por medio de la aplicación Web.

Método de diseño: abarcan la serie de técnicas de diseño que abordan el contenido, la aplicación y la arquitectura de información, así como el diseño de interfaz y estructura de navegación de la aplicación Web.

Métodos de prueba: incorporan revisiones técnicas formales, tanto del contenido y el modelo de diseño como de una amplia variedad de técnicas de prueba que abordan conflictos a nivel de componente y arquitectónicos, prueba de navegación, de facilidad de uso, de seguridad y de configuración.

Cabe señalar que a pesar de que los métodos de ingeniería Web adoptan muchos de los conceptos y principios subyacentes a los métodos de ingeniería de software, los mecanismos de análisis, diseño y prueba deben adaptarse conforme a las características especiales de las aplicaciones Web.

4.4 Herramientas y tecnología.

A lo largo de la década pasada han evolucionado un amplio conjunto de herramientas y tecnología conforme las aplicaciones Web se han vuelto más complejas y extendidas. Dichas tecnologías abarcan un amplio conjunto de descripción de contenido y lenguajes de modelado, por ejemplo HTML, VRLM, XLM entre otros, lenguajes de programación, recursos de desarrollo basados en componentes, por ejemplo CORBA, COM, ActiveX, .NET, navegadores, herramientas multimedia, herramientas de autoría de sitio, herramientas de conectividad de base de datos, herramientas de seguridad, servidores, utilidad de servidor, y herramientas de administración y análisis de sitios.

4.5 El proceso de Ingeniería Web.

Los atributos de los sistemas y aplicaciones basados en la Web tienen una profunda influencia sobre el proceso de ingeniería Web que se elija. Al igual que un ingeniero de software elige un proceso basado en los atributos del software que va a desarrollar, el ingeniero Web elige un proceso basado en la aplicación Web que habrá de construir.

Si la inmediatez y la evolución continua son atributos principales de una aplicación Web, el equipo de ingeniería Web debe elegir un modelo de proceso ágil que produzca liberaciones de la aplicación a un ritmo vertiginoso. Por otra parte si la aplicación Web será desarrollada durante un largo periodo se puede elegir un modelo de proceso incremental.

La intensa naturaleza de las aplicaciones de la red en este dominio sugiere una diversa población de usuarios, que por lo tanto realizan demandas especiales acerca de la respuesta y modelado de requisitos, y una arquitectura de aplicación que puede ser altamente especializada, que en consecuencia realiza demandas al diseño. Dado que con frecuencia las aplicaciones Web son conductoras de contenido, con énfasis en la estética, es probable que se proyecten actividades de desarrollo paralelas dentro del proceso de Ingeniería Web e involucren un equipo de personal tanto técnico como de diseño.

4.5.1 Definición del marco de trabajo.

La efectividad de cualquier proceso de ingeniería depende de su adaptabilidad. Esto es, la organización del equipo de proyecto, los modos de comunicación entre miembros del equipo, las actividades de ingeniería y las tareas que deben realizarse, la información que se recolecte y se cree, y los métodos empleados para elaborar un producto de alta calidad deben adaptarse a la gente que realiza el trabajo, el plazo y las restricciones del proyecto, y el problema que se quiere resolver. Antes de definir un marco de trabajo para el proceso de ingeniería Web se debe reconocer que:

- o Las aplicaciones Web con frecuencia se entregan de manera incremental. Esto es, las actividades del marco de trabajo ocurrirán de manera repetida conforme cada incremento se someta a ingeniería y se entregue.
- o Los cambios ocurrirán frecuentemente. Estos cambios pueden ocurrir como resultado de la evaluación de un incremento entregado o como consecuencia de cambiar las condiciones de negocios.
- o Los plazos son cortos. Esto disminuye la creación y revisión de documentación voluminosa de ingeniería, pero no excluye la simple, que se realiza en el análisis crítico, el diseño y la prueba deben de registrarse de alguna forma.

Además, se deben aplicar los principios para el desarrollo de software ágil de una manera adecuada al desarrollo de la aplicación.

Teniendo en cuenta lo anterior, se aborda el proceso de ingeniería Web, dentro del proceso genérico de marco de trabajo para el desarrollo de software.

Comunicación con el cliente. La comunicación con el cliente se caracteriza por medio de dos grandes tareas: el análisis del negocio y la formulación. El análisis del negocio define el contexto empresarial-organizativo para la aplicación Web. Se identifican los participantes, se predicen los cambios potenciales en el ambiente o los requisitos del negocio, y se define la integración entre la aplicación Web y otras aplicaciones de negocios, bases de datos y funciones. La formulación es una actividad de recopilación de requisitos que involucra a todos los participantes. El intento es

describir el problema que la aplicación Web habrá de resolver aprovechando la información disponible. Además se intenta identificar las áreas de incertidumbre y donde ocurrirán cambios potenciales.

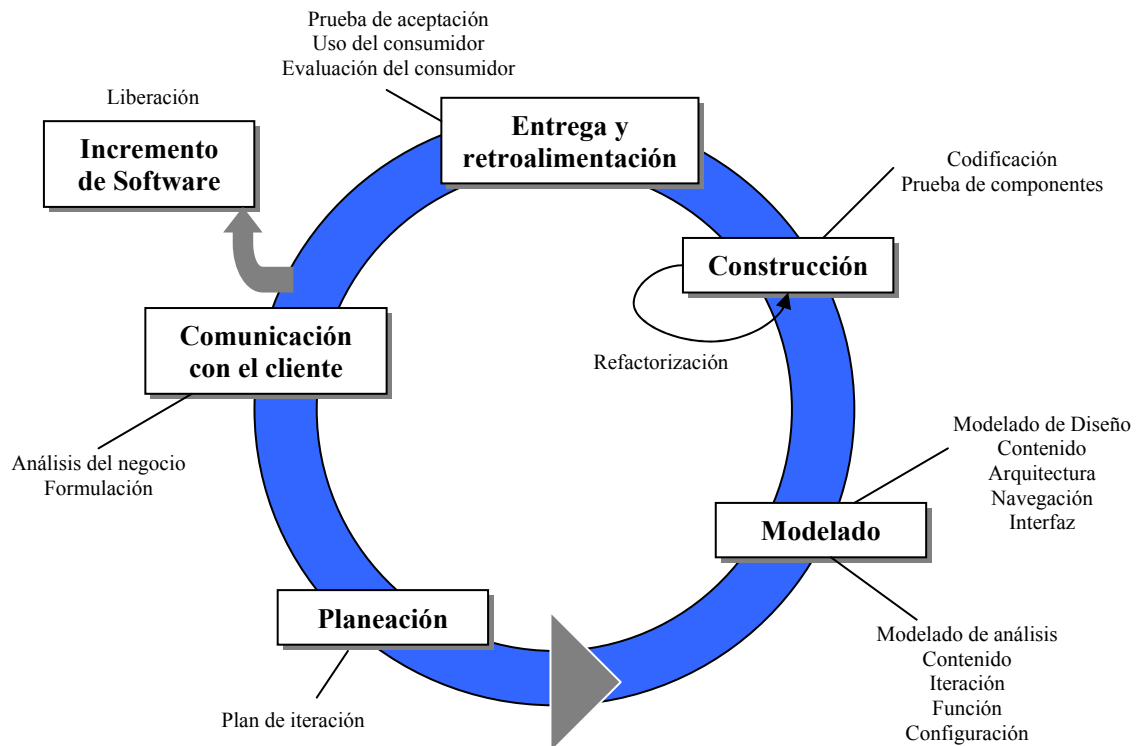
Planeación. Se crea el plan del proyecto para el incremento de la aplicación Web. El plan consiste en la definición de tareas y plazos respecto al periodo proyectado para el desarrollo incremental de la aplicación Web.

Modelado. Las labores convencionales de análisis y diseño de ingeniería de software se adaptan al desarrollo de la aplicación Web, se mezclan y luego se funden en la actividad de modelado para la ingeniería Web. Se pretende desarrollar análisis rápido y modelos de diseño que definan requisitos y al mismo tiempo representen una aplicación Web que los satisfaga.

Construcción. Las herramientas y la tecnología de ingeniería Web se aplican para construir la aplicación que se ha modelado. Una vez que se construye el incremento de la aplicación Web, se dirigen una serie de pruebas rápidas para asegurar que se cubran los errores de diseño, es decir contenido, arquitectura, interfase, navegación. Además de otras pruebas adicionales.

Despliegue. La aplicación Web se configura para su ambiente operativo, se entrega a los usuarios finales y luego comienza un período de evaluación. La retroalimentación acerca de la evaluación se presenta al equipo de ingeniería Web y el incremento se modifica conforme se requiera. Estas cinco actividades del marco de trabajo se aplican empleando el flujo de proceso incremental como se muestra en la siguiente figura 4.2.

Figura 4.2 Proceso de Ingeniería Web.



Fuente: Elaboración propia.

4.5.2 Refinamiento del marco de trabajo.

Debido a que el modelo del proceso de ingeniería Web debe ser adaptable, la definición de las tareas de ingeniería requeridas para refinar cada actividad del marco de trabajo se deja a juicio del equipo de ingeniería Web. En algunos casos, una actividad del marco de trabajo se dirige de manera informal; en otros, se definirá una serie de tareas distintas y las dirigirán miembros del equipo. En todo caso, el equipo es responsable de producir un incremento de la aplicación Web de alta calidad dentro del periodo acordado.

Cabe destacar que las tareas asociadas con las actividades del marco de trabajo de ingeniería Web pueden modificarse, eliminarse o extenderse con base en las características del problema, el producto, el proyecto y la gente que conforma el equipo de ingeniería Web.

4.6 Mejores prácticas en Ingeniería Web.

Si lo que se pretende es construir aplicaciones Web de calidad, se deben aplicar un conjunto fundamental de mejores prácticas, adaptadas de las prácticas clásicas de la ingeniería de software.

Estas son:

1. Tomar tiempo para entender las necesidades del negocio y los objetivos del producto, incluso si los detalles de la aplicación Web son difusos.
2. Describir como interactuarán los usuarios de la aplicación Web usando un enfoque basado en escenarios.
3. Desarrollar un plan de proyecto, incluso si es muy breve.
4. Modelar la aplicación a construir.
5. Revisar la consistencia y calidad de los modelos.
6. Utilizar herramientas y tecnología que permita construir el sistema con tantos componentes reutilizables como sea posible.
7. No apoyarse en usuarios anteriores para depurar la aplicación Web; diseño de pruebas amplias y que se ejecutan antes de liberar el sistema.

Las aplicaciones Web son sistemas de software intensivo que se están convirtiendo más complejos, y están insertando funciones de misión más críticas. Una forma de gestionar la complejidad en los sistemas de software es mediante la abstracción del modelo.

4.7 Un mecanismo formal de extensión a UML.

Teniendo en cuenta que un sistema de software típicamente tiene múltiples modelos, cada uno representa diferentes puntos de vista, niveles de abstracción y detalles, que el propio nivel de abstracción y además los detalles dependen de los artefactos y actividades de trabajo en el desarrollo de procesos, ya que UML es el lenguaje de modelado estándar para sistemas de software intensivos, este se puede igualmente utilizar.

4.7.1 Modelando.

La mayoría de las técnicas de modelado que son practicadas en la actualidad están bien adaptadas al desarrollo de varios modelos de una aplicación Web, y no necesitan mucha discusión. Un

modelo muy importante, sin embargo, es el modelo de análisis y diseño (ADM) el cual presenta algunas dificultades cuando se intentan incluir en las páginas Web, y asociarlo con el código ejecutable y otros elementos en el modelo.

Cuando se decide la forma de modelar algo, se determina el correcto nivel de abstracción y detalle. Cuando se modelan los objetos, éstos representan entidades de la “vida real” que serán contruidos y manipulados al producir el producto final. Modelar las cosas internas de un servidor Web, o los detalles del buscador Web no siempre ayuda a los diseñadores y arquitectos de una aplicación Web. Modelar las páginas, los enlaces de cada una, todos los contenidos dinámicos que vienen dentro de las páginas, y el contenido dinámico de las páginas de cada cliente es muy importante. Esto es lo que un diseñador diseña y que un ejecutor implementa. Páginas, hipervínculos y contenido dinámico en el cliente y en el servidor es lo que se necesita para modelar.

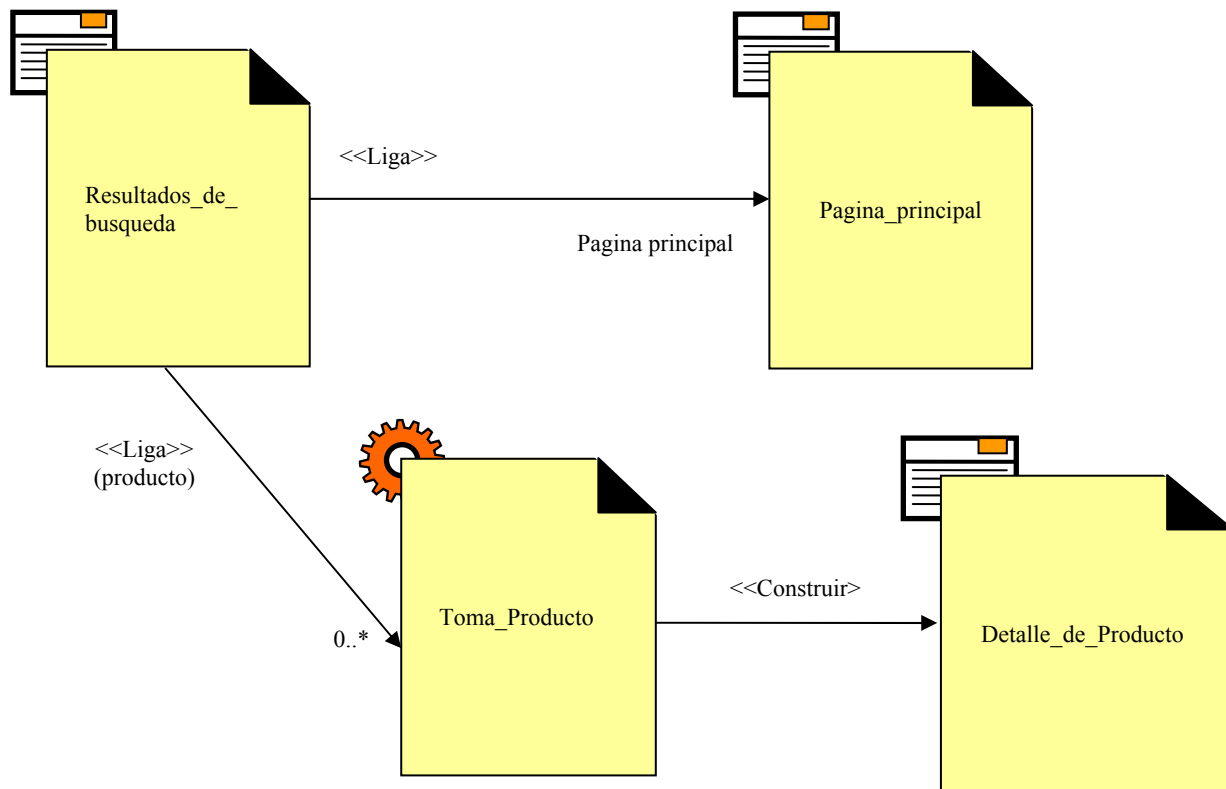
Lo siguiente es mapear los elementos del modelo. Los hipervínculos por ejemplo, naturalmente mapean la asociación de los elementos del modelo. Un hipervínculo representa la ruta de navegabilidad de una página a otra. Extendiendo este punto, las páginas pueden mapear las clases en la vista lógica del modelo. Si una página Web es una clase en el modelo, los scripts de las páginas pueden mapear naturalmente las operaciones de la clase. Cualquier página que tenga variables en los scripts podría mapear los atributos de las clases. Surge un problema cuando se considera que una página Web puede contener un conjunto de scripts que se ejecutan en el servidor (preparando el contenido dinámico de la clase) y un conjunto de scripts completamente diferente que únicamente se ejecuta en el cliente (JavaScript). En este escenario, cuando se ve una clase en una página Web en el modelo, hay confusión sobre que operaciones, atributos e incluso relaciones están activas en el servidor (mientras la página esta siendo iniciada) y cuales están activas cuando el usuario interactúa con la página en el cliente. Adicionalmente, una página Web que es emitida en una aplicación Web se considerara realmente un componente del sistema. Simplemente el mapeo de una página Web como una clase de UML, no nos ayuda a entender mejor el sistema.

Los creadores de UML dijeron que siempre habría situaciones en las que UML, fuera del contexto, no sería suficiente al capturar la semántica de un dominio particular o de la estructura.

Al elaborar este trabajo se consideró un mecanismo de extensión formal, el cual se ha definido para permitir a los profesionales extender la semántica de UML. Este mecanismo permite definir estereotipos, etiquetas de valor y limitaciones que pueden ser aplicadas a los elementos del modelo.

Un estereotipo es un adorno que permite definir un nuevo significado de la semántica para un elemento del modelo (Ver figura 4.3). Las etiquetas de valor son pares de claves de valor que pueden asociarse con un elemento del modelo que nos permite etiquetar algún valor dentro del elemento del modelo. Las limitaciones son reglas que definen la buena forma del modelo. Ellos pueden ser expresados como texto de formas libres o de manera mas forma Lenguaje de Restricciones de Objetos (OCL).

Figura 4.3 Asociación entre clases estereotipadas.



Fuente: Elaboración propia.

Un punto final en el modelado: Se necesita hacer una distinción muy clara entre la lógica y la lógica de la presentación. Para las aplicaciones típicas de negocios, solo la lógica de negocios podría ser parte del modelo de análisis y diseño. Los detalles de la presentación, así como la animación de los botones, la ayuda sobre el manejo y otras mejoras de la interfaz de usuario no pertenecen normalmente al modelo de análisis y diseño. Si una parte separada del modelo de interfaz de usuario es construida por la aplicación, entonces en este lugar se deben representar tales cosas. El modelo de análisis y diseño necesita estar centrando en la expresión de los problemas de los negocios y su solución. En estos días los artistas de la Web, miran si el aspecto de una página Web esta bien diseñada e implementada por los especialistas (técnicamente artistas gráficos) y no por un desarrollador tradicional.

4.7.2 *Arquitectura de aplicaciones Web.*

La arquitectura básica de una aplicación Web incluye, buscadores, una red y un servidor Web. Los buscadores requieren páginas Web del servidor. Cada página es una mezcla de contenido y formato de instrucciones expresadas con HTML. Algunas páginas incluyen scripts del lado del cliente que son interpretados por el buscador. Estos scripts definen la dinámica adicional del comportamiento para la pagina expuesta y a menudo interactúan con el buscador, con el contenido de la página y con los controles adicionales (Applets, controles ActiveX y plug-ins) encontrados en la página. El usuario visualiza e interactúa con el contenido de esta página. Algunas veces el usuario introduce información en los elementos de campos en la página y los envían al servidor para procesarlos. El usuario puede también interactuar con sistemas para navegar en las diferentes páginas del sistema por medio de hipervínculos. En caso contrario, el usuario esta suministrando la entrada al sistema la cual puede alterar el estado del negocio del sistema.

Desde la perspectiva de los clientes, la página Web siempre esta en un documento en formato HTML. En el servidor, sin embargo, una página Web puede manifestarse de diferentes formas. En un principio en las aplicaciones Web, las páginas Web dinámicas eran construidas con la Common Gateway Interface (CGI). La CGI define una interfaz para scripts y compila los módulos para poder acceder a la información que pasó dentro de una página de petición. En un sistema basado en CGI, un directorio especial se configura en el servidor Web al ejecutar los scripts en respuesta a la petición de la página.

Cuando un script CGI es solicitado, el servidor, en lugar de limitarse a responder el contenido del archivo (como podría ser algún archivo en formato HTML), procesa o ejecuta el archivo con el interprete apropiado (usualmente el shell de PERL) y ejecuta la salida de regreso a la solicitud del cliente. El último resultado de este procesamiento es la ejecución de un formato HTML que es enviado de regreso a la solicitud del cliente. La lógica de negocios se ejecuta en el sistema mientras se procesa el archivo. Durante ese tiempo se tiene el potencial para interactuar con los recursos del lado del servidor como las bases de datos y los componentes de nivel medio.

En la actualidad los servidores Web tienen que mejorar este diseño básico. Los servidores Web actuales son mucho más seguros e incluyen mejoras en la gestión del estado del cliente en el servidor, en la integración del proceso de transacción, en la administración remota y en los recursos por mencionar solo algunos. Colectivamente las generaciones pasadas de los servidores Web están dirigiendo estos temas importantes a arquitectos de misiones críticas, escalables y aplicaciones robustas.

Cuando observamos los roles de los script CGI en la actualidad, los servidores Web pueden dividirse en tres categorías principales: páginas de scripts, páginas compiladas y páginas híbridas. En la primera categoría, cada página Web que un buscador cliente puede solicitar se representa en el sistema de archivos del servidor Web como un archivo script. Este archivo es una combinación de HTML y algún otro lenguaje de script. Cuando la página se solicita el servidor Web delega el procesamiento de esta página a un motor que la reconoce, y da como resultado final el formato HTML ejecutable que se envía de regreso a la solicitud del cliente. Ejemplo de esto son Microsoft's Active Server Page y ColdFusion.

En la segunda categoría, las páginas compiladas, el servidor Web carga y ejecuta un componente binario. Este componente, como página script, tiene acceso a toda la información que viene con la página de petición (valores de campos de formas y parámetros). El código compilado usa los detalles de petición, e introduce al servidor los recursos para producir el HTML ejecutable que se regresa al cliente. Aunque no hay reglas, las páginas compiladas tienden a abarcar una gran funcionalidad de las páginas script. Pero pasar los parámetros a una página compilada requiere contar con diferentes funcionalidades. Algunos componentes compilados pueden incluir

actualmente todas las funcionalidades de un directorio completo de páginas script. Las tecnologías que representan este tipo de arquitectura son Microsoft's ISAPI y Netscape's NSAPI.

La tercera categoría representa páginas script que una vez solicitadas son compiladas, y esta versión de compilación se usa posteriormente por todas las peticiones subsecuentes. Solo cuando el contenido de la página original cambia, la página realizaría otra compilación. Esta categoría es una conjunción entre la flexibilidad de las páginas de secuencia de comandos y la eficiencia de las páginas compiladas: JSP es un ejemplo en esta categoría.

4.7.3 *Modelando páginas Web.*

Las páginas Web, ya sean secuencias de comandos o compilaciones, mapean uno a uno los componentes de UML. Un componente es una parte física y reemplazable del sistema. La vista de implementación (Componente de vista) del modelo describe los componentes del sistema y sus relaciones. En una aplicación Web, esta vista describe todas las páginas Web del sistema, y sus relaciones con cada uno de los demás hipervínculos. En cada nivel, un componente del diagrama del sistema Web es como un mapa del sitio.

Dado que los componentes solo representan la paginación física de las interfaces, no son adecuados para modelar la colaboración dentro de las páginas. Este nivel de abstracción, el cual es extremadamente importante para el diseñador y el ejecutor, necesita ser parte del modelo. Al iniciar, podemos decir que cada página Web es un modelo UML en el modelo de vista de diseño (vista lógica), y que las relaciones con otras páginas (asociaciones) representan hipervínculos. Pero estas abstracciones se rompen cuando se considera que cualquier página Web puede representar potencialmente un conjunto de funciones y colaboraciones que existen solo en el servidor y un conjunto completamente diferente que solo existe en el cliente. Las páginas Web de secuencia de comandos del servidor que emplean HTML dinámico (del lado de la secuencia de comandos del cliente) como parte de su salida son ejemplo de tales páginas. La reacción refleja a este problema podría ser estereotipar cada atributo u operación en la clase para indicar si era o no válida en el servidor o en el lado del cliente. Este punto del modelo, que originalmente buscaba simplificar las cosas, se está volviendo más complejo.

Un mejor enfoque al problema es considerar el principio de “separación de preocupaciones”. Lógicamente hablando, el comportamiento de una página Web en el servidor es completamente diferente de lo que es en el cliente. Mientras se ejecuta en el servidor al que tiene acceso a (es decir, las relaciones con) los recursos del lado del servidor (componentes de nivel medio, bases de datos, sistema de archivos). Esa misma página (o la secuencia de salida HTML de la página) en el cliente tiene un comportamiento completamente diferente y un conjunto de relaciones. En el cliente, una página de guión tiene relaciones con el propio navegador (a través del Documento Modelo de objetos o DOM6) y con cualquier Applets Java, controles ActiveX o plug-in de la página específica. Para el diseñador serio no puede haber más relaciones con otras páginas activas en el cliente que las que aparecen en otro marco HTML o navegador.

El aspecto del lado del servidor de una página Web puede ser modelado con una clase y el aspecto del lado del cliente con otra, y distinguir los dos mediante el uso del mecanismo de extensión de UML para definir los estereotipos y los iconos para cada uno: «servidor de la página» y «cliente de la página». Los estereotipos en UML nos permiten definir la nueva semántica de un elemento de modelado. Las clases estereotipadas pueden ser presentadas en un diagrama UML con un icono personalizado, o simplemente adornado con el nombre del estereotipo entre « ». Los iconos son útiles para los diagramas de visión general, utilizar el método sencillo de etiquetas es mejor cuando los atributos de clase y las operaciones están expuestos.

Para las páginas Web, los estereotipos indican que la clase es una abstracción del comportamiento lógico de una página Web ya sea en el cliente o en el servidor. Las dos abstracciones están relacionadas entre sí con una relación direccional entre los dos. Esta asociación es estereotipada: «construir», ya que se puede decir que un servidor de la página se basa en una página cliente. Cada página Web dinámica (páginas cuyo contenido se determina en tiempo de ejecución) se construye con un servidor de la página. Cada cliente de la página es construida por un único servidor de la página (en la mayoría de los casos), sin embargo, es posible que un servidor de la página pueda crear múltiples páginas clientes.

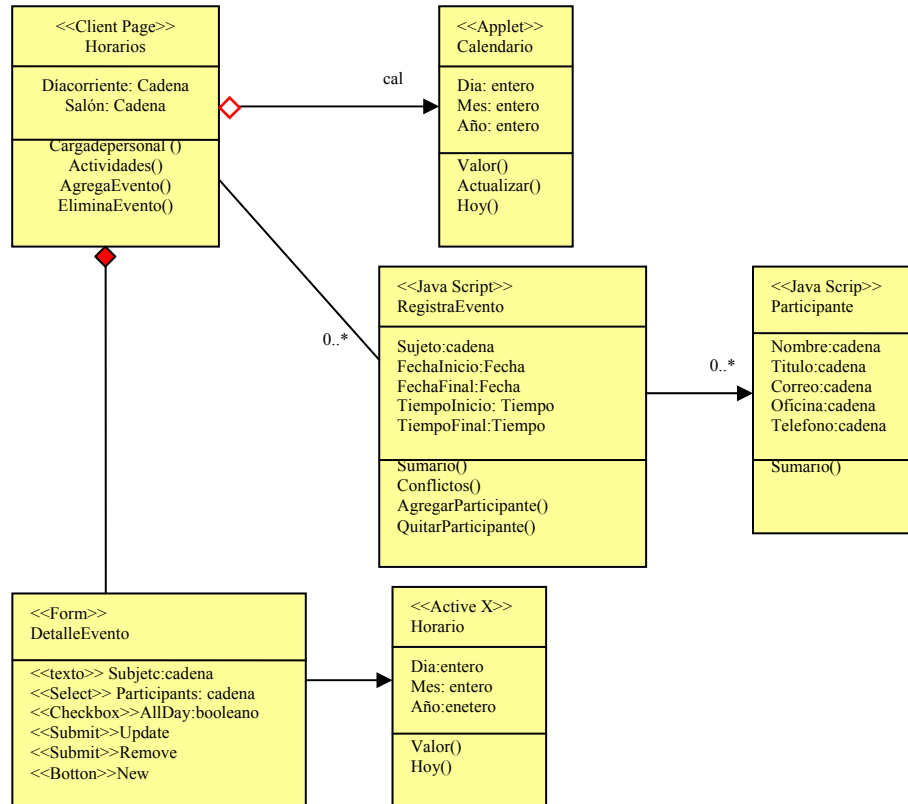
Una relación entre páginas Web es el hipervínculo. Un hipervínculo en una aplicación Web representa una ruta de navegación a través del sistema. Esta relación se expresa en el modelo con

un «vínculo» estereotipo de asociación. Esta asociación siempre se origina en un cliente de la página y los puntos de un cliente o servidor de la página. Los hipervínculos se implementan en el sistema como una solicitud de una página Web, que esta modelada como un componentes en la vista de implementación. Un vínculo de asociación de una página cliente es esencialmente equivalente a un vínculo de asociación con la página del servidor que construye la página cliente. Esto se debe a que un vínculo es en realidad una petición de una página, no propiamente la clase de abstracciones. Desde una página Web un componente comprende la página de abstracciones, un enlace a alguna de las clases realizadas por el componente de la página es equivalente.

Los valores etiquetados se utilizan para definir los parámetros que se pasan junto con una petición de enlace. El «vínculo» de asociación y el valor etiquetado "Parámetros" son una lista de nombres de parámetros (y valores opcionales) que se espera y se utilicen por el servidor de la página que procesa la solicitud. En la Figura 4.3, la página Buscar_resultado contiene un número variable de hipervínculos (0 ...*) a la de Toma_producto de la página del servidor, donde cada eslabón tiene un valor diferente para el parámetro ProductoID. Toma_producto construye el Detalle_de_Producto de la página del producto especificado por el parámetro ProductoID.

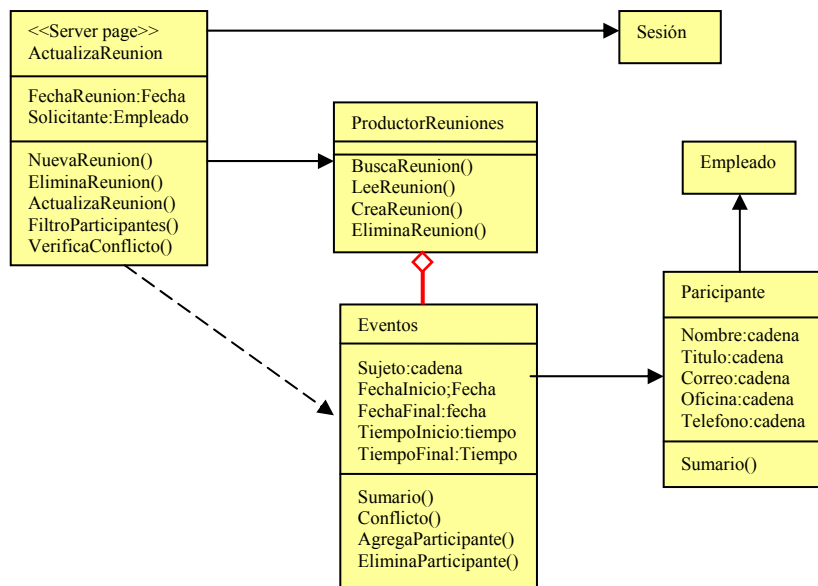
El uso de estos estereotipos hace que sea más fácil un modelo de página de scripts y sus relaciones. El «servidor de la página» de las operaciones de la clase también funcionan en la página de scripts del lado del servidor, y sus atributos alcanzan las variables de la página (accesible a nivel mundial por la página de funciones). El «Client Page» de las operaciones de la clase y los atributos del mismo modo son funciones y variables visibles en el cliente. La ventaja fundamental de separar el servidor y los aspectos del lado del cliente de una página dentro de diferentes clases esta en las relaciones entre páginas y otras clases del sistema. Las páginas cliente son modeladas en las relaciones de los recursos de lado del cliente: DOM, Java applets, controles ActiveX y plug-ins. <<Server Page>> es modelado con las relaciones de los recursos del lado del servidor, componentes de nivel medio, componentes de acceso a bases de datos, sistema operativo del servidor, y así sucesivamente (ver Figuras 4.4 y 4.5).

Figura 4.4 Colaboración del cliente.



Fuente: Elaboración propia

Figura 4.5 Colaboración del servidor



Fuente: Elaboración propia.

El uso de los estereotipos de clase para modelar el comportamiento lógico de páginas Web mediante sus colaboraciones con los componentes del lado del servidor se pueden expresar en mucho de la misma manera que cualquier otra colaboración del lado del servidor. El «Server page» es simplemente otra clase que participa en la lógica de negocios del sistema. En un plano más conceptual, las páginas del servidor suelen tomar el rol de los controladores, es necesario orquestar los objetos de actividad de negocio para cumplir las metas de negocio iniciado por el navegador de la página de solicitud.

En el lado del cliente, las colaboraciones pueden ser un poco complicadas, debido en parte a la variedad de tecnologías que pueden emplearse. Un cliente de la página en su forma más simple es un documento HTML que contiene tanto el contenido y la presentación de información. Podemos hacer navegadores de páginas HTML utilizando el formato de instrucciones en la página, y a veces con hojas de estilo independientes. En la lógica del modelo esta relación se puede expresar con una dependencia de una página cliente o una «Hoja de Estilo» clase estereotipada. Sin embargo, las Hojas de estilo, son principalmente una cuestión de presentación, y son a menudo dejadas de lado por el método de desarrollo de la arquitectura.

4.7.4 Formularios.

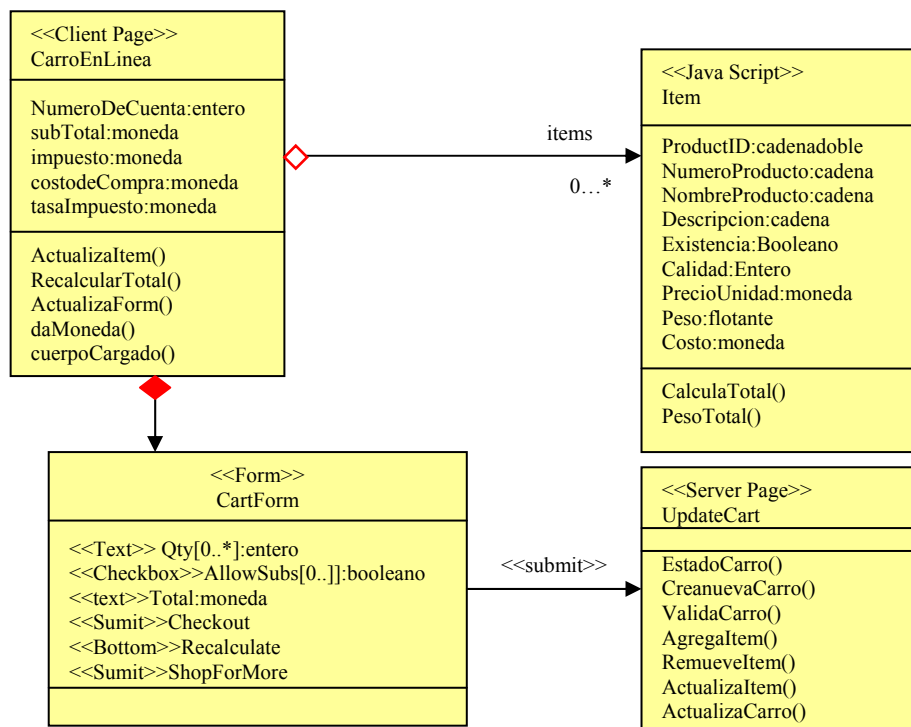
El principal mecanismo de entrada de datos de las páginas Web son los formularios. Los formularios se definen en el documento HTML con etiquetas <form>. Cada formulario especifica la página que va a presentar. Un formulario contiene una serie de elementos de entrada, expresado en todas las etiquetas HTML. Las más comunes son las etiquetas <input>, <select>, y <textarea>. La etiqueta de entrada es un poco sobrecargada, ya que puede ser un campo de texto, casilla de verificación, botón, pulsador, la imagen o un campo oculto, así como algunos otros tipos menos comunes. Modelar formas significa otra clase de estereotipo: «Form». Un «modelo» no tiene operaciones, ya que todas las operaciones que pueden definirse en una etiqueta <form> son realmente propiedad de la página del cliente. Una forma de entrada de los elementos son los atributos estereotipados de la clase «form». Una «form» puede tener relaciones con Applets o controles ActiveX que actúan como controles de entrada. Cada formulario tiene también una relación con una página del servidor del que procesa la forma de presentación. Esta relación es el

estereotipo «submit». Dado que las formas están completamente contenidas en un documento HTML, se expresan en un diagrama UML con una fuerte forma de agregación. La Figura 4.6 muestra un simple carrito de compras que define un formulario de la página, y muestra la presente relación con el procesamiento de servidor de la página.

En la Figura 4.6 el estereotipo «JavaScript» de la clase es un objeto que representa los elementos de la compra. El arreglo de sintaxis se utiliza en la descripción de las propiedades del formulario para los campos que tienen un número variable de las instancias. En el caso de este carro de compras en línea, la compra significa que el carro puede tener de cero a muchos ítems, cada uno con una calidad, existencia, costo, y un costo total <input>.

Dado que toda la actividad de la página cliente se ejecuta con JavaScript, y JavaScript es un tipo de lenguaje, los tipos de datos especificados para cualquiera de estos atributos son utilizados únicamente para la ejecución de la aclaración. Cuando se aplica en JavaScript o como etiquetas HTML de entrada, el tipo se pasa por alto. Esto también se aplica a los parámetros de la función, que aunque no aparecen en esta figura, forman parte del modelo.

Figura 4.6 Forms presentado al servidor de páginas



Fuente: Elaboración propia.

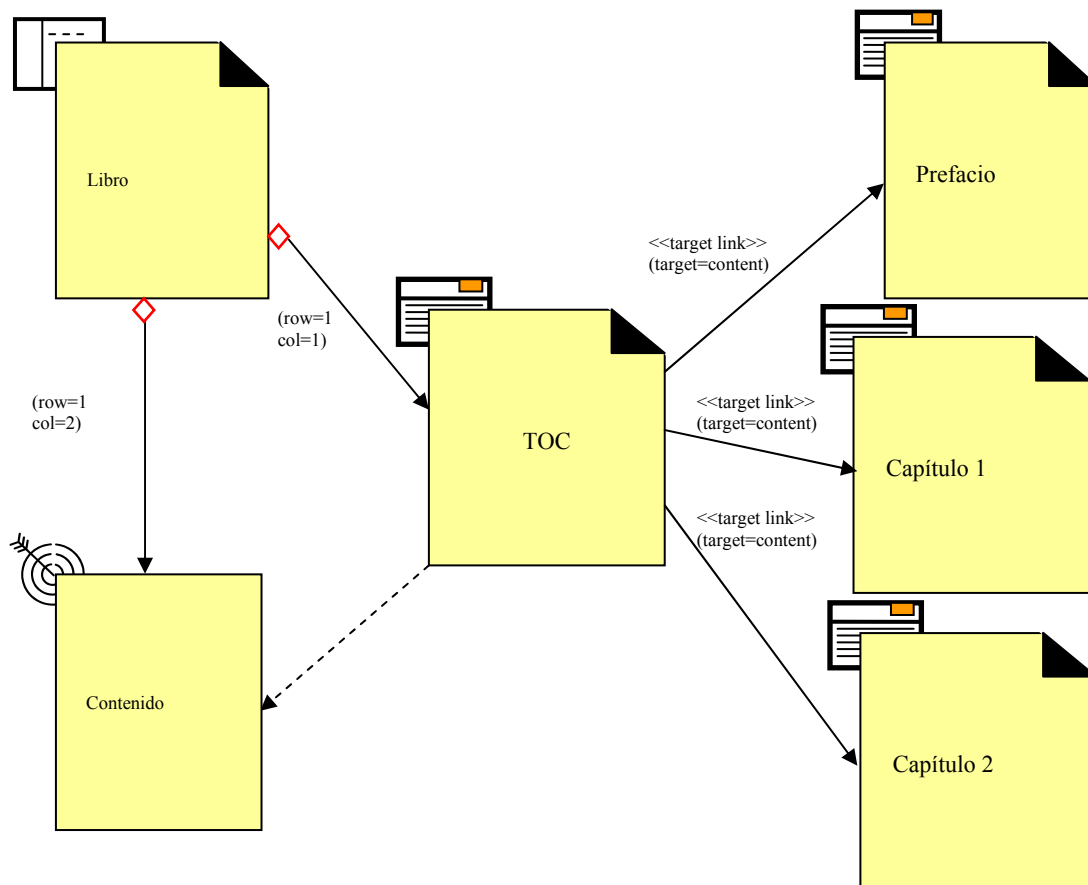
4.7.5 Marcos.

El uso de marcos (frames) HTML en un sitio o aplicación Web ha sido objeto de un gran debate desde su introducción. Los marcos permiten que múltiples páginas se activen y sean visibles para el usuario en cualquier momento. El último conjunto de características de los navegadores comunes hoy en día, permiten también múltiples instancias del mismo, las cuales se activan en el equipo del usuario. La dinámica de uso de los scripts HTML y los componentes en estas páginas pueden interactuar unos con otros. El potencial de interacciones complejas en el cliente es importante, y la necesidad de este modelado aún mayor.

El empleo de marcos o varias instancias del navegador en una aplicación, es una decisión del arquitecto. Si es así, entonces por las mismas razones que se han mencionado anteriormente, el comportamiento del modelo del lado del cliente tiene que estar representado en el modelo de análisis y diseño. Para el marco del modelo definimos el uso de dos más estereotipos de clase: «frameset» y «target», y una asociación estereotipo «targeted link». Un conjunto de marcos clase representa un objeto contenedor, y los mapas directamente a la etiqueta HTML <frameset>. Este contiene páginas de sus clientes y los objetivos. Un objetivo es una clase de nombre de un marco o una instancia del navegador que se referencia por otras páginas de sus clientes. Un enlace de asociación orientada es un enlace a otra página, pero que se dicta en un objetivo específico.

En el ejemplo de la figura 4.7, una vista de esquema se representa en un navegador utilizando dos marcos. Un marco es nombrado como una meta (de contenido), el otro marco sólo contiene la página cliente. Este marco de la página cliente es la tabla de contenido de libro (TOC). Los hipervínculos en esta página están dirigidos, de manera que se hacen en el marco de contenido. El efecto es una tabla estática de contenido en el lado izquierdo y el contenido del libro, capítulo por capítulo, a la derecha de la página.

Figura 4.7 Ejemplo de frames.



Fuente: Elaboración propia.

Muchos de los detalles actuales de presentación son capturados por los valores marcados en el frame y las asociaciones. Dos valores marcados en la relación de agregación entre el frameset y el target de la página cliente especifica las filas y columnas de los frames a los cuales pertenecen al target o página. El valor de la etiqueta target en la orientación del vínculo de asociación identifica el «target» en el que la página podría ser referenciada.

Cuando el objetivo se agrega con un conjunto de frames esto significa que una instancia diferente del navegador se utilizará para hacer las páginas. Es importante tener en cuenta que esta notación es la expresión de una única instancia de una máquina cliente. Los múltiples objetivos independientes son todos los que se supone se ejecutan en la misma máquina, y el diagrama expresa el comportamiento del lado del cliente de una instancia del mismo. Cualquier otra

configuración de despliegue tendría que ser muy documentada en el modelo para una mejor comprensión.

4.8 Metodología WAE para el desarrollo de las aplicaciones WEB con UML.

A continuación se describe la extensión propuesta por Jim Conallen para diseñar aplicaciones Web.

4.9 Estereotipos.

Tabla 4-1 Elementos de la página del servidor

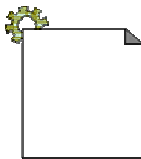
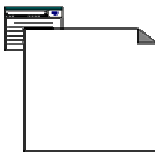
Nombre	Página del servidor		
Meta-modelo de clase	Clase		
Descripción	Una página del servidor representa una página Web que tiene scripts que son ejecutados por el servidor. Estos scripts actúan recíprocamente con recursos en el servidor (bancos de datos, lógica de negocio, sistemas externos, etc). Los funcionamientos del objeto representan las funciones en el script, y sus atributos representan las variables que son visibles en el alcance de la página (accesible por todas las funciones en la página).		
Restricciones	Las páginas del servidor pueden tener sólo relaciones con objetos en el servidor.		
Valores etiquetados	Artefacto de Scripting - O el lenguaje o artefacto que debe ser usado para ejecutar o interpretar esta página (JavaScript, VBScript, Perl, etc.)		

Tabla 4-2. Elementos de la página del cliente

Nombre	Página del cliente		
Meta-modelo de clase	Clase		
Descripción	Un caso de una página del cliente es una estructura HTML de la página Web. Como cualquier página HTML es una mezcla de datos, presentación y lógica. Las páginas del cliente son dadas por browsers del cliente, y puede contener scripts que son interpretados por el browser. Las páginas del cliente pueden tener asociaciones con otras páginas del cliente o del servidor.		
Restricciones	Ninguna		

Valores etiquetados	TitleTag – El título de la página desplegado por el browser. BaseTag – El URL de la base para no referenciar el URL relativo. BodyTag – El juego de atributos para la etiqueta del <body> que pone el texto de fondo y valor por defecto.
---------------------	--

Tabla 4-3. Elementos del Formulario

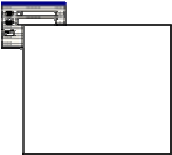
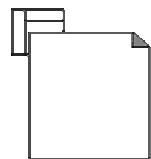
Nombre	Formulario (Form)		
Meta-modelo de clase	Clase		
Descripción	Una clase estereotipada como un «form» es una colección de campos de entrada que son parte de una página del cliente. Una clase del formulario se mapea directamente a HTML. Sus atributos representan los campos de la entrada del formulario de HTML (input boxes, text areas, radio buttons, check boxes, y los campos ocultos). Un «form» se opera, desde que no puede encapsularse su funcionamiento en un formulario. Cualquier funcionamiento que actúa recíprocamente con el formulario sería la propiedad de la página que contiene al formulario.		
Restricciones	Ninguna		
Valores etiquetados	Método - el método suministra datos a la acción URL, cualquiera GET o POST.		

Tabla 4-4 Elementos del Frame Set

Nombre	Frame Set	
Meta-modelo de clase	Clase	
Descripción	Un juego de frames es un contenedor de múltiples páginas Web. La vista de los rectángulos son áreas divididas en rectángulos más pequeños de frames. Cada frame puede ser asociado mediante un solo nombre «target», aunque no necesariamente. Los contenidos de un frame pueden estar en una página Web ó en otro juego de frames. Un estereotipado de la clase de Frame se mapea directamente a una página Web, y el estereotipado HTML enmarca la etiqueta. Un frameset es una «página del cliente», que puede tener funcionamiento y atributos también, pero éstos sólo son activados por browsers que no devuelve frames.	
Restricciones	Ninguna	

Valores etiquetados	Filas (Rows) - El valor del atributo de las filas es la etiqueta <frameset> de HTML. Esto es una secuencia con los anchos delimitados de la fila. Columnas (Cols) - El valor del atributo del las Columnas (Cols) es la etiqueta <frameset> de HTML. Esto es una secuencia con anchuras de columna delimitadas.
---------------------	---

Tabla 4-5 Elementos de Target

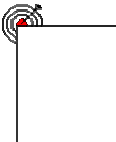
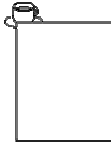
Nombre	Target	
Meta-modelo de clase	Clase	
Descripción	Típicamente un target es un marco en una ventana definida por un frameset, sin embargo un target podría ser un complemento de un nuevo caso del browser o de la ventana. «Targeted link» las asociaciones especifican targets como el lugar donde una nueva página Web será devuelta.	
Restricciones	El nombre de un target debe ser único para cada cliente del sistema. Esto significa, que sólo un caso de un target puede existir en el mismo cliente.	
Valores etiquetados	Ninguno	

Tabla 4-6. Elementos de JavaScript

Nombre	JavaScript	
Meta-modelo de clase	Clase	
Descripción	En un Javascript permitido por el browser es posible simular en un usuario los objetos definidos con funciones del Javascript. Los casos del «JavaScript» existen solamente en el contexto de las páginas del cliente.	
Restricción	Ninguna	
Valores etiquetados	Ninguno	

Link: un link es un indicador de una página del cliente a otra «Página». En un diagrama de clase, un link es una asociación entre una «página del cliente» y cualquier otra «página del cliente» o a una «página del servidor».

Target link: similar a una «link» asociación, un «targeted link» es una link donde la página asociada se da en otro target. Esta asociación traza directamente a HTML, con el target especificado por el atributo del target de la etiqueta.

Submit: un «submit» es siempre la asociación entre una «form» y una «página del servidor». Los Forms envían sus valores del campo al servidor a través de «pagina del servidor» para procesarse. El servidor de la Web procesa la «página del servidor» que acepta y usa la información en la Form enviada.

Está relación indica que pagina (o páginas) puede procesar la Form, y qué Forms en una «página del servidor» tienen poco conocimiento sobre.

Builds: el «builds» es una relación especial que conecta el vació entre el cliente y las páginas del servidor. Las páginas del servidor sólo existen en el servidor.

La asociación «build» identifica qué página del servidor es responsable en la creación de una página del cliente. Ésta es una relación direccional, desde que la página del cliente no contiene conocimiento de cómo esta existe.

Una página del servidor puede construir múltiples páginas del cliente, pero una página del cliente sólo puede ser construida a través de una página del servidor.

Redirect: un «redirect» es una asociación unidireccional con otra página Web. Puede dirigirse a ambos, el cliente y las páginas del servidor. Si la relación origina de una «página del servidor» entonces, indica que el proceso de la demanda de la página puede continuar con la otra página. Esto siempre indica que esa página del destino participa en la construcción de una página del cliente. Esta relación particular no es completamente estructural, desde que la invocación real del funcionamiento del redireccionamiento debe hacerse programáticamente en el código de la página originado.

Si la relación origina una «página del cliente» entonces, esto indica que la página del destino será pedida automáticamente por el browser, sin la entrada del usuario. Un valor de retraso en el tiempo puede decirse que especifica un retraso (en segundos) antes de que la segunda página se solicite. Este uso de redireccionamiento corresponde a la etiqueta de META y HTTP-EQUIV valor de "Refresh".

4.9.1 *Servidor Scripting.*

Es importante notar que la conexión entre el cliente y el servidor sólo existe durante una petición de la página. Una vez que la petición se cumple, la conexión de esta se rompe. Toda actividad en el servidor ocurre durante la petición de la página. Esto representa una distinción muy significativa entre las aplicaciones de servidor y las tradicionales del cliente. La lógica comercial en el servidor sólo es activada por la ejecución de scripts dentro de las páginas solicitadas por el navegador.

Dependiendo del tipo específico de scripts, las páginas escritas pueden contener variables definidas por el usuario, sub-rutinas y funciones. En algunos casos, incluso permiten la definición e interacción de objetos.

4.9.2 *Cliente Scripting.*

El servidor no es el único componente en una aplicación Web que ejecuta scripts. El propio navegador puede ejecutar código escrito en una página. Cuando el browser ejecuta un script, sin embargo, no tiene acceso directo a los recursos del servidor. Típicamente los scripts que se ejecutan en el cliente, aumentan la interacción con la interfaz del usuario ya que definen y llevan a cabo la esencia de la lógica comercial.

4.10 *Conclusión.*

Las ideas y conceptos descritos en este capítulo son una introducción a las cuestiones y soluciones para el modelado de aplicaciones Web con elementos específicos de UML de acuerdo a la propuesta hecha por Conallen en 2002. El objetivo de este trabajo es presentar una forma coherente y completa de integración de los elementos específicos modelados en la Web con el resto de la aplicación modelada, de tal manera que el nivel de detalle y abstracción sea apropiado para los diseñadores, ejecutores, y arquitectos de aplicaciones Web. La extensión de UML para

aplicaciones Web proporciona una forma común a los arquitectos y diseñadores, para expresar la totalidad de sus aplicaciones Web mediante el diseño con UML.

Las aplicaciones Web son sistemas de software complejos que implican al menos una dimensión adicional respecto a los sistemas tradicionales, la navegación a través de hipervínculos y el envío de formularios. La comprensión de estas aplicaciones es una tarea difícil y el nivel de diseño basado en UML proporciona poco apoyo. La experiencia muestra que los diagramas de especificación para la Web con estereotipos, como la propuesta de Conallen, provee una contribución sustancial y significativa a la comprensión de las actividades. La ingeniería inversa para convertir los diagramas de Conallen en código requiere cierto esfuerzo, selección dedicada de las vistas apropiadas, y de la información que se mostrará o se ocultará. Sin embargo, los resultados indican que el esfuerzo tiene un efecto positivo significativo en las sucesivas actividades de comprensión.

Como ocurre con los estudios empíricos, la replicación en diferentes contextos, con diferentes sujetos y objetos, es la única manera de corroborar lo que se encontró al desarrollar el presente proyecto de investigación. Sería interesante considerar diferentes alternativas experimentales haciendo ajustes en varios aspectos, pero quizás la más importante es el perfil de los sujetos implicados. Es muy importante comprender cómo diferentes subgrupos de población de programadores hacen uso de la información proporcionada por los diagramas de Conallen. Se espera que los programadores novatos se comporten de forma muy diferente de los programadores expertos.

Capítulo 5

5.1 Introducción a la propuesta metodológica.

Actualmente, la mayor parte de los modelos de proceso que se desarrollan con UML son guiados por los casos de uso. Un caso de uso se puede definir como una secuencia de acciones, que incluyen variaciones que el sistema puede ejecutar y que produce un resultado observable de valor para un actor que interactúa con el sistema (Booch, Rumbaugh, y Jacobson, 1999). Aunque el éxito de los casos de uso se suele justificar con el hecho de que constituyen una técnica simple e intuitiva, algunos autores (ver por ejemplo Cockburn 1997, Korson 1998, Korson, 1999) señalan las dificultades que existen para la obtención y la especificación de casos de uso verdaderamente útiles, y la falta de consenso sobre cómo organizarlos y manejarlos.

En 1998 Korson defiende que los requisitos (y por tanto los casos de uso) se deben de organizar jerárquicamente, además establece que cada nivel de casos de uso no debe añadir nuevos requisitos, sino refinar los del nivel superior y que la jerarquía de los casos de uso no debe ser el resultado de una descomposición funcional, y debe de ser desarrollada de manera iterativa e incremental.

Por otro lado, Cockburn en 1997 utiliza el concepto de objetivo (goal) para organizar jerárquicamente los casos de uso. Distingue básicamente entre objetivos estratégicos (los procesos de negocios de la organización) y objetivos del usuario (las funciones del sistema). Los objetivos estratégicos corresponden a un conjunto de objetivos del usuario y, de igual modo, un objetivo del usuario puede ser descompuesto, a su vez, en un conjunto de objetivos del usuario.

Otra cuestión importante es la ubicación del modelado de casos de uso dentro del modelo del proceso. Normalmente se concibe el modelado de casos de uso como un paso previo al modelado conceptual. Sin embargo, Korson en 1999 argumenta que no es posible crear casos de uso adecuados y útiles (ni implementarlos correctamente) sin comprender el dominio, y por lo tanto,

el modelado de casos de uso y el modelado conceptual deben ser actividades realizadas en paralelo.

Por esta razón se considera elaborar una guía que permita la organización de los casos de uso, la cual abarque desde el modelado de negocios, hasta el modelado Web.

5.2 Propuesta de modelado.

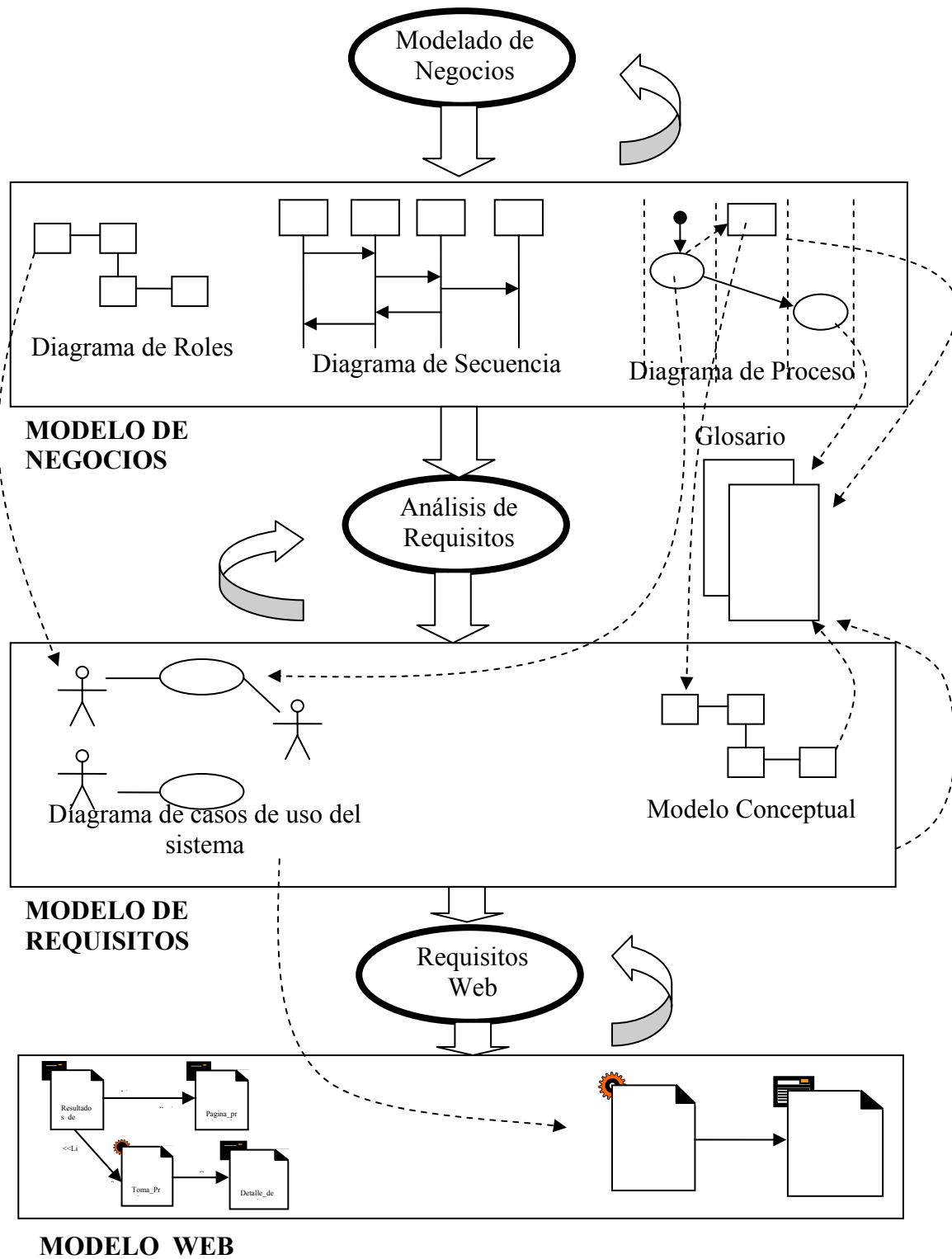
Normalmente, los casos de uso son elicitados de forma intuitiva a partir de la especificación del sistema y, posteriormente, las entidades del modelo conceptual se extraen a partir de las especificaciones de los casos de uso. En las siguientes secciones, se presenta una propuesta para obtener de manera sistemática el modelo de casos de uso y el modelo conceptual, a partir del modelo de negocios y como realizar el modelado Web a partir del modelado del sistema, como se muestra en la Figura 5.1.

Tomando al modelo de negocios como base para la especificación de los requisitos más importantes del sistema que dará soporte al negocio, siendo por lo tanto el propio negocio lo que determine los requisitos.

Una vez identificados los procesos de negocio de la organización y descritos sus flujos de trabajo mediante diagramas de actividades, los casos de uso se elicitán y estructuran a partir de las actividades de cada proceso, mientras que las entidades del modelo conceptual se obtienen de los datos que fluyen entre tales actividades. Además, se identifican las reglas de negocios y se incluyen en un glosario como parte de la especificación de los datos y las actividades.

En la presente propuesta, el modelado de casos de uso y el modelado conceptual se realizan al mismo tiempo, facilitando, por lo tanto, la identificación y especificación de los casos de uso adecuados.

Figura 5.1 Diagrama de trazabilidad entre el modelo de negocios, de lo requisitos del sistema y la Web.



Fuente: Elaboración propia.

5.3 Modelado de Negocios.

Para conseguir sus objetivos, una empresa organiza su actividad por medio de un conjunto de procesos de negocio. Cada uno de ellos se caracteriza por una colección de datos que son producidos y manipulados mediante un conjunto de tareas, en las que ciertos agentes (por ejemplo, trabajadores o departamentos) participan de acuerdo a un flujo de trabajo determinado. Además, estos procesos están sujetos a un conjunto de reglas de negocio, que determinan la estructura de la información y las políticas de la empresa. Por lo tanto, la finalidad del modelado de negocios es describir cada proceso del negocio, especificando sus datos, actividades o tareas, roles o agentes y sus reglas de negocio.

5.3.1 *Identificar los procesos de negocio.*

El primer paso del modelado de negocios consiste en capturar los procesos del negocio de la organización bajo estudio. La definición del conjunto de procesos de negocio es fundamental, ya que determina a posteriori los límites del proceso de modelado.

Para identificar de manera adecuada los diferentes procesos de negocio de una organización a partir de la determinación y estructuración de sus objetivos, se toma el concepto de objetivo estratégico introducido por Cockburn en 1997. En primer lugar, identificamos los objetivos estratégicos de la empresa. Teniendo en cuenta que estos objetivos van a ser muy complejos y de un nivel de abstracción muy alto, cada uno de ellos puede ser descompuesto en un conjunto de subobjetivos más concretos, que deberán cumplirse para conseguir el objetivo estratégico original. Estos subobjetivos pueden a su vez ser descompuestos en otros, de manera que se defina una jerarquía de objetivos. Con una descomposición de dos o tres niveles suele ser suficiente.

Para ilustrar la propuesta se tomó el caso de un hospital que pretende brindar el servicio de consulta y cuidados médicos de manera remota apoyados por un sistema (Teleservices and Remote Medical Care System - TRMCS).

El objetivo estratégico es que el sistema TRMCS proporcionará ayuda médica, a los pacientes en su domicilio o a pacientes móviles que estén conectados al sistema mediante algún dispositivo de comunicación, este objetivo es el proceso general del sistema TRMCS.

Este objetivo se puede dividir en tres objetivos particulares:

Objetivo 1: Atender las llamadas de solicitud de servicio, emitidas por algún paciente para llevar a cabo una consulta remota.

Objetivo 2: Generar automáticamente una llamada de auxilio cuando sea necesario según el monitoreo continuo de la condición de salud de los pacientes.

Objetivo 3: Incorporar a nuevos clientes al sistema mediante su respectiva suscripción.

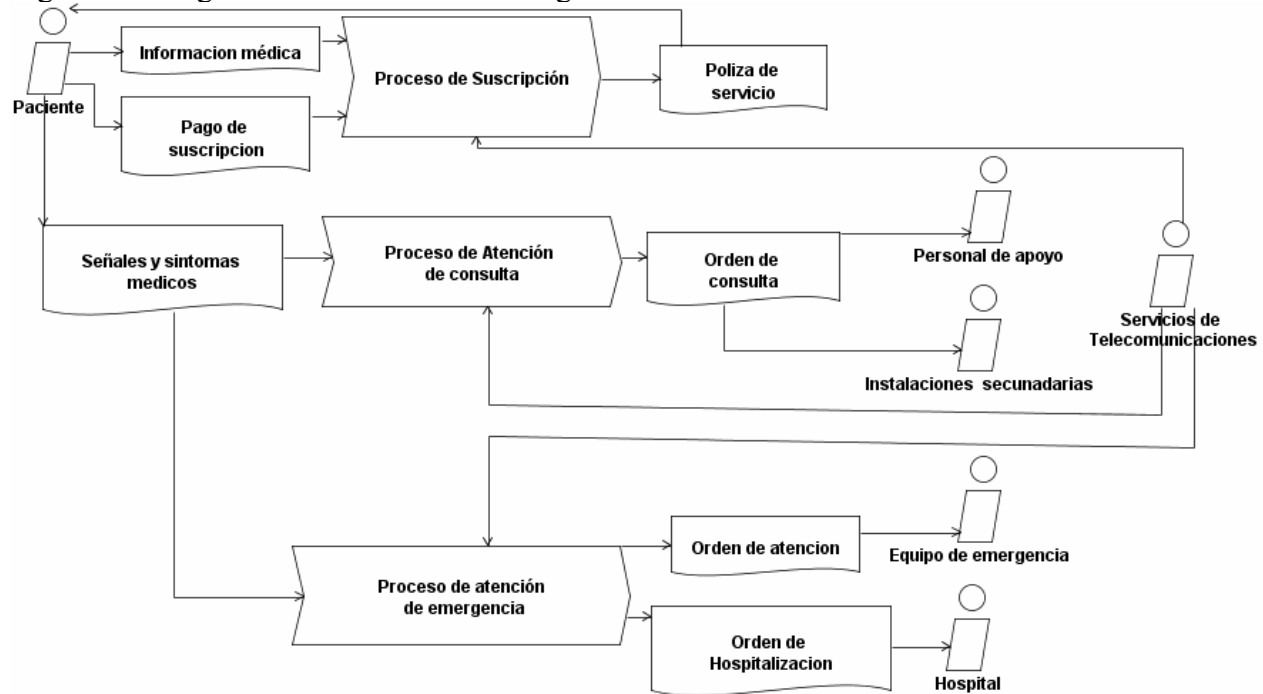
5.3.2 *Identificar los roles del entorno de negocio.*

Una vez identificados los procesos de negocio, es necesario encontrar a los agentes involucrados en su realización. Cada uno de estos agentes o actores del negocio desempeña cierto papel cuando colabora con otros para llevar a cabo las actividades que conforman dicho caso de uso del negocio. De hecho, se identifican los roles que son jugados por agentes de la propia empresa (trabajadores, departamentos y dispositivos físicos) o agentes externos (como clientes u otros sistemas).

Los agentes externos son aquellos con los que la organización interactúa para llevar a cabo sus procesos de negocio. En el ejemplo tenemos los roles Cliente y Proveedor, claramente externos al sistema.

Para tener una visión general de los diferentes procesos de negocio de la organización, puede construirse un diagrama de casos de uso del negocio, en el cual aparece cada proceso del negocio como un caso de uso. Este diagrama muestra los límites y el entorno de la organización bajo estudio. Por esto, sólo aparecerán en el diagrama los actores del negocio correspondientes a los roles externos al sistema, de forma que los procesos de negocio en los que sólo tomen parte roles internos a la organización no estarán conectados a ningún actor. (Ver Figura 5.2)

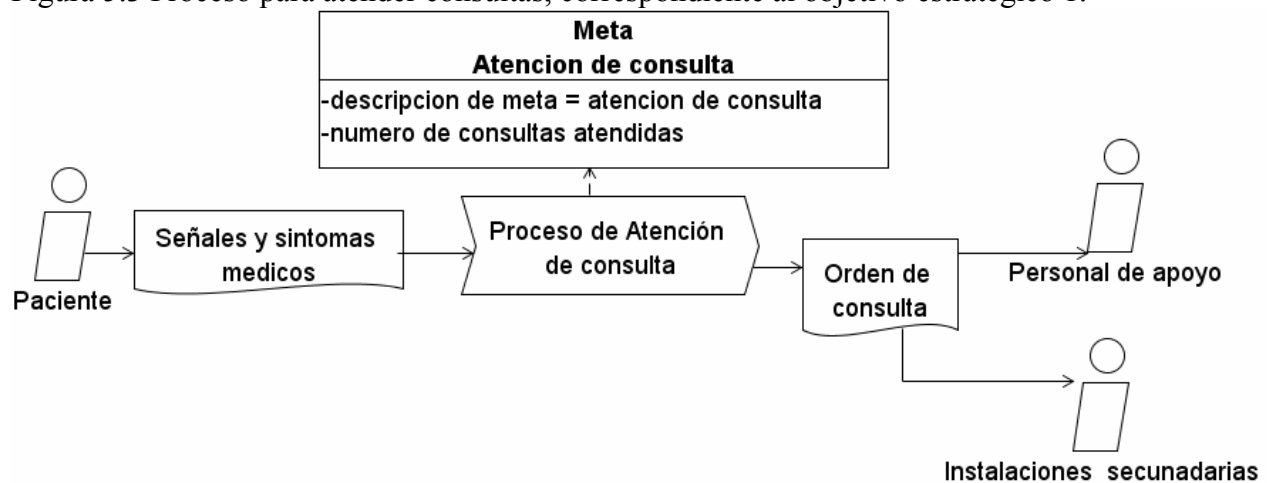
Figura 5.2 Diagrama de caso de uso del negocio.



Fuente: Elaboración propia.

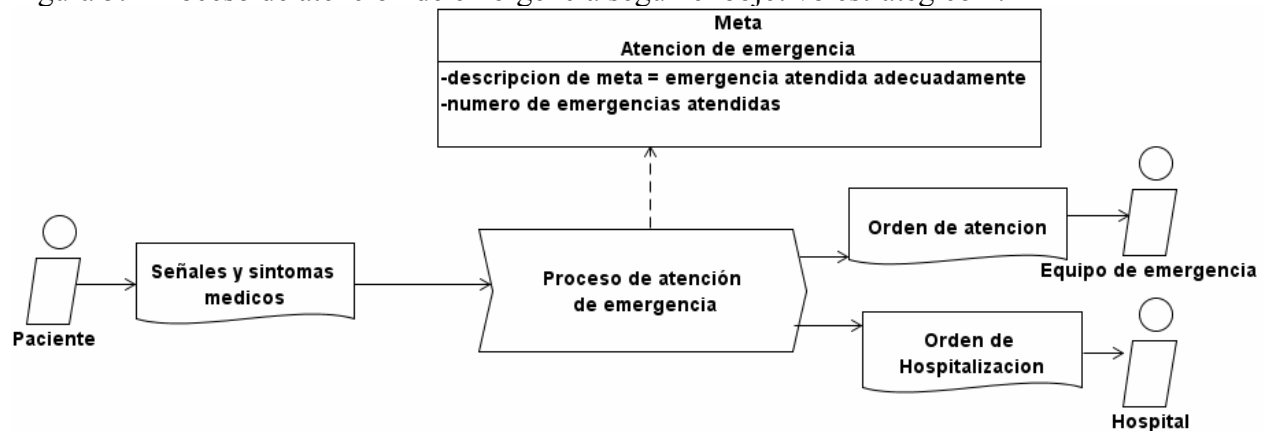
Para cada objetivo que no se puede descomponer en otros, se define un proceso de negocio cuyo propósito será dar soporte a dicho objetivo, esto es realizarlo. Representamos cada proceso del negocio mediante un caso de uso del negocio. Cada objetivo corresponde a un proceso en el sistema TRMCS (ver figuras 5.3, 5.4 y 5.5)

Figura 5.3 Proceso para atender consultas, correspondiente al objetivo estratégico 1.



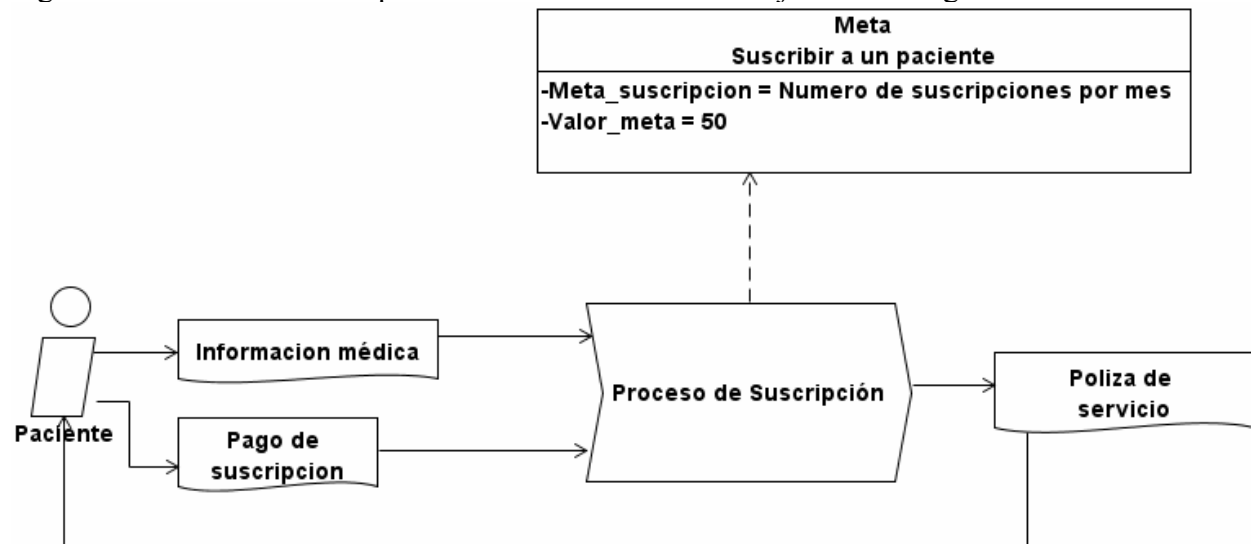
Fuente: Elaboración propia.

Figura 5.4 Proceso de atención de emergencia según el objetivo estratégico 2.



Fuente: Elaboración propia.

Figura 5.5 Proceso de Suscripción de nuevos clientes del objetivo estratégico 3.



Fuente: Elaboración propia.

5.3.3 Describir los casos de uso del negocio.

El siguiente paso dentro del modelado de negocios es profundizar en cada uno de los casos de uso del negocio identificados, para describirlos en detalle. Inicialmente se rellena una plantilla de descripción, y después, a partir de la información reflejada en dicha plantilla, se construye un conjunto de diagramas que describen completamente el caso de uso del negocio. En este caso tomamos el caso de uso Suscribir, cuya descripción se muestra en la figura 5.6. La plantilla de

descripción de casos de uso del negocio (inspirada en el conjunto de valores etiquetados utilizados por Eriksson y Penker (2000), se usa para describir un proceso de negocios), contiene los campos objetivo (lo que se intenta conseguir), descripción (especificación informal de lo que hace el proceso), prioridad (importancia del proceso, por ejemplo si es fundamental o primario, de administración, o de soporte), riesgos (por ejemplo errores o fallos que pueden ocurrir al ejecutar este proceso), posibilidades (cambios o mejoras futuras del proceso), y por último, tiempo y costo aproximados de ejecución. Esta descripción puede ser validada fácilmente por los usuarios.

La siguiente etapa consiste en determinar los agentes internos que juegan un rol en el caso de uso del negocio. Es necesario estudiar la descripción de cada caso de uso del negocio, y observar el conjunto completo de roles involucrados, tanto externos como internos a la organización.

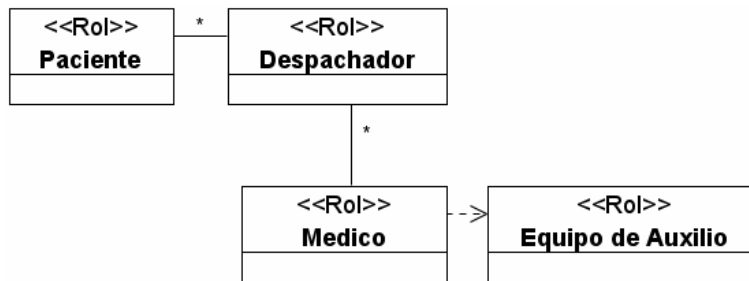
Figura 5.6 Descripción del caso de uso Suscribirse.

Nombre del caso de uso:	Suscribir
Actores:	Pacientes, sistema.
Propósito	Poder suscribir a un paciente para tener acceso a las opciones que brinda el sistema
Flujo principal: Este caso de uso se da cuando el paciente se quiere suscribir al sistema.	Subflujos:
El sistema pide al paciente los siguientes datos:	S1 Personales.
Datos personales S1: Personales	Son todos los datos personales del paciente.
❖ Apellido paterno	.
❖	.
Domicilio S2:	S6 Salir.
❖ País	El sistema cierra la ventana de suscripción y regresa a la pantalla Acceso E3.
❖	
Ficha clínica S3 Ficha Clínica	
❖ Estatura	
❖	
Esta pantalla contiene las siguientes acciones:	Excepciones:
❖ Aceptar S4: Aceptar	E1
❖ Cancelar S5: Cancelar	Si los datos solicitados no son llenados de forma correcta manda un mensaje de error.
❖ Salir S6: Salir	
Tipo:	Primario.
Referencias cruzadas:	Pantalla: Banco
	Caso de uso:
Riesgos	
Posibilidades	
Tiempo de ejecución	
Costo de ejecución	

Fuente: Elaboración propia.

El aspecto estructural de la colaboración entre los roles para llevar a cabo un caso de uso del negocio, puede ser representado en un diagrama de roles, en el que cada rol (una clase UML estereotipada) aparece asociado con los roles con los que puede colaborar (ver la figura 5.7). Por lo tanto, este diagrama permite expresar el conocimiento que unos roles tienen de otros, así como las características (como la multiplicidad) de cada relación entre roles. Este diagrama permite mostrar las características de los roles identificados, tales como sus atributos y responsabilidades, Ortín y García Molina (1999) discuten con más detalle el modelado de roles con UML.

Figura 5.7 Diagrama de roles para el caso de uso de Atención de emergencia.



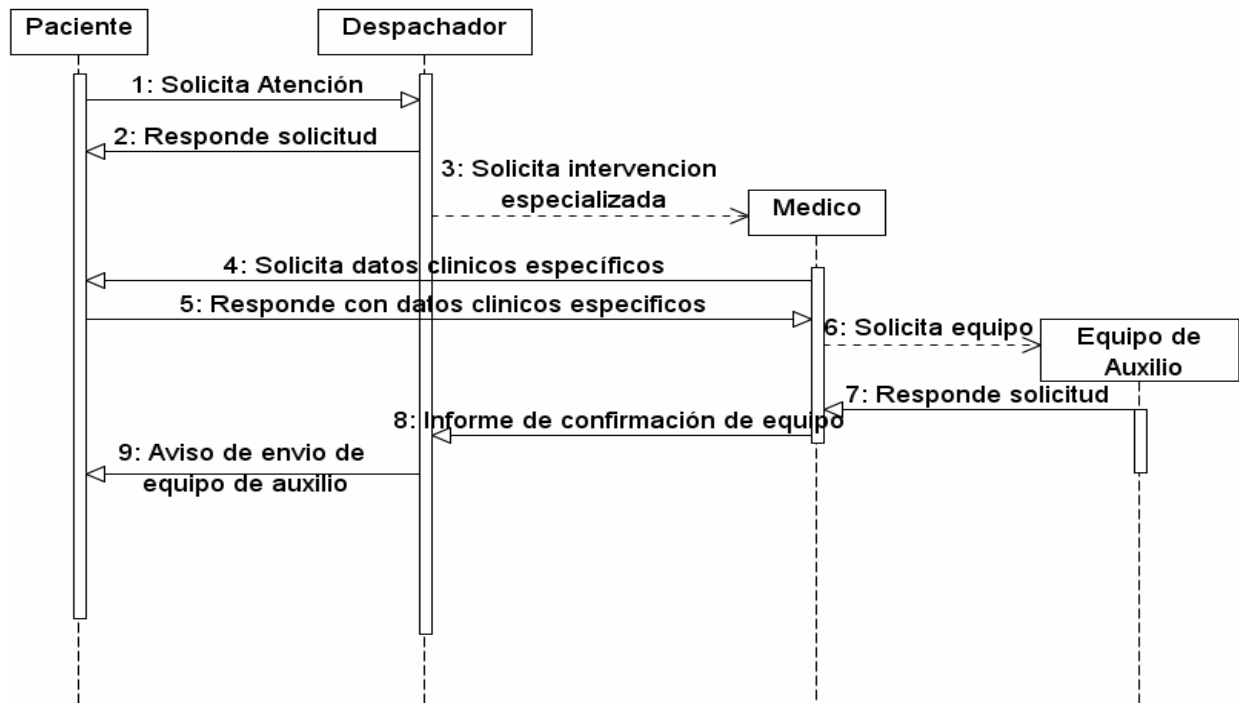
Fuente: Elaboración propia.

Lo siguiente es crear escenarios para mostrar el comportamiento de la colaboración, para lo cual se utilizan diagramas de secuencia UML (ver la figura 5. 8), donde los objetos denotan las instancias de los roles que intervienen en la interacción. En cada proceso se puede distinguir entre el flujo básico o normal de la interacción (en el ejemplo, la solicitud de atención aceptada) y los posibles flujos alternativos (por ejemplo, la cancelación o confirmación de la solicitud de atención). Es conveniente asociar varios escenarios a un mismo caso de uso del negocio, en lugar de mostrar en una única secuencia todas las posibilidades.

Para mostrar de forma más detallada el flujo de trabajo que realiza cada proceso del negocio, se utilizan diagramas de actividades con columnas (swimlanes), llamados diagramas de proceso. La figura 5.9 muestra el diagrama de proceso que incluye el escenario de la figura 5.7.

Existe una columna por cada rol participante en el escenario, que incluye las actividades que realiza dicho rol. El diagrama también muestra la información que necesita y produce cada actividad, y la sincronización requerida entre las diferentes actividades.

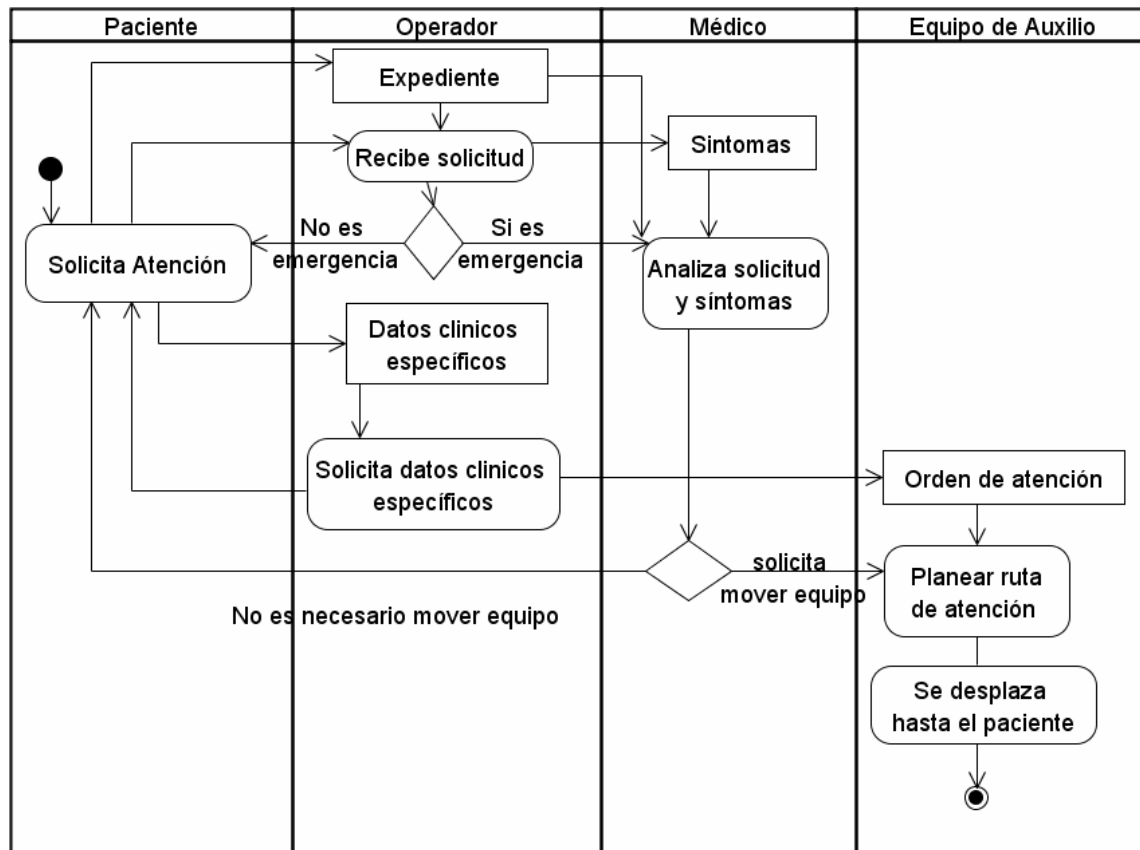
Figura 5.8 Diagrama de secuencia para el caso de uso de Atención de emergencia



Fuente: Elaboración propia.

Los datos aparecen como objetos que fluyen entre las actividades y pueden tener un estado. Durante la descripción de un proceso de negocios mediante un diagrama de proceso, es posible encontrar una actividad cuya complejidad sea tal que sea necesario describirla mediante otro diagrama de proceso adicional. Por lo tanto, este nuevo diagrama de proceso describirá un subobjetivo en relación con el objetivo ligado al proceso del negocio original. De este modo los procesos de negocio se organizan jerárquicamente. También es posible mostrar en diferentes diagramas de proceso el flujo normal y los flujos alternativos.

Figura 5.9 Diagrama de proceso.



Fuente: Elaboración propia.

5.3.4 Especificación de las reglas de negocios.

En una organización, tanto los procesos como los datos que estos manejan, están restringidos por las reglas del negocio. Estas reglas aseguran que la actividad de la empresa se lleva a cabo de acuerdo a restricciones impuestas desde el entorno (leyes o normas) o desde dentro de la propia organización. Como afirma B. Whitenack (1995), las reglas del negocio rara vez son capturadas de forma explícita durante el desarrollo del producto, a pesar de que suelen ser importantes restricciones sobre el comportamiento del sistema. Las reglas del negocio muchas veces son ignoradas hasta la fase de implementación, debido al hecho de que no exista un marco de trabajo bien definido en el que situar las reglas, y a la existencia de una gran variedad de tipos de reglas de difícil comprensión.

Con el propósito de tener en cuenta todos los tipos de reglas que aparecen en la especificación de requisitos, se utiliza la clasificación descrita por Martin y Odell (1997). Esta clasificación es sencilla pero completa, cubriendo prácticamente todos los tipos de reglas de negocios. Las categorías de reglas de negocios son las siguientes:

- Reglas de Restricción: especifican políticas o condiciones que restringen la estructura y comportamiento de los objetos y procesos. Estas reglas pueden ser divididas en reglas de estímulo-respuesta (que restringen el comportamiento y especifican las condiciones que deben cumplirse para activar una operación), reglas de restricción de operación (que especifican condiciones que deben cumplirse antes y después de ejecutarse una operación) y reglas estructurales (que especifican restricciones sobre los tipos de objetos y las asociaciones).
- Reglas de Derivación: especifican políticas y condiciones para inferir o calcular hechos (información) a partir de otros hechos existentes en el negocio. Por otro lado, están las Reglas de Existencia, introducen otro tipo de reglas del negocio, que establecen cuándo puede existir un determinado objeto (Eriksson y Penker (2000)).

De acuerdo con esta clasificación, recogemos de manera explícita cada tipo de regla en el modelo del negocio mediante la especificación de las actividades y objetos de información que aparecen en los diagramas de proceso. Estas especificaciones se reúnen en un glosario. La figura 5.10 muestra parte del glosario de términos.

Cada objeto de información se describe mediante un conjunto de atributos y sus restricciones de integridad (si las tuviera); por lo tanto, se establecen explícitamente las reglas estructurales, de derivación y de existencia. Por otro lado, la especificación de la semántica de cada actividad contendrá: origen (actividades que la preceden), agente (responsable de llevar a cabo la actividad), y pre y post-condiciones (que establecen qué tiene que cumplirse antes y después de la actividad). Estos dos últimos campos recogen las reglas de operación, mientras que las reglas de estímulo-respuesta quedan reflejadas mediante el origen, donde se expresa el orden entre las actividades, así como mediante aquellas precondiciones que representan las condiciones

necesarias para ejecutar la actividad. El glosario tendrá una estructura de hipertexto (referencias-cruzadas) con el objeto de mantener las relaciones de trazabilidad entre los procesos del negocio y las clases y los casos de uso que especifican la funcionalidad del sistema.

5.4 Modelado de requisitos: modelos conceptual y de casos de uso iniciales.

A partir del modelo de negocios descrito en la sección anterior, es posible obtener de manera sistemática y directa, tanto la colección inicial de casos de uso del sistema como el modelo conceptual preliminar. A continuación se describe cómo obtener cada modelo. Los requisitos elicitados y especificados en esta fase serán incluidos en un documento de especificación de requisitos del software (ERS).

Figura 5.10 Extracto del glosario: objetos de información y actividades.

Objeto de Información: Expediente Atributos: <i>Datos personales</i> Nº Expediente Apellido paterno Apellido materno Estado/Región País <i>Ficha clínica</i> Nº Expediente Tipo sanguíneo Síntomas, Malestar u otros comentarios adicionales. <i>Historial</i> Fecha de último Ingreso Fecha de alta Comentarios Restricciones: ▪ Los campos de datos personales deben de estar todos capturados. ▪ Los campos de datos de ficha clínica deben de estar capturados ▪ El campo de Médico responsable debe estar capturado. Clases del Dominio: <i>Pendiente por especificar.</i>	Actividad: Recibe solicitud Origen: Solicita Atención Agente: Operador Precondiciones: ▪ El solicitante es usuario activo del sistema. ▪ La solicitud por síntomas presentados es válida Postcondiciones: ▪ Se ha registrado la solicitud. ▪ Se ha determinado si es una emergencia o no lo es. Caso de uso del sistema: <i>pendiente por especificar</i> Actividad: Analiza Solicitud y síntomas Origen: Recibe solicitud Agente: Médico
---	---

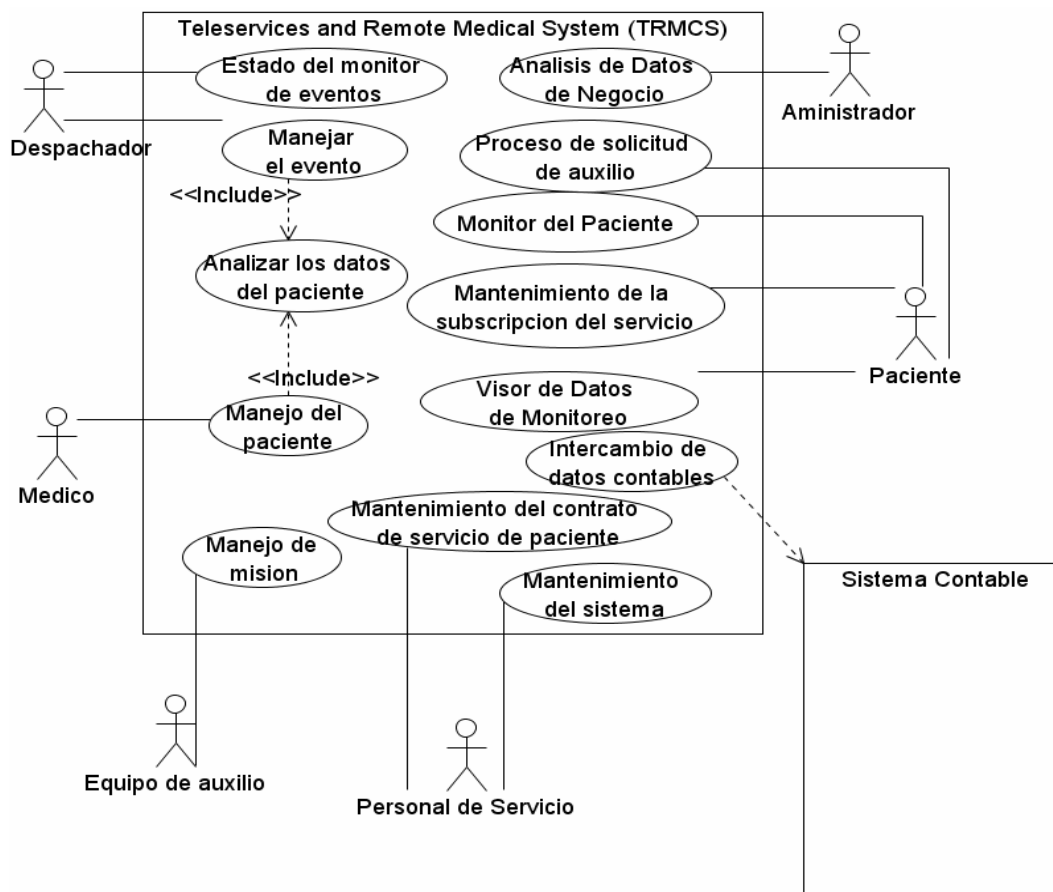
Fuente: Elaboración propia.

5.4.1 Obtención del modelo inicial de casos de uso del sistema.

Usualmente las actividades del diagrama de proceso tienen un nivel de granularidad adecuado para ser asociadas a un caso de uso del sistema. De tal forma que se puede crear un caso de uso del sistema por cada actividad del diagrama de proceso que deba ser soportada por el sistema software. Por lo tanto, el rol que lleva a cabo la actividad será el actor principal del caso de uso. Nótese que, de acuerdo con la definición de caso de uso, no todas las actividades del diagrama de proceso serán consideradas como casos de uso, sino solamente aquellas que sean de valor para algún actor.

La figura 5.11 muestra el diagrama de casos de uso del sistema, considerando que todas las actividades serán soportadas por el sistema software.

Figura 5.11 Diagrama inicial de caso de uso del sistema.



Fuente: Elaboración propia.

Cabe señalar que algunos casos de uso no se obtienen directamente a partir de los diagramas de proceso. Estos nuevos casos de uso se detectarían al describir los casos de uso identificados y al adquirir un mayor conocimiento sobre los requisitos que deben ser soportados, y representarían funciones que debe llevar a cabo el sistema para lograr algún objetivo asociado con algún caso de uso ya existente. Normalmente los casos de uso detectados de esta manera serán casos de uso de soporte, ya que no surgen directamente de la descripción de los procesos de negocio.

De este modo, el modelo de negocios permite obtener los casos de uso más importantes dentro de cada proceso del negocio, y además facilita la determinación del conjunto adecuado de pasos incrementales en el proceso iterativo de desarrollo, tal y como defiende Korson en 1998.

Los casos de uso se pueden organizar en varios niveles (dos o tres como máximo) de acuerdo con la descomposición jerárquica propuesta en el modelado del negocio. Cada caso de uso se escribirá mediante una plantilla que puede rellenarse a partir de la especificación de la actividad asociada, que se encuentra recogida en el glosario. La plantilla que propuesta se basa en la de Coleman (1998), a la que se le ha añadido un campo para las postcondiciones del caso de uso, tal como se muestra en la figura 5. 9.

Figura 5.12 Descripción del caso de uso del sistema

Nombre del caso de uso:	Principal
Actores:	Sistema, Doctor, Enfermera.
Propósito	Mostrar las opciones que el usuario puede ejecutar en el sistema.
Flujo principal: Este caso de uso se da cuando el usuario quiere ejecutar alguna acción el sistema.	Subflujos:
Esta pantalla contiene un menú desplegable con las siguientes opciones:	S1 Altas.
❖ Archivo:	Muestra la ventana de altas de pacientes, doctores o enfermeras dependiendo de que quiera dar de alta el usuario.
*Altas S1:Altas	S2 Bajas
Doctores	Muestra la ventana de bajas de pacientes, doctores o enfermeras dependiendo de que quiera dar de baja el usuario.
Pacientes	S3 Salir
Enfermeras	Sale del sistema.
*Bajas S2:Bajas	S4 Cuentas de usuario.
Doctores	Muestra la ventana de cuentas de usuario.
Pacientes	S5 Cambiar contraseñas.
Enfermeras	Muestra la pantalla de cuentas de usuario (véase caso

*Salir S3: Salir	de uso Cuentas de usuario).
❖ Herramientas:	S6 Citas.
*Cuentas de usuario S4: Cuentas de usuario	Muestra la pantalla de citas (véase caso de uso Citas).
*Cambiar contraseña S5: Cambiar contraseña	S7: Recetas.
❖ Consultas:	Muestra la ventana de recetas (véase caso de uso Recetas).
*Citas S6: Citas	S8: Historial.
❖ Reportes:	Muestra la ventana de historial (véase caso de uso Historial).
*Recetas S7: Recetas	S9, 13 Pacientes.
*Historial S8: Historial	El sistema muestra la interfaz de la ventana de pacientes (véase caso de uso Pacientes).
❖ Buscar:	S10, S14 Doctores el sistema muestra la interfaz de la ventana de doctores (véase caso de uso Doctores).
*Paciente S9: Paciente	S 15 Enfermeras el sistema muestra la interfaz de la ventana de enfermeras (véase caso de uso Enfermeras).
*Doctores S10: Doctores	S12: Emergencia.
*Enfermera S12: Enfermera	Muestra al usuario la pantalla emergencia del Sistema TRMCS.
❖ Emergencia:	S16 Buscar:
*Emergencia S11: Emergencia	El sistema un mensaje solicitando lo que se desee buscar E1.
Además cuenta con una barra de herramientas que contiene las siguientes opciones:	S17 Salir :
❖ Pacientes S13: Pacientes	Si el usuario pulsa el botón de salir sale del sistema E2.
❖ Doctores S14: Doctores	Excepciones:
❖ Enfermeras S15: Enfermeras	E1
❖ Buscar S16: Buscar	Si la información que se desee buscar no existe manda un mensaje de error. La información no existe.
❖ Salir S17: Salir	E2 Si el usuario pulsa en el botón de cancelar regresa al menú principal.
Tipo:	Primario
Referencias cruzadas:	Pantalla: Principal
Asunciones	Caso de uso: Manejo principal de botones del sistema
Poscondiciones	<i>Por especificar</i>
	<i>Por especificar</i>

Fuente: Elaboración propia.

Una vez descrito el caso de uso, se conectará a la especificación de la actividad asociada en el glosario, con el objeto de mantener la trazabilidad entre los casos de uso del negocio y los del sistema. También podrían encontrarse relaciones entre los casos de uso, tales como incluye, si se

detectan aspectos comunes entre varios casos de uso, y extend, para expresar caminos opcionales o alternativos en un caso de uso. No obstante, se recomienda no abusar de estas relaciones y no mostrarlas en los diagramas de casos de uso.

5.4.2 *Obtención del modelo conceptual inicial.*

Los objetos de información que fluyen entre las actividades de un caso de uso del negocio representan datos del dominio, por lo que suponen una buena base para crear el modelo conceptual inicial. Este modelo incluirá los conceptos y sus relaciones y se describirá mediante un diagrama de clases UML, en el que los conceptos se representan mediante clases (clases del dominio). La figura 5.13 muestra el diagrama de clases que describe el primer modelo conceptual del caso de estudio.

El modelo conceptual inicial será refinado posteriormente gracias a la experiencia del modelador. Este es un buen punto de partida, en lugar de abordar la construcción del modelo conceptual partiendo de la nada. Así, cada objeto de información del diagrama de proceso se convertirá ahora en un concepto (y en la etapa de diseño dará lugar a una clase si el sistema software debe dar soporte a dicho concepto).

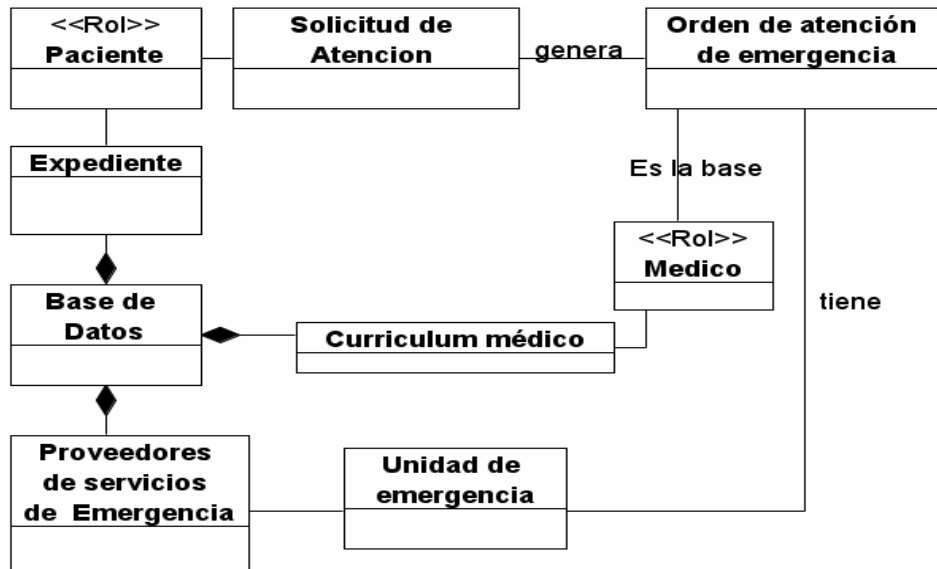
A partir de la especificación de cada objeto de información se obtiene la definición del concepto asociado, es decir, sus atributos, relaciones con otras clases y restricciones. Por ejemplo, a partir de la especificación de Expediente mostrada en la figura 5.10, se puede obtener: i) los atributos Datos personales, Ficha clínica e Historial; ii) las relaciones Paciente-Médico y Paciente-Familiar, y iii) restricciones que podrían ser expresadas textualmente o bien mediante OCL (Object Constraint Language), como {fechaUltimoIngreso > fechaAlta}.

Nótese que cuando un modelo conceptual evoluciona hacia un diagrama de clases, algunas responsabilidades se pueden obtener a partir de ciertas restricciones ya especificadas en el glosario. Por ejemplo, la clase Solicitud de Atención podría tener responsabilidades como ObtenerMédico, ElaborarDiagnóstico, EnviarEquipoDeEmergencia.

En esta etapa del desarrollo, es recomendable centrarse más en la identificación y especificación de los conceptos que en las relaciones entre ellos, puesto que éstas serán refinadas en fases posteriores.

Por el momento, se deben observar las asociaciones del tipo debe-conocer y la generalización para identificar las relaciones entre conceptos más importantes. Por ejemplo, a partir del glosario podemos establecer que un médico debe conocer al paciente que solicita la atención y los padecimientos que lo aquejan (ver figura 5. 10).

Figura 5.13 Modelo conceptual inicial para el caso de uso del negocio solicitud de atención.



Fuente: Elaboración propia.

Por otro lado, alguno de los roles identificados en el modelo de negocios, y por tanto especificado en el modelo de roles, podría ser incluido como una clase en el modelo conceptual. Es el caso de la clase Paciente en el ejemplo.

De igual forma que se conecta en el glosario las actividades con los casos de uso del sistema, se vincula cada objeto de información con la clase del dominio que lo representa en el sistema. El modelo del negocio permite además identificar aquellas clases cuyo comportamiento depende de

un conjunto no trivial de estados alcanzables. En estos casos, es necesario definir una máquina de estados mediante un diagrama statechart UML.

Estas clases se detectan con facilidad en los diagramas de proceso, puesto que se corresponden con objetos de información etiquetados con varios estados diferentes. En el ejemplo, Atención de Emergencia sería candidato para construir una máquina de estados que mostrase los estados de un proceso de atención de emergencia (solicitud, en evolución, diagnosticado, envío de equipo médico o atendido) y los eventos que producen los cambios entre estados.

5.4.3 Representación de la reglas del negocio.

Las reglas de negocio estructurales, de derivación y de existencia quedan plenamente expresadas en el modelo conceptual. La propia sintaxis del diagrama de clases de UML permite representar los atributos de los conceptos y sus relaciones, mediante los atributos de las clases correspondientes y asociaciones entre éstas. Las reglas de derivación pueden ser especificadas mediante expresiones OCL dentro de constraints de UML colocadas en el diagrama cerca del elemento derivado, o bien dentro de notas conectadas a dicho elemento. La multiplicidad de las asociaciones permite por ejemplo representar las reglas del tipo de una solicitud será requerida por uno y sólo un paciente. Otras restricciones pueden expresarse en OCL utilizando constraints cercanas a los elementos a los que restringen. También es posible utilizar OCL o lenguaje natural para expresar una restricción dentro de una nota conectada a los elementos afectados por dicha restricción. Las reglas de existencia están implícitas en el modelo conceptual, por ejemplo en el caso de un objeto agregado, como Base de Datos, cuya existencia no es posible a menos que existan los objetos que lo componen.

Por otro lado, las reglas de negocio de operación quedan expresadas en la plantilla de descripción de los casos de uso, puesto que las precondiciones y postcondiciones de las operaciones especificadas en el glosario se recogen respectivamente en los campos Asunciones y Postcondiciones. Por último, las reglas de estímulo/respuesta, expresadas mediante los campos origen y precondiciones de la especificación de las actividades, quedarán recogidas en el campo Asunciones de las plantillas de casos de uso.

5.4.4 *Identificación de requisitos no funcionales.*

Para completar esta fase se deben establecer los requisitos no funcionales, relacionados por ejemplo con el rendimiento, la disponibilidad o la seguridad. Cuando estén asociados a un caso de uso, podrán especificarse en la plantilla de caso de uso propuesta.

Los requisitos no funcionales que afecten a varios casos de uso o sean globales a toda la organización, serán recogidos en el apartado correspondiente de la plantilla de ERS elegida. El modelo del negocio puede ayudar a encontrar requisitos no funcionales, tal y como lo indican Eriksson y Penker (2000), pues por ejemplo los campos tiempo de ejecución y costo de ejecución de la plantilla de descripción de casos de uso del negocio expresan necesidades no funcionales que deben ser trasladadas a la plantilla de ERS y asociadas a los correspondientes casos de uso del sistema. Por otro lado, el que los procesos del negocio sean el resultado del análisis de los objetivos de la organización, permite que los requisitos (funcionales y no funcionales) del sistema puedan ser validados y verificados contra los objetivos.

5.5 Modelado Web.

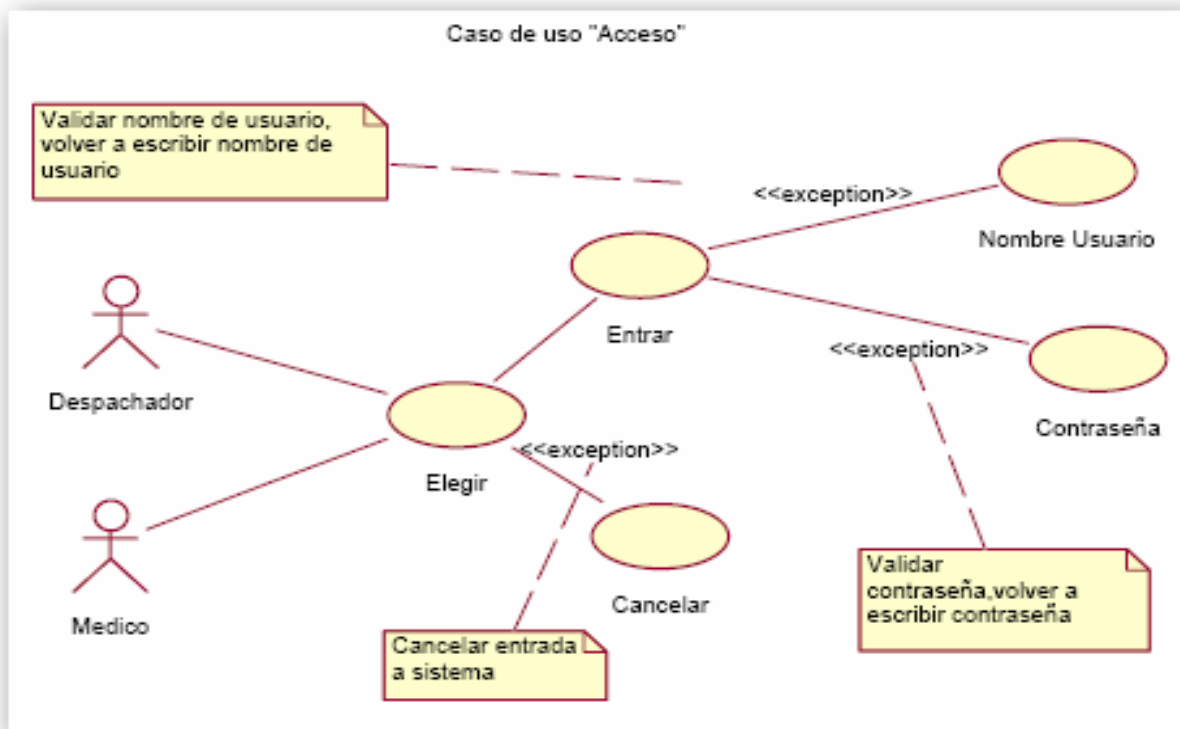
El término aplicación Web (Vidal, 2003; Conallen, 2002) ha ido evolucionando desde un pequeño sitio Web a una aplicación robusta y entera. En la actualidad entendemos por aplicación Web, a cientos de usuarios simultáneos, distribuidos alrededor del mundo, todos conectados a la vez a una aplicación de software e interactuando con ella. Este término varía de acuerdo a diferentes autores, y va desde aquellos que creen que una aplicación Web es cualquier página que use java, hasta aquellos que consideran que es cualquier sistema que usa un servidor Web. En este caso, una aplicación Web, está formada por un servidor Web en el cual los usuarios introducen datos que afectan la condición del negocio. La diferencia entre una aplicación Web y un sitio Web radica en su uso. Un sitio Web se limita a una dirección de Internet donde se muestran páginas referentes a cierta organización, mientras que una aplicación Web implementa la lógica del negocio de la organización, y usa cambios de estados del negocio. Así, las aplicaciones Web son sistemas de información donde una gran cantidad de datos volátiles y altamente estructurados, son consultados, procesados y actualizados mediante navegadores de Internet.

Esto permite mantener las correspondientes relaciones de trazabilidad entre los diferentes artefactos del modelado. En este trabajo hemos expuesto cómo el modelado del negocio puede facilitar la identificación de los requisitos tanto funcionales como no funcionales del sistema.

Además, el hecho de que tales requisitos surjan de la descripción de los procesos del negocio, y que éstos sean el resultado del análisis de los objetivos de la organización, posibilita que los requisitos del sistema sean validados y verificados contra los objetivos del negocio.

Para construir el modelo Web, se parte de los casos de uso del modelo de requisitos, en donde se identifican los servicios en la Web que ofrecerá el sistema. Para el caso de estudio se identificaron los servicios: Acceso, Actualizar, Consulta, Estados de TRMCS, Servicios y Suscripción. Para modelar el servicio de acceso en la Web, se diseñó un caso de uso llamado “**Acceso**”, en el que se toman en cuenta los actores que participan en el sistema y los procesos que realizan, como se muestra en la figura 5.14

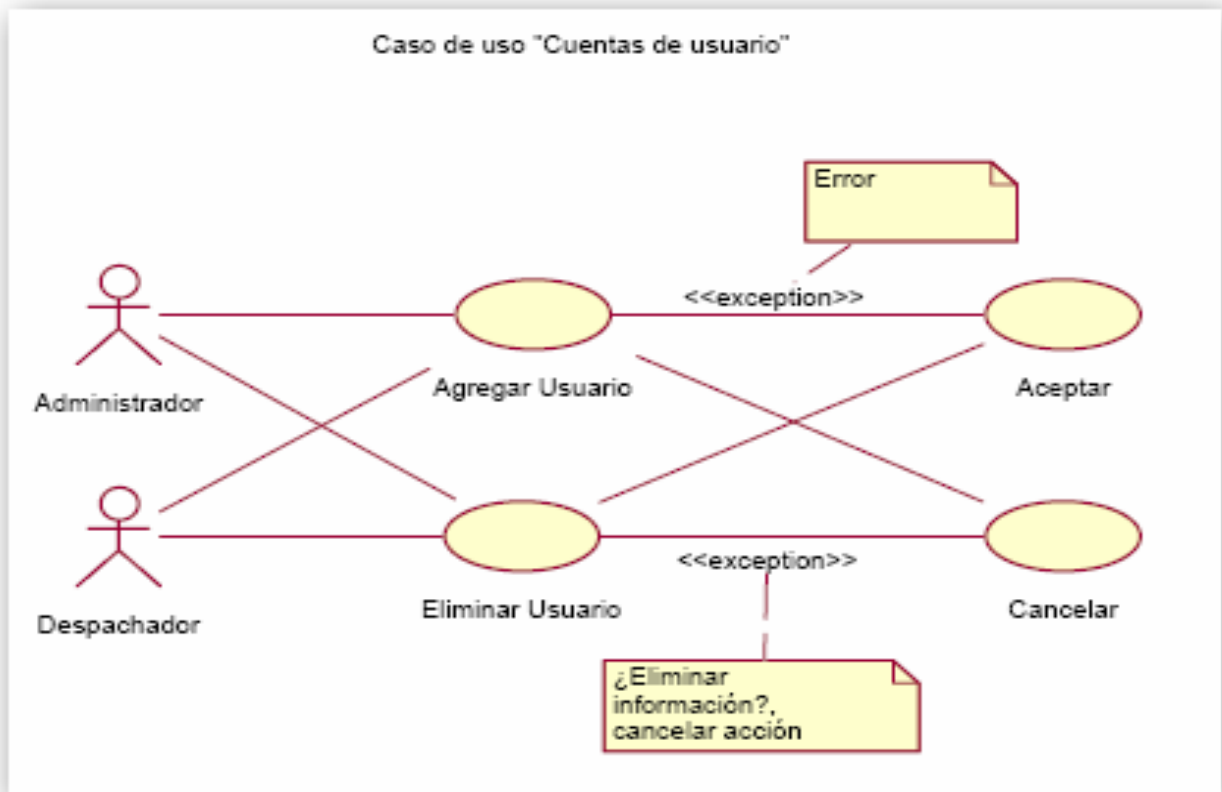
Figura 5.14 Caso de uso Acceso.



Fuente: Elaboración propia.

Con el propósito de contar con medidas de seguridad y control en cuanto al acceso por parte de los usuarios, es necesario contar con un conjunto de claves de acceso. En la figura 5.15 se muestra el caso de uso encargado de administrar las contraseñas.

Figura 5.15 Caso de uso Cuentas de Usuario

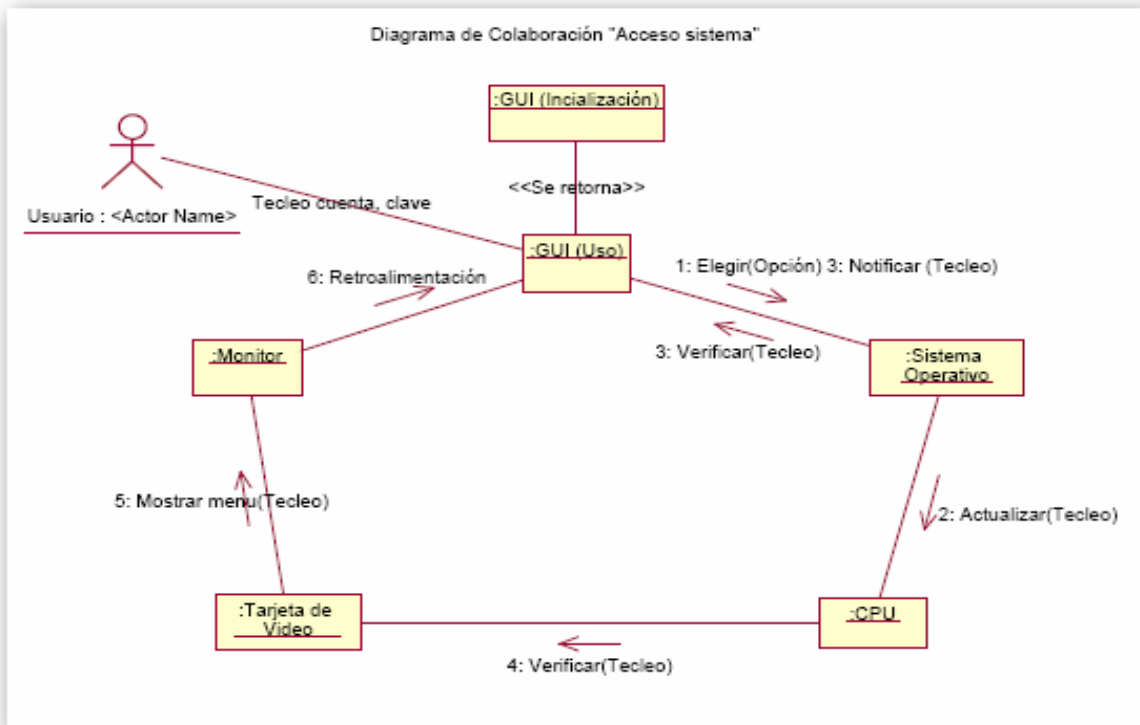


Fuente: Elaboración propia.

En la figura 5.16 se muestra el diagrama de colaboración "Acceso Sistema" en el que se muestra la secuencia de pasos para tener acceso al sistema.

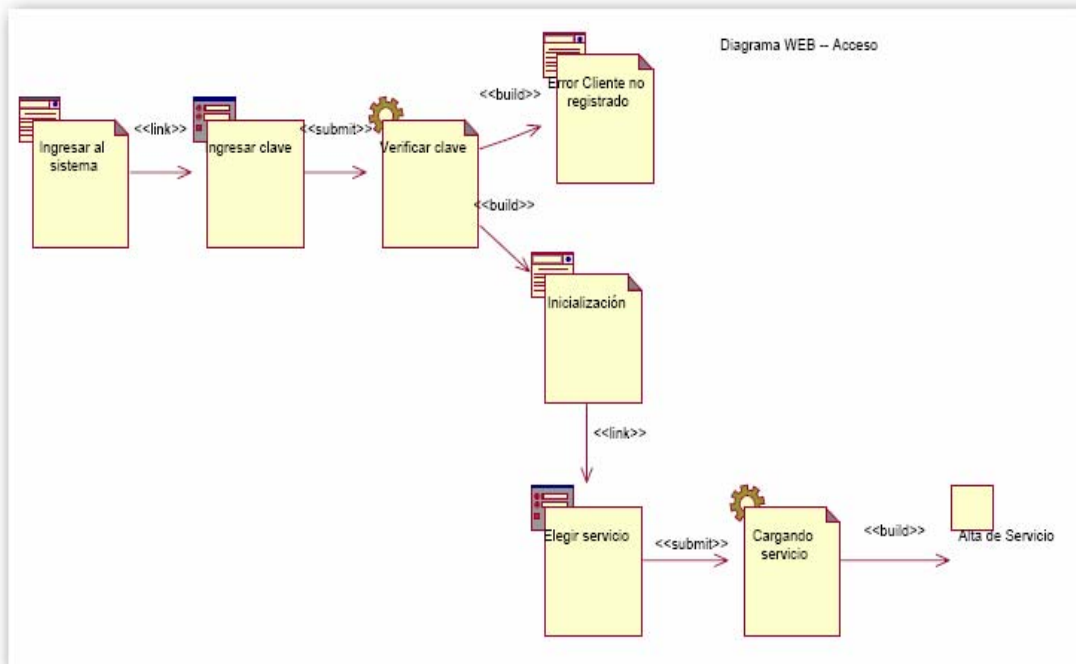
Con la información de los diagramas anteriores se diseña el diagrama correspondiente al servicio Web de acceso al sistema que se muestra en la figura 5.17.

Figura 5.16 Diagrama de colaboración Acceso_sistema.



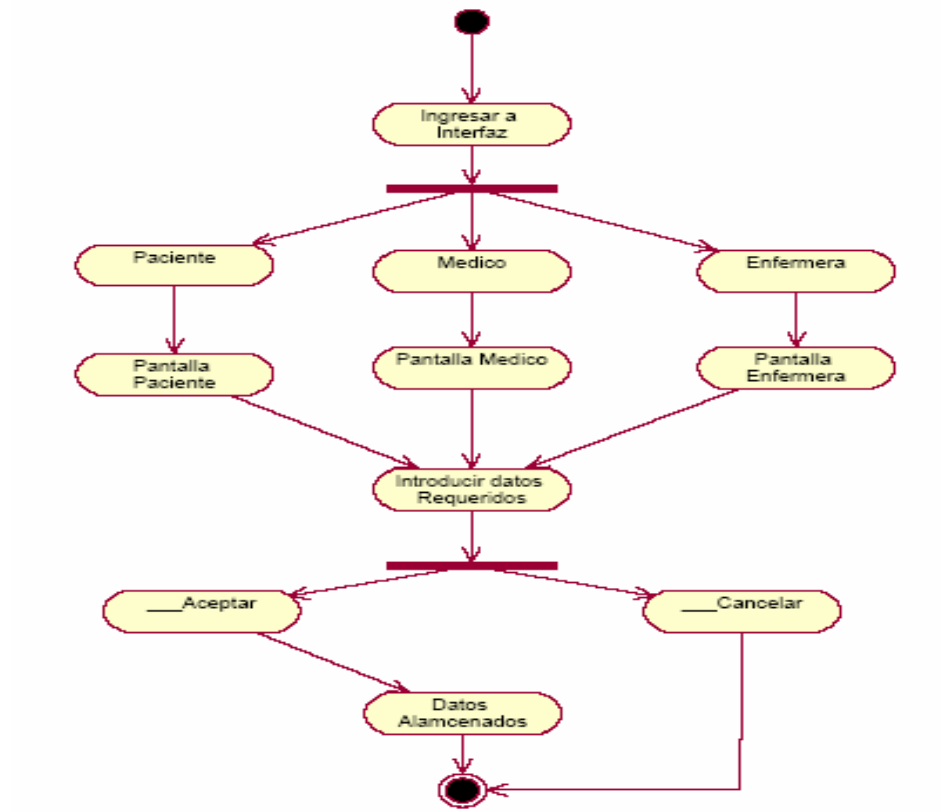
Fuente: Elaboración propia.

Figura 5.17 Diagrama Web Acceso



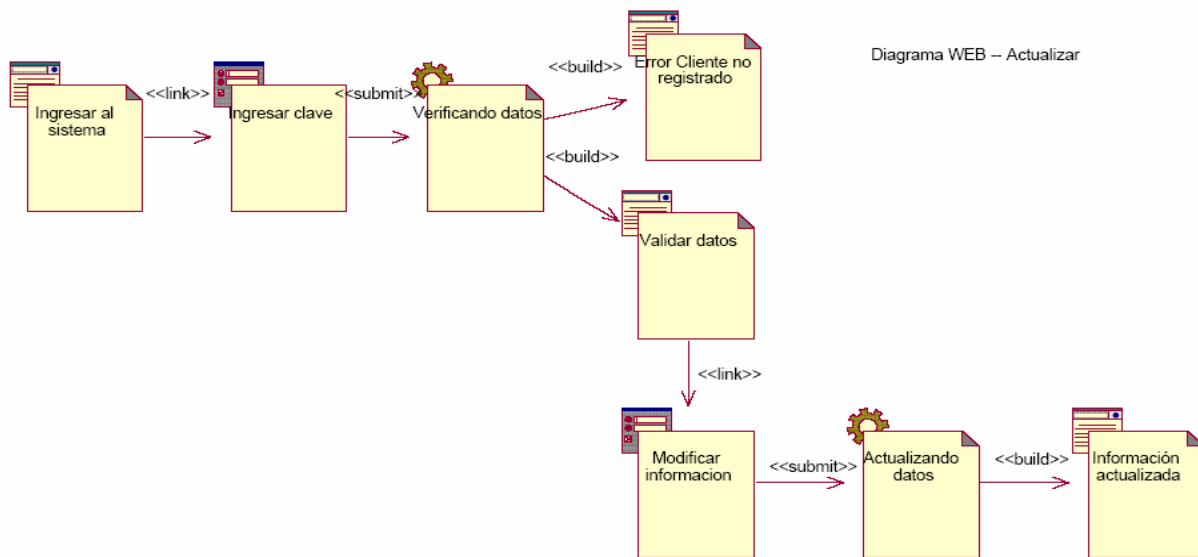
Fuente: Elaboración propia

Figura 5.19. Diagrama de actividades “Altas”.



Fuente: Elaboración propia.

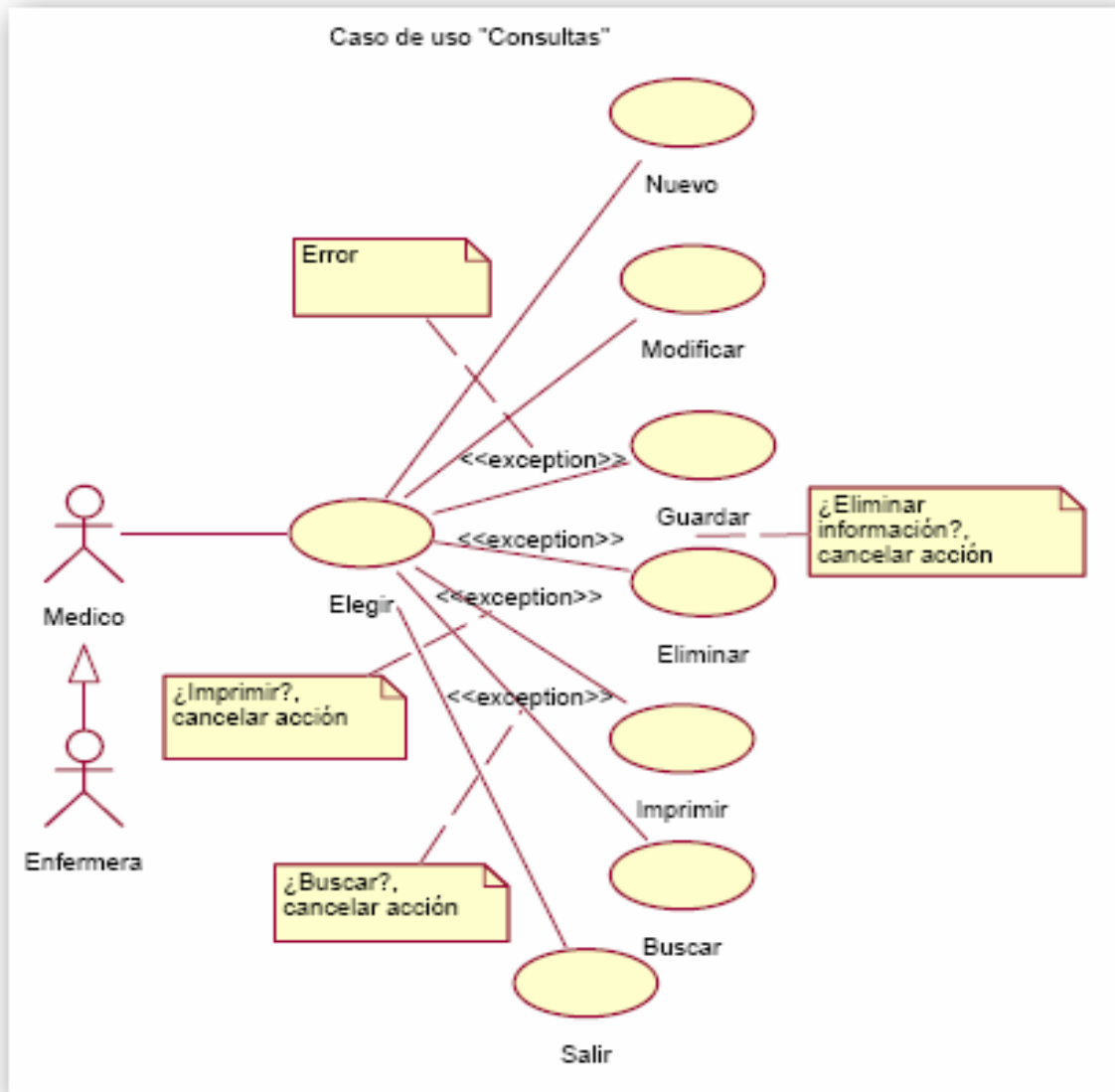
Figura 5.20 Diagrama Web “Actualizar”.



Fuente: Elaboración propia.

Para el servicio WEB “Consulta” de la figura 5.21 se toman en cuenta los casos de uso de los actores que forman parte del sistema, en este caso: “Doctor”, “Enfermera” y “Paciente”, también se utilizan otras funciones como la de modificar, historial, generación de reportes y el envío de equipo de emergencia si se da el caso.

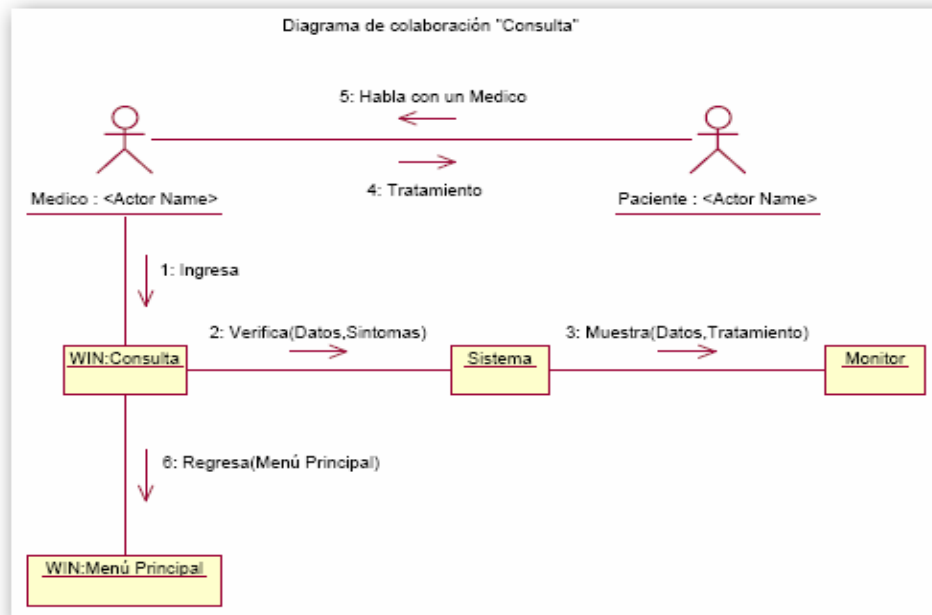
Figura 5.21 Caso de uso Consultas.



Fuente: Elaboración propia.

La figura 5.22 muestra la colaboración entre el usuario y el sistema para realizar una consulta.

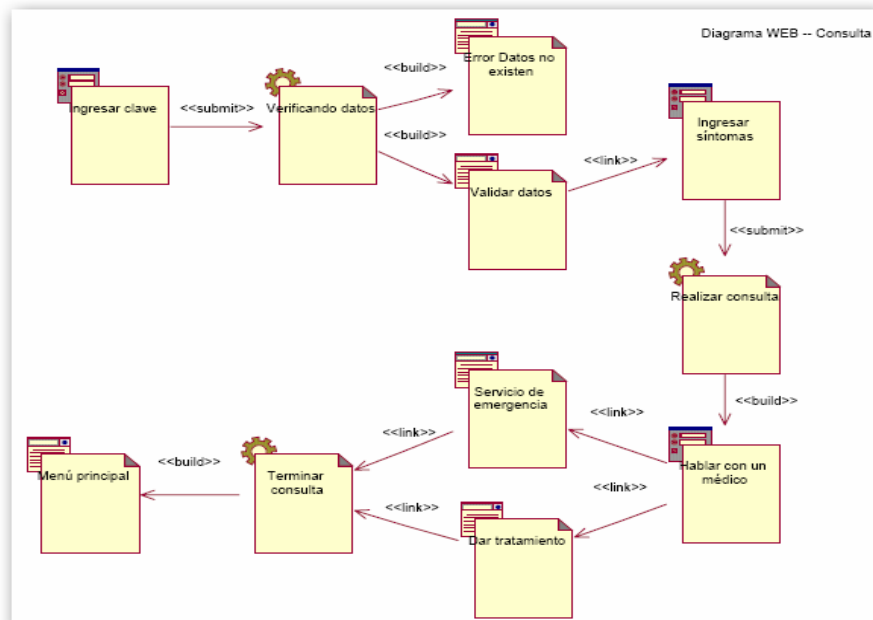
Figura 5.22 Diagrama de colaboración Consultas.



Fuente: Elaboración propia.

Con el diagrama de colaboración elaborado, se prosigue con el diseño del diagrama Web que identifica el servicio Consultas como se muestra en la figura 5.23.

Figura 5.23 Diagrama Web Consultas.

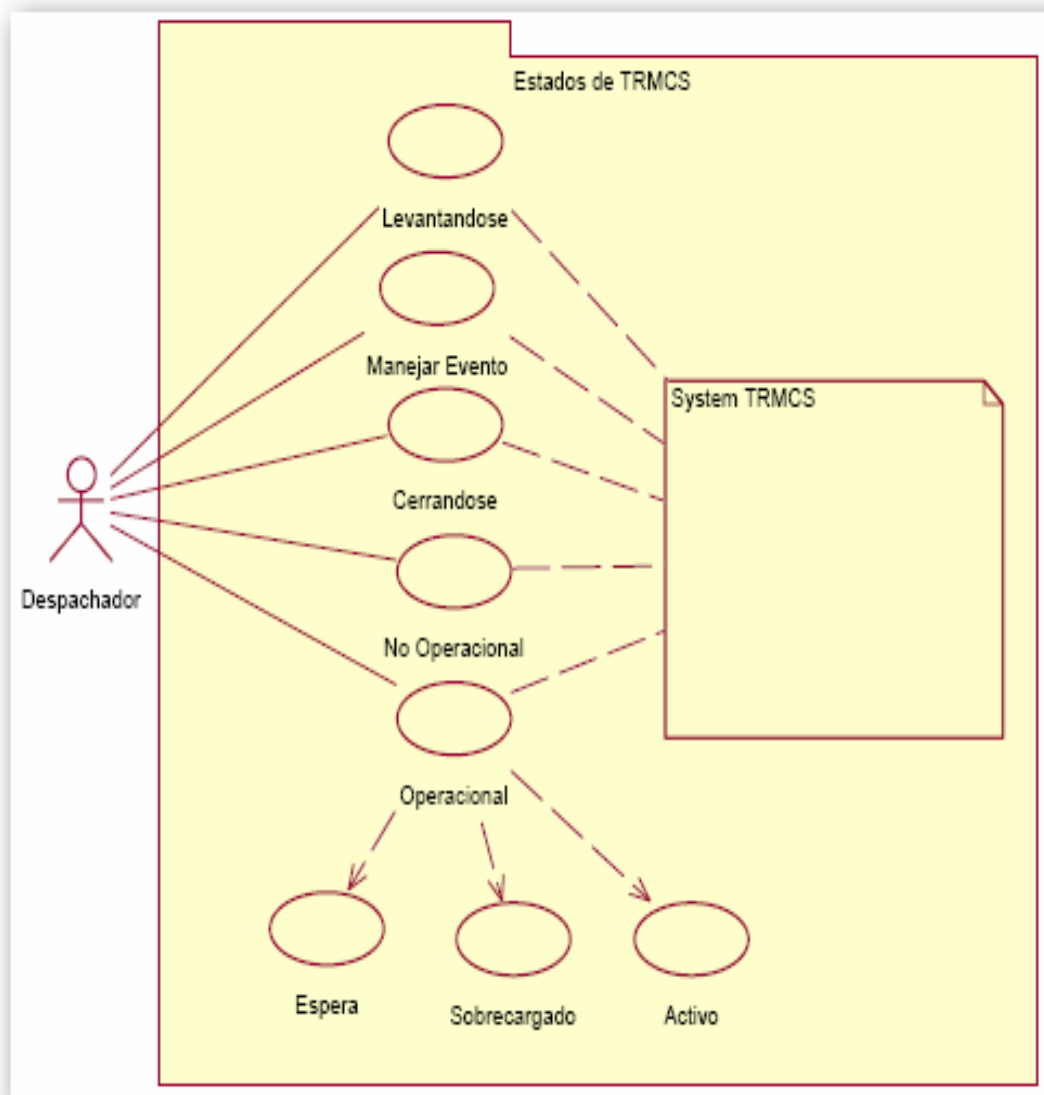


Fuente: Elaboración propia.

Con respecto a los diferentes estados del sistema, se desarrolló el diagrama de caso de uso “Estados de TRMCS” que se muestra en la figura 5.24, en el que se identifican los estados por lo que pasa el sistema desde el acceso hasta el cierre de cada función del sistema.

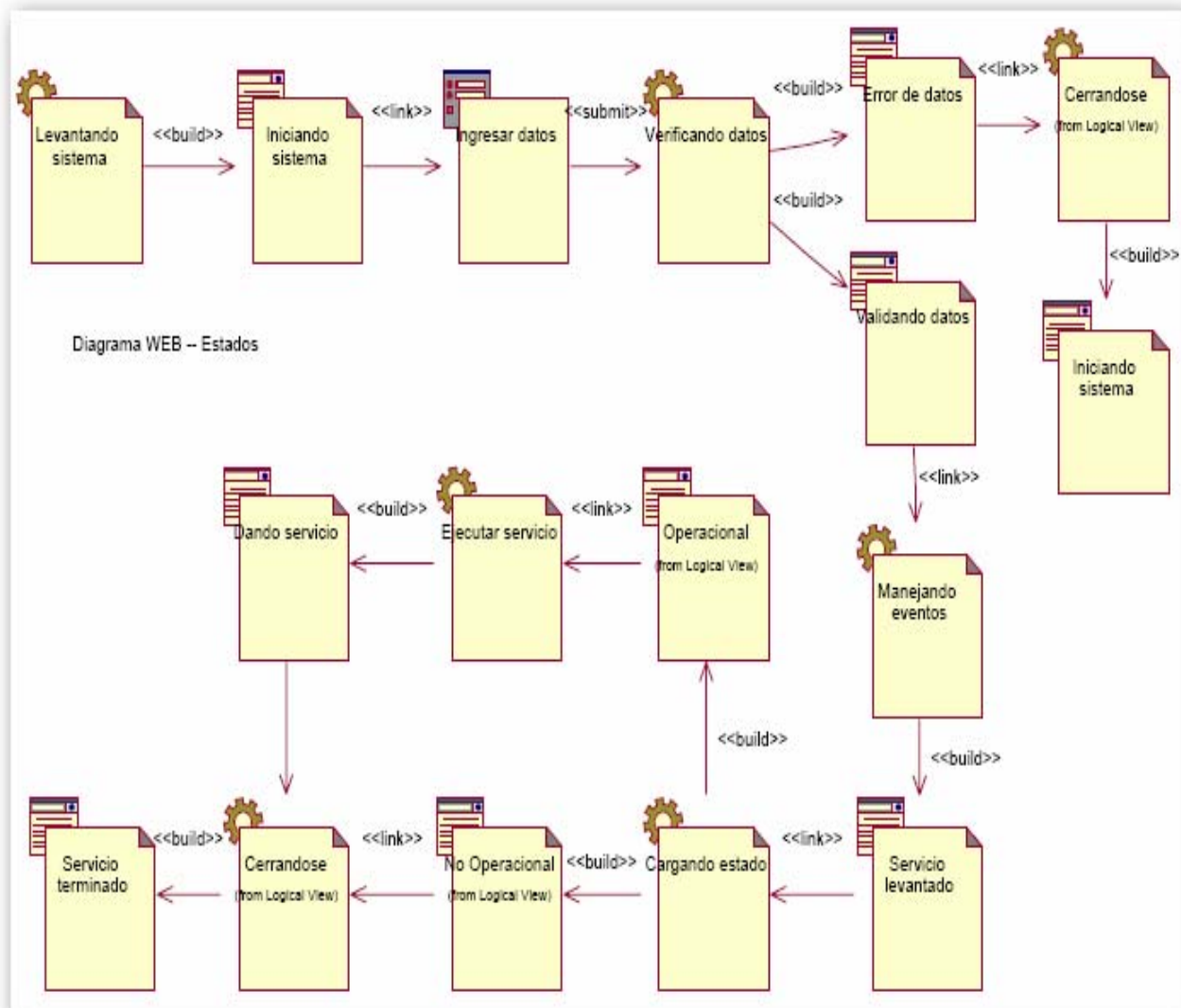
Habiendo diseñado este caso de uso, se diseña el diagrama Web que muestra los diferentes estados en los que puede estar el sistema. Ver la figura 5.25.

Figura 5.24 Casos de uso Estados



Fuente: Elaboración propia.

Figura 5.25 Diagrama Web Estados.

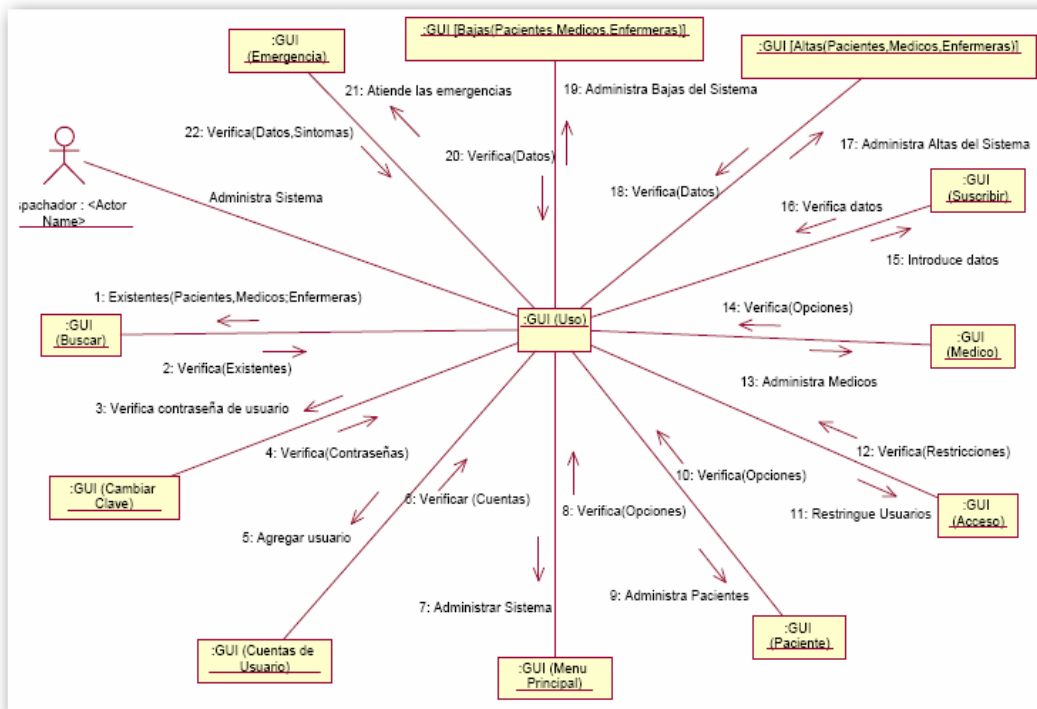


Fuente: Elaboración propia.

Los servicios que se ofrecerán se identifican a partir del caso de uso principal, ya que refleja las actividades del sistema de acuerdo a cada servicio, con base en este diagrama, se elabora el correspondiente a la colaboración entre el usuario y el sistema de acuerdo a los servicios que se ofrecerán en la Web como se muestra en la figura 5.26.

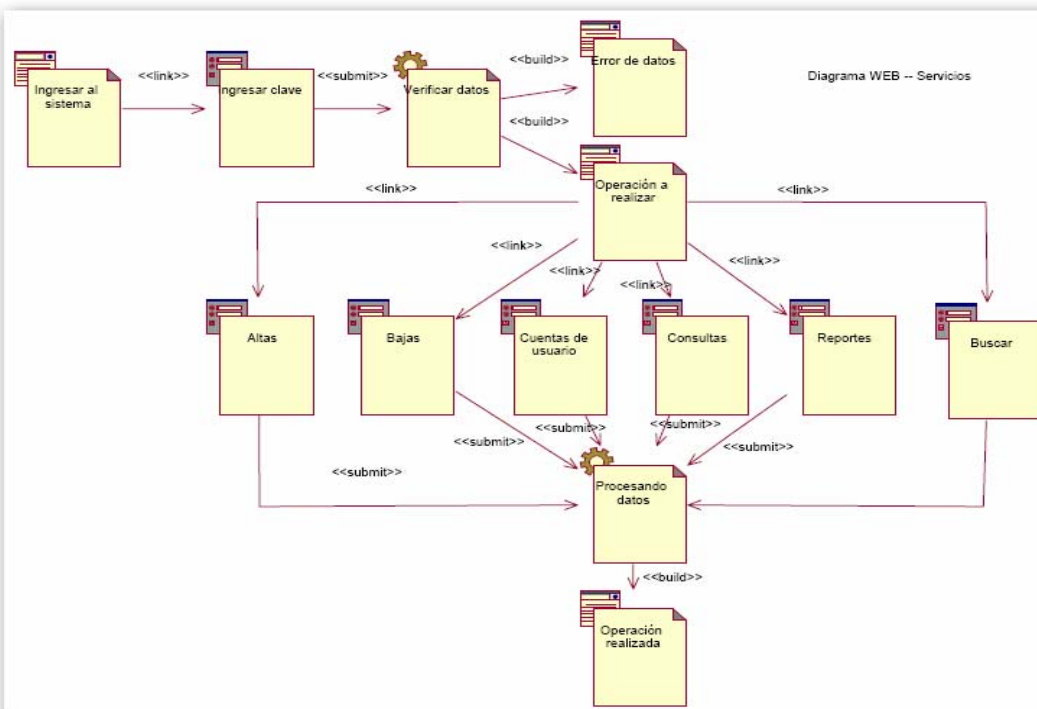
Contando con esta información, se realiza el diseño del diagrama Web en el que se muestra como se llevarán acabo los servicios que se van a ofrecer. Véase la figura 5.27.

Figura 5.26 Diagrama de Colaboración principal



Fuente: Elaboración propia.

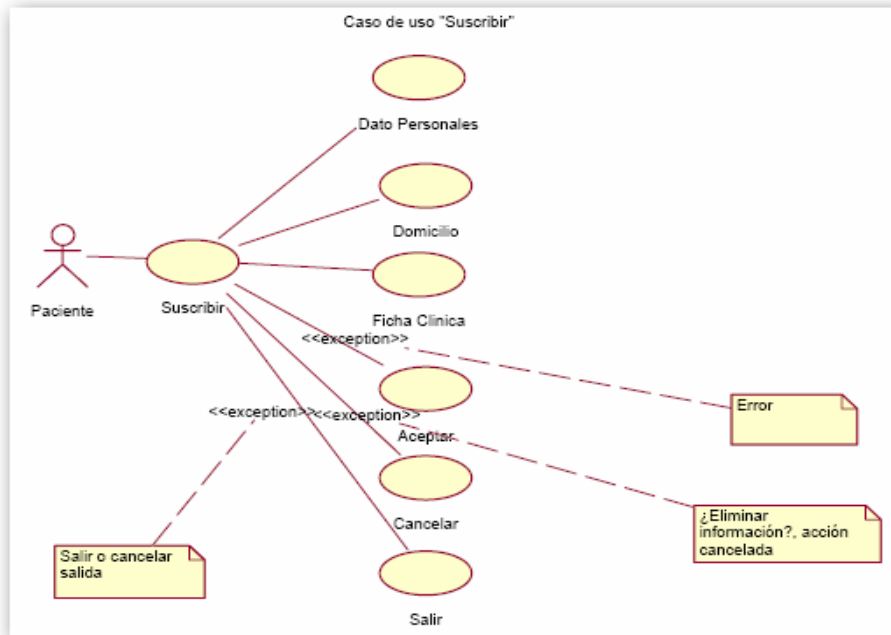
Figura 5.27 Diagrama Web Servicios



Fuente: Elaboración propia.

Por último está el **servicio “Suscripción”**, en este se toman los casos de uso “Cuentas de usuario” para crear una contraseña y el caso de uso “Paciente” en el que se deben registrar los datos personales y clínicos del paciente. Identificando estos casos de uso, se diseña un caso de uso “Suscribir” como se muestra en la figura 5.28.

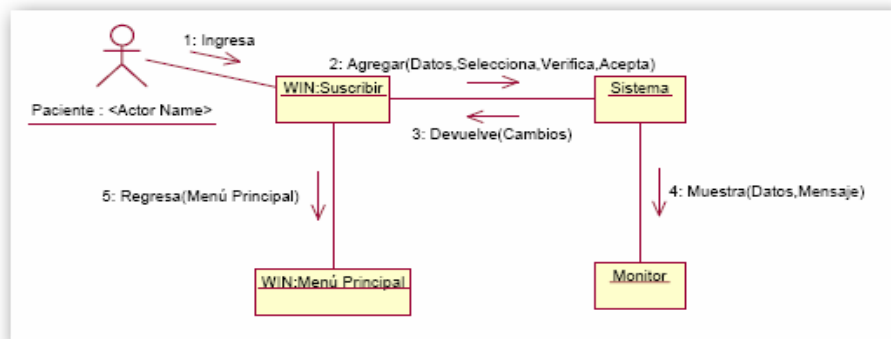
Figura 5.28 Caso de uso suscribir.



Fuente: Elaboración propia

Se elabora entonces el diagrama de colaboración entre el usuario y el sistema al momento de realizar una suscripción como se muestra en la figura 5.29.

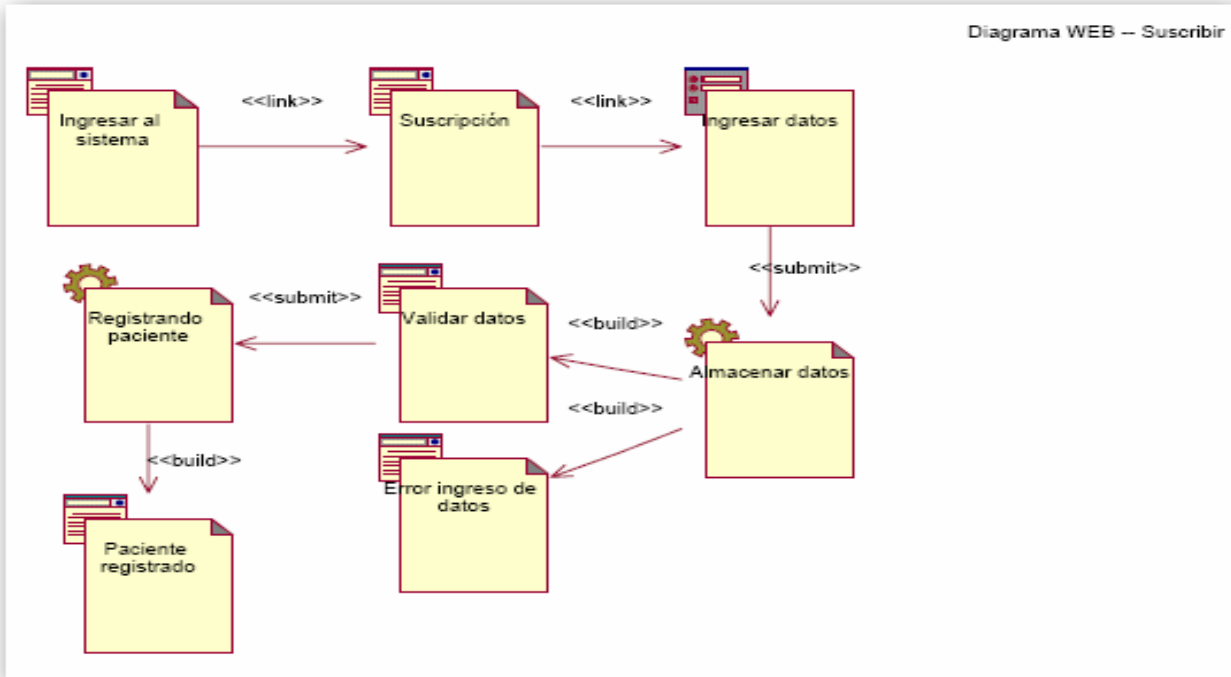
Figura 5.29 Diagrama de caso de uso Suscribir.



Fuente: Elaboración propia.

El diagrama Web que muestra el servicio de suscripción al sistema se muestra en la figura 5.30

Figura 5.30 Diagrama Web Suscribirse.



Fuente: Elaboración propia.

De esta manera, y teniendo como punto de partida los requisitos de negocios, se construye el modelo de negocios y el modelo del sistema, y son las especificaciones hechas en este último las que sirven de base para la construcción del modelo Web.

Capítulo 6

6.1 Introducción al modelo centrado en arquitectura de procesos.

Este capítulo define un enfoque metódico, llamado “Modelo centrado en la arquitectura de procesos (MAP)”, para enfrentarse de manera eficaz con la complejidad del diseño de arquitectura y la administración de la arquitectura de procesos y el ciclo de vida del desarrollo de los sistemas de información en un paradigma orientados a servicios. Este método comprensivo, cuenta con 4 dimensiones de actividades de la arquitectura: análisis de requisitos, especificación, validación y planeación (RSVP). El proceso se desglosa en 9 modelos: meta arquitectura, arquitectura conceptual, arquitectura lógica, arquitectura física, despliegue de la arquitectura, arquitectura de gestión/administración, arquitectura de la información, aspecto de la arquitectura, arquitectura de componentes. Una matriz 2-D sirve como un plan que muestra paso a paso los procedimientos que producen y administran los artefactos de la arquitectura y delibera el ciclo de vida del diseño de la arquitectura de los sistemas. El desarrollo y la administración. El marco holístico provee una vista multidisciplinaria de los principios de diseño, el inicio, el idioma y las estrategias de la arquitectura práctica de TI. Las características y mejoras de los elementos que constituyen el enfoque MAP son articuladas a gran detalle. Las recomendaciones y las tendencias futuras son solo presentadas en el contexto. Esto ayuda a crear un servicio de alta calidad orientado a las soluciones enfocadas en diferentes dominios y mientras tanto mantiene la agilidad, flexibilidad y adaptabilidad del método global. Este marco sistemático puede ser personalizado en diferentes formatos al diseñar varios sistemas de información en diferentes industrias.

En la actualidad, en la demanda mundial de negocios, los modelos de negocios electrónicos demandan cada vez más un mayor rendimiento de los sistemas de la tecnología de la información. Nosotros debemos hacer más con menos, así como proveer un mayor nivel en los servicios a un bajo costo para que los negocios puedan competir y tener éxito. Esto significa que las Tecnologías de la Información (TI) tuvieron una construcción muy compleja, flexible,

escalable, extensible y con ideas técnicas de solución, y cumplir siempre con las necesidades de los negocios.

En las grandes organizaciones como en las instituciones financieras en todo el mundo, prácticamente cientos, si no es que miles de aplicaciones TI y sistemas tienen que ser construidos o comprados para proveer servicios electrónicos para clientes externos e internamente empleados de los años pasados, utilizando tecnologías heterogéneas y arquitecturas que satisfagan la diversidad de requisitos funcionales de diferentes líneas de negocios. La industria de la banca no es la excepción. Las operaciones de los procesos de negocios generalmente contienen diferentes sectores de negocios al por menor, comercial, pequeños negocios, administración de patrimonio y administración de capital. En particular, los servicios son entregados en diferentes canales así como cajeros automáticos (ATM), buscadores Web, respuesta de voz interactiva, correo electrónico, asistencia por medio de agentes y dispositivos móviles. Administrar eficazmente la complejidad de la arquitectura, optimizar las prácticas de diseño en la disparidad del medio ambiente y un diseño multidisciplinario aprobado es crucial para disminuir preocupaciones, dividir responsabilidades, eliminar riesgos, encapsular la complejidad y racionalizar los procesos operativos.

6.2 Antecedentes.

Estudios previos en las décadas pasadas se esforzaron en dirigir el tema de la complejidad del diseño de la arquitectura, el cual ha crecido exponencialmente así como el paradigma de computadoras tuvo que envolverse en un servicio monolítico orientado a la arquitectura. El marco de Zachman (Zachman, 1987) es una estructura lógica para clasificar y organizar las representaciones descriptivas de una empresa de TI y su medio ambiente; importantes en la administración de la organización así como en el desarrollo de los sistemas de información de las empresas. Esto toma la forma de la matriz bidimensional (2-D), lo que ha logrado un nivel de penetración en el dominio de los negocios y la arquitectura de los sistemas de información y el modelado. Esto se utiliza principalmente como una planeación o una herramienta para resolver problemas. Sin embargo, esto tiende a alinearse implícitamente con un enfoque impulsado por los datos y los métodos de descomposición de procesos, esto opera por encima y a través del nivel de

proyectos individuales. El marco de la arquitectura de empresa extendidas (E2AF) (IEAD, 2004), toma un enfoque muy similar en el marco de Zachman. Su ámbito de aplicación incluye empresas, información, sistemas e infraestructura en una matriz bidimensional (2-D). La E2AF esta más orientada a la tecnología. Algunos de estos alcances son metodologías muy pesadas, las cuales requieren de aprendizaje de pasos muy largos para poderse iniciar.

El Proceso Unificado de Rational (RUP) (Krutchen, 2003) trata de superar estas deficiencias mediante la aplicación del lenguaje de modelado unificado (UML) en el manejo de casos de uso, orientación a objetos y enfoque basado en componentes. El concepto de vistas 4+1 interpreta la estructura del sistema global desde múltiples perspectivas. RUP esta más orientado a procesos, y enraizado en una cascada como enfoque. RUP apenas dirige el mantenimiento del software y las operaciones y carece de cobertura en profundidad en la topología física y en el desarrollo de herramientas de pruebas. Esto principalmente opera los niveles de cada proyecto. RUP recientemente ha sido extendido a Procesos Unificados de Empresas (EUP), de las cuales algunas se han convertido en código abierto, como por ejemplo: OpenUP en el proyecto del marco de procesos de Eclipse (EPF) (Eclipse, 2007).

En otro enfoque avanzado, el grupo abierto del marco arquitectónico (“The Open Group Architectural Framework”, TOGAF) (The Open Group, 2007), es un marco detallado con un conjunto de herramientas de soporte para desarrollar una arquitectura de empresas para cumplir con los negocios y las necesidades de información tecnológica de una organización. Las tres partes del núcleo de TOGAF son el método de desarrollo de la arquitectura (ADM), la arquitectura continua de la empresa y los recursos base de TOGAF. El alcance de TOGAF cubre la arquitectura de procesos de negocio, la arquitectura de las aplicaciones, la arquitectura de la información y la arquitectura de la tecnología. El punto de enfoque de TOGAF es el nivel de arquitectura de las empresas, en lugar del nivel de aplicación de cada arquitectura. De otra manera, el modelo impulsado por la arquitectura (MDA) (OMG, 2007) tiene un enfoque ágil. El objetivo de MDA es separar la lógica de negocios o la lógica de la aplicación de la plataforma de la tecnología subyacente. Lo fundamental de MDA es el modelo de plataforma independiente (PIM), el cual provee gran portabilidad e interoperabilidad así como el aumento de la

productividad y el mantenimiento. MDA es primeramente para la parte de modelado de software en el desarrollo del ciclo de vida de los procesos.

Otros trabajos relacionados en el marco de la arquitectura de TI son demasiado medidos en un dominio particular. Ellos usan referencias cuando un equipo crea sus propios modelos para sus organizaciones. El marco de la arquitectura C4ISR (DoD, 1997) ofrece una guía de la arquitectura comprensiva para varios comandos, servicios y agencias dentro del departamento de defensa de los Estados Unidos de Norteamérica, con el fin de garantizar la interoperabilidad y un costo efectivo de los sistemas militares. El marco de la arquitectura de empresas federales (FEA) (Federal Office of Management and Budget, 2007) provee dirección y guía a las agencias federales de Estados Unidos de Norteamérica para estructurar la arquitectura de las empresas. El marco de la arquitectura de las empresas de tesorería (TEAF) (Treasury department, 2000) es la guía de la planeación y del desarrollo de la arquitectura de las empresas en todas las oficinas del departamento de tesorería. La arquitectura de referencias de la empresa Purdue (PERA) (Purdue University, 1989) esta alineada al equipo integrado de fabricación.

ISO/IEC 14252 (IEEE Standard 1003.0) es una arquitectura de un marco construido en sistemas estándar abiertos POSIX. El modelo de referencia ISO para procesamiento distribuido abierto (RM-ODP) (Putman, 2001) es una coordinación de un marco para la estandarización de procesamiento distribuido abierto en ambientes heterogéneos. Esto crea una arquitectura que integra un soporte de distribución, trabajo interno y portabilidad, usando cinco puntos de vista y ocho transparencias. El método de análisis de arquitectura basado en escenarios (SAAM) (SEI, 2007) fue desarrollado al analizar un sistema modificable pero es utilizado para probar algún aspecto no funcional. Las arquitecturas son examinadas vía escenarios en SAAM con respecto a la consecución de calidad de los atributos. La arquitectura de solución de aplicaciones N-Tier (Shan & Hua, 2006) presenta un modelo multi-capa y multi-pilar para aplicaciones basadas en la Web.

A pesar del uso esporádico de los métodos previamente discutidos, en el mundo real actual las practicas en los sistemas de información desarrollados siguen siendo ad hoc, manuales y propensas a errores. Este tipo de madurez inevitable conduce a resultados caóticos y fallas en la

ejecución del proyecto. Una reciente encuesta (Standish Group, 2007) revela que una gran mayoría de proyectos de sistemas de información son retrasados, por arriba del presupuesto o cancelados. La inadecuación sistemática del marco describió las prácticas claves del diseño y los artefactos en el paradigma orientado a servicios son directamente atribuidos a esta situación. Un nuevo marco se propone en la siguiente sección con descripciones mas detalladas, características claves y las características de los componentes. El uso de mejores prácticas y las tendencias futuras son presentados en las subsecuentes secciones.

6.3 Método exhaustivo.

Como se vio en la sección anterior, la mayoría de los métodos revelan los aspectos arquitectónicos de una aplicación de software en cierta medida en un nivel más alto o desde una perspectiva restringida. La necesidad de un enfoque exhaustivo de una arquitectura extremo a extremo para soluciones distribuidas es cada vez más evidente, demandando una forma de disciplina sistemática. Un mecanismo estructurado avanzado se diseña también en este capítulo para cumplir siempre con las necesidades y presenta una vista exhaustiva y holística de todos los elementos de la arquitectura, componentes, conocimiento, plataformas, planeación y su relación entre ellos. El diseño de los procedimientos está establecido de acuerdo al enfoque para facilitar la creación, organización y administración de los activos de la arquitectura y las soluciones en diferentes niveles en una gran organización.

6.4 Filosofía de diseño.

El desarrollo de este mecanismo es disciplinado, siguiendo algunos principios de diseño claves, parte de los cuales están adaptados de TOGAF (The open Group, 2007) pero significativamente modificados/extendidos al ser adaptados al desarrollo de aplicaciones de procesos distribuidos orientados a servicios.

6.4.1 Principios Empresariales.

- **Primicia de los principios:** Todos los grupos y las partes interesadas dentro de una organización deben seguir estos principios de solución de arquitecturas.

- **Maximizar beneficios:** La maximización de los beneficios se obtendrá de toda la organización.
- **Continuidad de los negocios:** Las operaciones del negocio no serán interrumpidos pese a los cambios del sistema.
- **Compromiso activo:** Todos los interesados que participan en el proceso deben cumplir los objetivos del negocio.
- **Conformidad con las regulaciones:** La arquitectura de los procesos cumplirá con lo relevante como la ley, las políticas y las regulaciones.
- **Rendición de cuentas de las Tecnologías de la Información (TI):** Las cuentas del grupo de TI para poseer e implementar los procesos TI y la infraestructura que activa las soluciones son para satisfacer los requisitos del negocio para la funcionalidad, niveles de servicio, costos y los plazos de entrega.
- **Innovaciones:** La simulación y la protección de las simulaciones corporativas se aplican en la arquitectura de TI, la administración y la guía de los procesos.

6.4.2 *Principios Técnicos.*

- **Flexibilidad:** El modelo técnico es ágil al ser adaptado en respuesta a las necesidades futuras del negocio.
- **Responder al cambio de administración:** Cambiar la arquitectura/infraestructura del ambiente de la corporación esta planeado e implementado en un enfoque gradual.
- **Alcance de control de requisitos:** Evitar el arrastre del enfoque y el alcance de cascada.
- **Estandarización de la tecnología:** La diversidad tecnológica es controlada al minimizar la inmadurez, las soluciones patentadas y los productos.
- **Interoperabilidad:** El software, el Hardware, la red y la infraestructura deben cumplir con los estándares definidos que promuevan la compatibilidad para los datos, las aplicaciones, los servicios, las comunicaciones, la integración, la seguridad y la tecnología.

6.4.3 *Principios de solución.*

- **Facilidad de uso:** Las soluciones son amigables para los usuarios, con tecnología transparente para los usuarios, en las que puedan concentrarse en tareas a mano.

- **Independencia tecnológica:** Las soluciones técnicas son independientes de la especificación de tecnología elegida y también puede operar con una variedad de plataformas de tecnología.
- **Servicios y componentes comunes:** Minimizar la redundancia en el desarrollo de funcionalidades similares para promover servicios y componentes comunes a través de la organización.

6.4.4 *Principios de los datos.*

- **Recursos de datos:** Un dato es un recurso que le da valor a la empresa y acorde a esto es administrado.
- **Datos de propiedad:** Cada elemento de datos de propiedad por una entidad da cuentas para la calidad de los datos.
- **Meta-datos y vocabulario común:** Un dato es definido consistentemente en toda la organización, y el meta-dato esta estandarizado y disponible para todos los usuarios.
- **Datos compartidos:** Un dato es compartido por líneas de negocio para aplicaciones individuales y sistemas que cumplen su deber.
- **Acceso a datos:** Un dato es accesible para usuarios que cumplen sus funciones.
- **Seguridad de datos:** Un dato es protegido de usos no autorizados y rebeldías. En adición a aspectos tradicionales de la clasificación de seguridad nacional, esto se incluye, pero no es limitado a la protección sensitiva y a la información personal.

6.4.5 *Principios de servicio.*

- **Encapsulación:** Los elementos de servicio y los detalles están encerrados dentro de entidades abstractas.
- **Perdida de acoplamiento:** Los servicios mantienen una relación que minimiza las dependencias y solo requiere que estos mantengan una conciencia de los otros.
- **Contrato:** Mas allá de lo que se describe en el contrato de servicio, los servicios ocultan la lógica del mundo exterior.
- **Reusabilidad:** La lógica se divide dentro de los servicios como la intención de promover el reuso.

- **Componibilidad:** Las colecciones de servicios pueden estar coordinadas y montadas en forma de servicios compuestos.
- **Autonomía:** Los servicios tienen un completo control sobre la lógica de encapsulamiento.
- **Sin estado:** Los servicios minimizan la retención de la información específica de una actividad.
- **Descubrimiento:** Los servicios están diseñados para ser exteriormente descriptivos, así que ellos pueden ser encontrados, evaluados y están disponibles a través de mecanismos de descubrimiento.

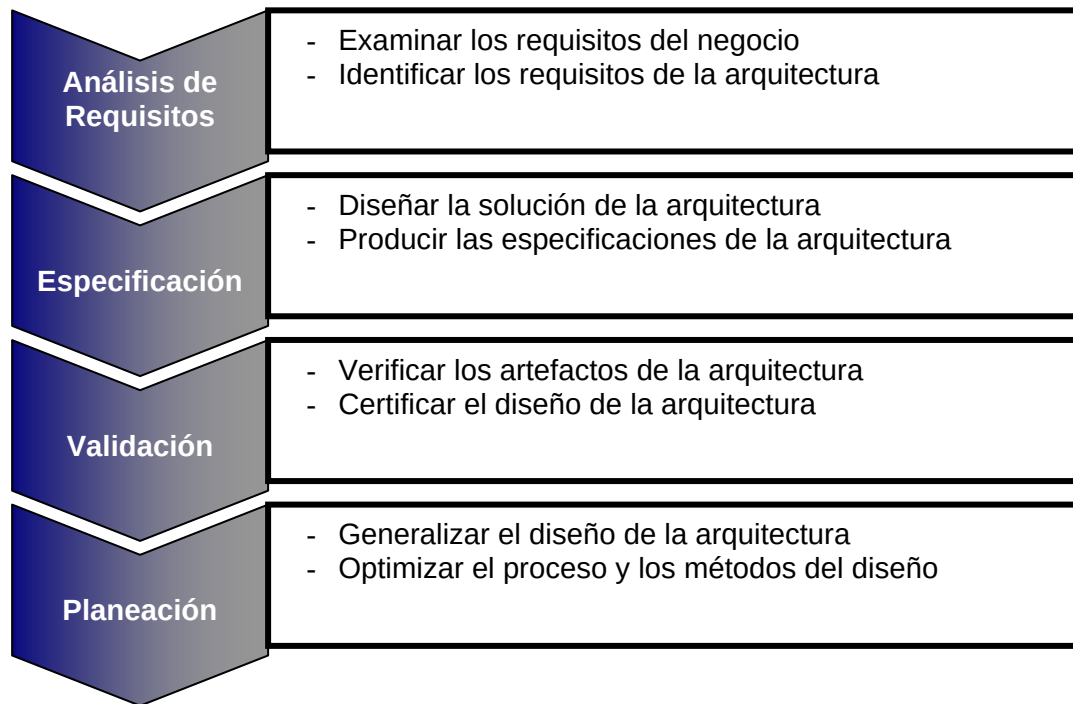
6.5 Planteamiento general.

El Modelo centrado en Arquitectura de Procesos (MAP) es diseñado en este trabajo como un planteamiento multidisciplinario. Se define como un método exhaustivo al control del diseño de la aplicación y al desarrollo de las prácticas para una entrega de calidad de sistemas de información. El marco MAP es una disciplina holística que ayuda a realizar una estrategia y a analizar el concepto de los procesos, métodos, compensaciones y patrones en el diseño de la arquitectura.

Como se ilustra en la figura 6.1, hay cuatro dimensiones definidas en MAP: análisis de requisitos, especificación, validación y planeación (RSVP), las cuales se alinean con las mejores prácticas en sistemas y en la ingeniería de software. El análisis de requisitos examina a fondo los requisitos del negocio e identifica los requisitos de la arquitectura. La dimensión de especificación trata con el diseño de la arquitectura basada en la salida de los resultados del análisis y la producción de las especificaciones de la arquitectura. La dimensión de validación es para la verificación objetiva/subjetiva y la certificación del diseño de la arquitectura terminada. Por ultimo, el enfoque de la dimensión de planeación es para la arquitectura de aplicación individual y la arquitectura del portafolio. La granularidad varía en la empresa, división, canal, dominio de negocios y en los niveles del proyecto. Los puntos de referencia son usados para medir y rastrear la madurez de los modelos de arquitectura. La planeación enfatiza la reusabilidad de la arquitectura y la generalización, la cual conduce a la arquitectura de los estilos y los patrones para

un entorno informático empresarial. Las dimensiones de RSVP representan los estados mayores en el ciclo de vida de la arquitectura de solución de diseños.

Figura 6.1. Dimensiones RSVP de MAP.



Fuente: Shan y Hua (2008).

MAP forma una base común al implementar la arquitectura orientada a servicios, la integración, los procesos y la administración de los sistemas de información. Esto enfocado primeramente en un rango amplio de modelos de arquitectura como lo ilustra la figura 6.2: meta arquitectura, arquitectura conceptual, arquitectura lógica, arquitectura física, despliegue de la arquitectura, arquitectura de gestión/administración, arquitectura de la información, aspecto de la arquitectura, arquitectura de componentes.

Meta arquitectura.- La organización fundamental y la descripción formal de un sistema de portafolio de TI, interrelaciones, decisiones y sus racionalizaciones asociadas.

Arquitectura conceptual.- El agrupamiento de las capacidades y las prácticas que hacen uso del conceptualismo en la arquitectura.

Arquitectura lógica.- La estructura estática y dinámica de la capa, los módulos y los bloques construidos que están mapeados en el servidor y las plataformas de tecnología, los cuales son los contenedores base para las aplicaciones.

Arquitectura física.- El entorno del tiempo de ejecución que esta mapeado en los productos de software y hardware en los que las aplicaciones son instaladas y ejecutan sus operaciones.

Despliegue de la arquitectura.- El diseño topológico de sistemas de solución que serán desplegados en diferentes locaciones geográficas con conexión de red.

Arquitectura de gestión/administración.- La arquitectura y la infraestructura para gestionar, guiar y administrar los recursos de TI y de una empresa están distribuidos en un sistema de computadoras.

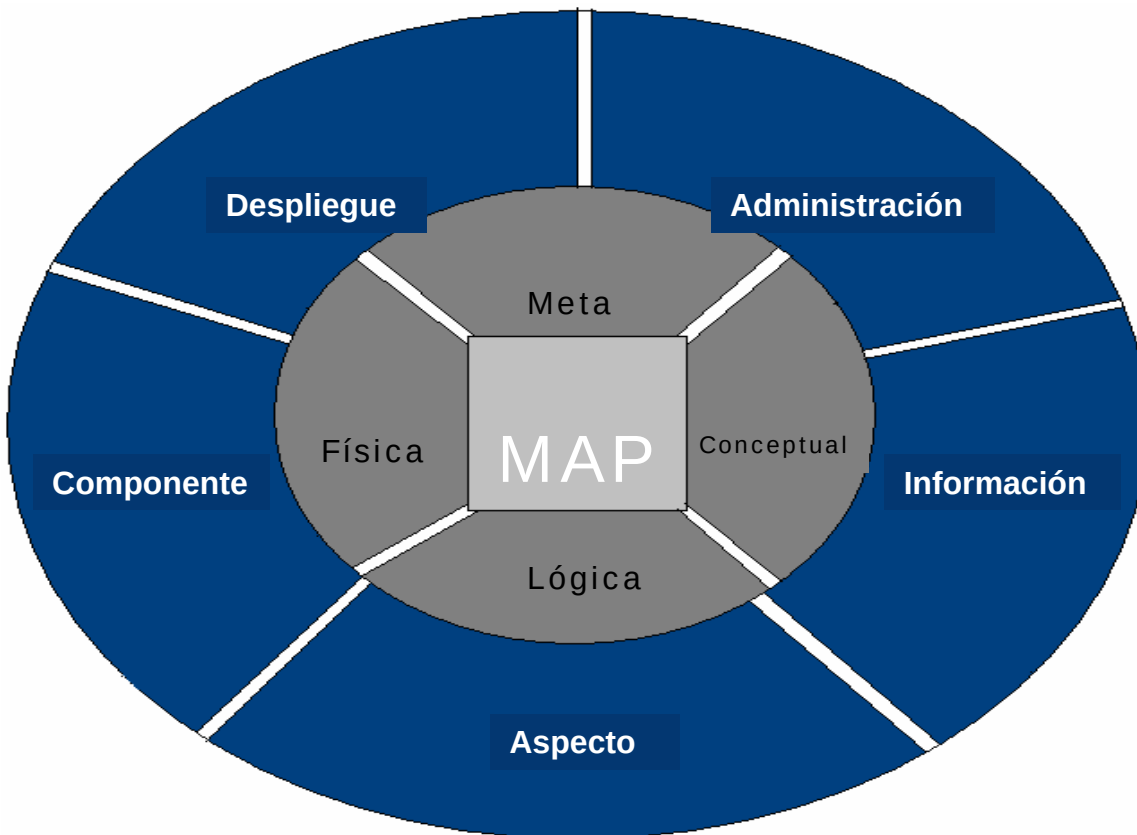
Arquitectura de la información.- La practica de procesamiento, almacenamiento y utilización de la información (datos o conocimientos), los cuales están a menudo estructurados en relación con este contexto en interacciones con usuarios y grandes bases de datos.

Aspecto de la arquitectura.- La organización de la arquitectura, el control operacional y los aspectos de desarrollo que son transversales y asesorados.

Arquitectura de componentes.- Una evolución de diseño orientado a objetos donde el componente de un programa se refiere a un modulo genérico compuesto de un conjunto de elementos u objetos. La granularidad de varios componentes, siendo primitivo, agregado o compuesto. La estructura de un componente puede ser demasiado complejo ya

que el mismo componente necesita un único marco para organizar los artefactos del diseño.

Figura 6.2 Niveles de arquitectura MAP



Fuente: Shan y Hua (2008).

6.5.1 *Modelo centrado en arquitectura de procesos (MAP).*

MAP introduce el concepto y aplicación de nuevas disciplinas de modelado, e intensifica una atención sin precedentes a la clave de atributos de diseño para el paradigma orientado a servicios. La arquitectura de despliegue se distingue por la arquitectura física, con el fin de separar aún más las preocupaciones y mejorar la falta de acoplamiento de los modelos. La gestión de la arquitectura se convierte en un modulo independiente tomado en cuenta de la creciente demanda para la calidad de los servicios y los requisitos no funcionales. Los aspectos y los componentes recabados a un nivel de la arquitectura dan un punto de vista holístico de las claves de la arquitectura de diseño de los artefactos de punta a punta, eliminando la desconexión entre el diseño de arquitectura y el diseño de implementación en el enfoque tradicional. Además, el meta

modelo da luz al plan, se refiere a la aplicación de cruz, la hoja de ruta y la optimización a fin de allanar el camino hacia el servicio orientado a las empresas.

6.5.2 *Meta arquitectura.*

Los requisitos de esta arquitectura son la visión, los principios, dogmas, idiomas, dirección, y las tendencias. Esto incluye el alineamiento de los negocios y la tecnología de modelos, así como la racionalización de la carpeta de arquitectura. La meta arquitectura especifica un numero de artefactos clave de la arquitectura en la aplicación de carpeta: la misión y visión de la arquitectura, la estrategia, el mapa del camino, los métodos, las herramientas, el marco, el modelo de arquitectura de aplicación cruzada, los patrones de la arquitectura, los patrones de análisis del negocio y los estándares. En el estado de validación de la meta arquitectura, los estándares de la empresa y las políticas son establecidos, acoplándose con la referencia del modelo y los patrones de solución. La planificación de la meta arquitectura enfatiza una estrategia, la arquitectura dominio/canal, optimización de la arquitectura, la arquitectura del cianotipo, la arquitectura taxonómica por clasificación y ontología por semántica.

Figura 6.3 Meta Arquitectura en MAP



Fuente: Shan y Hua (2008).

6.5.3 *Arquitectura conceptual.*

Esta arquitectura trata el modelado de las funcionalidades del sistema, identificando los requisitos no funcionales, determinando de acuerdo al nivel de servicio, analizando los procesos de negocios, y alcanzando las capacidades. Lo que realiza la arquitectura conceptual son el diseño de los diagramas de casos de uso, la especificación de los casos de uso, los alcances del sistema, la contención y suposición de la arquitectura, los riesgos de la arquitectura, la especificación de la interfaz de usuarios, y la especificación de los servicios de acceso (API y servicio de interfaz).

La validación de la arquitectura conceptual, los patrones de negocio y el esqueleto de la arquitectura son evaluados por la conducta, el impacto y la agilidad del análisis. Los estándares de la industria son verificados y las notaciones son estandarizadas. Con respecto a la planificación, la arquitectura conceptual se reparte con la arquitectura de los procesos, contenido, contexto, reuso y generalización.

Figura 6.4 Arquitectura Conceptual en MAP.



Fuente: Shan y Hua (2008).

6.5.4 *Arquitectura lógica.*

La arquitectura lógica considera lo que engloba la estructura del sistema, opciones de tecnología y el criterio de selección, modulando la organización y aplicación del dominio a modelar. En la arquitectura lógica, el diseño es especificado por los diagramas de arquitectura, las responsabilidades del subsistema y sus relaciones, los diagramas de comunicación, los diagramas de actividades, la selección de la tecnología clave, la capa de la arquitectura y la interoperabilidad.

La validación de la arquitectura lógica esta en los patrones de la arquitectura, reglas del negocio, workflow, patrones de integración y modelos de referencia. Desde el punto de vista de la planificación de la arquitectura, esta se debe alinear estrechamente con la arquitectura de los procesos de negocios. El modelo podría ser una plataforma independiente y un producto agnóstico, permitiendo la apertura y flexibilidad a largo plazo y estabilizar el diseño.

Figura 6.5 Arquitectura Lógica de MAP



Fuente: Shan y Hua (2008).

6.5.5 *Arquitectura física.*

Esta arquitectura maneja la plataforma de criterio de selección, acercamiento al trazado del producto, capacidad de planificación, y el análisis del atributo de calidad. Esta arquitectura captura la selección de la plataforma, el modelo de hardware, la gestión del sistema, el retraso del sistema y la recuperación de desastres, la escalabilidad, alta disponibilidad, autenticación y autorización, seguridad, y requisitos de red.

Los requisitos regulatorios y de complacencia son evaluados. Cuando la arquitectura dura mucho tiempo cuantifica la evolución del diseño de arquitectura física.

Figura 6.6. Arquitectura Física en MAP



Fuente: Shan y Hua (2008).

6.5.6 *Despliegue de la arquitectura.*

Los requisitos que alcanza esta arquitectura contienen el ambiente en tiempo de ejecución del anfitrión de las aplicaciones individuales con la organización topológica en los centros de datos geográficos. Los elementos clave en esta arquitectura son la topología de red, explotación o

alojamiento, modelos de construcción desplegable, especificación de configuración, diagramas de despliegue, especificación de instalación, guía de usuarios, y centro de operaciones de datos.

En esta arquitectura, los patrones de despliegue podrían ser considerados como criterios de aplicación de ingeniería. Otros artefactos también son examinados, así como la conformidad, SSO, autorización, encriptación, y búsqueda de recursos vía servidor de alojamiento y particionamiento.

Figura 6.7 Despliegue de la arquitectura en MAP



Fuente: Shan y Hua (2008).

6.5.7 *Arquitectura de gestión /administración.*

Los requisitos de esta arquitectura son provisionamiento de la solución, automatización, capacidad, servicio de versiones, seguridad de servicio de interacción, y monitoreo de los acuerdos de nivel de servicio (SLA). Las especificaciones que cubre esta arquitectura son cubrir el manejo de eventos, corregir fallas, diseño del producto, poner a punto el traslado de los datos y su análisis, problemas de aislamiento y diagnóstico, servicios de visualización de relaciones y estado de los procesos del negocio.

Los artefactos claves son examinados como una colección de datos, análisis, persistencia, visualización, optimización del ambiente, políticas de gobierno, certificación y registro. Esta arquitectura planifica el enfoque que engloba la arquitectura de gestión, servicio de repositorio, plan de recuperación del sistema, salpicadero, y proceso de gobierno.

Figura 6.8 Gestión de la Arquitectura en MAP



Fuente: Shan y Hua (2008).

6.5.8 *Arquitectura de la información.*

Esta arquitectura maneja los requisitos de datos y el contenido, así como la persistencia, los reportes, la integración de datos, la calidad de los datos, análisis, la inteligencia de los negocios, la provisión de datos, y transformar la carga de un extracto de datos. La especificación de esta arquitectura define el modelo de datos, el almacén, DBMS, el esquema de la base de datos, la replicación, la transferencia de datos, el respaldo de datos, el almacenamiento de datos, los reportes, meta datos, las consultas, la prioridad de trabajo y el control de acceso.

La validación de esta arquitectura consiste en la evaluación de la normalización de los datos, seguridad, integridad, la transacción contra la analítica, reuso, replicación, calidad, análisis, el caudal, y la auditoría. La planificación tiene que ver con la referencia de los datos, la integración y virtualización de los datos el conocimiento de la ingeniería, la retención y alcance de los datos.

Figura 6.9 Arquitectura de la Información en MAP



Fuente: Shan y Hua (2008).

6.5.9 Aspecto de la arquitectura.

Esta arquitectura trata las preocupaciones transversales en los sistemas utilizando los patrones de la arquitectura. Los aspectos corresponden a la aplicación del marco, los patrones de diseño, los diagramas de secuencia, los diagramas de colaboración, ajuste del rendimiento, la gestión de la sesión, y los productos de código abierto. Los primeros artefactos que validan esta arquitectura son los estándares de la empresa, la complacencia, la dependencia de inyección, las anotaciones, el Proxy, los estándares de la industria y el reuso de los recursos.

Figura 6.10 Aspecto de la Arquitectura en MAP.



Fuente: Shan y Hua (2008).

6.5.10 *Arquitectura de componentes.*

Esta arquitectura es el micro nivel de diseño y construcción en software de aplicación. El análisis de los componentes necesita una descomposición progresiva de los procesos con el método de nivel de ayuda para identificar los componentes reutilizables y las herramientas a utilizar, ya sean comerciales o productos de código abierto.

Esta arquitectura cubre la especificación de componentes, las tarjetas CRC, los diagramas de clases, los patrones de diseño, la tecnología, los paquetes, y utilidades de código abierto. La validación es relacionada a los estándares, la complacencia, los repositorios con recursos de código abierto, las interfaces, la composición de componentes y la orquestación. En esta arquitectura, se enfoca en la jerarquía de componentes, mantenimiento, convenciones, catálogo y repositorio de componentes, y eficiencia de reuso.

Figura 6.11 Arquitectura de Componentes en MAP.



Fuente: Shan y Hua (2008).

	Meta Arquitectura	Arquitectura Conceptual	Arquitectura Lógica	Arquitectura Física	Despliegue de la Arquitectura	Arquitectura de Gestión/Administración	Arquitectura de la Información	Aspecto de la Arquitectura	Arquitectura de Componentes
Requisitos de la arquitectura (Análisis)	- Principios de la Arquitectura - Tendencias de la Arquitectura - Biz y Alineamiento de la Tecnología - Racionalización de la Arquitectura y simplificación	- Modelamiento Funcional - Requisitos No Funcionales - SLA - Procesos del Flujo de Negocios - Sistema de Alcance - Capacidad	- Estructura del Sistema de TI - Opciones y Selección de la Tecnología - Modelado del dominio de la Aplicación - Organización	- Requisitos de la plataforma - Mapeos de Productos - Requisitos de la Red - Análisis de la Calidad de los atributos	- Tiempo de Ejecución del entorno - Escenario - Piloto - Tiempo de recuperación y Objetivos Puntuales - Liberación y Nuevo plan	- Provisión de la Solución - Automatización y capacidad - Versión de Servicios múltiples - Interacción del servicios d - Seguridad - Monitor SLA	- Persistencia de Requisitos - Reportes - ETL - BI - Almacén de datos - Datos mart - Riesgo de datos - OLTP/OLAP	- Preocupaciones Transversales - Aplicación, Software, Hardware, Red y nivel de infraestructura - Aspecto de Patrones	- Diseño detallado/Micro - Componentes de servicio - Operaciones → proceso → Servicio → componentes → O bjetos → Descompos ición de Método
Especificación de la arquitectura (Modelado)	- Arquitectura cross-app - Hoja de ruta - Patrones de la Arquitectura - Patrones de análisis de negocios - Métodos de la Arquitectura - Estándares de la Arquitectura - Marcos de la Arquitectura - Herramientas de modelado	- Especificación y Diagramas de Caso de Uso - Contexto del Sistema - Limitaciones y Supuestos de la Arquitectura - Riesgos de la Arquitectura - Especificación de la Interfaz de Usuario - Especificación del Servicio de Acceso - Cálculos de Independencia	- Diagrama de Arquitectura - Responsabilidades del subsistema y sus Relaciones - Diagrama de Comunicación - Diagrama de Actividades - Selección de la Tecnología Clave - Capas de la Arquitectura - Interoperabilidad	- Selección de la plataforma - Modelo de Hardware - Rendimiento del sistema - Respaldo del sistema y DR - Escalabilidad - Alta disponibilidad - Autenticación - Seguridad - Capacidad de planeación	- Topología de Red - Farm/Hotel - Construcción de modelos - Especificación de configuración - Entornos DEV, TEST, UAT, PROD - Diagrama de Despliegue - Especificación de la instalación - Guía de Usuario - DMZ y Corta Fuegos	- Modelamiento Funcional - Manipulación del caso y solución de problemas - Rastreo del producto y depuración de análisis y transferencia de datos - Aislamiento del problema y diagnostico - Visualización de las relaciones de servicio - Estados del proceso de negocios	- Modelo de datos - Almacenamiento - DBMS - Esquema de la Base de Datos - Replicación - Transferencia de datos - Respaldo de datos - Cacheo de Datos - Reportes - Meta datos - Consultas - Prioridad de carga de trabajo - Control de acceso	- Marcos de Aplicación - Patrones de diseño - Diagrama de secuencia - Diagrama de colaboración - Ajustes de rendimiento - Administración de sesión - Productos de código abierto - Sistemas y navegación UI	- Especificación de componente - Definición de servicio - Tarjetas CRC - Diagramas de clase - Patrones de diseño - Tecnología - Paquetes - Utilidades de código abierto - Librerías - Widgets - Kit de herramientas - API
Validación de la arquitectura (Dirección)	- Evaluación - Prototipo y POC - Modelos de Referencia - Patrones de Solución - Directrices de Buenas-Prácticas - Recetarios - Estándares - Puntos de Referencia	- Evaluación - Análisis de Impacto - Esqueleto de la Arquitectura - Patrones de Negocio - Estándares de la Industria - Estándares de la Notación - Análisis de Rapidez	- Revisión - Patrones de la Arquitectura - Reglas del Negocio - Flujo de Trabajo - Integración de Patrones - SOA, BPM, ESB, WS - Coreografía del proceso Biz	- Evaluación - Estándares de la Empresa - Conformidad - Racionalizar excepciones - Modelos de Referencia - Requisitos Regulatorios - Construcción vs Compra	- Examinar - Patrones de Despliegue - Estándares de la Empresa - Conformidad - SSO y Autorización - Encriptación - Compartir Recursos	- Inspección - Colección de datos - Análisis - Persistencia - Visualización - Optimización del entorno - Guía de las políticas - Certificación - Registro	- Normalización - Seguridad de datos - Integridad de datos - Transaccional vs Analítica - Reuso de datos - Replicación de datos - Calidad de los datos - Análisis de los datos - Flujo de datos y auditoria de flujo	- Justificación - Estándares de la empresa - Conformidad de Inyección de dependencia - Anotaciones - Proxy - Estándares de la industria - Activos Reusable	- Revisión - Estándares de la empresa - Conformidad - Repositorio de valor de código abierto - Interfaz - Composición de componentes - Orquestación de componentes
Planeación de la arquitectura (Visión)	- Estrategia - Dominio/Canal de la Arquitectura - Optimización - Plan - Taxonomía - Ontología	- Proceso - Contenido - Contexto - Reuso de la Arquitectura - Generalización de la Arquitectura	- Alineamiento del modelo con la arquitectura del proceso Biz - Plataforma independiente y Tecnología agnóstica	- Ciclo de Vida del producto - Hoja de Ruta del producto - Madurez de la Arquitectura - Longevidad	- Planeación del centro de Datos - Continuidad del Negocio - Recuperación de la Tecnología - Virtualización	- Administración del Sistema - Repositorio de servicios - Plan de recuperación del sistema - Desbordamiento	- Datos de Referencia - Integración de información y Virtualización - Conocimiento de Ingeniería - Retención de datos y archivado	- Preocupaciones comunes - Patrones - Métodos de diseño orientados a servicios - Inversión de control	- Jerarquía de componentes - Mantenimiento-habilidad - Convenciones de componentes y repositorio
RSVP	Nivel de Portafolio	Nivel del Sistema					Nivel de Software		

Tabla 6-1. Tarjeta de Referencia del Modelo centrado en Arquitectura de Procesos

ARQUITECTURA	FUNCIONALIDAD	REQUISITOS
META	La visión, los principios, dogmas, idiomas, dirección, y las tendencias. Esto incluye el alineamiento de los negocios y la tecnología de modelos, así como la racionalización de carpeta de arquitectura.	La misión y visión de la arquitectura, la estrategia, el mapa del camino, los métodos, las herramientas, los marcos, el modelo de arquitectura de aplicación cruzado, los patrones de la arquitectura, los patrones de análisis del negocio y los estándares.
CONCEPTUAL	Las funcionalidades del sistema, identificando los requisitos no funcionales, determinando de acuerdo al nivel de servicio, analizando los procesos de negocios, y alcanzando las capacidades.	El diseño de los diagramas de casos de uso, la especificación de los casos de uso, los alcances del sistema, la contención y suposición de la arquitectura, los riesgos de la arquitectura, la especificación de la interfaz de usuarios, y la especificación de los servicios de acceso (API y servicio de interfaz).
LÓGICA	Considera lo que engloba estructura del sistema, opciones de tecnología y el criterio de selección, modulando la organización y aplicación del dominio a modelar.	El diseño es especificado por los diagramas de arquitectura, las responsabilidades del subsistema y sus relaciones, los diagramas de comunicación, los diagramas de actividades, la selección de la tecnología clave, la capa de la arquitectura y la interoperabilidad.
FÍSICA	Maneja la plataforma de criterio de selección, acercamiento al trazado del producto, capacidad de planificación, y el análisis del atributo de calidad.	Captura la selección de la plataforma, el modelo de hardware, la gestión del sistema, el retraso del sistema y la recuperación de desastres, la escalabilidad, alta disponibilidad, autenticación y autorización, seguridad, y requisitos de red.
DESPLIEGUE	El ambiente de corre- tiempo al anfitrión de las aplicaciones individuales con la organización topológica en los centros de datos geográficos.	La topología de red, explotación o alojamiento del hotel, modelos de construcción desplegable, especificación de configuración, diagramas de despliegue, especificación de instalación, guía de usuarios, y centro de operaciones de datos.
GESTIÓN	Las especificaciones que cubre esta	Los artefactos claves son

	arquitectura son cubrir el manejo de eventos, corregir fallas, diseño del producto, poner a punto el traslado de los datos y su análisis, problemas de aislamiento y diagnóstico, servicios de visualización de relaciones y estado de los procesos del negocio.	examinados como una colección de datos, análisis, persistencia, visualización, optimización del ambiente, políticas de gobierno, certificación y registro.
INFORMACIÓN	Maneja los requisitos de datos y el contenido, así como la persistencia, los reportes, la integración de datos, la calidad de los datos, análisis, la inteligencia de los negocios, la provisión de datos, y transformar la carga de un extracto de datos.	El modelo de datos, el almacén, DBMS, el esquema de la base de datos, la replicación, la transferencia de datos, el respaldo de datos, el almacenamiento de datos, los reportes, meta datos, las consultas, la prioridad de trabajo y el control de acceso.
ASPECTO	Corresponden a la aplicación del marco, los patrones de diseño, los diagramas de secuencia, los diagramas de colaboración, ajuste del rendimiento, la gestión de la sesión, y los productos de código abierto.	Los primeros artefactos que validan esta arquitectura son los estándares de la empresa, la complacencia, la dependencia de inyección, las anotaciones, el Proxy, los estándares de la industria y el reuso de los recursos.
COMPONENTES	Es el micro nivel de diseño y construcción en software de aplicación. El análisis de los componentes necesita una descomposición progresiva de los procesos con el método de nivel de ayuda e identificar los componentes reutilizables y las herramientas que están apalancadas., ya sean comerciales o productos de código abierto.	Cubre la especificación de componentes, las tarjetas CRC, los diagramas de clases, los patrones de diseño, la tecnología, los paquetes, y utilidades de código abierto. La validación es relacionada a los estándares, la complacencia, los repositorios con recursos de código abierto, las interfaces, la composición de componentes y la orquestación.

6.6 Conclusión.

Para gestionar con eficacia las arquitecturas técnicas complejas y organizar diferentes activos en los sistemas de información, es necesario un método que en resumen permita definir las distintas funciones y responsabilidades, y presentar una visión holística de los aspectos arquitectónicos de un modo estructurado. El modelo centrado en la arquitectura de procesos (MAPA) que se presenta en este capítulo, es un marco global para facilitar la arquitectura de aplicaciones

distribuidas de manera sistemática. Se ofrece una amplia taxonomía de artefactos de la arquitectura, tanto en tiempo de ejecución como en perspectivas de desarrollo. Se basa en la arquitectura de soluciones concretas, las cuales al trabajarse en distintos ámbitos, mantienen la agilidad, flexibilidad y capacidad de adaptación del método global.

Se presento el contexto de los principios de diseño de esta propuesta. El enfoque general comprende un conjunto de múltiples dimensiones: requisitos, especificación, validación y planificación (RSVP). MAPA se compone de una serie de modelos de la arquitectura: la arquitectura de metadatos, la arquitectura conceptual, la arquitectura lógica, la arquitectura física, la arquitectura de despliegue, gestión de la arquitectura, la arquitectura de la información, la arquitectura de aspecto y la arquitectura de componentes. Los elementos de cada módulo se analizan conforme a sus características y se articulan en detalle.

Gracias a su amplitud, este enfoque general se utiliza ampliamente en el desarrollo de aplicaciones orientadas a servicios. Se puede simplificar como una variante desarrollos ágiles adaptados a determinados dominios o proyectos individuales. El enfoque ha tenido éxito en el mundo real de trabajo del proyecto en términos de ahorro de costos, productividad, normalización, y comunicaciones. Además, este método es escalable y flexible para la dinámica de extensiones y ampliaciones, y puede servir como un meta-marco para incorporar a otros marcos generales o especializados.

Capítulo 7

7.1 Resultados.

Con el propósito de aplicar la estrategia propuesta, se eligió como caso de estudio para el modelado el sistema **Teleservices and Remote Medical Care System (TRMCS)**, el cual tiene como objetivo proporcionar ayuda médica, a los pacientes en su domicilio o a pacientes móviles que estén conectados al sistema mediante algún dispositivo de comunicación.

El TRMCS tiene como objetivos particulares:

1. El TRMCS proporcionará dos servicios principales para los pacientes:
 - a. Atender las llamadas de solicitud de servicio emitidas por algún paciente.
 - b. Monitorear de manera continúa la condición de la salud de los pacientes y generar automáticamente una llamada de auxilio cuando sea necesario.
2. Los servicios estarán disponibles sin importar la localización geográfica real del paciente, pero dentro los límites definidos al contratar el servicio.
3. El TRMCS apoyará y coordinará múltiples y proveedores de servicio distribuidos geográficamente.
4. Los servicios proporcionaron por el TRMCS tendrán el mismo nivel de la confiabilidad, seguridad, accesibilidad y ética médica que un servicio proporcionado un medico de manera personal ofrece.

El tipo de usuario al que esta orientado éste sistema, es un paciente que no necesita asistencia continua, pero puede requerir de asistencia prioritaria cuando exista alguna emergencia, en cuyo caso el paciente llamaría a un centro de auxilio para pedir ayuda.

Escenarios del sistema.

Los siguientes escenarios ilustran este nuevo tipo de servicios. Ambos asumen algunos acontecimientos previos como el hecho de que un paciente en el sistema ha visto a un médico de manera previa, se ha aprobado su perfil para poder recibir ayuda en casa y que el centro de auxilio mantiene su historial médico.

Escenario 1. Un paciente con una emergencia médica envía una petición de auxilio a un centro del sistema. El paciente espera recibir la mejor ayuda posible para su caso en un lapso de tiempo crítico. Esta ayuda puede ser un servicio de ambulancia para la emergencia o un diagnóstico en línea con el tratamiento de un profesional de la salud a distancia.

Escenario 2. Un paciente puede tener un conjunto de monitores médicos que dan lecturas continuas sobre una plataforma de Internet, (por ejemplo: presión, pulso). El centro de ayuda por su parte tendrá personal leyendo los monitores en red y los cuales vigilarán si se dispara una alarma cuando se detecten valores peligrosos. En esencia, el software de monitoreo es quien dispara la solicitud de auxilio y no el paciente.

En ambos escenarios, se considera que el paciente estará en su hogar, el cual tiene un acceso “razonable” a Internet. Algunos pacientes, sin embargo, pueden ser viajeros, en este caso el escenario puede implicar ediciones orientadas a cómputo móvil.

Para ganar la aprobación de las agencias reguladoras del gobierno, el centro de ayuda debe demostrar que puede proporcionar la atención correcta en un tiempo correcto. En particular, el software de monitoreo debe demostrar que es exacto en términos de detección y alarma dentro de rangos críticos de tiempo. Debe también demostrar que la disponibilidad del servicio se puede proporcionar en una base 24 horas por 7 días.

Se mantendrá un registro histórico de la información del sistema (por ejemplo fecha, hora, tipo de petición, estado de la petición) el cual puede utilizarse para certificar que el sistema se comportó correctamente en situaciones críticas.

Requerimientos del Sistema.

El propósito del sistema de TRMCS es proporcionar y garantizar los servicios de auxilio a los usuarios móviles o ubicados en su casa. El sistema liberado deberá proporcionar información médica dentro de los estándares y regulaciones requeridos. Esto incluye los estándares de privacidad decretados por el gobierno, y los estándares de la industria desarrollados por los consorcios.

El sistema funcionará sobre la red actual y la infraestructura de cómputo se escalará conforme las nuevas tecnologías se vuelvan aplicables. Por ejemplo, los dispositivos de supervisión médica personales serán compatibles con la red, mediante conexiones alámbricas e inalámbricas. Un centro de la ayuda debe contar con un grupo de servidores y un ancho de banda hacia los dispositivos caseros y móviles.

El sistema debe proporcionar una arquitectura que permita que los nuevos componentes médicos sean incorporados al sistema. De manera semejantemente por el lado de cómputo, el sistema debe soportar un rango de servidores, con una mezcla de los recursos locales (en casa) y remotos (granjas de cómputo).

Los usuarios del sistema incluyen a pacientes, a profesionales del cuidado de la salud encargados de introducir y mantener los datos de los pacientes (médicos y socorristas), a los familiares de los pacientes y a los expertos en el sistema para su mantenimiento encargados de garantizar la calidad del sistema (por ejemplo aprobar las regulaciones federales).

Los profesionales del cuidado de la salud hacen juicios del tratamiento en situaciones de emergencia (por ejemplo llamar a una ambulancia o llamar para establecer una sesión de teleconsulta con un médico que atienda la misma) como se muestra en la figura 7.1.

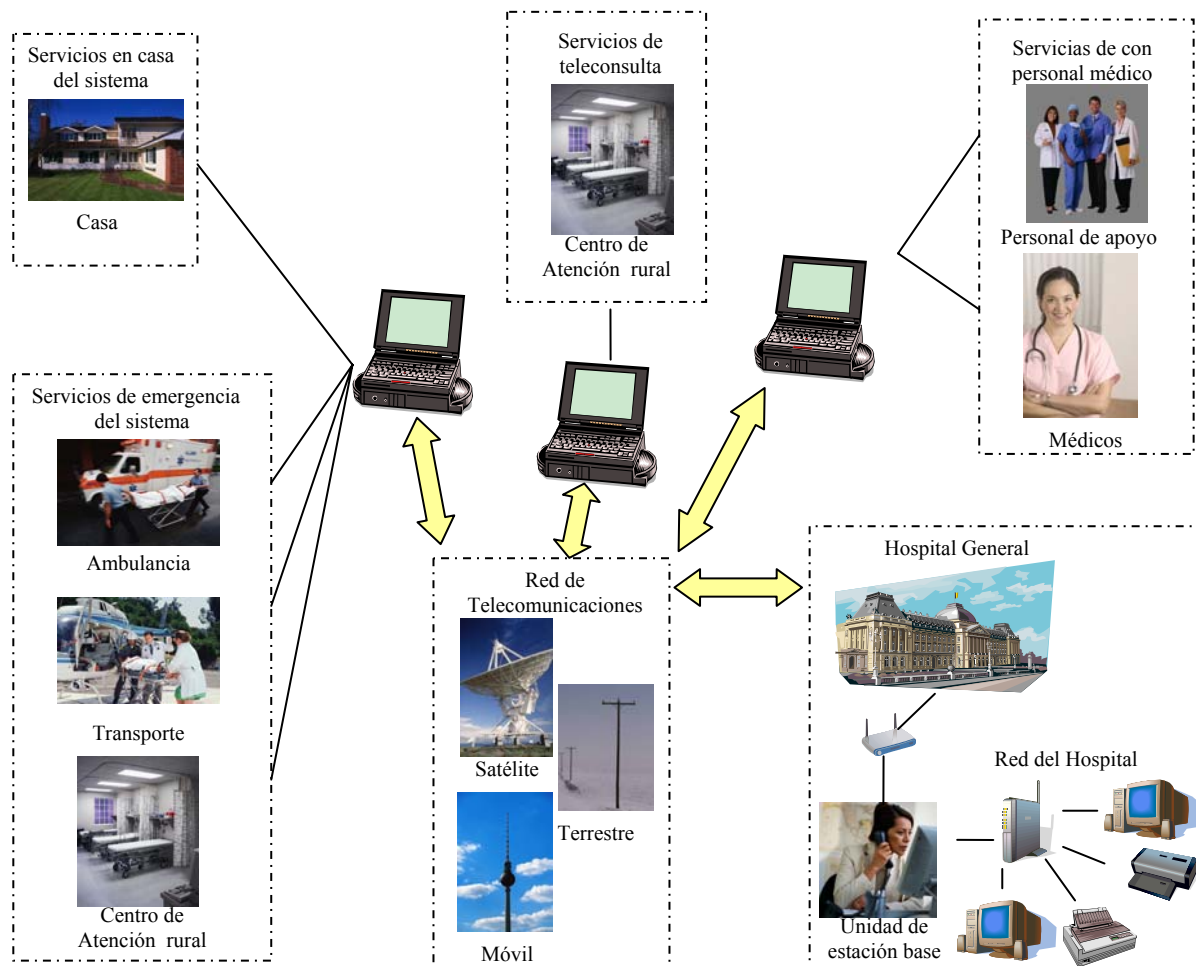


Figura 7.1 Infraestructura de comunicaciones para el sistema TRMCS.

Con las características del sistema antes descritas, se construyeron los siguientes diagramas.

7.1.1 Modelo de negocio.

La siguiente lista muestra los diagramas que pertenecen al modelo del negocio del sistema TRMCS.

- Diagrama de Roles
 - Diagrama de roles
- Diagramas de Clases
 - Dominio_TRMCS
 - Estados_TRMCS
 - Diagrama_Asoc_TRMCS
 - Elemento_Activo
 - Eventos_basicos_negocio
 - Paciente
 - Despachador

- o Empleado
 - o Medico
 - o Transacciones
 - o Tratamientos
 - o Monitor
 - o Equipo_Auxilio
 - o Medios_comunicacion
 - o Solicitud_atención
 - o Tipo_Servicio
- Diagramas de Colaboración
 - o Acceso_Despachador
 - o Acceso_Paciente
 - o Acceso_Sistema
 - o Despachador
 - o Medico
 - o Paciente
 - o Consulta
 - o Recetas
 - o Cuentas_Usuario
 - o Cambiar_Contraseña
- Diagramas de Secuencia
 - o sdPrincipal
 - o sdAcceso
 - o sdSuscribir
 - o sdDespachador
 - o sdMedico
 - o sdRecetas
 - o sdCambiar_contraseña_usuario
 - o Atención de emergencia
- Diagramas de Estados
 - o TRMCS
 - o Estados_TRMCS_4
 - o Transicion_Interfaz
 - o Paciente
 - o Medico
 - o Enfermeras
 - o Busqueda
 - o Consultas
 - o Recetas
 - o Cuentas_Usuario
 - o Cambiar_Contraseña
- Diagrama de Procesos
 - o Flujo de trabajo

- Diagramas de Actividades
 - o Cargar_Sistema
 - o Altas
 - o Bajas
 - o Actividades_Consulta
 - o Cuentas_Usuario
 - o DAEmergencia
 - o DAMonitor
 - o DASuscribir
 - o Disponibilidad_Servicio
- Diagrama de Componentes
 - o Cargar_Sistema_TRMCS_5
 - o Principal
 - o Interfaz
 - o Paciente
 - o Consulta
 - o Contraseña
 - o Cambiar_Usuario

7.1.2 Modelo de requerimientos.

La siguiente lista muestra los diagramas que pertenecen al modelo de requerimientos del sistema.

- Diagramas de Casos de Uso del sistema
 - o Sistema_TRMCS
 - o Subsistema_TRMCS
 - o Actividades_TRMCS_2
 - o Despachador_TRMCS_3
 - o CAEstado
 - o Principal
 - o Principal_2
 - o Acceso
 - o Paciente
 - o Paciente_2
 - o Despliegue_Paciente
 - o Medico
 - o Medico_2
 - o MedicoNuevo
 - o Despachador
 - o Enfermera
 - o Paramedico
 - o Proveedor de Servicios
 - o Altas
 - o Bajas
 - o Busqueda

- o Consultas
 - o CuentasUsuario
 - o Suscribirse
- Modelo Conceptual
 - o Solicitud de atención

7.1.3 Modelo Web.

La siguiente lista muestra los diagramas que pertenecen al modelo WEB del sistema

- Diagramas WEB
 - o WEB_Acceso
 - o WEB_Actualizar
 - o WEB_Consulta
 - o WEB_Estados
 - o WEB_Servicios
 - o WEB Suscribirse

La siguiente lista muestra los diagramas externos que forman parte del modelo del negocio.

- Diagramas de Comunicación
 - o Acceso_Despachador
 - o Acceso_Paciente
 - o Acceso_Sistema
 - o Cambiar_Contraseña
 - o Despachador
 - o Medico
 - o Paciente
 - o Consulta
 - o Cuentas_Usuario
 - o Recetas
- Diagramas de Despliegue
 - o Acceso
 - o Actualizar
 - o Consulta
 - o Estados
 - o Servicios
 - o Suscribirse
- Diagrama de Arquitectura
 - o TRMCS

7.2 Conclusiones.

Con base en la elaboración de los modelos correspondientes a las estrategias antes mencionadas, se elaboró un comparativo entre ambas con el propósito de identificar fortalezas y debilidades de cada una de ellas.

Respecto al modelado de la Meta arquitectura, los puntos considerados por la propuesta MAP son:

1. Misión y Visión
 - o Objetivo estratégico
 - o Objetivos particulares
2. Estrategia
 - o Reestructuración de procesos de negocio.
3. Métodos
 - o Identificación de procesos del negocio mediante Técnica de Grupos Nominales.
4. Herramientas.
 - o Herramientas CASE (UML).
5. Los Framework.
6. El modelo de arquitectura de aplicación cruzado.
7. Los patrones de la arquitectura.
8. Los patrones de análisis del negocio.
9. Los estándares.

Los puntos 1, 2, 3, 4, 5 son cubiertos por ambas metodologías. Los puntos 6 y 9 no se contemplan en la metodología propuesta sin embargo puede agregarse para robustecerla. Los puntos 7 y 8 no se tienen en la metodología propuesta pero se puede obtener fácilmente a partir de los diagramas diseñados.

Tabla 7-1 Comparativa con la Meta arquitectura

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO		
1. Misión y Visión		
a. Objetivo estratégico	√	√
b. Objetivos particulares		
2. Estrategia	√	√
a. Reestructuración de procesos de negocio.		
3. Métodos		
a. Identificación de procesos del negocio mediante TGN.	√	√
4. Herramientas	√	√
a. Herramientas CASE		
5. Los Framework	√	√
6. El modelo de arquitectura de aplicación cruzado		√
7. Los patrones de la arquitectura		√
8. Los patrones de análisis del negocio		√
9. los estándares.		√

Fuente: Elaboración propia.

Para la Arquitectura conceptual, la metodología MAP considera los puntos:

1. Diseño de los diagramas de casos de uso.
2. Especificación de los casos de uso.
3. Los alcances del sistema.
4. La contención y suposición de la arquitectura.
5. Los riesgos de la arquitectura.
6. La especificación de la interfaz de usuarios.
7. La especificación de los servicios de acceso (API y servicio de interfaz).

En este caso, los puntos 1 y 2 son considerados por ambas propuestas, el punto 3 no lo considera la propuesta pero se puede obtener de los diagramas de colaboración. El punto 5 no se considera, pero se puede agregar a la propuesta, el punto 6 no se tiene pero se puede obtener de los diagramas de caso de uso al igual que el punto siete que no se considera, pero que se puede obtener de los diagramas Web.

Tabla 7-2 Comparativa Arquitectura conceptual.

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO		
1. Diseño de los diagramas de casos de uso.	√	
2. Especificación de los casos de uso.	√	
3. Los alcances del sistema.		√
4. La contención y suposición de la arquitectura.		√
5. Los riesgos de la arquitectura.		√
6. La especificación de la interfaz de usuarios.		√
7. La especificación de los servicios de acceso (API y servicio de interfaz).		√

Fuente: Elaboración propia.

Para la Arquitectura lógica, la propuesta MAP incluye los puntos:

1. Diagramas de arquitectura.
2. Las responsabilidades del subsistema y sus relaciones.
3. Diagramas de comunicación.
4. Diagramas de actividades.
5. Selección de le tecnología clave.
6. La capa de la arquitectura y la interoperabilidad.

La propuesta elaborada incluye los cinco primeros puntos, como parte de su modela, únicamente el punto 6 no es incluido, sin embargo se puede obtener de los diagramas Web y de despliegue.

Tabla 7-3 Comparativa Arquitectura lógica.

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO		
1. Diagramas de arquitectura.	√	√
2. las responsabilidades del subsistema y sus relaciones.	√	√
3. Diagramas de comunicación.	√	√
4. Diagramas de actividades.	√	√
5. Selección de le tecnología clave.	√	√
6. La capa de la arquitectura y la interoperabilidad.		√

Fuente: Elaboración propia.

En cuanto a la Arquitectura física, la propuesta MAP considera los puntos.

1. Capturar la selección de la plataforma.
2. El modelo de Hardware.
3. La gestión del sistema.
4. Retraso del sistema y recuperación de desastres.
5. Escalabilidad.
6. Alta disponibilidad.
7. Autenticación y autorización.
8. Seguridad.
9. Requisitos de Red.

La estrategia que se propone en la presente investigación, al estar orientada a la parte lógica del sistema, solo considera dentro del modelado los puntos 6, 7 y 8 el resto de los puntos, aunque se encuentran implícitos dentro de la descripción del sistema, no se hacen evidentes en algún documento.

Tabla 7-4 Comparativa Arquitectura lógica.

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO		
1. Capturar la selección de la plataforma.		√
2. El modelo de Hardware.		√
3. La gestión del sistema.		√
4. Retraso del sistema y recuperación de desastres.		√
5. Escalabilidad.		√
6. Alta disponibilidad.	√	√
7. Autenticación y autorización.	√	√
8. Seguridad.	√	√
9. Requisitos de Red.		√

Fuente: Elaboración propia.

Respecto al Despliegue de la arquitectura, la propuesta MAP incluye los puntos:

1. Topología de Red
2. Modelos de construcción desplegable.
3. Especificación de configuración.
4. Diagramas de despliegue.

5. Especificación de instalación.
6. Guía de usuarios.
7. Centro de operaciones de datos.

La metodología propuesta solo incluye el punto 4, sin embargo los otros puntos se incluyen al hacer la especificación de los requisitos, por lo que sería conveniente documentarlos para hacerlos evidentes.

Tabla 7-5 Comparativa Arquitectura de despliegue.

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO		
1. Topología de Red		√
2. Modelos de construcción desplegable.		√
3. Especificación de configuración.		√
4. Diagramas de despliegue.	√	√
5. Especificación de instalación.		√
6. Guía de usuarios.		√
7. Centro de operaciones de datos.		√

Fuente: Elaboración propia.

La propuesta MAP considera para el modelo de Gestión/administración de la arquitectura los siguientes puntos:

1. Colección de datos.
2. Análisis
3. Persistencia
4. Visualización.
5. Optimización del ambiente.
6. Políticas de gobierno.
7. Certificación.
8. Registro.

De los cuales, la propuesta de la presente investigación solo incluye el punto 1 dentro del diccionario de datos, el punto 2 para la construcción del modelo y el punto 6 dentro de la especificación de las reglas del negocio.

Tabla 7-6 Comparativa gestión/administración de la arquitectura.

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO		
1. Colección de datos.	√	√
2. Análisis	√	√
3. Persistencia		√
4. Visualización.		√
5. Optimización del ambiente.		√
6. Políticas de gobierno.	√	√
7. Certificación.		√
8. Registro.		√

Fuente: Elaboración propia.

En cuanto a la Arquitectura de la información, la propuesta MAP considera los puntos:

1. Modelo de datos.
2. Almacén.
3. DBMS
4. Esquema de la B. D.
5. Replicación.
6. Transferencia de datos.
7. Respaldo de datos.
8. Almacenamiento de datos.
9. Reportes.
10. Meta-datos.
11. Consultas.
12. Prioridad de trabajo.
13. Control de acceso.

En este sentido, todos los puntos son cubiertos por la arquitectura propuesta.

Tabla 7-7 Comparativa Arquitectura de la información.

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO		
1. Modelo de datos.	√	√
2. Almacén.	√	√
3. DBMS	√	√

4. Esquema de la B. D.	√	√
5. Replicación.	√	√
6. Transferencia de datos.	√	√
7. Respaldo de datos.	√	√
8. Almacenamiento de datos.	√	√
9. Reportes.	√	√
10. Meta-datos.	√	√
11. Consultas.	√	√
12. Prioridad de trabajo.	√	√
13. Control de acceso.	√	√

Fuente: Elaboración propia.

Respecto a los aspectos de la arquitectura, la propuesta MAP considera los puntos:

1. Estándares de la empresa.
2. Complacencia.
3. Dependencia de inyección.
4. Anotaciones.
5. Proxy
6. Estándares de la industria.
7. Reutilización de recursos.

En este caso, la propuesta de modelado no contempla ninguno de los puntos anteriores, lo cual representa un área de oportunidad para robustecerla.

Tabla 7-8 Comparativa aspectos de la arquitectura.

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO		
1. Estándares de la empresa.		√
2. Complacencia.		√
3. Dependencia de inyección.		√
4. Anotaciones.		√
5. Proxy		√
6. Estándares de la industria.		√
7. Reutilización de recursos.		√

Fuente: Elaboración propia.

Finalmente, para la arquitectura de componentes, la propuesta MAP, considera los puntos:

1. Especificación de componentes.
2. Tarjetas CRC.
3. Diagramas de clases.
4. Patrones de diseño.
5. Tecnología.
6. Paquetes.
7. Utilidades de código abierto.
- 8.

La propuesta de la presente investigación incluye el punto 3 dentro de los modelos construidos y el punto 5 dentro de la especificación de la plataforma tecnológica que soportará al sistema. En particular el punto 4 se considera como un área de interés ya que se podría robustecer ésta metodología incluyendo patrones de diseño.

Tabla 7-9 Comparativa arquitectura de los componentes.

ESTRATEGIA	Propuesta	MAP
REQUERIMIENTO	SI	NO
1. Especificación de componentes.		√
2. Tarjetas CRC.		√
3. Diagramas de clases.	√	√
4. Patrones de diseño.		√
5. Tecnología.	√	√
6. Paquetes.		√
7. Utilidades de código abierto.		√

Fuente: Elaboración propia.

Con base en los resultados antes mostrados, se puede concluir que la principal aportación de la estrategia propuesta en esta investigación, es que basa todo su desarrollo sobre una plataforma estándar (UML), ocupando algunas extensiones al mismo para el desarrollo del modelado la Web. Esto facilita al diseñador, el manejo y comprensión entre los diferentes diagramas de modelo: negocios, sistema y la Web. Además, la estrategia propuesta representa de forma más detallada y específica aspectos referentes a la arquitectura lógica y a la arquitectura de la información durante el modelado, lo cual no ocurre con la estrategia MAP.

Ambas estrategias atienden los aspectos relevantes de la Meta arquitectura. Los aspectos que no se atienden en la estrategia propuesta en la primera parte de esta investigación se pueden agregar a la misma sin ninguna dificultad. En cuanto a la arquitectura conceptual, la propuesta MAP representa con mayor detalle aspectos de alcances y riesgos del sistema.

Finalmente en los aspectos arquitectura física, despliegue de la arquitectura, gestión /administración de la arquitectura y de los componentes, la propuesta MAP consigue un modelo con mayor información, mientras que la propuesta de esta investigación puede enriquecerse para incluir aquellos aspectos que no atiende.

7.3 Trabajo Futuro - Análisis navegacional del sistema.

Con la propuesta para el modelado hipermedia de Garzotto et al. (1993) se comenzó a hablar de metodologías para sistemas hipermedia, con lo que se sientan las bases para el nacimiento de la ingeniería Web. En los primeros trabajos se comienza a analizar porqué la nueva tendencia de desarrollo hipermedia en la Web, requiere de nuevas técnicas de desarrollo, diferentes a las propuestas en métodos clásicos de ingeniería del software. Analizando dichos estudios se puede concluir que las diferencias principales que obligan a la inclusión de nuevos modelos y adaptación de los existentes son:

- a. La complejidad de la estructura hipermedia: el complejo sistema navegacional que se establece debido a la existencia de hiperenlaces e hipervínculos causa, en muchas ocasiones, que el usuario pueda llegar a perderse en el hiperespacio si el sistema no se construye basado en alguna técnica que garantice la calidad del mismo (Shubin & Meehan 1997).
- b. La colaboración interdisciplinar: la gran cantidad y diversidad de roles que deben tomar parte en el desarrollo de sistemas Web complica su desarrollo en comparación con los sistemas clásicos (Koch 2001).
- c. La visión externa de la Web: mientras que en los sistemas clásicos el número de usuarios finales estaba normalmente definido y cerrado o al menos controlado a un grupo de usuarios concreto, en los sistemas Web la visión cambia. El número de usuarios no es sólo ilimitado, si no que, además, es totalmente impredecible en la

mayoría de los casos. Los roles de usuarios finales que interactúan con el sistema es abierto y hay que adaptar el desarrollo a que se puedan satisfacer las necesidades de cualquier usuario anónimo sin llegar a conocer su perfil.

- d. El mantenimiento: al ser más abiertos, los sistemas Web requieren de un mantenimiento tanto aumentativo, como adaptativo, como correctivo, más crítico que en los sistemas clásicos, con el añadido de que, además, los sistemas Web normalmente deben funcionar 24 horas al día y 7 días a la semana, lo que hace disponer de menos tiempo para las tareas de mantenimiento.

La navegación se vuelve entonces una característica crítica dentro de la ingeniería Web (Cachero & Koch 2002). Se han propuesto muchos modelos y técnicas para el estudio de la navegación y son muchos los grupos que trabajan en este ámbito.

Sin embargo, aún existen algunos puntos sin tratar. Algunos aspectos como el hecho de que la navegación haya sido un concepto tratado principalmente en análisis, o que no se haya involucrado al usuario y cliente final en su definición, la falta de soporte CASE para modelar los elementos relacionados con la navegación o la necesidad de garantizar la calidad del modelado y el tratamiento de la misma durante todo el ciclo de vida, son las fuentes de interés en esta línea de investigación.

Definición de la navegación.

Uno de los primeros problemas que se plantea a la hora de trabajar con la navegación es definir realmente qué se entiende por navegación. El concepto de navegación ha adquirido un interés particular desde la aparición de las primeras aplicaciones hipertexto hasta los actuales sistemas de software orientados a la Web. Pero realmente, no existe una definición estándar de lo que significa la navegación. A continuación se presentan varias definiciones de navegación que permiten deducir varias ideas.

- *La navegación es la sensación que el usuario tiene cuando navega hacia un objeto dentro del dominio de la aplicación (Schwabe & Rossi 1998a).*

- *La navegación es el proceso cognitivo de adquirir conocimiento sobre un espacio, estrategias para moverse a través del espacio, y cambiar el metaconocimiento del espacio. (Schwabe & Rossi 1998b).*
- *La navegación expresa cómo las páginas y las unidades con contenido son ligados para formar hiperespacios (Ceri et al. 2000).*
- *La forma en que se alcanza un nodo mediante un enlace es navegación. La navegación es la más importante característica de la hipermedia (Suh & Lee 2001).*
- *La navegación se define como un subconjunto de relaciones semánticas establecidas entre clases. Permite ir de un objeto a otro relacionado semánticamente*

La primera de ellas es que la navegación implica mostrar una determinada visión de la información almacenada en el sistema. Los sistemas manejan y gestionan una información que es presentada de manera adecuada durante la navegación del usuario por el sistema.

La navegación implica también la idea de movimiento. En este sentido, cuando se presenta una determinada información al usuario también se le ofrece una serie de destinos posibles donde se puede dirigir o navegar a partir del punto en el que se encuentra. Esta idea de movimiento unida a la idea de visión de la información es lo que se define como hiperespacio: un área que marca el ámbito de la aplicación estructurada mediante zonas estáticas que se pueden navegar para conseguir la información necesaria.

Pero la navegación implica otros elementos más. Por un lado la adaptabilidad al entorno. En un sistema navegacional es necesario adaptar el entorno de navegación a la situación concreta. Así, si el usuario se encuentra en un punto determinado, se pueden abrir diferentes posibilidades de navegación dependiendo de dónde se haya venido y del tipo de usuario que esté conectado: un sistema navegacional puede cambiar dependiendo del usuario que en cada momento interactúe con el sistema (De Troyer et al. 2003) (González et al. 2001).

La navegación también está relacionada con el concepto de la funcionalidad. Las posibilidades funcionales del sistema deben ofrecerse al usuario en el momento preciso y adecuado a lo largo de su visita por el sistema.

Con todo esto, se puede llegar a concluir que la navegación es la característica del software que permite estructurar cómo se desea mostrar la información al usuario y poner a su disposición la funcionalidad de una manera adecuada a sus necesidades, ofreciendo, además, un sistema oportuno y sencillo para acceder a todo ello mediante estructuras navegacionales complejas y avanzadas.

Existen diversas metodologías Web que están integrando la navegación como parte de sus modelos, entre otras se encuentra:

- HDM- Hypermedia Design Model
- RMM- Relationship Management Method
- EORM- Enhanced Object Relationship Methodology
- OOHDM- Object-Oriented Hypermedia Design Method
- WSDM- Web Site Design Method
- SOHDM- Scenario-based Object-oriented Hypermedia Design Methodology
- RNA- Relationship-Navigational Análisis
- HFPM- Hypermedia Flexible Process Modelling Strategy
- Building Web Applications with UML
- WebML- Web Modelling Language
- UWE-UML Based Web Engineering
- W2000
- Proyecto UWA- Ubiquitous Web Applications
- OOH- Object-Oriented Hypermedia Method

Cada metodología tiene su propio concepto de navegación y no todas entienden el concepto de la misma forma. Existe un ciclo de vida en el que se estudia la navegación, pero el problema es saber en que parte del ciclo de vida se debe estudiar.

Cada una de ellas lleva a cabo una serie de iteraciones que permiten abordar los flujos de trabajo que hay que estudiar en cada fase: requisitos, análisis, diseño, implementación y pruebas. Existen también diferentes técnicas para tratar la navegación:

- Casos de uso.
- Escenarios.
- Enriquecimiento ER.
- Clases navegacionales.
- Contextos navegacionales.
- Matriz de enlaces y modelo ASN.
- Patrones de diseño.
- Descripciones textuales.
- Prototipos.

Se propone entonces, realizar un análisis de los diferentes enfoques con los que se aborda el problema del modelado navegacional, con el propósito de seleccionar aquellas herramientas que mejor se adecuen a la propuesta del presente trabajo de investigación de esta manera robustecer la propuesta elaborada.

La parte de navegación se incluirá como una continuación del diseño hasta ahora realizado. Se tomarán los puntos identificados durante el planteamiento del problema como base para el diseño de navegación. La siguiente tabla muestra los principales aspectos que se tienen que cubrir dentro de los trabajos futuros de la presente investigación.

Tabla 7-10 Actividades futuras para la elaboración del modelo navegacional del sistema.

ACTIVIDAD	ESTADO
1. Elegir metodología.	X
2. Agregar técnicas para navegación (uso de prototipos o descripciones textuales).	X
3. Planteamiento del problema.	√
3.1 Entorno del problema.	√
3.2 Metodología del trabajo.	√
3.3 Objetivos	√
- Uso de estándares	X
- Incluir usuarios	√
- Búsqueda de sistematización y automatización	X
- Relaciones con otros entornos metodológicos	X
4. Estructura de la aproximación	X

4.1 Ingeniería de requisitos	X
- Captura de requisitos	X
- Definición de requisitos	X
- Validación de requisitos	X
Se realiza una valoración de requisitos, si hace falta algo o hay errores se debe realizar lo siguiente:	
- Tratamiento de requisitos	X
- Definir que modelos hay que desarrollar en la ingeniería de requisitos	X
- Aplicar técnicas de captura de requisitos adecuadas para el tratamiento de modelos	X
- Una vez conocidos los modelos a representar, ofrecer técnicas de representación de los mismos	X
- Definir técnicas para validar modelos	X
4.2 Análisis	X
4.3 Estructura del trabajo	X
5. Influencias	X
Modelos para tratamiento de navegación	
1. Modelos de la ingeniería de requisitos	X
1.1 Modelo de requisitos de almacenamiento de información	X
1.2 Modelo de actores	X
1.3 Modelo de requisitos funcionales	X
1.4 Modelo de requisitos de interacción	X
2. Modelo del análisis	X
2.1 Modelo conceptual	X
2.2 Modelo de navegación	X
Derivación y relación entre modelos	
1. Relación entre modelos	X
2. Derivación del modelo conceptual básico	X
2.1 Derivación de las clases desde los requisitos de almacenamiento y las naturalezas	X
2.2 Derivación de atributos desde los datos específicos	X
2.3 Derivación de asociaciones desde los datos específicos	X
3. Derivación del modelo de navegación básico	X
3.1 Derivación de los actores en estudio desde el grupo de actores	X
3.2 Derivación de los nodos desde los prototipos de visualización	X
3.3 Derivación de las queries desde los prototipos de visualización	X
3.4 Derivación de los índices desde los prototipos de visualización	X
3.5 Derivación de los menús	X
3.6 Derivación de los enlaces desde los prototipos de visualización	X
4. Construcción del modelo conceptual final	X
4.1 Revisión de las clases	X
4.2 Revisión de los atributos	X
4.3 Revisión de las asociaciones	X
4.4 Detectar la herencia	X
4.5 Detección de clases asociación y las asociaciones calificadas	X
4.6 Revisión de nombres y descripciones	X

4.7 Revisión de paquetes	X
5. Construcción del modelo de navegación final	X
5.1 Revisión de los nodos	X
5.2 Revisión de los índices y rutas guiadas	X
5.3 Revisión de las queries	X
5.4 Revisión de los menús	X
5.5 Revisión de los enlaces	X
5.6 Revisión de los nombres y descripciones	X
5.7 Revisión de la navegación del modelo	X
Metodología de desarrollo	
1. NDT (Navigational Development Techniques)	X
2. NDT-Tool	X
3. Aplicaciones de NDT y NDT-Tool	X

(X). No esta, pero se propondrá un diseño de acuerdo a la propuesta.

(√). Si esta, pero si se requiriere se podría replantear de acuerdo a la propuesta.

Fuente: Elaboración propia.

REFERENCIAS.

- Aalst, W. (1998). The Journal of Circuits, Systems and Computers. *The Application of Petri Nets to Workflow Management.*, 8(1):21–66.
- Aalst, W. (2004) *Pi calculus versus petri nets: Let us eat “humble pie” rather than further inflate the “pi hype”*. Artículo de discusión no publicado Recuperado en diciembre 2004 disponible en: <http://tmitwww.tn.tue.nl/research/patterns/download/pi-hype.pdf>.
- Aalst, W. y Hofstede, A. (2003). *YAWL: Yet Another Workflow Language. Technical Report FIT-TR-2003-04*, Queensland University of Technology,
- Aalst, W., Hofstede, A., Kiepuszewski, B., y Barros, A.(2003). *Workflow patterns. Distributed and Parallel Databases*, 14(3):5–51.
- Aalst W., Hofstede A., y Weske M.(2003). Business process management: A survey. En Lecture Notes in Computer Science. *Proceedings of the International Conference on Business Process Management (BPM 2003)*, volumen 2678, páginas 1–12. Springer- Verlag.
- Agostini, A. y de Michelis, G.(2000). Computer Supported Cooperative Work *A light workflow management system using simple process models*. 9(3/4):335–363.
- Ayoma, M(1998). Web-Based Agile Software Development, IEEE Computer, noviembre 1998, pp. 56-65.
- Baresi L., Garzotto F., Paolini P.(2001) Extending UML for Modelling Web Applications. Annual Hawaii International Conference on System Sciences. pp. 1285 - 1294. Maui , USA. Enero 2001.
- Barry y asociados. (2003). Recuperado en Agosto, 2004, disponible en <http://www.service-architecture.com>

- Booch, G., Rumbaugh, J y Jacobson, I.(1999). The Unified Modeling Language User Guide. Addison-Wesley.
- Cachero, C., Koch, N. (2002) Conceptual Navigation Analysis: a Device and Platform Independent Navigation Specification. 2nd International Workshop on Web-oriented Software Technology (IWWOST'02). Málaga, España.
- Ceri, S., Fraternali, P., Bongio.(2000) Web Modelling Language (WebML): A Modelling Language for Designing Web Sites. Conference WWW9/Computer Networks 33 (1-6) pp. 137-157.
- Cockburn, A. (1997). Using Goal-Based Use Cases. JOOP, Vol. 10, No. 7 (Nov/Dec 1997) pp 56-62
- Coleman, D. (1998).A Use Case Template: Draft for discussion. Recuperado en septiembre 2008, disponible en http://www.bredemeyer.com/use_case.pdf
- Conallen, J. (2002). *Modeling Web Applications with UML* (2ª ed.). Boston MA., EE. UU.: Addison Wesley
- Datz, T. (2004, January 15). What you need to know about service-oriented architecture. CIO Magazine. Recuperado en Agosto 2004, disponible en <http://64.28.79.79/archive/011504/soa.html>
- Deiters, W. y Gruhn, V.(1998). Automated Software Engineering, *Process management in practice applying the FUNSOFT net approach to large-scale processes*. 5(1):7-25.
- De Troyer, O. ,Plessers, P. ,Casteleyn, S. Conceptual View (2003). Integration for Audience Driven Web Design. WWW2003 Conference. Budapest, Hungría.

- Deshpande, Y., Marugesan, S., Ginige,A., Hanse,S., Schawabe,D., Gaedke, M, B. White. (2002). Web Engineering. Journal of Web Engineering. Vol. 1 N° 1. pp. 3-17.2002. Rinton Press. 2002.
- DoD (1997).C4ISR Architecture Working Group .C4ISR architecture framework. Version 2. Recuperado en Mayo 2007, disponible en <http://www.fas.org/irp/program/core/fw.pdf>
- Dumas, M. y Hofstede, A.(Octubre 2001). Proc. of the International Conference on the Unified Modeling Language (UML). *UML Activity Diagrams as a Workflow Specification Language*. Toronto, Canada. Springer Verlag.
- Dyché, J. (2002). The CRM handbook: A business guide to customer relationship management. Boston: Addison-Wesley Professional.
- Eclipse (2007). Eclipse process framework. Recuperado en Mayo 2007, disponible en <http://www.eclipse.org/epf>
- Eertink, H., Janssen, W., Lutthuis, P. O., Teeuw, W., y Vissers, C.(1999) Lecture Notes in Computer Science. *A business process design language*. En.Wing, J.M Woodcock, J., y Davies, J. (eds), Proceedings of Formal Methods (FM 99), Volume I, volume 1708 páginas 76–95. Springer.
- Ellis, C. y Nutt, G.(1993) Lecture Notes in Computer Science. *Modelling and enactment of workflow systems*. En Marsan, M. A.(ed.), *Application and Theory of Petri Nets* 1993, volume 691 páginas 1–16.
- Eriksson, H.E., Penker, M. (2000). Business Modeling with UML. Business Patterns at Work. John Wiley & Sons, Inc.
- Eshuis, R. (2002). *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*. PhD thesis, University of Twente.

- Eshuis, R. y Wieringa R.(2001a). *A formal semantics for UML activity diagrams*. Technical Report TR-CTIT-01-04, University of Twente.
- Eshuis, R. y Wieringa, R.(2001b). Lecture Notes in Computer Science. Proceedings of Fundamental Approaches to Software Engineering (FASE 2001), *A real-time execution semantics for UML activity diagrams*. Número 2029, pages 76–90.
- Eshuis, R. y.Wieringa, R. (2002). Proceedings of the International Conference on Software Engineering ICSE (2002). *Verification support for workflow design with uml activity graphs.*, páginas 166–176. ACM Press.
- Federal Office of Management and Budget (2007). Federal enterprise architecture framework. Recuperado en Mayo 2007, disponible en <http://www.whitehouse.gov/omb/egov/a-2-EAModelsNEW2.html>
- Garzotto F., Schwabe D. and Paolini P. (1993). HDM-A Model Based Approach to Hypermedia Application Design. ACM Transactions on Information System, 11 (1), pp 1-26. Enero, 1993.
- Georgakopoulos, D., Hornick, M. F., y Sheth, A. P.(1995). *An overview of workflow management: From process modeling to workflow automation infrastructure*. *Distributed and Parallel Databases*. páginas 119–153.
- González, M., Mejías, M., Escalona, M.J., Gasca, R.M., Ortega, J.A. (2001). Interacción con los Usuarios en bibliotecas digitales. I Jornadas Dolmen. pp. 113-120. Sevilla, España.
- Harel, D., Lachover, H., Naamad, A., Pnueli, A., Politi, M., R. Sherman, Shtull- Trauring, A., y Trakhtenbrot, M. (1990). IEEE Transactions on Software Engineering *STATEMATE: AWorking Environment for the Development of Complex Reactive Systems.*, 16(4):403–414.

- Harel, D., (1987). Science of Computer Programming Statecharts: *A visual formalism for complex systems*. 8(3):231–274.
- Hollingsworth, D. (1995). Workflow Management Coalition. - *The Workflow Reference Model*, WfMC TC00-1003 (Issue 1.1).
- IEAD (Institute for Enterprise Architecture Developments) (2004). Extended enterprise architecture framework. Recuperado en Mayo 2007, disponible en <http://www.enterprise-architecture.info>
- Jablonski, S. y Bussler, C. (1996). *Workflow Management: Modeling Concepts, Architecture and Implementation*. International Thomson Computer Press, Londres, Inglaterra.
- Janssen, W., Mateescu, R., Mauw, S., y Springintveld, J. (1998). *Verifying business processes using spin*. En Najm, E., Serhrouchni, A., y Holzmann, G. (eds), *Proceedings of the International Spin Workshop*.
- Král, J., & emlika, M. (2003). Software confederations - An architecture for global systems and global management. In S. Kamel, (Ed.), *Managing globally with information technology* (pp. 57-81). Hershey, PA: Idea Group.
- Král, J. (1995). Experience with the development and use of flexible manufacturing systems. Artículo no publicado.
- Král, J., & emlika, M. (2001). Electronic government and software confederations. In A. M. Tjoa, & R. R. Wagner (Eds.), *Twelfth International Workshop on Database and Experts System Application* (pp. 125-130). Los Alamitos, CA: IEEE Computer Society.
- Koch, N.(2001). Software Engineering for Adaptative Hypermedia Applications. Ph. Thesis, FAST Reihe Softwaretechnik Vol(12), Uni-Druck Publishing Company, Munich. Germany. 2001.

- Korson, T. (1998): Misuse of Use Cases. Recuperado de la dirección Web:
<http://software-architects.com/publications/korson/korson9803om.htm>.
- Korson, T. (1999). Constructing Useful Use Cases. Recuperado de la dirección Web
<http://softwarearchitects.com/publications/korson/usecase3>.
- Kruchten, P. (2003). The rational unified process:An introduction. 3rd Edition.
Massachusetts:Addison Wesley.
- Lang, M. (2002). Hypermedia System Development. Do we really need new Methods?. Site-
Where Parallels Intersect. Informing Science. pp. 883-891. June 2002.
- Lenzerini, M. (2001). Data integration is harder than you thought. Recuperado en agosto, 2004,
disponible en <http://www.science.unitn.it/coopis>
- Lowson, B., King, R., & Hunter, A. (1999). Quick response: Managing the supply chain to meet
consumer demand. New York: John Wiley & Sons.
- Martin, J. Odell, J. J. (1997).Object-Oriented Methods: A Foundation. Prentice Hall.
- McDonald, A y Welland R. (2001). Agile Web Engineering (AWE) Process, Department of
Computer Science. University of Glasgow, Technical Report TR-2001-98, disponible en
<http://www.dcs.gla.ac.uk/andrew/TR-2001-98.pdf>
- Milner, R. (1999). *Communicating and Mobile Systems: the π -Calculus*. Cambridge University
Press.
- Murata, T. (1989). Proc. of the IEEE. *Petri nets: Properties, analysis, and applications*.
77(4):541–580.
- Muruguesan, S. (1999)., WebE Home Page, recuperado en agosto 2008, disponible en
<http://fitserv.macarthur.uws.edu.au/san/WebEHome>.

- Muth, P., Wodtke, D., Weissenfels, J., Weikum, G. y Kotz-Dittrich, A. (1998). *Enterprisewide work ow management based on state and activity charts*. En Dogac, A., Kalinichenko, L., Ozsu, T., y Sheth, A (eds), *Workflow Management Systems and Interoperability*, NATO/ASI. Springer,
- Oberweis, A., Schätzle R., Stucky, W. , Weitz, W. y Zimmermann, G.(1997). *INCOME/WF: A Petri net based approach to workflow management*. En Krallmann, H. (ed.), *Wirtschaftsinformatik 97*, pages 557–580. Springer.
- OMG (Object Management Group) (2007). Model driven architecture. Recuperado en Mayo 2007, disponible en <http://www.omg.org/mda>
- Ortín, M.J., García-Molina, J. (1999). Modelado con Roles en UML. IV Jornadas de Ingeniería del Software y Bases de Datos. Cáceres, España
- Ould, M. A., (1995). *Business processes: modelling and analysis for re-engineering and improvement*. John Wiley & Sons.
- Peterson, J.(1981). *Petri Net Theory and the Modeling of Systems*. Prentice Hall.
- Powell T. A. (1998). *Web Site Engineering*, Pretice-Hall
- Pressman,R.S. (2005). *Ingeniería del Software. Un enfoque práctico*. Mc Graw Hill, 2005.
- Purdue University (1989). The Purdue enterprisereference architecture. Recuperado en Mayo 2007, disponible en <http://pera.net>
- Putman, J.R. (2001). *Architecting with RM-ODP*.New Jersey: Prentice Hall PTR.
- Reisig, W. y Rozenberg G. (1998). *Lecture Notes in Computer Science Lectures. Petri nets I: Advances in Petri nets*, volume 1491. Springer

- Salimifard, K. y Wright, M. (2001). European Journal of Operational Research, *Petri net-based modelling of work ow systems: An overview*. 134(3):218–230.
- Scheer, A.W.(1999). *ARIS - Business Process Modeling*. Springer, 2nd edición.
- Schwabe, D., Rossi, G. (1998a) An Object Oriented Approach to Web-Based Application Design, Theory and Practice of Object Systems 4(4), 1998. Wiley and Sons, New York. USA.
- Schwabe, D., Rossi, G. (1998b) Developing Hypermedia Application Using OOHDM. Workshop on Hypermedia Development Processes, Methods and Models (Hypertext 98), Pittsburgh, USA..
- Shan, T.C. y Hua, W.W. (2006). Solution architecture of N-Tier applications. In Proceedings of 3rd IEEE International Conference on ServicesComputing (pp. 349-356). California: IEEE Computer Society.
- Shan T. C. y Hua W.W.(2008) Model-Centric Architecting Process. Software Engineering for Modern Web Applications Methodologies and Technologies. Christian Brothers University, USA
- Shubin, H., Meehan, M.M.(1997). Navigation in Web Applications. ACM Interactions Magazine. IV. Noviembre 1997.
- Suh W., Lee, W. (2001) A Methodology for Building Content-oriented hypermedia systems. The Journal of Systems and Software, Vol. 56. pp. 115-131.
- Software Engineering Institute (SEI) at CMU (2007). Scenario-based architecture analysis method. Recuperado en Mayo 2007,disponible en http://www.sei.cmu.edu/architecture/scenario_paper.

Standish Group (2007). The Standish Group Chaos Report 2006. Recuperado en Mayo 2007, disponible en <http://www.standishgroup.com>

The Open Group (2007). The Open Group architecture framework. Recuperado en Mayo 2007, disponible en <http://www.opengroup.org/togaf>

Treasury Department CIO Council (2000). Treasury enterprise architecture framework. Version 1. Recuperado en mayo 2007 disponible en <http://www.eaframeworks.com/TEAF/teaf.doc>

Verbeek, H., Basten, T. y Aalst, W. (2001). The Computer Journal. *Diagnosing workflow processes using woflan.*, 44(4):246–279.

Vidal, M. Á. (2003). *Diseño de un prototipo de servicios Web para la contratación de un seguro y del transporte del café.* Recuperado en septiembre 2003 disponible en <http://www.lania.mx/biblioteca/rtecnicos/Lania-RT-2003-07/LANIA-RT-2003-07-MAVA.pdf>

Whitenack, B. (1995). RAPPeL: A Requirements Analysis Process Pattern Language for Object-Oriented Development. In: Coplien, J.O., Schmidt, D.C. (eds.): *Pattern Languages of Program Design*. Addison-Wesley pp 259-291

Winograd, T. y Flores, R. (1987). *Understanding Computers and Cognition*. Addison–Wesley.

Workflow Management Coallition. (2002). *Workflow Standard - Workflow Process Definition Interface - XML Process Definition Language*. WPMC-TC-1025 (Version 1.0).

W3C. (2001). Web service definition language. A proposal of W3 Consortium. Recuperado en Agosto 15, 2004, disponible en <http://www.w3.org/TR/wsdl>

Zachman, J.A. (1987). A framework for information systems architecture. *IBM Systems Journal*, 26(3), 276-295.

ANEXO

MODELO DEL NEGOCIO

Diagramas de Clases

Diagrama de clases "Despachador"

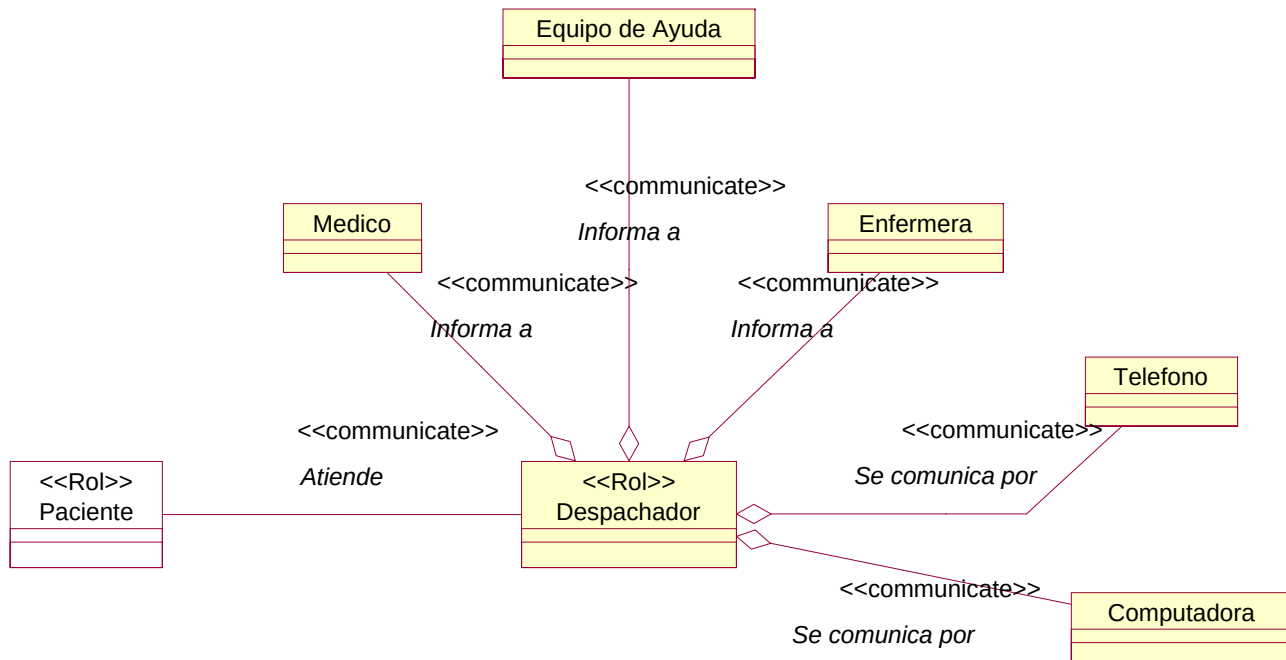
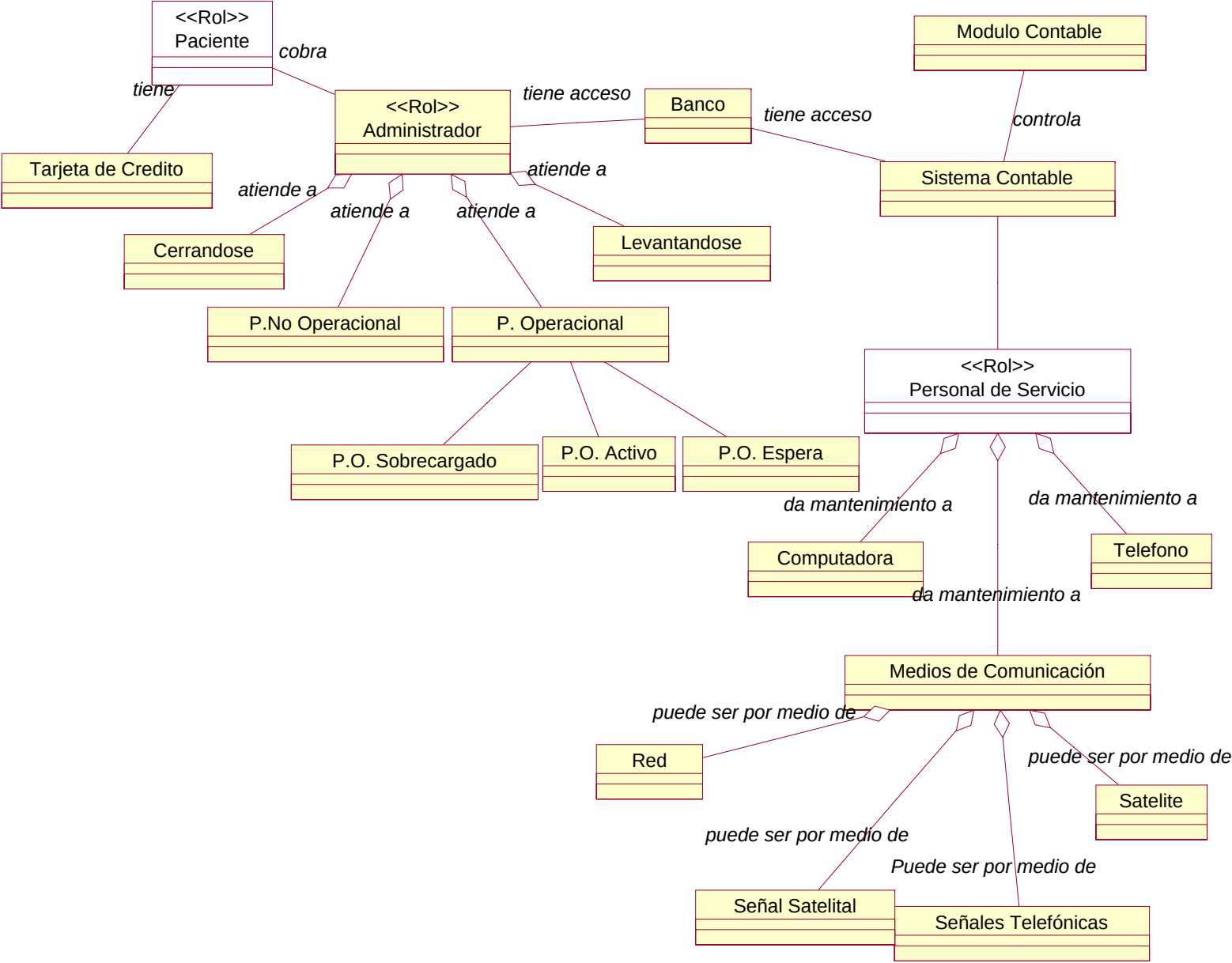


Diagrama de clases "Asociacion TRMCS"



Dominio TRMCS

paciente

Medico

Proveedor

Enfermera

Monitor

P.O. Espera

P.O. Activo

Celular

<<Rol>>
Despachador

Medicamento

Banco

Emergencia

Satelite

Red

Equipo de Ayuda

Visor de Datos

P. Operacional

P. Levantandose

P. Cerrandose

P.O. Sobrecarga

Elemento Activo

Computadora

<<Rol>>
Administrador

Señal Satelital

Telefono

Equipo Medico

Teléfono Privado

Consulta

Monitor de Paciente

Modulo Contable

P. No Operacional

Equipo de Primeros Auxilios

Antenas Telefónicas

Tarjeta de Credito

Teléfono Público

<<Rol>>
Personal de Servicio

Equipo de Emergencia

Sistema Contable

Palm

Suscribirse

Laptop

Diagrama de clases "Elemento Activo"

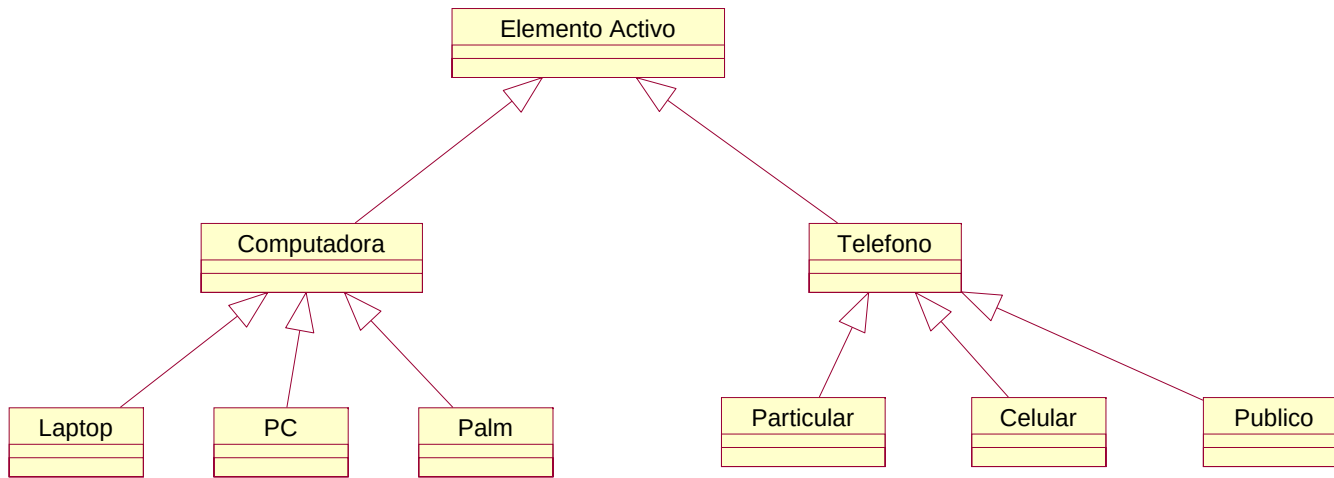


Diagrama de clases "Empleado"

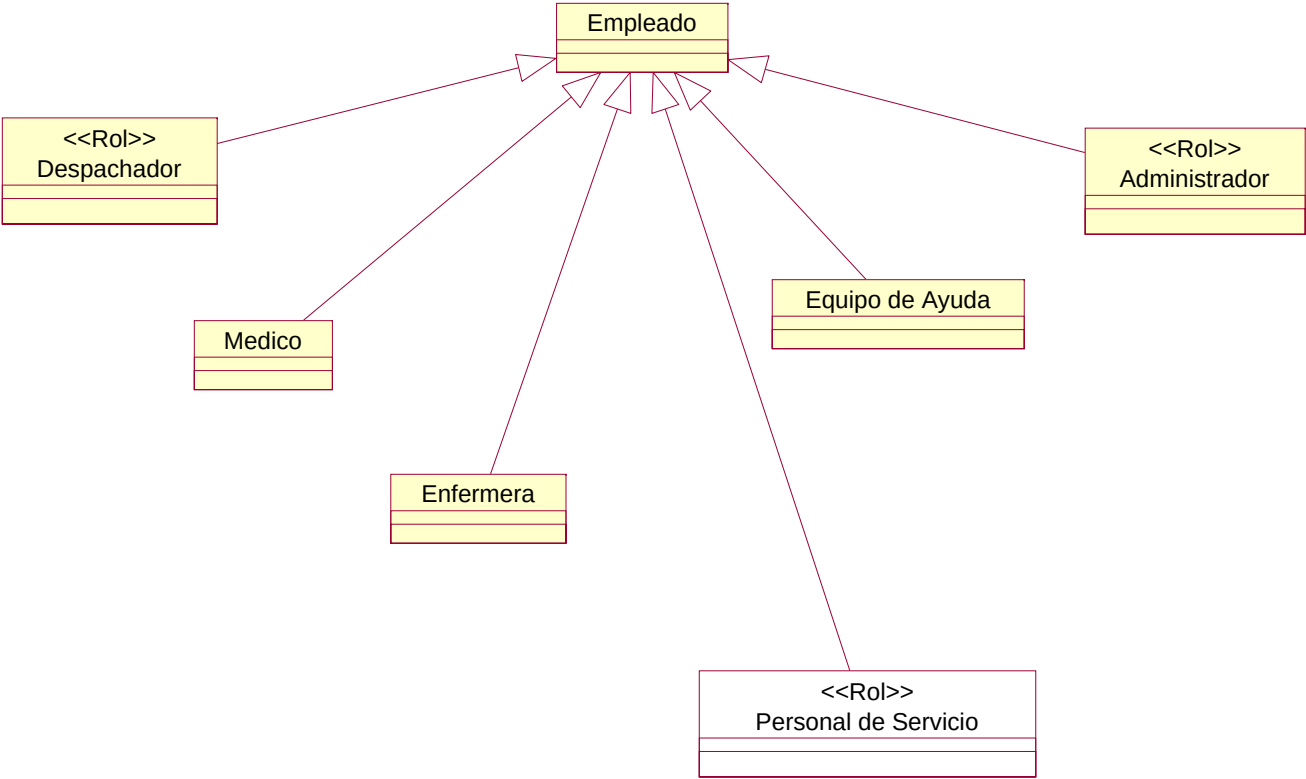
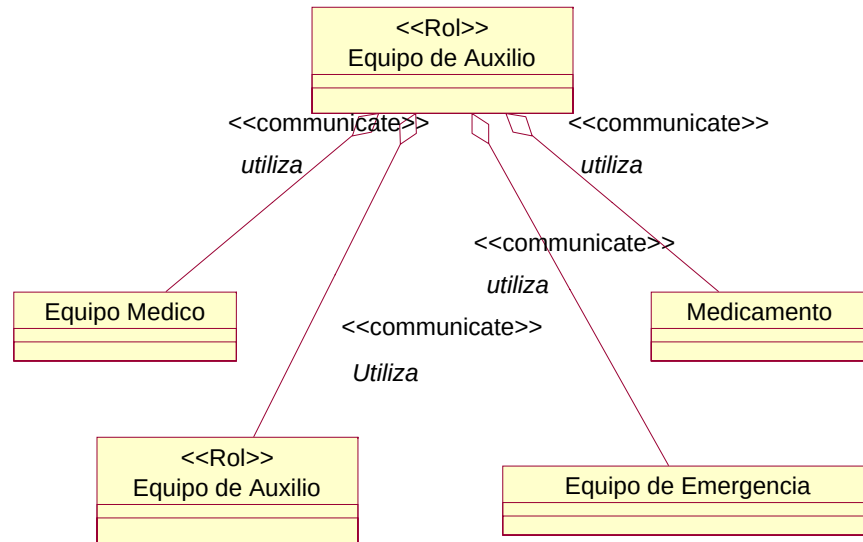


Diagrama de clases "Equipo de Auxilio"



Diagra de clases "Estados TRMCS"

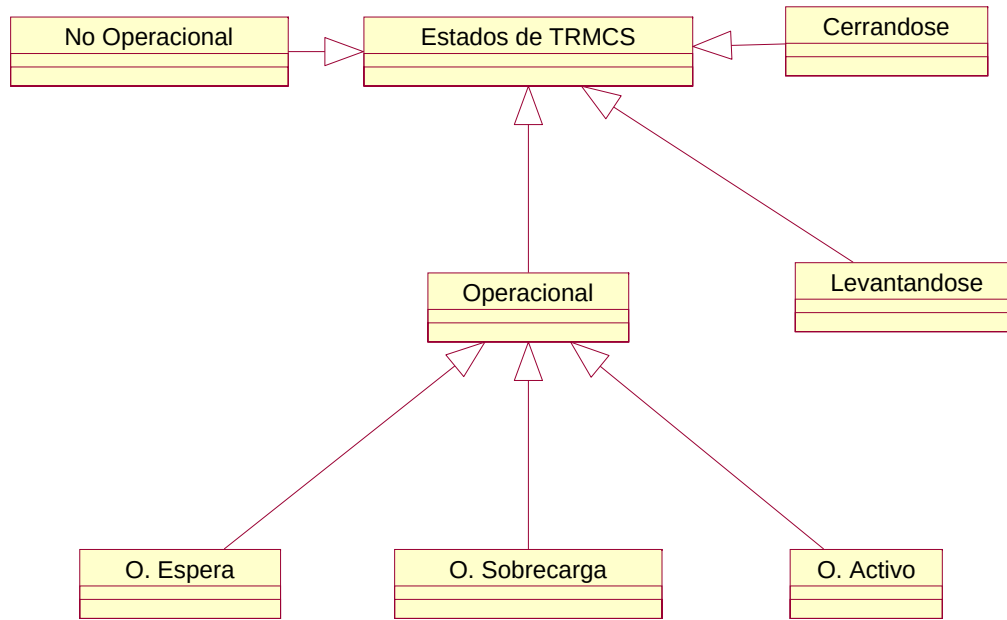


Diagrama de clases "Eventos de negocio"

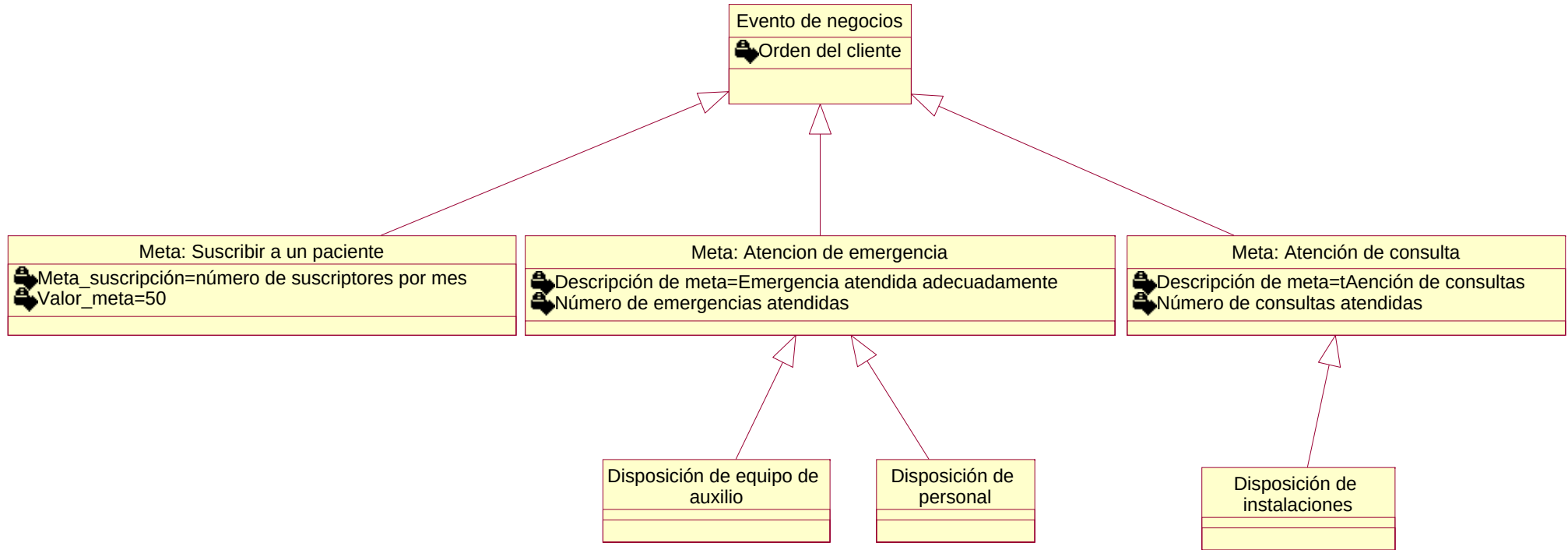


Diagrama de clases "Consulta"

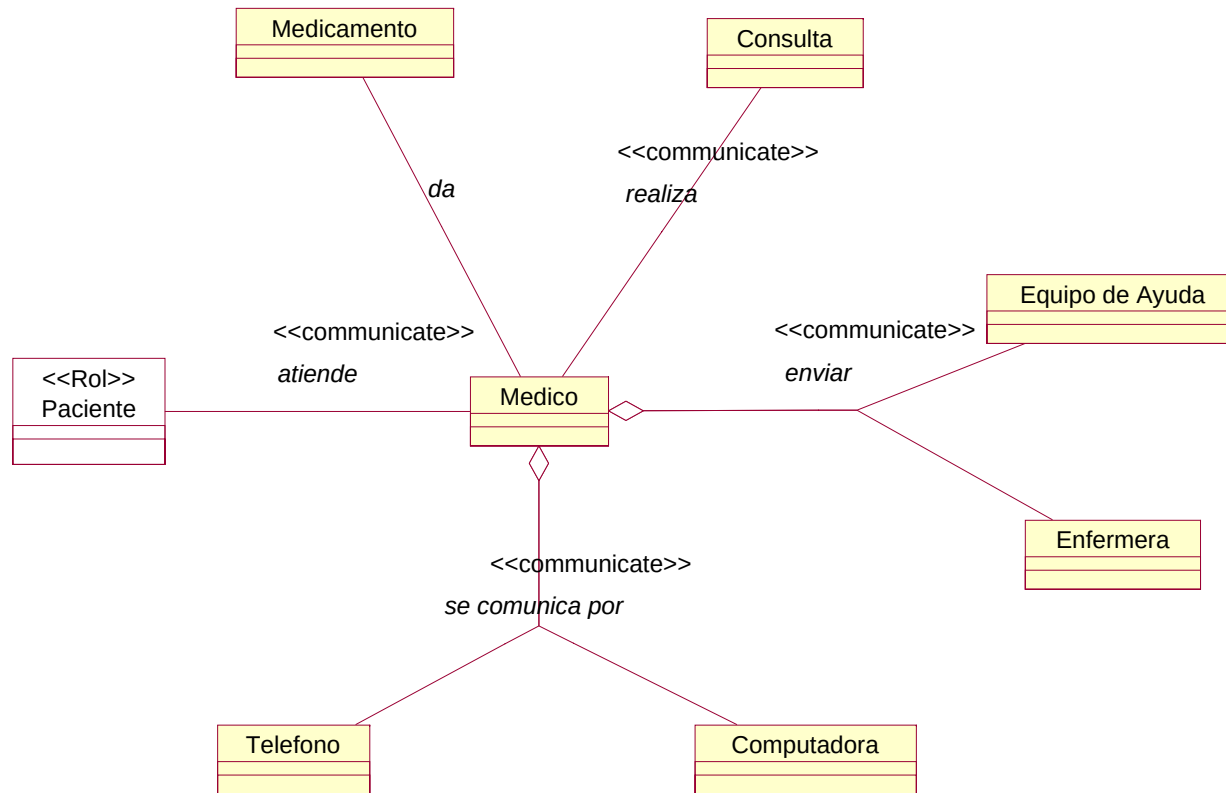
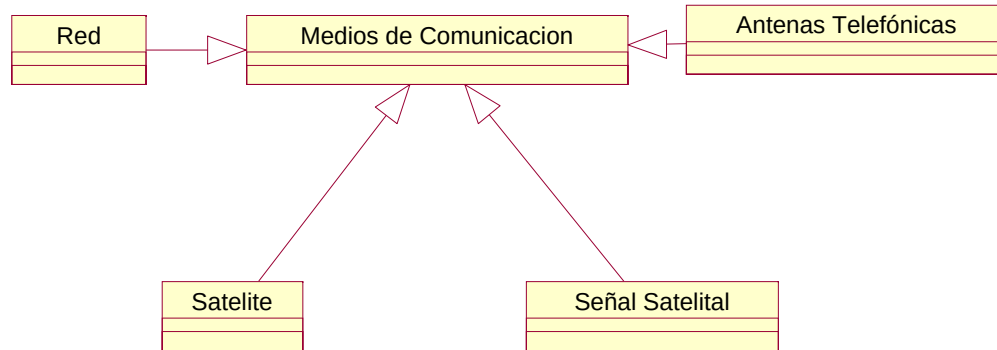


Diagrama de clases "Medios de comunicación"



Diagramas de clases "Monitor"

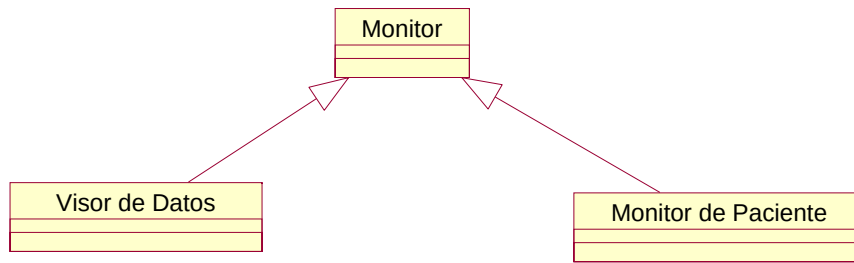
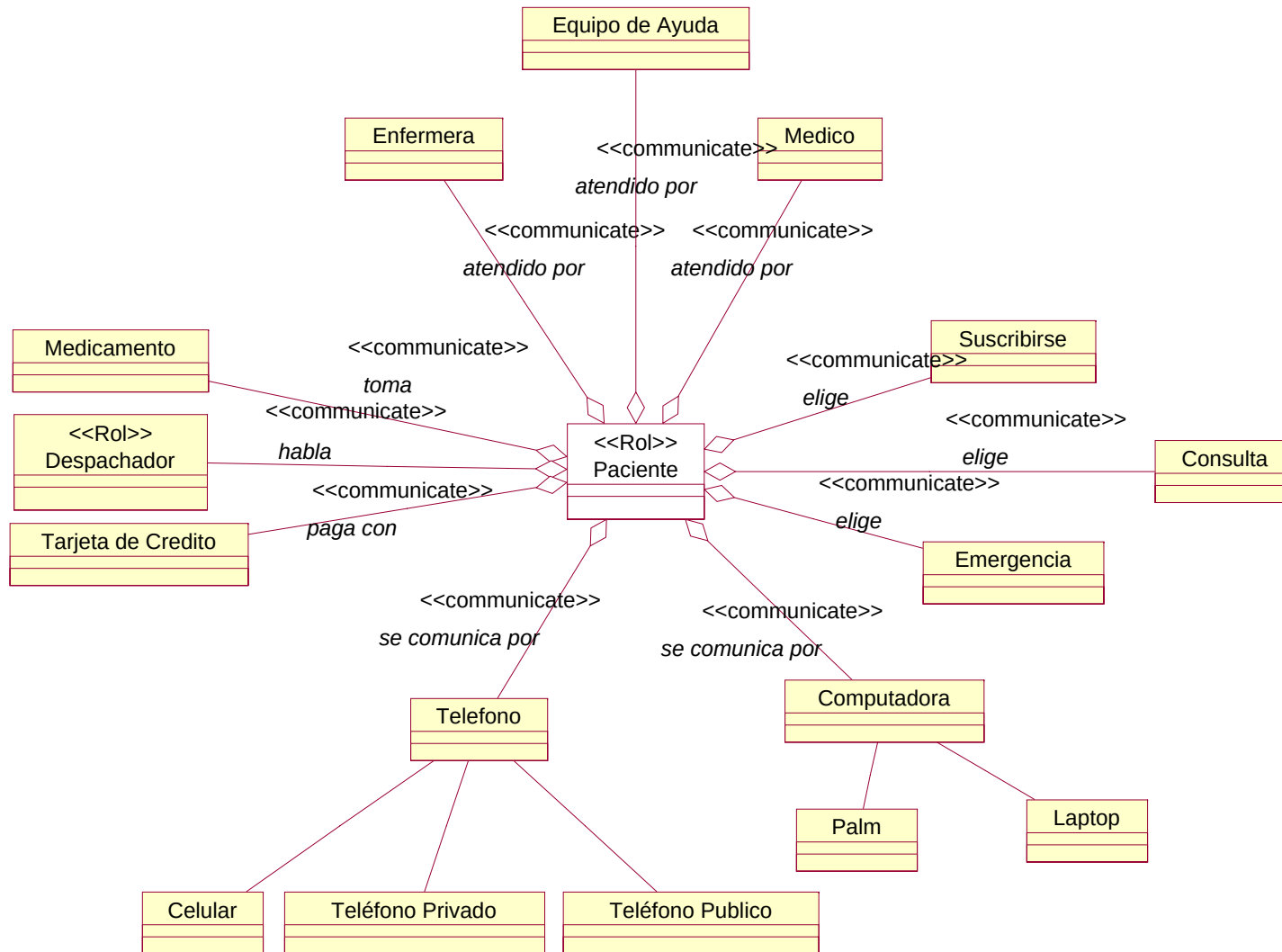


Diagrama de clases "Paciente"



Modelo Conceptual

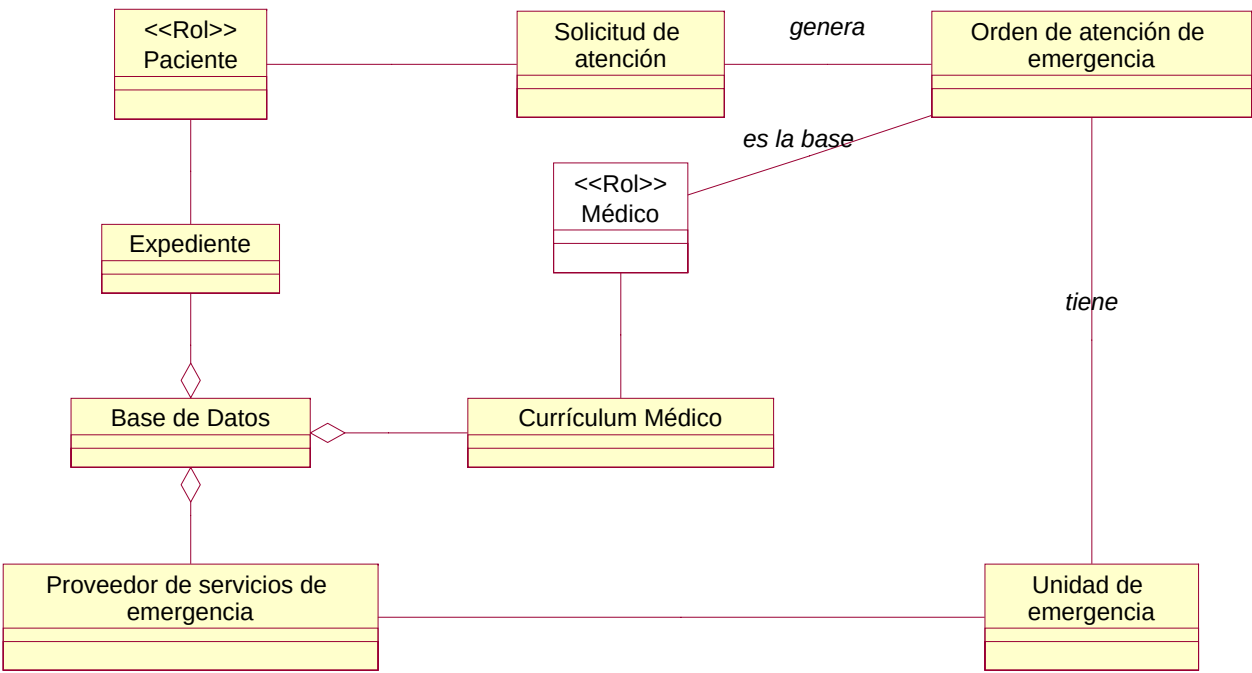


Diagrama de clases "Tipo de servicio"

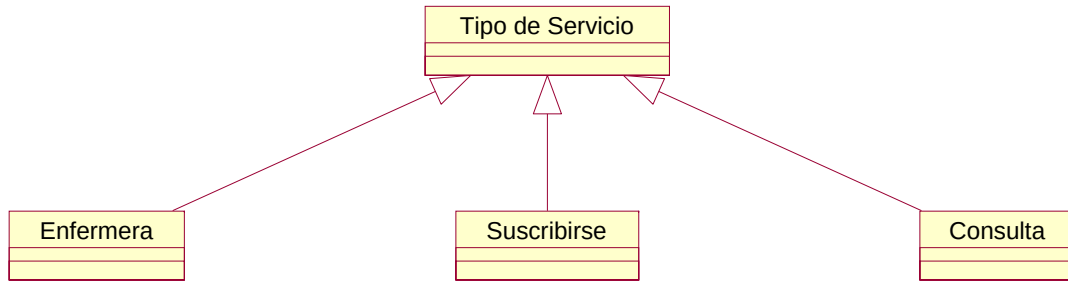


Diagrama de clases "Transacciones"

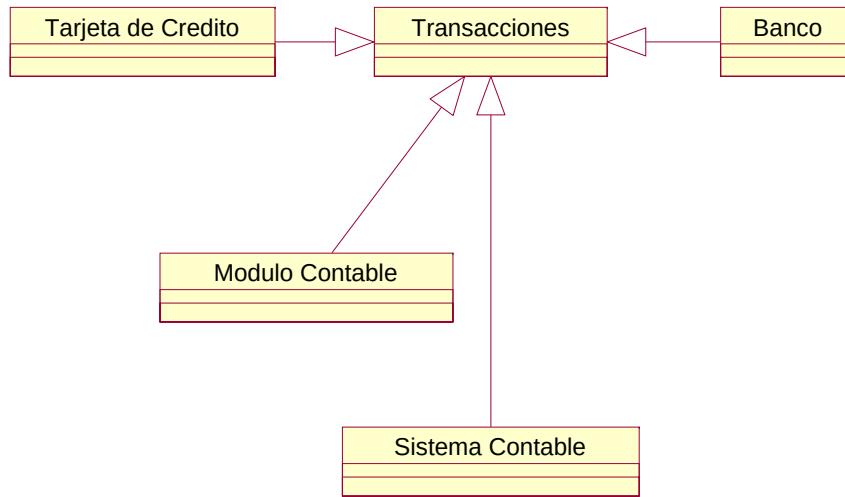
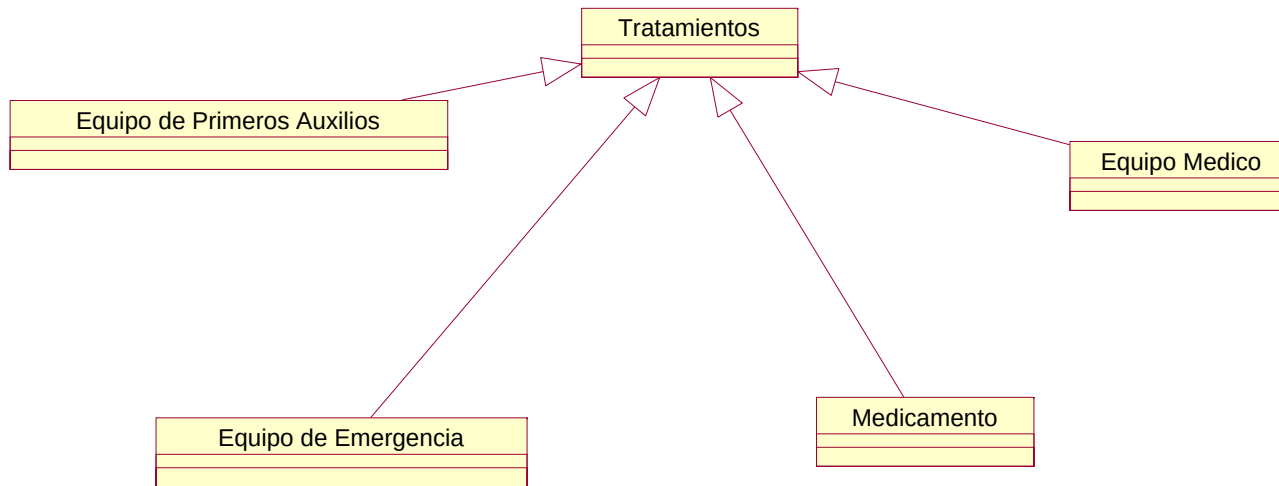


Diagrama de clases "Tratamientos"



Diagramas de Colaboración

Diagrama de colaboración "Acceso Despachador"

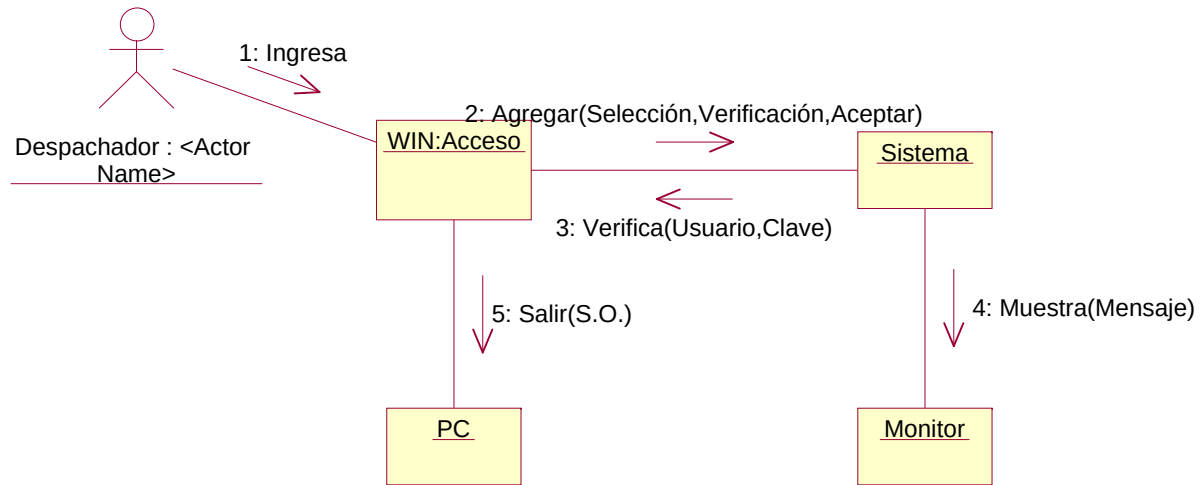


Diagrama de colaboración "Acceso Paciente"

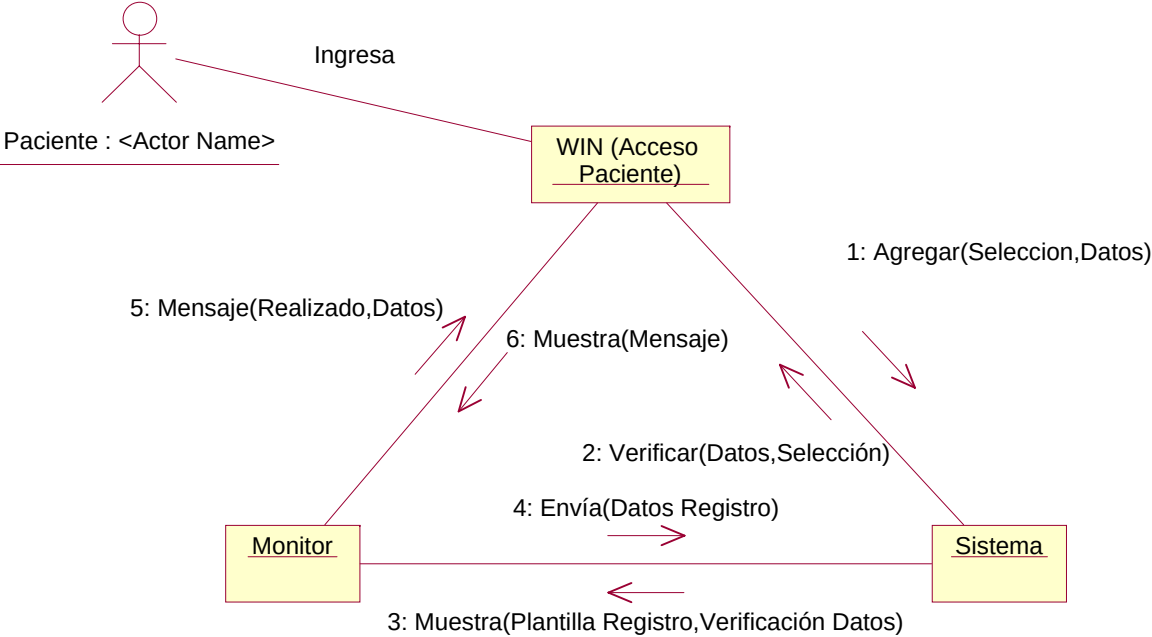


Diagrama de Colaboración "Acceso sistema"

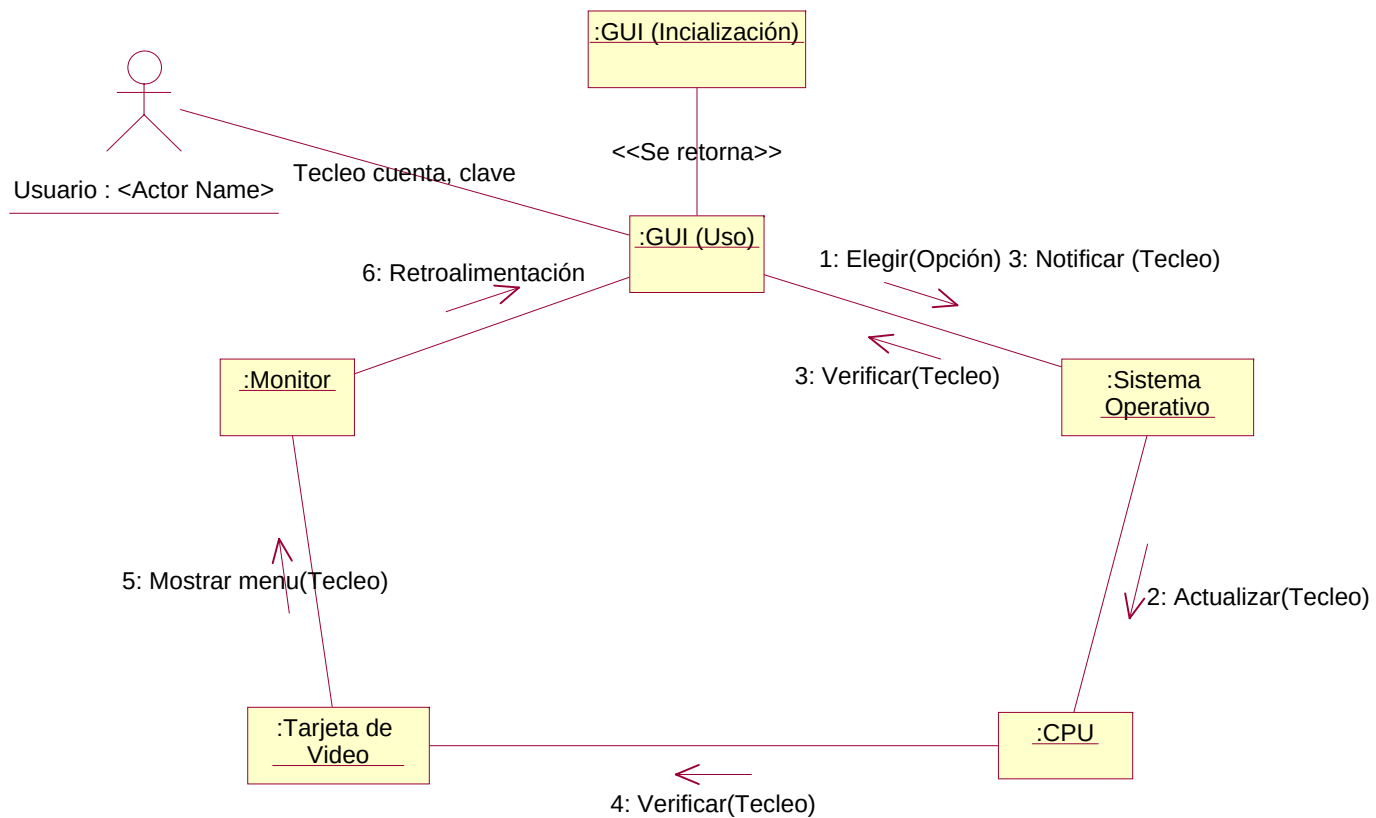


Diagrama de colaboración "Cambiar contraseña"

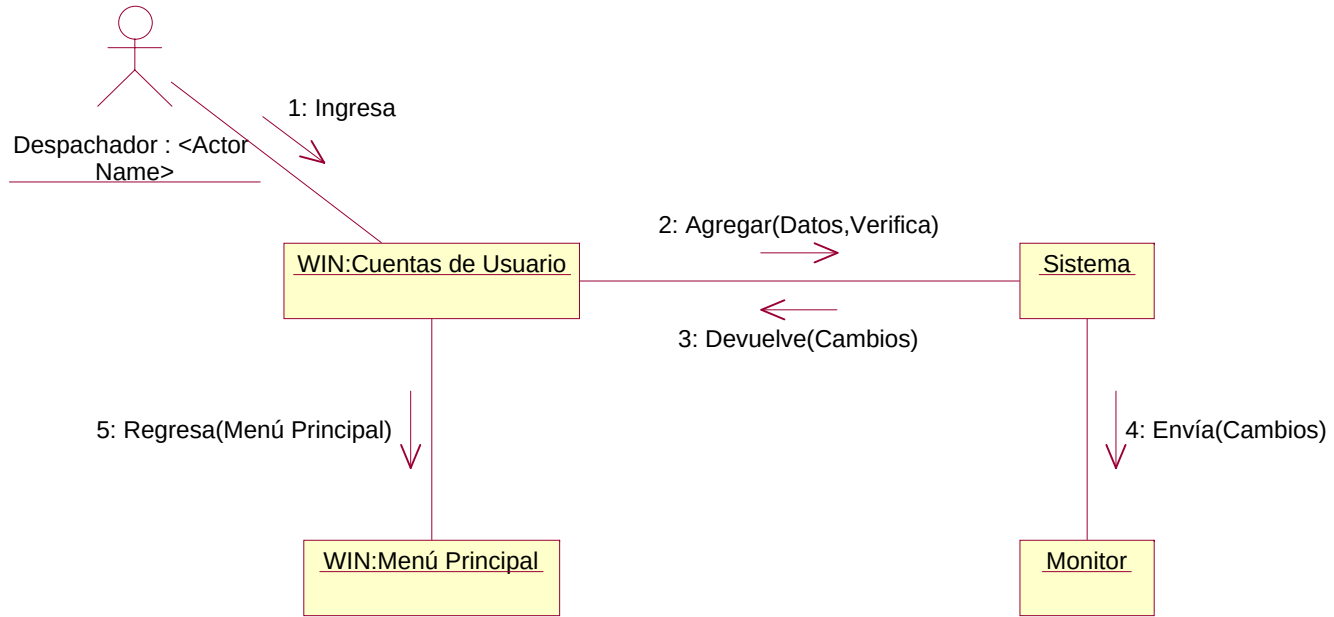


Diagrama de colaboración "Consulta"

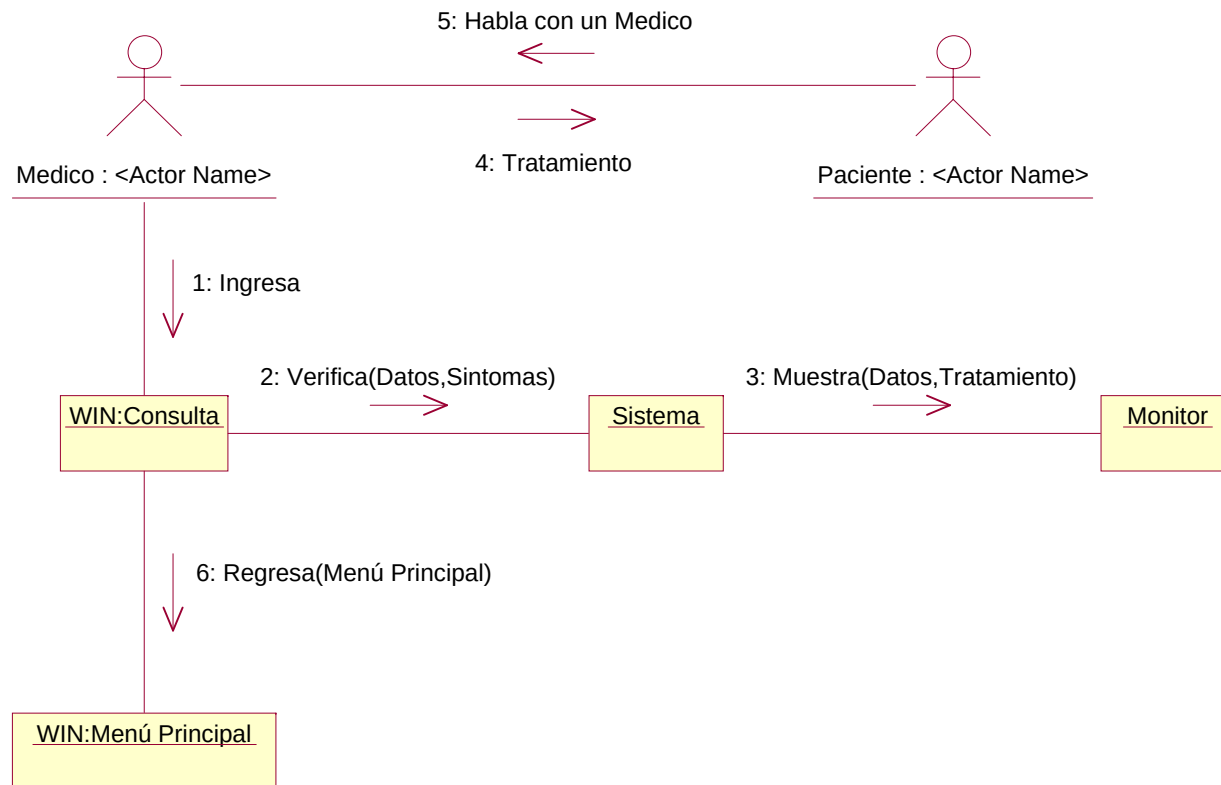
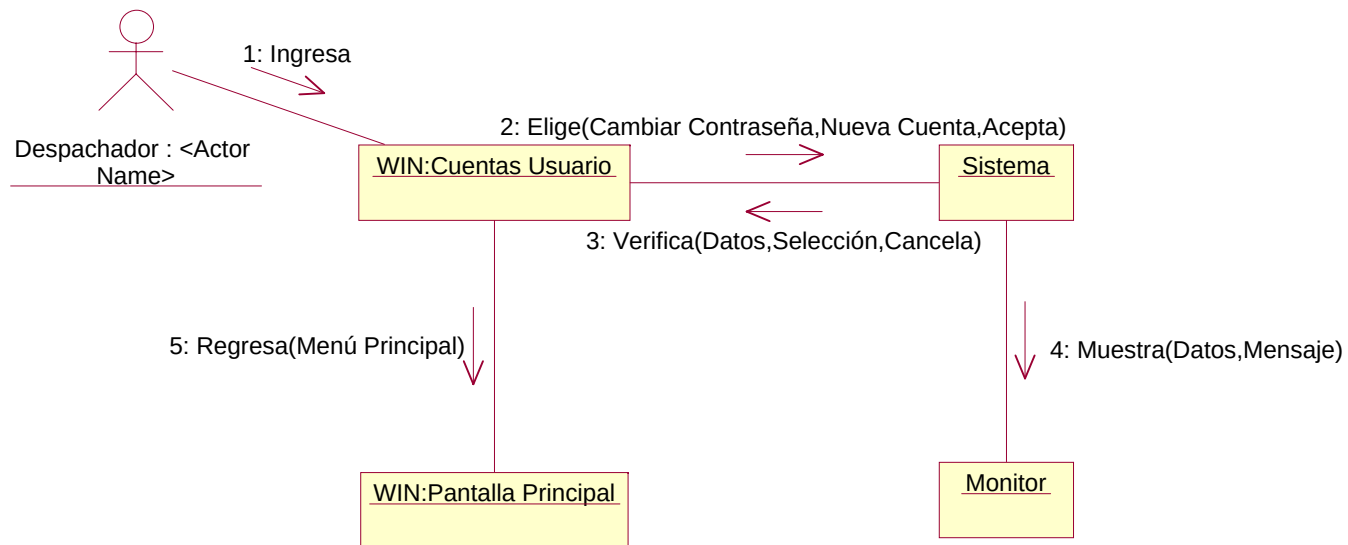


Diagrama de colaboración "Cuenta de usuario"



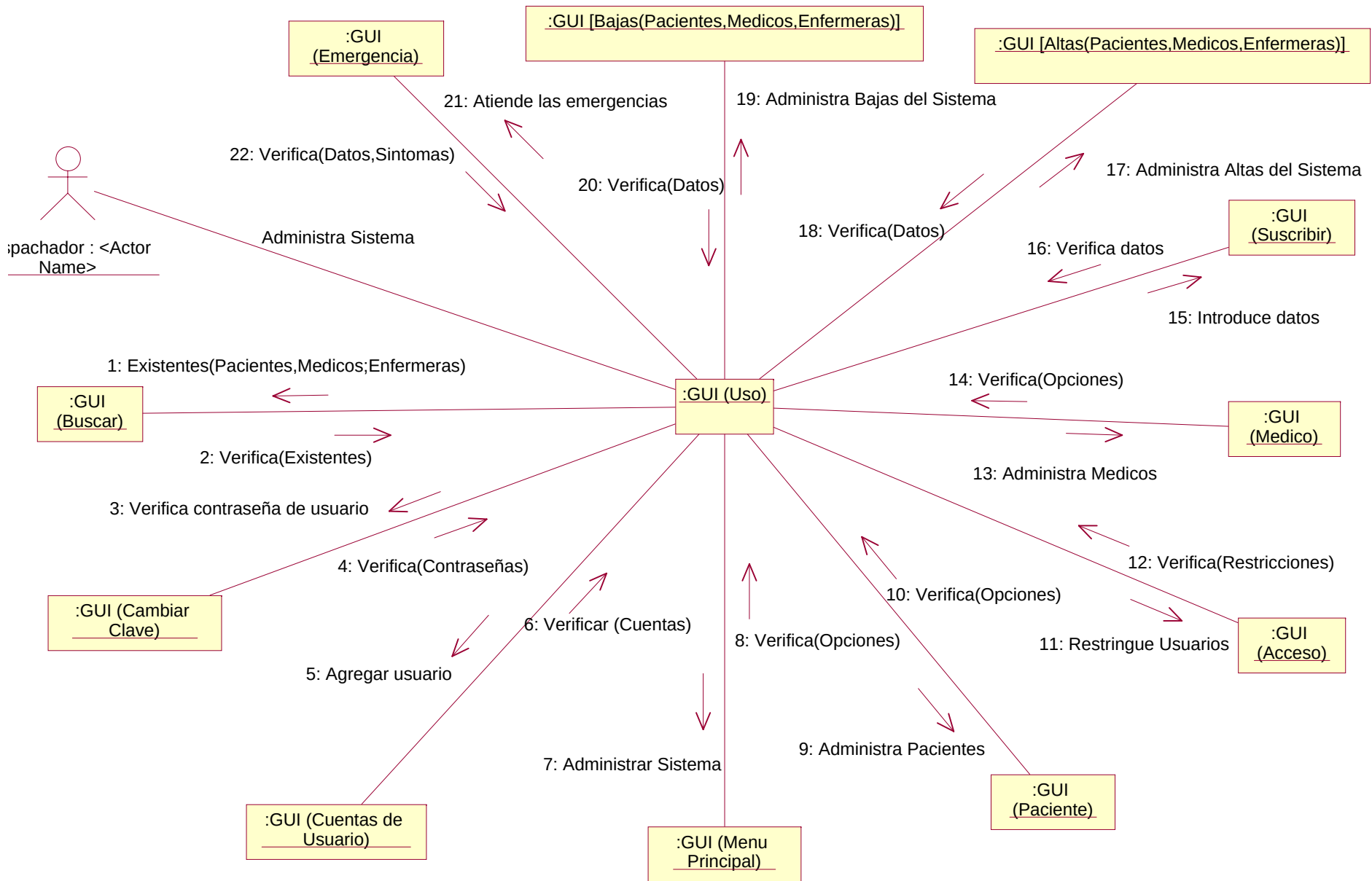


Diagrama de colaboración "Médico"

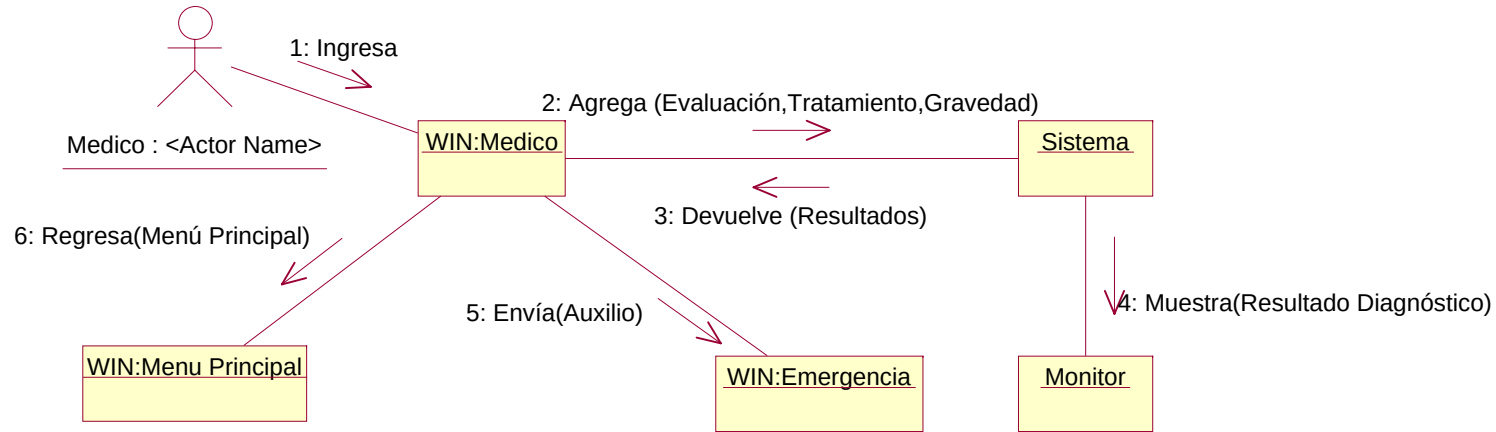


Diagrama de colaboración "Paciente"

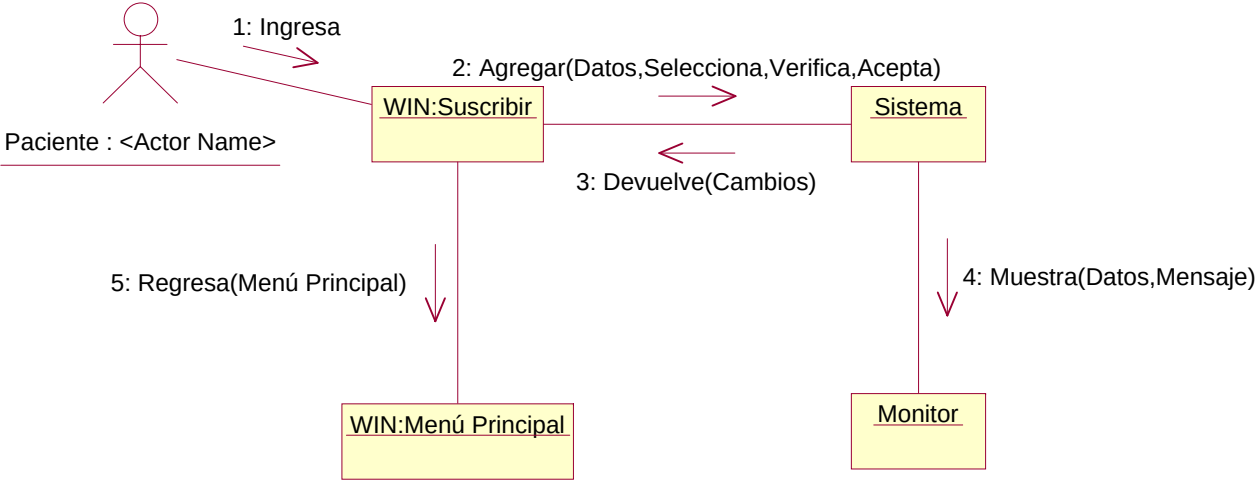
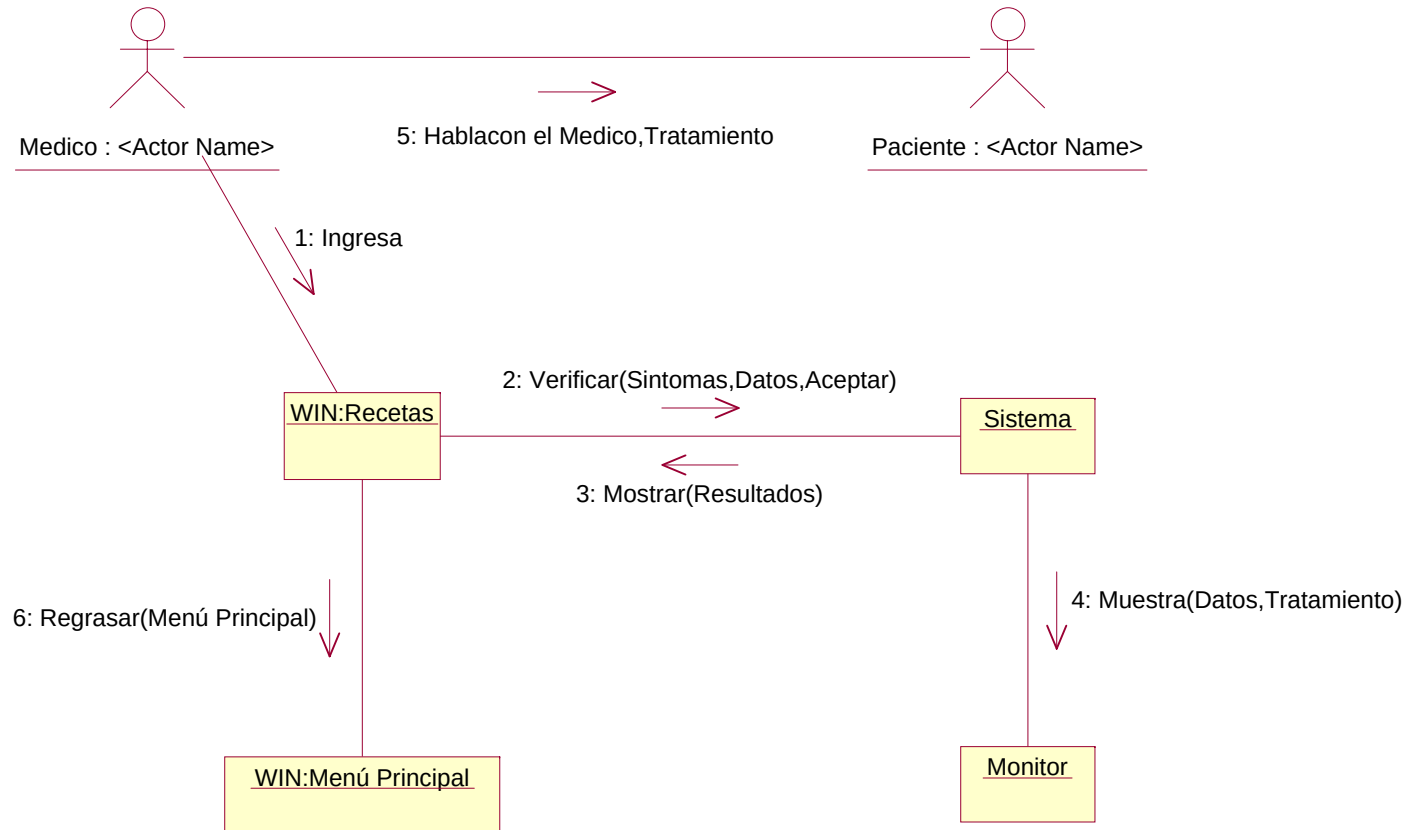


Diagrama de colaboración "Recetas"



Diagramas de Secuencia

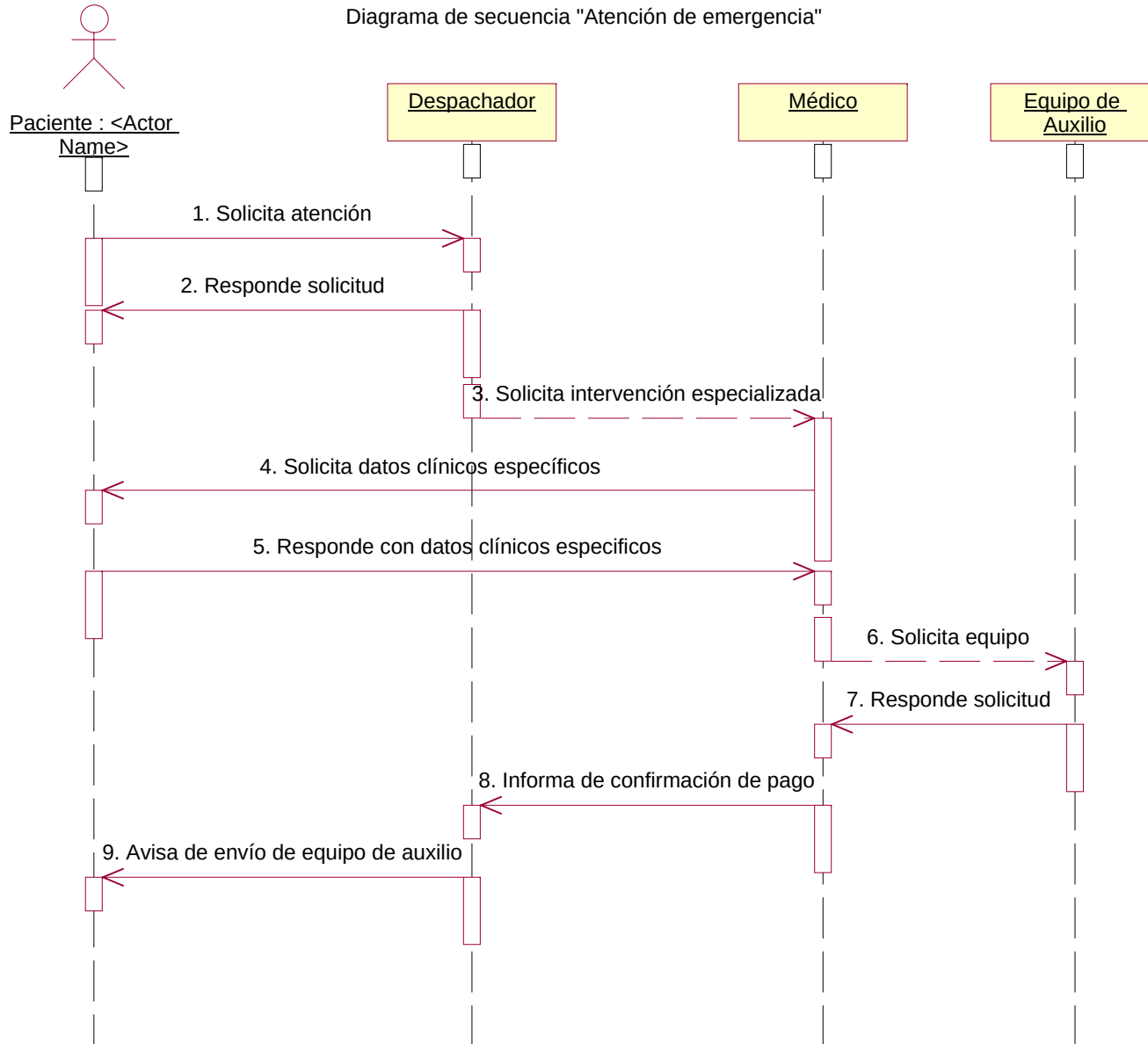
Pacie...

Despachador

Médico

Equipo de Auxilio

Diagrama de secuencia "Atención de emergencia"



Desp...

WIN:Acceso

:BD

Diagrama de secuencia "Acceso"

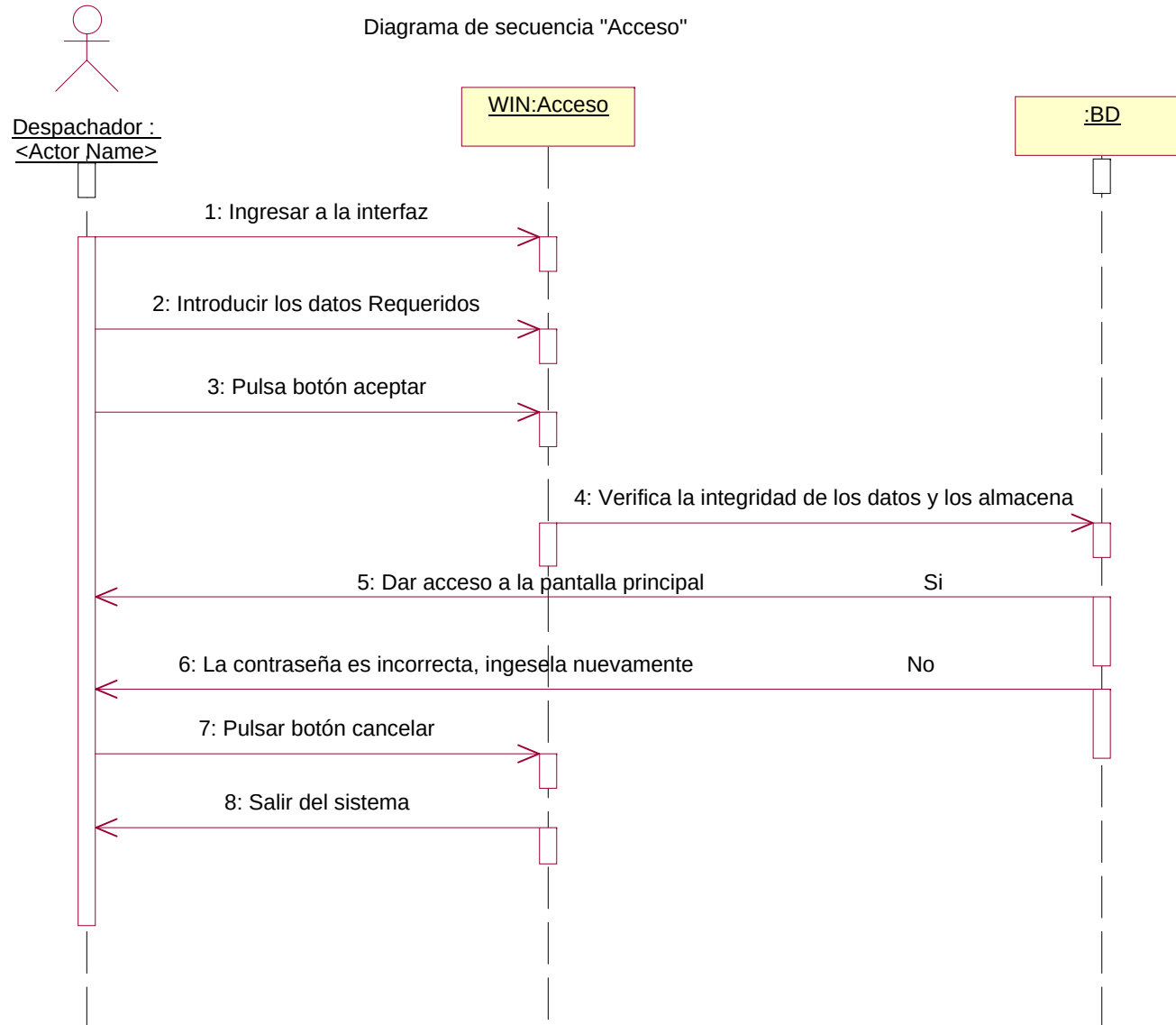


Diagrama de secuencia ""Cambiar Contraseña

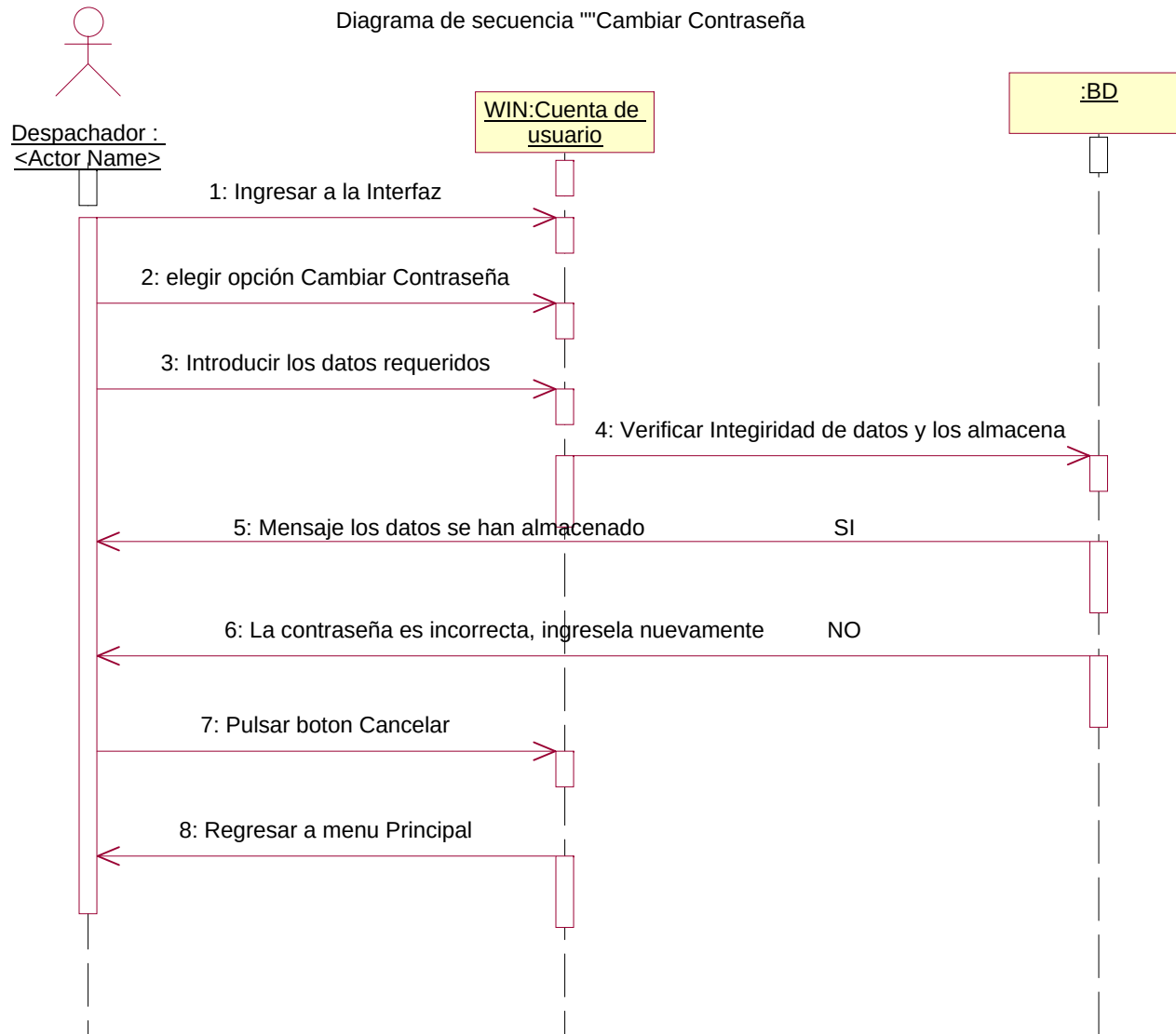
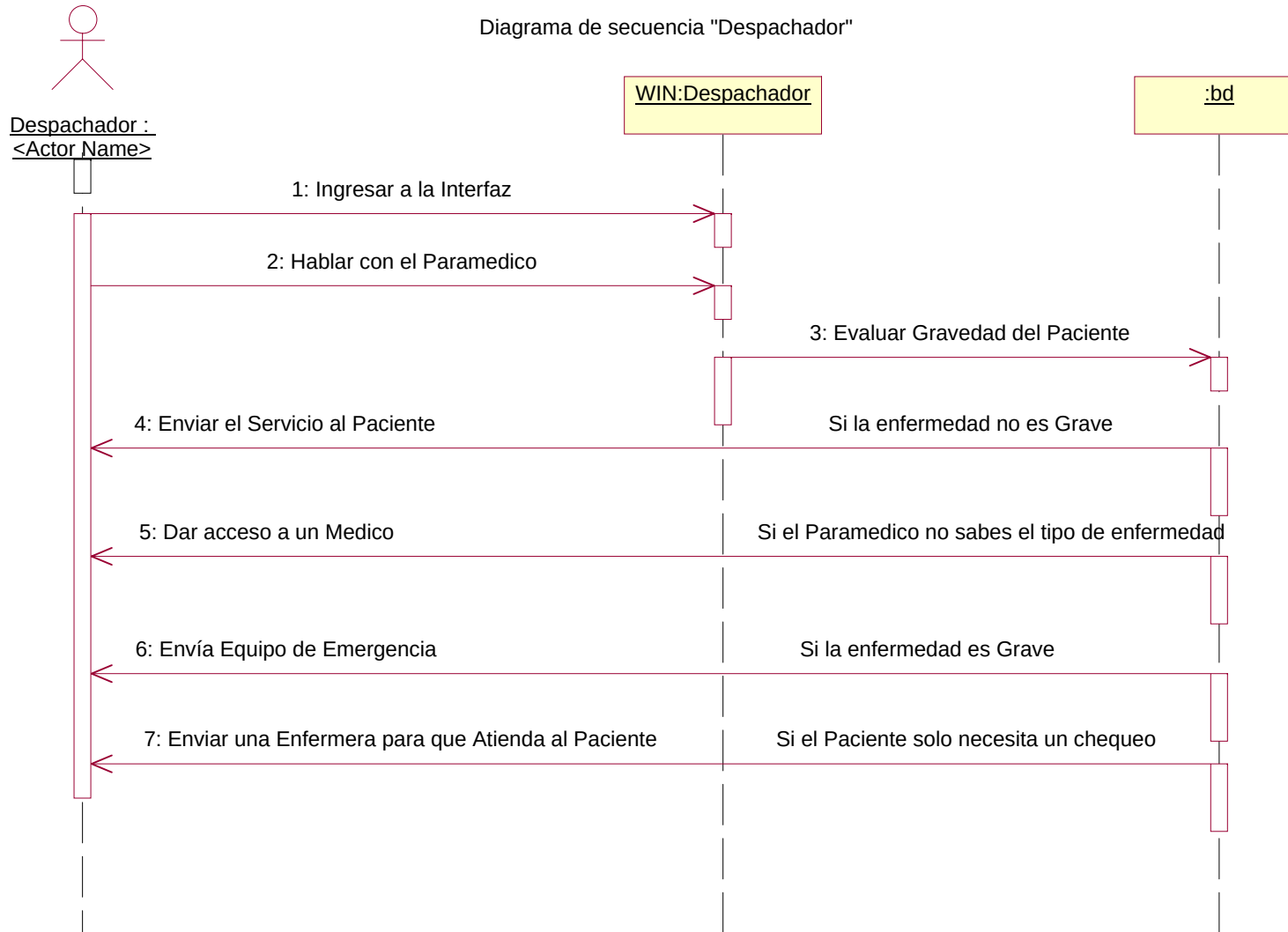


Diagrama de secuencia "Despachador"



Medic...

WIN:Medico

:db

Diagrama de secuencia "Médico"

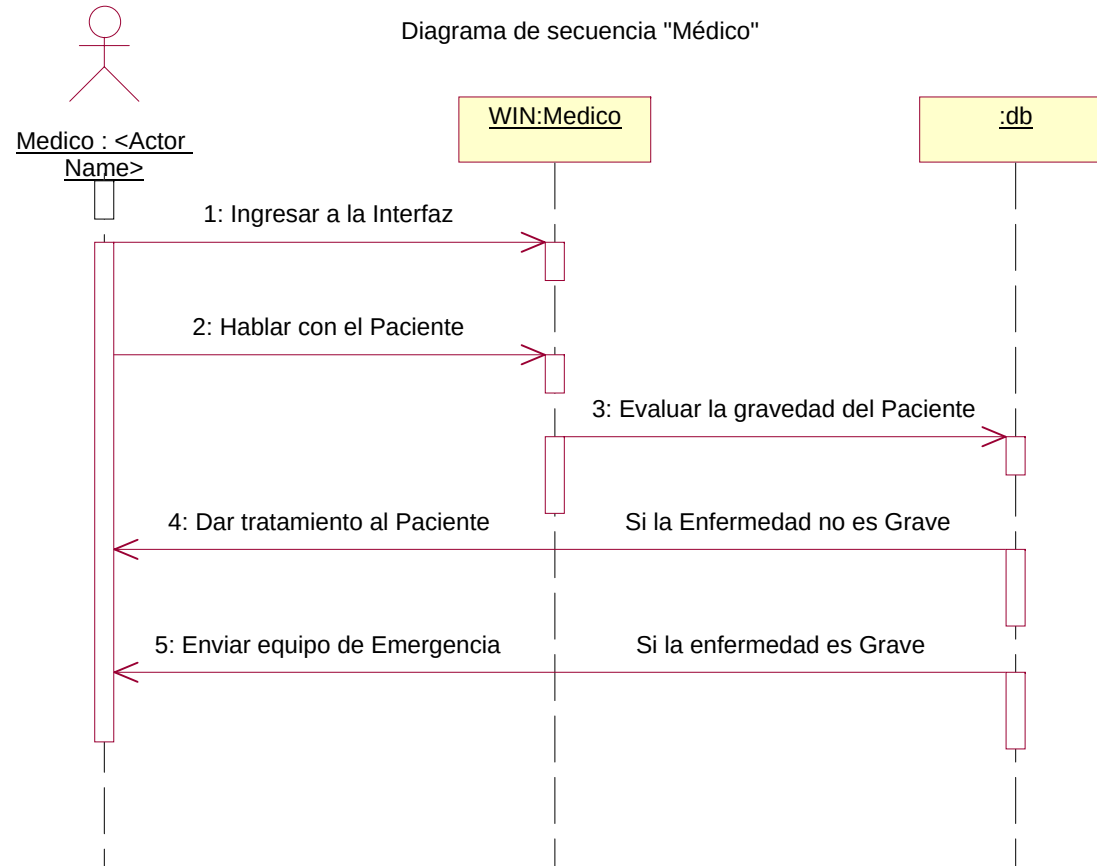
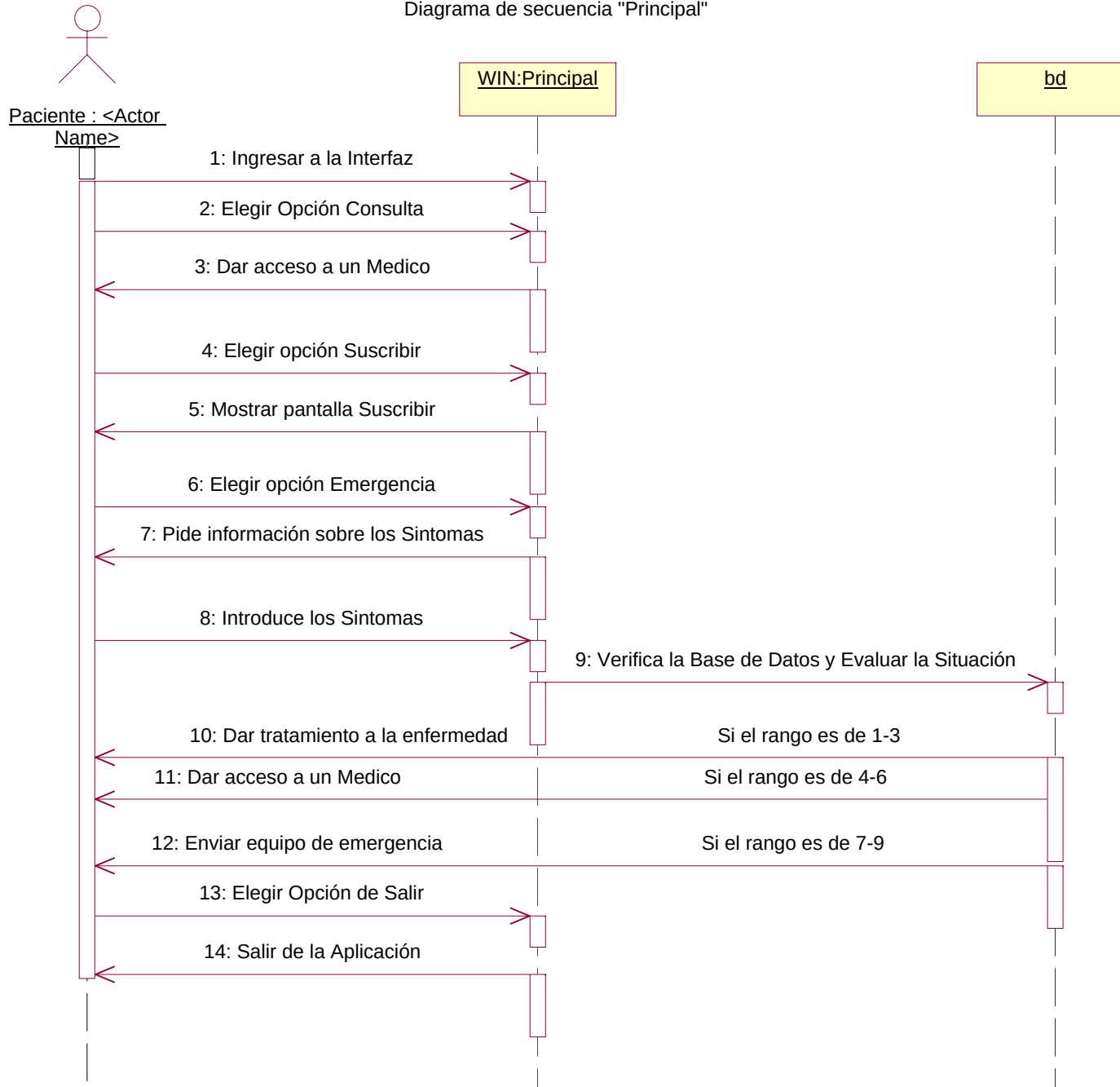


Diagrama de secuencia "Principal"



Medic...

Pacie...

WIN:Recetas

:BD

Diagrama de secuencia "Recetas"

Medico : <Actor
Name>

Paciente : <Actor
Name>

WIN:Recetas

:BD

1: Ingresar a la Interfaz

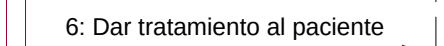
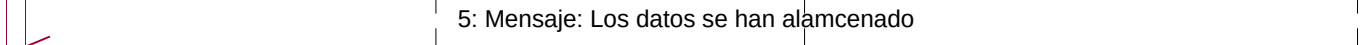
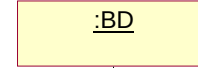
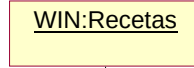
2: Hablar y Evaluar la Situación

3: Introducir los datos requeridos

4: Verificar los datos ingresados y los almacena

5: Mensaje: Los datos se han almacenado

6: Dar tratamiento al paciente

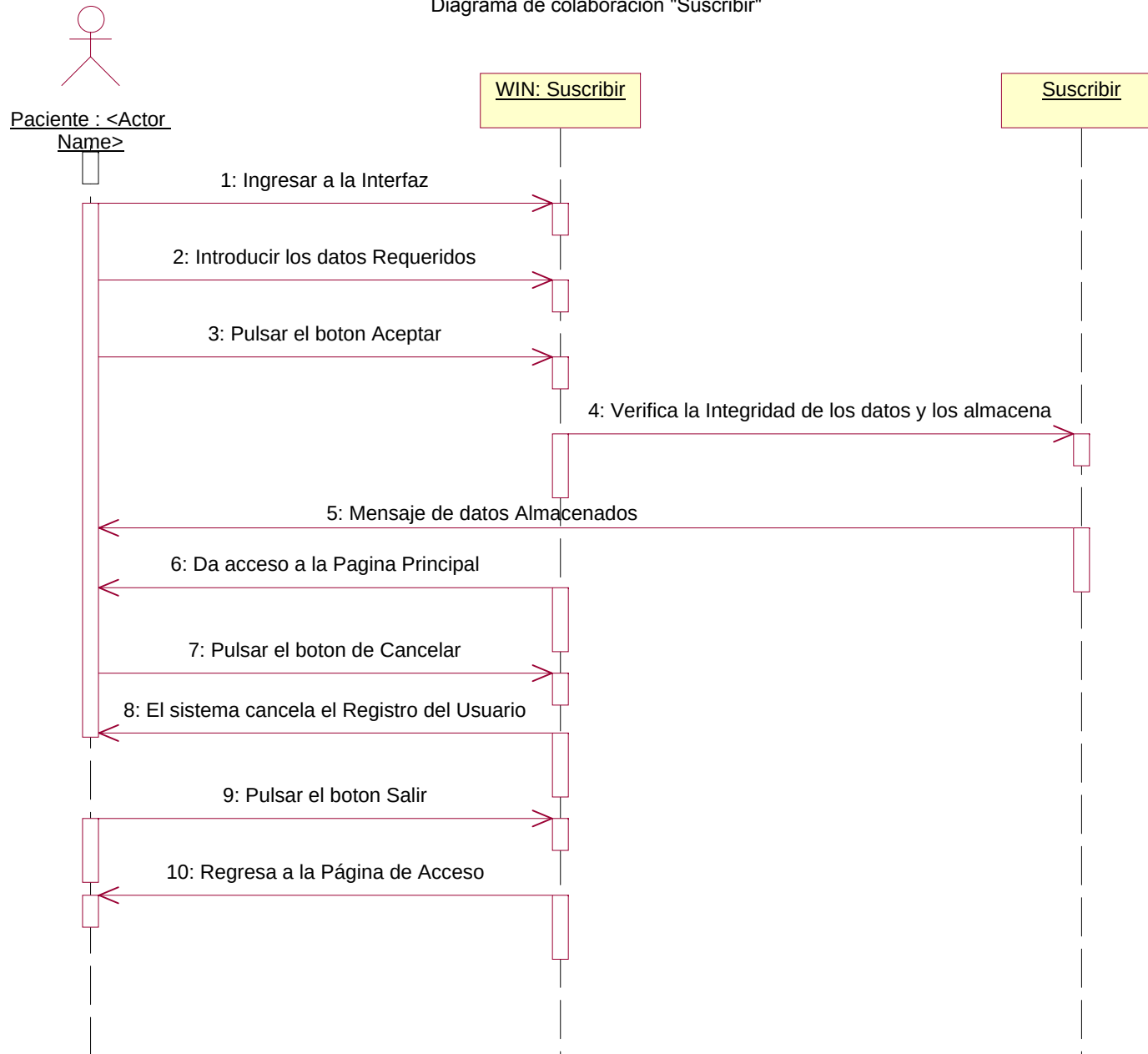


Pacie...

WIN: Suscribirse

Suscribir

Diagrama de colaboración "Suscribir"



Diagramas de Estados

Diagrama de estados "Búsqueda"

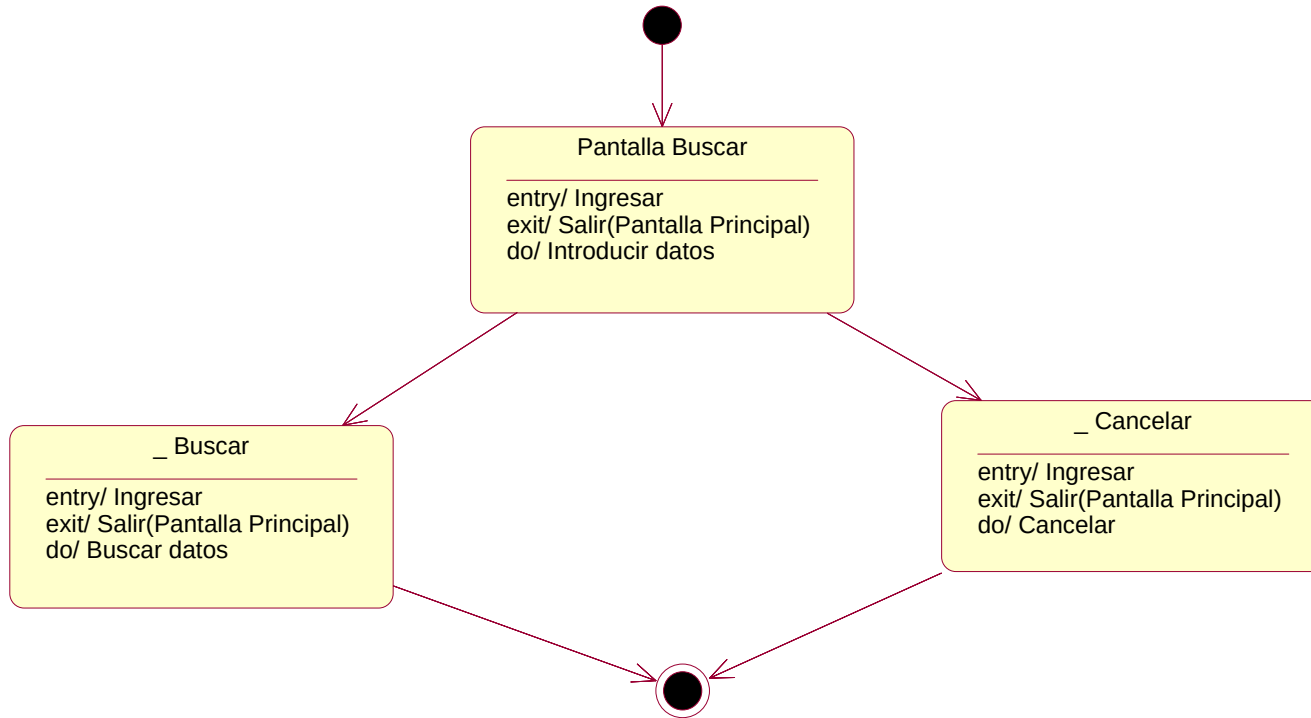


Diagrama de estados "Cambiar contraseña"

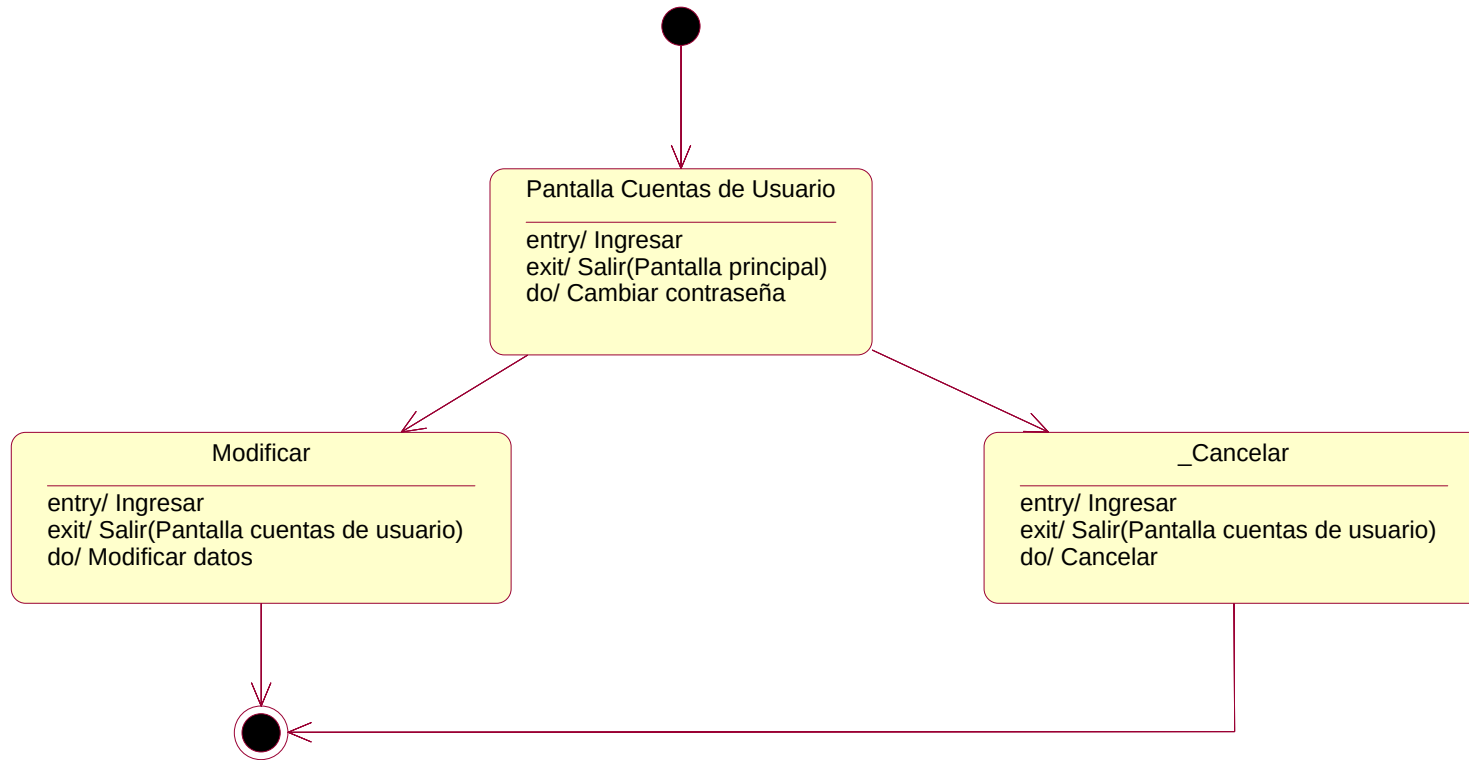


Diagrama de estados "Consultas"

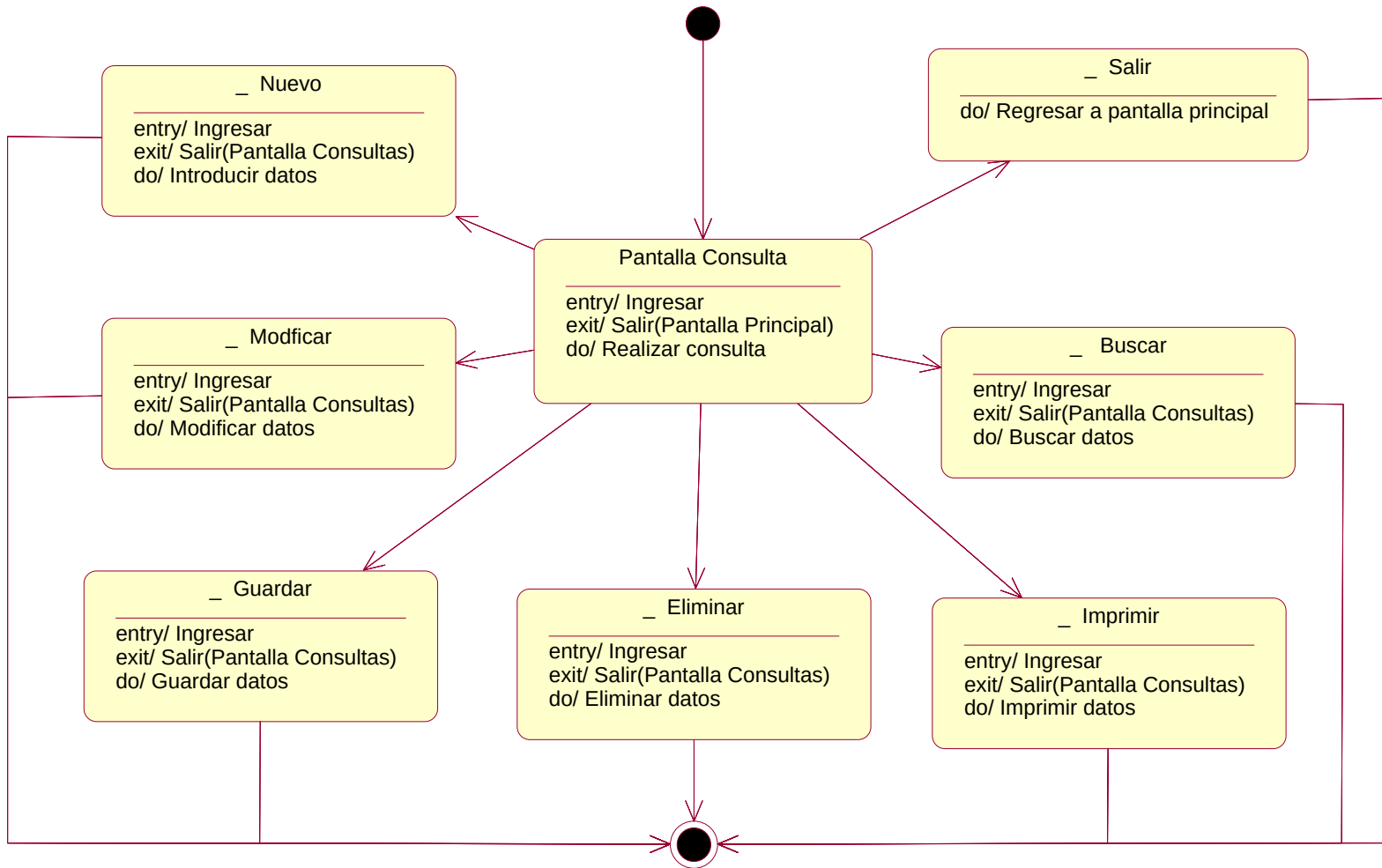


Diagrama de estados "Cuentas de usuario"

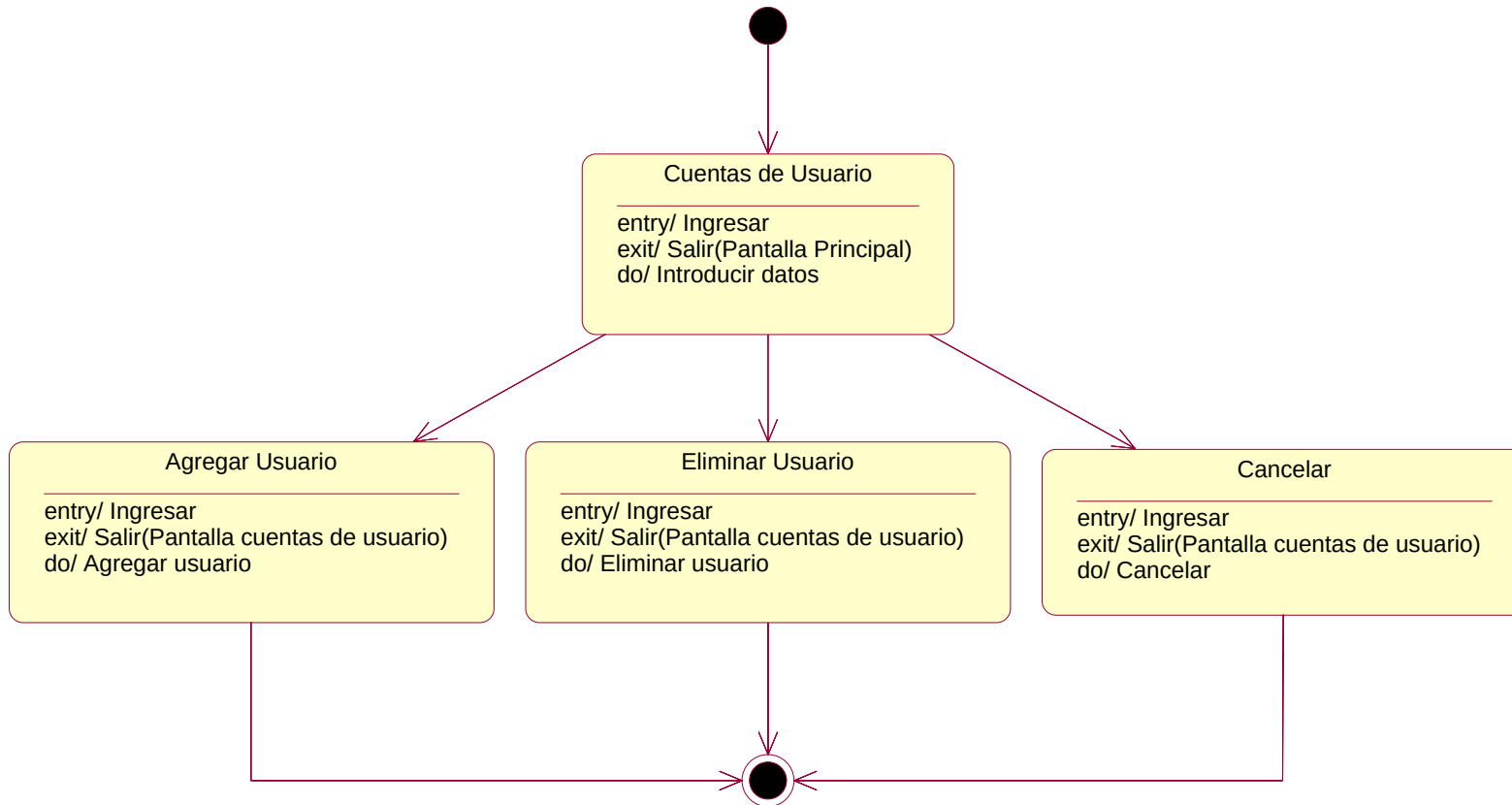


Diagrama de estados "Enfermeras"

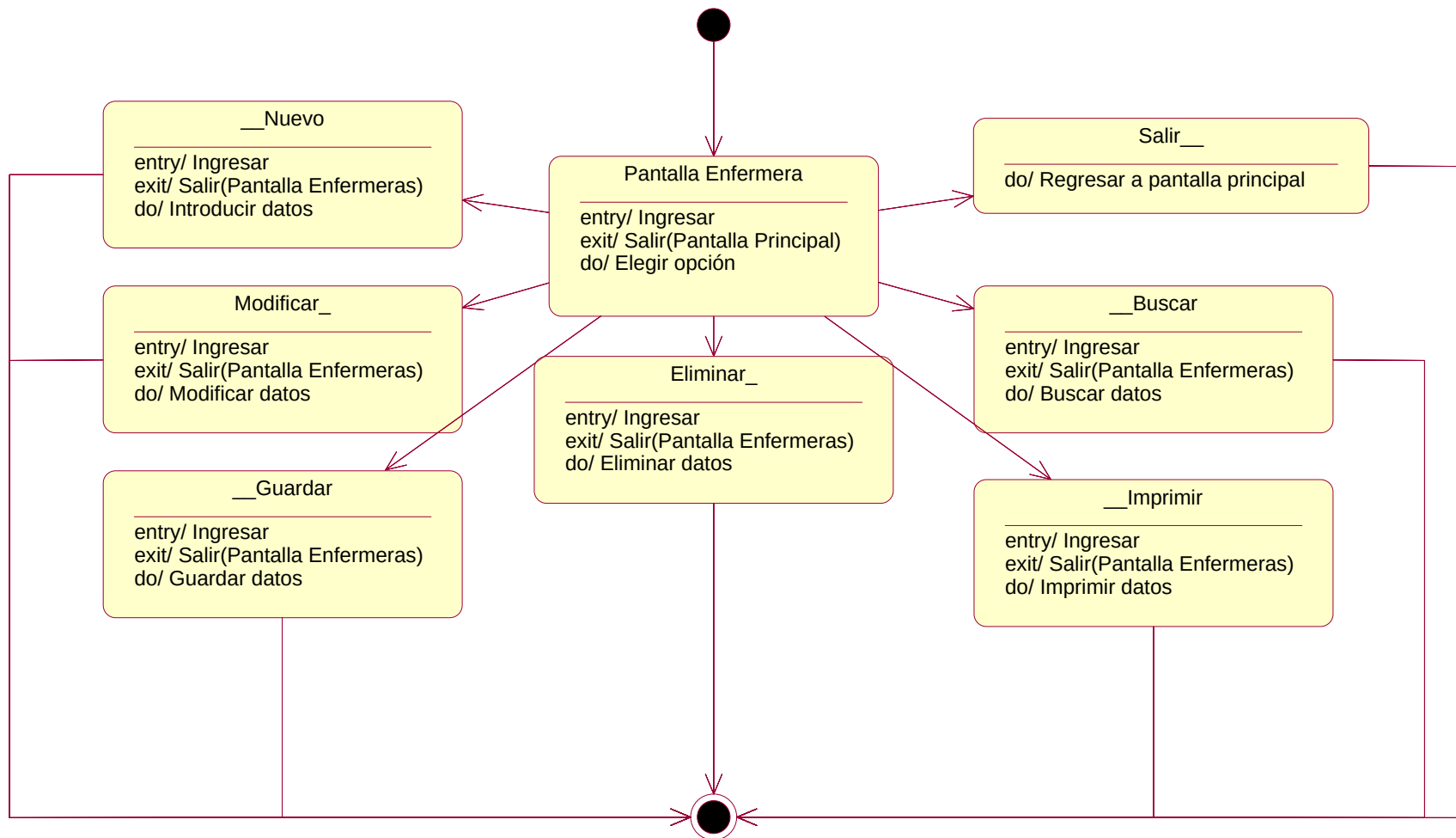


Diagrama de estados "Médico"

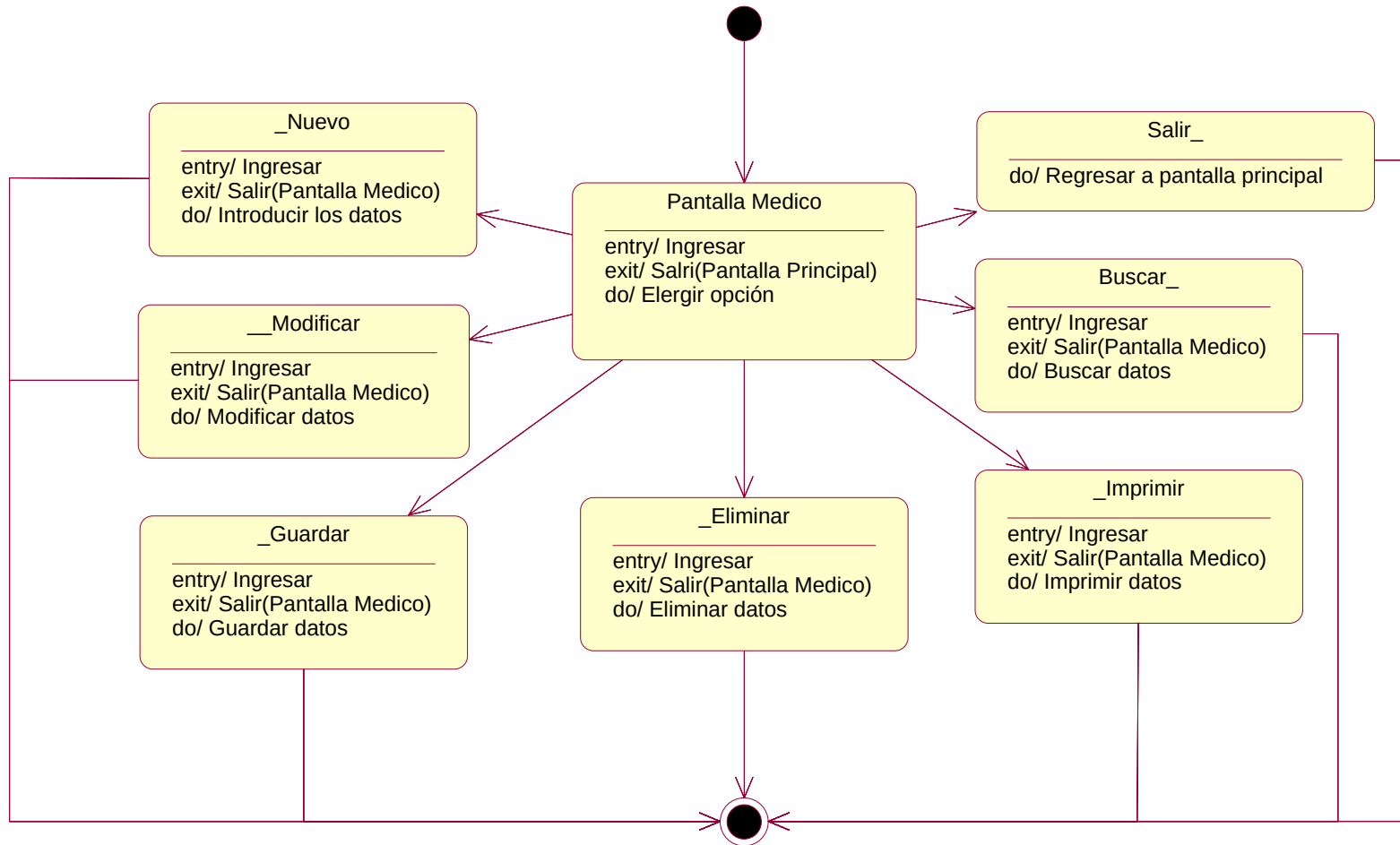


Diagrama de estados Paciente

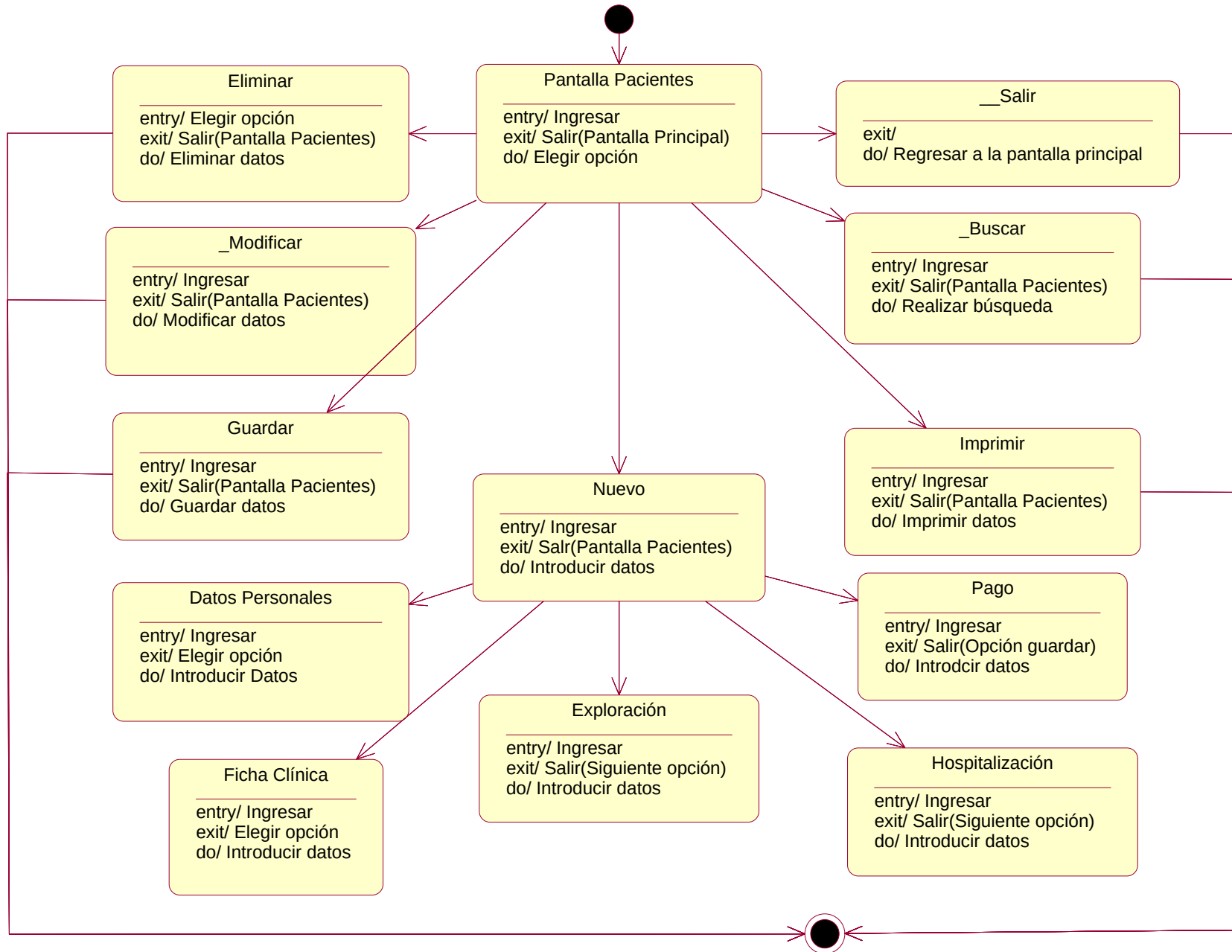


Diagrama de estados "Recetas"

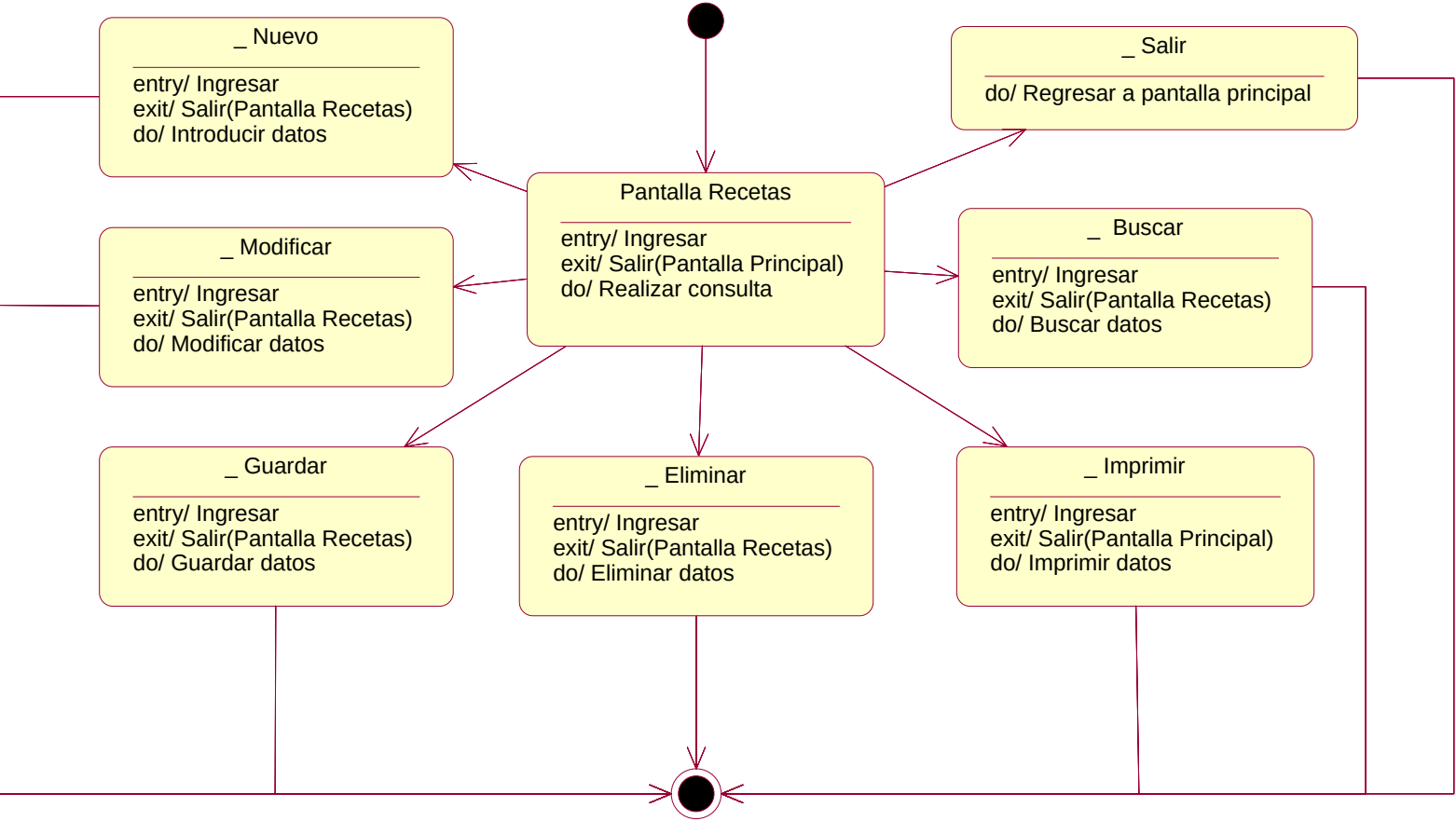
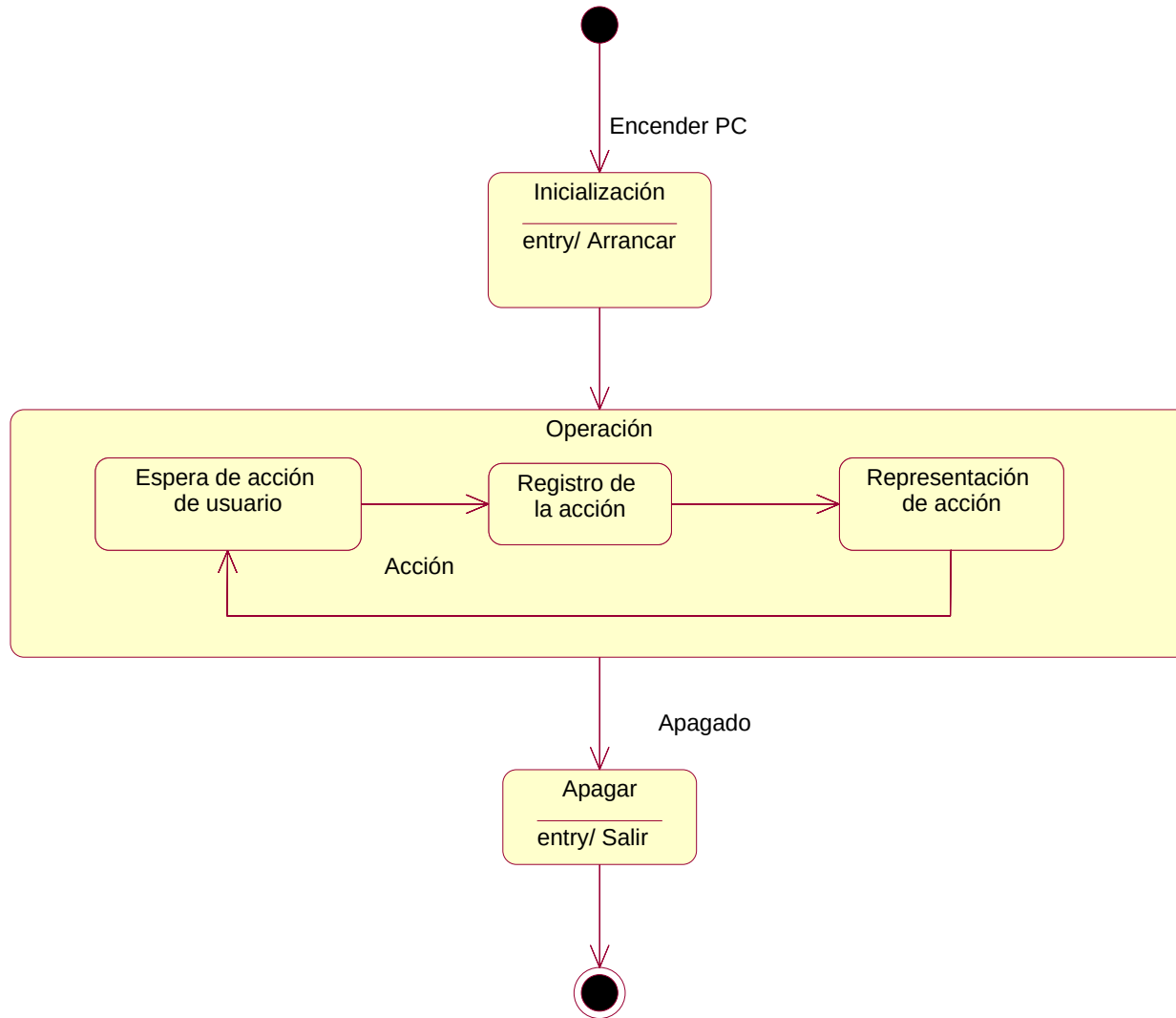


Diagrama de estados "Transición de einterfaz"



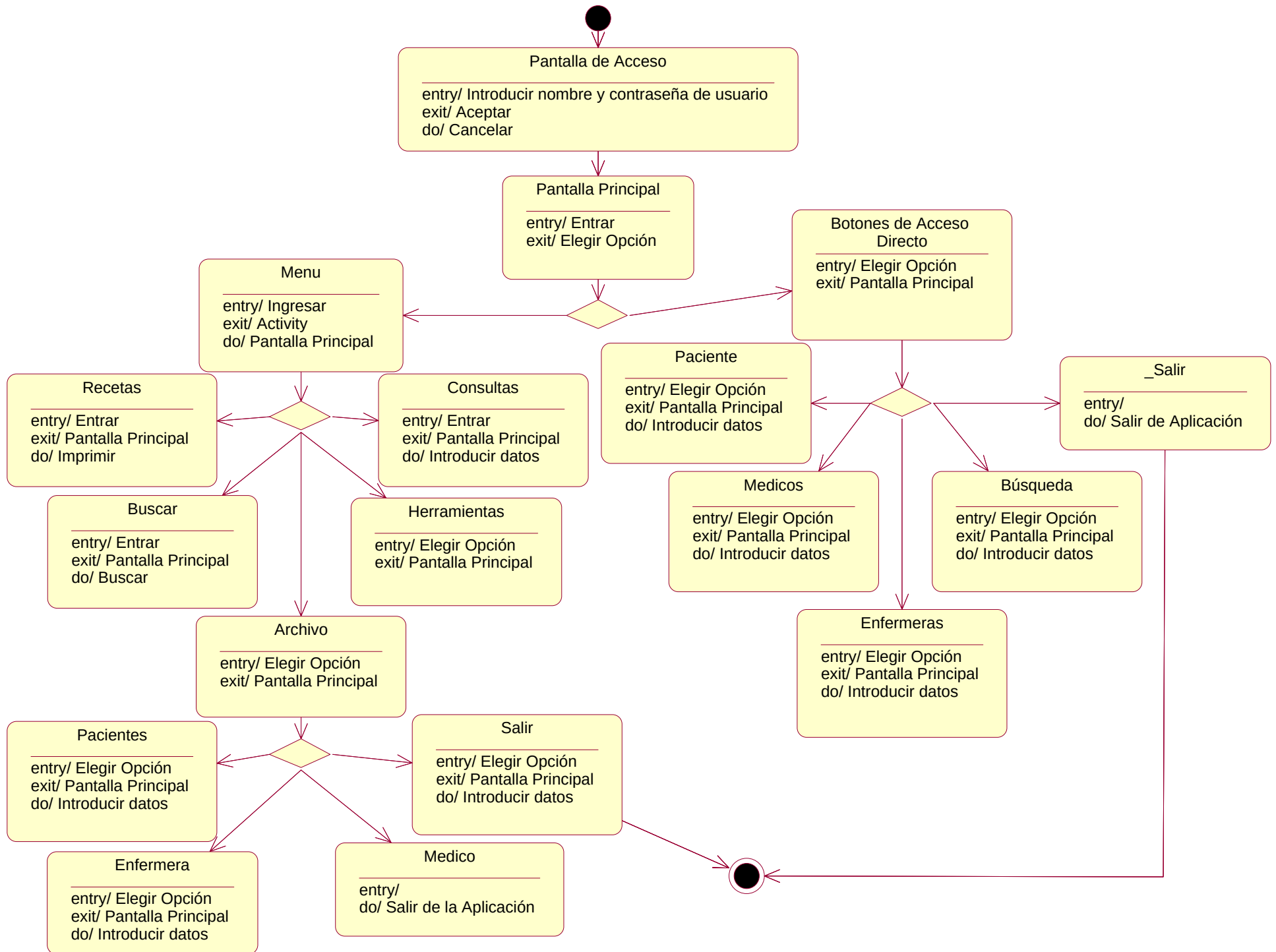
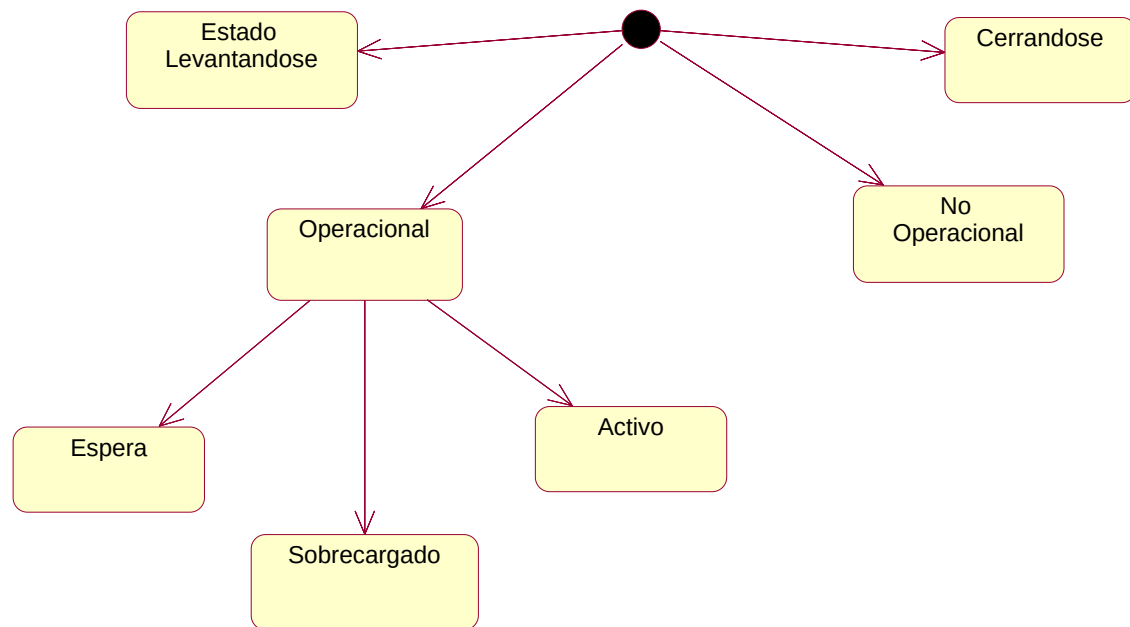


Diagrama de estados "Estados de TRMCS"



Diagramas de Actividades

Diagrama de actividades "Consulta"

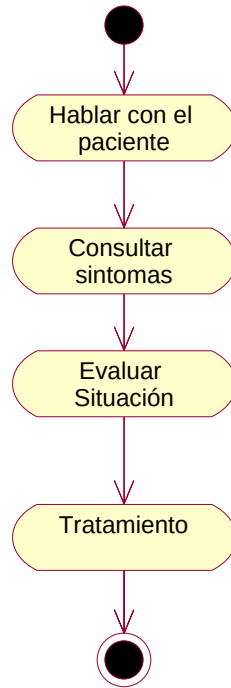


Diagrama de actividades "Altas"

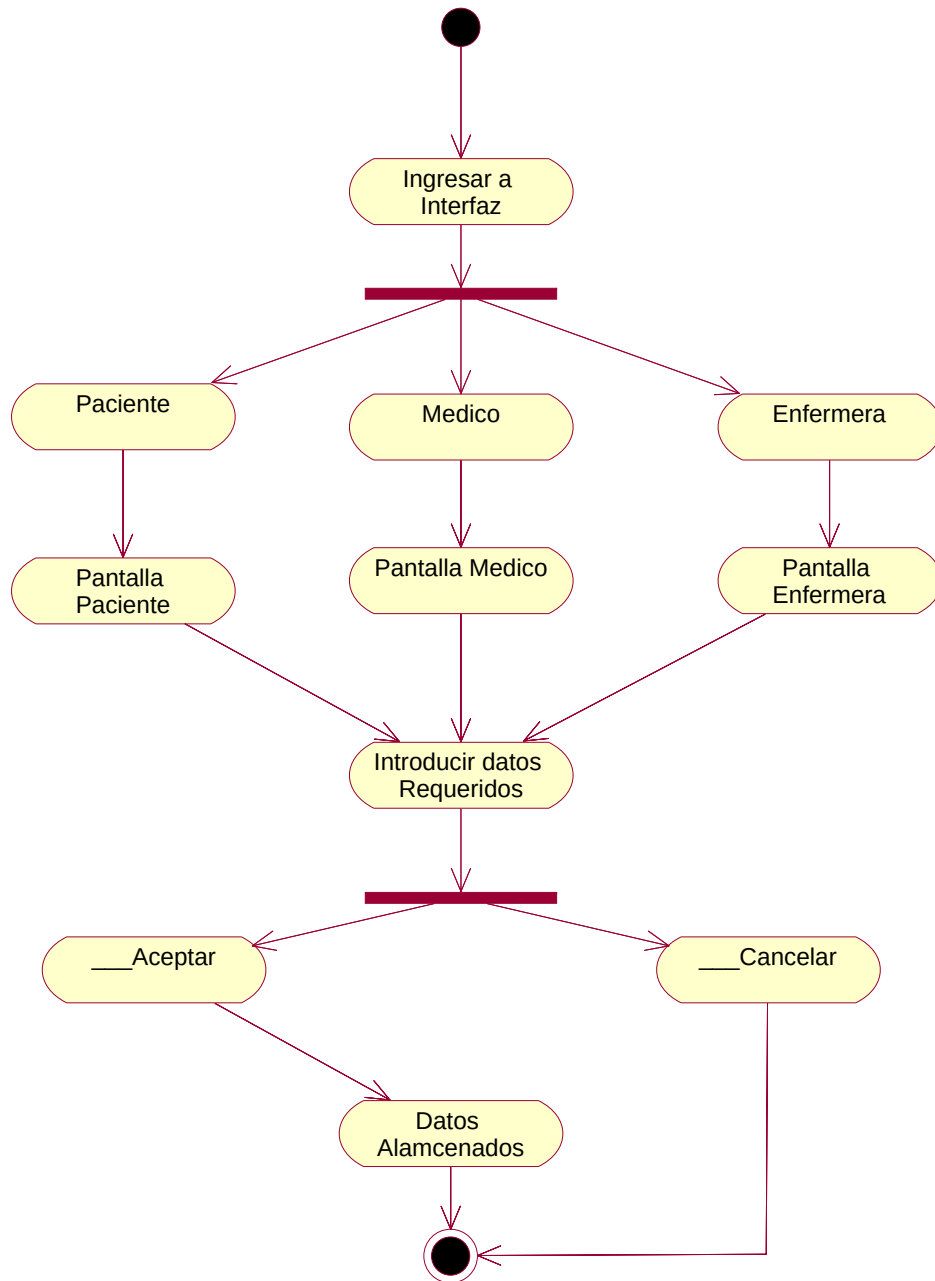


Diagrama de actividades "Bajas"

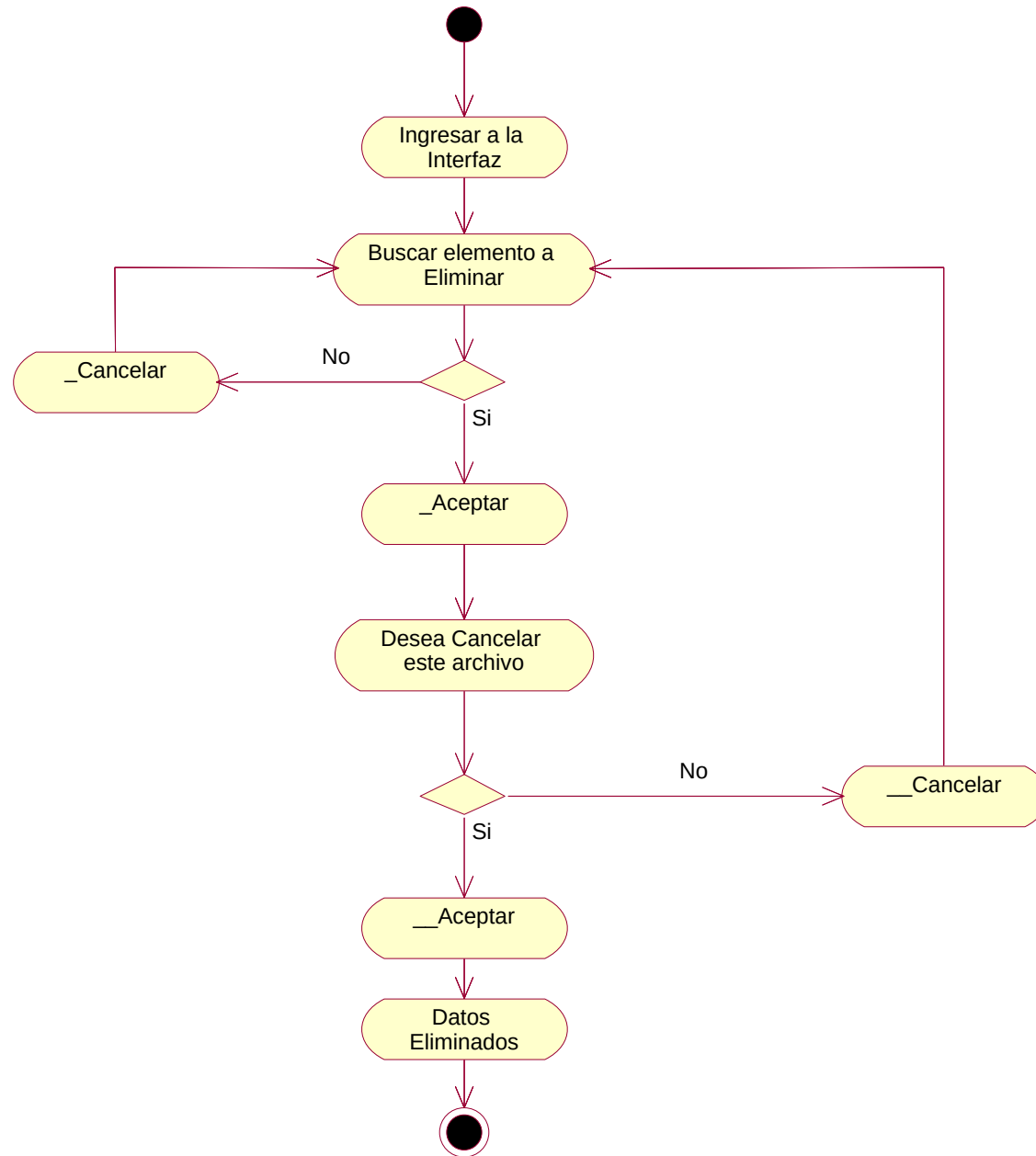


Diagrama de actividades "Cargar sistema"

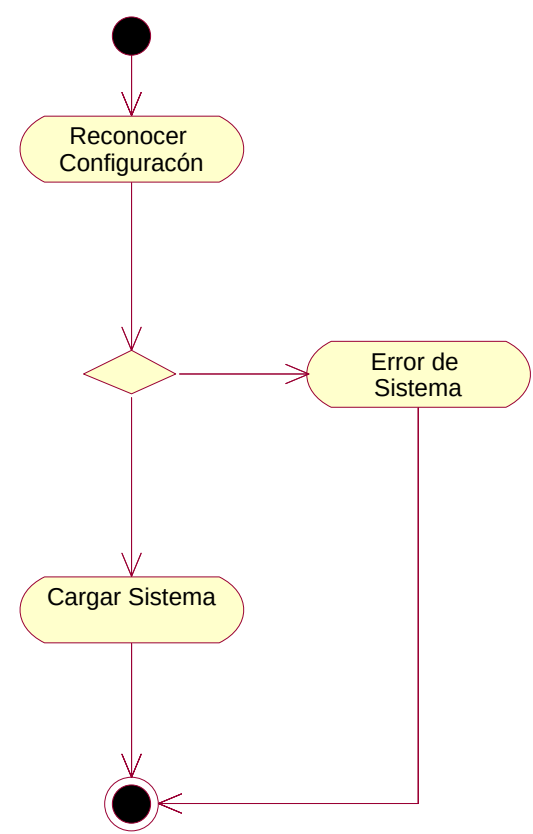


Diagrama de actividades "Cuentas de usuario"

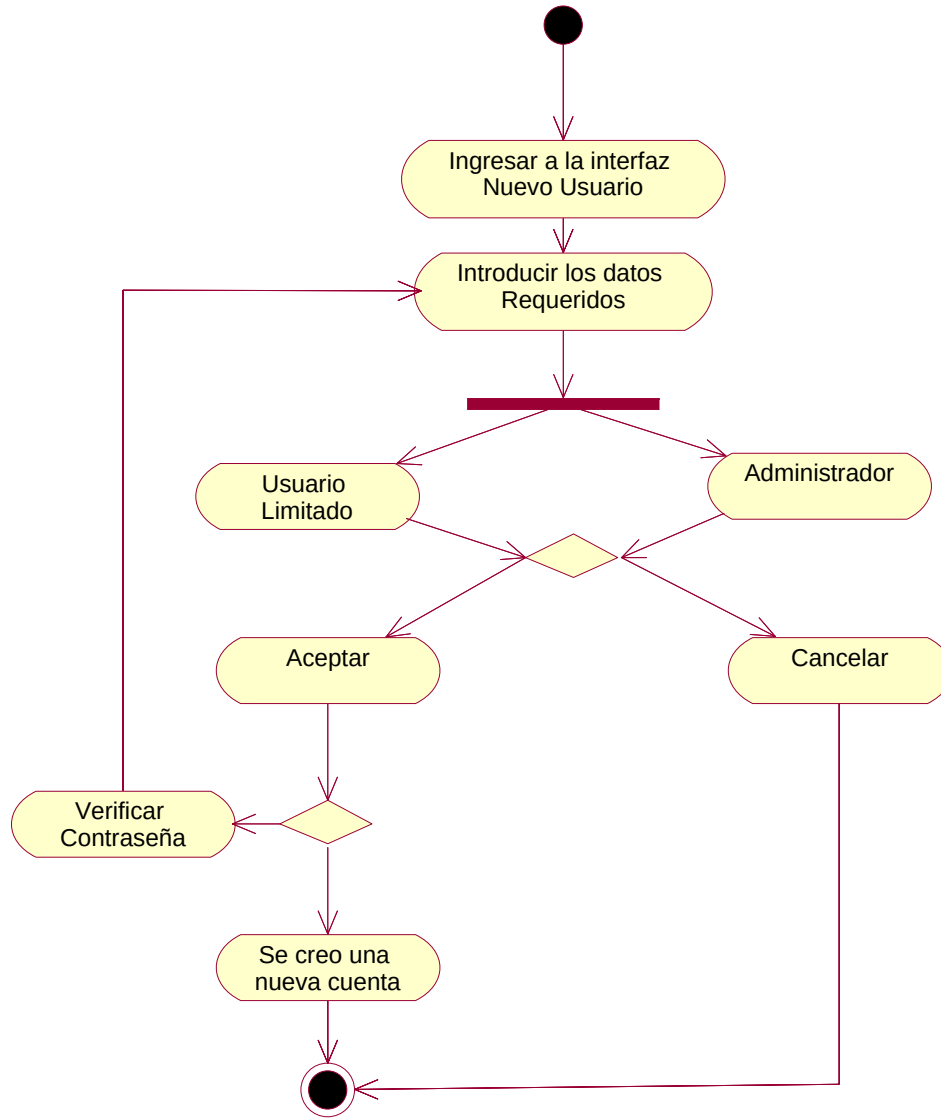


Diagrama de actividades "Emergencia"

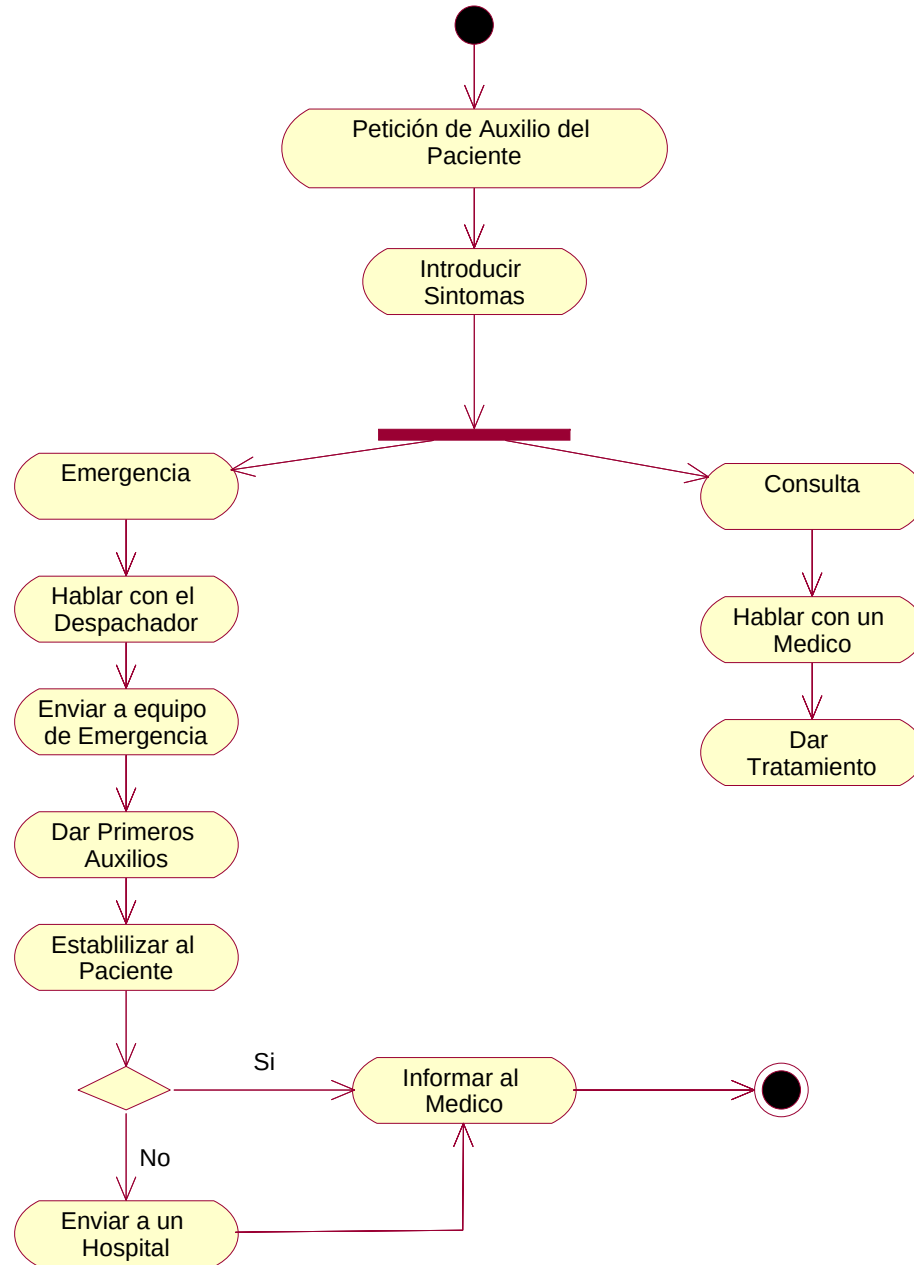


Diagrama de actividades "Monitor"

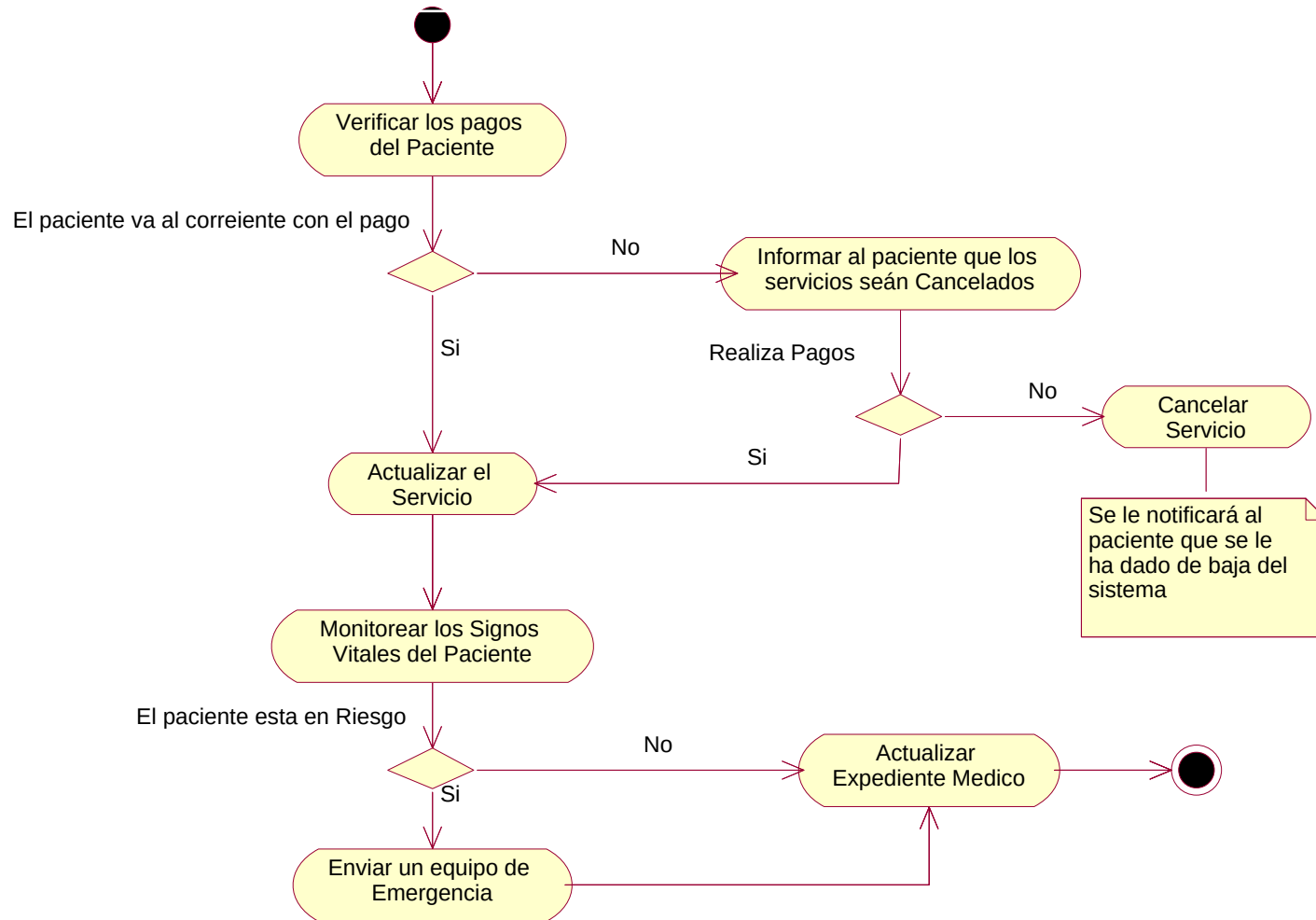


Diagrama de actividades "Suscribir"

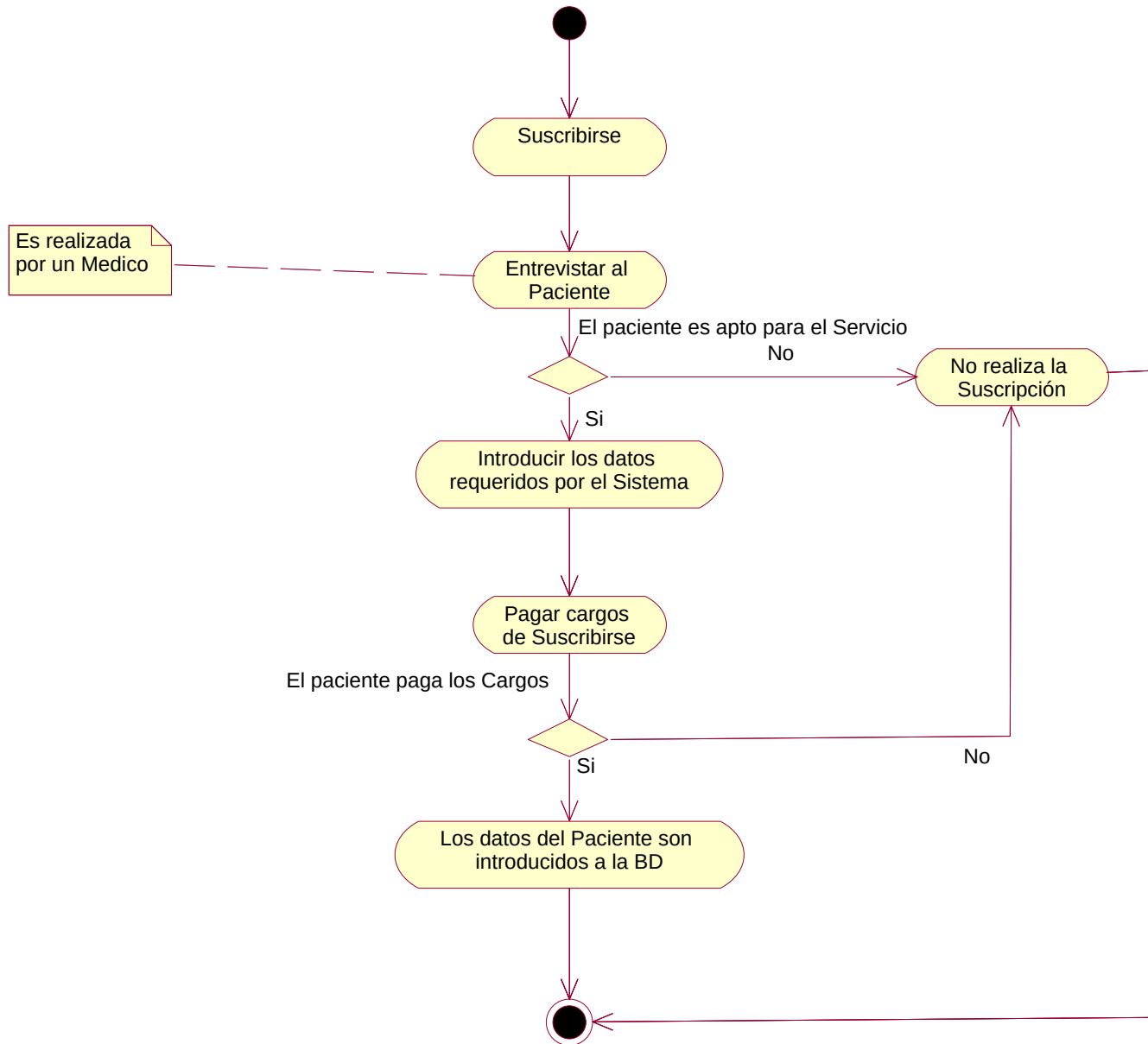
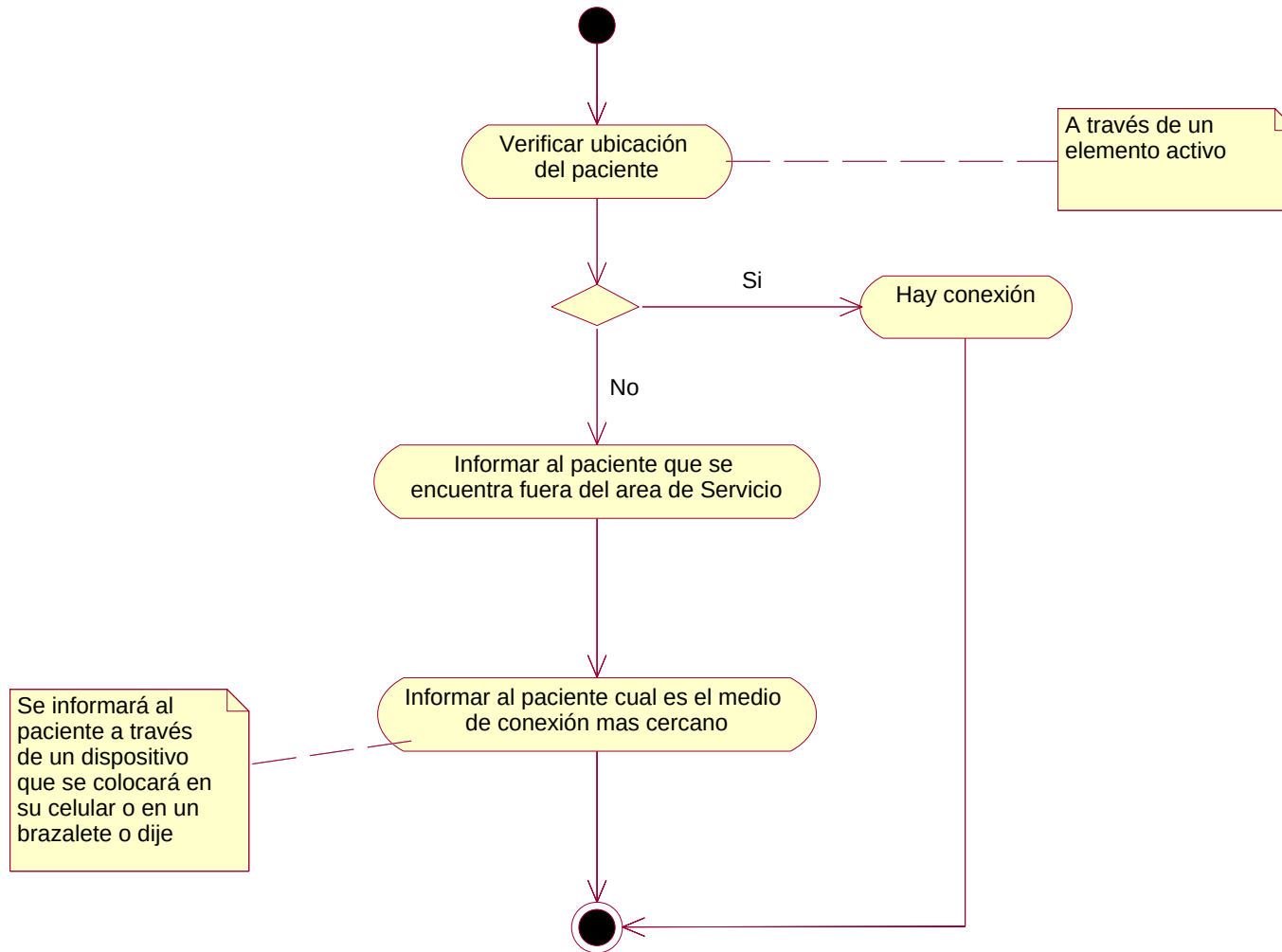


Diagrama de actividades "Disponibilidad de servicio"



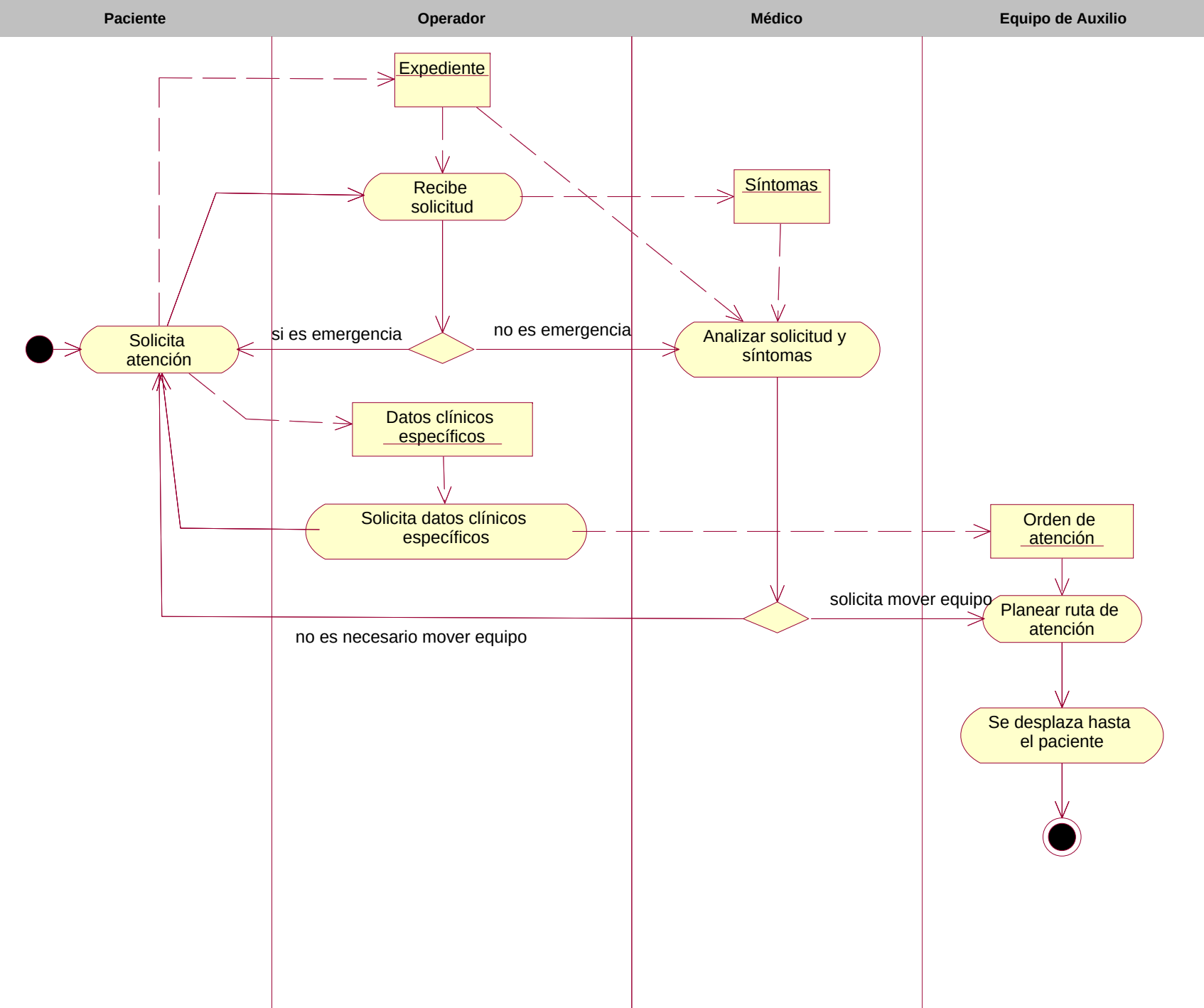


Diagrama "Flujo de trabajo"

Diagramas de Componentes

Diagrama de componentes "Cambiar usuario"

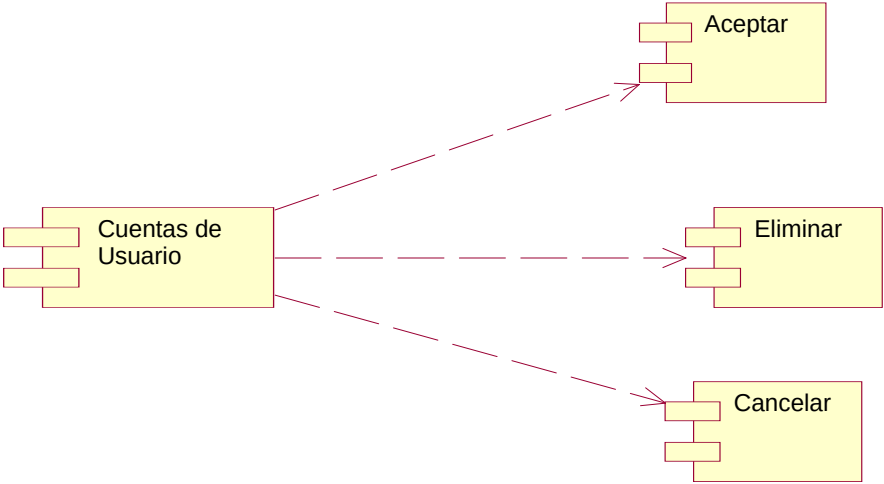


Diagrama de componentes "Consulta"

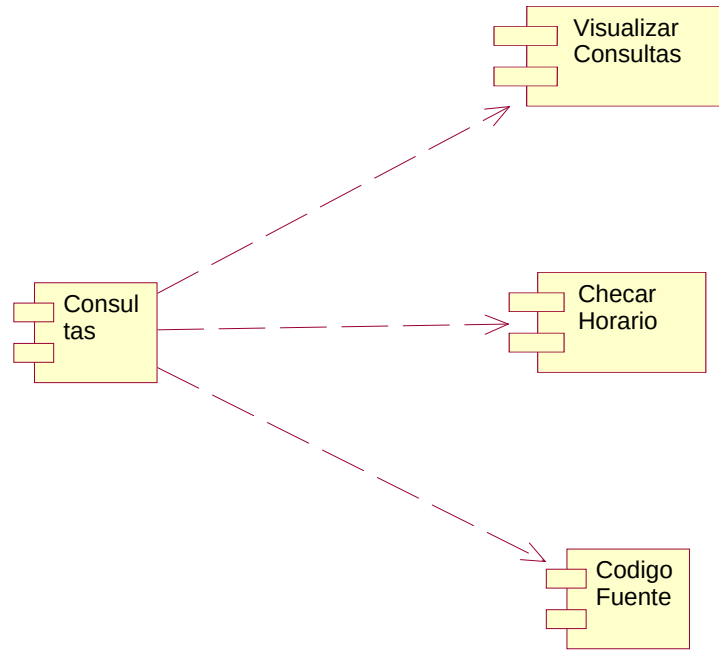


Diagrama de componentes "Contraseña"

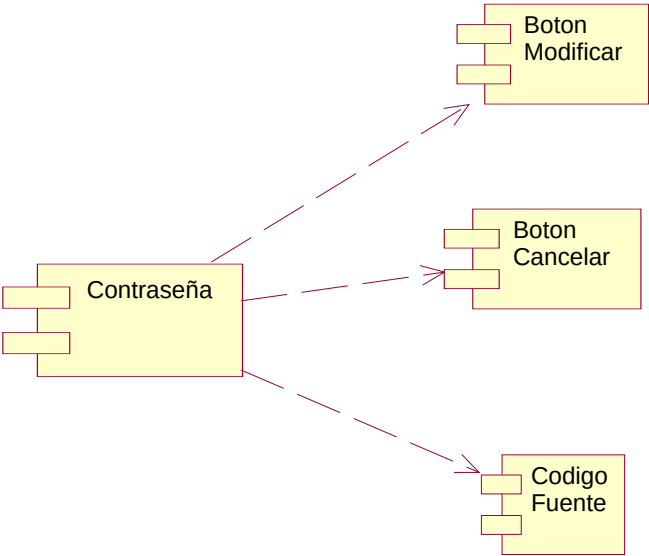


Diagrama de componentes "Interfaz"

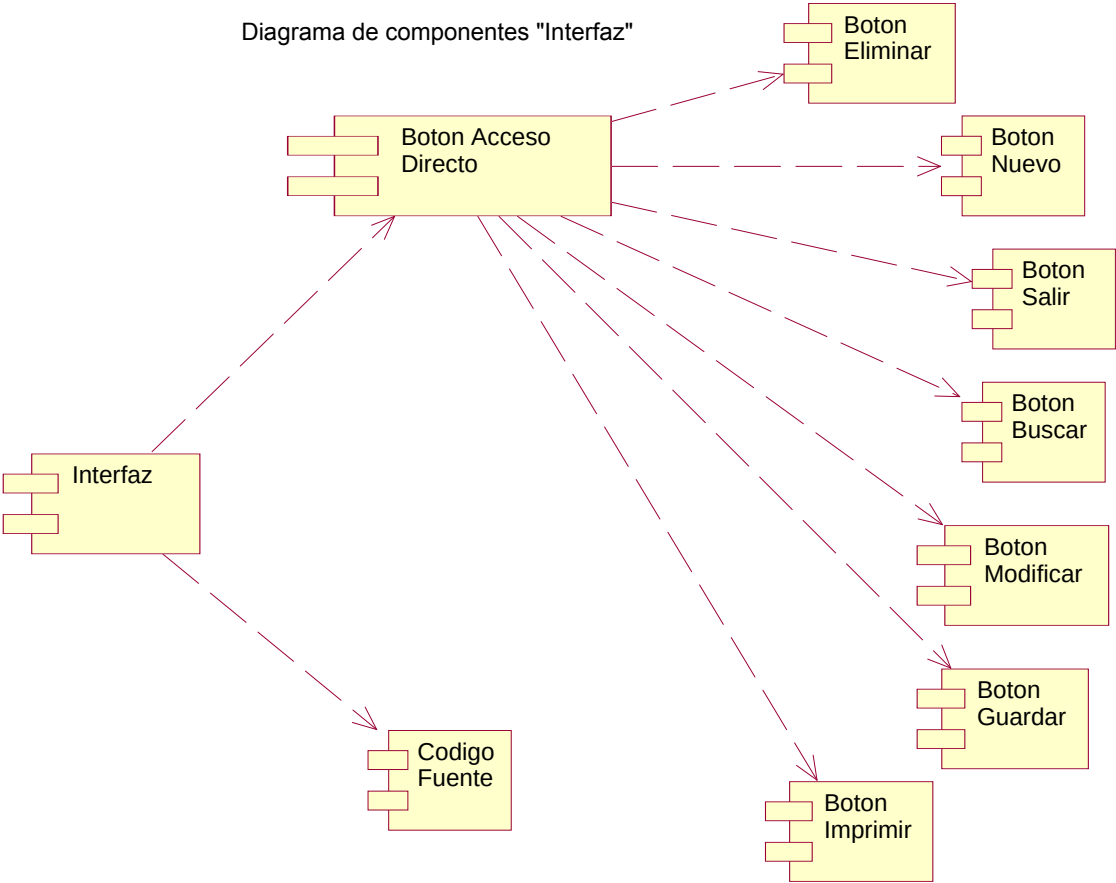


Diagrama de componentes "Paciente"

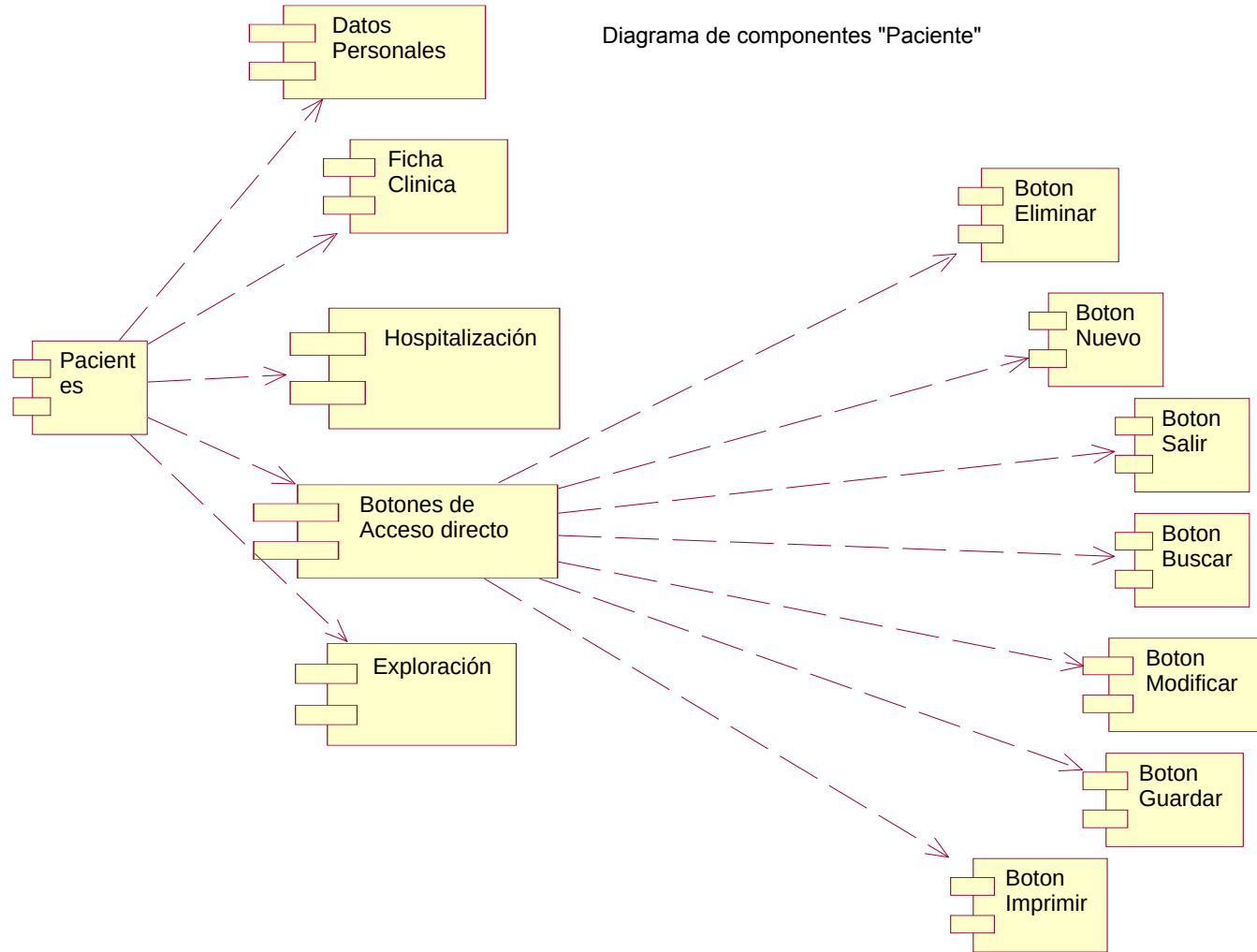


Diagrama de componentes "Principal"

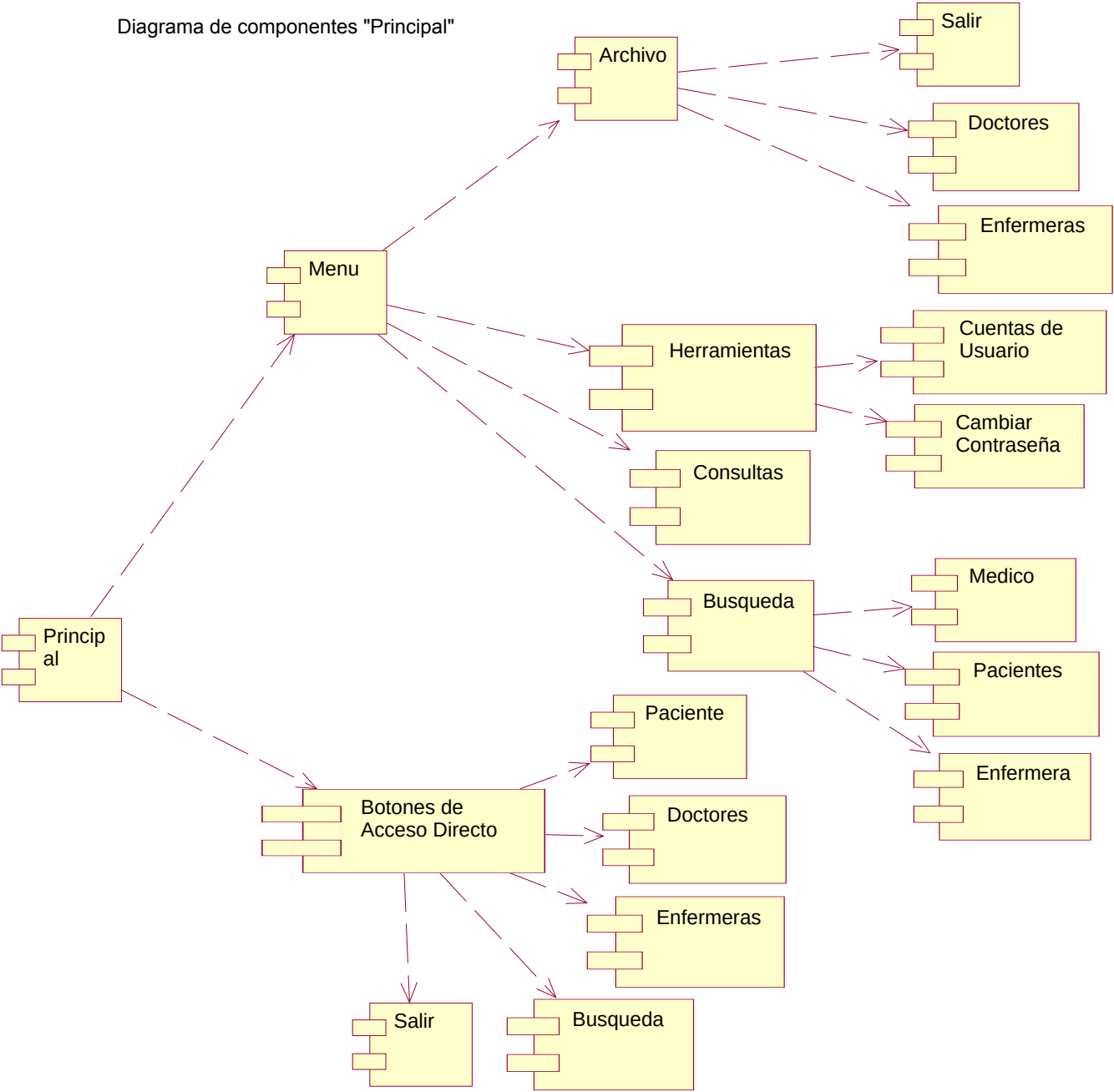
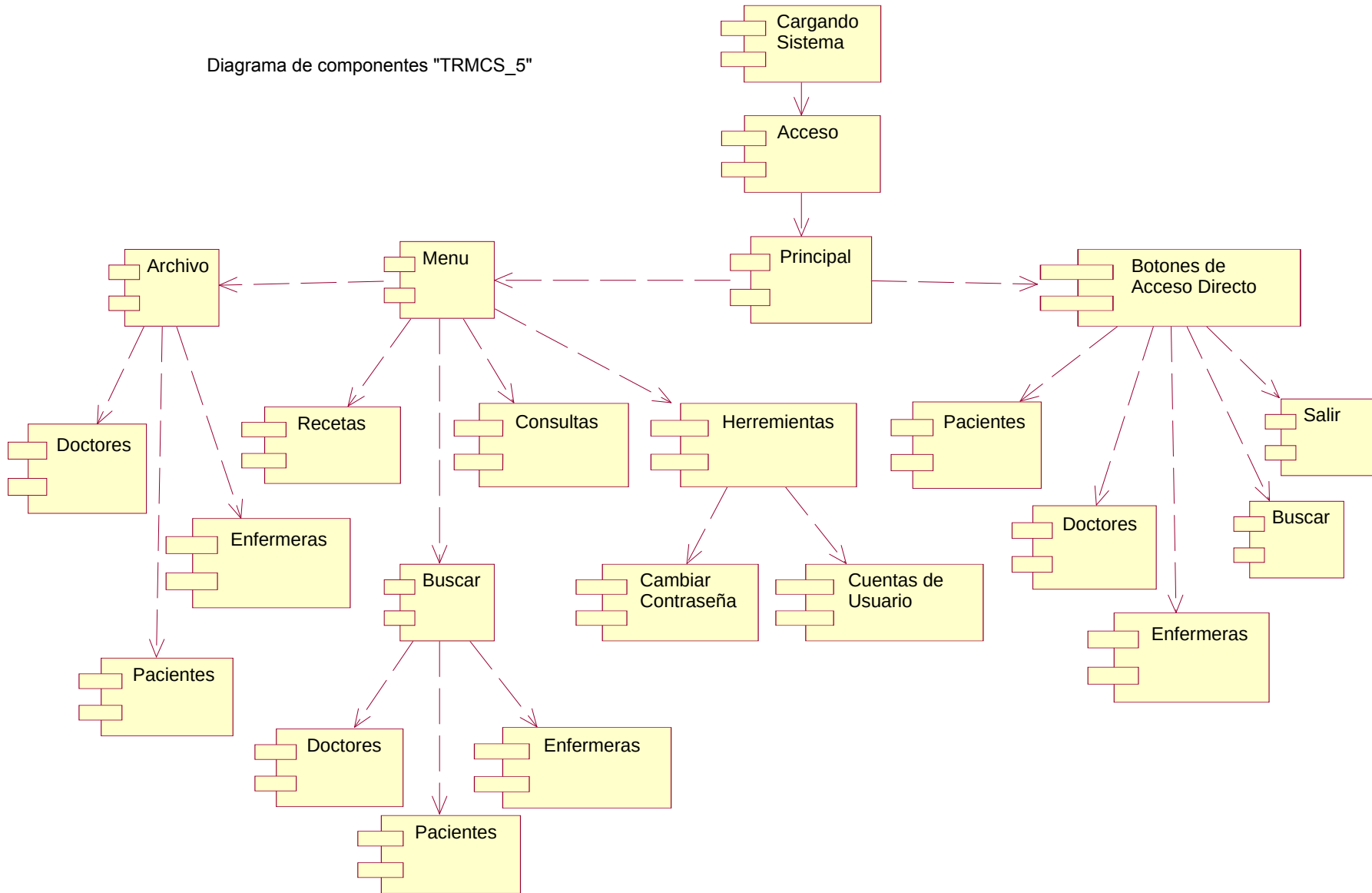
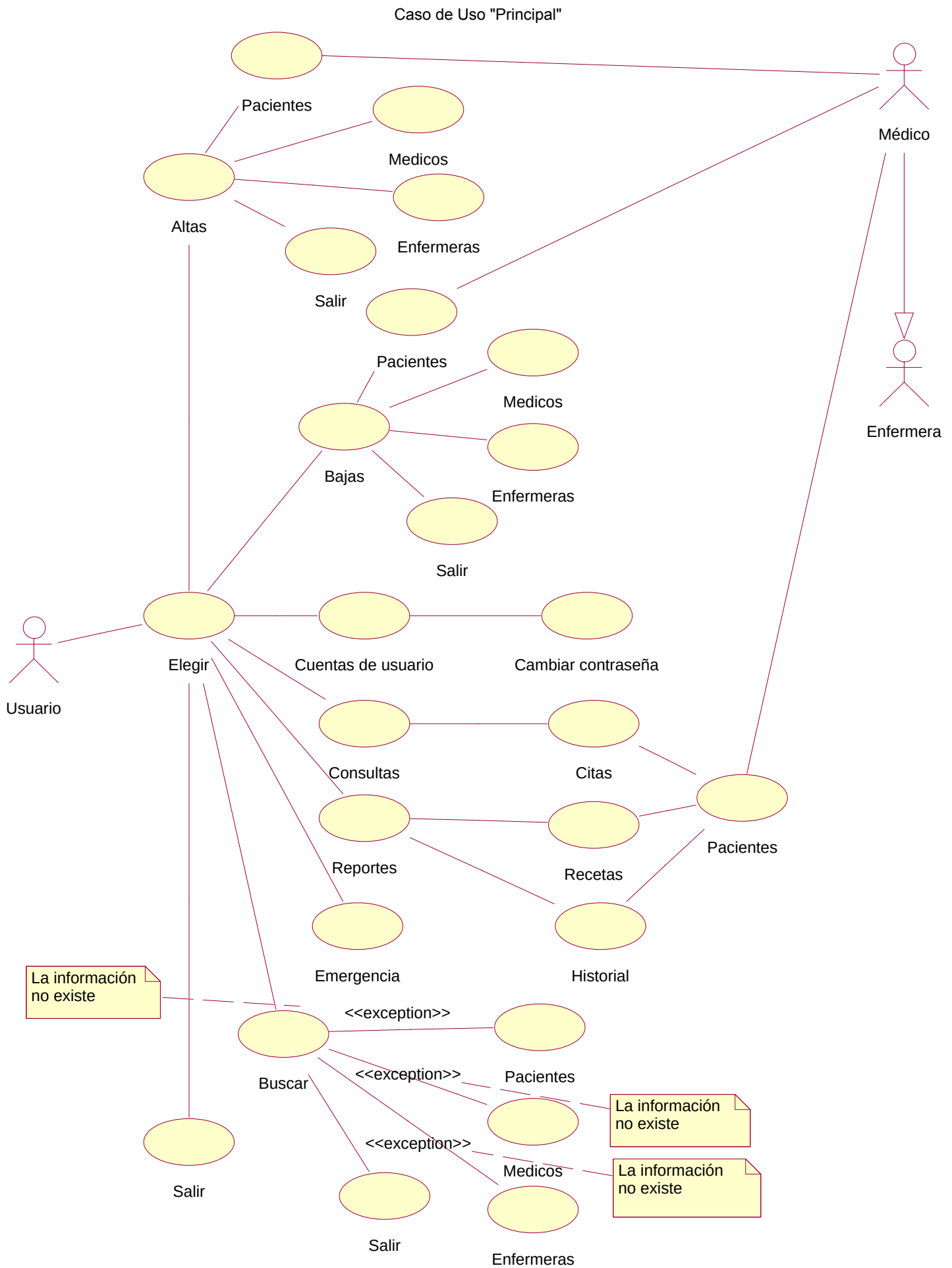


Diagrama de componentes "TRMCS_5"

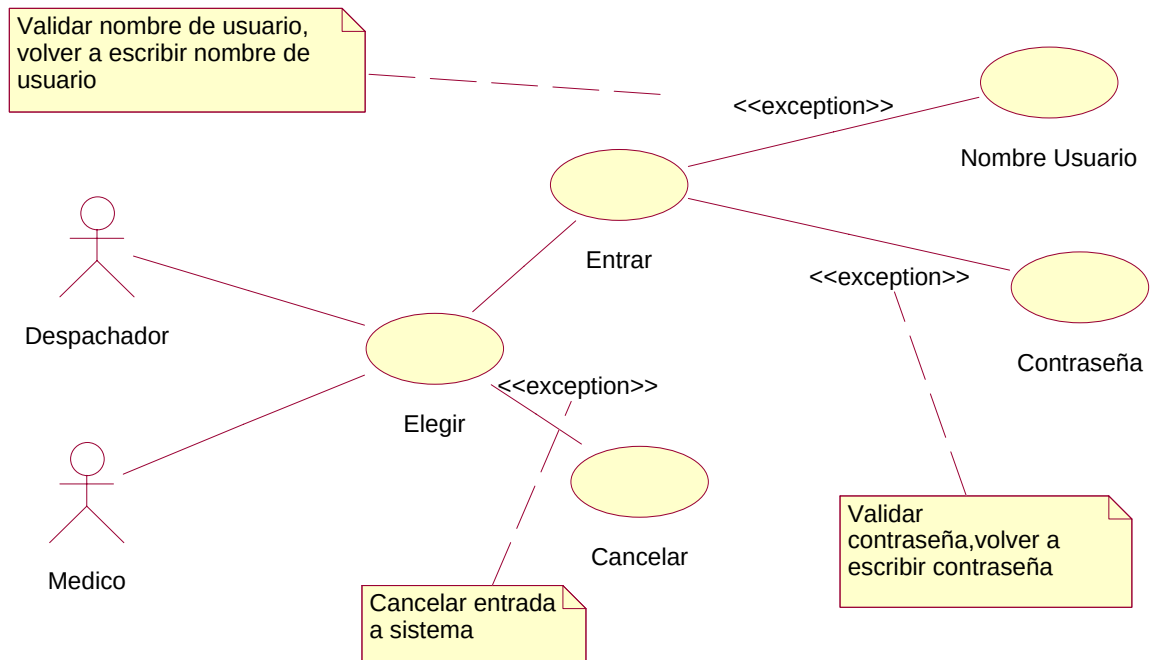


MODELO DE REQUERIMIENTOS

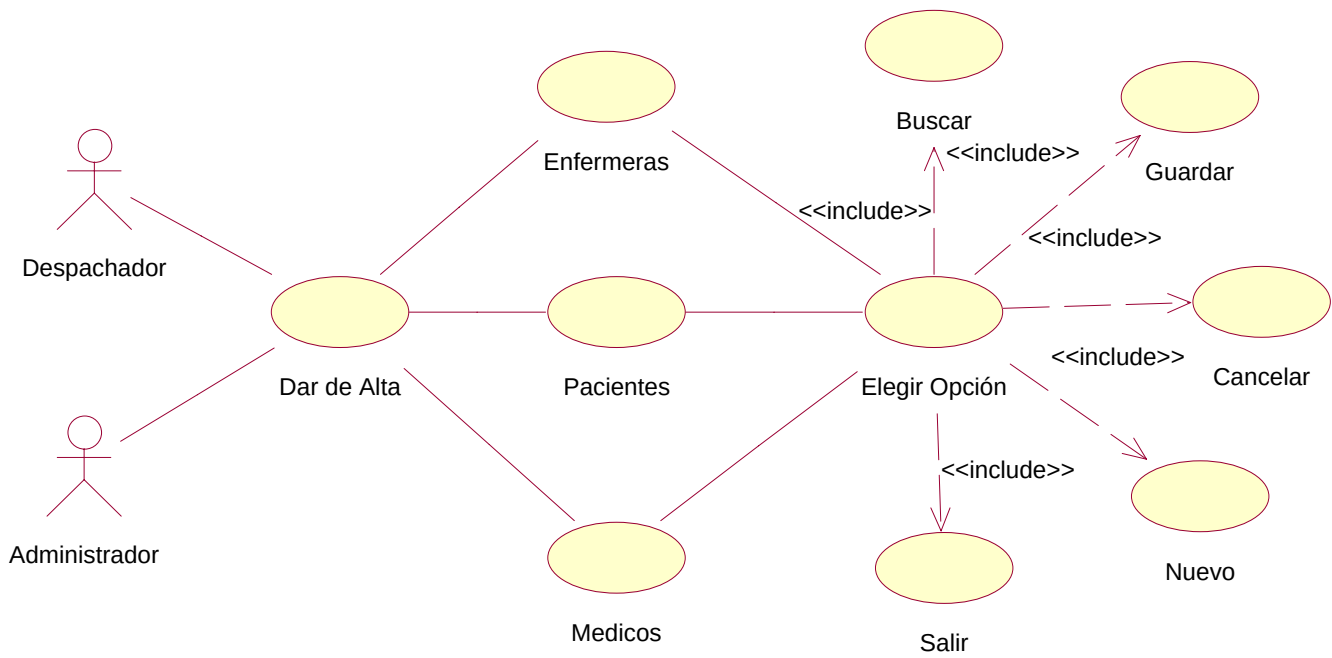
Diagramas de Casos de Uso



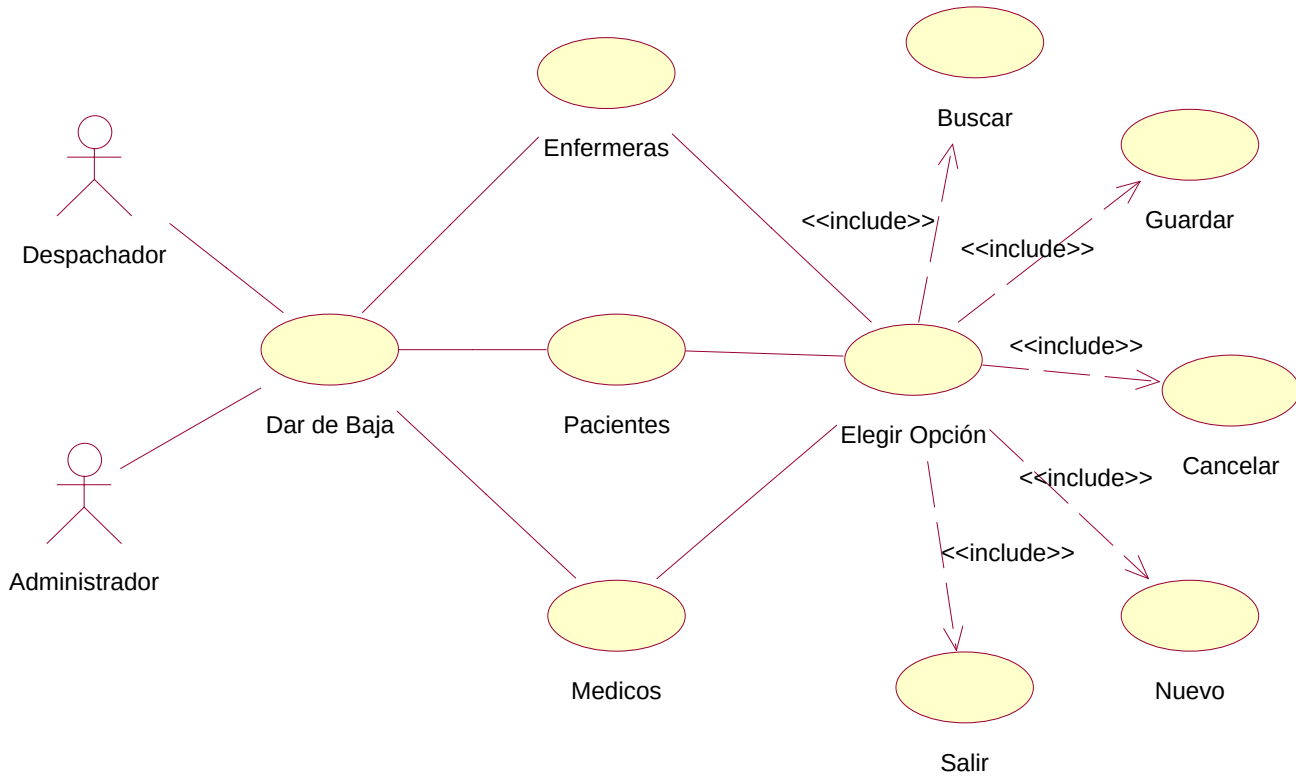
Caso de uso "Acceso"



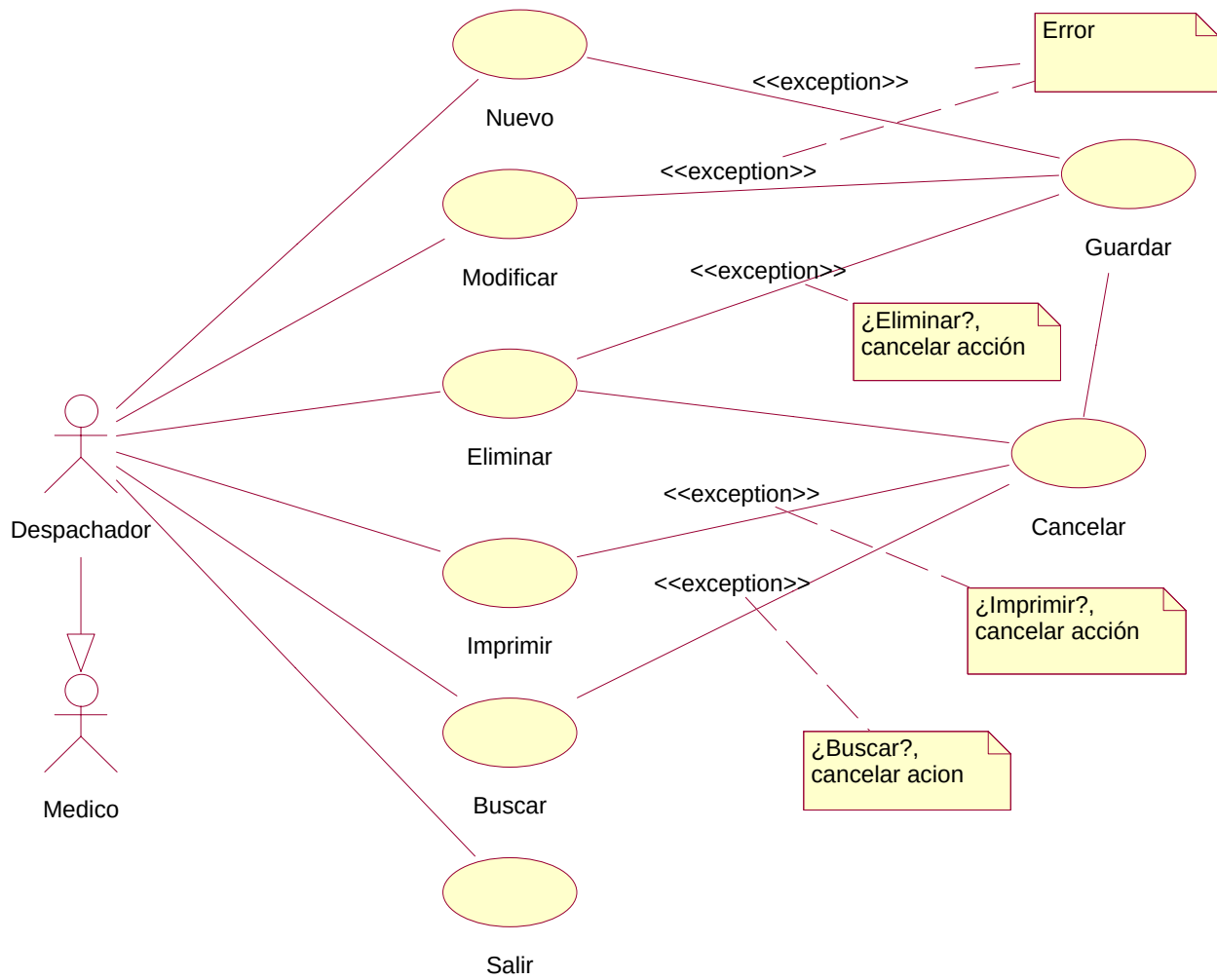
Caso de uso "Altas"

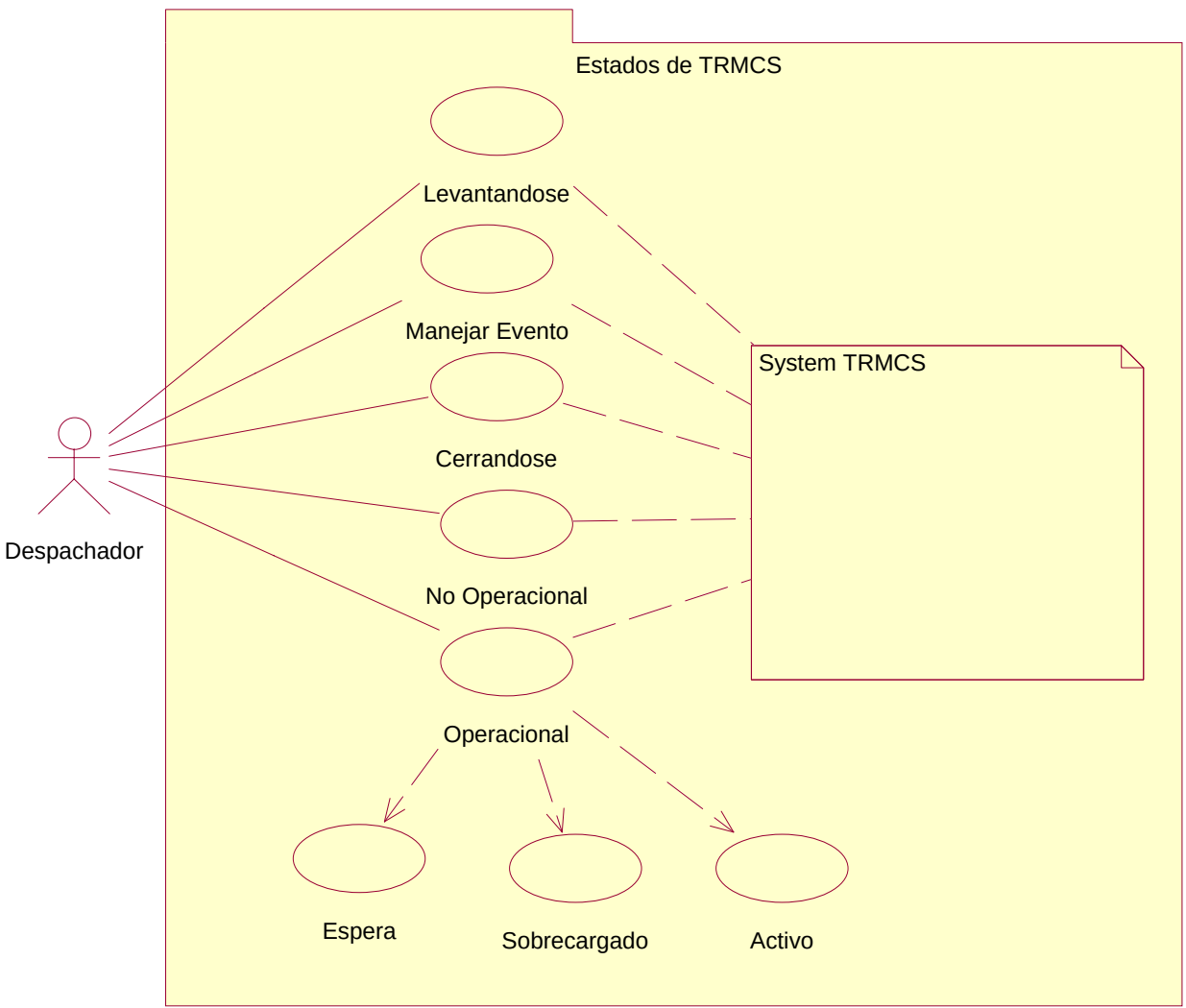


Caso de uso "Bajas"

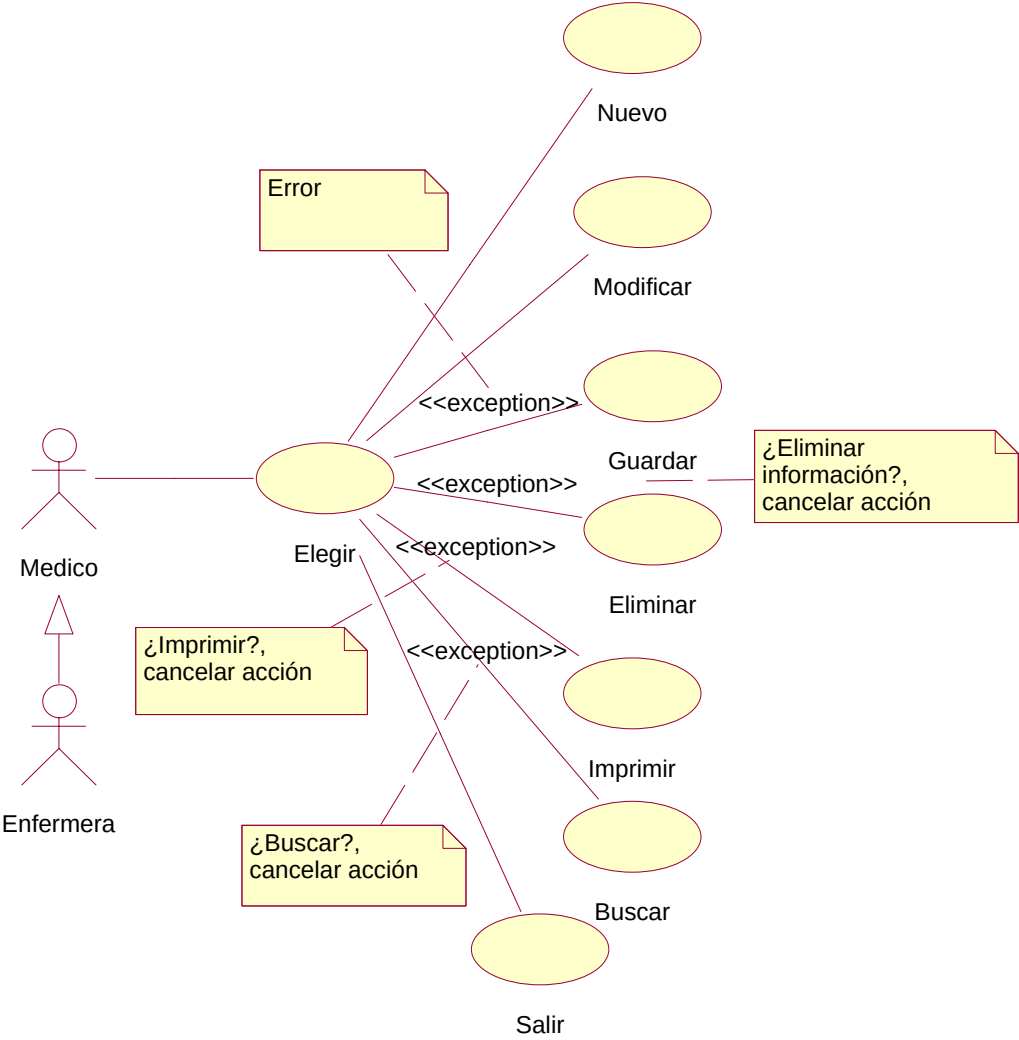


Caso de uso "Búsqueda"

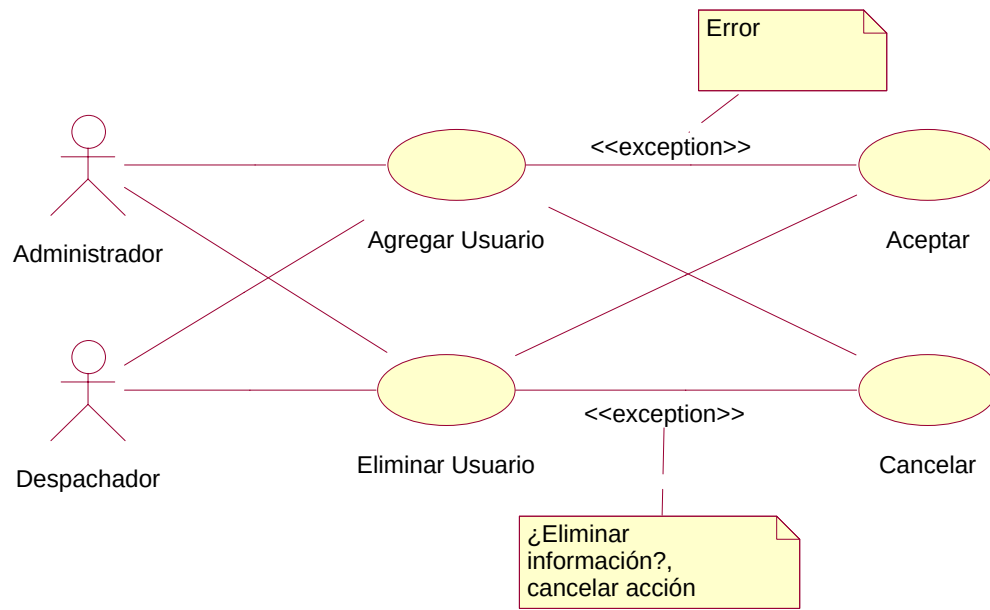




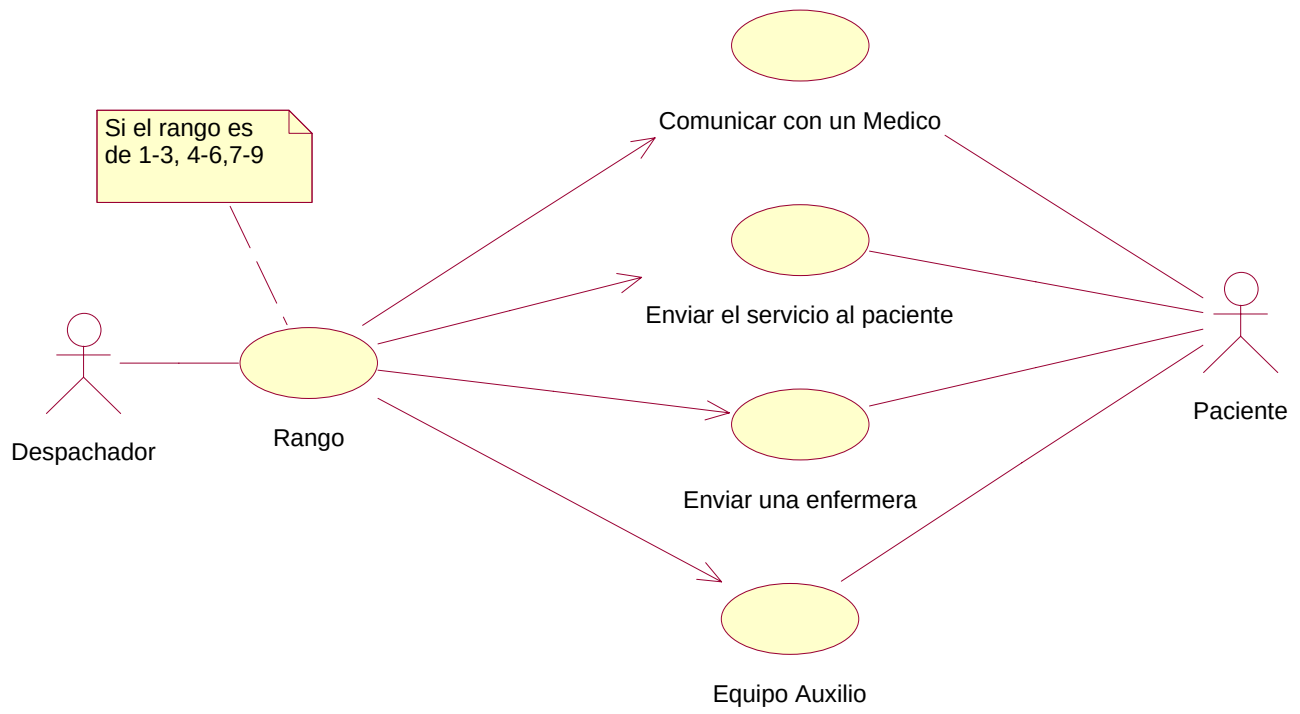
Caso de uso "Consultas"



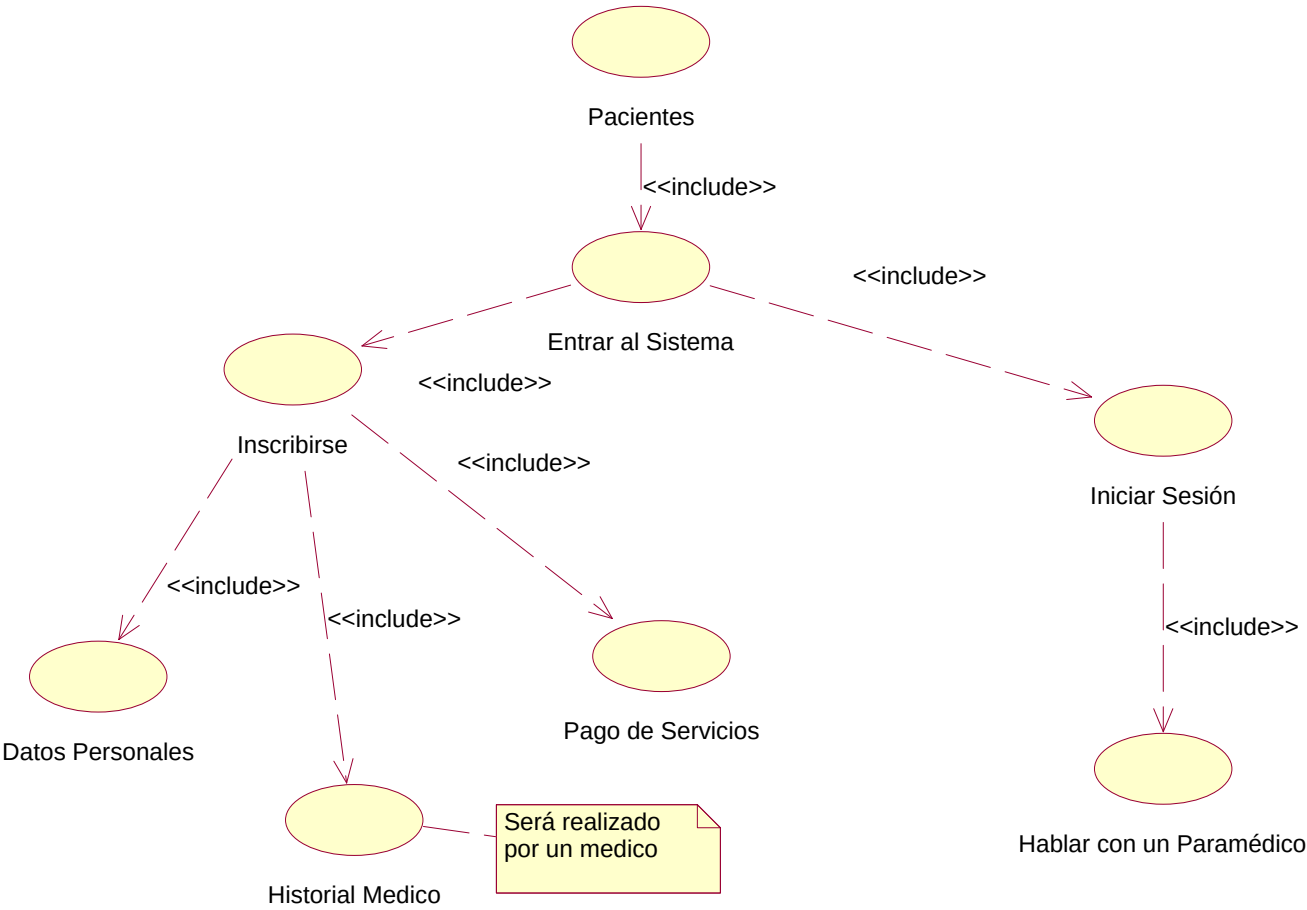
Caso de uso "Cuentas de usuario"



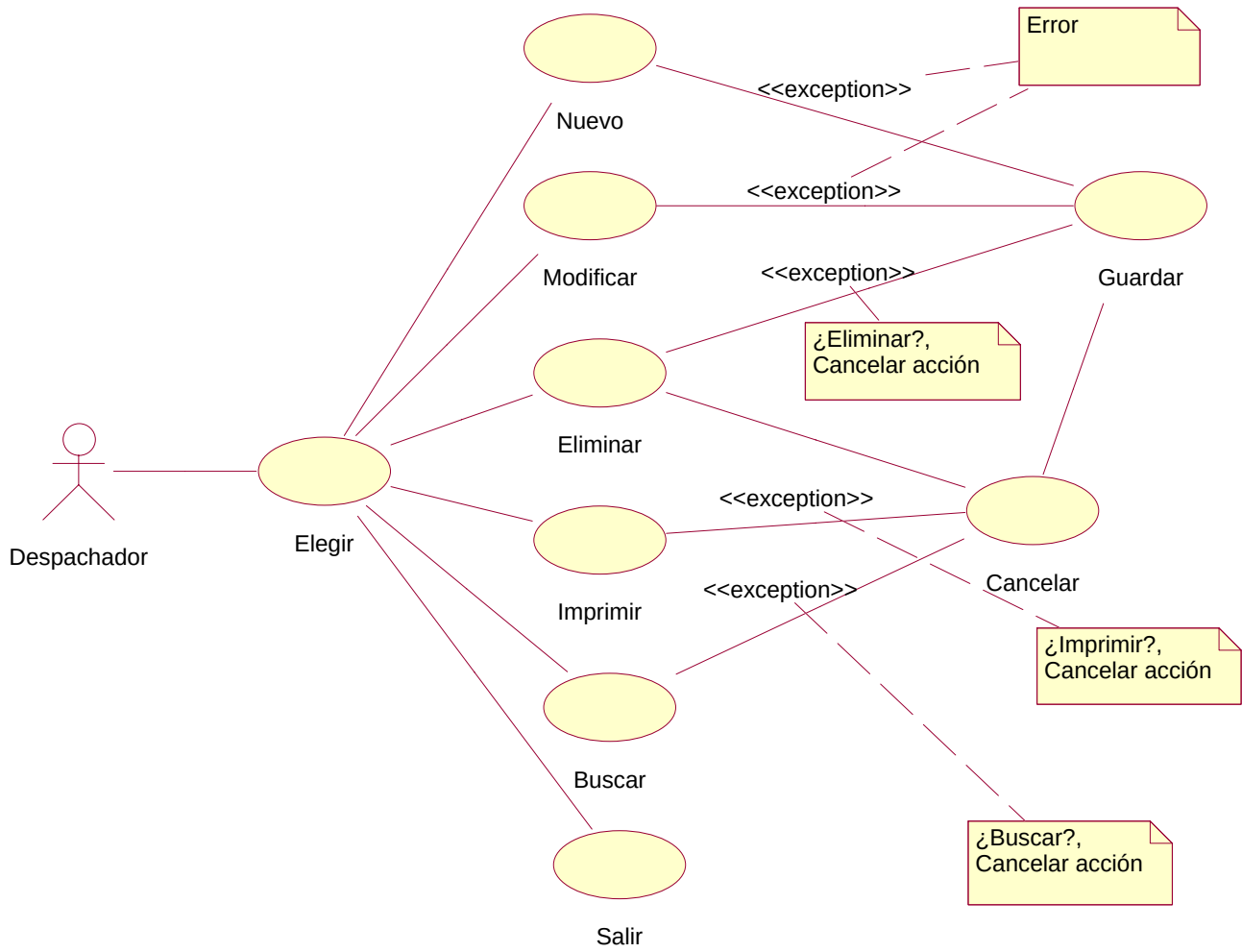
Caso de Uso "Despachador"



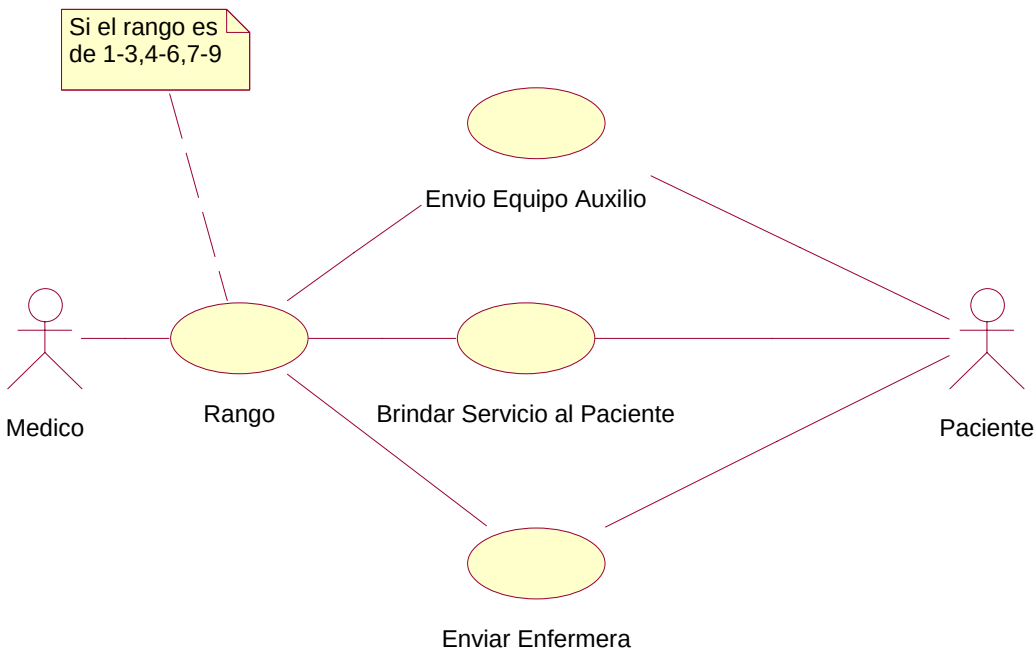
Despliegue Paciente



Caso de uso "Enfermera"



Caso de uso "Médico"



Despliegue Médico



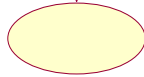
Medico_

<<include>>



Hablar con el paciente

<<include>>



Evaluar la situación

<<include>>



Decidir y Actuar

<<include>>

<<include>>

<<include>>



Enviar equipo de emergencia

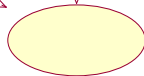
Dar tratamiento al paciente

Enviar a una enfermera

<<include>>

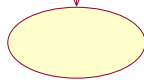
<<include>>

<<include>>



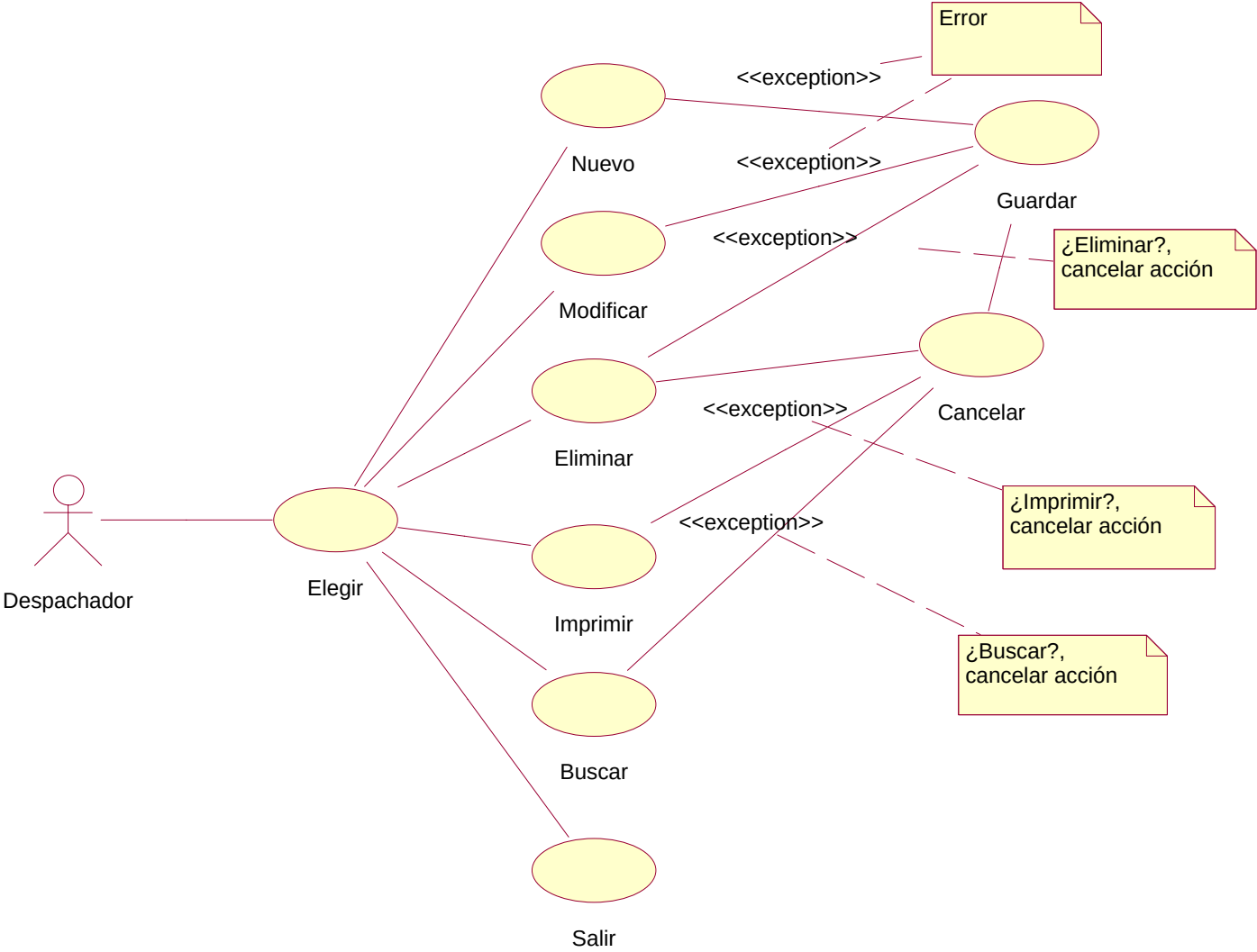
Evaluar el costo

<<include>>

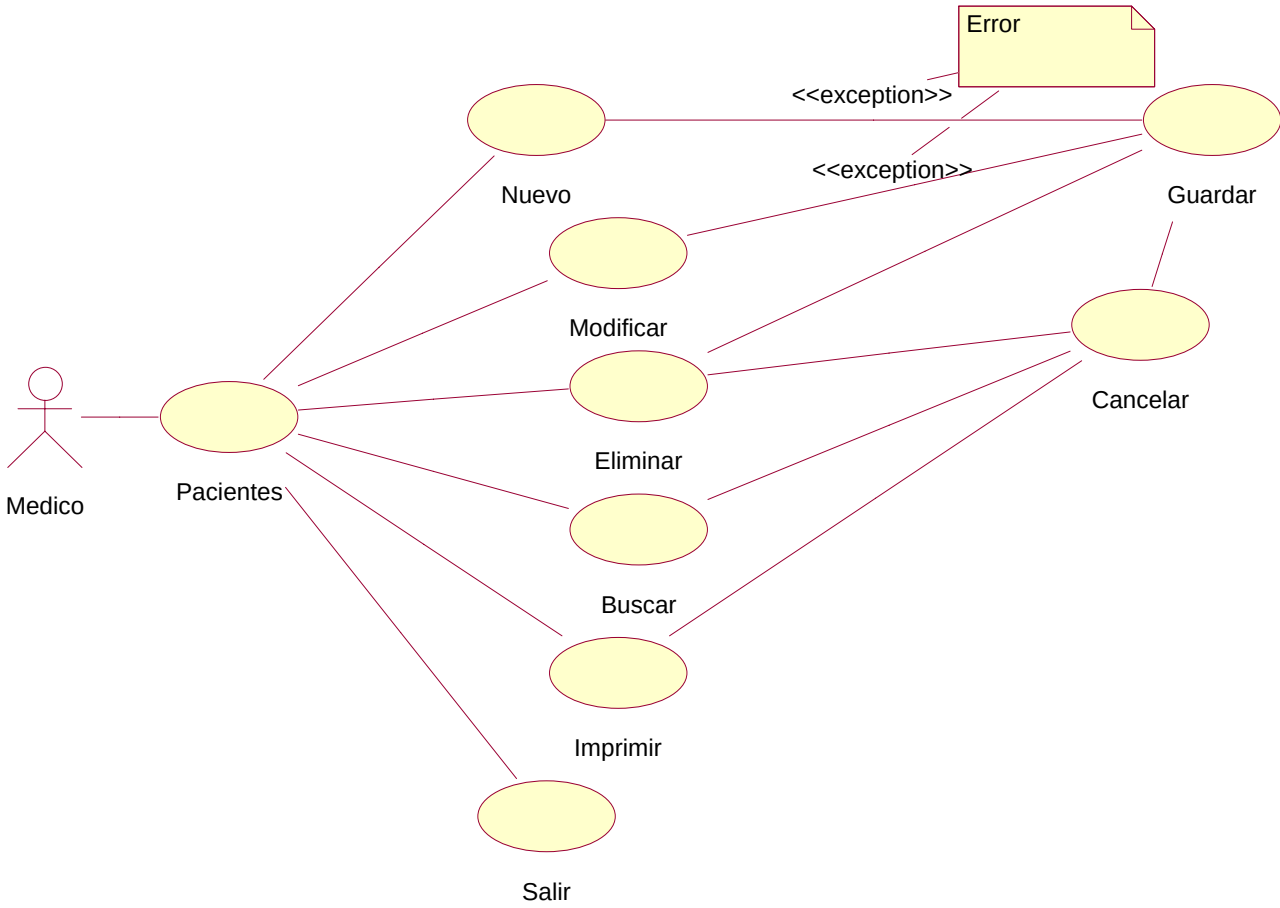


Cobrar al paciente

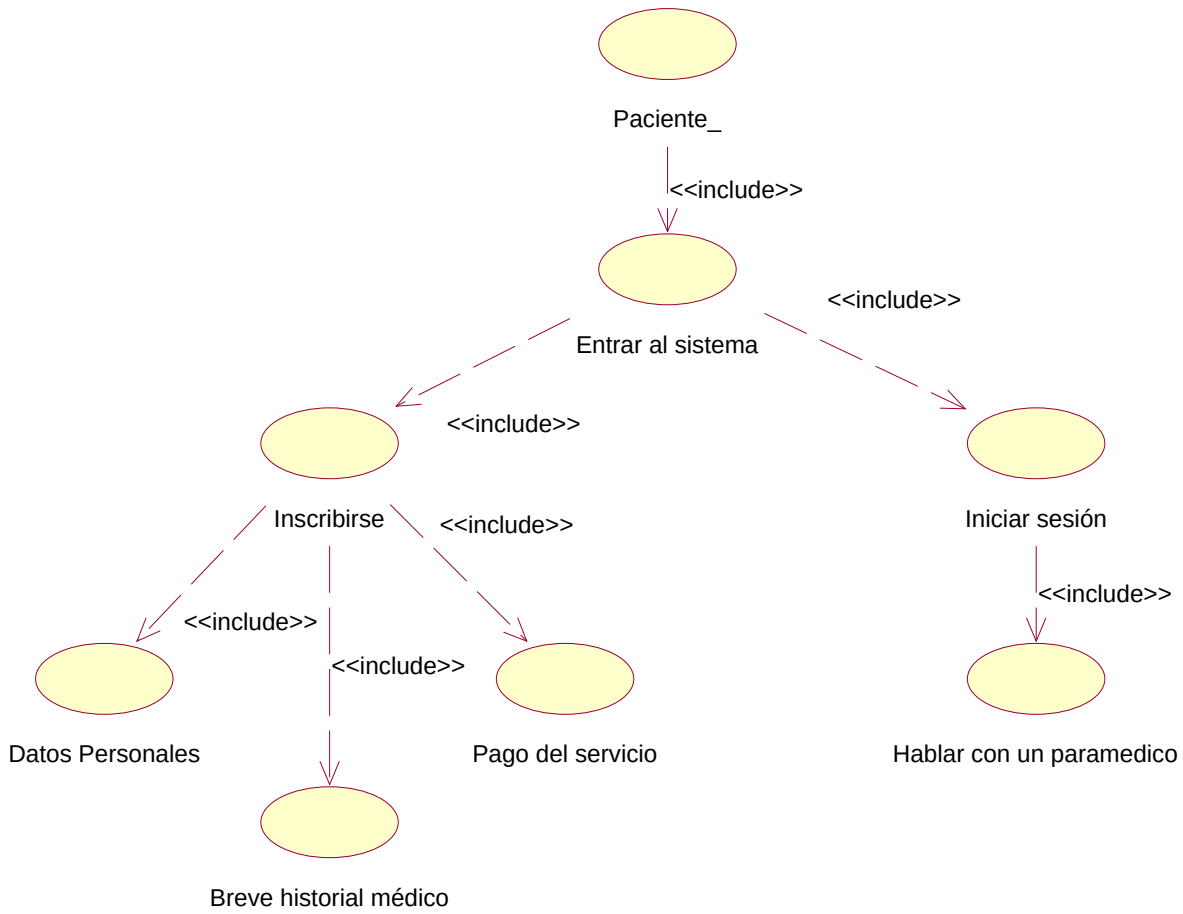
Caso de uso "Médico nuevo"



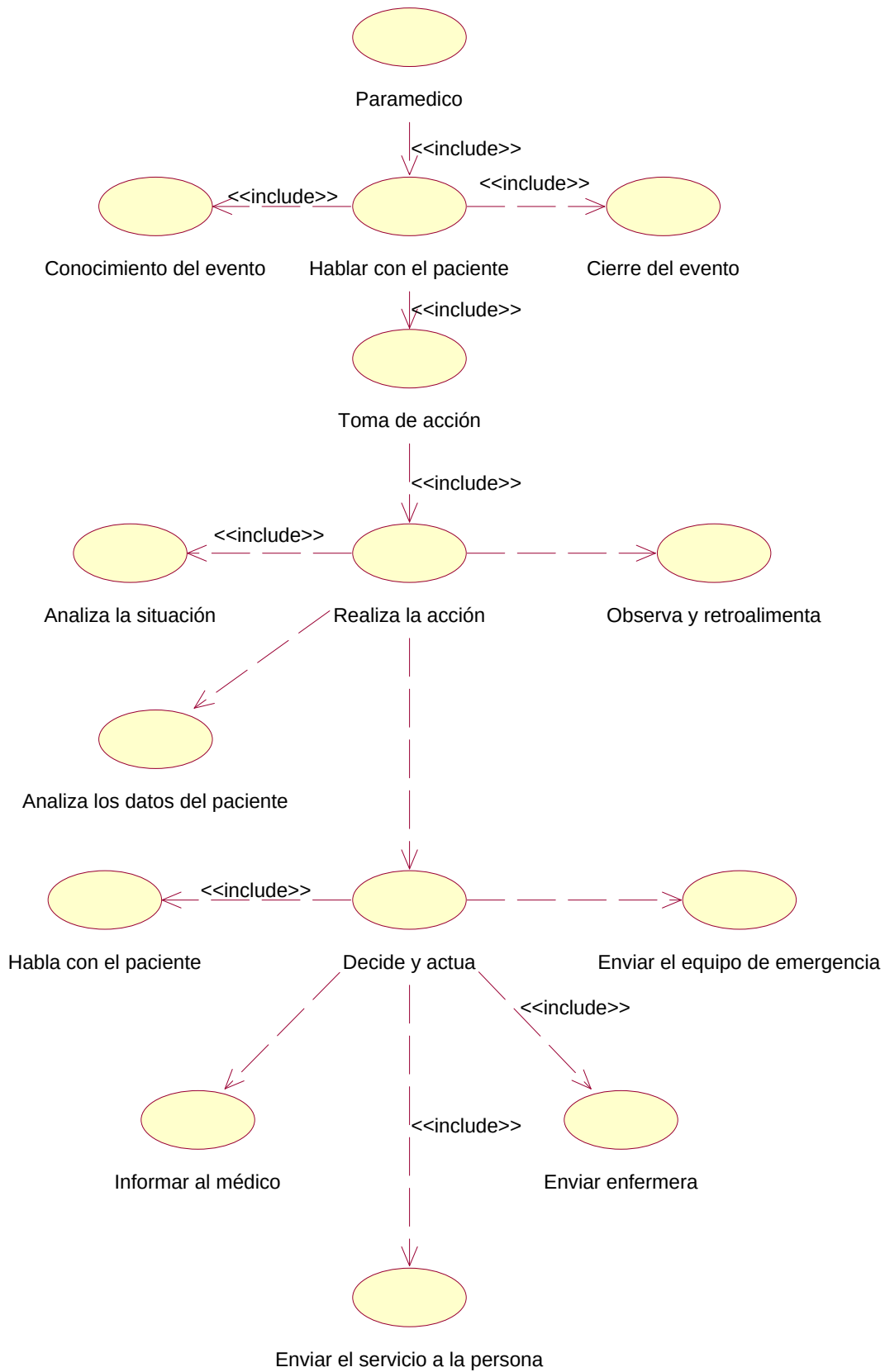
Caso de uso "Paciente"



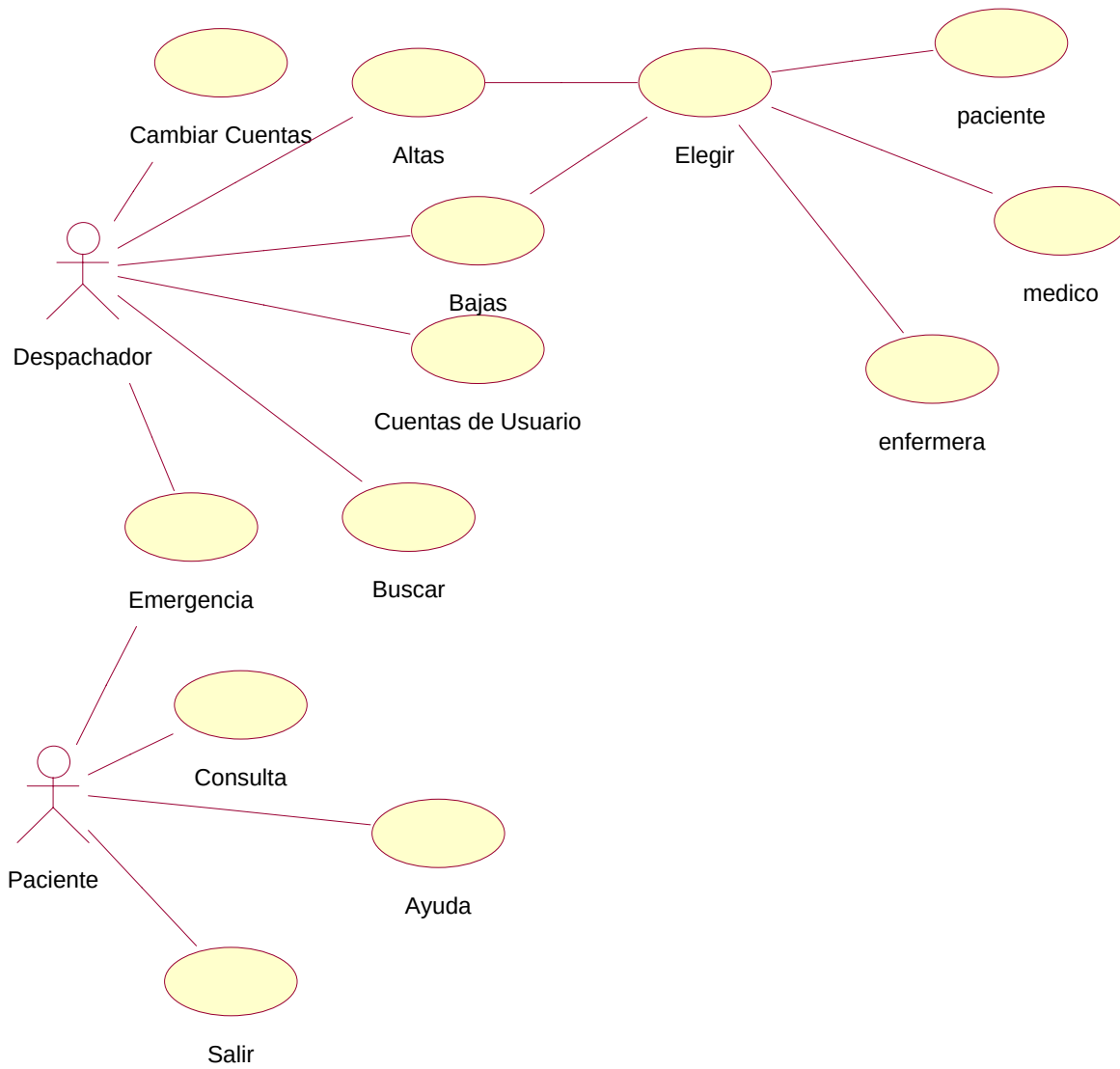
Despliegue Paciente

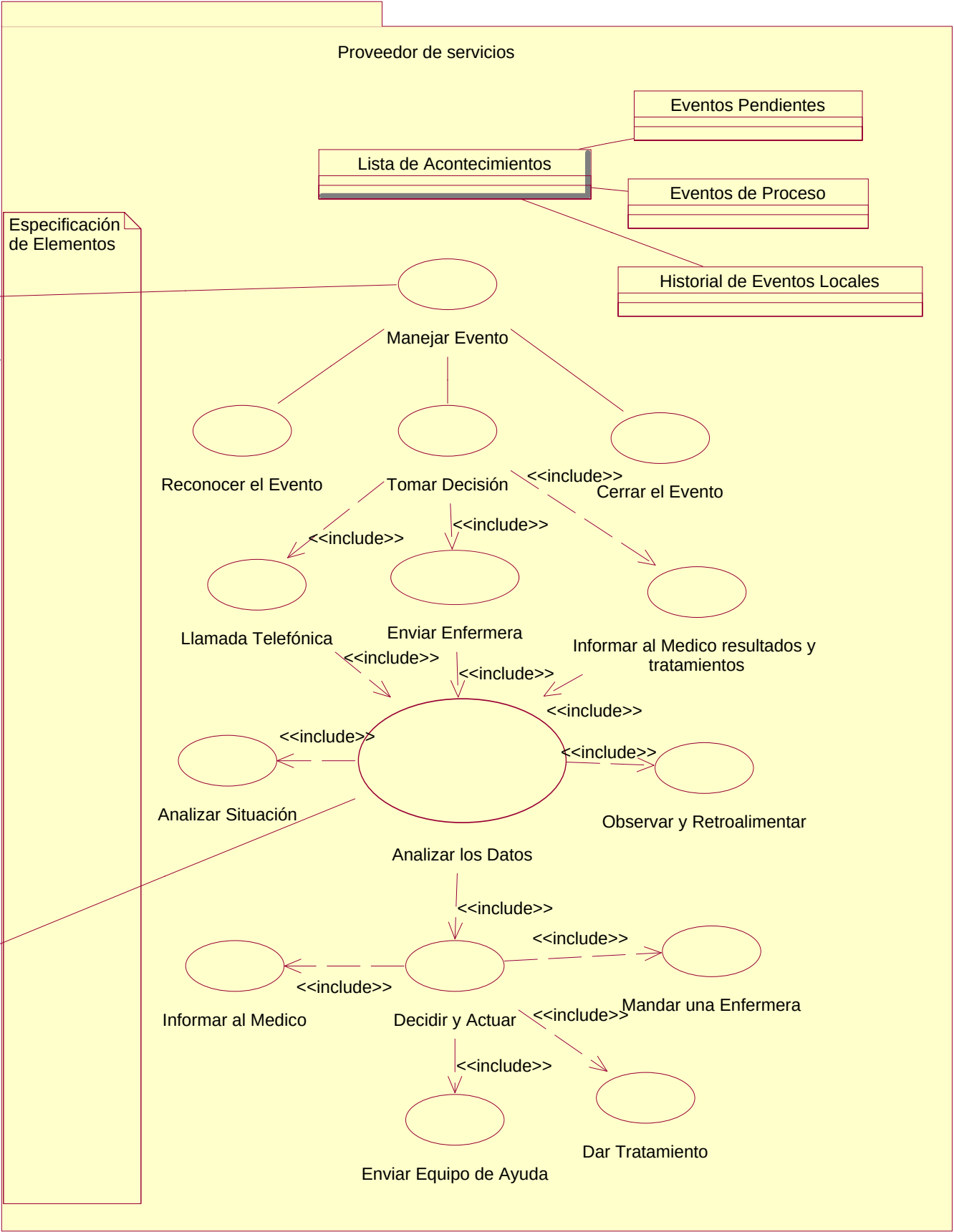


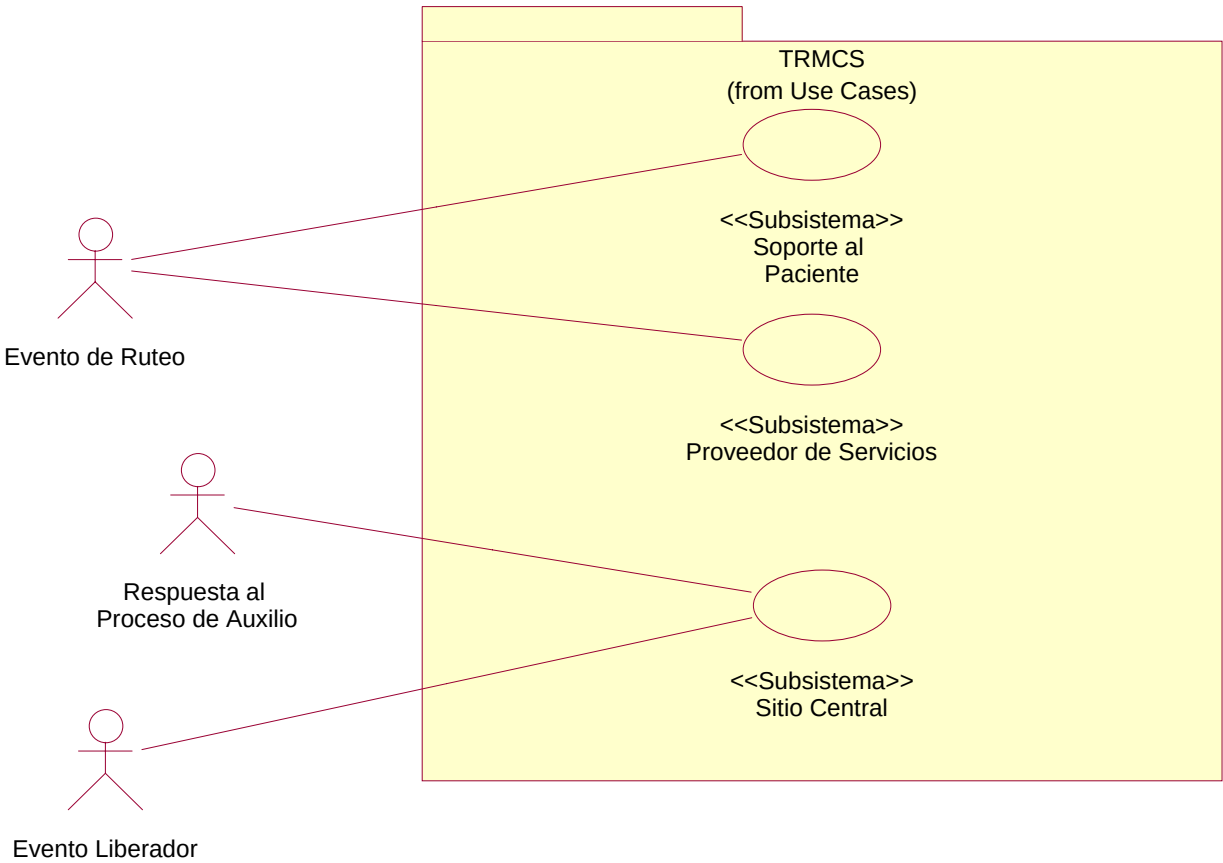
Despliegue "Paramédico"



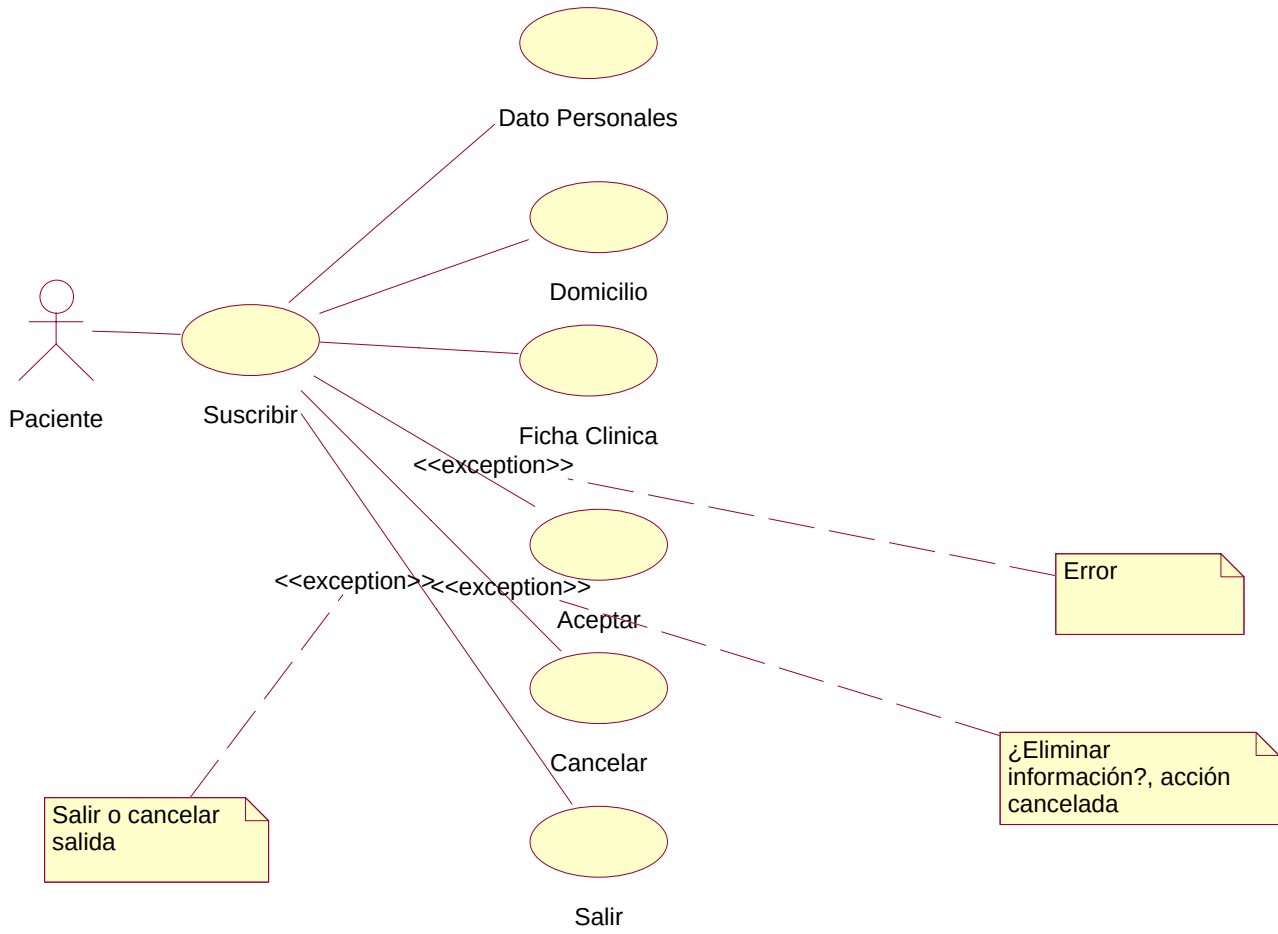
Caso de uso "Principal_2"

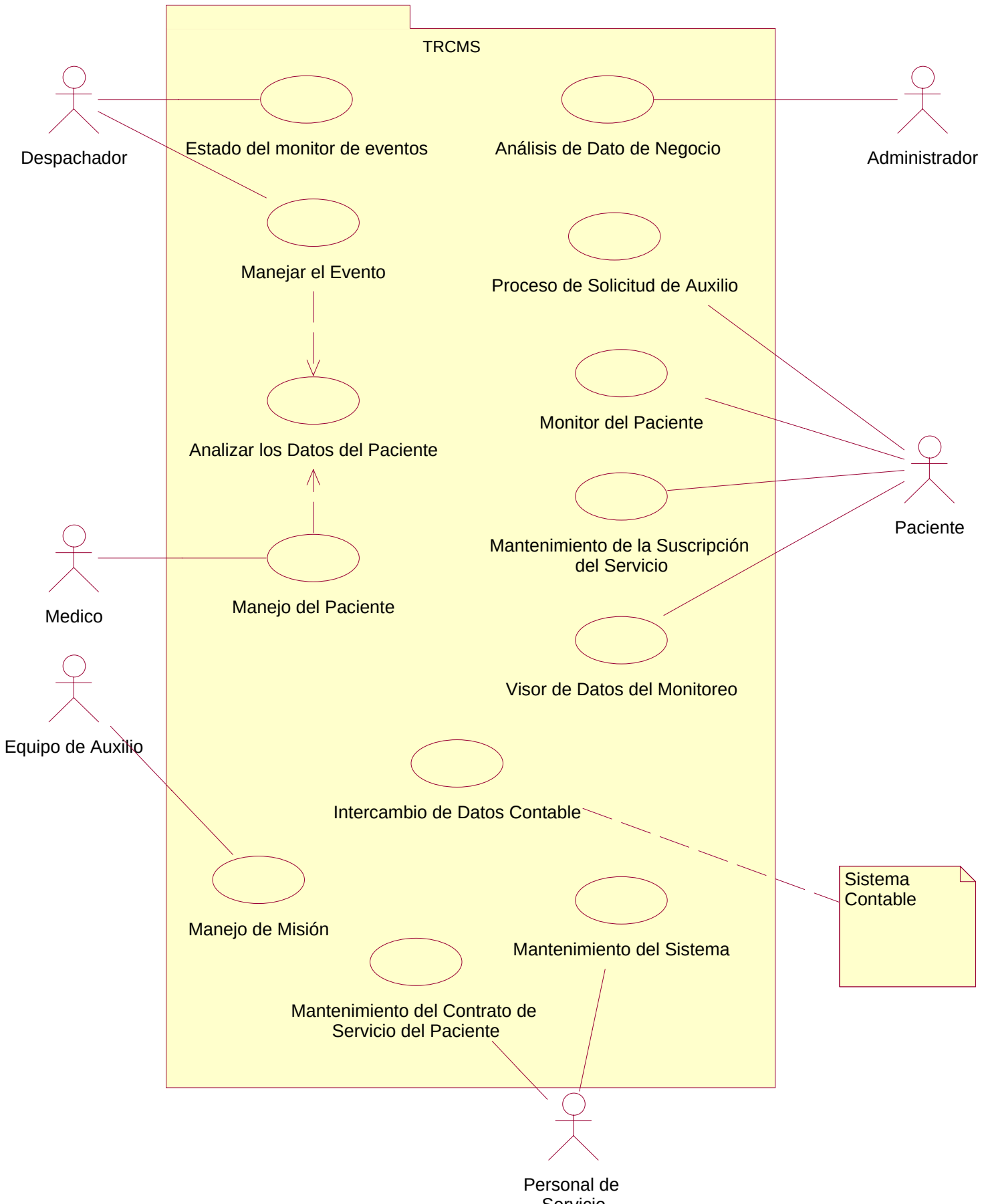


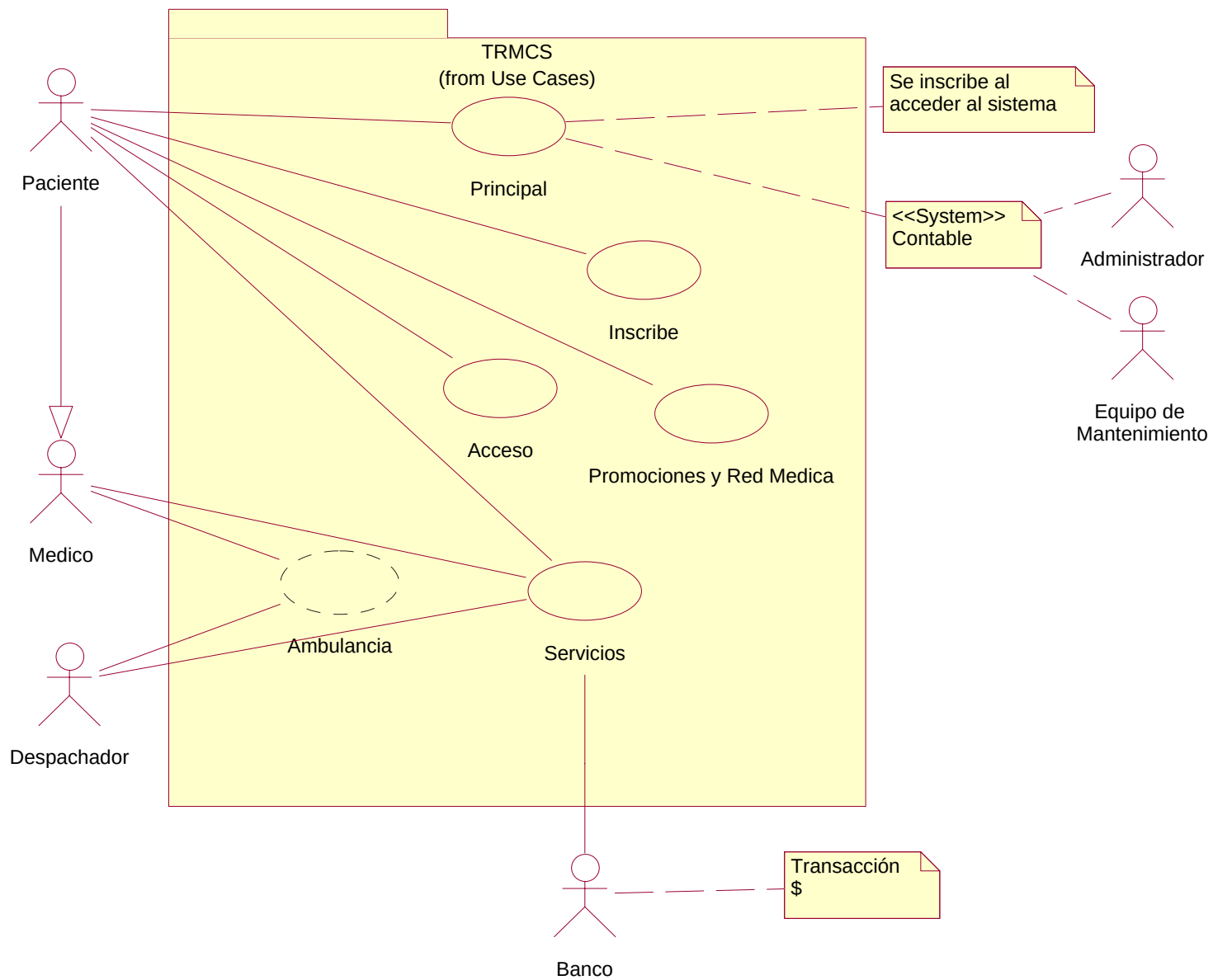


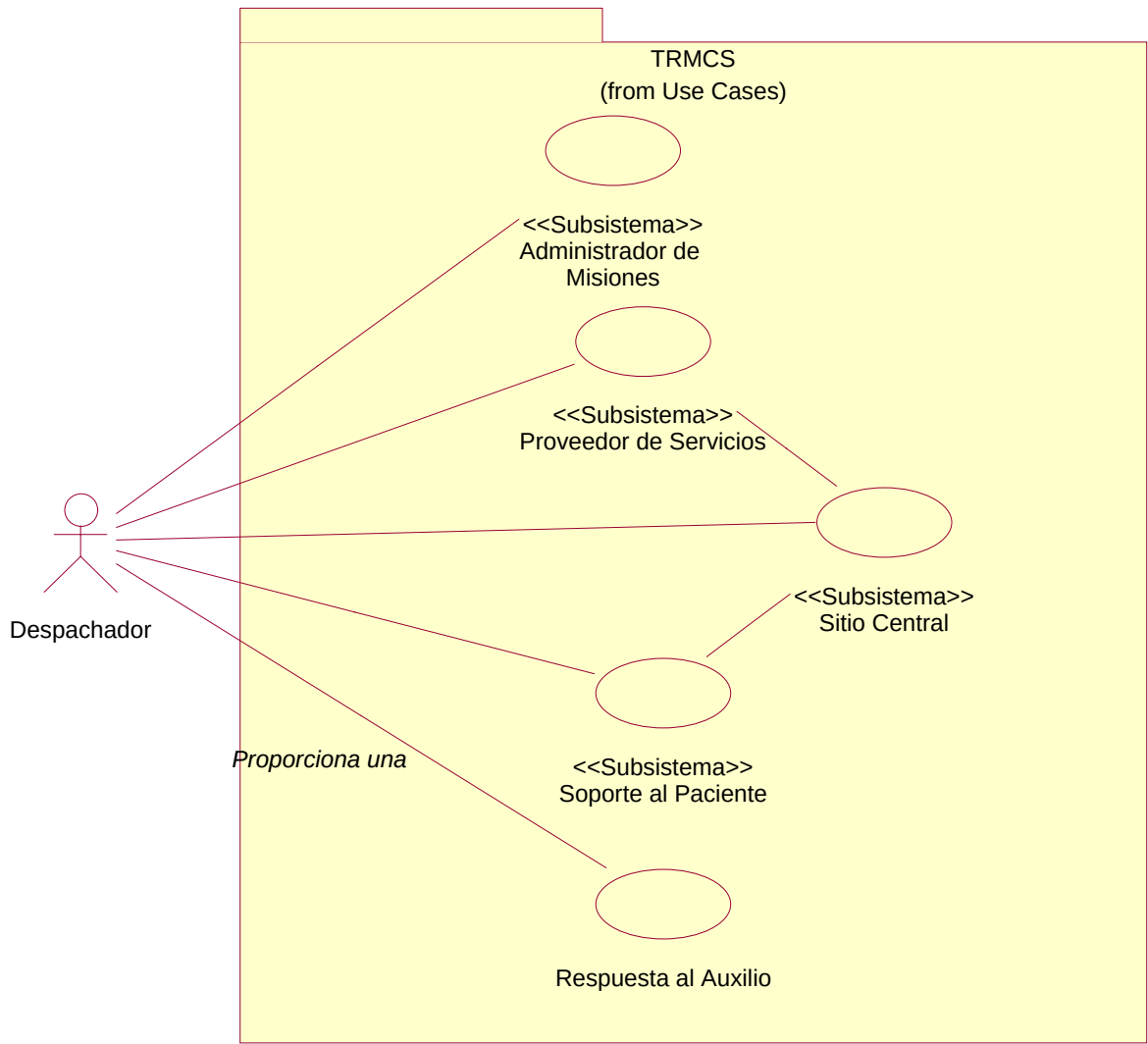


Caso de uso "Suscribir"



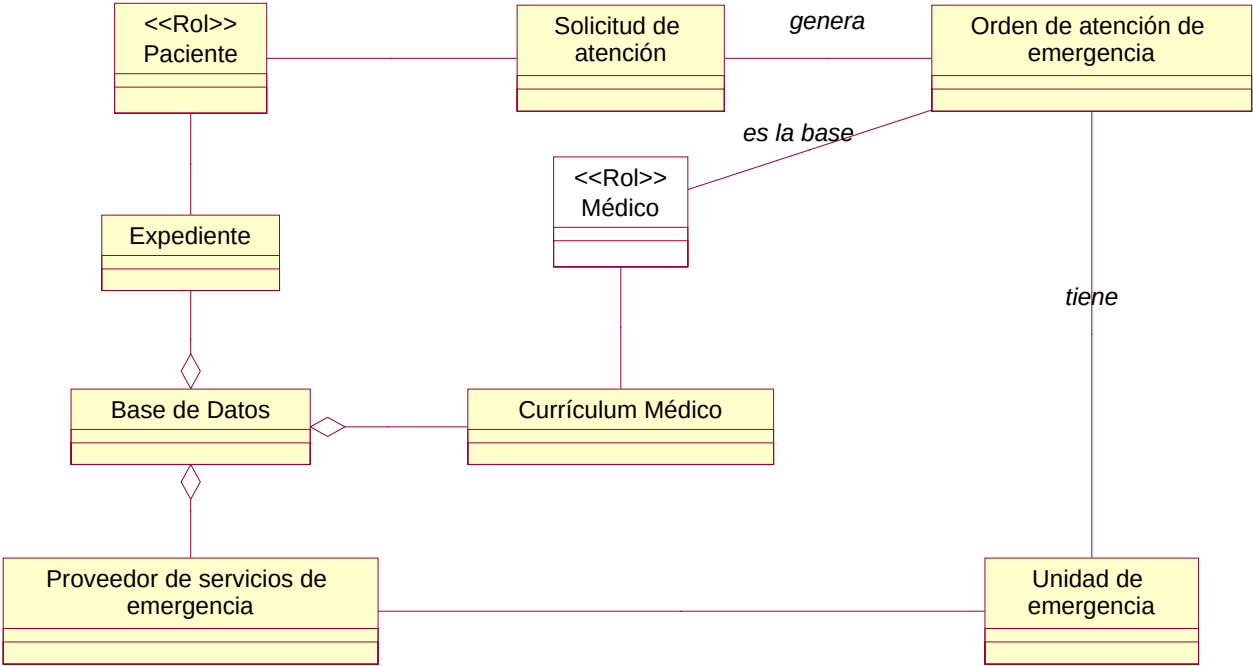






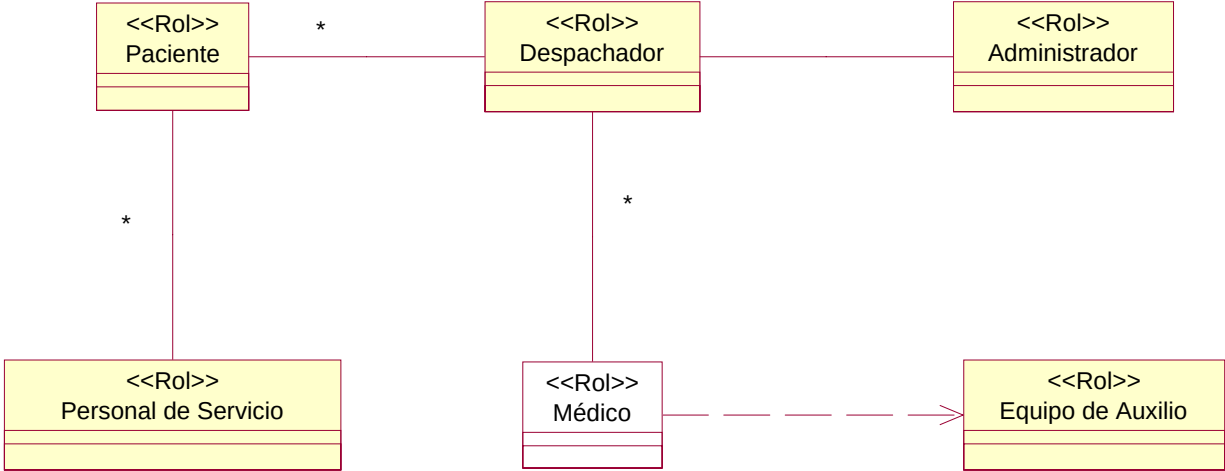
Modelo Conceptual

Modelo Conceptual



Diagramas de Roles

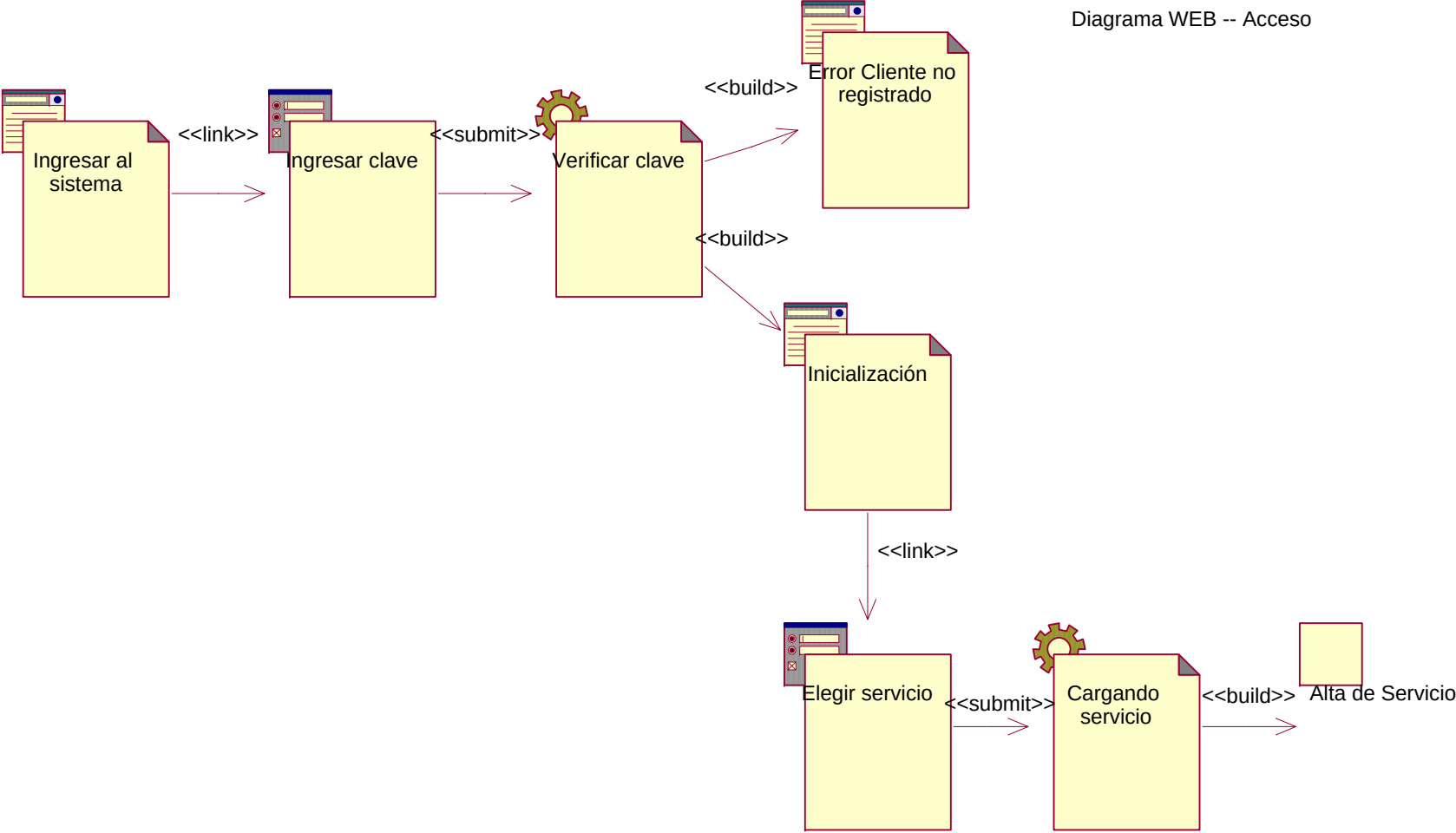
Diagrama de Roles



MODELO WEB

Diagramas WEB

Diagrama WEB -- Acceso



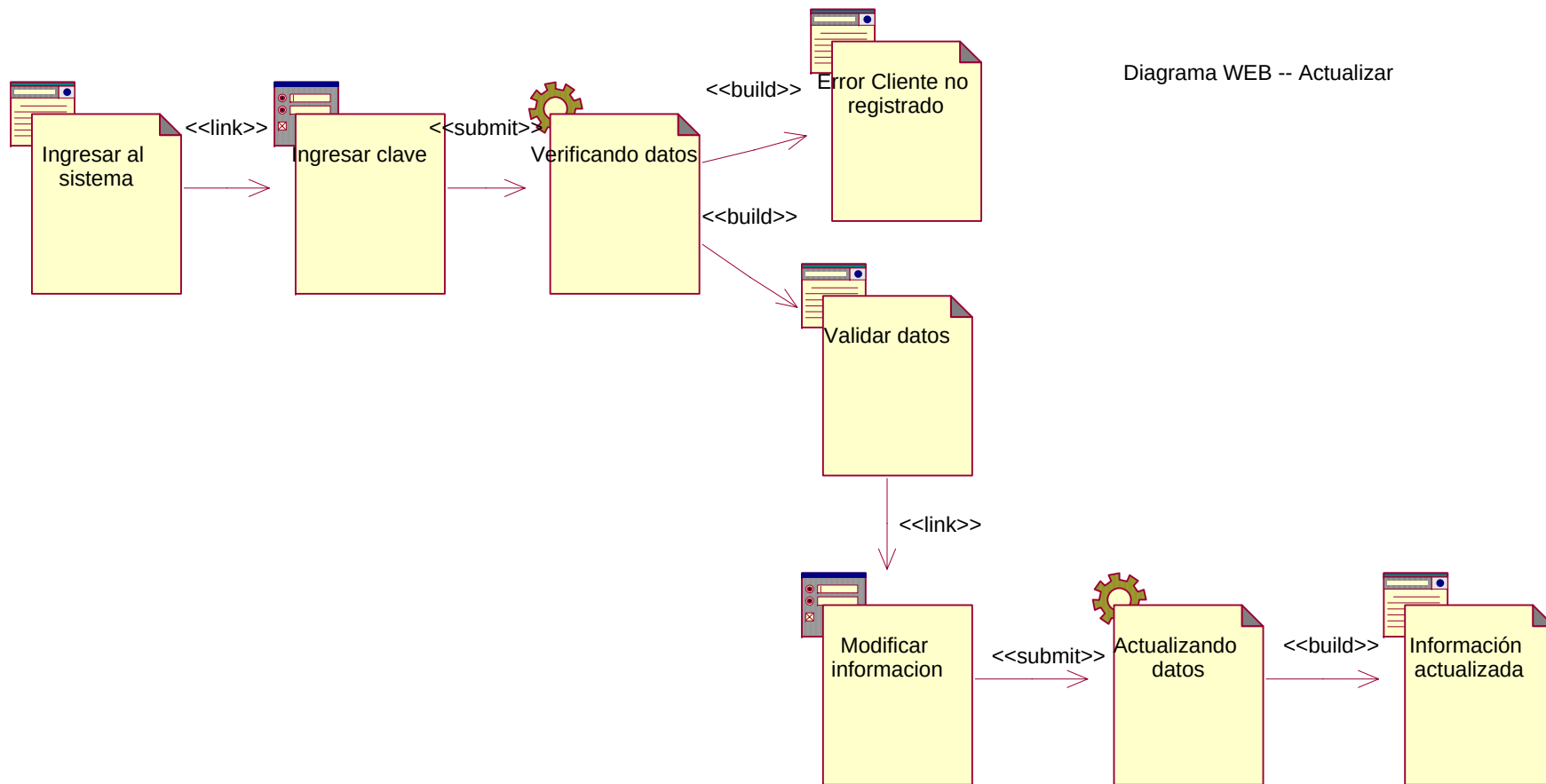
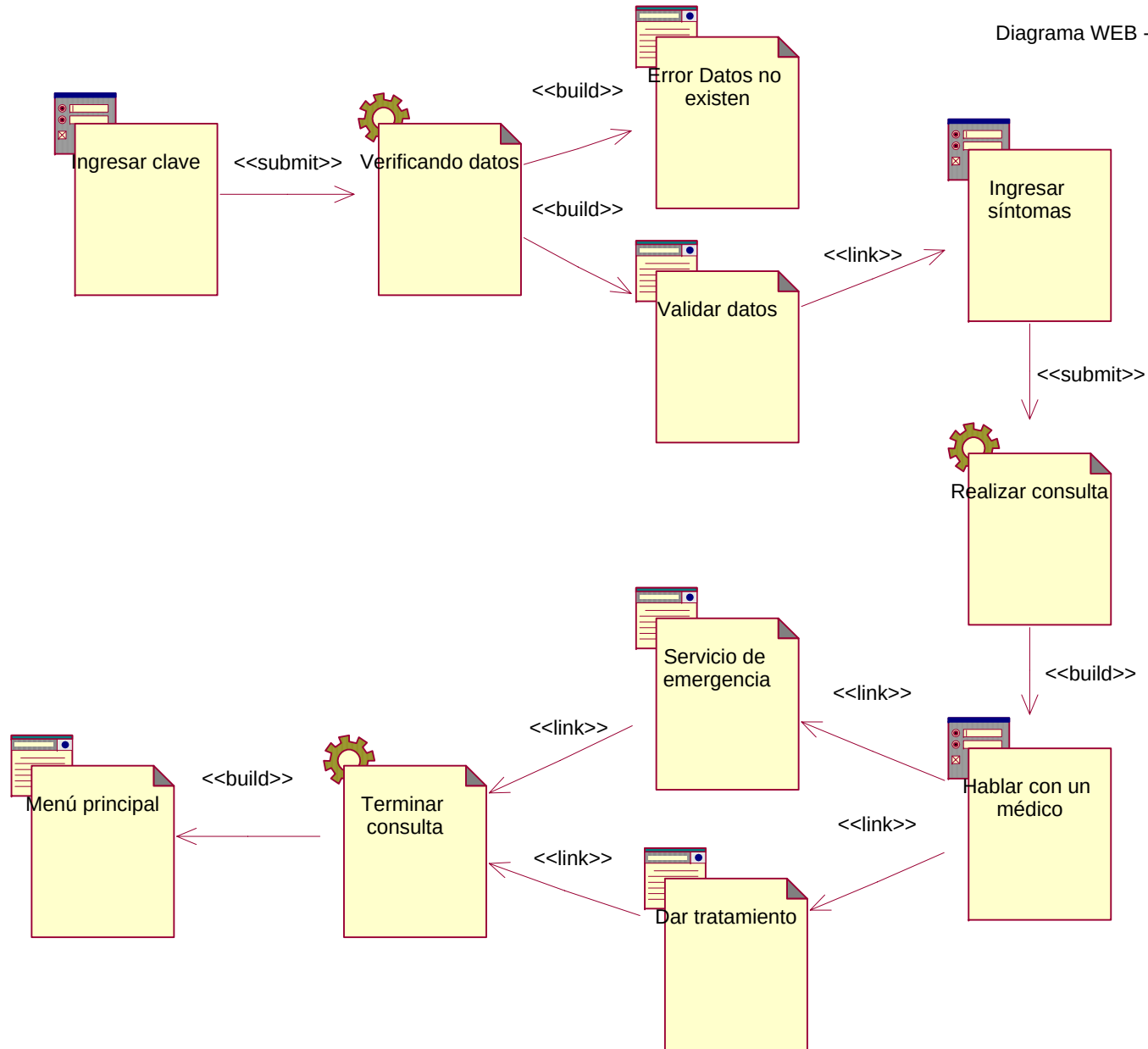
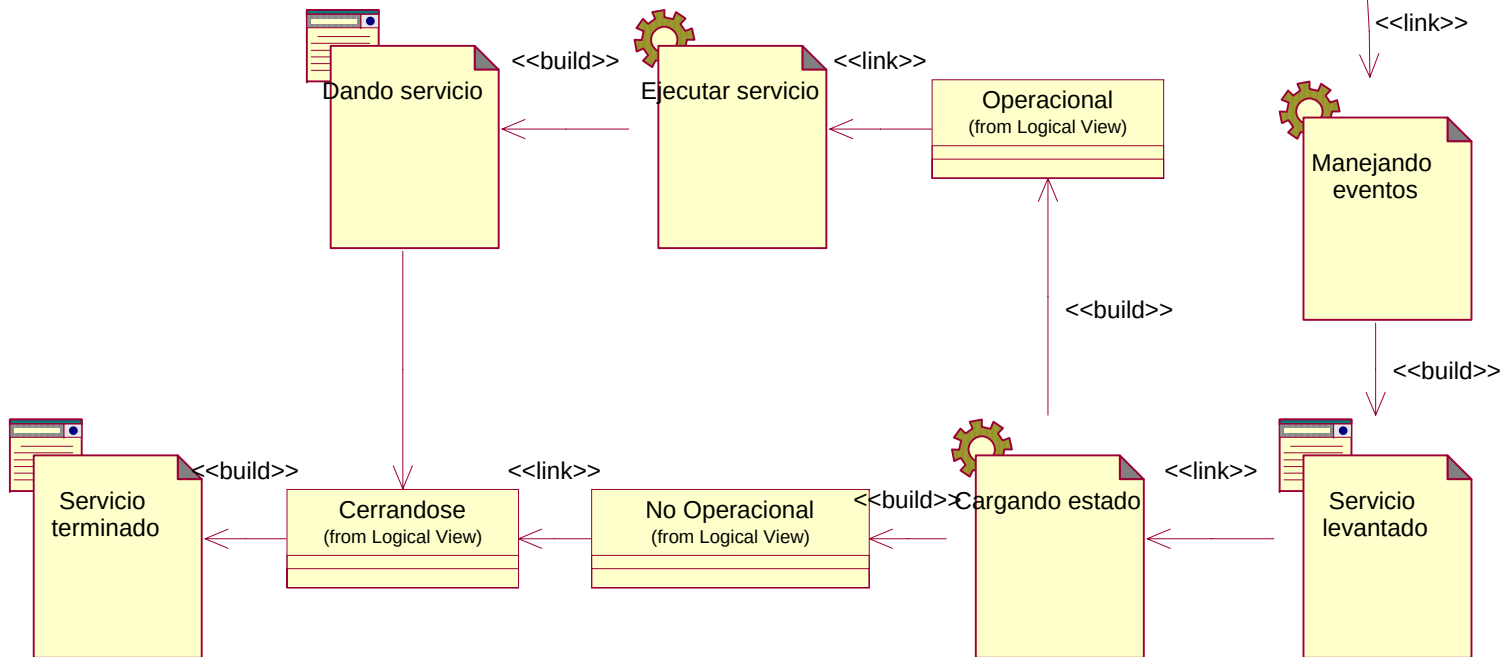
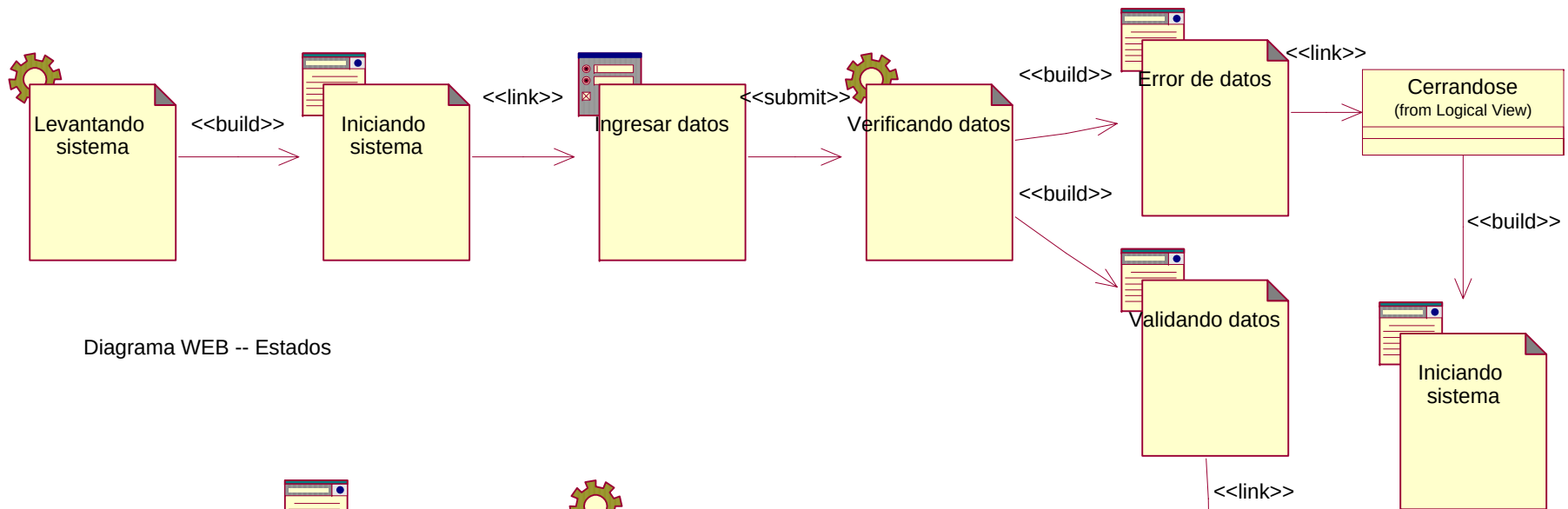


Diagrama WEB -- Actualizar

Diagrama WEB -- Consulta





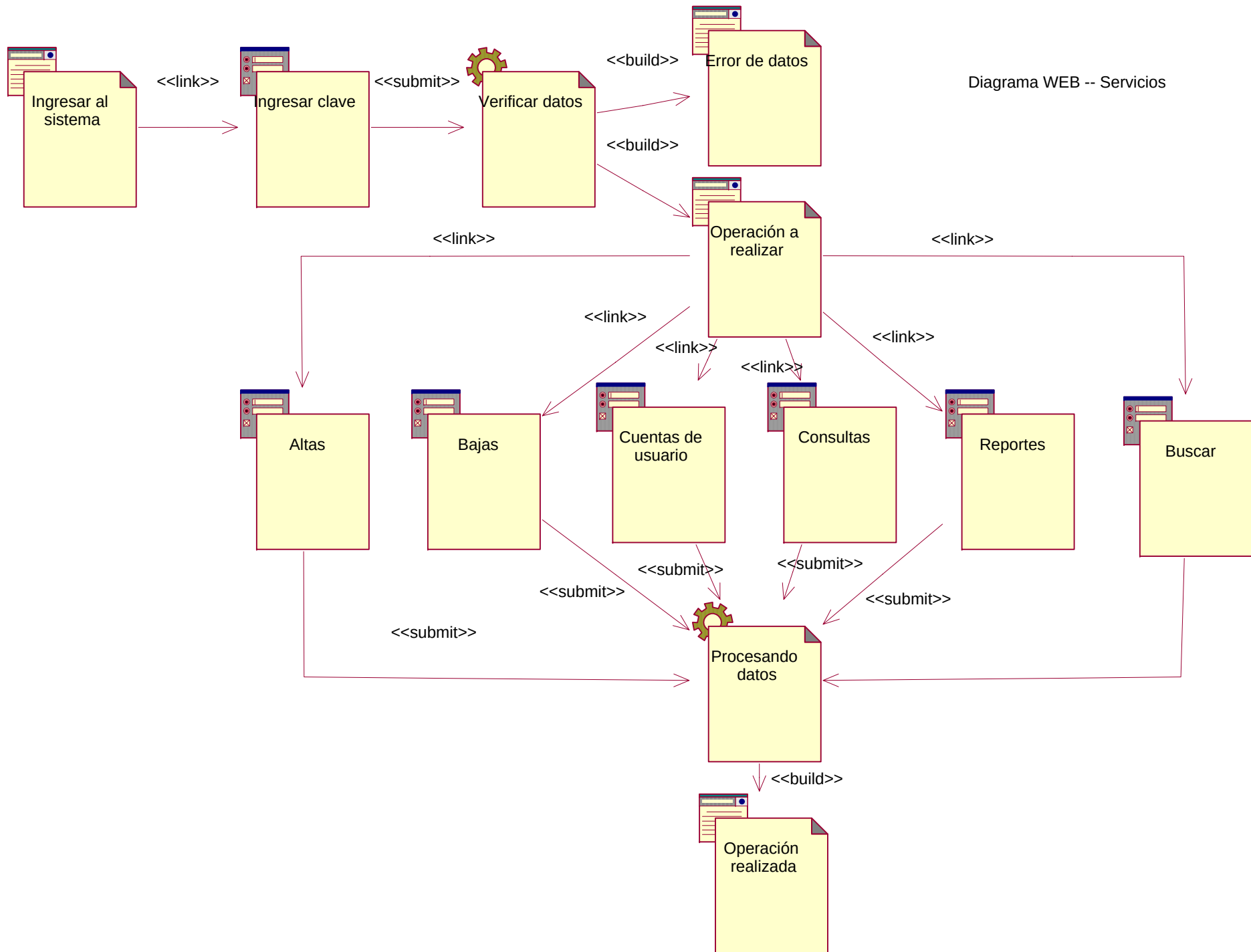
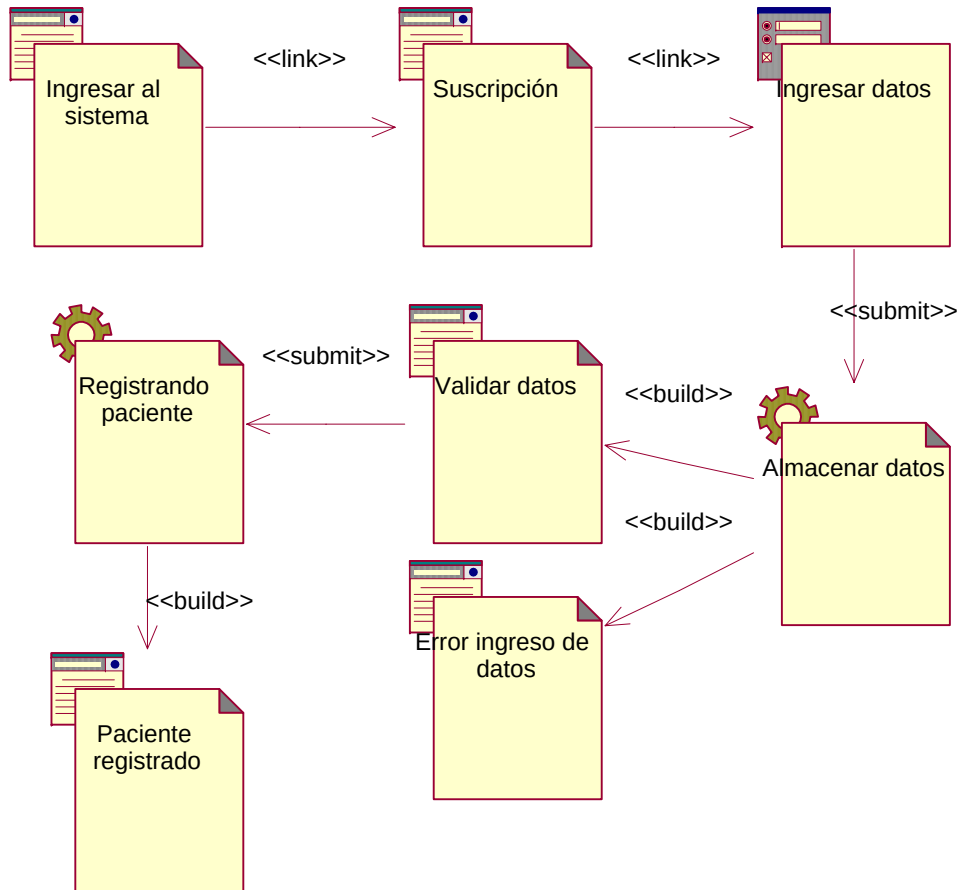
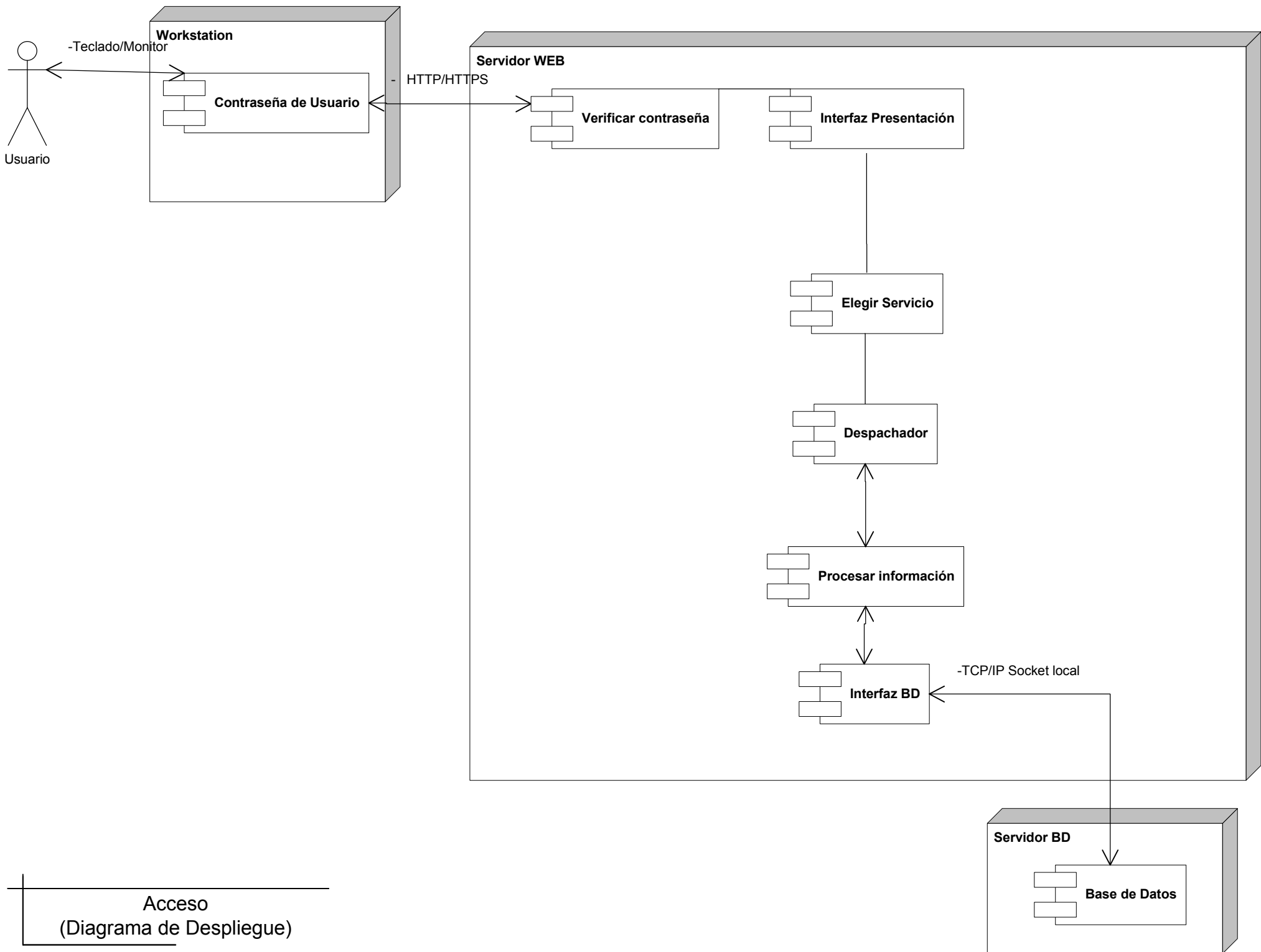
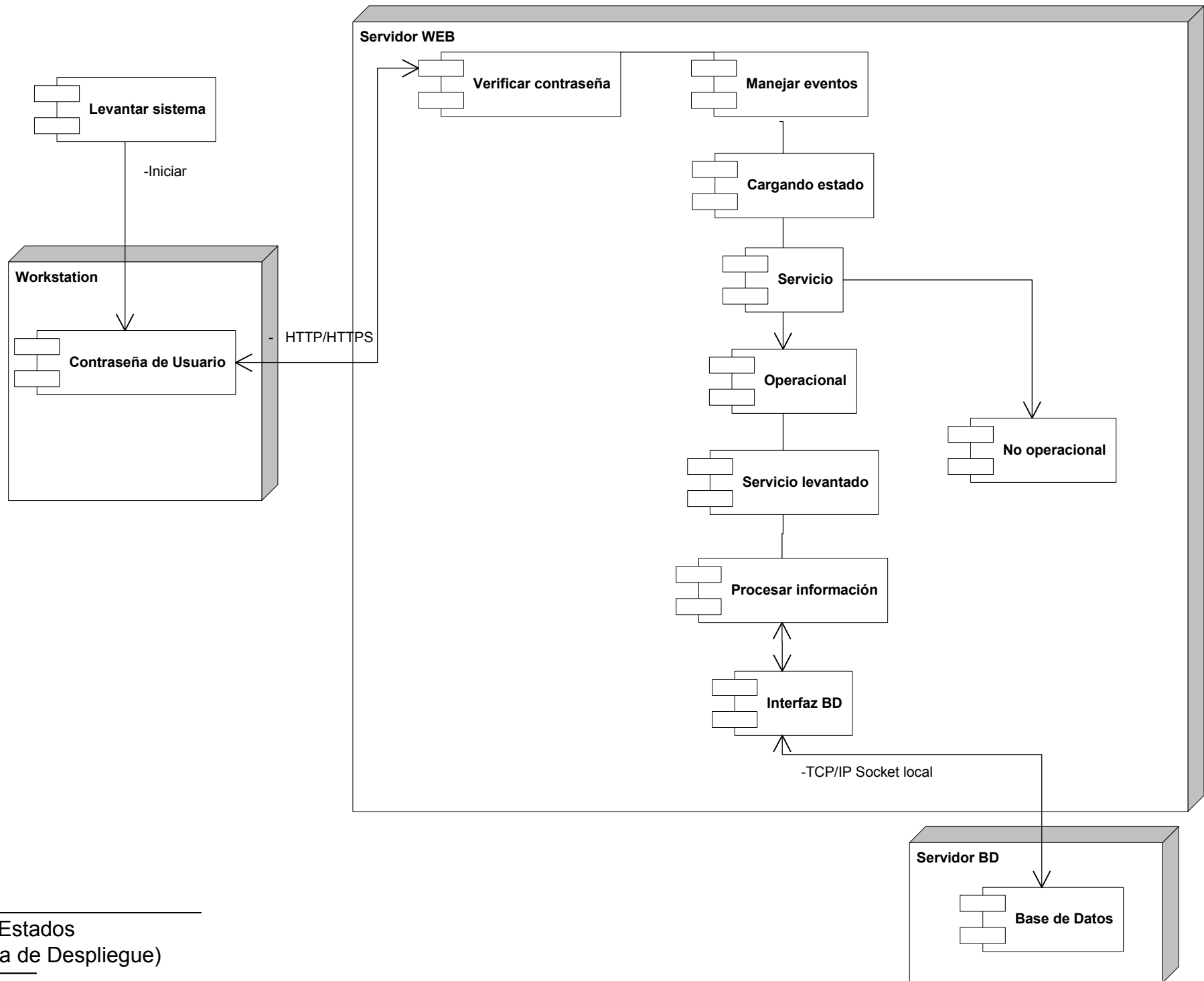


Diagrama WEB -- Suscribir

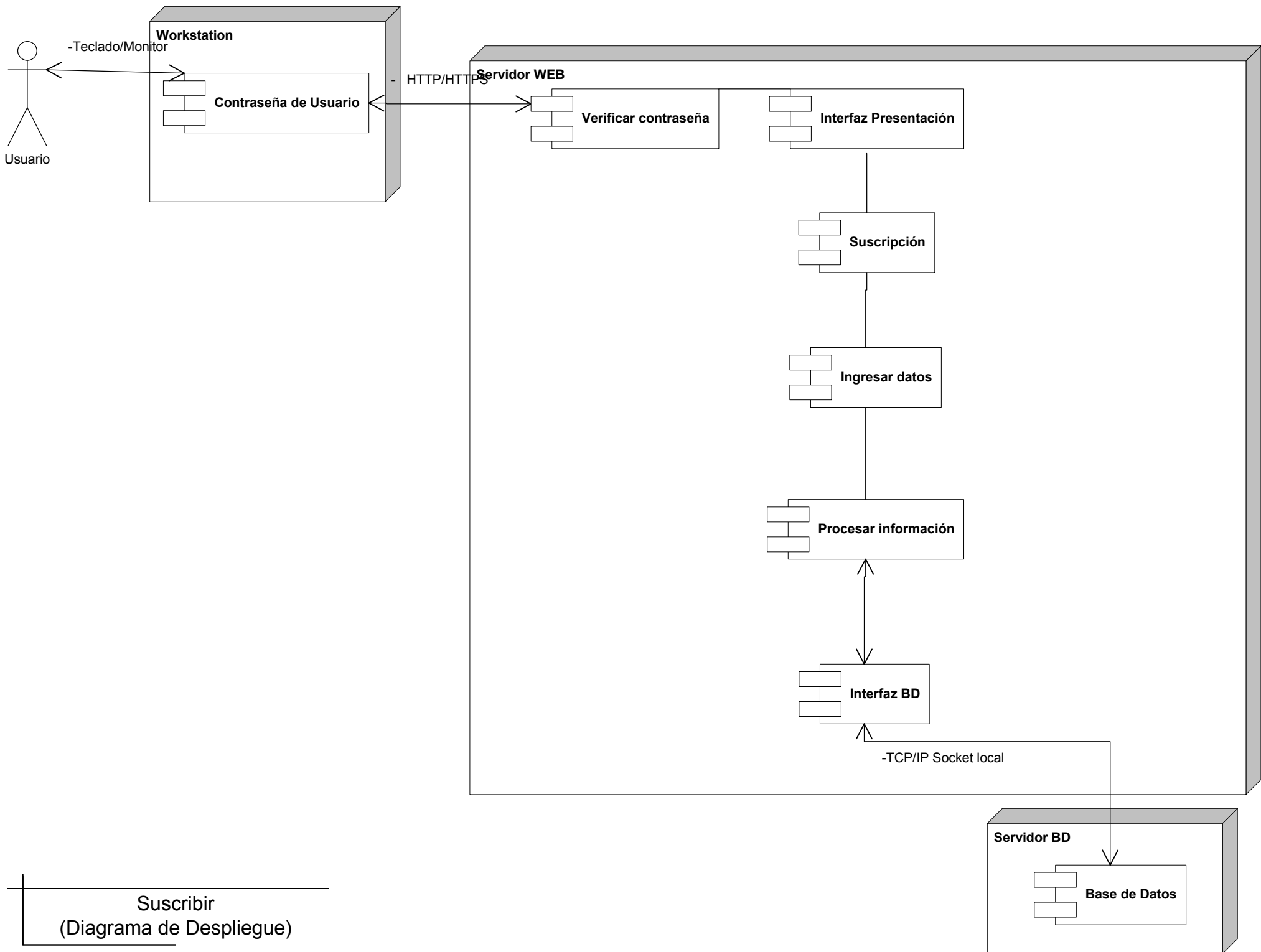


Diagramas de Despliegue

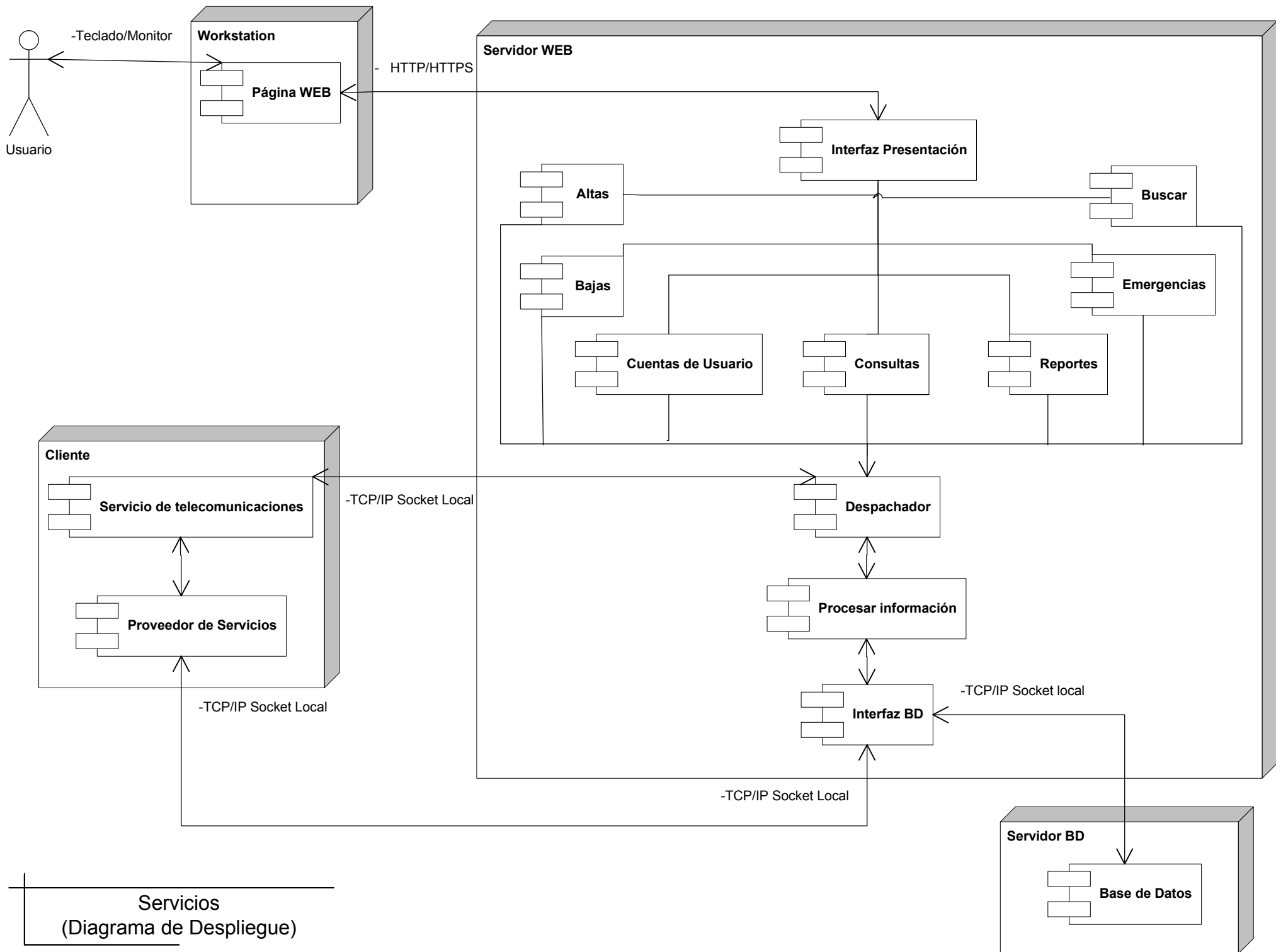




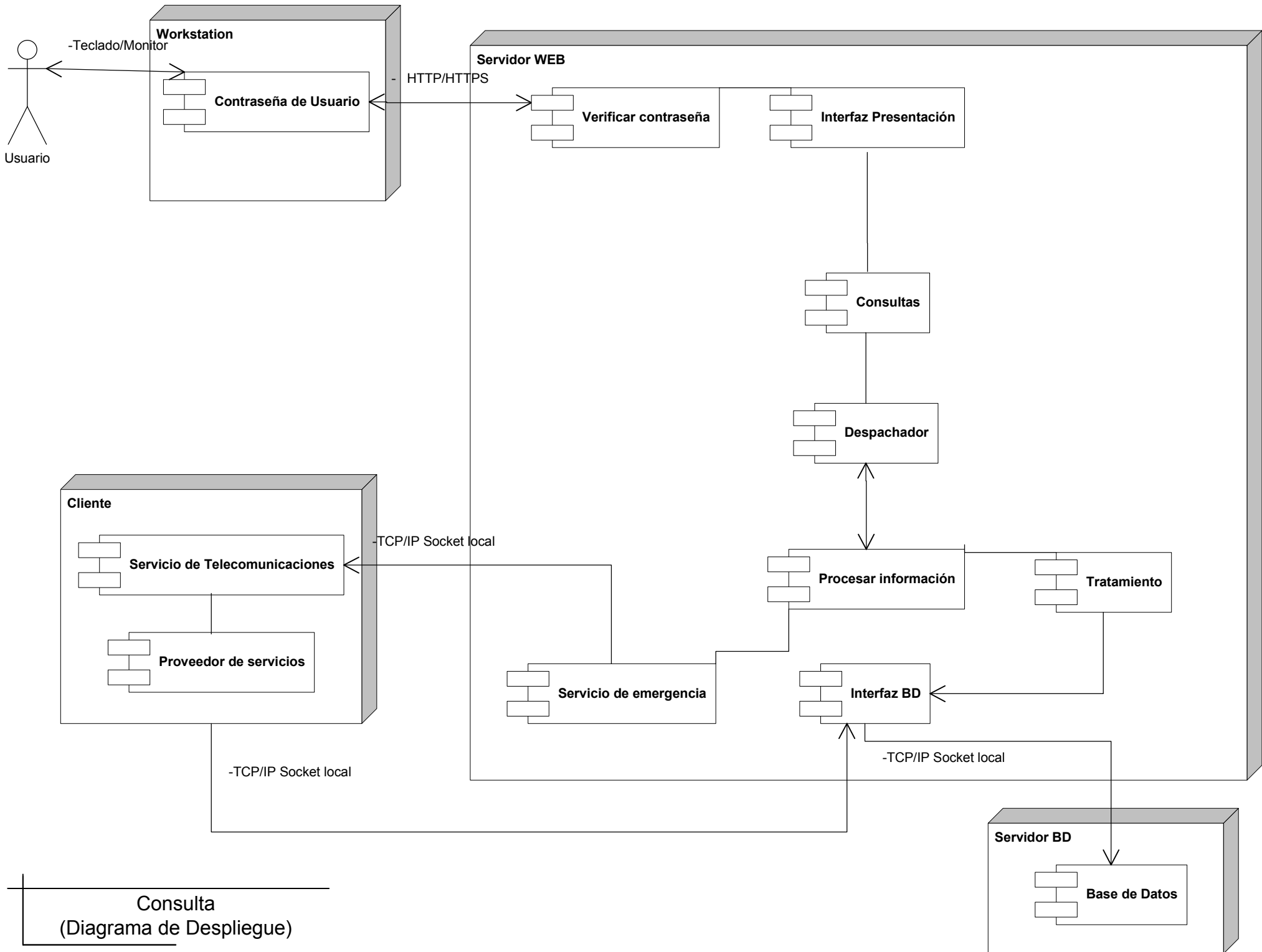
Estados
(Diagrama de Despliegue)



Suscribir
(Diagrama de Despliegue)



Servicios
(Diagrama de Despliegue)



Consulta
(Diagrama de Despliegue)

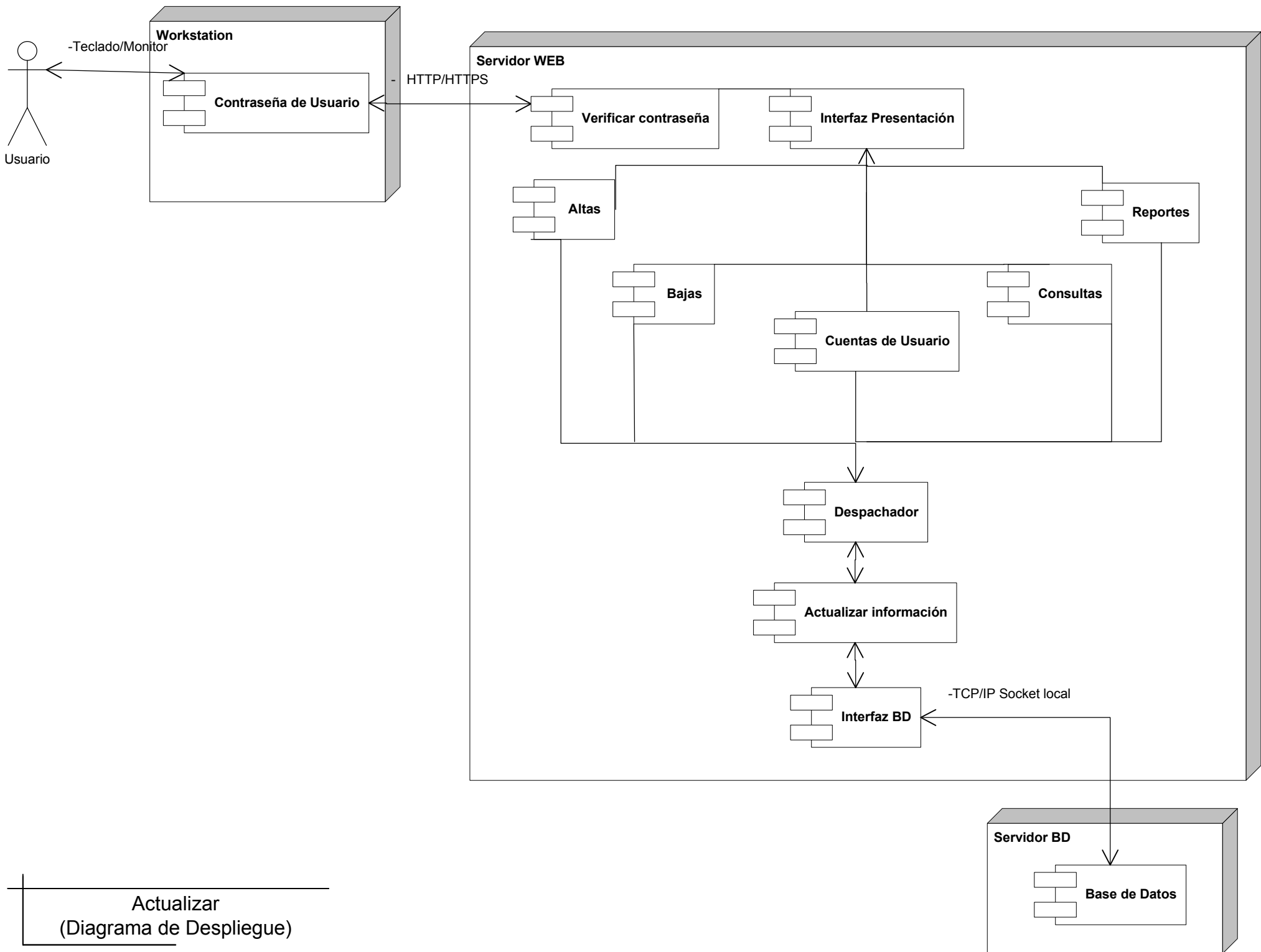


Diagrama de Estructura

TRMCS

Diagrama de Arquitectura

BROWSER

HTTP/HTTPS

SERVIDOR WEB

J2EE Aplicación Servidor

JDBC

Lenguajes
de
Desarrollo

Servicios
WEB

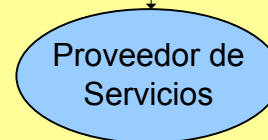
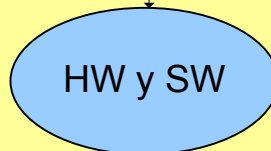
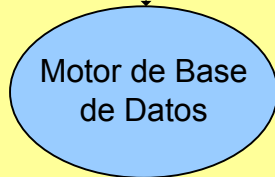
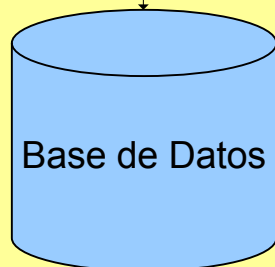
Base de Datos

Motor de Base
de Datos

HW y SW

Proveedor de
Servicios

Servicio de
Telecomunicaciones



Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

Dra. Rocío Morales Salgado
PRESENTE

Por medio de la presente, el que suscribe **Dr. Abraham Sánchez López**, profesor-investigador de la Facultad de Ciencias de la Computación de la Benemérita Universidad Autónoma de Puebla, en calidad de asesor del candidato a Dr. **Pedro Vargas García**. Le comunico que se ha revisado en su totalidad la tesis que avala el trabajo de investigación desarrollado por el C. Dr. Pedro Vargas, título de la tesis: ***Una estrategia para el modelado de negocios en la Web***. Motivo por el cual le solicito a usted tenga a bien asignar fecha y jurado para la defensa de su trabajo de tesis.

Sin más por el momento y en espera de la atención a la presente, quedo de ustedes.

Atentamente
“Pensar bien para vivir mejor”
H. Puebla de Z., a 15 de Mayo de 2009



Dr. Abraham Sánchez López
Profesor/Investigador de la FCC-BUAP
Asesor