UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOÁTEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

"DESARROLLO DE UN SOFTWARE DE AUTOMATIZACIÓN PARA EL CONTROL DE LAS ACTIVIDADES DEL ÁREA DE SALUD, ADSCRITA A LA DELEGACIÓN ESTUDIANTIL, NÚCLEO ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"

Realizado por

Vázquez B Ricardo J **Sequea A Luisandro J**

Trabajo de grado presentado como requisito parcial para optar al Título de

INGENIERO EN COMPUTACIÓN

Barcelona, Agosto de 2010

UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOÁTEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

"DESARROLLO DE UN SOFTWARE DE AUTOMATIZACIÓN PARA EL CONTROL DE LAS ACTIVIDADES DEL ÁREA DE SALUD, ADSCRITA A LA DELEGACIÓN ESTUDIANTIL, NÚCLEO ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"

TUTOR.

Ing. Cortínez Claudio

TUTOR ACADÉMICO

Barcelona, Agosto de 2010

UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOÁTEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

"DESARROLLO DE UN SOFTWARE DE AUTOMATIZACIÓN PARA EL CONTROL DE LAS ACTIVIDADES DEL ÁREA DE SALUD, ADSCRITA A LA DELEGACIÓN ESTUDIANTIL, NÚCLEO ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"

JURADO CALIFICADOR:

Ing. Mujica Yulitza
Jurado Principal

Ing. Rodríguez Rhonald
Jurado Principal

Ing. Claudio Cortínez

Tutor Académico

Barcelona, Agosto de 2010

ARTÍCULO N° 41 Del Reglamento de Trabajo de Grado

"Los Trabajos de Grado son de exclusiva propiedad de la Universidad y sólo podrán ser utilizados a otros fines con el consentimiento del Consejo de Núcleo respectivo, quien lo participará al Consejo Universitario."

Dedicatoria

A mi hijo Andrés Sebastián Vázquez P que desde su nacimiento ha sido una inspiración y que mi esfuerzo y constancia al cumplir esta meta, le sirva de ejemplo para enfrentar con fortaleza y empeño todos las circunstancias que la vida le pueda brindar. Te Amo mucho hijo.

Es importante el empeño, que tiene todo estudiante para desarrollar una tesis de grado, pero en mi caso no es más importante como el apoyo incondicional que he recibido por parte de mi esposa sin ella no hubiese logrado este gran sueño. Todo lo que soy, todo lo que tengo te lo debo a ti, gracias madre por darme el privilegio de nacer y conocer lo maravilloso que es este mundo

A mis Suegros Clemilde Bruzual, Luis esteban Perdomo, A Elisabeth Azocar, María Fernanda Perdomo por todo el apoyo prestado en todos estos momentos que siempre los necesitamos en el cuidado de nuestro hijo.

Ricardo J Vázquez B

Dedicatoria

A mis padres luisa y Alejandro por todo su apoyo, cariño y compresión durante toda mi carrera a ellos mil bendiciones.

Luisandro J. Sequea A

Agradecimiento

Gracias a Dios por estar siempre a mi lado, guiándome por el camino correcto y dándome fuerzas para superar cada una de sus pruebas.

A mis padres que a pesar de todo en su momento me prestaron todo su apoyo y aportaron un granito de arena para la culminación de mi tesis gracia los quiero mucho…

A mis suegros por todo y dulzura brindadas a mi hijo, por sus cuidados que yo como padre y tesista en ocasiones no lo pude hacer. Papá gracias por tu comprensión y toda la paciencia que tuviste en estos últimos días de tesis.

A mi hijo, por ser todo un **HOMBRE SITO** y portarse bien para así poder terminar esa larga tarea que le habían mandado a su padre en la UDO.

A Javier por ser mi compañero y mi amigo, gracias por su apoyo en esos momentos tan difíciles en el desarrollo de mi tesis.

A María Luisa, por su constante pregunta: "**Para cuando Gordo**" que me motivaba a terminar la tesis.

A mi pana Douglas Gene por su apoyo y momentos compartidos para la culminación de este proyecto mil gracias Mi pana

Agradezco humildemente a mis amistades y profesores. Todos ustedes me ayudaron con su apoyo incondicional a ampliar mis conocimientos y estar más cerca de mis metas profesionales.

A la Universidad de Oriente "la Casa más alta", por darme la oportunidad de cumplir uno de mis sueños (graduarme en esta casa de estudio).

Ricardo J Vázquez B.

Agradecimiento

A Dios Todo poderoso, a los Ángeles del cielo, por darme la iluminación y la fortaleza necesaria para así concretar este proyecto, amen.

A mis padres luisa y Alejandro por todo su apoyo, cariño y compresión durante toda mi carrera a ellos mil bendiciones, a mis abuelas Sinercia y Beatriz por estar muy pendiente .gracias por tantos afectos de amor y cariño, un agradecimiento muy especial a madre, amiga, Nora ballesta por todo su apoyo incondicional gracias por existir, también a mis hermanos(a) gracias a todos, a mis primos(a), y demás familiares.

A mis profesores Tirso, Zulirays, Claudio, Rhonald, Yulitza, Luis Felipe, pedro Dorta, gracias por sus enseñanzas. Al ing. Renee Guapache ,ing. Jesús Otamendi a todos el grupo de teleinformática UDO. Por su apoyo.

A mis compañeros de universidad a todos mil gracias, un reconociendo especial a Douglas por todo su apoyo.

Luisandro J Sequea A

RESUMEN

La delegación de Bienestar Estudiantil realiza todas las actividades del área de salud de forma manual, trayendo como consecuencia el retraso en el procesamiento de la información, por tal motivo se propuso el proyecto de Grado titulado: **"DESARROLLO DE UN SOFTWARE DE AUTOMATIZACIÓN PARA EL CONTROL DE LAS ACTIVIDADES DEL ÁREA DE SALUD, ADSCRITA A LA DELEGACIÓN ESTUDIANTIL, NÚCLEO ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"** el cual es un software de gestión basado en tecnología Web que permite a los usuarios realizar las operaciones necesarias para tal fin. Dicho sistema lleva por nombre SACMO, Este sistema maneja la información de los estudiantes como pacientes y sus registros médicos y odontológicos, registra las Historias tanto médicas como Odontológicas de los estudiantes, genera reportes de las consultas efectuadas por el personal de la delegación de Bienestar Estudiantil. Se utilizó la metodología del proceso Unificado de Desarrollo de Software (Jacobson, Booch y Rumbaugh), debido a su eficiencia en desarrollo de software aplicando las fases de inicio, elaboración, construcción y transición. Estas a su vez se llevan a cabo a través de una o varias iteraciones las cuales constan de cinco flujos de trabajo fundamentales como son: captura de requisitos, análisis, diseño, implementación y prueba, con su Extensión para Aplicaciones Web (WAE), esto en combinación con el nuevo esquema WebML. Este proyecto manipula bases de datos robustas que manejan gran cantidad de información, por tal motivo se evaluaron diversas estructuras de datos, seleccionando los árboles como la estructura ideal que se adoptó a nuestros requisitos. La arquitectura se construyó de acuerdo al patrón cliente-servidor, razón por la cual se utilizó el lenguaje de programación **PHP, MYSQL, APACHE, HTML Y EXTJS,** el cual representan la versión más novedosa en software Libre para desarrolladores muy apropiada para construir sistemas de información basados en tecnología Web. Dando como resultado una Aplicación Web robusta, eficiente y de fácil utilización.

Índice General

Índice de Figuras

ÍNDICE DE TABLAS

CAPÍTULO I

INTRODUCCIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

La Universidad de Oriente (UDO) Núcleo de Anzoátegui, ubicada en la Avenida Universidad de Barcelona, fue creada por resolución del Consejo Universitario el 20 de febrero de 1960, con el objetivo de brindarle a la población estudiantil de la zona nor-oriental la oportunidad de cursar estudios superiores de pre y postgrado formando así profesionales que responderían a las exigencias laborales de la región, cuyas actividades docentes iniciaron el 12 de febrero de 1963.

La Sección de Servicio Social inició sus actividades en el año 1960, bajo la denominación de "Servicio de Asistencia Económica y Social", dependiendo directamente del Vicerrectorado Académico; posteriormente paso a formar parte integrante de la Dirección de Servicios Estudiantiles, destacando entre sus programas más importantes lo referente a la asistencia de tipo económica, al ofrecer los siguientes servicios: Becas de Estudio, Becas de Transporte, Becas de Comedor , Préstamos Servicio de Vivienda.

A partir de la fecha señalada la Sección de Servicio Social ha ido evolucionando técnica y administrativamente, empleando otros métodos de trabajo y programas de acción, hasta lograr la organización actual, la cual está basada en las exigencias reales del estudiante universitario, los enfoques modernos del Trabajo Social y la re conceptualización del Bienestar Estudiantil. Desde la fundación de la Universidad de Oriente, se ha brindado este tipo de atención al estudiante, a través del Departamento Medico-Odonto1ógico y la orientación del mismo era hacia una labor preventiva-curativa

1.2 . EL PROBLEMA

En la actualidad dentro de la delegación de los servicios médicos estudiantiles en el área médico-odontológico se procesa una gran cantidad de información de forma manual; a raíz de este método de trabajo existe demora en el tiempo de búsqueda de la información; no hay seguridad en la información, ya que cualquier persona no autorizada puede tener acceso a ella y realizar modificaciones en algún dato de un registro.

Para solucionar estos problemas antes mencionados se propone el "Desarrollar un software para la automatización y el control Médico-Odontológico para la Universidad de Oriente Núcleo de Anzoátegui", dicho software se va a identificar con las siglas (S.A.C.M.O) y a través de él se optimizará y agilizará la manipulación de la información, beneficiando a los médicos de las unidades Medico y Odontológicos.

1.3 PRÓPOSITO

El propósito de éste trabajo de investigación será el de Desarrollar un Software de Automatización y control; que permita seguridad y confiabilidad, que contribuya a satisfacer los requerimientos de reducción de tiempo al momento de requerir información de las unidades Médico-Odontológico de los pacientes que son atendidos en el D.E.B.E. De esta forma se podrá simplificar e incrementar la productividad y eficiencia del mismo, permitiendo a su vez, automatizar las actividades anteriormente mencionadas. Con esto se pretende formar una base en el desarrollo de otros proyectos basados en software libre para el uso del departamento y de la universidad en general.

Este proyecto se llevará a cabo siguiendo las normas del Proceso Unificado de Desarrollo de Software PU (Proceso Unificado). Se deben crear módulos y procedimientos para permitir automatizar los procesos, que se hacen de manera manual y que acelerarían los servicios médicos odontológicos y mejoraría la efectividad en los diferentes tratamientos a realizar. Cabe destacar que este proyecto será elaborado con herramientas de Software Libre.

1.4 ALCANCE

El software está diseñado para el uso del personal médico de la delegación de servicios estudiantil del área médico odontológico del Núcleo Anzoátegui. No obstante se puede utilizar en cualquier otro núcleo de la universidad, ya que fue diseñado para esto.

Se espera que este software cumpla con las expectativas del personal que lo utilizará y que el mismo pueda ser extendido en el futuro, para satisfacer las necesidades que surgen en la delegación de servicios estudiantiles en el área Médico-Odontológico, ya que la aplicación ha sido diseñada de manera modular y flexible.

1.5 IMPORTANCIA

Debido a que la Universidad de Oriente es una Entidad Pública Nacional, es importante que se adapte a las normas del Decreto Nº 3390 lo más pronto posible. La aplicación de este proyecto en la delegación de servicios estudiantiles en el área Médico-Odontológico del Núcleo de Anzoátegui, contribuye con el proceso de transición de software propietario a software libre de esta Entidad de Educación Superior y a su vez con la Nación.

Además, la aplicación está diseñada para facilitar el desempeño de las labores asociadas con el control de las actividades del área de salud, adscrita a la delegación estudiantil, núcleo Anzoátegui de la Universidad de Oriente, lo cual es vital para el buen desempeño de la universidad como tal.

1.6 ORIGINALIDAD

La originalidad de éste proyecto, estará dada por ser el primero en el departamento de Computación y Sistemas y en las Unidades Médico- Odontológico del D.E.B.E en la Universidad de Oriente núcleo Anzoátegui, utilizando herramientas tecnológicas de punta como es Proceso Unificado de Modelado, mediante diagramas UML. Todo esto tomando en cuenta que será desarrollado usando aplicaciones web.

1.7 OBJETIVOS

1.7.1 Objetivo General:

"Desarrollar de un software de automatización para el control de las actividades del Área de Salud, adscrita a la Delegación Estudiantil, núcleo Anzoátegui de la Universidad de Oriente"

1.7.2 Objetivos Específicos:

- ✓ Identificar los requerimientos del software que permitirán el control de las actividades del Área de Salud de la Universidad.
- ✓ Elaborar el diseño inicial de la arquitectura candidata.
- ✓ Determinar los riesgos y requerimientos necesarios para la realización del proyecto.
- ✓ Diseñar la interfaz de usuario y realizar la codificación de los módulos que se utilizarán en el proyecto.
- ✓ Diseñar la base de datos para el almacenamiento de la información utilizada en el proyecto.
- ✓ Implantar el software diseñado.

CAPÍTULO II

MARCO TEÓRICO

2.1 ANTECEDENTES DE LA INVESTIGACIÓN

Tomando en cuenta que es la primera vez que se implementa un proyecto de este tipo utilizando el proceso unificado como herramienta y U.M.L (Lenguaje de Modelado Unificado) como metodología para la automatización y control de las actividades del Área de Salud, adscrita a la Delegación Estudiantil, núcleo Anzoátegui de la Universidad de Oriente. Es por esto que se ha decidido consultar trabajos que se han realizado con anterioridad, los cuales son similares o guardan alguna relación con el sistema a desarrollar. Entre ellos están:

María Gabriela Díaz Cova. "Diseño de un sistema de facturación para el manejo de las actividades administrativas que se realizan en una Clínica Municipal". Mayo de 2003. Este trabajo de grado es requisito parcial para la obtención del título de Ing. En Sistema en la Universidad de Oriente, Núcleo Anzoátegui. En dicho trabajo de grado, el objetivo principal fue diseñar un sistema para la automatización de las actividades administrativas que se realizan en una clínica municipal. Usando Proceso Unificado de Desarrollo de Software. (Díaz Cova, 2003)

Giampaolo R. Joseph, Rigual C. Rene J "Diseño de un sistema de planificación estratégica gerencial de un centro de especialidades Médicas para el mejoramiento Operativo". De 2003. Este trabajo de grado es requisito parcial para la obtención del título de Ing. En Sistema en la Universidad de Oriente, Núcleo Anzoátegui. En dicho trabajo de grado, el objetivo principal fue diseñar un sistema para la automatización de las actividades gerenciales de un centro de especialidades Médicas para el mejoramiento operativo. Utilizando el Proceso Unificado de Desarrollo de Software. (Giampaolo R & Rigual, C, 2003)

Adrián Eduardo Moya Guzmán. “Desarrollo de un Sistema para la Automatización del Proceso de Admisión y Control de Estudios en un Instituto Universitario”. Enero de 2.004. Este trabajo de grado, es requisito parcial para la obtención del título de Ingeniero en Computación en la Universidad de Oriente, Núcleo de Anzoátegui. El objetivo primordial era la automatización del proceso de admisión y control de estudios del Instituto Universitario Superior de Oriente y desarrollo Utilizando el Proceso Unificado de Desarrollo de Software. (Adrian, 2004)

Miguel Eduardo Simoni González. “Desarrollo de un Sistema de Información para la Automatización los Procesos Realizados en el Departamento de Ciencias de la Unidad de Estudios Básicos de la Universidad de Oriente, Núcleo de Anzoátegui”. Febrero de 2.004. Este trabajo de grado es requisito parcial para la obtención del título de Ingeniero en Computación en la Universidad de Oriente, Núcleo de Anzoátegui. El objetivo primordial era la gestión de procesos administrativos de las actividades del Departamento de Ciencias de la Unidad de Estudios Básicos de la Universidad de Oriente núcleo de Anzoátegui y su diseño fue elaborado Utilizando el Proceso Unificado de Desarrollo de Software. (Simoni, 2004)

Bárcenas Urbina, Beatriz y Cedeño Figueroa, Enrique “diseño de un plan estratégico para el departamento de registros médicos y estadísticas del Hospital Universitario Dr. Luís Razetti”. Diciembre 2001. Este trabajo de grado es requisito parcial para la obtención del título de Ingeniero en Sistemas en la Universidad de Oriente, Núcleo de Anzoátegui. El objetivo primordial era la Planificación de los registros Médico y estadísticos del Hospital Dr. Luis Razetti. Y su diseño fue elaborado Utilizando el Lenguaje de Modelado (UML) (Bárcenas & Cedeño, 2001)

2.2 Programación Orientada a Objetos

2.2.1 . ¿Qué es POO?

La programación orientada a objeto (POO) tiene la ventaja de ser un paradigma natural donde se pueden programar sistemas. Los seres humanos perciben el mundo como si estuviera formado por objetos: mesas, sillas, computadoras, coches, cuentas bancarias, partidos de fútbol, etc. También es un instinto humano intentar organizar estos objetos disponiéndolos de una forma concreta, optando por destacar determinadas características de algunos objetos que los destacan de otros.

Los niveles de dichas categorías y los métodos de clasificación de objetos en el mundo son infinitos. La forma en que las personas clasifican las cosas depende, en gran medida, de lo que deseen hacer con ellas y las características que más les llamen la atención. A la vez que se agrupan los objetos atendiendo a esquemas de clasificación, también se tiende a resaltar determinados atributos de objetos mostrando su preferencia sobre otros.

Esta idea de crear jerarquías de objetos relacionados se utiliza en la programación orientada a objetos. En 1960, los investigadores ya observaron que muchas de las entidades del modelo de programas de computadoras se podían nombrar y que se podían describir sus propiedades y comportamiento. Se dieron cuenta de que los programas estaban relacionados con cuentas bancarias, matrices, archivos y usuarios; en definitiva, algo parecido a los objetos en el mundo real.

La programación orientada a objetos se puede describir rápidamente como la identificación de objetos importantes, su organización en jerarquías, la adición de atributos a los objetos que describen las características relevantes en el contexto del problema y de la adición de las funciones (métodos) a los objetos para realizar las tareas necesarias en el objeto. Los detalles son algo más complejo, pero lo fundamental es que se trata de un proceso simple y natural.

No obstante, que sea un proceso simple y natural no significa que sea sencillo, ya que un conjunto de objetos se podría clasificar de muchas formas distintas. La clave es la posibilidad de identificar los atributos importantes de objetos y formar abstracciones y jerarquías idóneas. Incluso en el contexto de un dominio problemático, a veces resulta bastante difícil determinar los niveles correctos de abstracción y jerarquías de clasificación correctas. Simplemente, la decisión de la clase o grupo al que un objeto pertenece puede ser una tarea bastante difícil. (Schach, 2005)

2.2.2 Características de los Lenguajes Programación Orientada a Objetos

Los lenguajes de programación orientada a objetos (como C++, C# y Java) se caracterizan por tres conceptos clave: encapsulación, herencia y polimorfismo, que son compatibles con este aspecto natural de identificación y clasificación de objetos. (Joyanes , 1998)

2.2.3 Encapsulación

La encapsulación facilita la compresión de los grandes programas; la ocultación de datos les hace más eficaces. Los objetos pueden interactuar con otros objetos sólo a través de los atributos y métodos del objeto que se muestran públicamente. Cuantos más atributos y métodos se muestren públicamente, más difícil será modificar la clase sin que ello afecte al código que utiliza la clase. Una variable oculta se podría cambiar de un Long a un doublé, sin que ello afecte al código que utilicen los objetos creados (instanciados) de esa clase. El programador solo se debería preocupar por los métodos en la clase que han tenido acceso a esa clase, en lugar de por todos los sitios en el programa que un objeto ha instanciado desde los que se puede llamar a la clase. (Joyanes , 1998))

2.2.4 Herencia

La herencia proporciona dos ventajas evidentes a los programadores. La primera, y más importante, es que permite crear jerarquías que expresen las relaciones entre los diferentes tipos. Imagine que tiene dos clases, *CuentaAhorro* y *CuentaCorriente*, que

proceden de la clase principal Cuenta. Si tiene una función que necesite una clase Cuenta como argumento, podrá pasarle una *CuentaAhorro* o una *CuentaCorriente*, ya que ambas clases son de tipos de Cuenta. Cuenta es una clasificación general, mientras que *CuentaCorriente* y *CuentaAhorro* son tipos más específicos.

La segunda ventaja es que las clases pueden heredar características más generales de las clases superiores dentro de la jerarquía. En lugar de desarrollar nuevas clases a partir de cero, las clases nuevas podrán heredar las funciones de las clases existentes y, a continuación, modificar o ampliar esta función. La clase principal desde la que la nueva clase hereda las propiedades se conoce como la clase básica y la nueva clase se denomina la clase derivada. (Josi)

2.2.5 Polimorfismo

Polimorfismo significa, fundamentalmente, que las clases pueden tener el mismo comportamiento, pero implementarse de distintas maneras. Esto resulta muy útil en términos de programación, ya que permite trabajar con tipos de objetos genéricos cuando lo que interesa no es cómo implementa cada clase las funciones.

Se puede clasificar el polimorfismo en dos grandes clases:

- **Polimorfismo dinámico** (**polimorfismo paramétrico**) es aquél en el que el código no incluye ningún tipo de especificación sobre el tipo de datos sobre el que se trabaja. Así, puede ser utilizado a todo tipo de datos compatible.
- **Polimorfismo estático** (**polimorfismo *ad hoc***) es aquél en el que los tipos a los que se aplica el polimorfismo deben ser explicitados y declarados uno por uno antes de poder ser utilizados. (Joyanes , 1998)

2.2.6 Clases y Objetos

Como su propio nombre lo indica, la programación orientada a objetos está relacionada con objetos. Un objeto se compone de datos que describen al objeto y las operaciones

que se pueden realizar en el objeto. No obstante, cuando se crea un programa, en realidad se declaran y definen clases, no objetos.

Una clase es un tipo definido por el usuario, encapsula tanto los datos como los métodos que funcionen en esos datos. Una clase es algo muy parecido a una plantilla, que se utiliza para crear (instanciar) objetos. (Joyanes , 1998)

2.2.7 Ventajas de la Programación Orientada a Objetos

Son tres las ventajas clave de la programación orientada a objetos: compresión, reutilización del código y posibilidad de ampliación. La división del código en clases contribuye a imponer una estructura a medida que crecen los programas. La idea es unir los programas orientados a objetos de clases escritas previamente, así como realizar las modificaciones necesarias para admitir los nuevos requisitos mediante la herencia para derivar nuevas clases a partir de las ya existentes. La creación de sistemas a partir de componentes que se pueden volver a utilizar conduce, naturalmente, a una mayor productividad, que suele ser la ventaja más citada de todos los métodos orientados a objetos.

Las características de la programación orientada a objetos también proporcionan varias ventajas. La encapsulación facilita la escala de pequeños sistemas a grandes sistemas. En buena medida, independientemente del tamaño del sistema, el programador simplemente está creando objetos.

Por último, la posibilidad de ocultar datos también genera sistemas más seguros. El estado de objetos sólo se puede modificar mediante métodos expuestos públicamente; esto aumenta la posibilidad de predecir el comportamiento del objeto. (Joyanes , 1998)

2.3 Desarrollo de Software

2.3.1 El Proceso Unificado de Desarrollo de Software

El proceso unificado es un proceso de desarrollo de software. Un proceso de desarrollo de software es un conjunto de actividades necesarias para transforma requisitos de un usuario en un sistema software. Sin embrago, el proceso unificado es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistema de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto.

El proceso unificado utiliza el UML para preparar todos los esquemas de un sistema software. De hecho, UML es una parte esencial del proceso unificado. (Jacobson & Rumbaugh, 2000)

2.3.2 . Proceso Unificado está dirigido por Casos de Uso

Un caso de uso es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado importante. Los casos de uso representan los requisitos funcionales. Todos los casos de uso juntos constituyen el modelo de caso de uso, el cual describe la funcionalidad total del sistema. Puede decirse que una especificación funcional contesta a esta pregunta: ¿Qué de hacer el sistema? La estrategia de los casos de uso puede describirse añadiendo tres palabras al final de esta pregunta: ¿par cada usuario? Estas tres palabras albergan una implicación importante, nos fuerzan a pensar en términos de importancia para el usuario y no solo en términos de funciones que sería bueno tener. Sin embargo, los casos de uso no son solo herramientas que se usan para especificar los requisitos de un sistema, también guían su diseño, implementación y prueba; estos guían el proceso de desarrollo. Basándose en el modelo de casos de uso, los desarrolladores crean una serie de modelos de diseño e implementación que llevan a cabo los casos de uso. Los desarrolladores revisan cada uno de los sucesivos modelos para que sean conformes al garantizar que los componentes del modelo de implementación implementan correctamente los casos de uso. De este modo, los casos de uso no solo inician el proceso de desarrollo sino que le proporciona un hilo conductor. Dirigidos por

casos de uso quiere decir que el proceso de desarrollo sigue un hilo – avanza a través de una serie de flujo de trabajos que parten de los casos de uso. Los casos de uso especifican, se diseñan y los caos de uso finales son la fuente a partir de la cual los ingenieros de prueba construyen sus casos de prueba. (Jacobson & Rumbaugh, 2000)

2.3.3 El Proceso Unificado está Centrado en la Arquitectura

El concepto de arquitectura de software incluye los aspectos estáticos y dinámicos más significativos del sistema. La arquitectura surge de necesidades de la empresa, como la perciben los usuarios y los inversores, y se refleja en casos de uso. Sin embargo, también se ve influenciada por muchos otros factores, como la plataforma en la que tiene que funcionar el software, los bloques de construcción reutilizables de que se dispone, consideraciones de implantación, sistemas heredados y requisitos no funcionales.

La arquitectura es una vista del diseño completo con las características más importantes resaltadas, dejando los detalles de lado. Debido a que lo que es significativo depende en parte de una valoración, que a su vez se adhiere con las experiencias, el valor de una arquitectura depende de las personas que se hayan responsabilizado de su creación. No obstante, el proceso ayuda al arquitecto a centrarse en los objetivos adecuados, como la comprensibilidad, la capacidad de adaptación al cambio y la reutilización.

Cada producto tiene tanto una función como una forma. Ninguna es suficiente por sí misma. Estas dos fuerzas deben equilibrarse para obtener un producto de éxito. En esta situación, la función corresponde a los casos de uso y la forma de la arquitectura. Debe haber interacción entre los casos de uso y la arquitectura. En realidad, tanto la arquitectura como los casos de uso deben evolucionar en paralelo.

Por tanto, los arquitectos moldean el sistema para darle una forma. En esta forma, la arquitectura, la que debe diseñarse para permitir que el sistema evolucione, no solo en su desarrollo inicial, sino también a lo largo de futuras generaciones. Para encontrar es

forma, los arquitectos deben trabajar sobra la comprensión general de las funciones clave, es decir, sobre los casos de uso claves del sistema. Estos casos de uso clave pueden suponer solo el 5 o 10 por ciento de todos los casos de uso, pero son los significativos, los que constituyen las funciones fundamentales del sistema.

Se utilizan patrones para describir la arquitectura de un sistema. Los patrones de la arquitectura se centran en estructuras e interacciones entre subsistemas en incluso entre sistemas. Existen mucho patrones de arquitectura, entre alguno tenemos el patrón Broker y el patrón Layers.

El patrón layers es aplicable a muchos tipos de sistemas. Este patrón define como organizar el modelo de diseño en capas, lo cual quiere decir que los componentes de una capa solo pueden hace referencia a componentes en capas inmediatamente inferiores. Un sistema con una arquitectura en capas, pone a los subsistemas de paliación individuales en lo más alto. Esto se construye a partir de subsistemas en las capas más bajas, como son los marcos de trabajo y las bibliotecas de clase. Obsérvese en la Figura 2.1, la capa general de la aplicación contiene los subsistemas que no son especificaos de una sola aplicación, sino que puedan ser reutilizados por muchas aplicaciones diferentes dentro del mismo dominio o negocio. La arquitectura de las dos capas superiores se crea a partir de los casos de uso. (Jacobson & Rumbaugh, 2000)

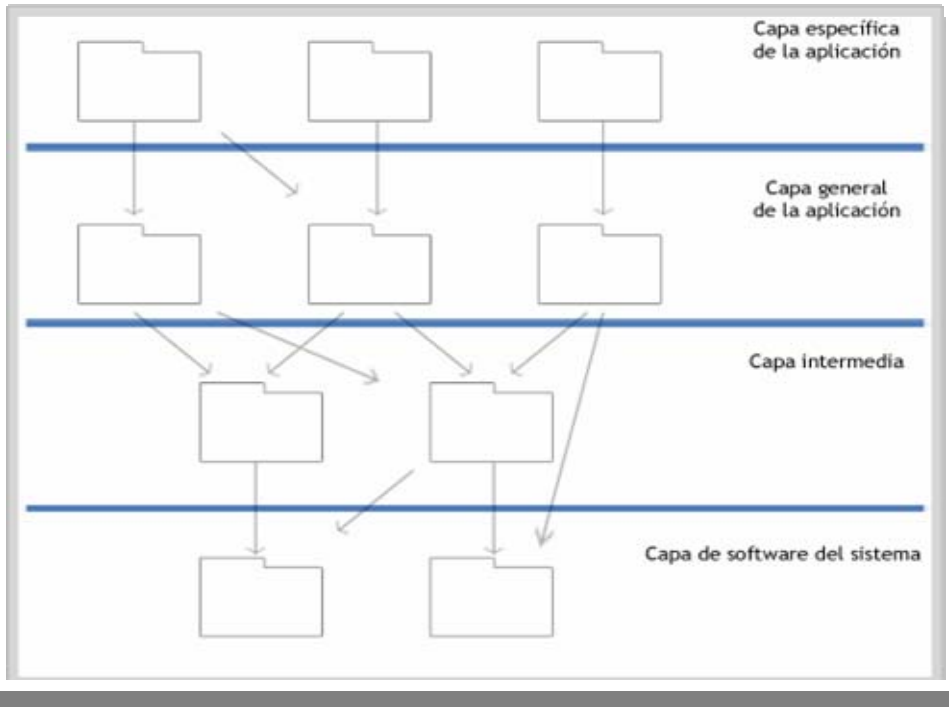


Figura 2.1: Patrones de capas de la arquitectura del sistema.
Fuente: **elaboración propia.**

2.3.4 Concepto de Método (o Metodología)

Una metodología de ingeniería del software es un proceso para producir software de forma organizada, empleando una colección de técnicas y convenciones de notación Predefinidas, Conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software.

La realidad, uno o varios modelos, derivados unos de otros, con el objetivo de lograr un modelo final o sistema. Así Un método es una guía que define las reglas de paso de un modelo a otro para evolucionar progresivamente hasta el modelo final. (Jacobson & Rumbaugh, 2000)

2.3.5 El Proceso Unificado es Iterativo e Incremental

El desarrollo de un producto de software comercial supone un gran esfuerzo que puede durar entre varios meses hasta posiblemente un año más. Es práctico dividir el trabajo en partes más pequeñas o mini proyectos. Cada mini proyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos de flujo de trabajo, y los incrementos, al crecimiento del producto. Para efectividad máxima, las iteraciones deben estar controladas; esto es, deben seleccionarse y ejecutarse de una forma planificada. Es por esto que son mini proyectos.

En primer lugar, la iteración trata un grupo de casos de uso que juntos amplían la utilidad del producto desarrollado hasta ahora. En segundo lugar, la iteración trata los riesgos más importantes. Las iteraciones sucesivas se construyen sobre los artefactos de desarrollo tal como quedaron al final de la última iteración. Por supuesto, un incremento no necesariamente es aditivo. Especialmente en las primeras fases del ciclo de vida, los

desarrolladores pueden tener que reemplazar un diseño superficial por uno más detallado o sofisticado. En las fases posteriores, los incrementos son típicamente aditivos.

Como se puede ver cada iteración tiene todo lo que tiene un proyecto de desarrollo de software, planificación, desarrollo de una serie de flujos de trabajo (requisitos, análisis, diseño, implementación y pruebas), y una preparación para la entrega. Las iteraciones se organizan dentro de cuatro fases (inicio, elaboración, construcción, y transición). Aunque cada iteración discurre a lo largo de los flujos de trabajo, tienen énfasis diferentes en las distintas fases, como se muestra en la Figura 2.2. Durante la fase de inicio y elaboración, la mayoría de los esfuerzos se dedica a la captura de requisitos y a un análisis y diseño preliminar. Durante la construcción el énfasis pasa al diseño detallado, la implementación y la prueba. El número de iteraciones planteado para cada fase depende, básicamente de la complejidad del sistema propuesto. Un proyecto muy simple podría ser realizado con una sola iteración por fase, mientras que un proyecto más complicado podría requerir más iteraciones. (Jacobson & Rumbaugh, 2000)

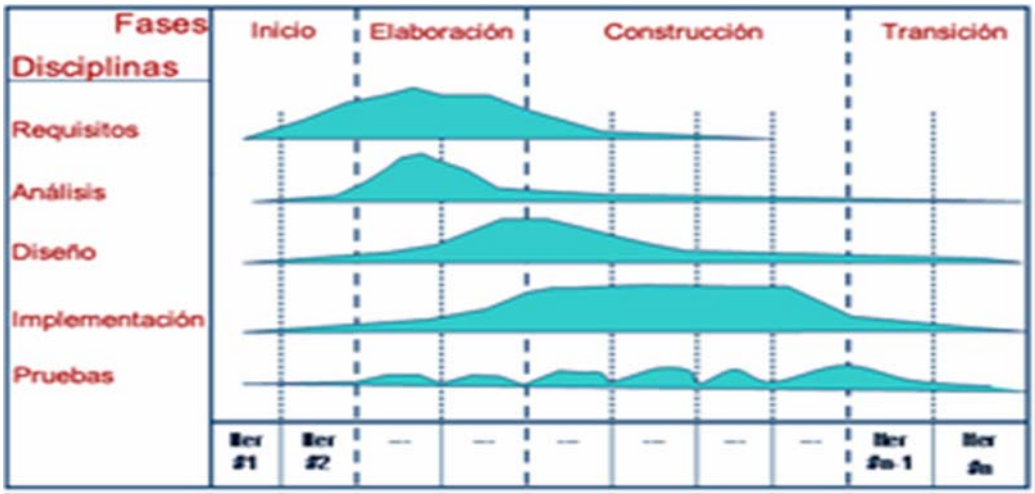


Figura 2.2: Relación de fases con las iteraciones del proceso unificado
Fuente: **(Jacobson & Rumbaugh, 2000)**

2.3.6 Fases del Proceso Unificado de Desarrollo de Software

2.3.6.1 Fase de Inicio.

Durante la fase de inicio, se desarrolla una descripción del producto final a partir de una buena idea y se presenta el análisis de negocio para el producto. Esencialmente, esta fase responde a las siguientes preguntas:

- ✓ ¿Cuáles son las principales funciones del sistema para sus usuarios más importantes?
- ✓ ¿Cómo podría ser la arquitectura del sistema?
- ✓ ¿Cuál es el plan de proyecto y cuanto costara desarrollar el producto?

La respuesta a la primera pregunta se encuentra en un modelo simplificado que contenga los casos de uso más críticos. Cuando los tengamos, la arquitectura es provisional y consiste típicamente en un simple esbozo que muestra los subsistemas más importantes. En esta fase, se identifican y priorizan los riesgos más importantes, se planifica en detalle de elaboración, y se estima el proyecto de manera aproximada.

2.3.6.2 **.** Fase de Elaboración

En esta fase se identifican en detalle la mayoría de los casos de uso del producto y se diseña la arquitectura del sistema. La relación entre la máquina del sistema y el propio sistema primordial.

Por tanto la arquitectura se expresa en forma de vistas de todos los modelos del sistema, los cuales juntos representan el sistema entero. Esto implica que hay vistas arquitectónicas del modelo de casos de uso, del modelo del análisis, del modelo de diseño, del modelo de implementación y del modelo de despliegue. La vista del modelo de implementación incluye componentes para probar que la arquitectura es ejecutable. Durante esta fase de desarrollo, se realizan los casos de uso más críticos que se identifican en la fase de comienzo. El resultado de esta fase es una la línea base de la arquitectura. Al final de la fase de elaboración, el director de proyecto está en disposición de planificar las actividades y estimar los recursos necesarios para terminar el proyecto. (Jacobson & Rumbaugh, 2000)

2.3.6.3 Fase de Construcción

En esta fase se crea el producto. Aquí la línea base de arquitectura crece hasta convertirse en el sistema completo. La descripción evoluciona hasta convertirse en un producto preparado para ser entregado a la comunidad de usuarios. El grueso de los recursos requeridos se emplea durante esta fase de desarrollo. Sin embargo, la arquitectura del sistema es estable, aunque los desarrolladores pueden descubrir formas mejores de estructurar el sistema, ya que los arquitectos recibirán sugerencias de cambios arquitectónicos de menor importancia. Al final de esta fase, el producto contiene todos los casos de uso que la dirección y el cliente han acordado para el desarrollo de esta versión. Sin embargo, puede que no esté completamente libre de defectos. Muchos de estos defectos se descubrirán y solucionaran durante la fase de transición. (Jacobson & Rumbaugh, 2000)

2.3.6.4 Fase de Transición

La fase de transición cubre el periodo durante el cual el producto se convierte en versión beta. En la versión beta un número reducido de usuarios con experiencia prueba el producto e informa de defectos y deficiencias. Los desarrolladores corrigen los problemas e incorporarán alguna de las mejores sugeridas en una versión general dirigida a la totalidad de la comunidad de usuarios. La fase de transición conlleva actividades como la fabricación, formación del cliente, el proporcionar una línea de ayuda y asistencia, y la corrección de los defectos en dos categorías: los que tienen suficiente impacto en la operación para justificar una versión incrementada y los que pueden corregirse en la siguiente versión nominal. (Jacobson & Rumbaugh, 2000)

2.3.7 Flujo de Trabajo del Proceso Unificado de Software

2.3.7.1 . Requisitos

El esfuerzo principal en la fase de requisitos es desarrollar un modelo del sistema que se va a construir, y la utilización de los casos de uso en una forma adecuada de crear este modelo. Esto es debido a que los requisitos funcionales se estructuran de forma natural mediante casos de uso, y a que la mayoría de los otros requisitos no funcionales son

específicos de un solo caso de uso, y pueden tratarse en el contexto de ese caso de uso. Así, los requisitos de un sistema se capturan en forma de:

- ✓ Un modelo del negocio o un modelo del dominio para establecer el contexto del sistema.

- ✓ Un modelo de casos de uso que capture los requisitos funcionales, y los no funcionales que son específicos de casos de uso concretos. El modelo de casos de uso se realiza mediante una descripción general, un conjunto de diagramas, y una descripción detallada de cada caso de uso.

- ✓ Un conjunto de esbozos de interfaces de usuario y de prototipos de cada actor, que representan el diseño de las interfaces de usuario.

- ✓ Una especificación de requisitos adicionales para los requisitos que son genéricos y no específicos de un caso de uso en particular.

Estos resultados son un buen punto de partida para los siguientes flujos de trabajos: análisis, diseño, implementación y prueba. Los casos de uso dirigirán el trabajo a lo largo de estos flujos de trabajo iteración por iteración. Para cada caso de uso del modelo de casos de uso identificaremos una realización de caso de uso correspondiente en las fases de análisis y diseño y un conjunto de casos de prueba en la fase de pruebas. Por tanto, los casos de uso enlazaran directamente los diferentes flujos de trabajo. (Jacobson & Rumbaugh, 2000)

2.3.7.2 Análisis

Durante el análisis, analizamos los requisitos que se describen en la captura de requisitos, refinándolos y estructurándolos.

El resultado del flujo de trabajo del análisis es el modelo de análisis, que es un modelo del objeto conceptual que analiza los requisitos mediante su refinamiento y estructuración. El modelo de análisis incluye los siguientes elementos:

- ✓ Paquetes del análisis y paquetes de servicio, y sus dependencias y contenidos. Los paquetes del análisis pueden aislar los cambios en un proceso del negocio, el comportamiento de un actor, o en conjunto de casos de uso estrechamente relacionados. Los paquetes de servicios aislarán los cambios en determinados servicios ofrecidos por el sistema, y construyen un elemento esencial para construir pensando en la reutilización durante el análisis.

- ✓ Clases de análisis, sus responsabilidades, atributos, relaciones y requisitos especiales. Cada una de las calases de control, entidad e interfaz aislaran los cambios al comportamiento y la información que representan. Un cambio en la interfaz de usuario o en una interfaz de comunicación normalmente se ubica en una o más clases de entidad; un cambio en el control, coordinación, secuencia, transacciones y a veces en la lógica del negocio compleja, que implica a varios objetos (de interfaz y/o de entidad) normalmente se ubica en una o más clases de control.

- ✓ Realizaciones de casos de uso-análisis que describen como se refinan los casos de uso en términos de colaboraciones dentro del modelo de análisis y de sus requisitos especiales. La realización de casos de uso aislaran os cambios en los casos de uso, debido a que si cambia un caso de uso, debe cambiarse también su realización.

- ✓ La vista arquitectura del modelo de análisis, incluyendo sus elementos significativos para la arquitectura. La vista de la arquitectura aislara los cambios de la arquitectura.

- ✓ El modelo de análisis se considera la entrada fundamental para las actividades de diseño subsiguientes. Cuando utilizamos el modelo de análisis con esa intención,

conservamos en todo lo posible la estructura que define durante el diseño del sistema, mediante el tratamiento de la mayor parte de los requisitos no funcionales y otras restricciones relativas al entorno de la implementación. Más en concreto, el modelo de análisis influirá en el modelo de diseño de las siguientes maneras:

- ✓ *Los paquetes de análisis y los paquetes de servicios* tendrán una influencia fundamental en los subsistemas de diseño y en los subsistemas de servicios, respectivamente, en las capas específicas y generales de la aplicación. En muchos casos tendremos una traza uno a uno entre paquetes y los correspondientes subsistemas.

- ✓ Las clases de análisis servirán como especificaciones al diseñar las clases. Se requieren diferentes tecnologías y habilidades al diseñar clases del análisis con diferentes estereotipos: por ejemplo, el diseño de las clases de entidad normalmente requiere el uso de tecnologías de bases de datos, mientras que el diseño de clases de interfaz normalmente requiere el uso de tecnologías de interfaz de usuario. Sin embargo, las clases del análisis y sus responsabilidades, atributos y relaciones sirven como una entrada lógica par la creación de las correspondientes operaciones, atributos y relaciones de las clases de diseño. Además, la mayoría de los requisitos especiales recogidos sobre una clase del análisis serán tratados por las clases de diseño correspondientes cuando se tienen en cuenta tecnologías como las de bases de datos y de interfaces de usuarios.

- ✓ Las realizaciones de casos de uso análisis tienen dos objetivos principales. Uno es ayudar a crear especificaciones más precisas para el caso de uso. En lugar de detallar cada caso en el modelo de casos de uso con diagramas de estado o diagramas de actividad. La descripción de un caso de uso mediante una colaboración entre clases del análisis da como resultado una especificación formal completa de los requisitos del sistema. Las realizaciones de casos de uso-análisis también sirven como entrada al diseño de los casos de uso.

La vista de la arquitectura del modelo de análisis se utiliza como entrada en la creación de la vista de la arquitectura del modelo de diseño. Es muy probable que los elementos de las diferentes vistas (de los diferentes modelos) tengas trazas entre ellos. Esto es debido a que la relación de relevancia para la arquitectura tiende a fluir suavemente a lo largo de los diferentes modelos mediante diferencias de traza. (Jacobson & Rumbaugh, 2000)

2.3.7.3 Diseño

En el diseño se modela el sistema y se encuentra la forma para que soporte los requisitos que se suponen. El principal resultado del diseño es el modelo de diseño. Se esfuerza en conservar la estructura del sistema impuesta por el modelo de análisis, y que sirve como esquema para la implementación. El modelo de diseño incluye los siguientes elementos:

- ✓ Subsistemas del diseño y subsistemas de servicio y sus dependencias, interfaces y contenidos. Los subsistemas del diseño de las dos capas superiores (las capas específicas de la aplicación) se obtienen a partir de los paquetes del análisis algunas de las dependencias entre subsistemas del diseño se obtienen a partir de las correspondientes dependencias entre paquetes del análisis. Algunas de las interfaces se obtienen a partir de las clases del análisis.

- ✓ Clases del diseño, incluyendo las clases activas, y sus operaciones, atributos y *requisitos* de implementación. Algunas clases del diseño relevantes para la arquitectura se obtienen a partir de las clases del análisis relevante para la arquitectura. Algunas clases activas se obtienen a partir de clases del análisis se utilizan como especificaciones al obtener las clases de diseño.

- ✓ Realizaciones de casos de uso-diseño, que describen como se diseñan los casos de uso en términos de colaboraciones dentro del modelo de diseño. En general, al obtener las realizaciones de caso de uso-diseño utilizamos las realizaciones de caso de uso-análisis como especificaciones.

- ✓ La vista arquitectónica del modelo de diseño, incluyendo esos elementos significativos de cara a la arquitectura.

2.3.7.4 . Implementación

El resultado principal de la implementación es el modelo de la implementación, el cual incluye los siguientes elementos:

- ✓ Subsistemas de implementación y sus dependencias, interfaces y contenidos.

- ✓ Componentes, incluyendo componentes ficheros y ejecutables, y las dependencia entre ellos. Los componentes son sometidos a pruebas de unidad.

- ✓ Las vista de la arquitectura del modo de implementación, incluyendo sus elementos arquitectónicamente significativos.

La implementación también produce como resultado un refinamiento de la vista de la arquitectura de modelo de despliegue, donde los componentes ejecutables son asignados a nodos. El modelo de implementación es la entrada principal de las etapas de pruebas que siguen a la implementación, más concretamente, durante la etapa de prueba cada construcción generada durante la implementación es sometida a pruebas de implementación, y posiblemente también a pruebas del sistema. (Jacobson & Rumbaugh, 2000)

2.3.7.5 Pruebas.

El resultado principal de la prueba es el modelo de prueba, el cual describe como ha sido probado el sistema. El modelo reprueba incluye:

- ✓ Casos de prueba, que especifican que probar en sistema.

- ✓ Procedimientos *de* prueba, que especifican como realizar los casos de pruebas.

- ✓ Componentes de pruebas, que automatizan los procedimientos de pruebas.

- ✓ La prueba da también como resultado un plan de prueba, evaluaciones de las pruebas realizadas y los defectos que pueden ser pasados como entrada a flujos de trabajos anteriores, como el diseño y la implementación.

2.4 Lenguaje Unificado de Modelado

UML se define como un lenguaje que permite especificar, visualizar, y construir los artefactos de los sistemas de software. Es un sistema notacional (que entre otras cosas incluye el significado de sus notaciones) destinado a los sistemas de modelado que utilizan conceptos orientados a objetos. UML nació como una notación estándar para la construcción de modelos orientados a objetos, que ayuda a crear modelos con buen diseño.

Cuando se modela algo, se crea una simplificación de la realidad para comprender mejor el sistema que se está desarrollando. Con UML se construyen modelos a partir de bloques de construcción básica, tales como clases, interfaces, colaboraciones, componentes, nodos dependencias, generalizaciones y asociaciones.

Los diagramas son los medios para ver estos bloques en construcción. Un diagrama es una presentación grafico a de un conjunto de elementos, que la mayoría de las veces se dibuja como un grafo conexo de nodos (elementos) y arcos (relaciones). Los diagramas se usan para visualizar el sistema de diferentes perspectivas. Como ningún sistema puede ser comprendido completamente desde una única perspectiva, UML define varios diagramas que permiten centrarse en diferentes aspectos del sistema independiente.

Cuando se modelan sistemas reales, sea cual sea el dominio del problema, muchas veces se dibujan los mismos tipos de diagramas, porque representan vistas comunes. Normalmente las partes estáticas de un sistema se representaran mediante uno de los cuatro diagramas siguientes: diagrama de clases, diagrama de objetos, diagrama de componentes y diagrama de despliegue. (Jacobson & Rumbaugh, 2000)

2.4.1 Diagrama de Clases.

Un diagrama de clases representa un conjunto de clases, interfaces y colaboraciones, y las relaciones entre ellas. Los diagramas de clases son los diagramas más comunes en el modelado de sistemas orientados a objetos. Los diagramas de clases que incluyen clases activas se utilizan para cubrir la vista de procesos de estática de un sistema. (Jacobson & Rumbaugh, 2000)

2.4.2 Diagramas de Componentes.

Un diagrama de componentes muestra un conjunto de componentes y sus relaciones. Los diagramas de componentes se utilizan para describir la viste e implementación estática de un sistema. Los diagramas de componentes se relacionan con los diagramas de clases en que un componente generalmente se compone de una o más clases, interfaces o colaboraciones. (Jacobson & Rumbaugh, 2000)

2.4.3 Diagramas de Caso de Uso.

Un diagrama de caso de uso representa un conjunto de caso de uso y sus actores (un tipo especial de clases) y sus relaciones. Los diagramas de casos de uso se utilizan para describir la vista de casos de uso estática de un sistema. Los diagramas de caso de uso son especialmente importantes para organizar y modelar al comportamiento de un sistema. (Jacobson & Rumbaugh, 2000)

2.4.4 Diagramas de Despliegue.

Un diagrama de despliegue muestra un conjunto de nodos y sus relaciones. Los diagramas de despliegue se utilizan para describir la vista de despliegue estática de una arquitectura. Los diagramas de despliegue se relacionen con los diagramas de componentes en que un nodo generalmente contiene uno o mesa componentes. (Jacobson L., 2000)

2.4.5 Diagramas de Secuencia.

Un diagrama de secuencia es un diagrama de interacción que resalta la ordenación temporal de los mensajes. Un diagrama de secuencia presenta un conjunto de objetos y los mensajes enviados y recibidos por ellos. Los objetos suelen ser instancias con nombres anónimas de clases, pero también pueden representar instancias de otros elementos, tales como colaboraciones, componentes y nodos. Los diagramas de secuencia se usan para describir la vista dinámica de un sistema.

En el diagrama de secuencia no se ponen situaciones erróneas (movimientos inválidos, jaques, etc.) puesto que poner todos los detalles puede dar lugar a un diagrama que no se entiende o difícil de leer. El diagrama puede acompañarse con un texto en el que se detallen todas estas situaciones erróneas y particularidades. (Jacobson & Rumbaugh, 2000)

2.4.6 Diagramas de Colaboración.

Un diagrama de colaboración es un diagrama de interacción que resalta la organización estructural de los objetos que envían y reciben mensajes. Un diagrama de colaboración muestra un conjunto de objetos, enlaces entre esos objetos y mensajes enviados y recibidos por esos objetos. Los objetos son generalmente instancias con nombre o anónimas de clases, pero también pueden representar instancias de otros elementos, como colaboraciones, componentes y nodos. Los diagramas de colaboración se usan para describir la vista dinámica de un sistema. (Jacobson & Rumbaugh, 2000)

2.5 Software Libre

2.5.1 ¿Qué es Software Libre?

El software libre es una cuestión de libertad, no de precio. Para comprender este concepto, debemos pensar en la acepción de libre como en libertad de expresión y no como en "barra libre de cerveza".

Con software libre nos referimos a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software. Nos referimos especialmente a cuatro clases de libertad para los usuarios de software:

- ✓ **Libertad 0:** la libertad para ejecutar el programa sea cual sea nuestro propósito.
- ✓ **Libertad 1:** la libertad para estudiar el funcionamiento del programa y adaptarlo a tus necesidades – el acceso al código fuente es condición indispensable para esto.
- ✓ **Libertad 2:** la libertad para redistribuir copias y ayudar así a tu vecino.
- ✓ **Libertad 3:** la libertad para mejorar el programa y luego publicarlo para el bien de toda la comunidad – el acceso al código fuente es condición indispensable para esto.

Software libre es cualquier programa cuyos usuarios gocen de estas libertades. De modo que deberías ser libre de redistribuir copias con o sin modificaciones, de forma gratuita o cobrando por su distribución, a cualquiera y en cualquier lugar. Gozar de esta libertad significa, entre otras cosas, no tener que pedir permiso ni pagar para ello. Asimismo, deberías ser libre para introducir modificaciones y utilizarlas de forma privada, ya sea en tu trabajo o en tu tiempo libre, sin siquiera tener que mencionar su existencia. Si decidieras publicar estos cambios, no deberías estar obligado a notificarlo de ninguna forma ni a nadie en particular. La distribución de programas en formato ejecutable es necesaria para su adecuada instalación en sistemas operativos libres. No pasa nada si no se puede producir una forma ejecutable o binaria, dado que no todos los

lenguajes pueden soportarlo, pero todos debemos tener la libertad para redistribuir tales formas si se encuentra el modo de hacerlo.

Para que las libertades 1 y 3 – la libertad para hacer cambios y para publicar las versiones mejoradas – adquieran significado, debemos disponer del código fuente del programa. Por consiguiente, la accesibilidad del código fuente es una condición necesaria para el software libre. Para materializar estas libertades, éstas deberán ser irrevocables siempre que no cometamos ningún error; si el desarrollador del software pudiera revocar la licencia sin motivo, ese software dejaría de ser libre.

Sin embargo, ciertas normas sobre la distribución de software libre nos parecen aceptables siempre que no planteen un conflicto con las libertades centrales. Por ejemplo, el copyleft, grosso modo, es la norma que establece que, al redistribuir el programa, no pueden añadirse restricciones que nieguen a los demás sus libertades centrales. Esta norma no viola dichas libertades, sino que las protege. De modo que puedes pagar o no por obtener copias de software libre, pero independientemente de la manera en que las obtengas, siempre tendrás libertad para copiar, modificar e incluso vender estas copias.

- *'**Software libre**'* no significa *'no comercial'*. Un programa libre debe estar disponible para uso comercial, desarrollo comercial y distribución comercial. El desarrollo comercial del software libre ha dejado de ser inusual; el software comercial libre es muy importante.
    - Pero el software libre sin `copyleft' también existe. Creemos que hay razones importantes por las que es mejor usar 'copyleft', pero si tus programas son software libre sin ser 'copyleft', los podemos utilizar de todos modos. (Stallman, 2003)

2.6 Bases de Datos

2.6.1 Definición.

Una base de datos es un conjunto de datos relacionados entre sí. Por datos, se entiende, hechos conocidos que pueden registrarse y que tienen un significado implícito.
Una base de datos tiene las siguientes propiedades implícitas:

Una base de datos representa algún aspecto del mundo real, en ocasiones llamado mini mundo o universo de discurso. Las modificaciones del mini mundo se reflejan en la base de datos. Una base de datos es un conjunto de datos lógicamente coherente, con cierto significado inherente. Una colección aleatoria de datos no puede considerarse propiamente una base de datos.

Toda base de datos se diseña, construye y puebla con datos para un propósito específico. Está dirigida a un grupo de usuarios y tiene ciertas aplicaciones preconcebidas que interesan a dichos usuarios. En otras palabras, una base de datos tiene una fuente de la cual se derivan datos, cierto grado de interacción con los acontecimientos del mundo real y un público que está activamente interesado en el contenido de la base de datos. La generación y mantenimiento de las bases de datos pueden ser manuales o mecánicos. El catálogo en tarjetas de una biblioteca es un ejemplo de base de datos que se puede crear y mantener manualmente. Las bases de datos computarizadas se pueden crear y mantener con un grupo de programas de aplicación escritos específicamente para esa tarea o bien mediante un sistema de gestión de bases de datos (Navathe & Elmasr, 2000)

2.6.2 Sistema de Gestión de Bases de Datos (SGBD).

En ingles, Data base Management System DBMS, es un conjunto de programas que permite a los usuarios crear y mantener una base de datos. Por tanto, el SGBD es un sistema de software de propósito general que facilita el proceso de definir, construir y manipular bases de datos para diversas aplicaciones. Para definir una base de datos hay que especificar los tipos de datos, las estructuras y las restricciones de los datos que se almacenarán en ella. Construir una base de datos es el proceso de guardar los datos en algún medio de almacenamiento controlado por el SGBD. No hace falta un software de SGBD de propósito general para implementar una base de datos computarizada.

Podríamos escribir nuestro propio conjunto de programas para crear y mantener la base de datos, con lo cual estaríamos creando de hecho nuestro propio software SGBD de propósito específico. En todo caso, ya sea que utilicemos un SGBD de propósito general o no, casi siempre requerimos un software de gran capacidad para manipular la base de datos, además de la base de datos misma. (Navathe & Elmasr, 2000)

2.6.2.1 Modelos de Bases de Datos

Un modelo de datos es básicamente una "descripción" de algo conocido como contenedor de datos (algo en donde se guarda la información), así como de los métodos para almacenar y recuperar información de esos contenedores.

Los modelos de datos no son cosas físicas: son abstracciones que permiten la implementación de un sistema eficiente de base de datos; por lo general se refieren a algoritmos, y conceptos matemáticos.

Además de la clasificación por la función de las bases de datos, éstas también se pueden clasificar de acuerdo a su modelo de administración de datos. (Navathe & Elmasr, 2000)

2.6.2.2 Modelos utilizados con frecuencia en las Bases de Datos

Bases de Datos Jerárquicas.

Éstas son bases de datos que, como su nombre indica, almacenan su información en una estructura jerárquica. En este modelo los datos se organizan en una forma similar a un árbol, en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se los conoce como hojas.

Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo

crear estructuras estables y de gran rendimiento. Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos. (Navathe & Elmasr, 2000)

***Base de Datos de Red*.**

Éste es un modelo ligeramente distinto del jerárquico; su diferencia fundamental es la modificación del concepto de nodo: se permite que un mismo nodo tenga varios padres.

Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos; pero, aun así, la dificultad que significa administrar la información en una base de datos de red ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales. El registro es similar al de una entidad como las empleadas en el modelo relacional., Una estructura de base de datos de red, llamada algunas veces estructura de plex, abarca más que la estructura de árbol, porque un nodo hijo en la estructura red puede tener más de un nodo padre. En otras palabras, la restricción de que en un árbol jerárquico cada hijo puede tener sólo un padre, se hace menos severa. (Navathe & Elmasr, 2000)

***Base de Datos Relacional*.**

Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Su idea fundamental es el uso de "relaciones".

Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Esto es pensando en cada relación como si fuese una tabla que está compuesta por registros (las filas de una tabla), que representarían las tuplas, y campos (las columnas de una tabla).

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos.

La información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información.

2.6.2.3 Requerimientos de las bases de datos

El análisis de requerimientos para una base de datos incorpora las mismas tareas que el análisis de requerimientos del software. Es necesario un contacto estrecho con el cliente; es esencial la identificación de las funciones e interfaces; se requiere la especificación del flujo, estructura y asociatividad de la información y debe desarrollarse un documento formal de los requerimientos. (Navathe & Elmasr, 2000)

2.6.3 Arquitectura Cliente – Servidor.

Se usa para caracterizar un SGBD cuando la aplicación se ejecuta físicamente en una máquina, llamada cliente, y otra, el servidor, se encarga del almacenamiento y el acceso a los datos. Los proveedores ofrecen diversas combinaciones de clientes y servidores; por ejemplo, un servidor para varios clientes. (Verse & Park, 2004)

2.6.4 SQL (Structured Query Language).

El lenguaje de consulta estructurado es un sublenguaje de base de datos utilizado para la consulta, actualización y administración de bases de datos relacionales, el estándar de facto para los productos de bases de datos. (Verse & Park, 2004)

2.6.5 MySQL.

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario, es un sistema de administración relacional de bases de datos. Una base de datos

relacional archiva datos en tablas separadas en vez de colocar todos los datos en un gran archivo. Esto permite velocidad y flexibilidad. Las tablas están conectadas por relaciones definidas que hacen posible combinar datos de diferentes tablas sobre pedido. (Suehring, 2002)

MySQL es software de fuente abierta. Fuente abierta significa que es posible para cualquier persona usarlo y modificarlo. Cualquier persona puede bajar el código fuente de MySQL y usarlo sin pagar. Cualquier interesado puede estudiar el código fuente y ajustarlo a sus necesidades. MySQL usa el GPL (GNU General Public Lísense) para definir qué puede hacer y que no puede hacer con el software en diferentes situaciones. (Verse & Park, 2004)

2.7 Introducción a tecnologías WEB

2.7.1 Intranet.

Red diseñada para el procesamiento de información dentro de una compañía u organización. Entre sus usos se incluyen servicios tales como distribución de documentos, distribución de software, acceso a bases de datos y aprendizaje. Las intranets deben su nombre a que en ellas se utilizan a menudo aplicaciones asociadas a Internet, tales como páginas Web, sitios FTP, correo electrónico, grupos de noticias y listas de distribución, a las cuales únicamente se pueden tener acceso a los terminales de la propia compañía u organización. (Conallen, 2002)

2.7.2 Protocolo http

El protocolo HTTP (hypertext transfer protocol, por sus siglas en inglés) es el protocolo base de la WWW. Se trata de un protocolo simple, orientado a conexión y sin estado. Ver Figura 2.3 (Verse & Park, 2004)

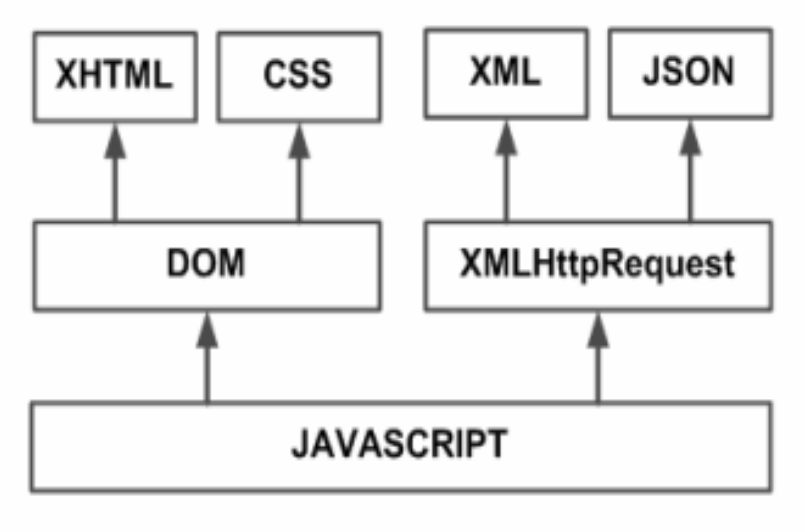


Figura 2.3: Tecnologías agrupadas bajo el concepto de AJAX.
Fuente: Eguíluz, J. 2008.

La razón de que esté orientado a conexión es que emplea para su funcionamiento un protocolo de comunicaciones (TCP, transport control protocol, por sus siglas en ingles) de modo conectado, un protocolo que establece un canal de comunicaciones de extremo a extremo (entre el cliente y el servidor) por el que pasa el flujo de bytes que constituyen los datos que hay que transferir, en contraposición a los protocolos de datagrama o no orientados a conexión que dividen los datos en pequeños paquetes (datagramas) y los envían, pudiendo llegar por vías diferentes del servidor al cliente. (Verse & Park, 2004)

El protocolo no mantiene estado, es decir, cada transferencia de datos es una conexión independiente de la anterior, sin relación alguna entre ellas, hasta el punto de que para transferir una página Web tenemos que enviar el código HTML del texto, así como las imágenes que la componen, pues en la especificación inicial de HTTP, la 1.0, se abrían y usaban tantas conexiones como componentes tenía la página, transfiriéndose por cada conexión un componente (el texto de la página o cada una de las imágenes). (Verse & Park, 2004)

2.7.3 Aplicación Cliente – Servidor.

Programa compartido en toda una red. El programa se encuentra almacenado en un servidor de red y puede ser utilizado simultáneamente por más de un cliente.

2.7.4 Servidor Web Apache

Apache es un servidor web de código libre robusto cuya implementación se realiza de forma colaborativa, con prestaciones y funcionalidades equivalentes a las de los servidores comerciales. El proyecto está dirigido y controlado por un grupo de voluntarios de todo el mundo que, usando Internet y la web para comunicarse, planifican y desarrollan el servidor y la documentación relacionada.

Estos voluntarios se conocen como el Apache Group. Además del Apache Group, cientos de personas han contribuido al proyecto con código, ideas y documentación. (Mateu, 2004)

.

2.7.5 HTML.

Es el acrónimo de Hypertext Markup Language (lenguaje de marcas de hipertexto). El lenguaje de marcas de hipertexto que se utiliza para documentos del World Wide Web. HTML es una aplicación de SGML que utiliza etiquetas para marcar los elementos, como texto y gráficos, en un documento para indicar como deberían visualizar los exploradores Web estos elementos al usuario y como deberían responder a las acciones del usuario, como la activación de un enlace presionando una tecla o haciendo clic con el ratón. HTML 2.0, definido por el Internet Engineering Task Force (IETF), incluye características del HTML común a todos los exploradores Web alrededor de 1995 y fue la primera versión de HTML ampliamente utilizado en el WWW. HTML 3.2, el último estándar propuesto, incorpora características ampliamente implementadas en 1996.

2.7.6 PHP.

PHP es un lenguaje de desarrollo web escrito por y para los desarrolladores web. PHP significa: Hypertext Preprocessor. El producto fue originalmente llamado Personal

Home Page Tools, Actualmente se encuentra en su quinta reescritura, llamado PHP5 o simplemente PHP.

Es un lenguaje de scripts del lado del servidor, que puede ser embebido en HTML o usado únicamente como binario (aunque el uso anterior es mucho más común).

PHP es un lenguaje ideal tanto para aprender a desarrollar aplicaciones web como para desarrollar aplicaciones web complejas. Añade a todo eso la ventaja de que el intérprete de PHP, los diversos módulos y gran cantidad de librerías desarrolladas para PHP son de código libre, con lo que el programador, dispone de un impresionante arsenal de herramientas libres para desarrollar aplicaciones.

PHP suele ser utilizado conjuntamente con Perl, Apache, MySQL o PostgreSQL en sistemas Linux, formando una combinación barata (todos los componentes son de código libre), potente y versátil. Tal ha sido la expansión de esta combinación que incluso ha merecido conocerse con un nombre propio LAMP (formado por las iniciales de los diversos productos). (Verse & Park, 2004)

2.7.7 Ext-js (Extend JS)

Ext JS es una librería JavaScript para construir aplicaciones (RIA), la cual se comenzó a desarrollar a principios del 2006 por Jack Slocum como un conjunto de extensiones para la librería Yahoo! User Interface (YUI), estas extensiones recibieron el nombre de yui-ext.

El otoño del 2006 la librería ganó tanta popularidad (con su versión 0.33) que cambió su nombre a Ext (con licencia BSD) como un reflejo de su madurez e independencia como framework. (http://extjs.com/learn/Ext_FAQ)

En 2007 se forma una empresa y se liberan las versiones 1, 1.1, 2.0 (actualmente se encuentra en construcción la 3.0).

Incluye:

- Alto rendimiento, widgets personalizables en entorno de usuario (UI).
- Bien diseñado y modelo de Componentes extensibles.
- Intuitivo, API fácil de utilizar.
- Licencias Comerciales y Open Source disponibles.

Ext JS soporta y es compatible con la mayoría de navegadores actuales.

- Internet Explorer 6+.
- Firefox 1.5+ (PC, Mac).
- Safari 3+.
- Opera 9+ (PC, Mac).

Ext Js trabaja conjuntamente con las librerías:

- Yahoo! UI (.12+).
- jQuery (1.1+).
- Prototype (1.5+) / Scriptaculous (1.7+).

En la Figura 2.4 y Figura 2.5 se puede observar la interacción de dichas librerías:

2.7.8 Frameworks

En el desarrollo de software, un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Un framework representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

Los frameworks son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional. Por ejemplo, un equipo que usa Apache Struts para

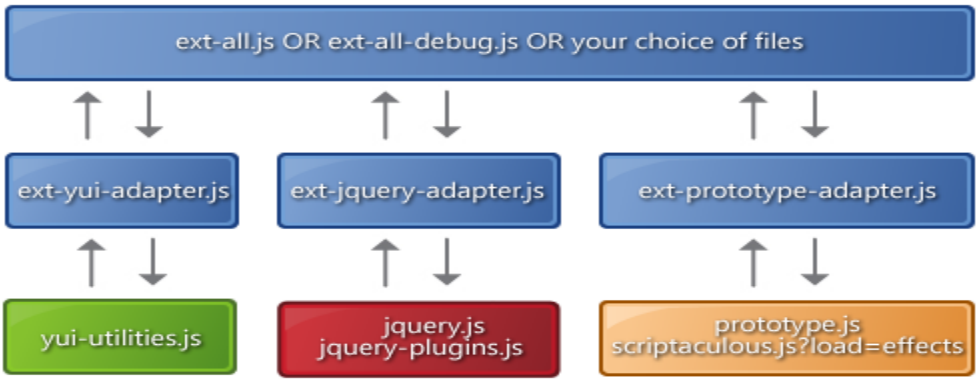


Figura 2.4: ExtJS 1.0.1.a y las bases de la relaciones de la librería.
Fuente: http://extjs.com/learn/Ext_Getting_Started , 2008

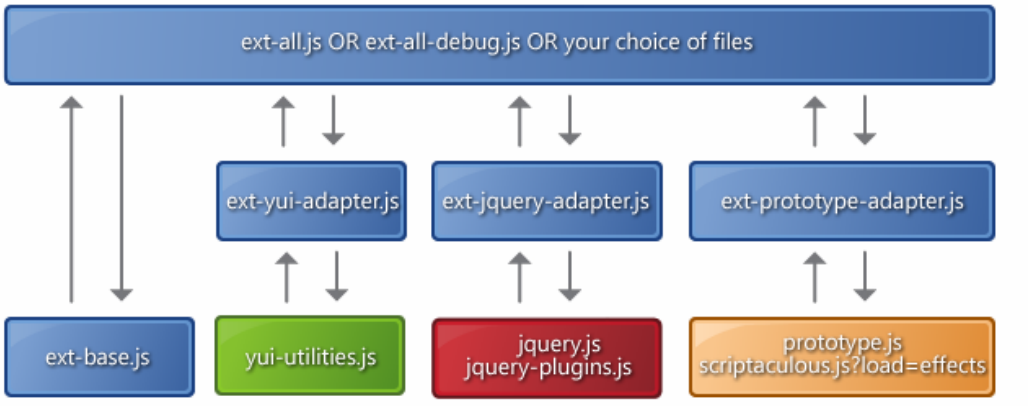


Figura 2.5 : ExtJS 1.1 y la base de relaciones de la librería.
Fuente: http://extjs.com/learn/Ext_Getting_Started , 2008.

Desarrollar un sitio web de un banco puede enfocarse en cómo los retiros de ahorros van a funcionar en lugar de preocuparse de cómo se controla la navegación entre las páginas en una forma libre de errores. Sin embargo, hay quejas comunes acerca de que el uso de frameworks añade código innecesario y que la preponderancia de frameworks competitivos y complementarios significa que el tiempo que se pasaba programando y diseñando ahora se gasta en aprender a usar frameworks.

Fuera de las aplicaciones en la informática, un framework puede ser considerado como el conjunto de procesos y tecnologías usados para resolver un problema complejo. Es el esqueleto sobre el cual varios objetos son integrados para una solución dada. (http://es.wikipedia.org/wiki/Framework)

2.8 WebML (Web Modeling Language)

2.8.1 Introducción al Desarrollo de Aplicaciones Web.

El desarrollo de una aplicación web de manejo intenso de datos es una actividad multidisciplinaria, que requiere de una variedad de habilidades, necesariamente para direccionar cada tarea, como el diseño de estructuras de datos para almacenar contenido, la concepción de interfaces de hipertexto para la exploración de información y administración de contenido, la creación de estilos de presentación efectivos, el ensamble de arquitecturas robustas y de alto desempeño. El desarrollo y mantenimiento de aplicaciones web de manejo intenso de datos requiere todas las herramientas y técnicas de ingeniería de software, incluyendo el proceso de desarrollo de software bien organizado, conceptos de diseño y notaciones apropiadas, y guías sobre cómo conducir las actividades.

La mezcla propuesta une los ingredientes tradicionales conocidos por los desarrolladores, como el diseño conceptual de datos con el modelo Entidad – Relación y especificación de casos de uso con UML, con nuevos conceptos y métodos para el diseño de hipertextos, los cuales se centralizan en el desarrollo web. Sin embargo, el valor de la propuesta no está en los ingredientes individuales, sino en la definición de

una estructura sistemática en la cual las actividades de desarrollo de aplicaciones web pueden ser organizadas de acuerdo a los principios fundamentales de desarrollo de software, y todas las tareas, incluyendo las más “web-céntricas”, encontrar el soporte adecuado en los conceptos, notaciones y técnicas apropiadas. (Mateu, 2004)

2.8.2 Modelados de Páginas Web

En la actualidad el diseño del manejo intensivo de datos que soporta un sitio Web es fundado sobre metodologías adquiridas de diferentes sectores, entre los cuales está la ingeniería de software y las bases de datos, es por ello que la falta de un modelo que permita controlar al desarrollador aspectos como: la igualdad en la estructura de las bases de datos para la navegación y la cantidad de códigos manuscritos se traducen en grandes esfuerzos hasta para la realización de un prototipo.

Algunos factores incrementan la complejidad de los sitios Web actuales, entre ellos esta: los múltiples dispositivos de salida para la información. Estos factores o requerimientos impactan sobre el costo de desarrollo de los sitios Web y la evolución de los sitios ya creados.

Por todo esto se crea un lenguaje para modelar la información a manejar durante el desarrollo de un sitio Web permitiendo así reducir el trabajo de los diseñadores gráficos, incrementar los niveles de abstracción, hacer un mejor uso de las destrezas disponibles para el análisis y diseño de alto nivel a pesar de la inmensa pérdida de tiempo en la codificación de las páginas ASP o PHP, las técnicas a implementar pueden enfocarse en el análisis de la ejecución y optimización a conseguir y por último los creativos del sitio pueden enfocarse en la “creación”. (Mateu, 2004)

Entre las ventajas del modelado de las aplicaciones Web se encuentran:

- Puede reducir esfuerzos en el desarrollo (costo y tiempo).
- Permite un proceso de desarrollo más estructurado.

- Produce resultados finales más coherentes y utilizables.
- Los modelos de diseño son siempre actualizados y auto-documentados.
- El desarrollo del prototipo puede ser inmediatamente alcanzado.

2.8.3 Modelo Hipertexto.

La meta del modelado de hipertexto es especificar la organización de las interfaces de la aplicación web. Para ser efectiva, tal especificación debe ser capaz de transmitir en una manera simple e intuitiva tales aspectos de la división lógica de la aplicación en módulos de alto nivel, cada uno representando un conjunto de funciones coherentes apuntando a una clase especifica de usuarios, la partición de los módulos de alto nivel en su módulos, para una mejor organización de grandes aplicaciones, y la actual topología de hipertexto de cada módulo, en términos de páginas, hechas de elementos de contenido, y enlazadas a apoyar la navegación de los usuarios y la interacción. El modelo de hipertexto debería estar al nivel correcto de abstracción; la especificación del hipertexto debe ser mantenida a un nivel conceptual, el cual significa que no debe comprometer mucho el diseño y la implementación de los detalles, tales como la distribución actual de la funcionalidad entre las varias capas de la aplicación web.

Diferente del modelado de datos, el cual es una actividad muy consolidada, el modelado de hipertexto es una disciplina más nueva, aun con falta de una base de conceptos bien establecida, notaciones, y métodos de diseño. WebML, provee las primitivas para el modelado de hipertexto, el cual permite al programador a expandir el esquema de datos de la aplicación con la especificación de los hipertextos usados para publicar y manipular datos. Los ingredientes clave del WebML son **páginas**, **unidades** y **enlaces** organizados en construcciones modularizadas llamadas **áreas** y **vistas** del sitio.

Unidades son las piezas atómicas de contenido publicable, ellos ofrecen formas alternativas de arreglar el contenido dinámicamente extraído de las entidades y relaciones del esquema de datos, y también permite la especificación de formularios que aceptan entradas de datos del usuario. Las unidades son los bloques de construcción de

las páginas, que son la interfaz actual de los elementos presentados al usuario. Las páginas son típicamente construidas ensamblando varias unidades de muchos tipos, para mantener el efecto de comunicación deseado.

Un conjunto de páginas puede ser agrupado en una vista de sitio, la que representa un conjunto bien definido de requisitos, por ejemplo, las necesidades de un grupo específico de usuarios. Algunas propiedades de las páginas y las áreas, como el inicio, predeterminado y puntos de referencia, permiten al diseñador ajustar el nivel de visibilidad de esas construcciones dentro de la estructura jerárquica de la vista del sitio. (Stallman, 2003)

2.8.4 Modelo de Presentación

Define como lucirá la vista del sitio. WebML incluye un modelo simple de presentación que permite colocar contenidos dinámicos en la página además de aplicar estilos distintos para cada uno.

Todas las metodologías propuestas para las aplicaciones Web desde mediados de los años noventa presentan su propia notación para casi todos sus diagramas. Una recopilación excelente de varios de estos métodos fue presentada en el primer taller internacional sobre tecnologías de software orientadas a Web. En ella se describen cada una de estas propuestas en base a un mismo caso de estudio. Estas metodologías, y otras muchas que se han propuesto, contribuyen con ideas importantes para el diseño de software orientado a Web. (Stallman, 2003)

A continuación la Figura 2.6 Muestra la estructura de un sitio Web modelado haciendo uso de WebML.

Una observación muy importante es el hecho de que WebML no es el mejor enfoque para sitios Web estáticos o pequeños. ((Stallman, 2003)

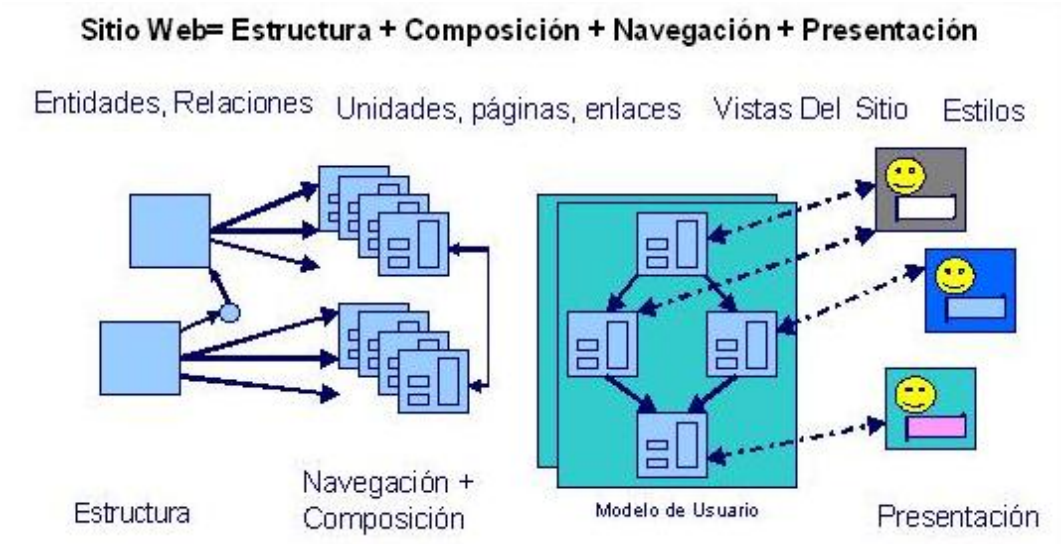


Figura 2.6: Sitio Web haciendo uso de los conceptos WebML.
Fuente: http://www.webml.org/webml/upload/webml_training1_introduction.pdf

2.8.5 Estereotipos WebML.

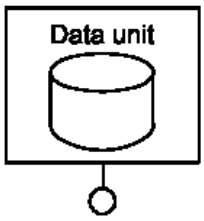
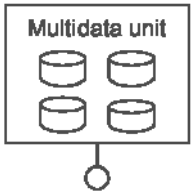
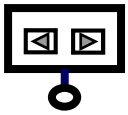
La herramienta de modelado WebML ofrece un amplio conjunto de notaciones y estereotipos que permiten representar la definición, estructuración e interpretación de aplicaciones web. En la Tabla 2.1 que se muestra a continuación, se describe cada estereotipo.

2.8.5.1 WebML Vs. UML

Desde el punto de vista del desarrollo de proyectos, existen claras diferencias entre una aplicación Web y una tradicional. (Ver:
Tabla 2.2)

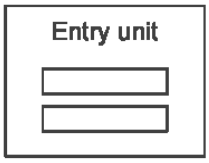
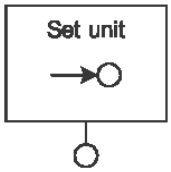
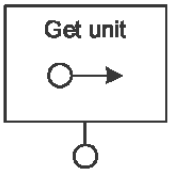
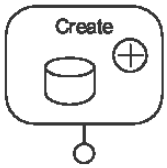
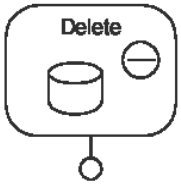
Las diferencias entre estos tipos de proyectos son de vital importancia en la selección de los perfiles del equipo de trabajo, en la estimación del tiempo de desarrollo y de costo. Es muy difícil, o casi imposible, utilizar las mismas métricas en ambos tipos de proyectos.

Tabla 2.1 : Resumen de Estereotipos de WebML para Aplicaciones Web (1 de 4)

Elemento WebML	Descripción	Propiedades
	Publica un solo objeto de una entidad dada	» Nombre » Entidad de Origen » Selector (opcional) » Atributos incluidos
	Presenta múltiples objetos de una entidad juntos, repitiendo la presentación de muchas unidades de dato	» Nombre » Entidad de Origen » Selector (opcional) » Atributos incluidos » Cláusula de orden (opcional)
	Presentan múltiples objetos de una entidad como una lista	» Nombre » Entidad de Origen » Selector (opcional) » Atributos incluidos » Cláusula de orden (opcional)
	Una variante del anterior, donde cada elemento de la lista está asociado con un "checkbox" permitiendo al usuario seleccionar múltiples objetos	» Nombre » Entidad de Origen » Selector (opcional) » Atributos incluidos » Cláusula de orden (opcional)
Scroller-Unit **(Unidad de desplazamiento)** 	Una unidad de desplazamiento provee comandos para desplazar los objetos en un escenario.	▪ Nombre. ▪ Entidad Fuente. ▪ Selector (opcional). ▪ Bloque de factores. ▪ Cláusula de Orden (opcional).

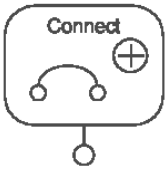
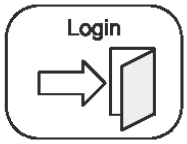
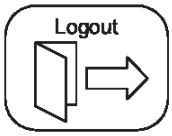
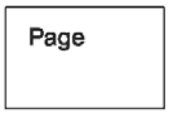
Fuente: **http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf**

Tabla 2.1. Resumen de Estereotipos de WebML para Aplicaciones Web (2 de 4)

Elemento WebML	Descripción	Propiedades
	Soporta entrada de datos basada en formulario	» Nombre » Por cada campo: » Nombre » Tipo » Valor inicial » Modificabilidad » Valor de predicado
Global Parameter	Guarda información disponible a múltiples páginas	» Nombre » Tipo » Valor por defecto
	Asigna un valor a un parámetro global	» Parámetro global
	Retira el valor de un parámetro global	» Parámetro global
	Habilita la creación de una nueva instancia de una entidad	» Nombre » Entidad de origen » Conjunto de agnaciones de valores
	Elimina uno o más objetos de una entidad dada	» Nombre » Entidad de origen » Selector

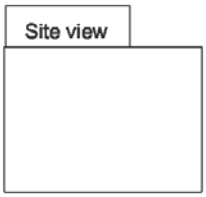
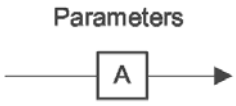
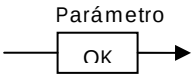
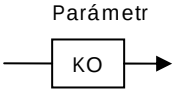
Fuente: http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf

Tabla 2.1. Resumen de Estereotipos de WebML para Aplicaciones Web (3 de4)

Elemento WebML	Descripción	Propiedades
	Crea nueva instancia de una relación	» Nombre » Rol de la relación » Selector de la entidad de origen » Selector de la entidad objetivo
	Verifica la identidad de un usuario de acceda al sitio	» Nombre de Usuario » Contraseña
	Lleva al usuario a la página principal sin control de acceso	» Ninguno
	Provee la capacidad de enviar mensajes de email	» Remitente » Recipiente » Asunto » Cuerpo » Adjuntos
	Define una operación genérica: los parámetros de entrada y salida deben estar definido por el diseñador	» Definidos por el diseñador
	Representa la interfaz actual navegada por el usuario, contiene unidades y/o subpáginas	» Nombre » Punto de referencia » Contenido: unidades o subpáginas
	Es un contenedor de páginas o, recursivamente, otras subareas, las cuales pueden ser usadas para darle una organización jerárquica al hipertexto	» Nombre » Punto de referencia » Contenido: páginas, subareas y página por defecto o subpágina

Fuente: http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf

Tabla 2.1. Resumen de Estereotipos de WebML para Aplicaciones Web (4 de 4)

Elemento WebML	Descripción	Propiedades
	Representa un hipertexto	» Nombre » Contenido: páginas, áreas y página inicial
Link 	Es una conexión orientada entre dos unidades o páginas. Abstrae el concepto de ancla y permite portar información (por medio de parámetros entre unidades) Pueden ser definidos como:	Enlaces normales, automáticos y de transporte » Nombre » Elemento origen » Elemento destino » Tipo de enlace » Parámetros de enlace
-Automático  -Transporte  **- OK Link (enlace)**  **- KO Link (enlace)** 	-Automático: son navegados sin la intervención del usuario. Transporte: permiten el paso de un parámetro Transporte: no funcionan como un ancla, pero son capaces de pasar parámetros. Enlaces en los que existen operaciones distinguidas: Enlaces OK: se ejecutan en caso de que la operación haya sido exitosa. Enlaces KO: se ejecutan en caso de que ocurra una falla.	Parámetros de enlace » Nombre » Valor origen Parámetros de enlace: ▪ Nombre. ▪ Valor Fuente. Enlaces OK/KO: ▪ Nombre. ▪ Elemento Fuente (unidad de operación). ▪ Elemento destino. Parámetros de enlace

Fuente: http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf

Un enfoque que parece razonable, desde el punto de vista de nuevas tendencias de desarrollo, parece ser la separación de un proyecto Web en dos sub-proyectos: uno referido a la funcionalidad de la aplicación y otro referido al diseño gráfico y al contenido. (Stefano & Fraternali,, 2003)

El primer sub-proyecto se puede atacar utilizando la experiencia y metodología del desarrollo tradicional de aplicaciones (UML). El segundo proyecto, enfocado en la interfaz de la aplicación (diseño, gráfico y contenido), debe ser realizado utilizando paradigmas y metodologías no tradicionales de la ingeniería de software (WebML). El tipo de personal, así como la estimación de costos y tiempo, varía para cada sub-proyecto. (Stefano & Fraternali,, 2003)

UML es aceptado como lenguaje estándar de modelado para sistemas de software y por consiguiente como mejor opción para el modelado de diseños de aplicaciones Web. Sin embargo, el gran consumo de tiempo en el proceso de modelado, unido a la gran experiencia que requiere el profesional en esta área junto a la necesidad de mostrar una visión detallada para páginas Web más complejas, despertó el interés de los desarrolladores en crear e implementar nuevas opciones que permitan un modelado igualmente eficiente y eficaz como el que brinda UML, pero de una manera más fácil y rápida, es por ello que surge el WebML como una herramienta especializada en el modelado de aplicaciones Web. (Stefano & Fraternali,, 2003)

WebML es un lenguaje completamente nuevo, más fácil de comprender que UML y por lo tanto más sencillo de implementar, consta de solamente cuatro modelos mientras UML cuenta con doce. WebML fue desarrollado específicamente para el modelado de aplicaciones Web al contrario que UML, el cual se desarrolló para ser utilizado en el más amplio rango de aplicaciones de acuerdo con las Especificaciones de Lenguaje Unificado de Modelado de OMG (de sus siglas en inglés, Grupo de Gestión de Objetos, consorcio dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos). (Stefano & Fraternali,, 2003)

Al realizar una comparación entre ambos lenguajes se determinó que el modelo estructural de WebML es similar y compatible al modelo conceptual y al diagrama de clases de UML, esto es un aspecto positivo, ya que si se está familiarizado con la notación de UML es más natural comprender cómo trabaja la notación de WebML. Igualmente se concluyó que los modelos más importantes de WebML son el Modelo de Datos y el Modelo de Hipertexto; este último permite describir la estructura y diseño de una página Web mejor que cualquier modelo del lenguaje UML, mostrando de cuales páginas consta la aplicación, qué contienen y cómo están vinculadas entre sí. (Stefano & Fraternali,, 2003)

Otra característica resaltante en el modelo de hipertexto es que muestra cuándo los datos son enviados entre páginas facilitando la visualización de su contenido. Dichas características dan una visión general de la aplicación Web, logrando una mayor eficiencia en el trabajo de los desarrolladores pues cuentan con una buena estructura a seguir. WebML proporciona habilidad de dirigir aplicaciones Web cuando crecen en tamaño y complejidad. (Stefano & Fraternali,, 2003)

Sin embargo, UML tiene una característica que lo ha mantenido vigente en la actualidad, esta es la capacidad de hacer un modelado de la aplicación Web desde la perspectiva de los usuarios, mediante diagramas de casos de uso. Estos son muy útiles para indicar la interacción entre el usuario y el sistema, por lo tanto, juegan un papel muy importante cuando se debe explicar al cliente cómo trabaja la aplicación Web deseada. (Stefano & Fraternali,, 2003)

En un principio WebML carecía de esa capacidad, actualmente WebML adopto los diagramas de casos de uso UML como parte principal y primordial para lograr una correcta recolección y definición de los requisitos y requerimientos necesarios para el desarrollo de la aplicación, entre los cuales se encuentran: la identificación de los usuarios, los diferentes niveles de acceso con los que contarán y los tipos de

mantenimiento a realizarse a la aplicación para lograr su correcto funcionamiento. (Stefano & Fraternali,, 2003)

Finalmente otra de las ventajas de WebML está en el modelo de hipertexto el cual da una visión detallada de la estructura y diseño de la(s) aplicación(es) Web a desarrollar, por ello y por lo anteriormente expuesto, WebML se a convertido en una de las mejores opciones de modelado para una empresa de desarrollo de software ya que al lograr una buena perspectiva general de la(s) aplicación(es) Web a desarrollar se obtiene una excelente comunicación con el clientes. A continuación se muestra una tabla comparativa entre un desarrollo tradicional y un desarrollo Web para una aplicación. (Stefano & Fraternali,, 2003)

2.8.6 Odontología

La Odontología es una rama que se deriva de la medicina, que se encarga de la prevención, diagnóstico, pronóstico y tratamiento de patologías bucales. Además el profesional de odontología puede detectar también patologías no bucales. (Jaime, 1990)

2.8.6.1 Importancia de la Odontología

La Odontología está dirigida a resolver los problemas del componente bucal de salud de la población, abarcando desde la prevención hasta la rehabilitación, ya sea en forma individual o colectiva, en el ámbito regional o nacional.

La crisis económica, que se ha traducido en el empobrecimiento del 80% la población, ha hecho que los problemas de salud bucal se incrementen y los afectados tengan cada vez menos acceso a los entes que prestan servido en el ámbito público y mucho menos a las clínicas privadas. Esto obliga a que los odontólogos al asumir un rol de promotores sociales, se conviertan en verdaderos agentes de cambio. (Jaime, 1990)

Tabla 2.2: Desarrollo Web Vs. Desarrollo Tradicional (1/2).

Características	Desarrollo Tradicional	Desarrollo Web
Objetivo Primario	Productos de calidad (mín. costo)	Productos de calidad, al mercado lo más rápido posible.
Tamaño típico del producto	Mediano a grande (equipos de cientos de miembros).	Pequeños (equipos de tres a cinco miembros).
Tiempo de desarrollo	De diez a dieciocho meses	De tres a seis meses.
Enfoque de desarrollo	Clásico basado en requisitos, entregas incrementales, casos de uso y documentación.	Desarrollo rápido de aplicaciones (RAD), agrupar bloques de construcción, prototipos, RUP.
Tecnologías de Ingeniería usadas	Orientación a objetos, lenguajes modernos, etc.	Métodos basados en componentes, lenguajes de cuarta y quinta generación, visualización, etc.
Procesos	Basados en CMM (Modelo de Madurez para la Capacidad de desarrollo de software).	AD HOC (específicamente para cada proceso).
Desarrollo de productos	Sistemas basados en código, re-uso, muchas interfaces externas, algunas aplicaciones complejas.	Sistema basado en objetos, componentes reutilizables, pocas interfaces externas, aplicaciones relativamente simples.

Fuente: **Reifer, D. 2000.**

Tabla 2.2: Desarrollo Web Vs. Desarrollo Tradicional (2/2).

Características	Desarrollo Tradicional	Desarrollo Web
Personal involucrado	Ingenieros de software profesionales, con cinco o más años de experiencia en al menos dos dominios de aplicaciones.	Diseñadores gráficos, ingenieros con poca experiencia (dos o más años), ingenieros recién graduados
Tecnologías de estimación	Uso de datos históricos, modelos basados en puntos por función, Estructura de Composición de Trabajos para proyectos pequeños.	Uso de la actual experiencia, diseño ajustable basado en recursos disponibles, WBS para proyectos pequeños.

Fuente: **Reifer, D. 2000.**

2.8.6.2 Historias clínicas

La historia clínica expediente clínico es un documento médico legal, que surge del contacto entre el médico y el paciente. En ella se recoge la información necesaria para la correcta atención de los pacientes. La historia clínica es un documento válido, desde el punto de vista clínico y legal, que recoge información de tipo asistencial, preventivo y social. El método clínico es una gran ayuda para tener un perfil más amplio de la salud actual del paciente y sabiendo descartar enfermedades.

Las historias clínicas odontológicas deben contemplar tres (3) aspectos fundamentales los cuales son los siguientes: (Jaime, 1990)

- **Anamnesis**

Es un examen subjetivo en el cual se recopilan los datos personales, motivo de consulta, enfermedad actual y antecedentes familiares.

- **Examen Físico**

Es un examen objetivo, en donde se realizan los exámenes de palpación (examen donde se detectan algunas tumoraciones o dolencias de ciertas zonas). Se realiza la

auscultación (se realiza la evaluación a nivel de tórax y espalda para detectar problemas respiratorios). (Jaime, 1990)

- **Exámenes Complementarios**

Son los exámenes de laboratorio y Rayos X.

2.8.6.3 Tratamientos Odontológicos

- ✓ **Tratamientos Operatorios**

En estos casos se realizan restauraciones dentales con amalgamas y resinas sin incluir cirugía o tratamientos protésicos.

Las preparaciones dentarias se clasifican en cavidades de tipo:

- ✓ **Clase 1:** Son caries de fosas y fisuras.
- ✓ **Clase II: Incluyen** la cara octusal del diente y la aproximal.
- ✓ **Clase III:** Abarca las caras proximales de los dientes anteriores (incisivos) sin abarcar el ángulo incisal.
- ✓ **Clase IV:** Es igual a la clase III, pero este caso si abarca el ángulo incisal.
- ✓ **Clase V:** Son caries que se ubican en el borde cervical (encía) de cualquier diente.
- ✓ **Clase VI:** Son las cavidades (caries) más complejas que abarcan del diente demesial al distal. (Jaime, 1990)

2.8.7 Tipos de materiales.

2.8.7.1 Materiales Odontológicos

Los materiales utilizados en las actividades odontológicas, se dividen en materiales de obturación y materiales de impresión. (Jaime, 1990)

2.8.7.2 Materiales de Obturación

Son todos aquellos materiales utilizados para el sellado de las preparaciones dentales, existen dos tipos que son temporales y permanentes. (Jaime, 1990)

2.8.7.3 . Materiales Permanentes

- Amalgama

Es el material más utilizado por muchos años ya que posee las mejores propiedades físicas, capaces de soportar las fuerzas masticatorias por mucho más tiempo. Esta es una aleación compuesta por materiales como: zinc. Níquel, plata, estaño y cobre, combinados con mercurio. Algunas de estas pueden estar ausentes todo depende de la casa fabricante. Una de sus mayores desventajas es el factor estético. (Jaime, 1990)

- Resinas (compósites, se adhieren químicamente al diente) auto curadas y fotocuradas.

Son materiales utilizados en forma de pasta que se activan y endurecen mediante la luz ultravioleta de una pistola especialmente diseñada para esta finalidad. Esta es altamente estética y más resistente que las de autopolimerización. La resina de autopolimerización se usa en forma de dos (2) pastas que se mezclan y no se usa pistola de luz ultravioleta ya que esta endurece por sí misma después de unos minutos. No es tan estética ya que utiliza un solo tipo de color (no se adhiere químicamente al diente). (Jaime, 1990)

2.8.7.4 . Materiales Temporales

- Óxido de Zinc – Eugenol

Es una mezcla de polvo (óxido de zinc) y eugenol (líquido a base de opio) que mezclados crean una pasta blanquecina que es insoluble en la cavidad bucal, y es utilizado como un material de obturación temporáneo. (Jaime, O., 1990)

- Resinas Acrílicas

Es una resina de auto polimerización que se utiliza para la confección de coronas acrílicas provisionales, puentes fijos, y prótesis dentales en ancianos. De ninguna manera se puede utilizar o trabajar dentro de la cavidad bucal con este material de manera exagerada ya que puede ser irritante para los tejidos por su alto compuesto químico. (Jaime, 1990)

2.8.7.5 Materiales de Impresión

Los materiales de impresión son utilizados en odontología para la reproducción en detalle de las estructuras que se encuentran en cavidad oral. Entre los de uso más frecuente se encuentran los yesos, los hidrocoloides reversibles e irreversibles, el poliéter, los mercaptanos, los compuestos de modelar, la pasta cinquenólica y las siliconas de adición y de condensación. Cada uno cuenta con características diferentes de las cuales depende su uso.(http://www.axon.es/paginas/pdf/77368_1.pdf)

- Pasta de Óxido de Zinc – Eugenol

Registra con exactitud la superficie a impresionar no necesita un medio separador, no absorbe secreciones mucosas las cuales pueden causar defectos en la parte palatina de la impresión. (Jaime, 1990)

- Hidrocálidas Irreversibles

Registran con exactitud los detalles si son controlados y almacenados adecuadamente, no requieren el uso de un medio separador no absorben las secreciones mucosas, los modelos deben ser vaciados inmediatamente o de lo contrario se distorsionará el registro.

Siliconas

Se suministran en tres consistencias: liviana, regular y pesada. Para prótesis totales se utilizará la consistencia liviana. Registran con exactitud la superficie a impresionar, la resistencia a la rasgadura es menor que la de los polisofuros de caucho. Necesita adhesivo para la cubeta. ((Jaime, 1990)

Take 1

Nuevo material de impresiones, Polivinilo de Adicción que es fácil de usar y que, junto a sus propiedades hidrofóbicas permite reproducir con precisión los detalles bajo cualquier condición. Take 1 tiene un sistema de relleno bimodal, permitiéndole estirarse alrededor de los socavones del diente sin deformarse o desgarrarse y sus colores brillantes dan impresiones de fácil lectura. Take 1 está disponible en fraguado regular o rápido, dándole el tiempo necesario para los casos múltiples o individuales de trabajo rápido

Si se decide emplearlas, tenga en cuenta las recomendaciones de la casa fabricante y no olvide los hallazgos que señala la literatura:

- Manipule el material con las manos muy limpias y no permita que se contamine con látex o talcos de los guantes.
- Asegúrese que realiza una apropiada limpieza de las superficies a impresionar para evitar contaminación con saliva y restos de cementos o acrílicos, que influyen negativamente en la adhesión entre las dos capas de silicona.
- Siempre utilice el agente de adhesión ya sea que utilice cubeta perforada o acrílica. Asegúrese que la cubeta tenga adicionalmente retención mecánica.
- Si va a utilizar el material para toma de impresiones en desdentados, evalúe bien el caso y la calidad de los tejidos para saber si empleará alta y baja viscosidad o mezclas entre ellas.

2.9 Medicina General.

La Medicina General es el primer paso dentro del sistema sanitario, y es imprescindible para la detección y tratamiento de las enfermedades, pero sobre todo para la prevención de las mismas. La detección, valoración y tratamiento de enfermedades agudas, es decir enfermedades puntuales, son de vital importancia para los casos en que se deba derivar a un especialista en el tema.

Los médicos generales tienen la responsabilidad y el imperativo ético y moral de ofrecer a la sociedad sus conocimientos y la tecnología a su alcance para contribuir al bienestar de la población y a la superación de su calidad de vida. Conscientes de ello, se ha venido estudiando un aspecto delicado pero de impostergable solución, el de la certificación de la calidad del ejercicio de los médicos que ejercen la medicina general y de la recertificación de los mismos, toda vez que son ellos en gran parte los responsables de la atención de primer contacto con los enfermos de este país. (Pietrek, 1993)

CAPÍTULO III

FASE DE INICIO

3.1 INTRODUCCIÓN

El Proceso Unificado de Desarrollo de Software es la metodología que regirá el marco de trabajo para el desarrollo del proyecto y definirá las actividades que se realizarán. Esta metodología define cuatro etapas o fases de desarrollo, fase de inicio, fase de elaboración, fase de construcción y fase de transición; actualmente nos encontramos en la fase de inicio. El Proceso Unificado recomienda el uso del lenguaje de modelado UML para representar el sistema. Para este proyecto se usará, adicionalmente, el lenguaje de modelado WebML, una herramienta novedosa que ofrece modelos para representar sistemas basados en web. Cabe destacar que se tiene planeado realizar la implantación del software a través de una aplicación web.

La mayoría de los proyectos requieren de una etapa inicial breve en la que se estudian los siguientes tipos de preguntas:

- ¿Cuál es la visión y el análisis del negocio para este proyecto?
- ¿Es viable?
- ¿Comprar y/o conseguir?
- Estimación aproximada del costo.
- ¿Deberíamos abordarlo o no seguir?

Para definir la visión y obtener una estimación del orden de magnitud es necesario llevar a cabo alguna exploración de los requisitos. Sin embargo, el objetivo de la etapa de inicio no es definir todos los requisitos, o generar una estimación creíble o plan de proyecto.

.

3.2 Planificación de la Fase de Inicio

El Proceso Unificado de desarrollo de software tiene como meta asegurar la producción de software de muy alta calidad que satisfaga las necesidades de los usuarios finales, en esta fase el propósito es realizar el análisis de los requerimientos solicitados por la empresa hasta lograr justificar la realización del proyecto.

Para un correcto desarrollo del proyecto, denominado **"DESARROLLO DE UN SOFTWARE DE AUTOMATIZACIÓN PARA EL CONTROL DE LAS ACTIVIDADES DEL ÁREA DE SALUD, ADSCRITA A LA DELEGACIÓN ESTUDIANTIL, NÚCLEO ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"** es necesario seguir una serie de pautas y procedimientos, los cuales de no realizarse de manera organizada, pueden traer errores con un alto impacto en el producto final. Por estas razones se ha elaborado una planificación dividida en varias etapas basadas en el modelo del Proceso Unificado de Desarrollo de Software: Inicio, Elaboración, Construcción y Transición. En cada una de ellas se realizan los procedimientos y técnicas correspondientes de tal manera que su ejecución ofrece las herramientas necesarias para proceder con la siguiente fase.

La fase de inicio no es un estudio completo del sistema propuesto, sino que, en ella se busca el porcentaje de casos de uso necesarios para fundamentar el análisis inicial. Para realizar este estudio se siguen cuatro pasos:

- Delimitar el ámbito del sistema propuesto, es decir, definir los límites del sistema, definir los requerimientos e identificar las interfaces con sistemas relacionados que están fuera de los límites.

- Describir una propuesta de la arquitectura del sistema (en especial en aquellas partes que son nuevas, arriesgadas o difíciles).

- Identificar riesgos críticos (los que afectan la capacidad de construir el sistema) y determinar si podemos encontrar una forma de mitigarlos, quizás en una etapa posterior.

- Demostrar a usuarios o clientes potenciales que el sistema propuesto es capaz de solventar sus problemas o de mejorar sus objetivos de negocio construyendo un prototipo.

3.3 Requisitos

3.3.1 Comprensión de los Requisitos

Los requisitos son capacidades y condiciones con las cuales debe ser conforme el sistema. El primer reto del trabajo de los requisitos es encontrar, comunicar y recordar lo que se necesita realmente, de manera que tenga un significado claro. Para establecer los requisitos iniciales (no muy explícitos), se consideró necesario implementar el modelo de dominio del sistema y modelo de casos de uso para obtener el modelo conceptual del negocio y así estudiar los requerimientos fundamentales del sistema.

3.3.2 Comprender el contexto del sistema

Para el estudio del contexto del sistema es necesario comprender las relaciones entre el software que se está diseñando y el entorno externo. Comprender esto ayuda a decidir cómo suministrar la funcionalidad requerida al sistema y cómo estructurar éste para que se comunique efectivamente con su entorno.

Para comprender el contexto del sistema fue necesario conocer, estudiar y analizar las actividades relacionadas por el proceso **Área de Salud** específicamente por **Servicios Médico y Odontológicos** perteneciente a la Universidad de Oriente., núcleo de Anzoátegui, para ello fue necesario realizar lo siguiente:

- La observación es la más común de las técnicas de recolección de datos, esta se fundamenta en el reconocimiento visual de los acontecimientos o fenómenos correspondientes al sistema actual objeto de estudio. Esta técnica se utilizó para

conocer el funcionamiento de los procesos relacionados con el control de citas en el área Odontológica y Medica.

- Se realizaron entrevistas de tipo no estructuradas con el fin de conocer el funcionamiento y necesidades de los procesos de control de citas y solicitudes médicas, así como también de propuestas para la mejora de dichos procesos, todo esto con el fin de canalizar el flujo de requerimientos para el desarrollo de la aplicación.

Todo esto para identificar las posibles debilidades del sistema actual y así tratar de corregirlas en el sistema en desarrollo, para generar una aplicación que cubra con toda la información obtenida y minimizar las posibles debilidades que el sistema pudiera presentar a futuro. Una vez estructuradas las informaciones necesarias se consiguió una aproximación del contexto del sistema representado mediante del modelo de dominio, describiendo los conceptos importantes del contexto como objeto del dominio y enlazando estos objetos entre sí.

3.3.3 Modelo de Dominio del Sistema

La etapa orientada a objetos esencial del análisis o investigación es la descomposición de un dominio de interés en clases conceptuales individuales u objetos (las cosas de las que somos conscientes). Un modelo de dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. También se les domina modelos conceptuales, modelo de objetos del dominio y modelos de objetos de análisis.

El proceso unificado define un modelo de dominio como uno de los artefactos que podrían crearse en la disciplina del modelado del negocio.

Utilizando la notación UML, un modelo del dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. Pueden mostrar:

- ✓ Objetos del dominio o clases conceptuales.
- ✓ Asociaciones entre las clases conceptúales.
- ✓ Atributos de las clases conceptuales.

Esbozar una propuesta de la arquitectura que soporte el ámbito del sistema, y permita crear una arquitectura estable, describiendo la misma a través de las primeras versiones de los modelos de casos de uso, análisis y diseño.

Desarrollar el análisis inicial del Proyecto.

Mitigar los riesgos críticos, identificándolos y determinando si existe una forma de evitarlos, limitarlos, atenuarlos o controlarlos. Demostrar a los usuarios o clientes que el sistema propuesto solventará sus problemas y mejorará sus objetivos de negocio. Adaptar el Proceso Unificado al tipo de sistema en desarrollo. Completar todas las tareas en una sola iteración. El Modelo de Dominio se describe mediante diagramas de UML (especialmente los diagramas de clases). En la Figura 3.1, presenta de forma global como se desarrolla el proyecto en donde se permite establecer las relaciones, asociaciones, composiciones y herencia que existen entre cada uno de las clases anteriormente descritas, permitiendo un mejor entendimiento del mismo.

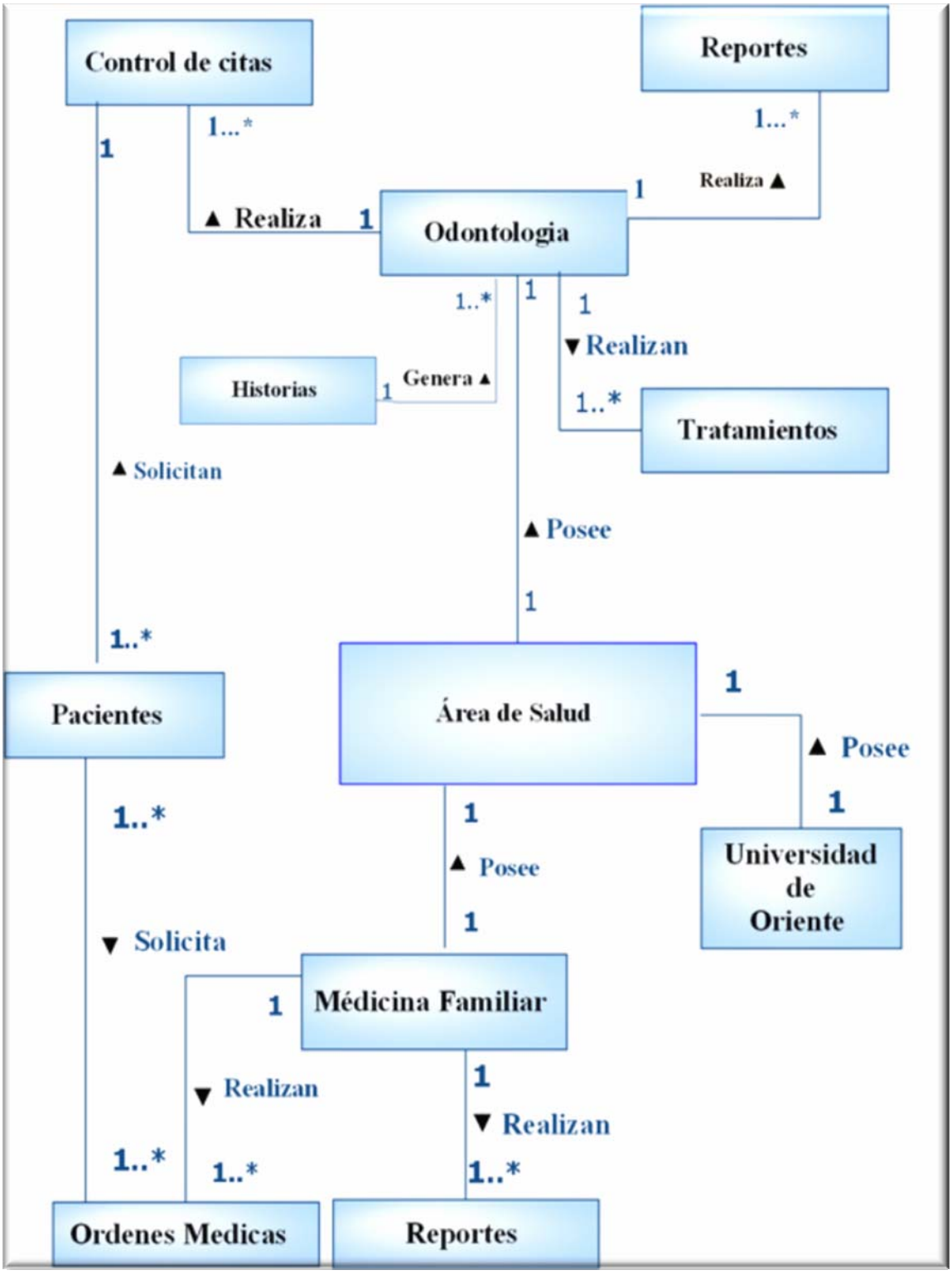


Figura 3.1: Diagrama de Clases del Dominio.
Fuente: **Elaboración Propia.**

3.3.4 Descripción del Modelo de Dominio.

La Universidad de Oriente, posee una unidad de servicios Médicos y Odontológicos, en el cual existe un empleado encargado de administrar todas las actividades que se llevan a cabo en cada una de las unidades Médicas con que cuenta la Universidad, además es el responsable de controlar las actividades los tratamientos realizados por los médicos en el periodo de clases.

3.3.4.1 Glosario de Términos

El glosario ayuda a los usuarios, desarrolladores, auditores y otros interesados a utilizar un vocabulario estándar. La terminología estandarizada es necesaria para compartir el conocimiento con diferentes diseñadores. Incluye y define la mayoría de los términos que requieren ser explicados para mejorar la comunicación y aminorar los riesgos de malos entendidos entre los desarrolladores y los clientes o usuarios.

El glosario de términos para el desarrollo del software SACMO se muestra en la Tabla 3.1

Tabla 3.1: Glosario de Términos (1/2)

Término	Descripción
Universidad	Centro de desarrollo integral del país, donde el hombre focaliza su búsqueda de conocimiento para la exploración de nuevos horizontes y poder realizar nuevas propuestas culturales y garantizar el desarrollo del país.
Unidades Medicas	Representa las áreas de odontología y medicina generar como entidad de bienestar estudiantil.
Tratamientos Odontológicos	Engloba todos los tratamientos que se les realizaran a los Pacientes en el área de odontología.

Fuente: **Elaboración Propia.**

Tabla.:3.1: Glosario de Términos(2/2)

Usuarios	Empleado de la empresa que interactúa con el sistema

	Doctores y Asistentes..
Reportes	Documento emitido por SACMO, con información pertinente a los récipes, tratamientos y constancias médicas.
Récipes Médicos.	Engloba todos los medicamentos que son recetados a los estudiantes.
Pacientes.	Engloba todos los usuarios que son los pacientes a tratar.
Ordenes de Medicina	Engloba todas las órdenes que son referenciados por la delegación de servicios estudiantiles a otros especialistas.
Control de Usuarios	Engloba los datos de los usuarios que utilizaran el sistema (SACMO).
Citas Médicas.	Engloba la disponibilidad, de atención a los pacientes por día.

Fuente: **Elaboración Propia.**

3.3.4.2 Identificación de Riesgos

Se conocen como riesgos a las variables del proyecto que ponen en peligro el éxito o no del mismo. Es entendido que en todo nuevo proyecto a emprender existen riesgos. Por tanto los riesgos influyen de terminantemente en el desarrollo de un nuevo software. Por tal motivo es necesario estudiarlos y tratarlos desde el principio, en este caso, en la fase de inicio.

Para determinar el contenido de cada iteración, el proceso unificado contempla una serie de parámetros que indican por lo menos de una forma intuitiva la correspondencia de ciertas actividades a dichas iteraciones y que permite la asignación de los casos de uso a las mismas. Se dice de forma intuitiva ya que todo depende de la naturaleza y dificultad del proyecto a desarrollar y de la comprensión del dominio del problema que se pueda tener. De cualquier forma, estos parámetros están constituidos por los riesgos.

Un riesgo es una probabilidad de que el proyecto se vea afectado en su desarrollo o posteriormente en su comportamiento, y deban ser tratados de acuerdo a su importancia y prioridad.

Entre los riesgos que se puedan presentar en un proyecto de desarrollo se encuentra el de no poder realizarlo. A este tipo de riesgos se le conoce como riesgo crítico y deben ser tratados muy temprano en el ciclo de desarrollo. Otros riesgos de menor prioridad y que afectan a ciertas funcionalidades del sistema son llamados riesgos secundarios.

Los riesgos se pueden clasificar en:

- ✓ Riesgos específicos de un producto particular: Estos riesgos están referidos a problemas que podrían surgir de las técnicas y herramientas que se piensan utilizar para abordar los requerimientos durante la implementación, o del comportamiento del sistema ante ciertas entradas o exigencias.

- ✓ Riesgo de no conseguir la arquitectura correcta: Uno de los riesgos más serios es el de no construir un sistema que pueda evolucionar suavemente por las fases siguientes o durante su tiempo de vida, es decir, el no establecer una arquitectura flexible.

- ✓ Riesgo de no conseguir los requisitos correctos: Constituye el riesgo de no construir un sistema que haga lo que los usuarios quieren realmente que haga. Este tipo de riesgo depende en gran medida del trabajo que se haga durante la captura de requisitos.

Una vez que se han identificado los riesgos, se procede a manipularlos de varias formas. Se cuenta fundamentalmente con cuatro elecciones: evitarlo, eliminarlo, atenuarlo o controlarlo. Algunos riesgos pueden o deberían evitarse, quizás mediante

una reestructuración del proyecto o un cambio en los requisitos, otros deberían limitarse, es decir, restringirse de modo que solo afecten a una parte del proyecto, otros riesgos pueden atenuarse ejercitándolos y observando si aparecen o no. Sin embargo hay riesgos que no pueden atenuarse.

Los riesgos críticos principales del proyecto, los cuales responden a las probabilidades de errores o insuficiencias que se puedan presentar en el desarrollo de cualquier sistema, tales como la inseguridad que pueda existir acerca del uso de las herramientas apropiadas para resolver el problema, las fallas en acceso a la base de datos o la conexión entre esta y el software. A continuación se presentan en la **Tabla 3.2**

Tabla 3.2. Riesgos Críticos. (1/3)

Riesgo	Falta de dominio del contexto.
Descripción	Este riesgo reside en la posibilidad de que no se pueda tener dominio completo sobre el manejo del sistema actual.
Prioridad	Crítico.
Responsable	Desarrollador.
Impacto	Obtención de los requisitos.
Contingencia	Insistir en el dominio del contexto.
Riesgo	No conseguir la arquitectura correcta.
Descripción	Es necesario conseguir una arquitectura software que permita la evolución del sistema por sus diferentes fases y durante su tiempo de vida.

Fuente: Elaboración Propia.

Tabla 3.2. Riesgos Críticos. (2/3)

Prioridad	Crítico.

Responsable	Desarrollador.
Impacto	Arquitectura del Sistema.
Contingencia	Analizar en profundidad el sistema y estudiar en detalle las fases del Proceso Unificado de Desarrollo de Software.
Riesgo	Emisión de Reportes.
Descripción	Se desea construir un sistema que sea capaz de emitir reportes impresos.
Prioridad	Secundaria.
Responsable	Desarrollador.
Impacto	Reportes del sistema.
Contingencia	Estudiar la posibilidad de conectar el sistema con aplicaciones que brinden soporte de impresión.
Riesgo	Configuración del Sistema.
Descripción	Este riesgo consiste en la administración general de los términos asociados al sistema, necesarios para el funcionamiento del mismo.
Prioridad	Crítico.
Responsable	Desarrollador.
Impacto	Todo el Sistema.

Fuente: **Elaboración Propia.**

Tabla 3.2. Riesgos Críticos. (3/3)

Riesgo	Fallas en la conexión a la base de datos.

Descripción	Este riesgo reside en la posibilidad de que se produzcan fallas al momento de codificar el acceso a la base de datos y no utilizar los componentes adecuados para ello.
Prioridad	Crítico.
Responsable	Desarrollador.
Impacto	Funcionalidad del Sistema.
Contingencia	Revisar la documentación relacionada con el acceso a base de datos a través de código y componentes.
Riesgo	Empleo de herramientas no apropiadas.
Descripción	Este riesgo reside en la dificultad que pueda involucrar el uso de un lenguaje de programación o herramientas que no brinden el soporte adecuado al desarrollo del sistema.
Responsable	Desarrollador.
Impacto	Funcionalidad del Sistema.
Contingencia	Analizar las herramientas disponibles en la empresa para verificar la vialidad de ellas en el desarrollo del software.
Riesgo	No conseguir requisitos correctos.
Descripción	Se desea construir un sistema que cubra las necesidades que exige el usuario y de la forma que éste espera. No tiene sentido desarrollar un sistema que no satisfaga los requerimientos del cliente .
Prioridad	Crítico.
Responsable	Desarrollador.
Impacto	Funcionalidad del Sistema.
Contingencia	Analizar en profundidad el sistema.

Fuente: **Elaboración Propia**

Como se puede observar, cada riesgo, está intrínsecamente asociado con una fracción de la funcionalidad del sistema.

Esta lista de riesgo será revisada nuevamente en un futuro para verificar si han sido mitigados por alguna parte en el diseño del software.

3.3.5 Requisitos Funcionales

Los requisitos funcionales especifican:

- ✓ Acciones que el sistema debe ser capaz de realizar, sin considerar restricciones físicas.
- ✓ El comportamiento de entrada / salida del sistema.

A través de entrevistas realizadas a las personas encargadas en cada uno de las unidades Medico Odontológicos de la Universidad, se logró obtener una lista de requerimientos para dar inicio al desarrollo del proyecto:

- ✓ Permitir el acceso a la base de datos a través del sistema: Inserción, modificación, eliminación y búsqueda de registros.
- ✓ Almacenar la información de los usuarios de acuerdo a los equipos asignados.
- ✓ Permitir un control y actualización en los datos de los pacientes.
- ✓ Mostrar los tratamientos realizados a los pacientes por parte de los Doctores.
- ✓ Implementar un módulo que permita generar reportes sobre una información consultada, como entrega de récipes a los pacientes.
- ✓ Mostrar la cantidad de material con que cuenta el almacén de cada clínica.
- ✓ Realizar una aplicación HTML conectada al sistema que permita al usuario obtener información general sobre el manejo del sistema.
- ✓ Realizar una interfaz gráfica sencilla y agradable que facilite el trabajo al usuario.

3.3.6 Modelo de Casos de Uso

Las especificaciones funcionales del sistema han de ser representadas en forma de casos de uso, los cuales conformarán, junto con los actores, la primera versión del modelo de casos de uso.

Describen lo que hace el sistema desde el punto de vista de un observador externo. Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede ejecutar y que produce un resultado observable de valor para un particular actor, así como el comportamiento deseado del sistema, que representan los requisitos funcionales del sistema, describiendo qué hace el sistema, no cómo lo hace.

3.3.6.1 Identificación de Actores

Un actor representa un conjunto coherente de roles que juegan los usuarios de los casos de uso al interaccionar con el sistema.

Un actor tiene las siguientes características

- ✓ Roles jugados por personas, dispositivos, u otros sistemas.
- ✓ El tiempo puede ser un actor (“procesos iniciados por el sistema”)
- ✓ No forman parte del sistema
- ✓ Un usuario puede jugar diferentes roles.
- ✓ En la realización de un caso de uso pueden intervenir diferentes actores.
- ✓ Un actor puede intervenir en varios casos de uso.
- ✓ Identificar casos de uso mediante actores y eventos externos.
- ✓ Un actor necesita el caso de uso y/o participa en él.
- ✓ Los actores pueden obtener o ingresar información al sistema

Los actores principales del sistema, identificados durante esta primera fase son los actores los mostrados en la **Tabla 3.3**

Tabla 3.3 Descripción de Actores

ACTOR	DESCRIPCIÓN

Doctor	Actor genérico que modela aquellos actores que tienen la función de administrar el Sistema. El doctor abarca el manejo de los tratamientos y la actualización de la base de datos mensual, entre otras. este actor puede ser representado por el actor asistente persona que se encarga de todo el proceso de generar citas odontológicas
MBD(Manejador de Base de Datos)	Es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.
Asistente.	Actor genérico que modela aquellos actores que tienen la función de administrar el Sistema El Asistente además de sus funciones específicas, también puede realizar las funciones que desempeña el actor Doctor.
Base de dato Sacmo	Representa donde se almacenará la información más importante para el sistema y que se despliega en la aplicación. A través de su implementación se solventan muchos problemas existentes en la empresa además de suavizar los riesgos estudiados anteriormente.

Fuente: Elaboración Propia.

3.3.6.2 Diagrama de Casos de Uso

El diagrama de casos de uso Figura 3.2 muestra la interacción de los actores con cada uno de los casos de uso del sistema.

3.3.6.3 Descripción detallada de los casos de uso

Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema como el análisis, diseño y prueba, al mostrar cómo reacciona una respuesta a eventos que se producen en el mismo

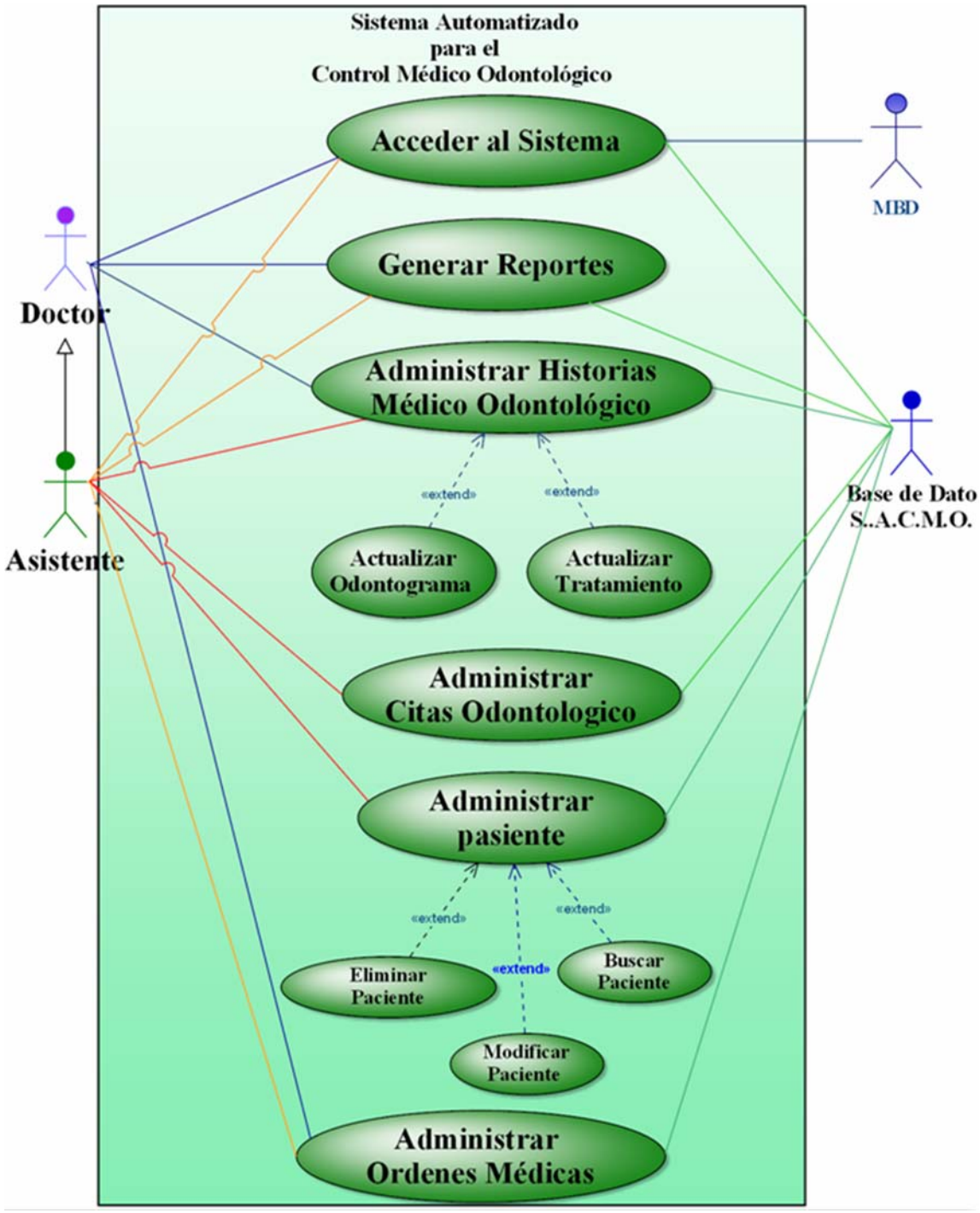


Figura 3.2: Diagrama De casos de Uso General del Sistema.
Fuente: Elaboración Propia.

.

Un diagrama que muestra la relación entre los actores y los casos de uso en un sistema. Una relación es una conexión entre los elementos del modelo, por ejemplo la relación y la generalización son relaciones.

Los casos de uso se pueden detallar más o menos dependiendo de la necesidad del problema, los flujos alternativos son los que permiten indicar qué es lo que hace el sistema en los casos menos frecuentes e inesperados. Por último, las pos condiciones son los hechos que se ha de cumplir si el flujo de eventos normal se ha ejecutado correctamente. Lo realmente útil de los casos de uso es el documento que describe el caso de uso (use case), en este documento se explica la forma de interactuar entre el sistema y el usuario. En la **Tabla 3.4**. Se muestra el flujo de estas relaciones.

Tabla 3.4: Descripción detallada de los casos de uso. (1/5)

Caso de Uso	Acceder al sistema
Actores	Doctor, Asistente, Manejador de Base de Datos.
Precondición	Datos de Doctor, Asistente.
Flujo de sucesos	1. Los actores invocan el caso de uso Acceder al sistema. 2. El Sistema carga o actualiza los datos de los actores. 3. El sistema solicita confirmación de la actividad realizada. 4. Finaliza el caso de uso.
Caminos alternativos	El Sistema repite el paso 2 tantas veces como cantidad de usuarios necesiten cargar o actualizar sus datos. El usuario puede en el paso 1,2 ó 3 cancelar la ejecución del caso de uso por lo cual puede saltar al paso 4.
Post-Condición	El caso de uso finaliza una vez que se actualiza la información en la base de datos.
Caso de Uso	Generar Reporte
Actores	Doctor, Asistente, Manejador de Base de Datos.
Precondición	Solicitud de reporte.

Fuente **Elaboración propia**

Tabla 3.4: Descripción detallada de los casos de uso.(2/5)

Flujo de sucesos	1. El Actor, invoca el caso de uso Generar Reporte. 2. El Actor elige la opción según la solicitud, datos de reporte. 3. El sistema muestra la pantalla relacionada con la opción seleccionada en el paso 2. 4. El Actor ejecuta la actividad relacionada con la Solicitud de datos El sistema solicita confirmación de la actividad realizada. 5. Finaliza el caso de uso.
Caminos alternativos	El Actor puede ejecutar los paso 2 ó 4 tantas veces como solicitudes le sean enviadas. El Actor puede en el paso 2 ó 4 cancelar la ejecución del caso de uso por lo cual puede saltar al paso 6.
Post-Condición	El caso de uso finaliza una vez que se actualiza la información en la base de datos.
Caso de Uso	Administrar Historia Médico Odontológico
Actores	Doctor, Asistente, Manejador de Base de Datos.
Precondición	Solicitud Historia Medico Odontológico.
Flujo de sucesos	1. El Actor invoca el caso de uso Historia Medico Odontológico 2. El sistema muestra la pantalla de Ingresar datos. 3. El Actor realiza la actividad relacionada con la solicitud. 4. El sistema solicita confirmación de la actividad realizada. 5. Finaliza el caso de uso
Caminos alternativos	El Actor repite el paso 3 tantas veces como solicitudes hayan sido realizadas. El Actor puede en el paso 3 cancelar la ejecución del caso de uso por lo cual puede saltar al paso 5.

Fuente : Elaboración Propia

Tabla 3.4: **Descripción detallada de los casos de uso.(3/5)**

Post-Condición	El caso de uso finaliza una vez que se actualiza la información en la base de datos.
Caso de Uso	Actualizar Odontograma
Actores	Doctor, Asistente, Manejador de Base de Datos.
Precondición	Administrar Paciente, Administrar Tratamientos
Flujo de sucesos	1. Este caso de uso, comienza cuando el actor selecciona en las opciones del paciente “Ver Odontograma”. 2. El sistema muestra el Odontograma y las enfermedades y/o tratamientos que pueden realizarse o estados que pueden “aplicarse 3. El actor selecciona un diente 4. El sistema “habilita” la opción de poder seleccionar un diente o modificar el estado del diente 5. El actor selecciona un diente 6. El sistema “habilita” la opción de modificar el estado del diente, o seleccionar una enfermedad. 7. El actor selecciona una enfermedad o un estado para el diente seleccionada anteriormente. 8. El sistema “asocia” la enfermedad o el estado, a esta cara del diente y muestra el *resultado* 9. Finaliza el caso de uso
Caminos alternativos	El Actor repite el paso 3 ó 5 tantas veces como solicitudes sean realizadas. El Actor puede en el paso 4 cancelar la ejecución del caso de uso por lo cual el sistema puede saltar al paso 5.
Post-Condición	El caso de uso finaliza una vez que se actualiza la información en la base de datos.

Fuente : Elaboración Propia

Tabla 3.4: **Descripción detallada de los casos de uso.(4/5)**

Caso de Uso.	Actualizar Tratamientos
Actores	Doctor, Base de datos SACMO.
Precondición	Administrar Historia Médico Odontológica.
Flujo de sucesos	1. El Actor invoca el caso de uso Consultar tratamiento. 2. Este caso de uso, comienza cuando el actor selecciona en la parte de tratamientos, la opción “agregar. 3. El sistema muestra los campos necesarios para llenar. 4. El sistema valida que el contenido de los campos sea el correcto, si es así agrega el tratamiento, sino *repite paso 3* 5. Finaliza el caso de uso.
Caminos alternativos	1. El Actor selecciona la opción modificar en la sección de tratamientos 2. . El sistema muestra los campos con los datos del tratamiento existente 3. El Actor modifica los datos que desea 4. El sistema valida que el contenido de los campos sea el correcto, si es así modifica el tratamiento, sino *repite paso 2*.
Post-Condición	El caso de uso finaliza una vez que se actualiza la información en la base de datos.
Caso de Uso.	Administrar Citas Odontológicas
Actores	Asistente , Manejador de Base de Datos
Precondición	Paciente, para reservar una Hora y turno
Flujo de sucesos	1. Comienza cuando el actor selecciona la opción agregar turno en la *agenda*. 2. El sistema muestra los odontólogos disponibles 3. El Actor selecciona el médico que desea 4. El sistema muestra los Turnos disponibles 5. El Actor selecciona el Turno disponible

Fuente: Elaboración Propia

Tabla 3.4: Descripción detallada de los casos de uso.(5/5)

Flujo de sucesos	6. El sistema muestra los horarios disponibles en la agenda para ese médico y ese Turno 7. El Actor reserva el turno que desea (en base al registro de asistencia sobre el paciente o si es nuevo, el que éste desee, siempre que esté disponible 8. El sistema agenda el turno y lo reserva. 9. Finaliza el caso de uso
Caminos alternativos	1. Comienza cuando el actor selecciona la opción cancelar turno en la *agenda*. 2. El sistema pide una confirmación al usuario 3. El usuario confirma que desea cancelar el turno 4. El sistema cancela el turno. Y regresa a selecciones nuevo .
Post-Condición	El caso de uso finaliza una vez que se actualiza la información en la base de datos.
Caso de Uso.	Administrar Pacientes
Actores	Doctor, asistente, Manejador de Base de Datos
Flujo de sucesos	1. Este caso de uso, comienza cuando el actor selecciona agregar paciente 2. El sistema muestra los campos necesarios para llenar 3. El usuario completa los datos 4. El sistema valida que el contenido de los campos sea el correcto, si es así agrega el paciente, sino *repite paso 2* 5. Finaliza el caso de uso
Caminos alternativos	El Actor repite el paso 3 tantas veces como sean realizadas. El Actor puede en el paso 4 cancelar la ejecución del caso de uso por lo cual el sistema puede saltar al paso 5. El caso de uso finaliza una vez que se actualiza la información en la base de datos.

Fuente: **Elaboración Propia**

3.3.7 Requisitos de dispositivos específicos

Servidor: se recomienda, Procesador Intel Xeon secuencia 3000. Memoria RAM mayor o igual a 2 GB, 250 GB de Disco Duro SATA y Sistema Operativo Windows 2000 service pack 4.

Clientes: se recomienda, un procesador Pentium 4, 1,60 GHz o más, 512 MB de memoria de acceso aleatorio (RAM)**,** un monitor de 14" mínimo, con capacidad para mostrar una resolución de 800 x 600 píxeles y una calidad de color de 32 bits.

El sistema operativo establecido es Ubuntu 10.4 y el software para Internet Mozilla Firefox 3.0.

3.3.8 Requisitos No Funcionales (Atributos de calidad)

Este tipo de requisitos tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares.

Rendimiento: el tiempo de respuesta de la aplicación depende de las características físicas de los equipos y va de la velocidad de procesamiento de los computadores.

Usabilidad: interfaz amigable para las personas autorizadas que usaran la aplicación, la cual debe generar comodidad durante la navegación, siempre enfocado en las actividades del negocio empresarial.

Disponibilidad: la aplicación estará disponible en todo momento para los usuarios autorizados.

Seguridad: se implementan claves de acceso para el control de ingreso al del sistema logrando así la protección de la información.

Mantenimiento: el administrador del sistema con su clave de acceso deberá poder insertar, modificar, eliminar y probar módulos y procesos del sistema lo cual generara un correcto mantenimiento del mismo.

Estándares: el sistema es flexible y se puede acceder desde distintas plataformas computacionales con diferentes sistemas operativos y navegadores webs

3.4 Flujo de trabajo análisis.

Durante este flujo se analizan, refinan y estructuran los requerimientos que han sido identificados.

Luego de este proceso descrito se obtiene una primera impresión del modelo de diseño y así se logra estructurar el sistema entero incluyendo su arquitectura, lo que facilita una mejor comprensión, preparación, modificación, y en general, un mejor mantenimiento de estos requerimientos

Dentro del modelo de análisis se maneja el concepto de clase del análisis, la cual representa una abstracción de una o varias clases y/o subsistemas del diseño.

3.4.1 Análisis de requerimientos y/o requisitos.

La revisión y formalización de requisitos recopilados es de suma importancia para obtener el conjunto de especificaciones necesarias para desarrollar la aplicación. Generalmente la revisión se realiza en términos de sub-etapas y sus diagramas correspondientes. Estas son:

3.4.2 Especificación de los grupos de usuarios formalmente descritos.

En esta sub-etapa se identifica los usuarios que intervienen en cada uno de los procesos que forman los distintos casos de uso. La especificación en grupos de usuarios determina

su ubicación dentro de la aplicación, permitiendo saber su nivel de acceso e importancia dentro de la misma, esto ayuda a disminuir los riesgos del sistema, así como también detalla que datos deben manejar cada uno de ellos y en que procesos están involucrados.

El actor Asistente posee los mismos beneficios del actor que el actor Doctor además de otros únicos para él.

La herencia explicada anteriormente se puede observar en el siguiente diagrama (Figura 3.3)

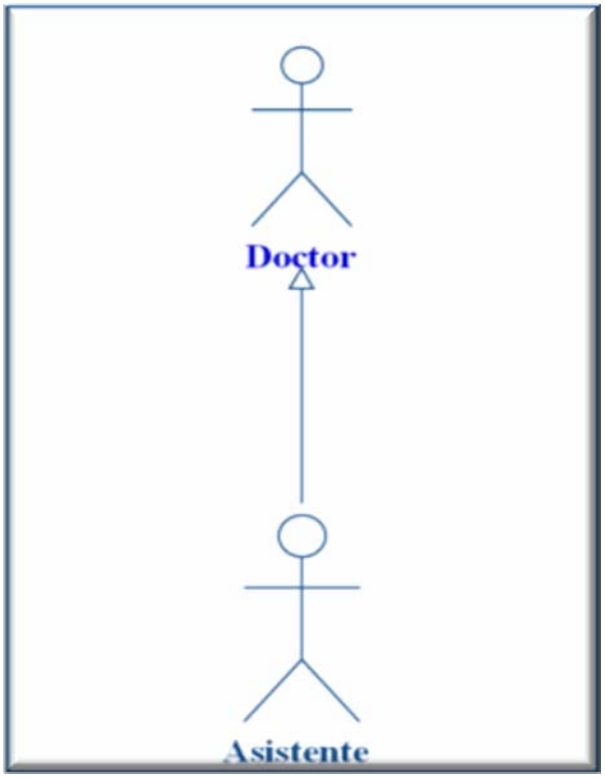


Figura 3.3: Especificación de Grupos formalmente descritos.
Fuente: **elaboración propia**

3.4.3 Especificación de los casos de usos por usuarios

En este punto se realiza la descripción formal de las unidades de interacción con la aplicación, de los usuarios o actores obtenidos en los grupos ya definidos, por medio de los casos de uso, tablas o diagramas de actividades.

En este caso se escogió los diagramas de casos de uso establecidos por el lenguaje unificado de modelado, UML

3.4.3.1 Actor Asistente

La herramienta WebML permite definir los casos de uso por usuario, en este caso se detalla al actor asistente, el cual tiene las opciones de: Acceder, Generar reportes, Buscar datos de pacientes, administrar citas, Modificar Datos, Eliminar Datos y Salir del Sistema. Ver Figura 3.4.

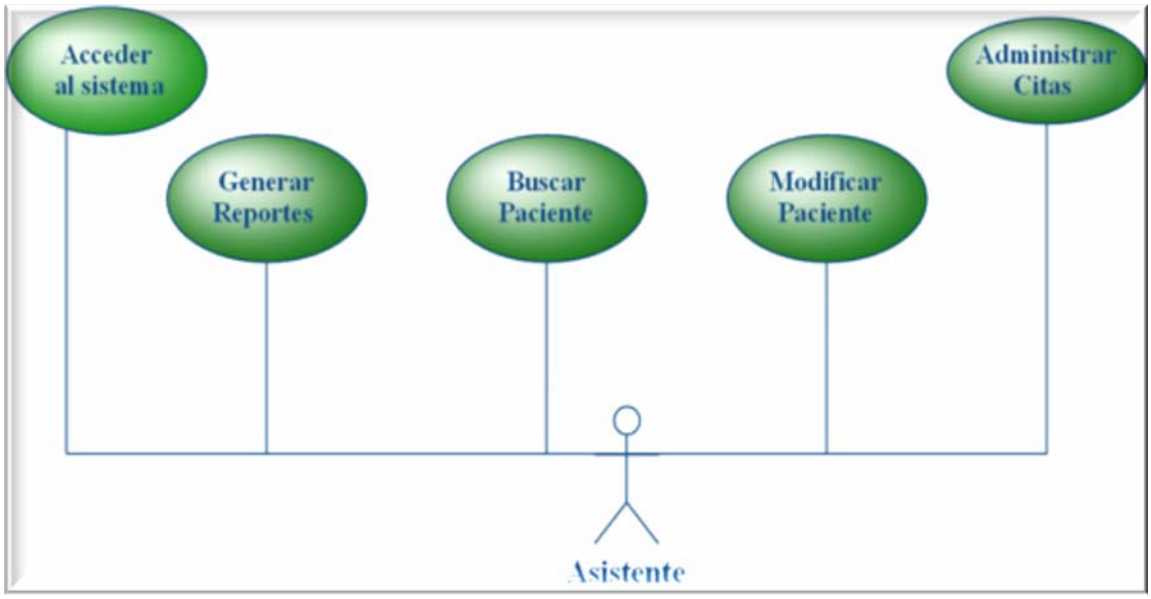


Figura 3.4: Usuario Asistente.
Fuete: Elaboración Propia.

3.4.4 Diagramas de análisis

Las clases de análisis identifican y describen los casos de uso más críticos del sistema, lo que permite que el sistema crezca incrementalmente a medida que se analicen los requisitos. Una clase de análisis se centra en el tratamiento de los requisitos funcionales y define atributos generalmente conceptuales y reconocibles en el dominio del problema.

Las clases de análisis siempre encajan en uno de tres estereotipos básicos: De interfaz, para modelar la interacción entre el sistema y sus actores, de control para

representar coordinación, secuencia miento, transacciones y control entre objetos o de entidad para modelar información de larga vida o permanente.

Cada estereotipo implica una semántica específica, lo cual constituye un método potente y consistente de identificar y describir las clases de análisis.

En este capítulo sólo se esboza una parte del análisis, por tanto se presentarán los diagramas de clase de análisis y de colaboración para los casos de uso más importantes dentro del sistema.

Entre los diagramas de clases de análisis que se tocara en este capítulo están,

- ✓ Clase de Interfaz: que son Acceder y Cargar datos de pacientes, estas ayudan a modelar la interacción entre los actores y el sistema, (es decir, usuarios y sistema externos). Esta interacción a menudo implica recibir (y presentar) información y peticiones de (y hacia) los usuarios y los sistemas externos.

- ✓ Clases de Entidad: que es LDAP, Usuario-paciente, las cuales representan la información de larga vida dentro del sistema Se utilizan para modelar información que posee una vida larga y que es a menudo persistente. Las clases de entidad modelan información y el comportamiento asociado de algún fenómeno o concepto, como una persona, un objeto del mundo real o un suceso del mundo real.

- ✓ Clases de Control: Representan coordinación, secuencia, transacciones y control de otros objetos y se usan con frecuencia para encapsular el control de un caso de uso en concreto.

Estos estereotipos están estandarizados en UML y se utilizan para ayudar a los desarrolladores a distinguir el ámbito de las diferentes clases. Cada uno de estos estereotipos de clase encapsula un tipo diferente de comportamiento (o funcionalidad, si

se quiere). Cada estereotipo tiene su propio símbolo, como se puede observar en la Figura 3.5.

Figura 3.5: Estereotipos de Clases.
Fuente: elaboración propia.

3.4.4.1 Diagrama de análisis para el caso de uso Acceder

Debido a que los actores que deben autenticarse para entrar al sistema son: Doctor y Asistente, se consideró el nombre "*Usuario*" para que de manera general se represente que puede ser cualquiera de los actores antes mencionado.

La Figura 3.6 muestra el diagrama de clases de análisis para el caso de uso Acceder, que expone la clase interfaz *Acceder*, utilizada por el *Usuario* para introducir los datos necesarios, los cuales serán validados por la clase de control *Autenticar Datos*, que consulta en el *LDAP* de las Unidad Médica si el usuario es personal de la Institución y en la tabla *Usuario-status* de la base de datos del sistema, la existencia del mismo.

3.4.4.2 Diagrama de análisis para el caso de uso Cargar datos de Paciente

La Figura 3.7 muestra el diagrama de clases de análisis para el caso de uso Cargar datos de Paciente, el cual describe el proceso de como el *Asistente* ingresa los datos de un paciente a través de la clase interfaz *Cargar datos de paciente*, que es la encargada de presentar al Asistente la pantalla de interacción necesaria para llevar a cabo la inserción de los datos

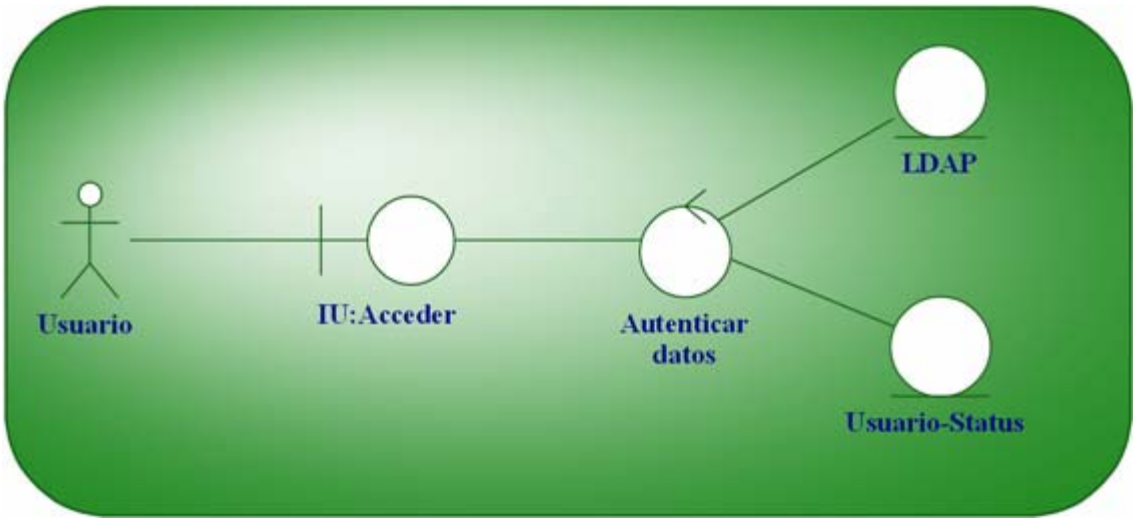


Figura 3.6: Diagrama de análisis para el caso de uso Acceder.
Fuente: Elaboración Propia.

Luego esta interfaz se conecta con la clase de control *Gestor cargar datos de paciente*, el cual se encarga de procesar la información de la solicitud, e inmediatamente es almacenada en las clases entidad *paciente*, que en este caso será una tabla de la base de datos del sistema donde se almacenan todos los datos concernientes a cada solicitud que se ingrese.

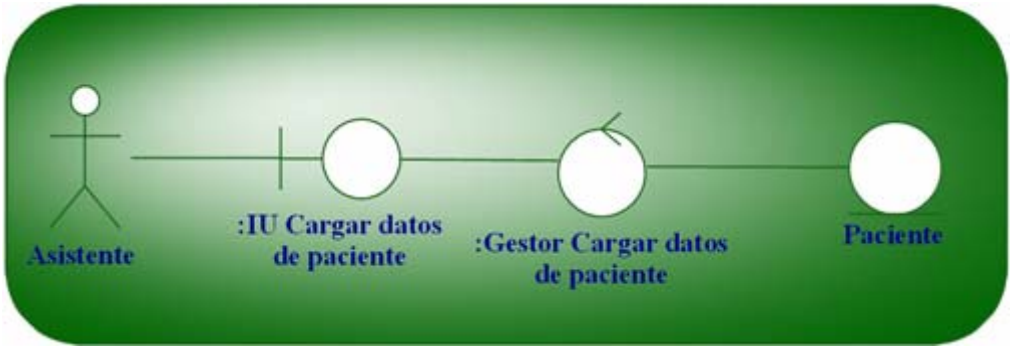


Figura 3.7: Diagrama de análisis para el caso de uso cargar datos de paciente.
Fuente: Elaboración Propia.

3.4.5 Diagramas de colaboración

Muestra la interacción entre varios objetos y los enlaces que existen entre ellos. Representa las interacciones entre objetos organizadas alrededor de los objetos y sus vinculaciones. A diferencia de un diagrama de secuencias, un diagrama de

colaboraciones muestra las relaciones entre los objetos, no la secuencia en el tiempo en que se producen los mensajes. Como se podrán observar los diagramas de colaboración poseen la misma estructura de los diagramas de clase de análisis, solo que ahora se representa la interacción entre estos componentes, y para ello se utilizan los mensajes, mediante un número se pretende organizar dicho mensajes y las flechas indican la dirección del flujo.

Describen el proceso a detalle señalando la interacción y comunicación entre las clases, actores y entidades. El nombre de un mensaje debería denotar el propósito del objeto invocante en la interacción con el objeto invocado

3.4.5.1 Diagrama de colaboración para el caso de uso Acceder

Debido a que los actores que deben autenticarse para entrar al sistema son: Consultor y Administrador, se consideró el nombre "*Usuario*" para que de manera general se represente que puede ser cualquiera de los actores antes mencionado. Ver Figura 3.8

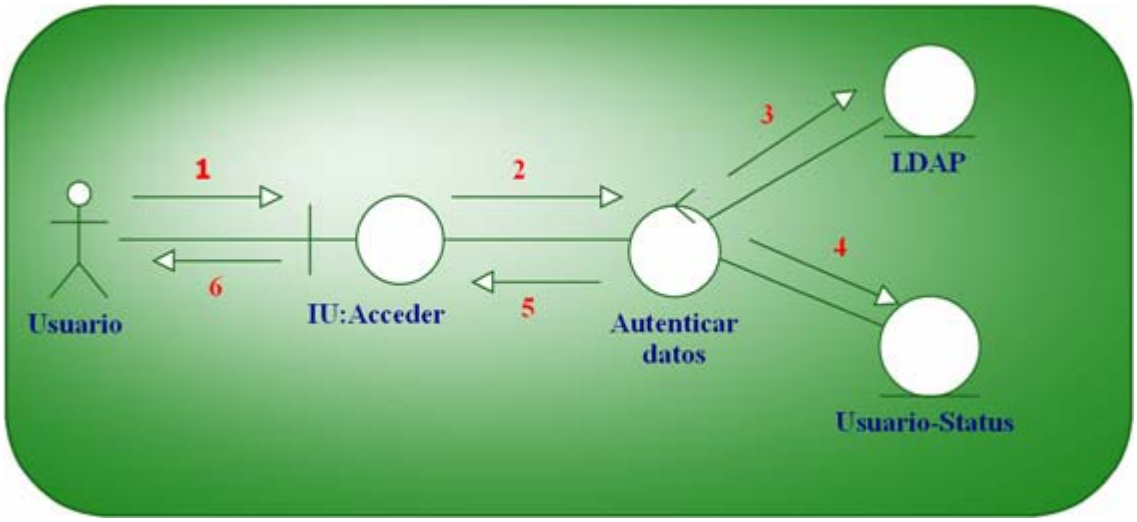


Figura 3.8: Diagrama de colaboración para el caso de uso Acceder.
Fuente: Elaboración Propia.

Leyenda:

1. Ingresar datos del usuario (Usuario y Contraseña).

2. Validar datos del usuario.
3. Consultar registro de usuario, en la clase entidad LDAP (Usuario y contraseña).
4. Consultar registro de usuario, en la clase entidad Usuario-Status (Usuario y Status).
5. Respuesta de acceso de usuario si es válido regresa el Status del usuario.
6. Muestra interfaz dependiendo de la respuesta anterior.

El Usuario decide entrar al sistema y activa el objeto interfaz IU Acceder donde éste debe ingresar los datos de acceso requeridos para su autenticación (1). IU Acceder valida si algún campo usuario y/o contraseña se encuentra vacío y si es el caso se muestra un mensaje de error y actualiza la interfaz requiriendo nuevamente los datos de autenticación; al estar ambos campos con sus respectivos datos esta interfaz solicita al objeto Autenticar datos que valide el acceso del mismo al sistema (2). Para ello el objeto Autenticar datos solicita al objeto LDAP, que representa el Doctor activo de la institución, la búsqueda en su registro de la existencia del actor como empleado de la Unidad Médica (3). Y al objeto Usuario, que representa una tabla de la base de datos del sistema, la búsqueda en su registro de la existencia del Actor como usuario autorizado para el uso del sistema, y si es el caso verifica el Status con el cual puede acceder al mismo (4). Finalmente después de validar la autenticación del usuario, éste recibe una respuesta de acceso (5), bien sea de acceso válido, mostrando la bienvenida al sistema, o de acceso no válido, prohibiendo su entrada e indicándole, clave o contraseña inválida. (6).

3.4.5.2 Diagrama de colaboración para el caso de uso Cargar datos de Paciente.

En la Figura 3.9 se puede observar las interacciones asociadas al caso de uso Cargar datos de paciente y sus respectivos pases de mensajes de un objeto a otro que nos ayuda a mostrar la implementación de esta operación.

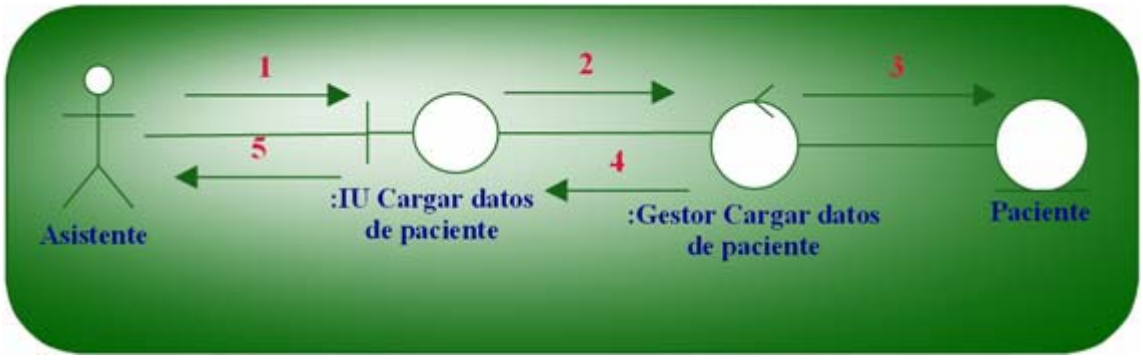


Figura 3.9: Diagrama de colaboración para el caso de uso Cargar datos de paciente.
Fuente: Elaboración Propia.

Leyenda:

1. Ingresar datos del paciente.
2. Validar los datos de solicitud.
3. Ingresar los datos del paciente en la clase entidad paciente.
4. Respuesta de la solicitud de ingreso del registro.
5. Confirmación de guardado.

Esta operación es permitida únicamente a aquellos usuarios registrados y autenticados y que además su Status sea de Asistente. Se realiza de la siguiente manera, el Asistente ingresa la información solicitado correspondiente a los datos de los pacientes (1). Estos archivos pasan por un proceso de validación (2) que lo hace el gestor cargar datos de paciente. La información ingresada una vez que se le piden los datos de los pacientes y estos fueron validados se almacenan en la base de datos (3) se envía la respuesta de solicitud valida o invalida (4) y por último, se efectúa la confirmación de guardado de datos (5).

3.5 Flujo de trabajo diseño

En el diseño se modela el sistema y se encuentra su forma (incluida la arquitectura) para que soporte todos los requisitos. Como esta es la fase de inicio, para este flujo de trabajo

solo se presentará un bosquejo de las interfaces principales que debería presentar el sistema.

3.5.1 Arquitectura candidata

La arquitectura propuesta para esta aplicación se muestra en la Figura 3.10 está dividida en cuatro partes, cada una representando algunas de las funcionalidades del sistema.

La interfaz de usuario, comprende los componentes relacionados con el diseño, captura de datos y validación de datos.

En controlador de interfaz, se encuentran todos los componentes que toman decisión sobre el flujo de eventos, que depende de las solicitudes que los usuarios pidan al sistema. Estas solicitudes no son más que tareas que el sistema debe realizar de manera automática o no. Tareas como consultas y actualizaciones.

La parte lógica de la aplicación contiene las entidades necesarias para la funcionalidad del sistema.

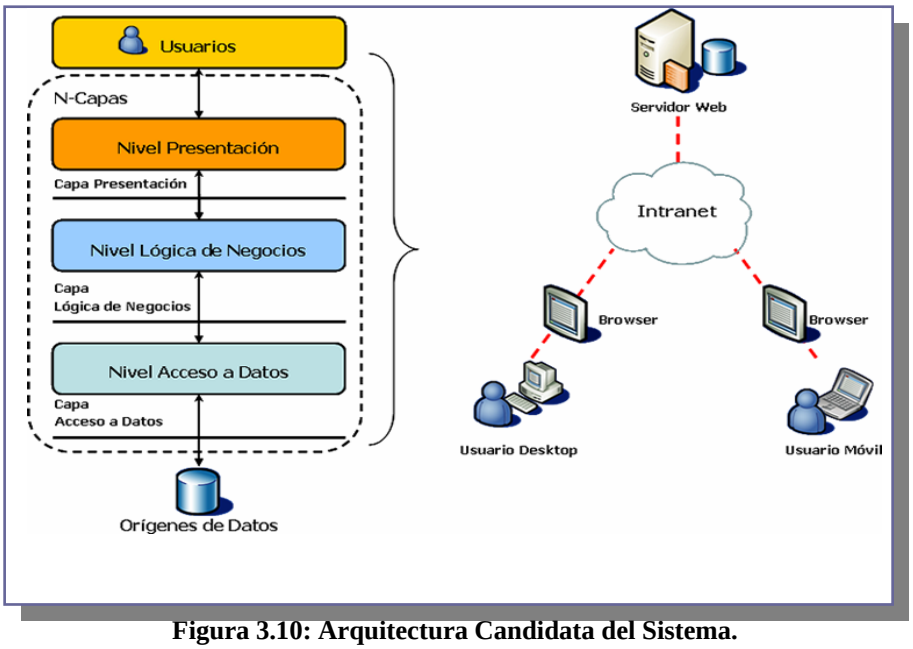


Figura 3.10: Arquitectura Candidata del Sistema.

Fuente: Elaboración Propia.

A continuación, en la Figura 3.11 se pueden observar cada uno de los niveles de la configuración de red con sus características correspondientes.

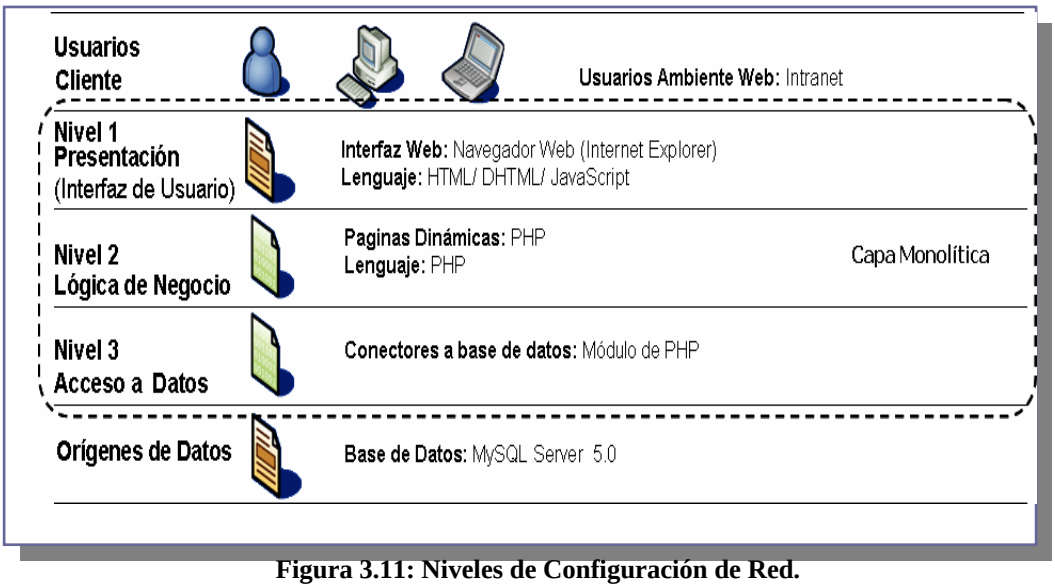


Figura 3.11: Niveles de Configuración de Red.
Fuente: Elaboración Propia.

3.5.2 Interfaz de inicio de sesión.

La interfaz de inicio de sesión, mostrada en la Figura 3.12, es la primera a la que tiene acceso cualquiera de los usuarios.

Figura 3.12. Prototipo de Interfaz de Inicio de Sesión.
Fuente: elaboración propia.

Se puede observar un área asignada para el banner. Un área central donde está un formulario para el ingreso de los datos de identificación del usuario.

3.6 PLANIFICACIÓN DE LAS SIGUIENTES FASES

3.6.1 Fase de Elaboración

El objetivo principal de esta fase llevar a cabo el diseño y estructura del software para que de soporte a todos los requisitos que se obtuvieron durante la fase de inicio. En el cual se diseña la base de datos, a través de Microsoft SQL estructurando los datos a través de tablas relacionales. Se desarrollará un prototipo de demostración del software mediante los formatos de pantalla.

3.6.2 Fase de Construcción

El objetivo de esta fase es codificar los procedimientos diseñados anteriormente, su implementación a través de integración y la realización de pruebas necesarias a los mismos, con la finalidad de detectar posibles errores, observar el comportamiento y consistencia de estos procedimientos, para luego realizar la integración de todo el sistema. En esta etapa se realizará la evaluación del software a fin de determinar si se está ajustando a las necesidades requeridas para garantizar la calidad del producto.

3.6.3 Fase de Transición

La fase de transición estará guiada por la prueba de la versión beta del sistema sobre la plataforma para la cual fue diseñado con la finalidad de ajustarlo a los parámetros del entorno de operación y corregir posibles defectos encontrados antes de la entrega de la versión final del software.

3.7 . Diseño

Para la fase inicial, el flujo de trabajo de diseño es muy breve porque aún no poseemos todos detalles del complejo sistema que se construirá. Lo que se obtiene de este flujo de trabajo es un esbozo del modelo de diseño y una arquitectura inicial estable. Basándose

en los requisitos obtenidos, y análisis realizados se puede definir la arquitectura del sistema. A continuación se trata el tema con más detalle.

3.8 EVALUACIÓN DE LA FASE

En esta fase se llevó a cabo la identificación y descripción de los requerimientos funcionales presentados en el modelo del dominio y el modelo de casos de uso respectivamente.

Los casos de uso identificados en el flujo de trabajo de los requisitos fueron estudiados en detalle en forma de diagramas de colaboración en el flujo de análisis. Además se identificaron y describieron las clases de análisis, las cuales fueron asignadas a paquetes del modelo de análisis. Se realizó un esbozo de la arquitectura del sistema con una primera vista del modelo de diseño

.

No se realizó ningún avance significativo de los flujos de trabajo de implementación y prueba por no ser necesario en esta fase.

Capítulo IV

FASE DE ELABORACIÓN

4.1 INTRODUCCIÓN

La fase de elaboración construye la línea base de la arquitectura del sistema. Los principales objetivos para esta fase:

- Recopilar la mayor parte de los requisitos que aún queden pendientes, formulando los requisitos funcionales como casos de uso.

- Establecer una base de la arquitectura sólida -la línea base de la arquitectura- para guiar el trabajo durante las fases de construcción y transición, así como en las posteriores generaciones del sistema.

- Continuar la observación y control de los riesgos críticos que aún queden, e identificar riesgos significativos hasta el punto de que se pueda estimar su impacto en el análisis negocio, y en particular en la apuesta económica.

- Completar los detalles del plan del proyecto.

Para esta fase se debe desarrollar alrededor del 80 por ciento de los casos de uso y abordar los riesgos que interfieran en la consecución de este objetivo.

En esta fase se trabajará a partir del flujo de trabajo que se refiere al análisis, debido a que no se identificaron nuevos actores, casos de uso, ni requisitos que se pudiesen agregar a los establecidos en la fase de inicio.

4.2 Flujo de trabajo análisis.

En la fase de inicio se desarrolló el análisis de la arquitectura sólo hasta el punto de determinar que había una arquitectura del sistema factible. Ahora en esta fase se extiende el análisis hasta el punto de que pueda servir como base a la línea principal de la arquitectura ejecutable.

4.2.1 Especificación de los casos de usos por usuarios.

En este punto se realiza la descripción formal de las unidades de interacción con la aplicación, de los usuarios o actores obtenidos en los grupos ya definidos, por medio de los casos de uso, tablas o diagramas de actividades. En este caso se escogió los diagramas de casos de uso establecidos por el lenguaje unificado de modelado, UML.

4.2.1.1 Actor Doctor

El actor Doctor, es el que posee privilegios para entrar en cualquier modulo del sistema, él es el encargado de administrar la base de datos de la que depende el sistema para su funcionamiento, esto abarca desde modificar y asignar Status para nuevos usuarios hasta poder modificar la información relacionada a los tratamientos.

En la Figura 4.1 se puede observar todos los casos de uso a los que el actor Doctor tiene acceso

4.2.2 Diagrama de análisis para el caso de uso Generar tratamiento.

Generar tratamiento es la encargada de presentar al actor la pantalla de interacción necesaria para enviar la información relacionada con los tratamientos para ello se toman los datos de la clase entidad cedula Paciente y de la clase entidad Citas Odontológicas, para poder listar en la interfaz los tratamientos al cargarse la misma.

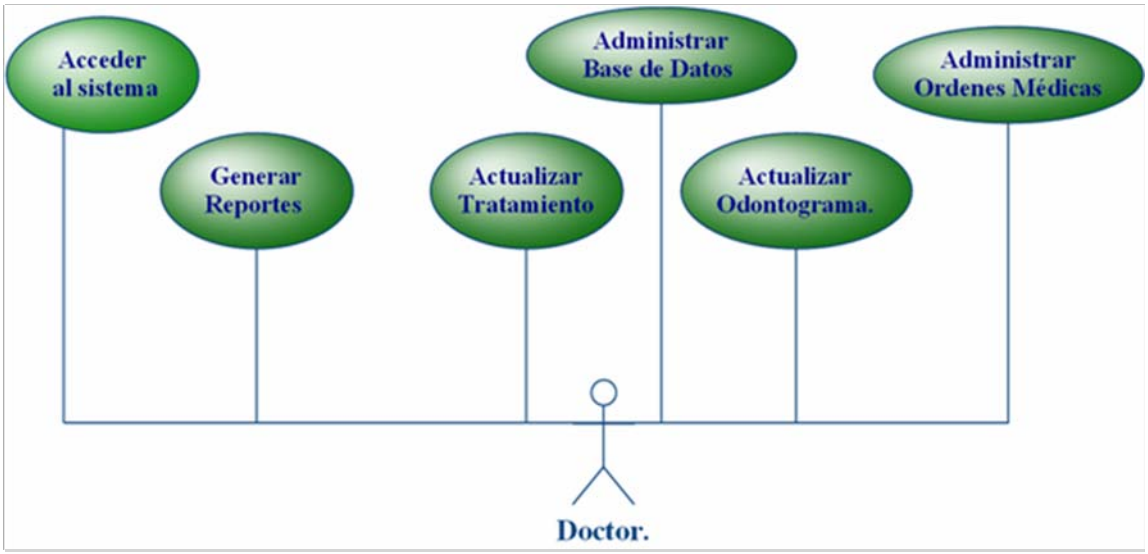


Figura 4.1: Actor Doctor.
Fuente: Elaboración Propia.

En esta interfaz se lleva a cabo la selección de tratamientos que desea actualizar, luego esta interfaz se conecta con la clase de control Gestor Generar datos de tratamientos, encargada de procesar los cambios realizados al Odontograma, que inmediatamente será almacenado en la clase entidad Tratamiento y en la clase entidad Historia Médica..

En la Figura 4.2 Puede observarse el diagrama de análisis para el caso de uso Generar datos de tratamiento, siendo el Doctor el que tiene acceso a este caso de uso.

4.2.2.1 Diagrama de análisis para el caso de uso Actualizar datos de tratamiento

Actualizar datos de tratamiento es la encargada de presentar al actor la pantalla de interacción necesaria para enviar la información relacionada al tratamiento, para ello se toman los datos de la clase entidad paciente y de la clase entidad control de citas, para poder listar en la interfaz los tratamiento al cargarse la misma.

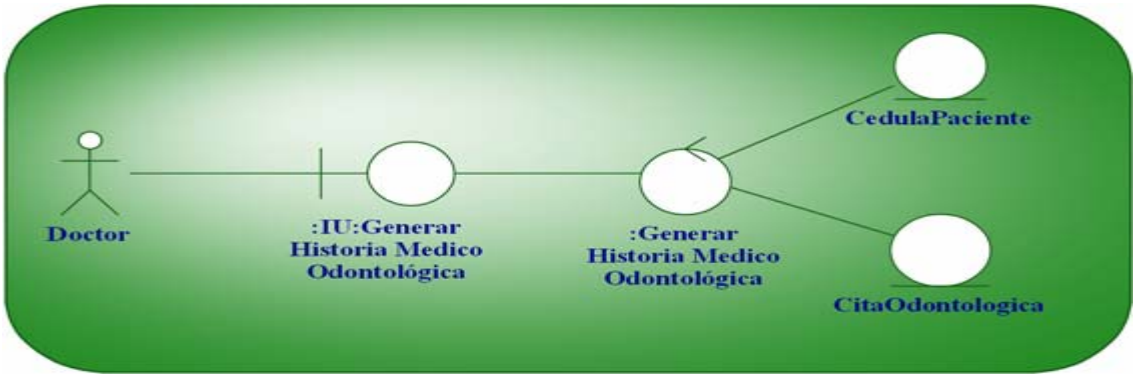


Figura 4.2: Diagrama de análisis para el caso de uso Generara Historia Medico Odontológico.
Fuente: Elaboración Propia.

En esta interfaz se lleva a cabo la selección del tratamiento que desea actualizar, luego esta interfaz se conecta con la clase de control Gestor actualizar datos de tratamiento, encargada de procesar los cambios realizados al tratamiento , que inmediatamente será almacenado en la clase entidad Tratamiento y en la clase entidad Historia. Odontológica

En la Figura 4.3 puede observarse el diagrama de análisis para el caso de uso Actualizar datos de teléfono, siendo el Administrador el que tiene acceso a este caso de uso.

4.2.2.2 Diagrama de análisis para el caso de uso actualizar Odontograma

Actualizar Odontograma es la encargada de presentar al actor la pantalla de interacción necesaria para la asignación o actualización del Odontograma de un paciente para ello se toman los datos de la clase entidad Odontograma, para poder mostrar en la interfaz historial odontológica de los pacientes.

En esta interfaz se lleva a cabo la actualización de los tratamientos específico a un paciente, luego esta interfaz se conecta con la clase de control Gestor actualizar Odontograma, que es la encargada de procesar la actualización que luego se almacenará en la clase entidad Odontograma, en la Figura 4.4 se observa el diagrama correspondiente.

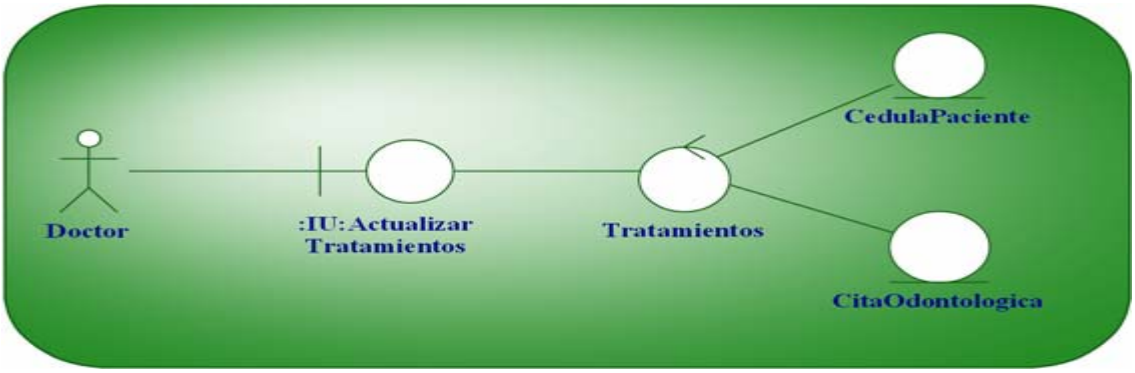


Figura 4.3: Diagrama de análisis para el caso de uso Actualizar datos de tratamiento.
Fuente: Elaboración Propia.

4.2.2.3 Diagrama de análisis para el caso de uso Administrar Orden Medica

Administrar Orden Medica. es la encargada de presentar al actor la pantalla de interacción necesaria para la asignación de Ordenes Medicas como, Oftalmología, ginecología, próstata, para ello se toman los datos de la clase entidad Datos del paciente , para poder mantener actualizados los datos dentro del sistema.

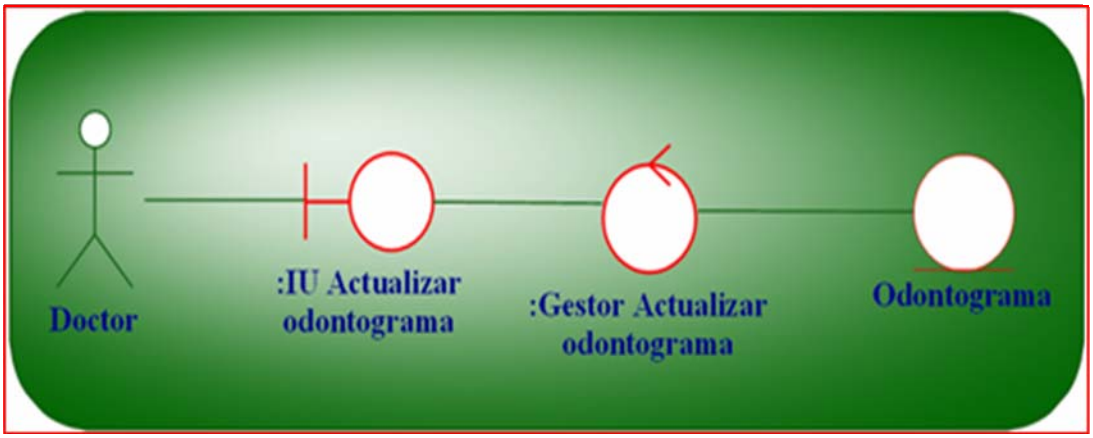


Figura 4.4: Diagrama de análisis para el caso de uso Actualizar Odontograma.
Fuente: Elaboración Propia.

En esta interfaz se lleva a cabo la Actualización de Ordenes Medicas luego esta interfaz se conecta con la clase de control Gestor agregar datos de Ordenes Medicas, que es la encargada de procesar dicha operación que luego se almacenará en la clase entidad

Datos de ordenes Medicas, y en la en la clase entidad paciente en la Figura 4.5, se observa el diagrama correspondiente.

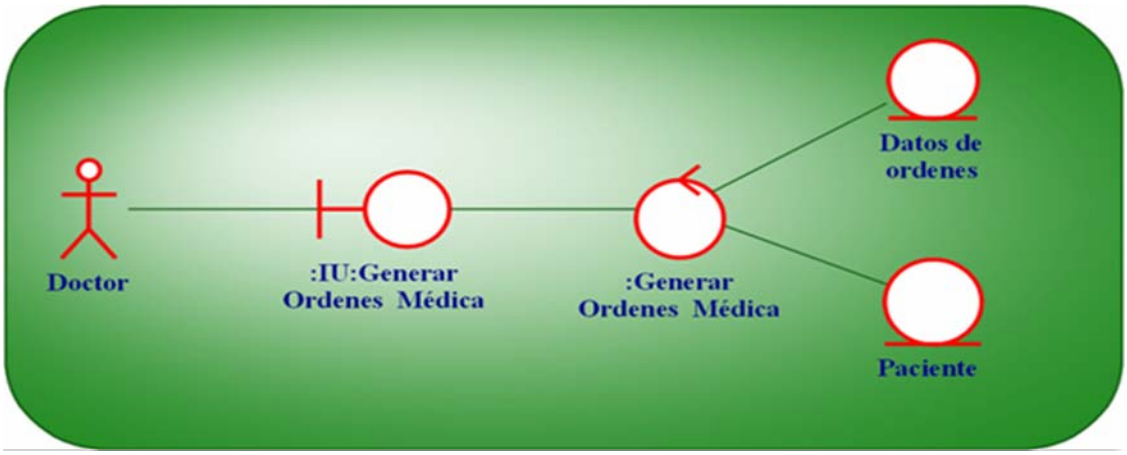


Figura 4.5: Diagrama de análisis para el caso de uso Generar Ordenes Médicas.
Fuente: Elaboración Propia.

4.2.3 Diagrama de colaboración

Un diagrama de colaboración es una forma alternativa al diagrama de secuencia de mostrar un escenario. Este tipo de diagrama muestra las interacciones entre objetos organizadas entorno a los objetos y los enlaces entre ellos. Proporcionan la representación principal de un escenario, ya que las colaboraciones se organizan entorno a los enlaces de unos objetos con otros.

Los diagrama de colaboración, describen el proceso a detalle señalando la interacción y comunicación entre las clases, actores y entidades. El nombre de un mensaje debería denotar el propósito del objeto invocaste en la interacción con el objeto invocado.

4.2.3.1 Diagrama de colaboración para el caso de uso Generar tratamiento

El diagrama de colaboración para el caso de uso generar tratamientos se muestra en la Figura 4.6, como este caso de uso puede ser accedido tanto por el actor Consultor como por el Administrador, se generalizo a estos actores por medio del actor Usuario.

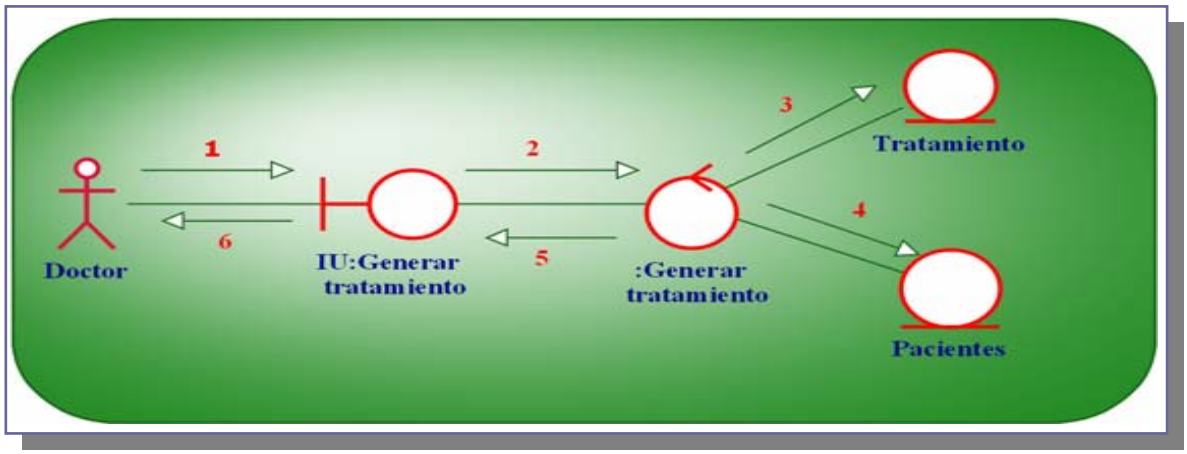


Figura 4.6: Diagrama de colaboración para el caso de uso Generar tratamiento
Fuente: Elaboración Propia.

Leyenda:

1. Verificar datos.
2. Fuente: Elaboración Propia.
3. Consultar tratamiento.
4. Consultar detalles de paciente.
5. Respuesta de solicitud de datos tratamientos.
6. Mostrar datos de tratamientos.

El actor Doctor desea verificar la información relacionada con un tratamiento y para ello activa la IU Generar tratamiento (1).

La IU Generar tratamiento solicita al objeto Gestor consultar datos de tratamiento la información asociada al tratamiento (2).

El objeto Gestor Generar tratamiento, solicita la información general del tratamiento la entidad Tratamiento (3) y la información detallada a la entidad paciente (4), para que esta a su vez maneje la respuesta de solicitud de datos de tratamiento (5) y así esta información sea mostrada al Usuario (6).

4.2.3.2 Diagrama de colaboración para el caso de uso Actualizar tratamiento

El diagrama de colaboración para el caso de uso Actualizar tratamientos se muestra en la Figura 4.7.

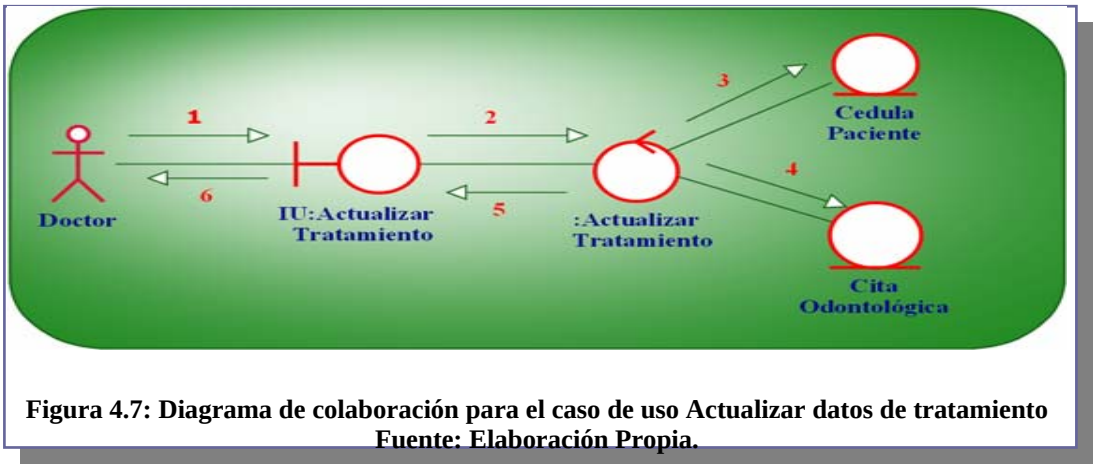


Figura 4.7: Diagrama de colaboración para el caso de uso Actualizar datos de tratamiento
Fuente: Elaboración Propia.

Leyenda:

1. Solicitar actualizar tratamiento.
2. Gestionar actualización.
3. Guardar cambios en paciente.
4. Guardar cambios en cita odontológica.
5. Respuesta a solicitud de actualización.
6. Mostrar mensaje de respuesta a solicitud de actualización.

El actor Doctor es el único con los permisos necesarios para actualizar datos de tratamientos y por medio de la IU Actualizar datos de tratamientos, solicita al sistema, insertar o modificar los datos de un tratamiento (1). IU Actualizar datos de tratamiento solicita al Gestor actualizar datos de tratamiento que se encarga de procesar los datos del tratamiento seleccionado y validar que acción se va hacer con la misma (2). Es así como el gestor solicita al objeto tratamiento y actualizar su registro, guardando los cambios que se hayan realizados (3). De la misma forma al proceso anterior el gestor solicita al objeto Citas odontológicas que guarde los cambios ocurridos (4). Luego el gestor le

envía a la clase interfaz la respuesta de la operación ya sea válida o invalida (5). Finalmente el doctor recibe un mensaje de respuesta por la operación de actualización, ya sea por haber insertado o modificado los datos de un tratamiento (6).

4.2.3.3 Diagrama de colaboración para el caso de uso actualizar Odontograma

El diagrama de colaboración para el caso de uso Asignar Roles se muestra en la Figura 4.8

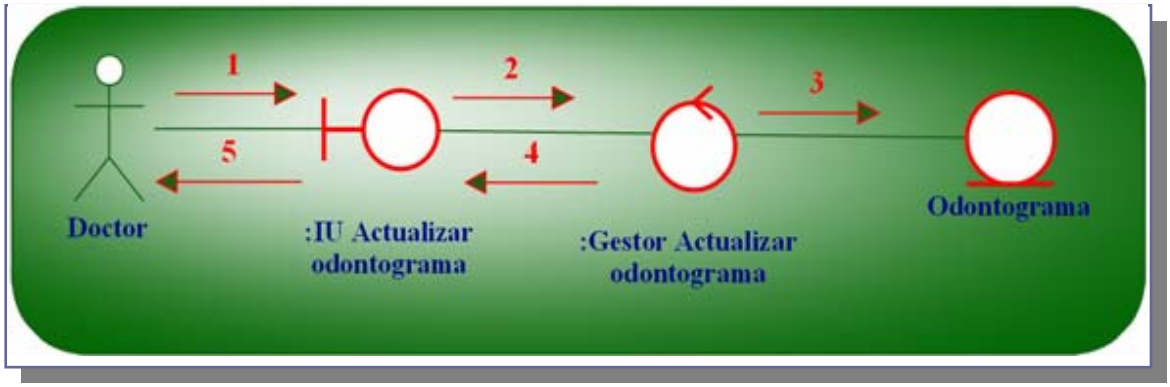


Figura 4.8: Diagrama de colaboración para el caso de uso actualizar Odontograma.
Fuente: Elaboración Propia.

Leyenda:

1. Solicitar actualizar Odontograma.
2. Gestionar actualización.
3. Guardar cambio.
4. Respuesta a solicitud de actualización.
5. Mostrar mensaje de respuesta a solicitud de actualización.

.El actor doctor es el único con derechos para la actualización del Odontograma y por medio de la IU actualizar Odontograma, realiza las actualizaciones del Odontograma de cada paciente (1). IU actualizar Odontograma solicita al gestor actualizar que se encarga de procesar la actualización de los Odontograma de los pacientes (2). Es así como el gestor solicita al objeto Odontograma y actualizar sus registros, guardando el cambio que haya hecho el doctor (3). Luego el gestor le envía a

la clase interfaz la respuesta de la operación ya sea válida o invalida (4). Finalmente el doctor recibe un mensaje de respuesta por la operación de actualizar Odontograma (5).

4.2.3.4 Diagrama de colaboración para el caso de uso generar ordenes Médicas

El diagrama de colaboración para el caso de uso administrar ordenes Medicas se muestra en la ¡Error! No se encuentra el origen de la referencia.

Leyenda:

1. Solicitar generar órdenes médicas.
2. Gestionar generar órdenes médicas.
3. Guardar cambios en los datos de órdenes.
4. Guardar cambios en pacientes.
5. Respuesta a solicitud de orden médica.
6. Mostrar mensaje de respuesta a solicitud de gestión orden médica.

El actor doctor es el que posee privilegios para generar órdenes médicas y lo realiza por medio de la IU generar órdenes médicas, solicita al sistema agregar una orden médicas, ya sea oftalmológica, ginecológica, urología. (1). IU generar órdenes médicas al Gestor generar órdenes médicas que procese estas solicitudes realizadas por el doctor (2). Es así como el Gestor solicita al objeto datos de ordenes actualizar su registro, guardando los cambios que se hayan realizados (3). Luego el gestor le envía a la clase interfaz la respuesta de la operación ya sea válida o invalida (4). Luego muestra la respuesta de solicitud de una orden médica (5). Luego Finalmente el Administrador recibe un mensaje con la respuesta obtenida de la generación de orden médica (6).

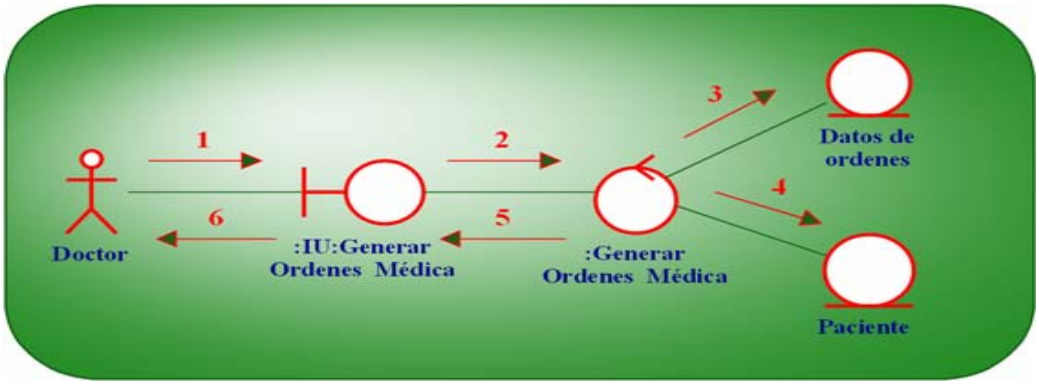


Figura 4.9: Diagrama de colaboración para el caso de uso generar órdenes Médicas.
Fuente: Elaboración Propia.

4.2.4 Análisis de la arquitectura

La definición de la arquitectura usando WebML se encuentra basada en la definición de las áreas (módulos de información) que constituyen a la aplicación Web y de las páginas que conforman dichas áreas, pues esto permite establecer de forma efectiva las vistas del sitio como producto final y a su vez concretar cómo estará compuesto el mismo y cómo será su navegación.

La visibilidad de las áreas está constituida de la siguiente manera:

- Área por defecto (Default área “D”): se dice que un área es por defecto cuando esta adjunta a la vista del sitio que es accedido.

- Landmark área “L”: cuando es globalmente accesible desde cualquier otra área dentro del sitio Web.

- Área Interna (internal área “I”): cuando es visible solo por medio de enlaces explícitos.
- La visibilidad de las páginas se diferencia de la siguiente manera:

- Página Principal (home page “H”): se presenta por defecto cuando el usuario ingresa al sitio Web.

- Página por defecto (default page “D”): se presenta por defecto cuando se accede al área.

- Landmark area “L”: tiene un alcance global desde todas las páginas encerradas en un mismo módulo (vista del sitio o área).

- Página interna (internal page “I”): implementado dependiendo del contenido.

Es por ello que esta metodología está altamente fundamentada en la implementación de diagramas explicativos que sustenten estos módulos de información. En La Figura 4.10muestra cómo se constituye la aplicación, con el propósito de mostrar al cliente la experiencia del sitio por medio de su página principal.

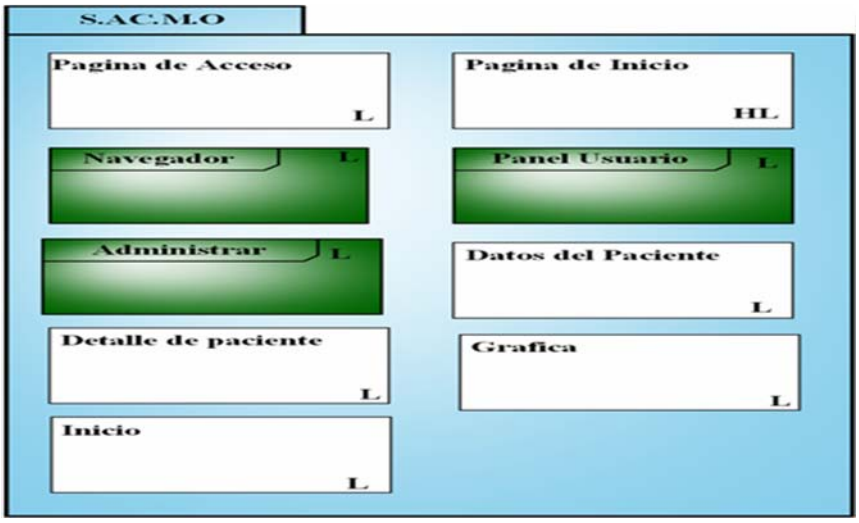


Figura 4.10: Vista Del Sitio. Pagina Inicial (Home Page).
Fuente: elaboración propia.

4.2.5 Descripción de paquetes del sistema SACMO.

Los paquetes se representan mediante rectángulos con pestañas y las relaciones de dependencias se muestran como flechas con líneas discontinuas.

Para el sistema desarrollado se identificaron una serie de paquetes, los cuales encapsulan los diferentes casos de usos que fueron definidos al realizar el análisis del sistema, los paquetes definidos fueron los siguientes: “Gestión de autentificar”, “Gestión de visualización”, “Gestión de administración”, “Gestión de teléfonos.

Gestión de Autentificar: Este paquete se encarga del proceso de autentificar a los usuarios que desean hacer uso de la aplicación.

Gestión de Visualización: Es el encargado de cómo mostrar a los usuarios toda la información referente a la aplicación.

Gestión de Doctor: Este paquete es el encargado de administrar y organizar toda la información de entrada y salida de la aplicación.

Gestión de Pacientes: Este paquete es el encargado de la relación de todos los datos referente a los Pacientes, tales como sus Historia, Odontograma, tratamiento.

4.3 Flujo de trabajo diseño.

En esta fase los casos de uso significativos se diseñan en términos de subsistemas de diseño.

Los propósitos del diseño son:

- Adquirir una comprensión en profundidad de los aspectos relacionados con los requisitos no funcionales y otras restricciones que se suponen.

- Crear una entrada apropiada y un punto de partida para actividades de implementación subsiguientes capturando los requisitos o subsistemas individuales e interfaces.

- Ser capaces de descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo, teniendo en cuenta la posible concurrencia.

4.3.1 Diseño de la Arquitectura

La arquitectura se divide en cuatro (4) capas para su mejor comprensión.

La capa específica, La Capa General de la Aplicación que contempla los paquetes: Gestión de Sesión, Gestión de Información, y Realización de Consultas, Administración de la Base de Datos, Gestión de Usuarios, Procesamiento de Pedidos. Estos paquetes se interconectan entre sí y son enlazados con el paquete Navegador Web en la Capa Intermedia.

La Capa Intermedia, que comprende el comportamiento lógico del sistema. En esta capa se cuenta con el Wamp Server, el cual contiene las herramientas: PHP que ejecuta los scripts que contienen la funcionalidad del sistema, MySQL que contiene la base de datos del sistema y la funcionalidad para establecer conexión con ésta y manipularla, y por último, el Servidor Apache para el manejo estático y dinámico del entorno. En este nivel también se encuentra el Navegador Web, el Lenguaje JavaSript, Lenguaje de Hojas de Estilo en Cascada (CSS) y el Sistema Manejador de Base de Datos (SMBD).

La Capa de Software del Sistema, que comprende el Sistema Operativo,Windows 2000/XP/Vista/7,donde se ejecuta la aplicación y el protocolo HTTP. En la Figura 4.11 se puede observar el Diseño de la Arquitectura del Sistema.

.

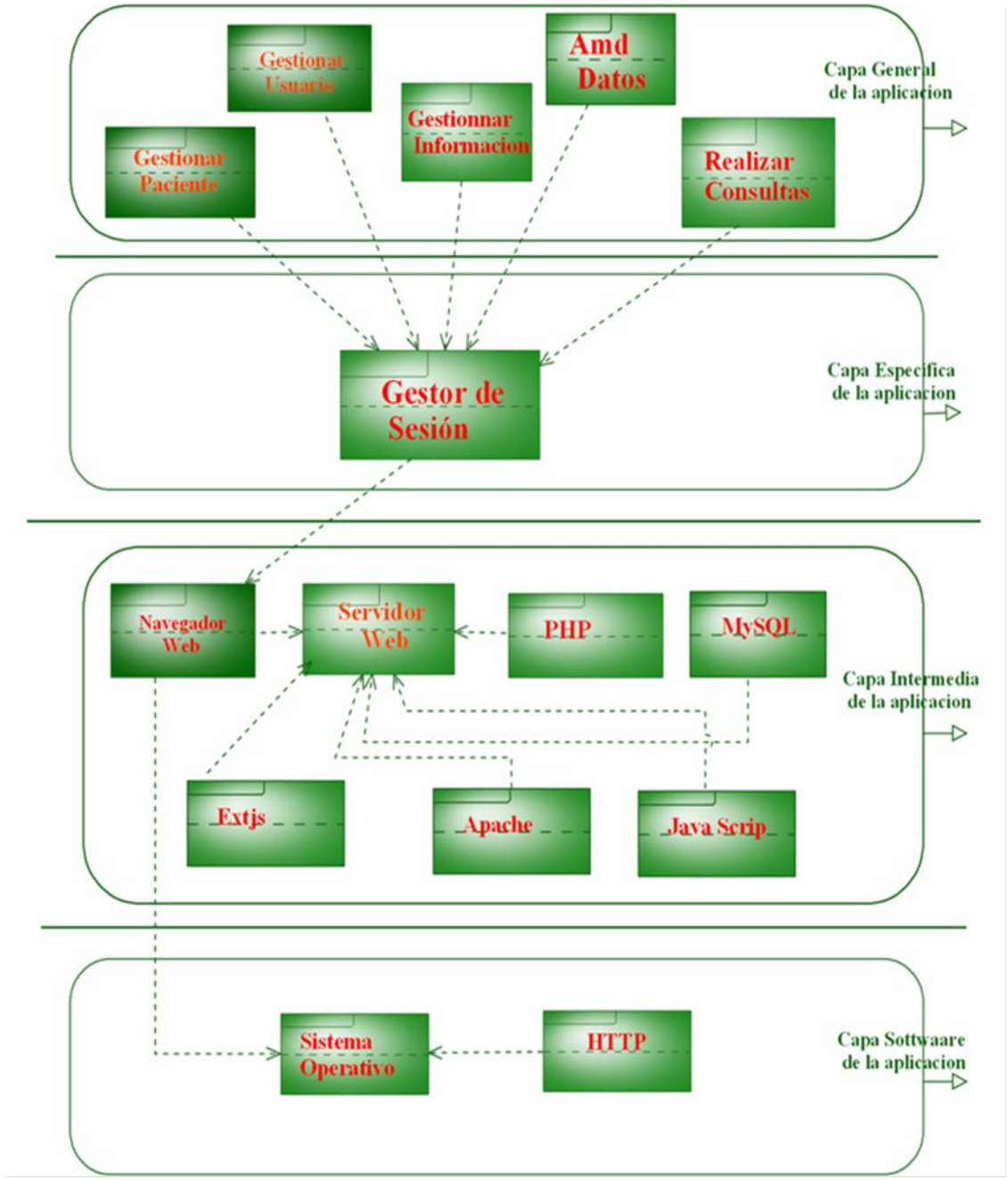


Figura 4.11: Arquitectura del Sistema S.A.C.M.O.
Fuente: elaboración propia

4.3.2 Diseño de la base de datos

Para el diseño de la base de datos su utilizaran el modelo de datos de la especificación de WEBML y la descripción de las tablas de la base de datos.

Conociendo el volumen de datos que podrá manejar el sistema SACMO, se determinó el diseño de una Base de Datos relacional Base de dato SACMO para el control y gestión de los datos.

Una base de datos relacional es un conjunto de dos o más tablas estructuradas en registros (líneas) y campos (columnas), que se vinculan entre sí por un campo en común, en ambos casos posee las mismas características como por ejemplo el nombre de campo, tipo y longitud; a este campo generalmente se le denomina ID, identificador o clave.

Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente.

Para el diseño de una Base de Datos relacional tienen que tomarse en cuenta las siguientes leyes básicas:

1. Una tabla sólo contiene un número fijo de campos.

2. El nombre de los campos de una tabla es distinto.

3. Cada registro de la tabla es único.

4. El orden de los registros y de los campos no está determinados.

5. Para cada campo existe un conjunto de valores posible.

4.3.2.1 Modelo de datos

El modelo de datos de WebML es una adaptación de los modelos conceptuales de diseño de base datos, es compatible con el modelo de datos Entidad – Relación usado en el

diseño conceptual de bases de datos, y también es compatible con los diagramas de clases UML empleados en el modelado orientado a objetos.

Esta rama de Ingeniería Web direcciona las ediciones de específica y está relacionada con el diseño y el desarrollo de Usos de Web. Particularmente, se centra en las notaciones del diseño y los usos visuales de los modelos que se pueden utilizar para la realización de un robusto, bien-estructurado, usables y conservables del Web. Diseñar un Web site dato-intensivo asciende a especificar sus características en términos de varias abstracciones relacional. Los modelos relacionales principales que están implicados en diseño complejo del uso del Web son: estructura de datos, composición contenta, trayectorias de la navegación, y modelo de la presentación.

El modelo de datos define la estructura de la base de datos. En la **¡Error! No se encuentra el origen de la referencia.** pueden verse los atributos de cada entidad así como su relación con otras entidades.

4.3.2.2 Descripción de la Base de Datos

Tomando el modelo de datos y realizando la normalización de las entidades, se pudo diseñar la base de datos del sistema, utilizando el administrador de base de datos MySQL como herramienta

La base de datos del sistema SACMO (denominada “Base de dato SACMO”) está formada por un conjunto de trece (13) tablas que serán descritas a continuación.

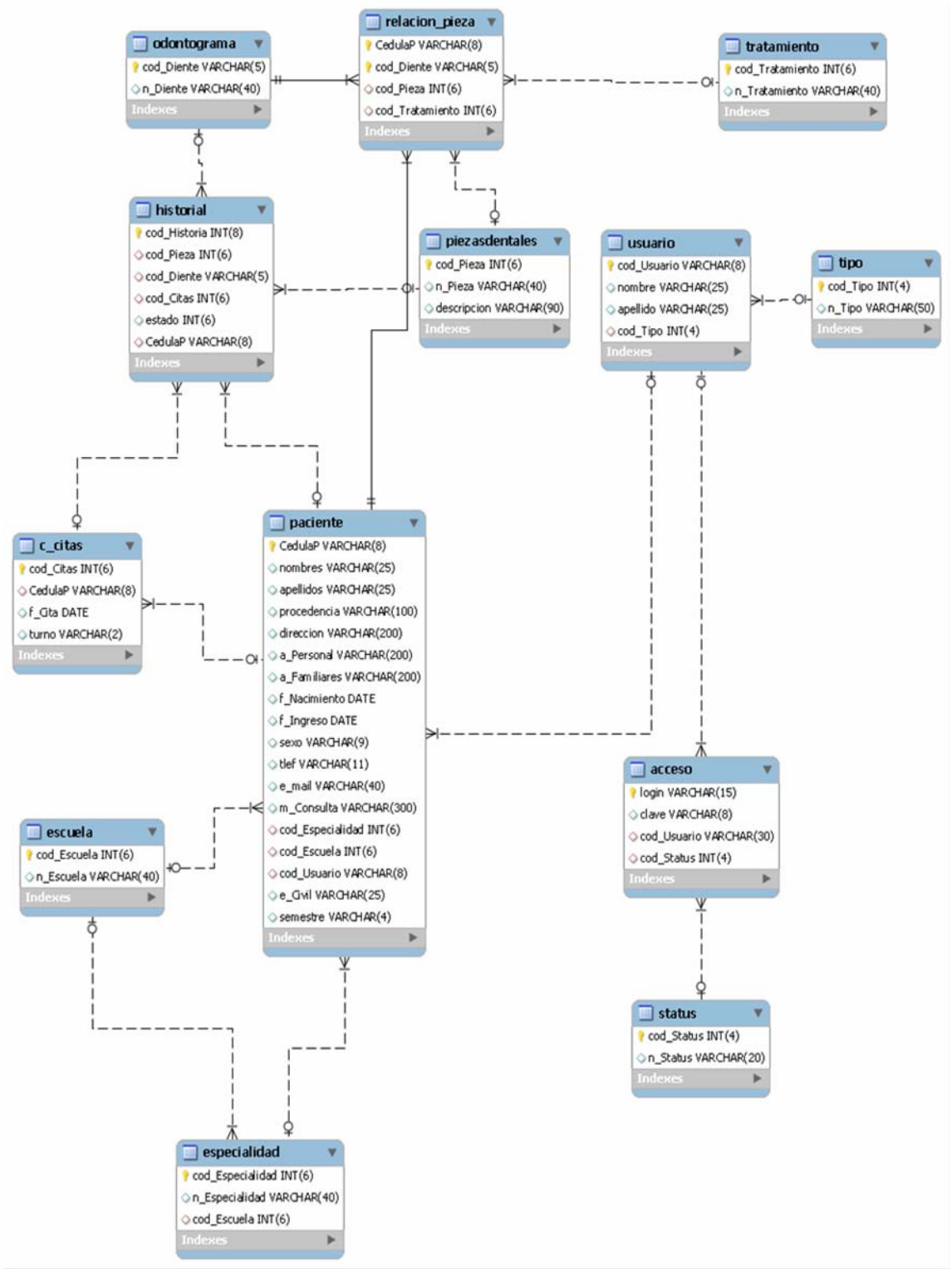


Figura 4.12: Relación de tablas sistema SACMO.
Fuente: Elaboración Propia.

Tabla Acceso de Usuarios: Contiene el cargo y el tipo de acceso que tendrá el usuario al ingresar al sistema, estos datos se usan para lograr que el usuario entre solo a los módulos asignados. Se muestra en la **¡Error! No se encuentra el origen de la referencia.**

Figura 4.13: Tabla Acceso de Usuarios.
Fuente: Elaboración Propia.

✓ Tabla control de pacientes

Contiene la información relevante al estado actual de un paciente y para facilitar el control médico. Ver: Figura 4.14

Figura 4.14: Tabla Control de Paciente.

Fuente.: Elaboración Propia

✓ Tabla Usuarios.

Es la tabla donde se guardan los datos personales de los Doctores que tiene el área de servicios estudiantiles. Ver. Figura 4.15

Figura 4.15.: Tabla Control de Usuarios.
Fuente: Elaboración Propia.

✓ Tabla Especialidad.

Es la tabla donde se guardan los datos de las especialidades con que cuenta la institución Ver: Figura 4.16

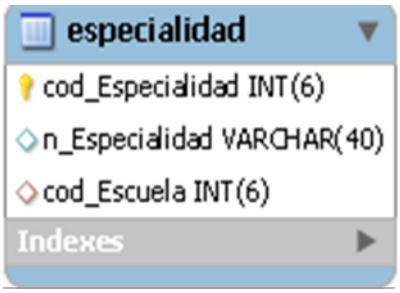


Figura 4.16: Tabla control de Especialidad
Fuente.: Elaboración Propia

✓ Tabla escuela.

Es la tabla donde se guardan las distintas escuela con que cuenta al institución los detalles se muestran en la. Figura 4.17

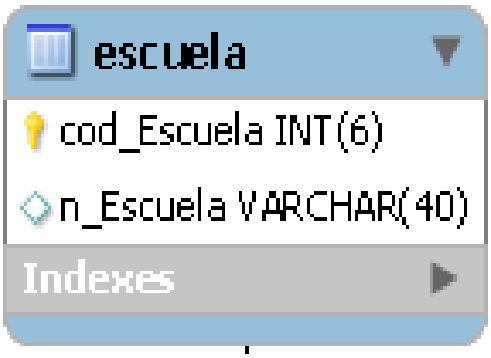


Figura 4.17: Tabla control de las escuelas.
Fuente Elaboración propia.

✓ Tabla control de citas.

Es el tabal donde se guardan todas citas que son otorgadas en el semestre por la unidad de servicios estudiantiles en específico el área de odontología. Los detalles se muestran en la. Figura 4.18

Figura 4.18: Tabla para el control de las citas.
Fuente.: Elaboración Propia.

✓ Tabla historial.

Es la tabla en donde se guarda el historial de los tratamientos realizados en el centro de servicios estudiantiles en el área de odontología. Los detalles se muestran en la. Figura 4.19

Figura 4.19.: Tabla para el Control del Historial.
Fuente.: Elaboración propia.

✓ Tabla Relación de Piezas

Es la tabla en donde se guarda se guarda la relación que tienen los dientes con los de la boca de la persona los detalles se muestran en la.

Figura 4.20: Tabla para el control de relación piezas.
Fuente: Elaboración Propia.

✓ **Tabla Odontograma.**

Es la tabla en donde se guardan todas posiciones de los dientes y los tratamientos realizados y por realizar los detalles se muestran en la. Figura 4.21

Figura 4.21: Tabla para el control del Odontograma.
Fuente: Elaboración Propia.

✓ **Tabla tratamiento.**

Es la tabla en donde se guardan todos los tratamientos realizados a los estudiantes en el centro de bienestar estudiantil en el área de odontología. Los detalles se muestran en la. Figura 4.22

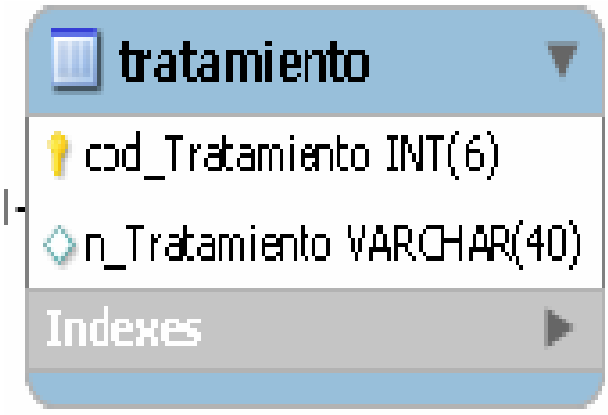


Figura 4.22.: Tabla para el Control de los tratamientos.
Fuente: Elaboración Propia.

✓ **Tabla Pieza dental.**

Es la tabla en donde se guardan la relación de piezas dentales y su descripciónFigura 4.23

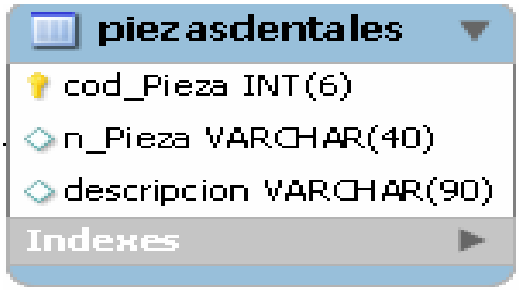


Figura 4.23: Tabla para el control de las piezas dentales.
Fuente.. Elaboración Propia.

4.3.3 Diseño de Hipertextos

El Modelo de Hipertexto describe qué páginas y contenidos componen el sitio y cómo están enlazadas a través de dos submodelos: el Modelo de Composición y el Modelo de Navegación. Definiciones. El hipertexto es una tecnología que organiza una base de información en bloques distintos de contenidos, conectados a través de una serie de enlaces cuya activación o selección provoca la recuperación de información (Díaz et al, 1996].)

El hipertexto ha sido definido como un enfoque para manejar y organizar información, en el cual los datos se almacenan en una red de nodos conectados por enlaces. Los nodos contienen textos y si contienen además gráficos, imágenes, audio, animaciones y video, así como código ejecutable u otra forma de datos se les da el nombre de hipermedio, es decir, una generalización de hipertexto. Es la expresión de una realidad o sistema complejo mediante algún lenguaje formal o simbolismo gráfico que facilita su comprensión y el estudio de su comportamiento. En la Figura 4.24 se muestra el modelo de Hipertexto del sistema SACMO.

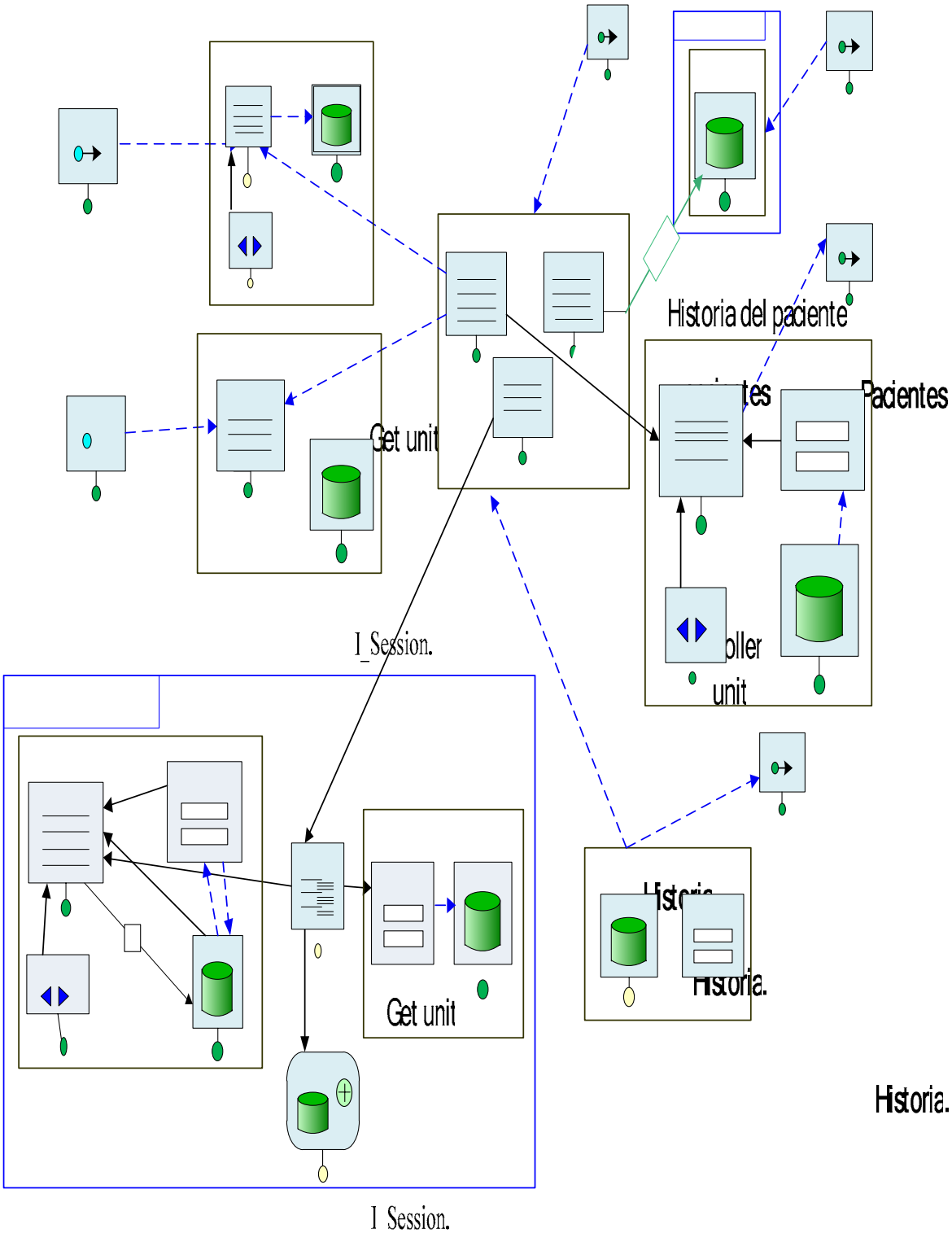


Figura 4.24: Modelo de Hipertexto de la aplicación SACMO
Fuente: Elaboración Propia.

4.3.4 Modelo de gestión de contenidos

Las aplicaciones web realizan con frecuencia operaciones en los datos, ejemplo de ello son las adiciones de elementos en un carrito de compra virtual o la actualización de contenidos publicados en la web. En todos estos casos, las acciones realizadas a través de la interfaz web tienen efectos secundarios como el de cambiar el contenido de algunos datos fuentes conectados al sitio web.

Además de actualizar los datos, las aplicaciones web pueden invocar programas definidos externamente, dotándoles de entradas que dependen del contenido de la página actual o de las selecciones de los usuarios, ejemplo de esto son las operaciones de acceso de un usuario, el envío de correos electrónicos y así sucesivamente

El modelo de Gestión de Contenidos (MGC) viene a ser una extensión del modelo de Hipertexto, que se constituye en dos ampliaciones: La primera añade unidades para la manipulación de datos y unidades para la ejecución de servicios externos y la segunda se refiere a los enlaces salientes de las unidades de operaciones (OK-KO) para la captura del éxito o fracaso de las operaciones permitiendo al diseñador plasmar cursos alternativos dependiendo de los resultados de las operaciones.

A continuación se presenta el modelo de Gestión de Contenidos de la Aplicación SACMO. La cual está centrada a los casos de uso Consultar datos de Pacientes, Cargar datos de Pacientes, Actualizar datos de Historia, Actualizar datos de Odontograma.

La personalización tiene tres factores que el diseñador debe tomar en cuenta a la hora de llevarla a cabo durante el proceso de modelado de la aplicación. Es importante destacar que la personalización no es más que la definición de los filtros a los que la información será sometida para mantener su integridad y seguridad dentro del sistema según los usuarios que tengan acceso a ella. Ver Figura 4.25

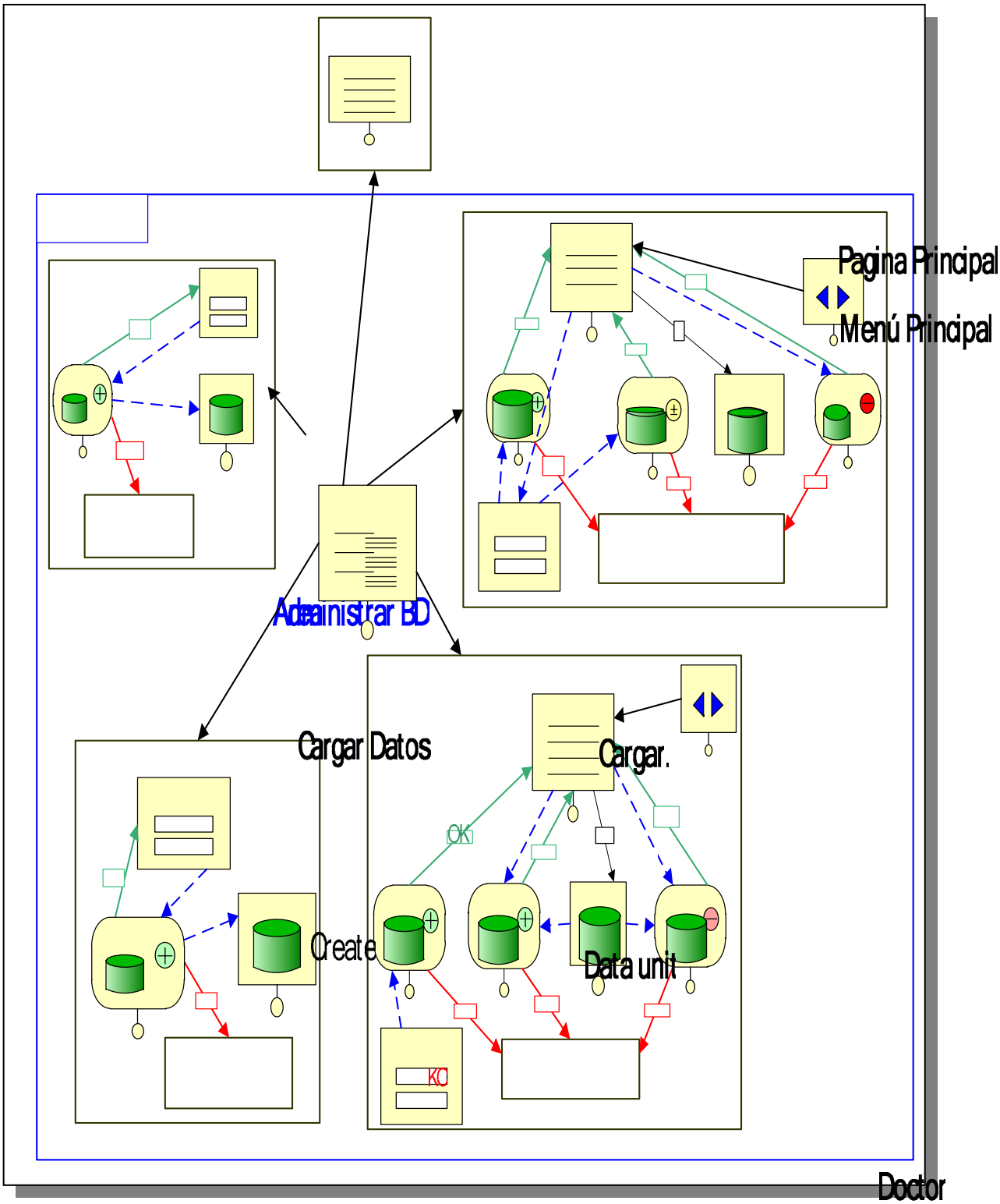


Figura 4.25: Modelo de gestión de Contenido de la aplicación SACMO

Fuente: Selección Propia

4.3.5 Modelo de personalización

La personalización tiene tres factores que el diseñador debe tomar en cuenta a la hora de llevarla a cabo durante el proceso de modelado de la aplicación. Es importante destacar que la personalización no es más que la definición de los filtros a los que la información será sometida para mantener su integridad y seguridad dentro del sistema según los usuarios que tengan acceso a ella. Los factores antes mencionados serán descritos a continuación.

Control de Acceso: se refiere a la implementación de claves de inicio y cierre de sesión por usuario para de esta manera delimitar las operaciones correspondientes y permitidas para cada usuario.

Vistas del Sitio Asignadas: dependiendo del grupo al que el usuario pertenece, algunas vistas del sitio son accesibles, una o más vistas del sitio por grupo.

Personalización de Páginas: el contenido de las páginas depende del usuario o del grupo al cual pertenecen.

Claves de Inicio y Cierre de Sesión para los usuarios

Un sitio puede contener una página que permita al usuario conectarse a un área restringida del sistema mediante un inicio de sesión. Cada sitio por seguridad debe permitir el cierre de sesión para los usuarios, de esta forma se evita la entrada de usuarios no autorizados. En caso de que el usuario pertenezca a más de un grupo se debe incluir la opción de cambio de grupo sin necesidad de salir del sistema.

En el proceso de modelado de la operación de inicio de sesión (login), cierre de sesión (logout) y cambio de grupo de usuario, se representan mediante símbolos los elementos que permiten la ejecución de cada una de estas actividades en el sistema

(elementos de validación de usuarios) con el fin de proporcionar un mejor entendimiento de ellas. La simbología utilizada se muestra a continuación. (Ver Figura 4.26.

Figura 4.26: Símbolos para la personalización del sistema.
Fuente: Elaboración Propia.

4.3.5.1 Página Ingresar al Sistema (Login)

La página contiene un formulario de nombre y contraseña. El usuario introduce el nombre y la clave y presiona el botón entrar. Si el usuario no tiene una sesión activa, se le da acceso al sistema. En caso de que el nombre y contraseña no coincidan, se regresa a la página de entrada con un mensaje de error. Si el usuario tiene una sesión activa, se le pide que cierre la sesión anterior antes de poder ingresar al sistema. Observe la Figura 4.27

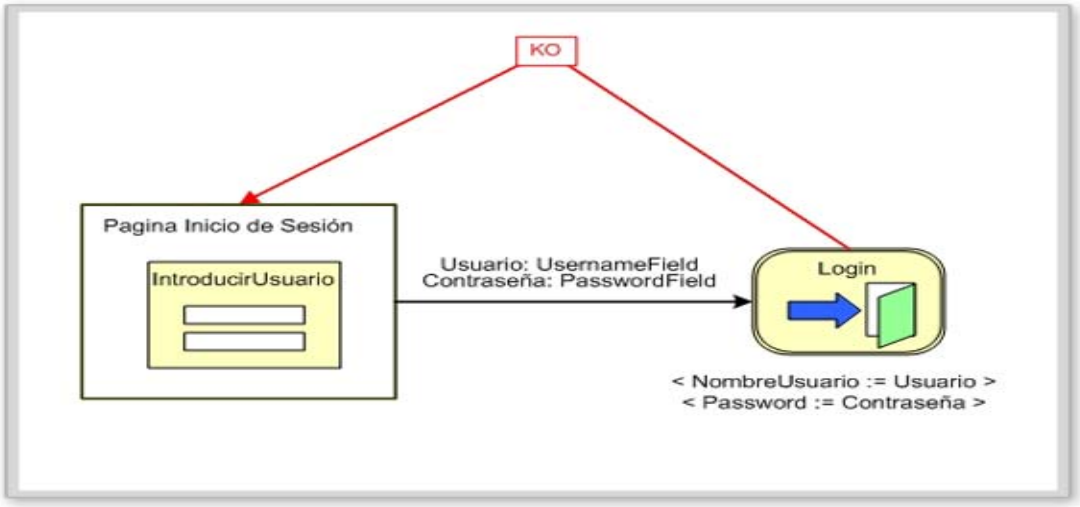


Figura 4.27. Diagrama de gestión de contenido inicio de sesión
Fuente: Elaboración Propia

4.3.5.2 Páginas de agregar datos

Par agregar contenido en el sistema, se utiliza un modelo de gestor de contenido generalizado que consiste en los siguientes pasos:

1. El usuario introduce los datos a ser guardados en el sistema en un formulario de entrada.
2. Al finalizar de agregar los datos, presiona el botón guardar.

3. Una unidad de creación se encarga de grabar los datos en la unidad de datos correspondientes.

4. Otra unidad de creación, del registro del sistema, recibe el nombre de usuario de una variable global de sesión, y lo guarda en el registro del sistema.

5. Si en cualquier momento falla la ejecución, se regresa a la misma página con un mensaje de error, por el contrario, si se ejecuta sin problemas, se regresa a la misma página con un mensaje de éxito.

A continuación en la Figura 4.28 se muestra una serie de diagramas de gestión de contenido de las páginas que usan este modelo generalizado de agregar contenido al sistema.

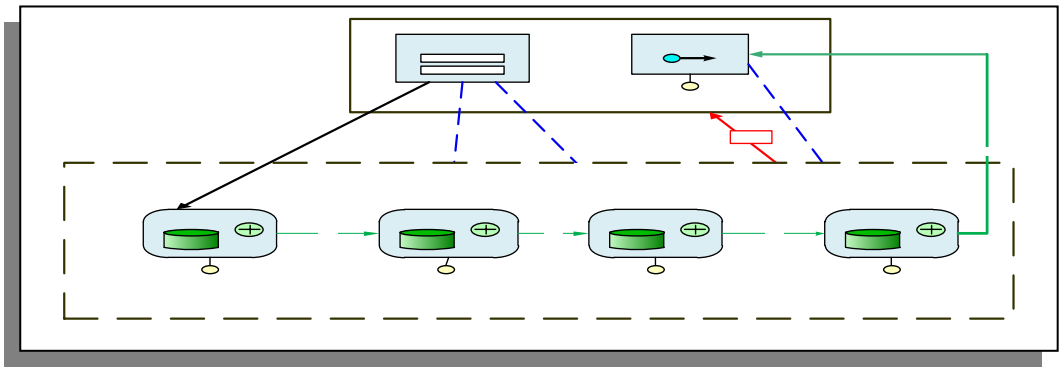


Figura 4.28: Diagrama de gestión de contenido agregar Paciente
Fuente: Elaboración Propia

4.3.5.3 Páginas de modificar datos

Para modificar los datos del sistema, se diseñó un modelo de gestor de contenido muy similar al de agregar los datos. Consiste en los siguientes pasos:

1. El usuario llena el formulario de búsqueda donde se localizara el registro a ser modificado.

2. El usuario presiona el botón aceptar al finalizar las modificaciones.

3. Una unidad de modificación se encarga de realizar el cambio de la información en la unidad de datos adecuada.

4. Otra unidad de creación, del registro del sistema, recibe el nombre de usuario de una variable global de sesión, y lo guarda en el registro del sistema.

5. Si en cualquier momento falla la ejecución, se regresa a la misma página con un mensaje de error, por el contrario, si se ejecuta sin problemas, se regresa a la misma página con un mensaje de éxito. Las páginas modificación de datos siguen el esquema representado a través de la siguiente Figura 4.29

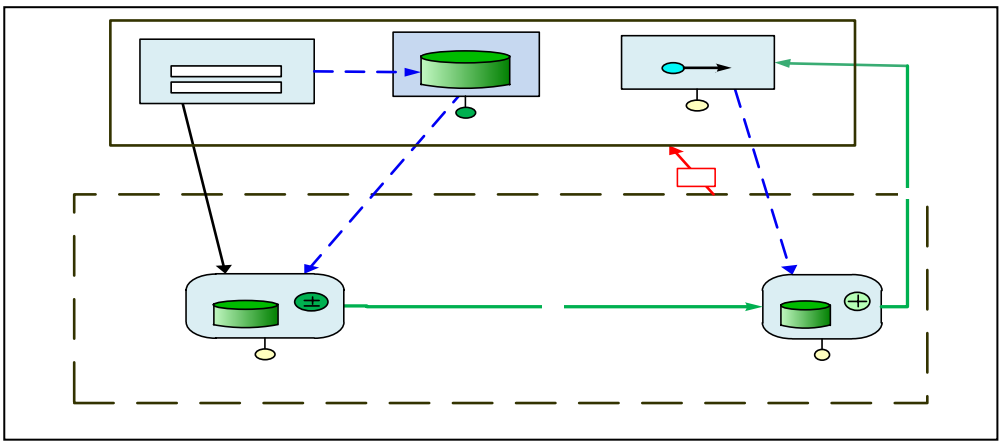


Figura 4.29. Diagrama de gestión de contenido modificar Paciente.
Fuente: Elaboración Propia.

4.3.5.4 Páginas de Ordenes Medicas

Se encargan de procesar las órdenes a los estudiantes, mediante el uso de un oficio del servicio médico familiar correspondiente. Todas funcionan de una manera muy similar que consiste en crear un nuevo registro en la tabla de aprobaciones y a su vez eliminar el registro que mantenía la tabla de solicitudes correspondiente y finalmente crear un registro en el Log de la transacción realizada.

Se le muestra al usuario la lista de solicitudes realizadas durante el semestre actual, y fácilmente escoja cuales desea mostrar Debe ingresar el número cod de la orden.
Observe la Figura 4.30 donde se muestra algunos diagramas de páginas de aprobación de solicitudes.

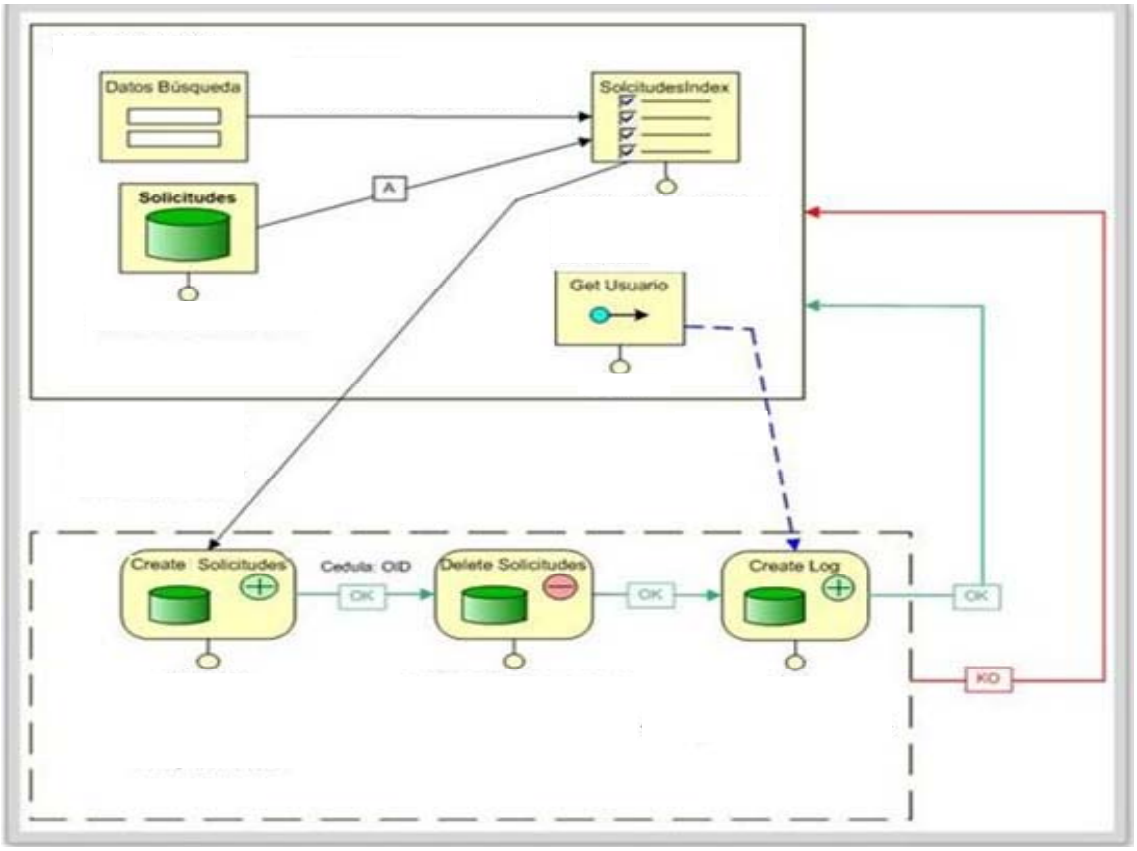


Figura 4.30: Diagrama de gestión de contenido aprobar ordenes médicas
Fuente: Elaboración propia.

Aprobar Ordenes Medicas

Cedula:OID
CodordenesMedicas

4.4 Implementación de la Arquitectura

Basado en la vista de la arquitectura en el modelo de despliegue se identificarán los componentes necesarios para implementar los subsistemas de servicio.

4.4.1 Diagrama de despliegue

El diagrama de despliegue muestra como el usuario va a interactuar con el sistema para obtener la información que desea, ahora la información se le puede mostrar por pantalla entre ellas información personal de los Pacientes, Tratamientos, Solvencias, etc., el usuario también puede realizar operaciones de mantenimiento a los datos (agregar, modificar, eliminar información), obtener reportes impresos, etc. Ver Figura 4.31

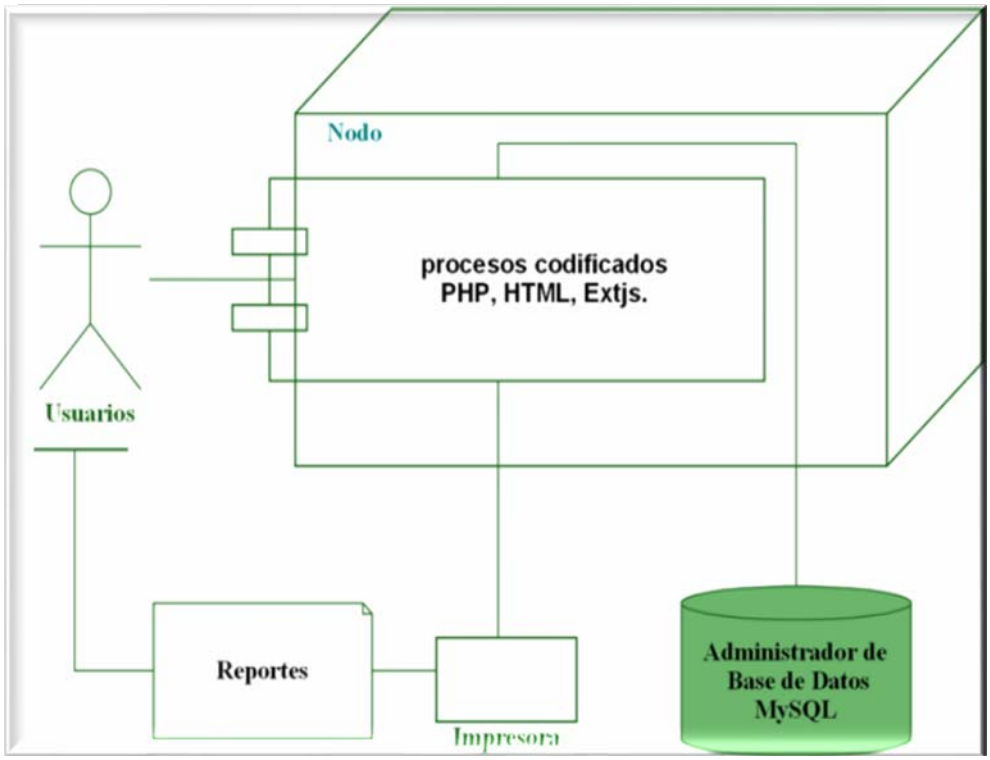


Figura 4.31: Diagrama de despliegue del sistema
Fuente: Elaboración Propia.

4.4.1.1 Identificación de los Componentes de la Arquitectura

Existe un servidor donde se alojan un conjunto de páginas Web que representan el sistema como tal, y que se ejecutarán a través del motor de scripts PHP, y serán

distribuidas por la red mediante el servidor Web Apache por el protocolo TCP/IP. Adicional a estos componentes existe el servidor de base de datos MYSQL que es el encargado de manejar las transacciones que solicite el cliente a través del navegador Web.

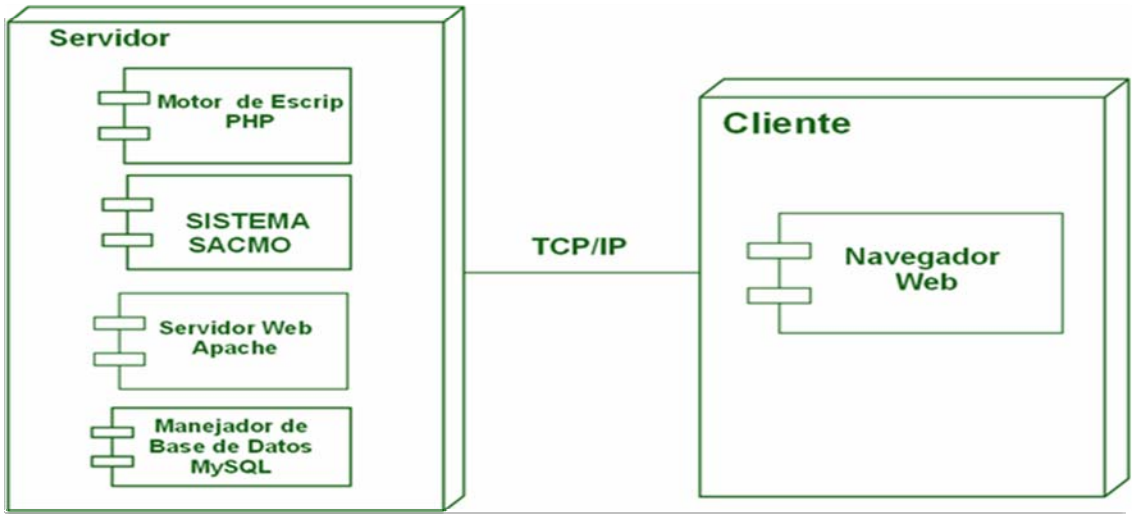


Figura 4.32: Diagrama Componentes de la arquitectura
Fuente: Elaboración Propia

4.4.2 Implementación de la arquitectura

En esta fase se implementa la arquitectura a través de la construcción de un prototipo de la página principal de inicio de sesión denominada index, la cual se puede ver en la Figura 4.33 la cual conforma el caso de uso Acceder, y se ejecutan los componentes que conforman esta arquitectura. Esta página está realizada utilizando HTML, mostrando la interfaz de inicio de sesión y conteniendo las instrucciones necesarias para conectar y consultar la base de datos.

Código Fuente: Pantalla de inicio de sesión

Figura 4.33: Pantalla de inicio de sesión Sistema SACMO.
Fuente: Elaboración Propia.

```
*
 * ================ Función para el Login de usuario :::::::::::::::::::::::: */
  var winLogin;
                    // Cajas de Textos Usuarios y Login
          var txtUsuario = new Ext.form.TextField({
                    name: 'usr',
                    hideLabel: true,
                    width: 180,
                    x: 70,
                    y: 5,
                    allowBlank: false,
                    blankText: 'EL LOGIN ES REQUERIDO.',
```

```
        enableKeyEvents: false,
        selectOnFocus: true,
        listeners: {
            keypress: function(t,e){
                if(e.getKey()==13){
                    txtClave.focus();
                }
            }
        }
});

var txtClave = new Ext.form.TextField({
            name: 'clave',
            hideLabel: true,
            inputType:'password',
            width: 180,
            x: 70,
            y: 35,
            allowBlank: false,
            blankText: 'EL PASSWORD ES REQUERIDO.',
            enableKeyEvents: true,
            selectOnFocus: true,
            listeners: {
                keypress: function(t,e){
                    if(e.getKey()==13){
                        btnAceptar.focus();
                    }
                }
            }
});
```

```
// Labels
var lblUsuario = new Ext.form.Label({
        text: 'Login :',
        x: 10,
        y: 10,
        height: 20,
        cls: 'x-label'
});

var lblClave = new Ext.form.Label({
        text: 'Password :',
        x: 10,
        y: 40,
        height: 20,
        cls: 'x-label'
});

// botones
var btnAceptar = new Ext.Button({
        id: 'btnAceptar',
        x: 85,
        y: 75,
        text: 'Aceptar',
        icon: 'entrar.png',
        iconCls: 'x-btn-text-icon',
        minWidth: 80,
        handler:function(){
                frmLogin.validarAcceso();
        }
```

```
});

var btnLimpiar = new Ext.Button({
        id: 'btnLimpiar',
        x: 170,
        y: 75,
        text: 'Limpiar',
        icon: 'limpiar.png',
        iconCls: 'x-btn-text-icon',
        minWidth: 80,
        handler:function(){
                var frm = frmLogin.getForm();
                frm.reset();
                frm.clearInvalid();
                txtUsuario.focus(true, 100);
        }
});

var frmLogin = new Ext.FormPanel({
        frame:true,
        layout: 'absolute',
        items:[lblUsuario, lblClave, txtUsuario, txtClave, btnAceptar,
btnLimpiar],
        validarAcceso: function(){
                if (this.getForm().isValid()) {
                        this.getForm().submit({
                                url: 'php/pruebasPHP.php',
                                method: 'POST',
                                params:{funcion:'userLogIn'},
                                waitTitle: 'Conectando',
```

```
                                        waitMsg: 'Validando login..',
                                        success: function(form, action){
                                                window.location =
'../Medico_Odontologico/formas/Menu.html';
                                        },
                                        failure: function(form, action){
                                                if (action.failureType == 'server') {
                                                        var data =
Ext.util.JSON.decode(action.response.responseText);
                                                        Ext.Msg.alert('Conexión Fallida',
data.error, function(){
                                                                txtUsuario.focus(true, 100);
                                                        });
                                                }
                                                else {
                                                        Ext.Msg.alert('Error!', 'El servidor de
autenticacion es inalcanzable : ' + action.response.responseText);
                                                }
                                                frmLogin.getForm().reset();
                                        }
                                });
                        }
                }
        });
    //Funcion Para abrir la forma del login
        function abrirLogin(){
                if (!winLogin) {
                        winLogin = new Ext.Window({
                                layout: 'fit',
                                title: 'Login',
```

```
                iconCls: 'inicio',
                width: 280,
                height: 150,
                resizable: false,
                closeAction: 'hide',
                closable: false,
                draggable: false,
                plain: true,
                border: false,
                modal: true,
                items: [frmLogin],
                listeners: {
                        hide: function(){
                                var frm = frmLogin.getForm();
                                frm.reset();
                                frm.clearInvalid();
                        },
                        show: function(){
                                txtUsuario.focus(true, 300);
                        }
                }
        });
    }

    winLogin.show();
}

Ext.onReady(function(){

    Ext.BLANK_IMAGE_URL = 'ext-3.1.1/imagenes/s.gif';
```

```
            Ext.QuickTips.init();
            abrirLogin();
        //Evento onclick del boton Abrir login.
    });
```

Como prueba se introdujeron los datos de usuario Luisa y contraseña 123456, la cual no está contenida en la tabla usuarios obteniendo el resultado de la Figura 4.34

Figura 4.34Página index.html usuario no válido.
Fuente: Elaboración Propia.

Con esto se pudo comprobar el funcionamiento de la arquitectura, ejecutándose los componentes que la conforman, el servidor web, el motor de scripts PHP y el manejador de base de datos MySQL, así como el servidor web que permite la interacción entre el cliente y el servidor utilizando el protocolo de Internet http

4.4.3 Evaluación de la Fase de Elaboración

La fase de elaboración fue muy productiva para el sistema. Se obtuvo la arquitectura definitiva que sustentará el sistema. Se considera que se determinaron mayoría de los requisitos mediante el modelo de casos de uso. Se requirió de una sola iteración para completar los objetivos planteados al inicio de la fase.

Para efectos de diseño, ya se posee una clara visión del sistema y de su contexto. Se logró hacer un diseño efectivo de las páginas web que conformarán el sistema mediante el uso del modelo de hipertexto y el modelo de gestión de contenido de WebML.

En conclusión, se considera que ya se han obtenido todos los componentes necesarios para implementar el sistema en la siguiente fase.

CAPÍTULO V

FASE DE CONSTRUCCIÓN

5.1 . Introducción

La fase de construcción del Proceso Unificado desarrolla o adquiere loa componentes del software que harán que cada caso de uso sea operativo para los usuarios finales. Lograr esto requiere que los modelos de análisis y diseño, realizados durante la fase de elaboración, reflejen la versión final del incremento del software.

5.2 Flujo de trabajo implementación

El principal objetivo de esta fase es obtener una solución de software operativa, satisfaciendo todos los requerimientos planteados en las fases anteriores, ofreciendo una alta calidad funcional y practicidad. Esta versión del software que se obtendrá se cataloga como “versión beta”. Una versión lista para ser entregada a los usuarios finales de la aplicación, en este caso, .a los usuarios empleados de la unidad de servicios estudiantiles de al universidad de oriente núcleo Anzoátegui.

Para lograr los objetivos planteados en esta fase, se realizará la codificación de las páginas web que se diseñaron en el modelo de hipertexto de WebML. La implementación de estas páginas implica también la construcción visual de las mismas, deben tener un aspecto elegante y un esquema de navegabilidad practico, siguiendo el prototipo de interfaz planteado en la fase de elaboración.

5.3 Implementación

El Proceso Unificado define el Modelo de Implementación como aquel que contiene los artefactos de implementación como el código fuente, las páginas web de servidor, entre otros. Por lo tanto, el código que se va a crear en esta fase forma parte del modelo de implementación.

Para la codificación de las páginas se usará el lenguaje de programación PHP a través del ambiente de desarrollo que ofrece EXTJS 3.1.1. Las operaciones de base datos serán gestionadas por el administrador de base datos MYSQL. Se usará MySQL Workbench 5.2 CE para crear la estructura de datos de MySQL .La integración de todos estos elementos formará la arquitectura de trabajo estable que se diseñó en la fase anterior.

5.3.1 Implementación del Modelo de Hipertexto

5.3.1.1 Computación de una Página Dinámica

Comenzaremos la explicación mostrando el típico flujo de trabajo para procesar una página dinámica de un contenido guardado en una base de datos. Las actividades requeridas están esquematizadas en la Figura 5.1

El primer paso, es analizar la solicitud HTTP para extraer los posibles parámetros, típicamente usados por las consultas en la base de datos para sacar el contenido de la página. En el segundo paso, la conexión a la base de datos es establecida y las consultas para extraer el contenido necesitado para llenar la página son ensambladas y ejecutadas. En la mayoría de los casos, las consultas tienen una estructura fija y requieren solo los parámetros de entrada; en unos pocos casos, el código fuente de la consulta debe ser ensamblado en tiempo de ejecución, justo antes de ejecutar la consulta. Luego que la consulta es enviada a la base de datos, sus resultados son

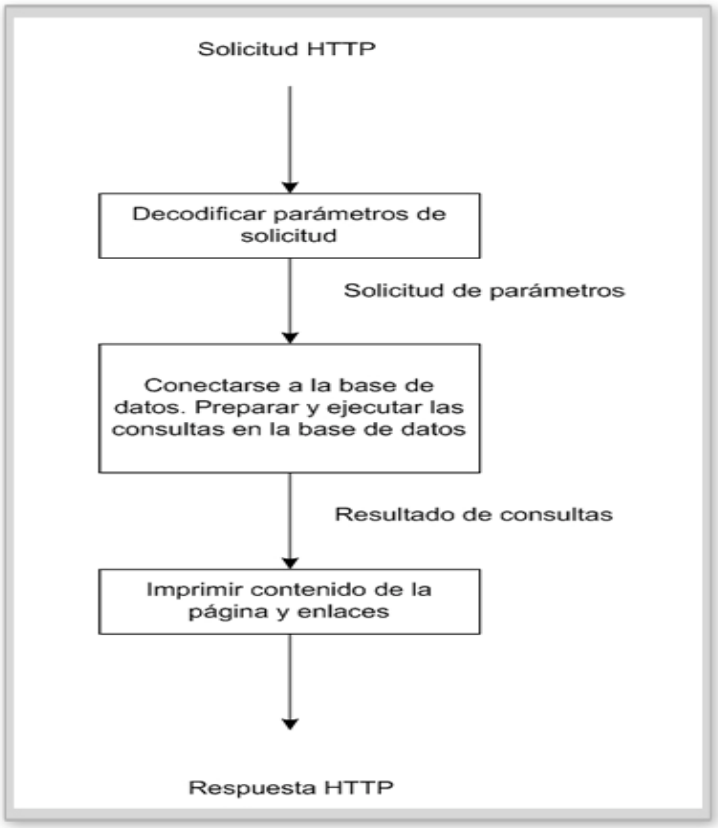


Figura 5.1: Procesamiento de una página con contenido desde una base de datos
Fuente: Elaboración Propia.

El primer paso, es analizar la solicitud HTTP para extraer los posibles parámetros, típicamente usados por las consultas en la base de datos para sacar el contenido de la página. En el segundo paso, la conexión a la base de datos es establecida y las consultas para extraer el contenido necesitado para llenar la página son ensambladas y ejecutadas. En la mayoría de los casos, las consultas tienen una estructura fija y requieren solo los parámetros de entrada; en unos pocos casos, el código fuente de la consulta debe ser ensamblado en tiempo de ejecución, justo antes de ejecutar la consulta. Luego que la consulta es enviada a la base de datos, sus resultados son.

Transferidos a las estructuras de datos y pueden ser usadas para determinar el valor de los parámetros usados en otras consultas de extracción de contenido. Por lo tanto, la ejecución de consultas es iterada hasta que todas las consultas necesarias para

retirar el contenido de la página han sido procesadas. Finalmente en el último paso, cuando todas las piezas del contenido necesario para construir la página HTML han sido extraídas, la página es producida y retornada como resultado de la petición HTTP. Específicamente, los resultados son usados para construir la parte dinámica de la página, la cual consiste típicamente de contenido (texto, imágenes, etc), y enlaces, expresados como etiquetas de enlace HTML.

La parte esencial del procesamiento de las páginas son la construcción y ejecución de las consultas de retiro de información y la producción del código HTML. Cada unidad en la página tiene sus propias reglas para retirar el contenido y producir el código HTML, y cuando la página contiene múltiples unidades enlazadas, el orden en el cual las consultas son ejecutadas es importante, porque una unidad puede requerir entrada de otras unidades. Es en la solución de este tipo de problemas que la especificación WebML de las páginas ayuda: WebML clasifica los elementos de contenido que pueden aparecer en la página en categorías bien definidas, correspondientes a las diferentes tipo de unidades, y establece reglas bien definidas para el orden en el cual las unidades son procesadas, representado por el procedimiento de procesamiento de alto nivel de páginas. Por lo tanto, el esquema general de actividades ilustrado en la figura 5.1, puede ser especializado al caso de una página WebML, consistiendo en muchas unidades enlazadas, para obtener un esquema general de implementación como el descrito a continuación.

Parte 1: Extraer parámetros de la petición HTTP.

Parte 2: Conectar a la base de datos.

Parte 3: Preparar y ejecutar las consultas:

Construir la declaración de la consulta de la unidad actual.
Ejecutar la consulta.

Si existen unidades dependientes, enlazar la salida de la unidad actual a la entrada de las unidades dependientes y repetir los pasos 1-3.

Parte 4: Procesar el contenido de la página dinámica:

Construir el marco HTML para renderizar la unidad actual, del resultado de la consulta asociada a él. Construir los enlaces de salida de la unidad:

Construyendo la parte fija del URL.
Construyendo los parámetros asociados con el enlace.

Parte 5: Eliminar recursos temporales

La primera parte del esquema general implementa la asignación inicial de los valores de los parámetros a las unidades de las páginas. La navegación de un enlace por el usuario resulta en una solicitud HTTP, posiblemente conteniendo los valores de los parámetros necesitados para inicializar

Las unidades de la página. Estos parámetros representan valores “frescos” producidos por la navegación de los enlaces, usados para obtener nuevo contenido para algunas unidades, o valores “preservados” usados para “recordar” escogencias pasadas hechas por el usuario en navegaciones previas o enlaces entre páginas. Desde un punto de vista técnico, la extracción de parámetros del llamado debe arreglarse con las diferentes formas de codificar los parámetros en las llamadas HTTP.

La segunda parte de la plantilla de las páginas, direcciona la conexión a la base de datos, preliminar a la ejecución de las consultas de retiro de datos necesarias para obtener el contenido de las unidades de la página. Esta es una simple tarea técnica, la cual requiere el conocimiento de la interfaz del lenguaje de programación a la base de datos.

La parte 3 es el núcleo de la plantilla, la cual embebe el procesamiento semántico de la página. Su meta es procesar todas las unidades procesables de la página, tomando en cuenta el hecho de que la página puede ser accedida a través de diferentes enlaces, los cuales corresponden a diferentes parámetros en la petición HTTP.

Finalmente, la última sección de la plantilla simplemente elimina los objetos temporales usados en las fases previas.

5.3.1.2 Codificación de las Páginas Web

Las páginas web que han sido diseñadas hasta ahora se deben implementar en código fuente a través del lenguaje de programación **PHP**. El código fuente obtenido será considerado como componente de software representado por medio de un archivo de extensión ".php".

Para lograr una codificación efectiva e incrementar la calidad de producción de páginas se elaboraron una serie de componentes que representan librerías reutilizables en toda la aplicación. Se utilizó librerías **extjs3.1.1** que suministran las funciones de validación, características gráficas y controles prácticos durante la ejecución de la aplicación en el explorador web. También se crearon librerías de **PHP**, que contienen funciones de validación interna de la aplicación, consultas genéricas a la base de datos de **MYSQL**, control de sesión de usuarios, ejecución de procedimientos almacenados que evitan la inyección de SQL, y otros.

En el sistema resultante se codificaron más de 50 páginas de servidor. A continuación se muestra un solo ejemplo de código fuente, de la página que gestiona la agregación de un estudiante al sistema.

5.3.2 Página principal del sistema SACMO.

En la Figura 5.2, se observa la página principal de la aplicación SACMO para el tipo de vista "Menu Principal", en donde se Puede apreciar los módulos de Navegación y Administración BD representando estos la funcionalidad total de la aplicación.

Al activar esta página, se muestra un formulario para seleccionar una pantalla en la aplicación que puede estar o no registrado en la base de dato Los datos ingresados son validados a través de funciones de Javascript, y luego se envían a la base de datos para ser almacenados. A continuación se presenta el código fuente de la página

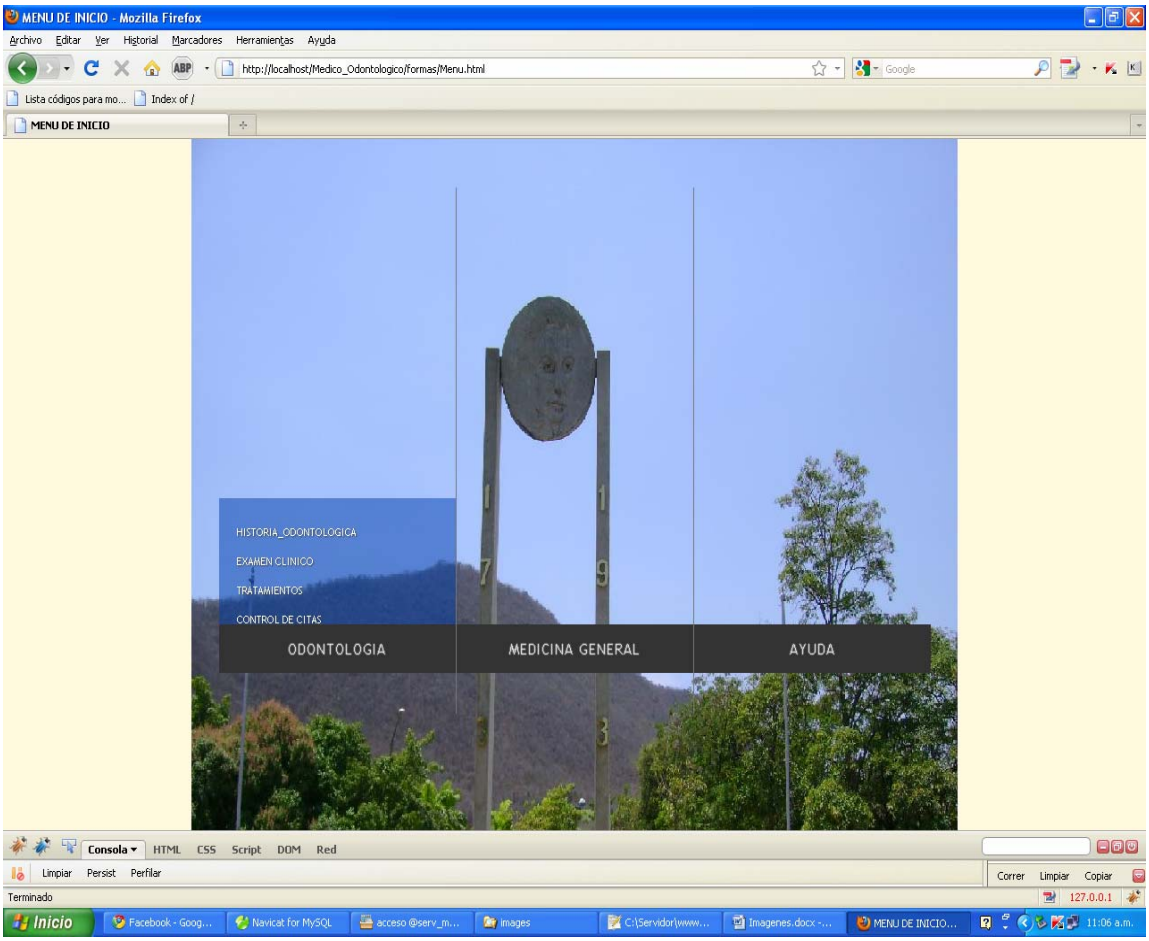


Figura 5.2Pantalla Principal de la aplicación SACMO..
Fuente: Elaboración Propia.

Como se pudo ver en la figura anterior la información pertinente a los datos de los servicios Médicos - Odontológicos prestados por LA UDO, se visualizan en la parte

central de la aplicación, se observa un menú en forma de acordeón, dentro de este se puede visualizar las diferentes opciones que tenemos en el sistema

Código Fuente Pantalla Principal:

```
<html>
  <head>
    <title>MENU DE INICIO </title>
     <style>
       *{
          margin:0;
          padding:0;
       }
       body{
          font-family:Arial;
          padding-top:30px;
          background:#FFF9DF url(Moneda2.jpg) no-repeat top center;
       }
       a.back{

          position:absolute;
          width:150px;
          height:27px;
          outline:none;
          top:2px;
```

```
        right:0px;
    }
    .reference{
        margin:20px auto;
        width:600px;
        padding:20px;
    }
    .reference p a{
        text-transform:uppercase;
        text-shadow:1px 1px 1px #fff;
        color:#666;
        text-decoration:none;
        font-size:10px;

    }
    .reference p a:hover{
        color:#333;
    }
</style>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
<meta name="description" content="MENU DE INICIO" />
<meta name="keywords" content="jquery, background image, animate, menu, navigation, css3, cross-browser compatible"/>
<link rel="stylesheet" href="css/style.css" type="text/css" media="screen"/>
```

```
</head>
<body>
  <div id="content">
    <div id="menuWrapper" class="menuWrapper bg1">
      <ul class="menu" id="menu">
        <li class="bg1" style="background-position:0 0;">
          <a id="bg1" href="#">ODONTOLOGIA</a>
          <ul class="sub1" style="background-position:0 0;">
            <li><a
href="../formas/historiaMedicoOdon.html">Historia_Odontologica</a></li>
            <li><a href="../formas/odontogramas.html">Examen
CLinico</a></li>
                                                <li><a
href="#">tratamientos</a></li>
            <li><a href="#">control de citas</a></li>
          </ul>
        </li>
        <li class="bg1" style="background-position:-266px 0px;">
          <a id="bg2" href="#">MEDICINA GENERAL</a>
          <ul class="sub2" style="background-position:-266px 0;">
            <li><a href="#">Historia_Medica</a></li>
            <li><a href="#">Ordenes Medicas</a></li>
            <li><a href="#">Tratamientos</a></li>
          </ul>
        </li>
        <li class="last bg1" style="background-position:-532px 0px;">
```

```
            <a id="bg3" href="#">AYUDA</a>
            <ul class="sub3" style="background-position:-266px 0;">
               <li><a href="#">Manual De Usuario</a></li>
               <li><a href="#">Quienes Somos</a></li>
               <li><a href="#">Salir</a></li>
            </ul>
         </li>
      </ul>
   </div>

   <div class="reference">

   </div>
</div>

 <script type="text/javascript" src="../js/jquery.min.js"></script>
 <script type="text/javascript" src="../js/jquery.bgpos.js"></script>
            <script type="text/javascript">
   $(function(){
      /*posición de la <li> que actualmente se muestra*/
      var current = 0;
                              var loaded  = 0;
                              for(var i = 1; i <4; ++i)
                                    $('<img />').load(function(){
```

```
                                        ++loaded;
                                        if(loaded == 3){

        $('#bg1,#bg2,#bg3').mouseover(function(e){
                                                    var $this = $(this); /* Si se
pasa el actual, entonces no hace nada */
                                                    if($this.parent().index() ==
current)
                                                        return;
                                                    var item = e.target.id;
        /*el tema es bg1 o BG2 o BG3, dependiendo donde se ciernen*/
                                                    /*
                                                    esta es la superposición de
submenú. Vamos a ocultar el actual
                        si pasa el <li> primera o si venimos de la  última posicin del
Menu,
                        a continuación, la plantalla debe moverse  de izquierda
hacia la -> derecha,
                        y si es al contrario la pantalla se Mueve de  derecha hacia -
> izquierda
                                                     */
                                                    if(item == 'bg1' || current
== 2)
                                                        $('#menu
.sub'+parseInt(current+1)).stop().animate({backgroundPosition:"(-266px
0)"},300,function(){

        $(this).find('li').hide();
                                                    });
```

```
                                                    else

                                                        $('#menu
.sub'+parseInt(current+1)).stop().animate({backgroundPosition:"(266px
0)"},300,function(){


        $(this).find('li').hide();

                                                        });



                                                    if(item == 'bg1' || current
== 2){

                                                        /* if we hover the
first <li> or if we come from the last one, then the images should move left ->
right */

                                                        $('#menu >
li').animate({backgroundPosition:"(-800px 0)"},0).removeClass('bg1 bg2
bg3').addClass(item);

                                                        move(1,item);

                                                    }

                                                    else{

                                                        /* if we hover the
first <li> or if we come from the last one, then the images should move right ->
left */

                                                        $('#menu >
li').animate({backgroundPosition:"(800px 0)"},0).removeClass('bg1 bg2
bg3').addClass(item);

                                                        move(0,item);

                                                    }


                                                    /*
```

```
                                                    Lo que Queremos es que
si vamos desde el primero hasta el último (sin Mover la central),

                    o desde el último al primero, el menú de superposición
medio también debe deslizarse, ya sea

                    de izquierda a derecha o de derecha a izquierda.

                                                     */

                                                    if(current == 2 && item ==
'bg1'){

                                                        $('#menu
.sub'+parseInt(current)).stop().animate({backgroundPosition:"(-266px 0)"},300);

                                                    }

                                                    if(current == 0 && item ==
'bg3'){

                                                        $('#menu
.sub'+parseInt(current+2)).stop().animate({backgroundPosition:"(266px 0)"},300);

                                                    }

                                                    current =
$this.parent().index();/* cambiar el elemento actual */

                    /*vamos a hacer la superposición de la actual aparecen*/

                                                    $('#menu
.sub'+parseInt(current+1)).stop().animate({backgroundPosition:"(0
0)"},300,function(){


        $(this).find('li').fadeIn();

                                                    });

                                                });

                                            }

                                        }).attr('src', 'images/'+i+'.jpg');
```

```
/*
dir:1 - move left->right
dir:0 - move right->left
 */
function move(dir,item){
   if(dir){
      $('#bg1').parent().stop().animate({backgroundPosition:"(0
0)"},200);
      $('#bg2').parent().stop().animate({backgroundPosition:"(-266px
0)"},300);
      $('#bg3').parent().stop().animate({backgroundPosition:"(-532px
0)"},400,function(){
         $('#menuWrapper').removeClass('bg1 bg2
bg3').addClass(item);
      });
   }
   else{
      $('#bg1').parent().stop().animate({backgroundPosition:"(0
0)"},400,function(){
         $('#menuWrapper').removeClass('bg1 bg2
bg3').addClass(item);
      });
      $('#bg2').parent().stop().animate({backgroundPosition:"(-266px
0)"},300);
      $('#bg3').parent().stop().animate({backgroundPosition:"(-532px
0)"},200);
   }
```

```
        }
      });
    </script>
  </body>
</html>
```

5.3.3 Página de cargar Pacientes.

Entre las actividades que conforman el área de Historial de Pacientes se encuentra el ingresar un paciente, Buscar un paciente, actualizar datos, generar Odontograma asociados a cada uno de los pacientes, dependiendo los permisos que tenga el usuario para ingresar a la aplicación Se puede visualizar esta interfaz en la Figura 5.3.

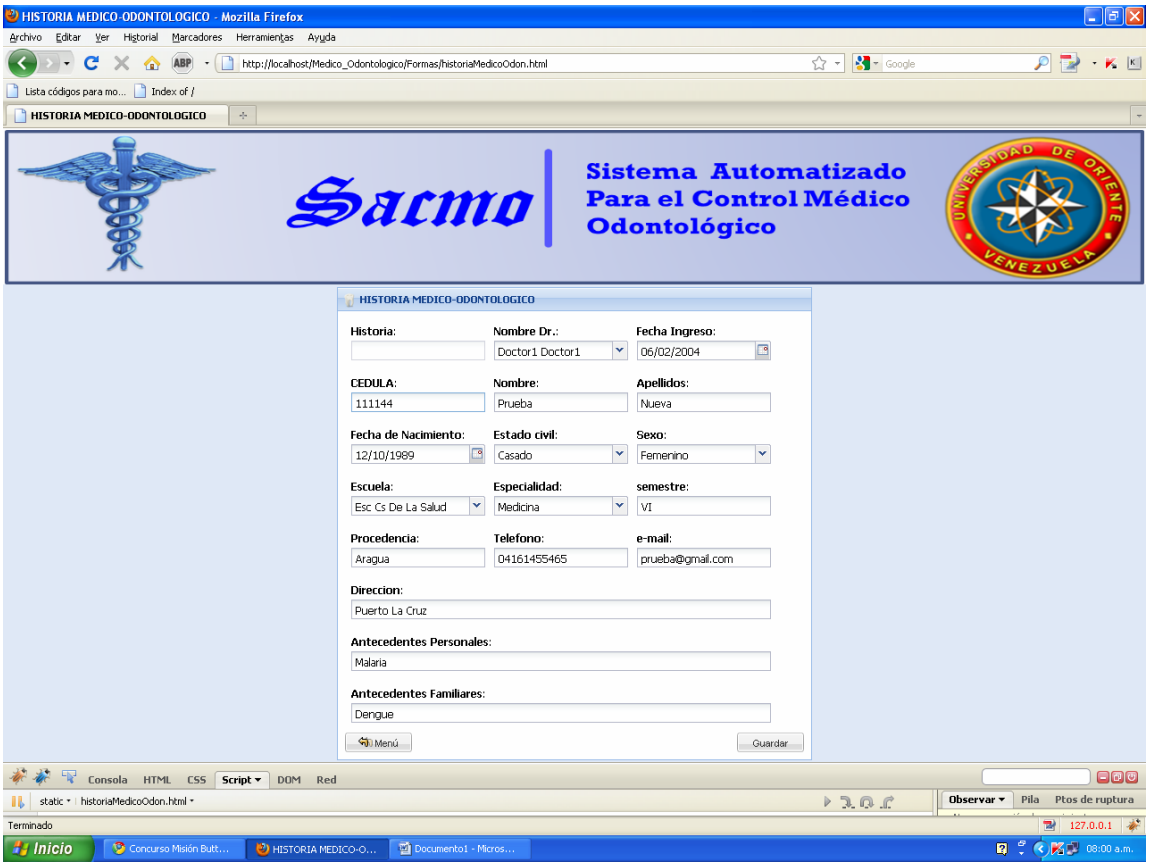


Figura 5.3: Pantalla de Cargar pacientes.
Fuente: Elaboración Propia.

```
// Módulo: Agregar datos de Pacientes » agregar_Paciente.php
//  Descripción:  La  página  presenta  un  formulario  de  datos  para  agregar  a  un
//Paciente    nuevo. Los datos ingresados se guardan en la base de datos.
// Autor:
// Fecha de última modificación: 29/07/2010,
// Fecha de creación: 28/09/2009


Ext.ns("Forma");
Ext.SSL_SECURE_URL  = '../resources/images/default/s.gif';
Ext.BLANK_IMAGE_URL = '../resources/images/default/s.gif';


/*var revisarSesion = function (response, request){
        var dat = Ext.util.JSON.decode(response.responseText);
        if (dat.session == 1)
                Forma.TableLayoutFormulario.init;
        else
                document.location.href='../index.html';
};*/
Forma.TableLayoutFormulario = function(){
var contObj = this;
return{
        init: function(){
                var win = new Ext.Panel({
                        id:"MOdonto",
                        title: "HISTORIA MEDICO-ODONTOLOGICO",
                        width: 532,
                        height: 530,
                        draggable: false,
                        iconCls: "diente",
                        layout: "fit",
```

```
items: [{
        xtype: "form",
        id:"formita",
        url:'../php/pruebasPHP.php',
        title: "",
        labelWidth: 100,
        labelAlign: "left",
        layout: "table",
        border: false,
        buttonAlign : "left",
        height: 500,
        bodyStyle: "padding:10px",
        layoutConfig: {columns: 3},
        items: [{
                xtype: "panel",
                layout: "form",
                border: false,
                labelAlign: "top",
                padding: 5,
                items: [{
                        xtype: "textfield",
                        fieldLabel: "Cod_historia",
                        name:"codhistoria",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                layout: "form",
                border: false,
                labelAlign: "top",
```

```
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "textfield",
                        fieldLabel: "Nombre_Dr",
                        name:"nombreDr",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                layout: "form",
                border: false,
                labelAlign: "top",
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "datefield",
                        name:"fechaIngreso",
                        fieldLabel: "Fecha Ingreso",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                layout: "form",
                border: false,
                labelAlign: "top",
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "numberfield",
```

```
                                        fieldLabel: "CEDULA",
                                        maxLength: 8,
                                        blankText: 'solo 8 numeros',
                                        name:"CedulaP",
                                        anchor: "100%",
                                        listeners: {
                                                scope      : this,
                                                specialkey : function(field, e) {
                                                        if (e.getKey() === e.ENTER)
{
                                                                if   (field.tabIndex   ==
1){

        //Ext.getCmp('pass').focus();

        console.debug(field);
                                                                }else{

        console.debug(field.getValue());

        Ext.getCmp('formita').getForm().load({params:{funcion:'PruebaFormat',CedulaP
:field.getValue()}});
                                                                }
                                                        }
                                                }
                                        }]
                                },{
                                        xtype: "panel",
                                        layout: "form",
```

```
                labelAlign: "top",
                border: false,
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "textfield",
                        fieldLabel:"Nombre",
                        name:"Nombres",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                border: false,
                layout: "form",
                labelAlign: "top",
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "textfield",
                        fieldLabel: "Apellidos",
                        name:"Apellidos",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                layout: "form",
                labelAlign: "top",
                border: false,
                padding: 5,
                defaults: {width: 150},
```

```
                items: [{
                        xtype: "datefield",
                        fieldLabel: "Fecha de Nacimiento",
                        name:"F_Nacimiento",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                layout: "form",
                labelAlign: "top",
                border: false,
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "textfield",
                        fieldLabel: "Edad",
                        name:"Edad",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                layout: "form",
                labelAlign: "top",
                border: false,
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "combo",
                        fieldLabel: "Sexo",
                        name: "Sexo",
```

```
                anchor: "100%"
            }]
        },{
            xtype: "panel",
            layout: "form",
            labelAlign: "top",
            border: false,
            padding: 5,
            defaults: {width: 150},
            items: [{
                xtype: "textfield",
                fieldLabel: "Escuela",
                name:"Escuela",
                anchor: "100%"
            }]
        },{
            xtype: "panel",
            layout: "form",
            labelAlign: "top",
            border: false,
            padding: 5,
            defaults: {width: 150},
            items: [{
                xtype: "textfield",
                fieldLabel: "Especialidad",
                name:"Especialidad",
                anchor: "100%"
            }]
        },{
            xtype: "panel",
```

```
                layout: "form",
                labelAlign: "top",
                border: false,
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "textfield",
                        fieldLabel: "semestre",
                        name:"Semestre",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                layout: "form",
                labelAlign: "top",
                border: false,
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "textfield",
                        fieldLabel: "Procedencia",
                        name:"procedencia",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                layout: "form",
                labelAlign: "top",
                border: false,
                padding: 5,
```

```
                defaults: {width: 150},
                items: [{
                        xtype: "textfield",
                        fieldLabel: "Telefono",
                        name:"Tlef",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                border: false,
                layout: "form",
                labelAlign: "top",
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: "textfield",
                        fieldLabel: "e-mail",
                        name : "e-mail",
                        anchor: "100%"
                }]
        },{
                xtype: "panel",
                border: false,
                layout: "form",
                colspan: 3,
                labelAlign: "top",
                padding: 5,
                items: [{
                        xtype: "textfield",
                        fieldLabel: "Direccion",
```

```
                name: "Direccion",
                anchor: "100%"
            }]
        },{
            xtype: "panel",
            border: false,
            layout: "form",
            colspan: 3,
            labelAlign: "top",
            padding: 5,
            items: [{
                xtype: "textfield",
                fieldLabel: "Antecedentes Personales",
                name : " Apersonal",
                anchor: "100%"
            }]
        },{
            xtype: "panel",
            border: false,
            layout: "form",
            colspan: 3,
            labelAlign: "top",
            padding: 5,
            items: [{
                xtype: "textfield",
                fieldLabel: "Antecedentes Familiares",
                name : " AFamiliares",
                anchor: "100%"
            }]
        }],
```

```
                    fbar: {
                        xtype: "toolbar",
                        items: [{
                            xtype: "button",
                            text: "Regresar",
                            handler :
function(){document.location.href='../formas/Menu.html';}
                        },'->',{
                            xtype: "button",
                            text: "Guardar",
                            handler :
function(){enviarForma("PruebaFormat","formita","MOdonto",contObj.Forma.TableLa
youtFormulario.respuetaFormu);}
                        }]
                    }
                }]// cierre del item formita
            });//Fin de Panel win
            var panelForma = new Ext.Panel({//COntenedor Padre inicio
                region : "center",
                autoScroll : true,
                border : true,
                //layout : "fit",
                listeners : {//afterrender :
                    resize : function (obj,w,h,rw,rh){
                        if(Ext.fly(win.id) != null){

    //Ext.fly(prueba.id).center(Ext.get('contenedorID'));
                            Ext.fly(win.id).center(panelForma.body);
                            var obj1 = Ext.fly(win.id).getPositioning();
                            centrado(obj1,win.id);
```

```
                                        }
                                    }
                                },
                                items : [win]
                            });
                            var contenedor = new Ext.Viewport ({
                                layout : "border",
                                id     : 'idPrincipal',
                                defaults: {border : false},
                                items : [{
                                    xtype : "panel",
                                    region : "north",
                                    height : 175,
                                    html : "<table width='100%' height='100%'><tr><td><img
width='100%' height='100%' src='../Imagin/Encabezado.png' /></td></tr></table>"
                                },panelForma]
                            });
                            panelForma.fireEvent('resize',panelForma);//Forza el evento resize del
panel que contiene el form
                        },//Fin de Init
                        respuetaFormu : function (form,action){
                            var data = action.result.existe;
                            if (data == 0){
                                var cedula = action.result.data.CedulaP;

                        Ext.getCmp('formita').getForm().load({params:{funcion:'PruebaFormat',CedulaP
:cedula}});
                            }
                            else
                                alert (action.result.msg);
```

```
	},

	revisarSesion : function (response, request){
		var dat = Ext.util.JSON.decode(response.responseText);
		if (dat.session == 1)
			this.Forma.TableLayoutFormulario.init();
		else{
			console.debug ("Ya paso desde el servidor");
			document.location.href='../index.html';
		}
	},

	inicio : function (){
		conectar_Servidor ('verificarSesion',this.revisarSesion,'null');
	}
}//cierra el return
}();
/*
function mostrarVentanaPDF (id,url,nombre){
	var pdf = !Ext.getCmp(id);
	if (pdf){
		var stringHtml = "<iframe id='" + id + "' src='" + url + "' width=880
height=660 scrolling='no' frameborder='0'></iframe>";
		ventana0123= new Ext.Window({
			layout:'fit',
			width: 900,
			height:700,
			title: nombre,
			closeAction:'hide',
			modal:true,
```

```
                    plain: true,
                    html: stringHtml
                });
        }
        ventana0123.show(this);
}
window.onload = function(){
        revisarSesion();
}
Ext.getCmp("formuInsertElimin").getForm().loadRecord(linea);
*/

Ext.onReady(Forma.TableLayoutFormulario.inicio,Forma.TableLayoutFormulario);
```

5.3.4 Interfaz de búsqueda de tratamientos

La operación de buscar un tratamiento por cada estudiante fue construida para ofrecer facilidad de búsqueda y mucha flexibilidad. Al usuario se le muestra un grupo de cajas de selección de opciones múltiples de búsqueda, entre ellas, cédula, nombre, apellido y. Después de escoger los parámetros de búsqueda, se muestra el odontograma con los datos de cada estudiante y los diferentes tratamientos que se han realizado al paciente. El resultado de la búsqueda es una tabla de varios campos con información relacionada al o los estudiantes encontrados. A continuación se presenta el código fuente de la página, y en la Figura 5.4se puede observar la vista gráfica de la misma.

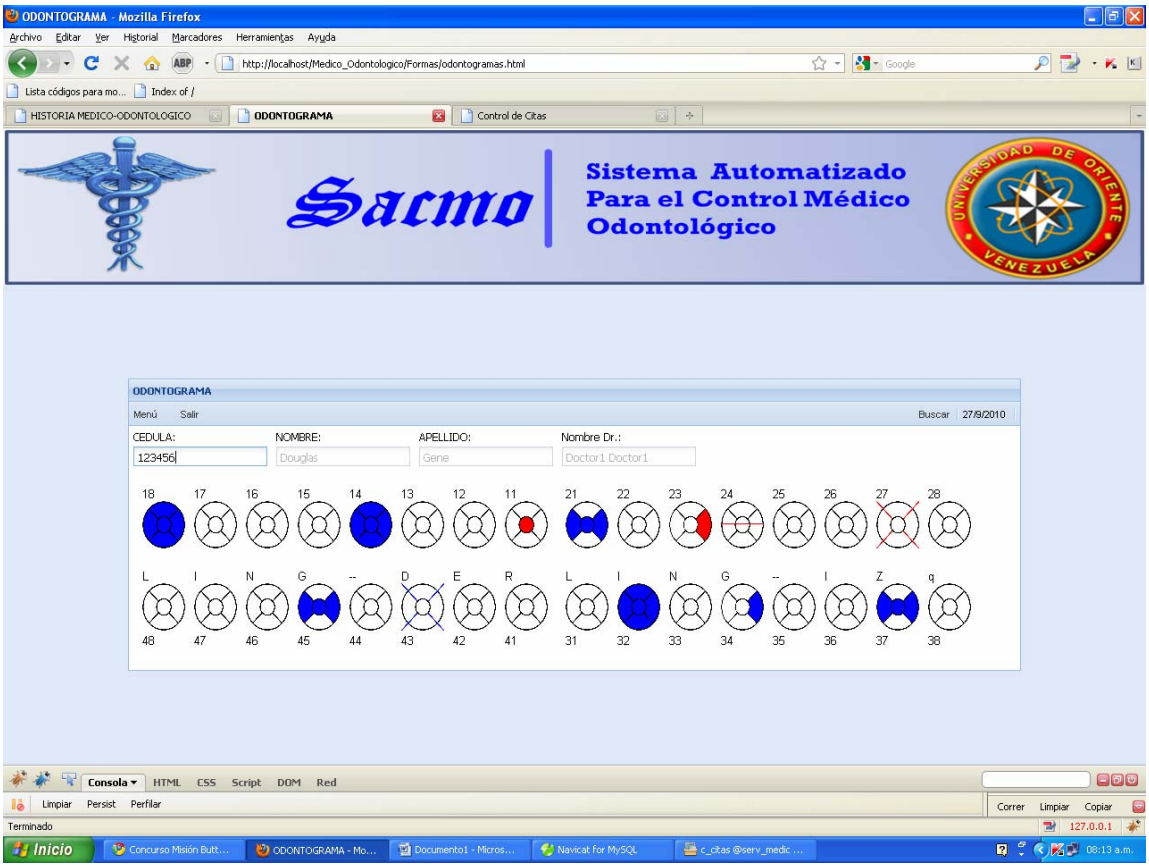


Figura 5.4Vista de la interfaz de búsqueda Historial Odontologico.
Fuente: Elaboración Propia

Código fuente Odontograma.

```
Ext.ns("Forma");


var pieza;

var parametros;

var arrayOdonto = new Array();

var wn = new Ext.Window({

        title: "PIEZAS",

        width: 200,

        height: 250,
```

```
        layout: "fit",
        closeAction : 'hide',
        //closable : true,
        draggable : true,
        items: [{
                xtype: "panel",
                layout: "table",
                border: false,
                bodyStyle: "padding:10px",
      autoScroll:true,
      layoutConfig: {columns: 1},
                items: [{xtype: "panel",
                        border: false,
                        padding: 5,
                        defaults: {width: 150},
                        html: '<td>Diente_Ausente</TD><DIV align=center><IMG
id="1"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Diente_Sano.png"
width=48 height=53></DIV>'
                    },{
                        xtype: "panel",
                        border: false,
                        padding: 5,
                        defaults: {width: 150},
```

```
                    html: '<td>Diente_Ausente</TD><DIV align=center><IMG
id="2"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Diente_Ausente.png"
width=48 height=53></DIV>'

                  },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {

                    width: 150

                  },

                  html: '<td>Conducto_Realizado</TD><DIV
align=center><IMG id="3"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Conducto_Realizado.png"
width=48 height=53></DIV>'

                  },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {

                    width: 150

                  },

                  html: '<td>Obsturacion_Incisal</TD><DIV
align=center><IMG id="4"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Obsturacion_Incisal.png"
width=48 height=53></DIV>'
```

```
                    },{
                      xtype: "panel",
                      border: false,
                      padding: 5,
                      defaults: {
                      width: 150
                    },
                    html: '<td>Obsturacion_Mesial</TD><DIV
align=center><IMG id="5"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Obsturacion_Mesial.png"
width=48 height=53></DIV>'
                    },{
                     xtype: "panel",
                     border: false,
                     padding: 5,
                     defaults: {
                     width: 150
                    },
                    html: '<td>Obsturacion_Distal</TD><DIV align=center><IMG
id="6"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Obsturacion_Distal.png"
width=48 height=53></DIV>'
                    },{
                     xtype: "panel",
                     border: false,
```

```
                    padding: 5,

                    defaults: {

                    width: 150

                   },

                   html: '<td>Obsturacion_modal</TD><DIV
align=center><IMG id="7"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Obsturacion_modal.png"
width=48 height=53></DIV>'

                   },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {

                    width: 150

                   },

                   html: '<td>Diente_Con_Corona</TD><DIV
align=center><IMG id="8"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Diente_Con_Corona.png"
width=48 height=53></DIV>'

                   },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {

                    width: 150
```

```
                    },

                    html: '<td>Caries_Octusal</TD><DIV align=center><IMG
id="9"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Caries_Octusal.png"
width=48 height=53></DIV>'

                    },{

                     xtype: "panel",

                     border: false,

                     padding: 5,

                     defaults: {

                     width: 150

                    },

                    html: '<td>Caries_Mesial</TD><DIV align=center><IMG
id="10"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Caries_Mesial.png"
width=48 height=53></DIV>'

                    },{

                     xtype: "panel",

                     border: false,

                     padding: 5,

                     defaults: {

                     width: 150

                    },

                    html: '<td>Caries_Distal</TD><DIV align=center><IMG
id="11"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
```

```
metros2);" alt="click para ver detalles" src="../Imagin/Caries_Distal.png"
width=48 height=53></DIV>'

                        },{

                         xtype: "panel",

                         border: false,

                         padding: 5,

                         defaults: {

                         width: 150

                        },

                        html: '<td>Caries_Modal</TD><DIV align=center><IMG
id="12"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+parseInt(this
.id)+parametros2);" alt="click para ver detalles"
src="../Imagin/Caries_Modal.png" width=48 height=53></DIV>'

                        },{

                         xtype: "panel",

                         border: false,

                         padding: 5,

                         defaults: {

                         width: 150

                        },

                        html: '<td>Diente_Por_Extraer</TD><DIV align=center><IMG
id="13"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles" src="../Imagin/Diente_Por_Extraer.png"
width=48 height=53></DIV>'

                        },{

                         xtype: "panel",
```

```
                        border: false,

                        padding: 5,

                        defaults: {

                        width: 150

                       },

                       html: '<td>Conducto_Por_Realizar</TD><DIV
align=center><IMG id="14"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles"
src="../Imagin/Conducto_Por_Realizar.png" width=48 height=53></DIV>'

                       },{

                        xtype: "panel",

                        border: false,

                        padding: 5,

                        defaults: {

                        width: 150

                       },

                       html: '<td>Limpieza_Por_Realizar</TD><DIV
align=center><IMG id="15"
onClick="mensajeAlerta(\'pruebaConectar\',cambiar,parametros+this.id+para
metros2);" alt="click para ver detalles"
src="../Imagin/Limpieza_Por_Realizar.png" width=48 height=53></DIV>'

                       }]

   }]

});


Forma.Odontograma = {
```

```
init: function(){
        var win = new Ext.Panel({
                title: "ODONTOGRAMA",
                id: "formita",
                width: 1000,
                height: 'auto',
                layout: "fit",
                closable : false,
                draggable : false,
                renderTo: 'mostrar',
                items: [{
                        xtype: 'panel',
                        layout: 'border',
                        bodyStyle: {background: '#ffffff'},
                        border: false,
                        height: 300,    //Este es el del Border el contenedor
del form y el odontograma
                        items: [{
                                xtype: 'form',
                                id: 'datOndonto',
                                url:'../php/pruebasPHP.php',
                                height: 80, //Este es el del Form
                                region: 'north',
                                border: false,
                                layout: 'table',
```

```
                                    layoutConfig: {        columns: 4},
                                    tbar: {
                                            xtype: "toolbar",
                                            items: [{
                                                    xtype: "button",
                                                    text: "Men&#250;",
                                                    handler :
function(){document.location.href='../formas/Menu.html';}
                                            },{
                                                    xtype: "button",
                                                    iconCls : 'menu',
                                                    text: "Salir",
                                                    handler :
function(){document.location.href='../index.html';}
                                            },'->',{
                                                    xtype: "button",
                                                    text: "Buscar",
                                                    handler : function(){
                                                            var valor =
document.getElementById('cedulaOdonto').value;

        conectar_Servidor('dibujaPieza',buscarCedula,valor);
                                                    }
                                            }, '-',new Date().format('d/n/Y'), ' ', ' ', '-']
                                    },
                                    items: [{
```

```
                                        xtype: 'panel',
                                        layout: 'form',
                                        border: false,
                                        height: 50,
                                        labelAlign: 'top',
                                        padding: 5,
                                        defaults: {width: 150},
                                        items: [{
                                                xtype: 'textfield',
                                                id: 'cedulaOdonto',
                                                fieldLabel: 'CEDULA',
                                                name:'CedulaP',
                                                anchor: '100%',
                                                listeners: {
                                                        scope      : this,
                                                        specialkey : function(field,
e) {
                                                                if (e.getKey() ===
e.ENTER) {

        conectar_Servidor('dibujaPieza',buscarCedula,field.getValue())
                                                                }
                                                        }
                                                }
                                        }]
                                },{
```

```
                xtype: 'panel',
                layout: 'form',
                height: 50,
                border: false,
                labelAlign: 'top',
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: 'textfield',
                        fieldLabel: 'NOMBRE',
                        disabled : true,
                        name:'nombres',
                        anchor: '100%'
                }]
        },{
                xtype: 'panel',
                layout: 'form',
                height: 50,
                border: false,
                labelAlign: 'top',
                padding: 5,
                defaults: {width: 150},
                items: [{
                        xtype: 'textfield',
```

```
                    fieldLabel: 'APELLIDO',
                    disabled : true,
                    name:'apellidos',
                    anchor: '100%'
                }]
            },{
                xtype: 'panel',
                layout: 'form',
                height: 50,
                border: false,
                labelAlign: 'top',
                padding: 5,
                defaults: {width: 150},
                items: [{
                    xtype: 'textfield',
                    fieldLabel: 'Nombre Dr.',
                    disabled : true,
                    name:'nombreDr',
                    anchor: '100%'
                }]
            }]
        },{
            xtype: "panel",
            layout: "table",
```

```
id:'datOdontoGrama',

region: 'center',

hidden : 'true',

border: false,

height: 220,        //Este es el del odontograma

bodyStyle: "padding:10px",

layoutConfig: {columns: 17},

items: [{

        xtype: "panel",

        border: false,

        padding: 5,

        //defaults: {width: 150},

        html: '<td>   18</TD><DIV
align=center><IMG id="D18" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

},{

        xtype:"panel",

        border: false,

        padding: 5,

        defaults: {width: 150},

        html: '<td>   17</TD><DIV
align=center><IMG id="D17" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

},{

        xtype: "panel",

        border: false,
```

```
                    padding: 5,
                    defaults:{width:150},
                    html: '<td>   16</TD><DIV
align=center><IMG id="D16" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'
                },{
                    xtype:"panel",
                    border: false,
                    padding: 5,
                    defaults:{width: 150},
                    html: '<td>   15</TD><DIV
align=center><IMG id="D15" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'
                },{
                    xtype: "panel",
                    border: false,
                    padding: 5,
                    defaults: {width: 150},
                    html: '<td>   14</TD><DIV
align=center><IMG id="D14" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'
                },{
                    xtype: "panel",
                    border: false,
                    padding: 5,
                    defaults:{width: 150},
```

```
                    html: '<td>   13</TD><DIV
align=center><IMG id="D13" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults:{width: 150},

                    html: '<td>   12</TD><DIV
align=center><IMG id="D12" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width: 150},

                    html: '<td>   11</TD><DIV
align=center><IMG id="D11" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width: 150}

                },{

                    xtype: "panel",

                    border: false,
```

```
                padding: 5,
                defaults: {width:150},
                html: '<td>21      </TD><DIV
align=center><IMG id="D21" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'
            },{
                xtype: "panel",
                border: false,
                padding: 5,
                defaults: {width:150},
                html: '<td>22      </TD><DIV
align=center><IMG id="D22" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'
            },{
                xtype: "panel",
                border: false,
                padding: 5,
                defaults: {width:150},
                html: '<td>23      </TD><DIV
align=center><IMG id="D23" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'
            },{
                xtype: "panel",
                border: false,
                padding: 5,
                defaults: {width:150},
```

html: '<td>24 </TD><DIV align=center><IMG id="D24" onClick="mostrarWindows(this);" alt="Click para Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

},{

xtype: "panel",

border: false,

padding: 5,

defaults: {width:150},

html: '<td>25 </TD><DIV align=center><IMG id="D25" onClick="mostrarWindows(this);" alt="Click para Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

},{

xtype: "panel",

border: false,

padding: 5,

defaults: {width:150},

html: '<td>26 </TD><DIV align=center><IMG id="D26" onClick="mostrarWindows(this);" alt="Click para Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

},{

xtype: "panel",

border: false,

padding: 5,

defaults: {width:150},

html: '<td>27 </TD><DIV align=center><IMG id="D27" onClick="mostrarWindows(this);" alt="Click para Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'

```
},{
        xtype: "panel",
        border: false,
        padding: 5,
        defaults: {width:150},
        html: '<td>28      </TD><DIV
align=center><IMG id="D28" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV>'
},{
        xtype: "panel",
        border: false,
        padding: 5,
        defaults: {width: 150}
},{
        xtype: "panel",
        border: false,
        padding: 5,
        defaults: {width: 150}
},{
        xtype: "panel",
        border: false,
        padding: 5,
        defaults: {width: 150}
},{
        xtype: "panel",
```

```
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
```

```
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
        },{
            xtype: "panel",
            border: false,
            padding: 5,
            defaults: {width: 150}
```

```
},{
        xtype: "panel",
        border: false,
        padding: 5,
        defaults: {width: 150}
},{
        xtype: "panel",
        border: false,
        padding: 5,
        defaults: {width: 150}
},{
        xtype: "panel",
        border: false,
        padding: 5,
        defaults: {width: 150}
},{
        xtype: "panel",
        border: false,
        padding: 5,
        defaults: {width: 150}
},{
        xtype: "panel",
        border: false,
        padding: 5,
```

```
                    defaults: {width: 150},

                    html: '<td>L      </TD><DIV
align=center><IMG id="D48" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
48</TD>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width: 150},

                    html: '<td>I      </TD></Td><DIV
align=center><IMG id="D47" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
47</TD>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {

                    width: 150},

                    html: '<td>N      </TD><DIV
align=center><IMG id="D46" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
46</TD>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,
```

```
                    defaults: {width:150},

                    html: '<td>G      </TD><DIV
align=center><IMG id="D45" onClick="mostrarWindows(this);" alt="click para ver
detalles" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
45</TD>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width:150},

                    html: '<td> --    </TD><DIV
align=center><IMG id="D44" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
44</TD>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width:150},

                    html: '<td>    D</TD><DIV
align=center><IMG id="D43" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
43</TD>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width:150},
```

```
                    html: '<td>    E</TD><DIV
align=center><IMG id="D42" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
42</TD>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width:150},

                    html: '<td>    R</TD><DIV
align=center><IMG id="D41" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
41</TD>'

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width: 150}

                },{

                    xtype: "panel",

                    border: false,

                    padding: 5,

                    defaults: {width:150},

                    html: '<td>    L</TD><DIV
align=center><IMG id="D31" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>31
</TD>'

                },{
```

```
                    xtype: "panel",
                    border: false,
                    padding: 5,
                    defaults: {width: 150},
                    html: '<td>    I</TD></Td><DIV
align=center><IMG id="D32" onClick="mostrarWindows(this);" alt="click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
32</TD>'
                },{
                    xtype: "panel",
                    border: false,
                    padding: 5,
                    defaults: {width: 150},
                    html: '<td>N      </TD><DIV
align=center><IMG id="D33" onClick="mostrarWindows(this);" alt="click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
33</TD>'
                },{
                    xtype: "panel",
                    border: false,
                    padding: 5,
                    defaults: {width:150},
                    html: '<td>     G</TD><DIV
align=center><IMG id="D34" onClick="mostrarWindows(this);" alt="click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
34</TD>'
                },{
                    xtype: "panel",
```

```
                                        border: false,

                                        padding: 5,

                                        defaults: {width:150},

                                        html: '<td>   --  </TD><DIV
align=center><IMG id="D35" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
35</TD>'

                                    },{

                                        xtype: "panel",

                                        border: false,

                                        padding: 5,

                                        defaults: {width:150},

                                        html: '<td>     I</TD><DIV
align=center><IMG id="D36" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
36</TD>'

                                    },{

                                        xtype: "panel",

                                        border: false,

                                        padding: 5,

                                        defaults: {width:150},

                                        html: '<td>      Z</TD><DIV
align=center><IMG id="D37" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
37</TD>'

                                    },{

                                        xtype: "panel",

                                        border: false,
```

```
                                        padding: 5,
                                        defaults: {width:150},
                                        html: '<td>     q</TD><DIV
align=center><IMG id="D38" onClick="mostrarWindows(this);" alt="Click para
Modificar" src="../Imagin/Diente_sano.png" width=48 height=53></DIV><td>
38</TD>'
                                    }]


                                }]
                            }]
                        });//fin del win
                        var panelForma = new Ext.Panel({//COntenedor Padre inicio
                            region : "center",
                            autoScroll : true,
                            border : true,
                            bodyStyle:"background-color:#DFE8F6",
                            //layout : "fit",
                            listeners : {//afterrender :
                                resize : function (obj,w,h,rw,rh){
                                    if(Ext.fly(win.id) != null){

        //Ext.fly(prueba.id).center(Ext.get('contenedorID'));
                                        Ext.fly(win.id).center(panelForma.body);
                                        var obj1 = Ext.fly(win.id).getPositioning();
                                        centrado(obj1,win.id);
                                    }
```

```
                    }
                },
                items : [win]
            });
            var contenedor = new Ext.Viewport ({
                layout : "border",
                id     : 'idPrincipal',
                defaults: {border : false},
                items : [{
                    xtype : "panel",
                    region : "north",
                    height : 175,
                    html : "<table width='100%'><tr><td
height='173px'><img width='100%' height='100%'
src='../Imagin/EncabezadoLogo.png' /></td></tr></table>"
                },panelForma]
            });
            panelForma.fireEvent('resize',panelForma);//Forza el evento resize
del panel que contiene el form
        },//fin del init


        revisarSesion : function (response, request){
            var dat = Ext.util.JSON.decode(response.responseText);
            if (dat.session == 1)
                this.Forma.Odontograma.init();
            else{
```

```
                    document.location.href='../index.html';
                }
        },

        inicio : function (){
                conectar_Servidor ('verificarSesion',this.revisarSesion,'null');
        }
}

var cambiar = function (response, request){
        var dat = Ext.util.JSON.decode(response.responseText);
        pieza.src="../Imagin/" + dat.img;
}
function mostrarWindows (obj){
        wn.show();
        pieza = obj;
        parametros = '{"diente":"'+obj.id+'", "pieza":';
        parametros2 = '}';
}
function mensajeAlerta (funPhp,funJs,parametros){
        Ext.Msg.confirm('<b>CONFIRMAR<b>', 'Esta Seguro que desea
<b>Modificar Pieza</b>', function(btn){
                if (btn == 'yes'){
                        conectar_Servidor (funPhp,funJs,parametros);
                }
```

```
        });
}
var buscarCedula = function (response, request){
        var dat = Ext.util.JSON.decode(response.responseText);
        if (dat.success){

        Ext.getCmp('datOndonto').getForm().load({params:{funcion:'buscarDatos
OdontoForm', cedula:dat.cedula}});
                var i;
                for (i=0; i<32; i++){
                        ruta = "../Imagin/"+dat.odontograma[i]['n_Pieza']+".png";
                        id = dat.odontograma[i]['cod_Diente'];
                        document.getElementById(id).src=ruta;
                }
                Ext.getCmp('datOdontoGrama').setVisible(true);
        }else{
                Ext.Msg.show({
                        title: 'Error',
                        msg: dat.msg,
                        buttons: Ext.Msg.OK,
                        icon: Ext.MessageBox.ERROR
                });
                Ext.getCmp('datOndonto').getForm().reset();
                Ext.getCmp('datOdontoGrama').setVisible(false);
        }
```

```
}
var getOdonto = function (form,action){
        var dat = Ext.util.JSON.decode(response.responseText);
        arrayOdonto = dat.datos;
}
Ext.onReady(Forma.Odontograma.inicio,Forma.Odontograma);
```

En la siguiente Figura 5.5. Se muestra la opción para seleccionar un tratamiento que se le realizara al paciente y muestra los tratamientos ya realizados.

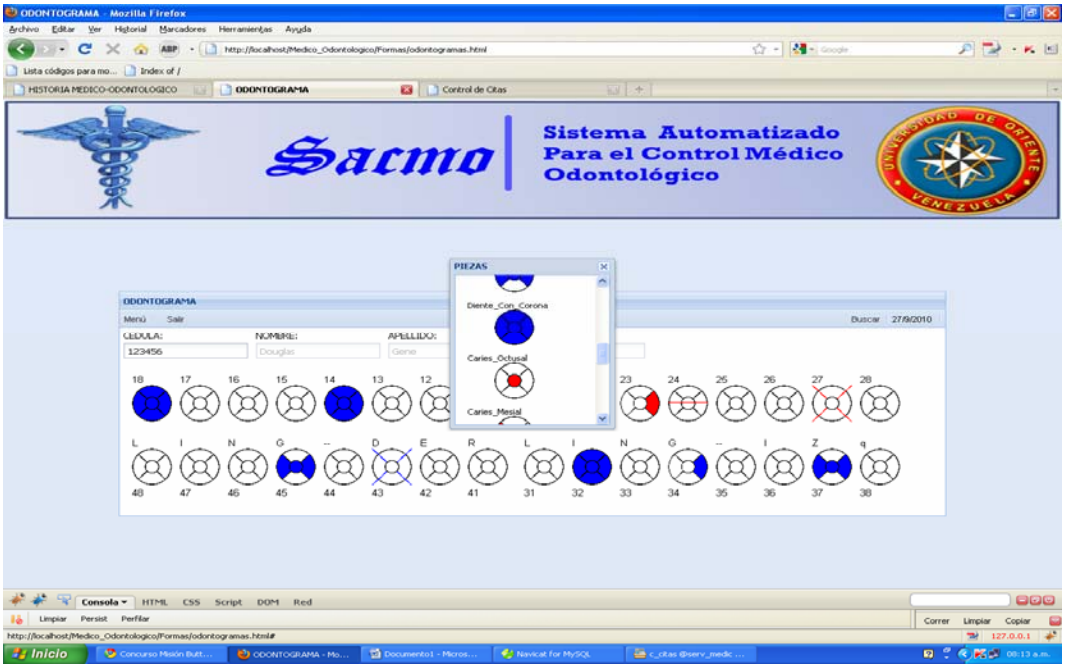


Figura 5.5:Página que muestra la selección de un tratamiento.
Fuente: Elaboración Propia.

5.3.5 Interfaz para otorgar citas odontológicas.

La operación de otorgar una Cita por cada paciente fue construida para ofrecer facilidad de búsqueda y mucha flexibilidad. Al usuario se le muestra un grupo de cajas de selección de opciones múltiples de búsqueda, entre ellas, cédula, día de la cita y para que turno desea que se le dé la cita. Después de escoger los parámetros de búsqueda, se muestra la página con las citas efectuadas a otros pacientes y los diferentes días

disponibles. El resultado de la búsqueda es una tabla de varios campos con información relacionada al o los estudiantes encontrados. A continuación se presenta el código fuente de la página, y en la Figura 5.6. se puede observar la vista gráfica de la misma.

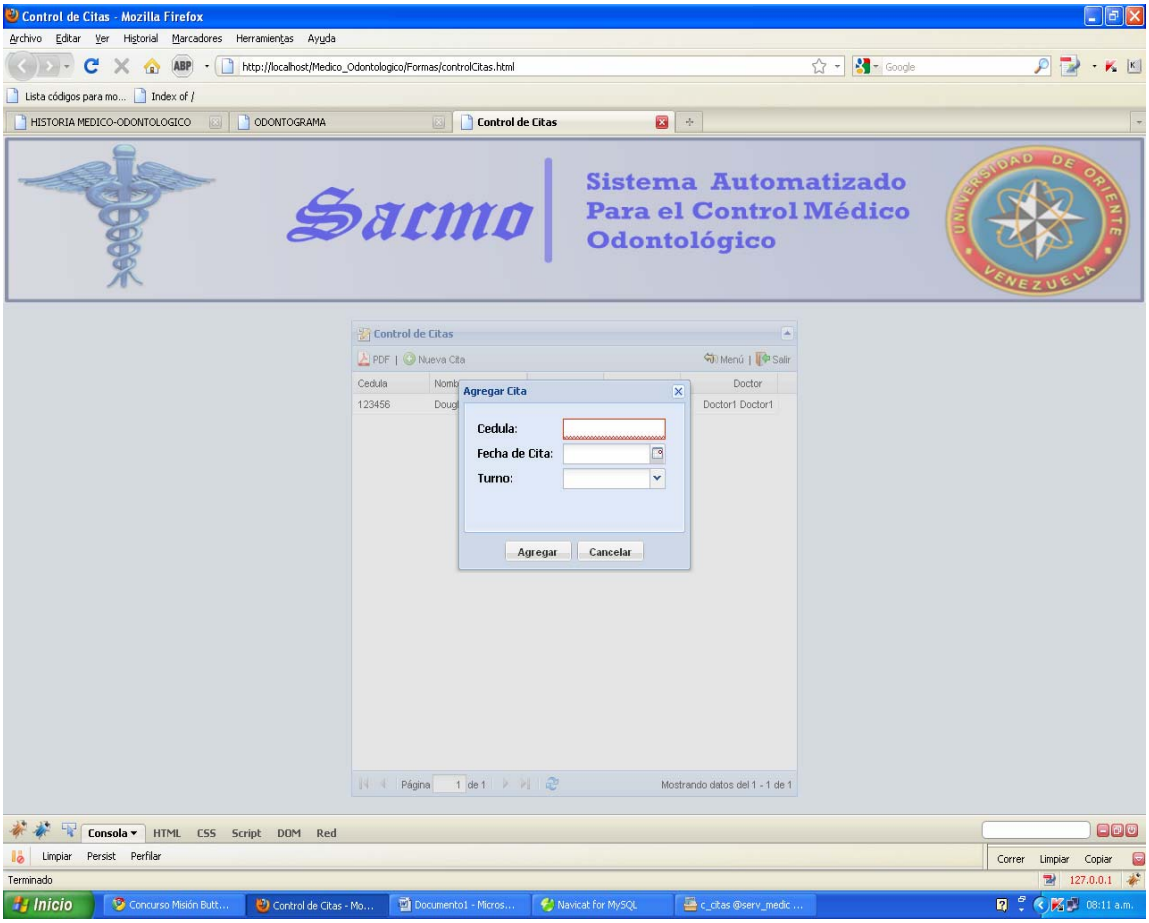


Figura 5.6: Página para el otorgamiento de citas odontológicas.
Fuente: Elaboración propia.

Código Control de Citas,,

```
Ext.ns("Forma");

Ext.SSL_SECURE_URL  = '../resources/images/default/s.gif';

Ext.BLANK_IMAGE_URL = '../resources/images/default/s.gif';

Ext.QuickTips.init();
```

```
Forma.ControlCitas = function(){
var contObj = this;
var formaWindow;
var dsControlCitas;

return{
        init: function(){
                dsControlCitas = new Ext.data.JsonStore({
                        url: '../php/pruebasPHP.php',
                        id:     'rControlCitas',
                        totalProperty: 'total',
                        root: 'data',
                        fields: [
                                {name : 'CedulaP'},
                                {name : 'nombreApellido'},
                                {name : 'f_Cita', type: 'date', dateFormat: 'Y-m-d'},
                                {name : 'turno'},
                                {name : 'nombreDr'}
                        ],
                        baseParams:{funcion:'gridControlCitas'},
                        sortInfo: {field: 'f_Cita', direction: 'DESC'},
                        remoteSort: true
                });//Fin del dsControlCitas
```

```
var ftControlCitas = new Ext.ux.grid.GridFilters({
        menuFilterText : 'Filtro',
        filters:[
            {type: 'string',  dataIndex: 'CedulaP'},
            {type: 'date', dataIndex: 'f_Cita', dateFormat: 'd/m/Y',
beforeText: 'Antes', afterText: 'Despu&#233;s', onText: 'En',},
            {type: 'list', dataIndex: 'turno', options: ['Mañana',
'Tarde'], phpMode: true},
            {type: 'string',  dataIndex: 'nombreDr'}
        ]
});//Fin del ftControlCitas


var cmControlCitas = new Ext.grid.ColumnModel([
        {dataIndex: 'CedulaP', header: 'Cedula', width: 80},
        {dataIndex: 'nombreApellido', header: 'Nombre', id:
'cmContCita'},
        {dataIndex: 'f_Cita', header: 'Fecha', align: 'left', width: 80,
renderer: Ext.util.Format.dateRenderer('d/m/Y')},
        {dataIndex: 'turno', header: 'Turno', width: 80, align: 'right',
renderer: this.turno},
        {dataIndex: 'nombreDr', header: 'Doctor', width: 100, align:
'right'}
]);//Fin del cmControlCitas
cmControlCitas.defaultSortable = true;


var bbarControlCitas = new Ext.PagingToolbar({
        store: dsControlCitas,
```

```
        pageSize: 25,
        plugins: ftControlCitas,
        displayInfo: true,
        displayMsg: 'Mostrando datos del {0} - {1} de {2}',
        emptyMsg: "No hay datos para Mostrar"
});//Fin del bbarControlCitas

var gridControlCitas = new Ext.grid.GridPanel({
        title:'Control de Citas',
        id: 'gdControlCitas',
        ds: dsControlCitas,
        cm: cmControlCitas,
        enableColLock: false,
        loadMask: true,
        plugins: ftControlCitas,
        autoExpandColumn: 'cmContCita',
        width:500,
        height:500,
        stripeRows: true,
        collapsible: true,
        animCollapse: true,
        iconCls: 'ControlCitas',
        enableColumnResize: false,
        enableColumnMove: true,
```

```
                    viewConfig: {
                            forceFit:true,
                            enableRowBody:true,
                            showPreview:true
                    },
                    tbar:{
            xtype: 'toolbar',
            items: [{
                                    xtype: 'button',
                                    iconCls : 'pdf',
                                    text: 'PDF',
                                    //handler : function(){crearPDF
('reportePacientePDF','../php/reportesPDF.php?f=reporteControlCitasOdonto','P
rueba PDF Pacientes');}
                            },'|',{
                                    xtype: 'button',
                                    iconCls : 'add',
                                    text: 'Nueva Cita',
                                    handler :
function(){Forma.ControlCitas.mostrarWindow();}
                            },'->',{
                                    xtype: 'button',
                                    iconCls : 'menu',
                                    text: 'Men&#250;',
                                    handler :
function(){document.location.href='../formas/Menu.html';}
```

```
                    },'|',{
                        xtype: 'button',
                        iconCls : 'salir',
                        text: 'Salir',
                        handler :
function(){document.location.href='../index.html';}
                    }]
                },
                bbar: bbarControlCitas,
                listeners: {
                    render: function(){
                        dsControlCitas.load({params:{start: 0, limit:
25}});
                    }
                }
            });//Fin de Panel gridControlCitas

            var panelForma = new Ext.Panel({//COntenedor Padre de
gridControlCitas
                region : "center",
                autoScroll : true,
                border : true,
                bodyStyle:"background-color:#DFE8F6",
                listeners : {//afterrender :
                    resize : function (obj,w,h,rw,rh){
                        if(Ext.fly(gridControlCitas.id) != null){
```

```
        Ext.fly(gridControlCitas.id).center(panelForma.body);
                                                var obj1 =
Ext.fly(gridControlCitas.id).getPositioning();
                                                centrado(obj1,gridControlCitas.id);
                                        }
                                }
                        },
                        items : [gridControlCitas]
                });//Fin de Panel panelForma
                var contenedor = new Ext.Viewport ({
                        layout : "border",
                        id     : 'idPrincipal',
                        defaults: {border : false},
                        items : [{
                                xtype : "panel",
                                region : "north",
                                height : 175,
                                html : "<table width='100%'><tr><td
height='173px'><img width='100%' height='100%'
src='../Imagin/EncabezadoLogo.png' /></td></tr></table>"
                        },panelForma]
                });//Fin de Panel contenedor
                panelForma.fireEvent('resize',panelForma);//Forza el evento resize
del panel que contiene el form
        },//Fin de Init
```

```
        mostrarWindow : function (){
                if (!formaWindow){
                        formaWindow = this.crearWindow();
                }
                formaWindow.show();
        },
crearWindow : function (){
Ext.EventManager.onWindowResize(function(){formaWindow.el.center();},this,tru
e);
                var addFormaW = new Ext.FormPanel({
                        id                      : "formaWin",
                        url                     : "../php/pruebasPHP.php",
                        bodyStyle       : 'padding: 10px',
                        labelAlign      : "left",
                        defaultType     : "textfield",
                        labelWidth      : 90,
                        defaults        : {width: 115},
                        frame           : true,
                        items           : [{
                                fieldLabel      : "<b>Cedula</b>",
                                id                      : "idCedula",
                                name            : "CedulaP",
                                allowBlank      : false,
                                fireKey         : function(e){
                                        if(e.getKey() == e.ENTER){}
```

```
            }
        },{
            xtype: "datefield",
            fieldLabel: "<b>Fecha de Cita</b>",
            name:"f_Cita",
            allowBlank:false,
            editable : false
        },{
            xtype           : "combo",
            fieldLabel      : "<b>Turno</b>",
            id                  : "idTurno",
            //name              : "turno",
            hiddenName          : "turno",
            triggerAction: "all",
            mode            : "local",
            valueField      : "valor",
            displayField   : "nombre",
            lazyRender      : true,
            allowBlank      : false,
            editable        : false,
            store           : new Ext.data.ArrayStore({
                id          : 'comboTurno',
                fields   : ["valor","nombre"],
                data   : [["AM", "Mañana"], ["PM", "Tarde"]]
```

```
                })
            }]
});
return  new Ext.Window({
            title           : "Agregar Cita",
            buttonAlign     : "center",
            //iconCls               : "telefonoKey",
            layout          : "fit",
            closeAction     : "hide",
            width           : 260,
            height          : 200,
            modal           : true,
            draggable       : false,
            center          : true,
            resizable       : false,
            border                  : false,
            items           : [addFormaW],
            listeners       : {
                hide: function(){
                        var frm = addFormaW.getForm();
                        frm.reset();
                        frm.clearInvalid();
                },
                show: function(){
                        Ext.getCmp("idCedula").focus(true, 300);
```

```
                }
            },
            buttons : [{
                text    : "<b>Agregar</b>",
                id      : "botonGuardar",
                handler : function (){
                    Ext.Msg.confirm('<b>CONFIRMAR<b>', 'Esta
Seguro que desea agregar la cita', function(btn){
                        if (btn == 'yes'){
                            if
(addFormaW.getForm().isValid()) {
                                mask = new
Ext.LoadMask(Ext.get(Ext.getBody().id), {msg:'Guardando. Espere por favor...',
removeMask: true});
                                mask.show();

        enviarForma("addCita","formaWin","botonGuardar",contObj.Forma.Contr
olCitas.addCita);
                            }else{
                                Ext.MessageBox.show({
                                    title: 'Error',
                                    msg: '<b>Hay
campos obligatorios vacios.</b>',
                                    buttons:
Ext.MessageBox.OK,
                                    animEl:
'botonGuardar',
                                    closable:false,
```

```
                                                                icon:
Ext.MessageBox.ERROR
                                                            });
                                                        }
                                                    }
                                                });
                                            },
                                            scope   : this
                                        },{
                                            text    : "<b>Cancelar</b>",
                                            id              : "botonCancelar",
                                            handler         : function(){
                                                addFormaW.getForm().reset();
                                                Ext.getCmp("idCedula").focus(true, 300);
                                            }
                                        }]
                                });
                        },//Fin de metodo crearWindow
                        turno : function (value,metadata,record){
                                if (value == "PM"){
                                        return "Tarde"; //return "<img src='../Imagin/"+value+".png'
alt='imegen' />"
                                }else{
                                        return "Mañana";
                                }
```

```
        },
        addCita : function (form, action){
                mask.hide();
                Ext.Msg.alert('Mensaje', action.result.msg);
                Ext.getCmp("formaWin").getForm().reset();
                Ext.getCmp("idCedula").focus(true, 300);
                dsControlCitas.load({params:{start: 0, limit: 25}});
        },
        revisarSesion : function (response, request){
                var dat = Ext.util.JSON.decode(response.responseText);
                if (dat.session == 1)
                        this.Forma.ControlCitas.init();
                else{
                        document.location.href='../index.html';

                }
        },//Fin de metodo revisarSesion
        inicio : function (){
                conectar_Servidor ('verificarSesion',this.revisarSesion,'null');
        }//Fin de metodo inicio
}//cierra el return
}();
Ext.onReady(Forma.ControlCitas.inicio,Forma.ControlCitas);
```

5.4 Flujo de trabajo Pruebas

Para cada caso de uso se deben establecer pruebas de aceptación que validarán la correcta implementación del caso de uso. El objetivo de la fase de pruebas de un programa es de detectar todo posible malfuncionamiento antes de que entre en producción.

5.4.1 Modelos de pruebas

Las pruebas de unidad evalúan los componentes implementados como unidades individuales. Para efectuar las pruebas de unidad se utilizará la técnica de pruebas de caja negra. Conociendo la función específica para que se diseñara el producto, se aplican pruebas, que demuestren que cada función es plenamente operacional, mientras se buscan los errores de cada función. Las pruebas de caja negra son las que se aplican a la interfaz del software. Una prueba de este tipo examina algún aspecto funcional de un sistema que tiene poca relación con la estructura lógica del software.

5.4.1.1 Partición Equivalente

La partición equivalente es un método de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba. Un caso de prueba ideal de manejo simple descubre una clase de errores (por ejemplo, procesamiento incorrecto de todos os datos de caracteres) que, de otra manera, requeriría la ejecución de muchos casos antes de que se observe el error general. La partición equivalente se esfuerza por definir un caso de prueba que descubra ciertas clases de errores, reduciendo así el número total de casos de prueba que deben desarrollarse.

El diseño de casos de prueba para una partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Una clase de equivalencia representa un conjunto de estados válidos y no válidos para las condiciones de entrada. Por lo general, una condición de entrada es un valor numérico específico, un

rango de valores, un conjunto de valores relacionados o una condición booleana. Las clases de equivalencia se definen de acuerdo con las siguientes directrices:

Si una condición de entrada específica un rango, se definen una clase de equivalencia cálida y dos no válidas.

- ✓ Si una condición de entrada requiere un valor específico, se definen una clase de equivalencia válida y dos no válidas.

- ✓ Si una condición de entrada específica un miembro de un conjunto, se definen una clase de equivalencia válida y dos no válidas.

- ✓ Si una condición de entrada es booleana, se definen una clase de equivalencia válida y otra no válida.

Al aplicar estas directrices para la derivación de clases de equivalencia, se desarrollarán y ejecutarán los casos de prueba para cada objeto de los datos del dominio de entrada. Los casos de prueba se seleccionan de modo que el mayor número de atributos de clase de equivalencia se ejercita una vez

5.4.1.2 . Identificación de Clases de Equivalencia

- Sólo números.

- Sólo caracteres.
- Caracteres y números.

- Email.

- Ningún carácter.

5.4.1.3 Grupo de Tipos de Entrada de Datos

- Numero Oficio: recibe de entrada sólo números enteros positivos, y estos no pueden ser mayores que el valor preestablecido de número de oficio máximo.

- Número: recibe de entrada sólo números enteros positivos.

- Alfanuméricos: recibe solamente caracteres alfabéticos en mayúscula y minúscula. Al procesar la entrada, la cadena de caracteres se convierte a mayúscula.

- Alfabéticos: recibe sólo entradas de tipo alfabéticas en mayúsculas y minúsculas. Al procesar la entrada, la cadena de caracteres se convierte a mayúscula.

- Email: procesa tipos de datos formados por una cadena de caracteres alfanuméricos, seguida por un arroba "@" y con otra cadena de caracteres al final.

- No Vacío: se refiere a entradas de datos que deben contener al menos un carácter.

- Año: procesa entradas numéricas de cuatro dígitos enteras y positivas. El número no puede ser mayor que el año en curso.

- Período: recibe de entrada, sólo números enteros positivos. No pueden ser mayores que el valor preestablecido de número de período máximo, y tampoco mayor que el número del período en curso.

- Mes: recibe sólo como entrada números mayores o iguales que 1 y menores o iguales que 31.
- Día: recibe sólo como entrada números mayores o iguales que 1 y menores o iguales que 12.

5.4.1.4 Aplicación de Casos de Prueba

La Tabla 5.1, que se muestra a continuación describe la evaluación de casos de prueba que se realizaron al sistema.

Tabla 5.1. Tabla de evaluación de casos de prueba (1 de 2)

Grupo	Caso de prueba	Válida	No válida	Clase de equivalencia
1	125	X		1
1	AbcABC		X	2
1	123abc456CDE		X	3
1	xyz@zzz.bec		X	4
1	“ ”		X	5
2	125	X		1
2	abcABC		X	2

Fuente: Elaboración Propia.

5.4.2 Pruebas de Integración

Conociendo el funcionamiento interno del producto, se aplican pruebas para asegurarse que “todas las piezas encajan”, en otras palabras, están integradas correctamente. Se basa en un examen cercano al detalle procedimental. Se prueban las rutas lógicas del software y la colaboración entre los componentes, al proporcionar casos de prueba que ejerciten conjuntos específicos de condiciones, bucles o ambos. Esta técnica de evaluación de software es denominada pruebas de caja blanca.

Estas pruebas se aplicaron a todos los componentes de software desarrollados, comprobando así la exitosa integración de la aplicación. Además, se probaron los enlaces de las páginas, que no hubiesen enlaces rotos o vacíos, ni enlaces que direccionaran hacia alguna página indebida. A continuación se presentan ejemplos de algunos casos de prueba de integración que se aplicaron al sistema.

Tabla 5.1. Tabla de evaluación de casos de prueba (2 de 2)

Grupo	Caso de prueba	Válida	No válida	Clase de equivalencia
2	123abc456CDE		X	3
2	xyz@zzz.bec		X	4
2	“ ”		X	5
3	125	X		1
3	abcABC	X		2
3	123abc456CDE	X		3
3	xyz@zzz.bec	X		4
3	“ ”	X		5
4	125		X	1
4	abcABC	X		2
4	123abc456CDE		X	3
4	xyz@zzz.bec		X	4
4	“ ”		X	5
5	125		X	1
5	abcABC		X	2
5	123abc456CDE		X	3
5	xyz@zzz.bec	X		4
5	“ ”		X	5
6	125	X		1
6	abcABC	X		2
6	123abc456CDE	X		3
6	xyz@zzz.bec	X		4
6	“ ”		X	5
7	125		X	1
7	2004	X		1
7	-3		X	1
7	abcABC		X	2
7	123abc456CDE		X	3
7	xyz@zzz.bec		X	4
7	“ ”		X	5
8	35		X	1
8	abcABC		X	2
8	123abc456CDE		X	3
8	xyz@zzz.bec		X	4
8	“ ”		X	5
9	77		X	1
9	5	X		1
9	abcABC		X	2
9	123abc456CDE		X	3
9	xyz@zzz.bec		X	4
9	“ ”		X	5
10	2	X		1
10	abcABC		X	2
10	123abc456CDE		X	3
10	xyz@zzz.bec		X	4

Fuente: Elaboración Propia.

Nota: el caso de prueba “” indica que la entrada no contiene caracteres de ningún tipo.

5.4.2.1 Diseño de Casos de Pruebas

Los casos de prueba deben realizarse sabiendo cómo funciona el programa, es decir, como es su desempeño internamente. A continuación se presenta un conjunto de tablas que describen los casos de prueba diseñados y la relación entre cada uno de ellos. Observe desde la Tabla 5.2

Tabla 5.2Caso de prueba – Iniciar sesión en el sistema

Caso de prueba	**Iniciar sesión en el sistema.**
Entrada	Nombre de usuario ppablo, contraseña 12345.
Resultado	El usuario es autenticado e ingresa al sistema.
Condiciones	El usuario está creado en la base de datos del sistema y no tiene una sesión abierta previamente.
Procedimiento	Escriba el nombre de usuario y la contraseña en los cuadro de texto correspondiente. Presione el botón entrar. Se ingresa al sistema.

Fuente. Elaboración propia.

Luego de iniciar sesión, procedemos ahora a agregar los datos de un Paciente en el sistema. Estas pruebas se muestran en la Tabla 5.3

5.4.3 Pruebas de integración

Las pruebas de integración se utilizan para verificar que los componentes interaccionan entre sí de la forma apropiada después de haber sido integrados.

El sistema SACMO está basado en un conjunto de archivos HTML, PHP, MySQL, JavaScript y CSS, entrelazados entre sí, por tal motivo, las pruebas de integración se realizaron probando todos y cada uno de los enlaces y simuladores presentes en el software.

Se navegó por todas las unidades haciendo link en cada una de las partes y procesos automatizados probando así su enlace y respuesta con las demás páginas, de igual manera se probaron los enlaces entre las páginas por medio del botón salir del menú de la aplicación.

Después de haber realizado todas las pruebas, tanto de unidad como de integración, se hizo una revisión de los resultados de estas y se determinó que el software ha superado las pruebas del sistema y que ha alcanzado la capacidad operativa inicial, por lo tanto no es necesario incluir otra iteración.

5.5 Evaluación de la Fase de Construcción

La fase de construcción se ha ejecutado con éxito. Se requirió una iteración donde se explotaron los flujos de trabajo de implementación y pruebas. Durante la implementación se realizó la codificación efectiva de las páginas web y los distintos componentes que conforman el software.

Cabe destacar que las herramientas que ofrece el lenguaje de modelado Web (WebML) fueron de mucha utilidad, ya que facilitan la codificación mediante estándares de programación web.

Todo software debe ser sometido a pruebas antes de su publicación o liberación. En esta fase, se le aplicaron pruebas al software, las cuales permitieron mejorar la calidad del software, ya que con la corrección de los errores presentados se garantiza la futura estabilidad operativa del mismo. Siendo más específico, se usaron técnicas conocidas, como los son, técnica de caja negra y de caja blanca

La ejecución de los procesos de implementación y pruebas han dado como resultado un producto estable y maduro como para ser entregado a la comunidad de usuarios para ser probado. Este proceso de prueba por parte de los usuarios debe realizarse en la próxima fase.

Tabla 5.3Caso de prueba – Iniciar Búsqueda de un paciente

Caso de prueba	**Búsqueda de un estudiante.**
Entrada	Se elige los campos por lo que se realizará la búsqueda. En este caso se seleccionan un campos de búsqueda: cédulaP, Se le introducen los datos de búsqueda, 16931487, pedrp pablo.
Resultado	Se muestra una tabla con los datos del estudiante.
Condiciones	El usuario está previamente autenticado. Los datos de la búsqueda concuerdan con los datos del usuario que se desea encontrar.
Procedimiento	Seleccione en el menú academia, el submenú Administrar datos de estudiantes y el enlace Buscar estudiante. Se muestra una página con cuadros de selección. Seleccione los campos de búsqueda en los cuadros de selección que se muestran. Presione el botón aceptar. Se muestra una nueva página con los campos de búsqueda seleccionados. Ingrese los datos en los cuadro de texto que se muestran según la selección que hizo en el paso anterior. Presione el botón aceptar. Se observa una tabla con los resultados obtenidos según la búsqueda realizada.

Fuente: elaboración Propia.

.

Conclusiones

1. SACMO: es un novedoso sistema que permite almacenar, seleccionar y generar rápida y efectiva respuesta a las solicitudes poder obtener historias médicas, odontograma, citas médicas, tratamientos, Ordenes Medicas., por tal motivo el mantenimiento se realice en un tiempo muy corto.

2. Logrando una correcta implementación del Proceso Unificado Racional (RUP) se obtuvo la definición, diseño y modelado del sistema SACMO. Las iteraciones que incluye esta metodología junto al flujo de trabajo planteado por la misma permitieron el desarrollo de un proyecto que diera con la solución a los problemas del cliente. Los distintos diagramas considerados tanto de WebML como de UML, como herramienta conjunta de desarrollo, hicieron posible la realización de las diferentes fases establecidas por la metodología del proceso unificado.

3. El análisis del sistema actual a través de los diferentes diagramas, facilitó la determinación de los requerimientos ya que a través de ellos se realizó la identificación, definición y recolección de los verdaderos requisitos necesarios para el desarrollo de la aplicación, que se centran en el usuario y sus necesidades.

4. Se logró establecer una arquitectura del sistema sólida y lista para ser puesta en práctica debido a que los diagramas WebML permitieron concretar los flujos y vías con la se trabajaran los datos para generar una información veraz al usuario. Por lo anteriormente descrito y junto a los diagramas de casos de uso implementados se llevó a cabo la fase de elaboración.

5. El adecuado diseño del modelo de la base de datos resultó fundamental en el desarrollo del software, ya que el mismo sirvió de base para erigir la arquitectura del sistema.

6. La utilización de software libre, cumpliendo con el decreto 3.390, es de gran importancia, ya que se obtiene software de calidad sin necesidad de comprar licencias. Además la utilización de PHP como lenguaje de programación para páginas Web, permitió el desarrollo de la aplicación de una manera más simple y efectiva.

7. Con la utilización de las tecnologías AJAX para la programación de páginas web, junto con la librería Ext-JS y el manejador de bases de datos MySQL, permitieron la creación del sistema de una manera más sencilla, clara y efectiva, debido a que estas tecnologías son de fácil entendimiento e implementación y se adaptan a cualquier entorno computacional.

8. La correcta codificación del SACMO dependió en gran parte de un correcto diseño de su arquitectura.

9. El poder tener todo lo referente a la conexión con la base de datos separado del resto del código permitió que el desarrollo de la aplicación fuera más fácil, incluyendo el mantenimiento de la misma.

Recomendaciones

1. Realizar todas las actividades pertinentes a la implantación del sistema en los servidores en el área de servicios médicos adscrito al área de salud de la Universidad de Oriente núcleo Anzoátegui.

2. La aplicación SACMO fue creada pensando en una posible masificación, por tanto se recomienda que realmente se implemente en el área de salud de la universidad de Oriente. Núcleo de Anzoátegui.

3. Llevar a cabo la debida promoción del sistema, para la participación total de los usuarios y obtener de esta forma una data siempre actualizada.

4. Planificar jornadas de adiestramiento al personal destinado a la utilización del sistema SACMO.

5. Se recomienda realizar periódicamente un respaldo de la base de datos del sistema, así como también el mantenimiento respectivo, debido a que con este software se maneja gran cantidad de información.

Se recomienda que, en caso de realizarse la actualización de SACMO, inspeccionar el código de los componentes y, de ser posible, optimizar los mismos

Bibliografía

Adrian, E. M. (2004). Desrrollo de un sistema para la automatizacion del proceso de administracion y Contol de estudio en un instituto univercitario. Barcelona, Venezuela: UDO.

Bárcenas, U. B., & Cedeño, F. E. (2001). Diseño de un plan estrategueico para el departamento de reguistro Médicos y estadisticas del hospital Universitario Dr Luis Razetti. Barcelona,Venezuela: UDO.

Conallen, J. (2002). *Building Web Application With UML*. Madrid: Addison Wesley.

Díaz Cova, M. G. (2003). Diseño de un sistema de facturación para el manejo de las actividades administrativas que realizan en una clinica Municipal. Barcelona, Venezuela : UDO.

Giampaolo R, J., & Rigual, C, R. J. (2003). Diseño de un sistema de planificación estrategueica gerencial de un centro de un centro de especialidades Médicas para el mejoramiento operativo. Barcelona,Venezuela: UDO.

Jacobson, L. B., & Rumbaugh, J. (2000). *El Proceso Unificado de Desarrollo de Software.* Madrid: Editorial Pearson Educacion.

Jaime, O. (1990). Administracion en Odontologia. Lima.

Josi, C. (s.f.). *Ingenieria de Software Orientada a Objeto.* Recuperado el 2009, de http://w.w.w. monografías .com /trabajos10/soft/soft.shtml,

Joyanes , l. (1998). *Programacion Orientada a Objetos.* España: Mc Graw Hill.

Mateu, C. (2004). *Desarrollo de Aplicaciones Web.* Barcelona, España: Eureka Media.

Mh, R. (1985). *Materiales detales de Odontología Clinica.* Madrid: EL manual Moderno.

Micrsoft. (2001). *Diccionario de Informatica e Internet.* Madrid: McGraw-Hill.

Navathe, S. B., & Elmasr, R. (2000). *Sistemas de Base deDatos conceptos Fundamentales.* Mexico: Pearson.

Pietrek, J. y. (1993). El ABC de gestión en consultas la Clinica Estomatológica como empresa II. Esopaña: Rev. Quintesesence.

Sanz, P. A. (2009). *Introduccion al UML*. Obtenido de http://www.yoprograma.com/articulo4.php

Schach, A. (2005). Analisis y Diseño Orientado a Objeto con UML y el Proceso Unificado. Madrid : Mc-Graw-Hill.

Simoni, G. M. (2004). Desarrollo de un sistema de informacion para la Automatizacion los procesos realizados en el departamento de ciencias de la unidad de cursos Básicos de la Unioversidad de Oriente Núcleo de Anzoategui. BVarcelona Venezuela: UDO.

Stallman, C. Y. (2003). *Designing Data Intensive Web Applications.* Milano Italia: Morgan Kaufmann Publishers.

Stefano, C., & Fraternali,, P. (2003). *Designing Data Intensive Web Applications.* Milano Italia: MorganKaufmann.

Suehring, S. (2002). *MysQL Biblie.* Estados Unidos: Wiley Publishing.

Verse, Y., & Park, J. (2004). *PHP5 and MySQL Bible.* United States of America: Wiley ublishing .

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

TÍTULO	**"DESARROLLO DE UN SOFTWARE DE AUTOMATIZACIÓN PARA EL CONTROL DE LAS ACTIVIDADES DEL ÁREA DE SALUD, ADSCRITA A LA DELEGACIÓN ESTUDIANTIL, NÚCLEO ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"**
SUBTÍTULO	

AUTOR (ES):

APELLIDOS Y NOMBRES	**CÓDIGO CULAC / E MAIL**
Vázquez B Ricardo J	**CVLAC: 11.173.747.** **E MAIL: rcsvazquez@Hotmail.com**
Sequea A Luisandro J.	**CVLAC:12.438.833** **E MAIL:jaavier@gmail.com**
	CVLAC: **E MAIL:**
	CVLAC: **E MAIL:**

PALABRAS O FRASES CLAVES:

UML, WEBML, RUP, PHP, MySQL, EXTJS, ACTIVIDADE

PASANTÍAS, TESIS, ELEGIBLES, PDVSA, SELECCIÓN

CAPTACIÓN

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ÀREA	SUBÀREA
INGENIERÍA Y CIENCIAS APLICADAS	INGENIERÍA EN COMPUTACIÓN

RESUMEN (ABSTRACT):

La delegación de Bienestar Estudiantil realiza todas las actividades del área de salud de forma manual, trayendo como consecuencia el retraso en el procesamiento de la información, por tal motivo se propuso el proyecto de Grado titulado: **"DESARROLLO DE UN SOFTWARE DE AUTOMATIZACIÓN PARA EL CONTROL DE LAS ACTIVIDADES DEL ÁREA DE SALUD, ADSCRITA A LA DELEGACIÓN ESTUDIANTIL, NÚCLEO ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"** el cual es un software de gestión basado en tecnología Web que permite a los usuarios realizar las operaciones necesarias para tal fin. Dicho sistema lleva por nombre SACMO, El desarrollo del proyecto estuvo basado en el Proceso Unificado de desarrollo de software, la herramienta UML y su extensión WEBML, dividiendo el proyecto en cuatro fases y estas en flujos de trabajo. Para la implementación se utilizaron las herramientas de PHP y MYSQL cumpliendo con las normas de la empresa, en cuanto al desarrollo de aplicaciones con software libre de acuerdo al decreto 3.390.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

CONTRIBUIDORES:

APELLIDOS Y NOMBRES	ROL / CÓDIGO CVLAC / E_MAIL				
Cortínez N., Claudio A.	ROL	CA	AS	TU X	JU
	CVLAC:	12.155.334			
	E_MAIL	Cl_cortinez@cantv.net			
	E_MAIL				
Mujica Yulitza	ROL	CA	AS	TU	JU X
	CVLAC:	12.576.448			
	E_MAIL	yulitzamujika@cantv.net			
	E_MAIL				
Rodríguez Rhonald	ROL	CA	AS	TU	JU X
	CVLAC:	14.077.185			
	E_MAIL	Rhoen2003@hotmail.com			
	E_MAIL				
Douglas Gened	ROL	CA	AS X	TU	JU X
	CVLAC:	12			
	E_MAIL	gpdouglas02@gmail.com			
	E_MAIL				

FECHA DE DISCUSIÓN Y APROBACIÓN:

2010	08	13
AÑO	MES	DÍA

LENGUAJE.SPA.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ARCHIVO (S):

NOMBRE DE ARCHIVO	TIPO MIME
TesisFinal.doc	Application/msword

CARACTERES EN LOS NOMBRES DE LOS ARCHIVOS: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z. a b c d e f g h i j k l m n o p q r s t u v w x y z. 0 1 2 3 4 5 6 7 8 9.

ALCANCE

ESPACIAL: _____ (**OPCIONAL**)

TEMPORAL _____(OPCIONAL)

TÍTULO O GRADO ASOCIADO CON EL TRABAJO:

Ingeniero en Computación.

NIVEL ASOCIADO CON EL TRABAJO

Pre-Grado.

ÁREA DE ESTUDIO:

Computación y Sistemas

INSTITUCIÓN:

Universidad de Oriente Núcleo Anzoátegui.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

DERECHOS

De acuerdo al artículo 44 del reglamento del trabajo de Grado:

"Los trabajos de grado son de propiedad exclusiva de la Universidad y sólo podrán ser utilizados a otros fines con el consentimiento del Consejo de Núcleo respectivo, quién lo participará al Consejo Universitario".

Ricardo Jesús, Vázquez Bacadare.
AUTOR

Luisandro J Sequea. A
AUTOR

Claudio Cortínez
TUTOR

Yulitza Mujica
JURADO

Rhonald Rodríguez
JURADO

POR LA SUBCOMISION DE TESIS

José Bastardo