MODELO DE CREACIÓN DE SERVICIOS DE TELEMEDICINA BASADO EN EL CONCEPTO DE RED INTELIGENTE - ANEXO

CLAUDIA MILENA HERNÁNDEZ BONILLA
CARLOS HERNÁN TOBAR ARTEAGA

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE ELECTRÓNICA, INSTRUMENTACIÓN Y CONTROL
POPAYÁN
2004

MODELO DE CREACIÓN DE SERVICIOS DE TELEMEDICINA BASADO EN EL CONCEPTO DE RED INTELIGENTE - ANEXO

CLAUDIA MILENA HERNÁNDEZ BONILLA
CARLOS HERNÁN TOBAR ARTEAGA

Director
Mg. RAFAEL RENGIFO PRADO

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE ELECTRÓNICA, INSTRUMENTACIÓN Y CONTROL
POPAYÁN
2004

TABLA DE CONTENIDO

Pág.

LISTA DE FIGURAS

Pág.

1. JAVA 2 PLATFORM ENTERPRISE EDITION - J2EE

La especificación de J2EE provee una aproximación basada en componentes para el diseño, desarrollo, ensamblaje y despliegue de aplicaciones empresariales. La plataforma J2EE ofrece un modelo de aplicación distribuido multicapas, componentes reutilizables, un modelo de seguridad unificado, control de transacciones flexibles y soporte para servicios web a través del intercambio de datos integrados en XML.

Algunos servicios de J2EE son HTTP/HTTPS, Java Transaction API, JDBC, Java Message Service, Java Authentication and Authorization Service, J2EE Connector Architecture, Java API for XML Parsing, RMI-IIOP, JavaIDL, JavaMail, y Java Activation Framework.

1.1. COMPONENTES DE UNA APLICACIÓN J2EE

En una aplicación J2EE la lógica de la aplicación está divida en componentes según la funcionalidad, y cada componente se instala en diferentes máquinas dependiendo de la capa de la aplicación a la que pertenezca. Como se ilustra en la figura 1.1. una aplicación J2EE típica está compuesta por:

- Componentes de la capa del cliente corriendo en la máquina del cliente.
- Componentes de la capa web corriendo en el servidor J2EE.
- Componentes de la capa del negocio corriendo en el servidor J2EE.
- Software de la capa del sistema de información empresarial (SIE) corriendo en el servidor SIE.

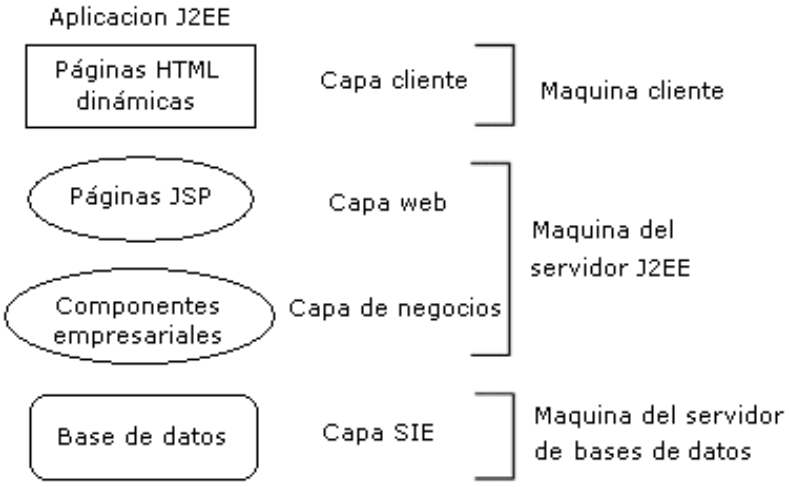

Figura 1.1. Aplicación J2EE

Clientes J2EE: un cliente puede ser un cliente web o un cliente de aplicación. Un cliente web tiene dos partes, paginas web dinámicas que son generadas por componentes web que corren en la capa web y un navegador web que renderiza las páginas recibidas desde el servidor. Un cliente de aplicación corre en una maquina cliente y facilita los mecanismos para que los usuarios ejecuten tareas a través de una interfaz de usuario gráfica, estos clientes pueden acceder directamente a los beans que corren en la capa de negocio, sin embargo también puede establecer comunicación con servlets de la capa web.

Componentes web: un cliente web puede ser un servlet o una página JSP. Los servlets son clases Java que dinámicamente procesan peticiones y construyen respuestas; una página JSP es un documento basado en texto que contiene dos tipos de texto: una plantilla de datos estática que puede expresarse en formatos HTML, XML, WML y elementos JSP que determinan como la página construye el contenido dinámico.

Componentes de negocio: el código del negocio, el cual es la lógica que resuelve las necesidades de un determinado dominio de aplicación es manejado por los beans empresariales

de la capa de negocio. Existen tres clases de beans empresariales: beans de entidad, de sesión y manejados por mensajes.

Un bean de sesión representa una conversación transiente con un cliente, cuando el cliente finaliza su ejecución, el bean de sesión y sus datos son eliminados. Un bean de entidad representa datos persistentes almacenados en una fila de una tabla de una base de datos, si el cliente termina su ejecución o si el servidor se cae, el bean de entidad es guardado. Finalmente un bean dirigido por mensajes combina los servicios de un bean de sesión y un oyente JMS que permite a un cliente de negocio recibir mensajes JMS asincrónicamente.

Los componentes se instalan en contenedores desde los que pueden utilizar los servicios de la plataforma. El Servidor J2EE provee el ambiente de ejecución, los servicios requeridos y contenedores para alojar los componentes. Los contenedores gestionan el ciclo de vida de los componentes, y los tipos de contenedores J2EE son:

- Contenedor de EJBs: gestiona la ejecución de EJBs en un servidor J2EE.
- Contenedor web: gestiona la ejecución de las páginas JSP y servlets en un servidor J2EE.
- Contenedor de aplicación cliente: gestiona la ejecución de los componentes de aplicación cliente en una máquina cliente.
- Contenedor de applet: navegador con plug-in Java en una máquina cliente.

Sistema de Información empresarial: es el software del sistema de información empresarial e incluye sistemas de infraestructuras empresariales tales como planeación de recursos empresariales, sistemas de bases de datos, mainframe para procesamiento de transacciones y otros sistemas de información propietarios.

1.2. EMPAQUETAMIENTO EN J2EE

Una aplicación J2EE es entregada en un archivo empresarial ear, un archivo Java estándar con una extensión .ear que contiene varios módulos J2EE. La utilización de archivos ear hace posible ensamblar un número de aplicaciones J2EE diferentes usando los mismos componentes.

Un módulo J2EE contiene uno o más componentes J2EE para el mismo tipo de contenedor y un documento XML, llamado descriptor de despliegue, donde se describen las características de despliegue de cada componente. Existen cuatro tipos de módulos J2EE:

- Módulos EJB: contienen los archivos de clases de los beans empresariales y un descriptor de despliegue EJB, son empaquetados como archivos JAR con extensión .jar.
- Módulos web: contienen archivos JSP, archivos de clases de los servlets, gifts y html y un descriptor de despliegue web, son empaquetados como archivos JAR con extensión .war.
- Módulos adaptadores de recursos: contienen todas las interfaces Java, clases, librerías nativas y otra documentación, además del descriptor de despliegue del adaptador de recursos. Son empaquetados como archivos JAR con extensión .rar.
- Módulos cliente de aplicación: contienen archivos de clases y el descriptor del cliente, son empaquetados como archivos JAR con extensión .jar.

2. EL LENGUAJE DE MODELADO UNIFICADO - UML

El Lenguaje de Modelado Unificado –UML, es un lenguaje estándar de la industria utilizado para la visualización, especificación, construcción y documentación de los artefactos de un sistema software. Tuvo su origen en la década de los 90's gracias a la colaboración de tres destacados metodologístas: Grady Booch, Ivar Jacobson y James Rumbaugh. En 1995 Booch y Rumbaugh buscaron unificar sus métodos para formar el Método Unificado v. 0.8, y un año más tarde, Jacobson colabora en la tarea de unificar sus lenguajes de modelado en UML 0.9.

UML cosechó los beneficios y asumió las responsabilidades de su origen privilegiado. Los usuarios rápidamente reconocieron las ventajas de un lenguaje de modelado común que podía ser utilizado para visualizar, especificar, construir y documentar los artefactos de un sistema software; rápidamente se acogieron y aplicaron los "drafts" iniciales del lenguaje a diversos dominios, desde las finanzas y la salud hasta las telecomunicaciones y la industria aeroespacial. De esta manera, gracias a la fuerte demanda de los usuarios, los vendedores de herramientas de modelado pronto incluyeron soporte para UML en sus productos.

Al mismo tiempo que UML fue siendo un estándar industrial de facto, un equipo internacional de expertos en modelado asumieron la responsabilidad de convertir el lenguaje en un estándar formal. Así, en 1996 se empezó a trabajar con "Los tres amigos", Booch, Jacobson y Rumbaugh, para proponer UML como el lenguaje de modelado estándar para la OMG. Se organizó un equipo de desarrollo de software, el cual siguió un proceso disciplinado, cuyo ciclo de vida de desarrollo fue iterativo e incremental, con el fin de producir progresivamente la serie de especificaciones que conforman UML.

Los enfoques iniciales fueron en mejorar la arquitectura y formalismo del lenguaje, y asegurar que fuera completamente de propósito general. También se definieron facilidades para intercambiar modelos UML entre herramientas y un lenguaje opcional para restricciones. La propuesta inicial de UML a la OMG (UML1.0) se realizó en Enero de 1997, y después de nueve meses de mejoras intensivas a la especificación, entregaron su propuesta final (UML1.1) en septiembre de 1997, la cual la OMG oficialmente adoptó como su estandar de modelamiento de objetos en noviembre de 1997.

Desde el origen de UML ha existido un paulatino y ordenado proceso de revisión y estandarización, su versión actual es la 2.0. Es importante destacar que esta versión incluye elementos para soportar la especificación de componentes.

A continuación se presentan las vistas definidas en UML y los diagramas que permiten describirlas, para esto se utiliza como ejemplo una aplicación de simulación quirúrgica basada en realidad virtual. Esta aplicación consiste de un actuador físico que replica los movimientos y cambios realizados en una réplica tridimensional del mismo.

2.1. VISTAS DE UML

Una vista es un subconjunto de UML que modela construcciones que representan un aspecto de un sistema. Las vistas se pueden dividir en tres áreas: clasificación estructural, comportamiento dinámico, y gestión del modelo. La clasificación estructural describe los elementos del sistema y sus relaciones con otros elementos, aquí se incluye la vista estática, la vista de casos de uso y la vista de implementación. El comportamiento dinámico describe el comportamiento de un sistema en el tiempo, incluye la vista de máquinas de estados, la vista de actividad y la vista de interacción. La gestión del modelo describe la organización de los propios modelos en unidades jerárquicas, aquí el paquete es la unidad genérica de organización para los modelos.

UML también contiene varias construcciones previstas para proporcionar una capacidad limitada, pero útil, de extensión. Estas construcciones incluyen restricciones, estereotipos y valores etiquetados y son aplicables a los elementos de todas las vistas.

A continuación se describen cada una de las vistas.

2.1.1. Vista estática

Esta vista modela los conceptos del dominio de la aplicación, así como los conceptos internos creados como parte de la implementación de la aplicación (solución). Esta visión es estática porque no describe el comportamiento del sistema dependiente del tiempo. Los componentes principales de esta vista son las clases y sus relaciones: asociación, generalización, y varias clases de dependencia, tales como realización y uso. La vista estática es descrita mediante diagramas de clases. Adicionalmente, es importante considerar en esta vista la definición de componentes.

Las clases se dibujan como rectángulos que incluyen la lista de las operaciones y los atributos, y las relaciones entre clases se dibujan como las líneas que conectan las clases. La figura 2.1 muestra un diagrama de clases para la aplicación de ejemplo.

Un componente es definido como una unidad modular con interfaces bien definidas que es reemplazable en su ambiente. El concepto de componente dirige el área de desarrollo basado en componentes y la estructuración de sistemas basados en componentes, donde un componente es modelado a través del ciclo de vida de desarrollo y sucesivamente refinado en despliegue y ejecución.

Un aspecto importante del desarrollo basado en componentes es el reuso de componentes previamente construidos. Un componente puede ser considerado como una unidad autónoma dentro de un sistema, tiene una o más interfaces provistas y requeridas. Aunque un componente

puede depender de otros elementos en términos de interfaces que son requeridas, un componente es encapsulado y sus dependencias son diseñadas de tal forma que pueda ser tratado tan independientemente como sea posible. Como resultado, los componentes y subsistemas pueden ser flexiblemente reutilizados y reemplazados por conexión (ensamblaje) entre ellos por medio de sus interfaces provistas y requeridas.

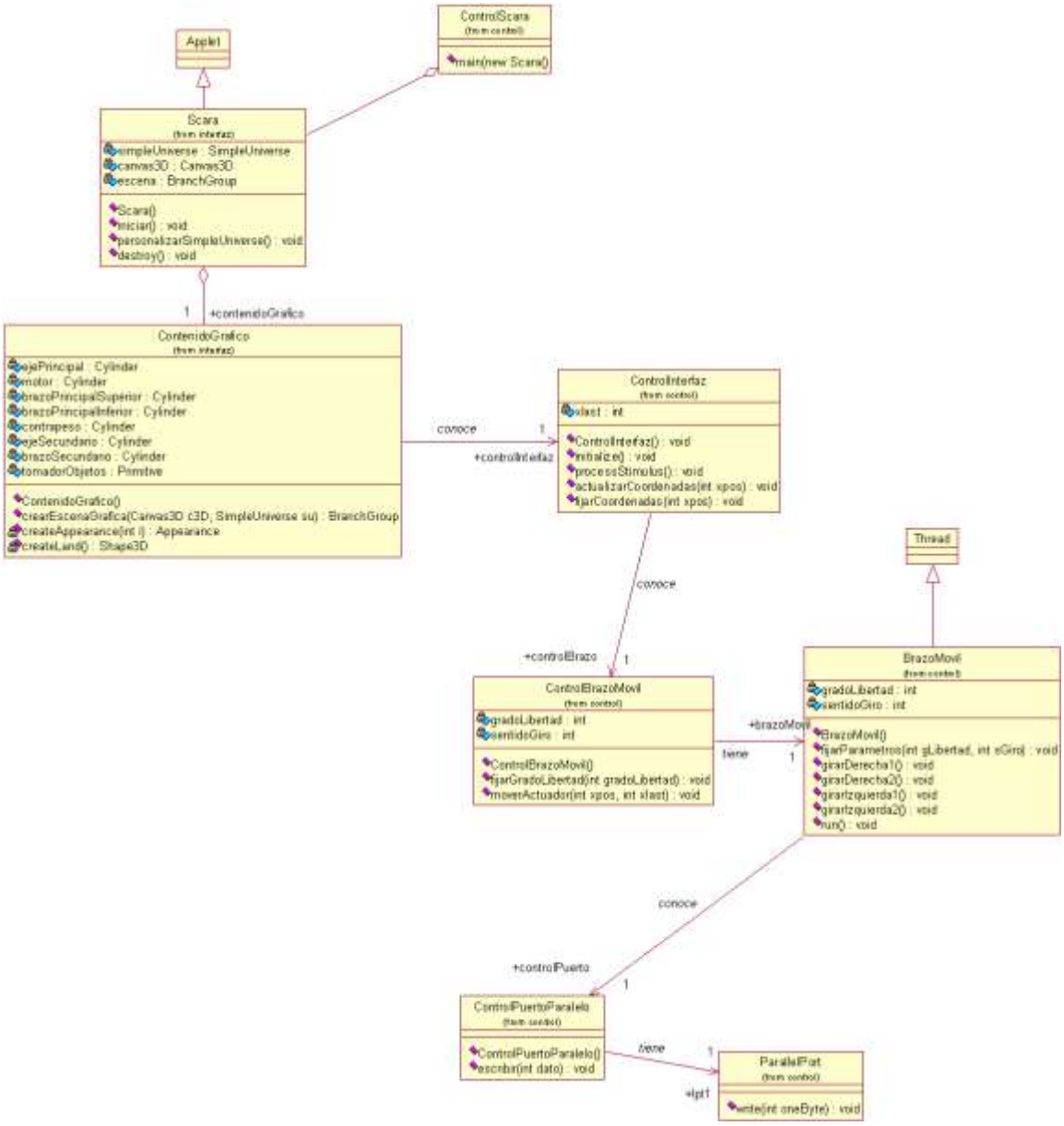

Figura 2.1. Diagrama de clases

Un componente se representa con un rectángulo con la palabra clave "component". Opcionalmente en la esquina superior derecha un icono de componente puede ser fijado. La figura 2.2 muestra el ejemplo de un componente denominado Visor3D que ofrece las interfaces Visualización y Manipulación, y requiere las interfaces Comunicación TCP/IP y Comunicación Pto Paralelo.

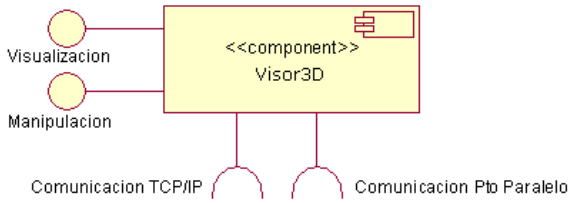

Figura 2.2. Representación de un Componente

2.1.2. Vista de casos de uso

Esta vista modela la funcionalidad del sistema según lo perciben los usuarios externos, llamados actores. Un caso de uso es una unidad coherente de funcionalidad, expresada como transacción entre los actores y el sistema. El propósito de la vista de casos de uso es enumerar a los actores y los casos de uso, y mostrar qué actores participan en cada caso de uso. En la figura 2.3 se presenta un diagrama de casos de uso para la aplicación de ejemplo.

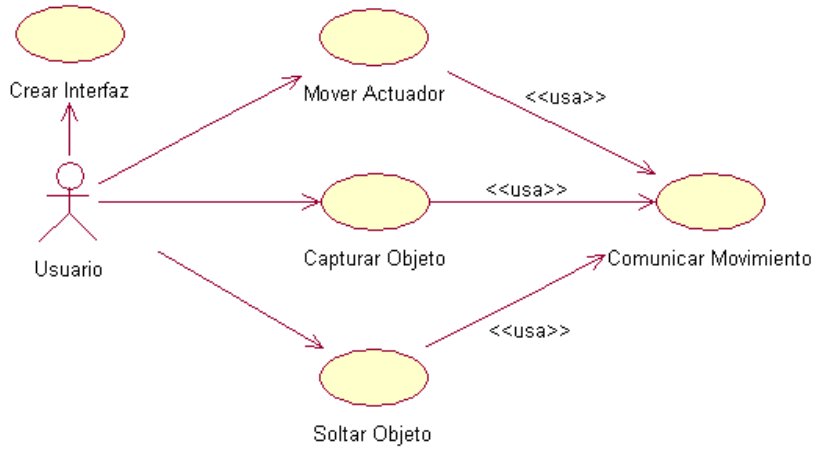

Figura 2.3. Diagrama de casos de uso

2.1.3. Vista de interacción

Esta vista describe secuencias de intercambios de mensajes entre los roles que implementan el comportamiento de un sistema. Un rol de clasificador, o simplemente "rol", es la descripción de un objeto, que desempeña un determinado papel dentro de una interacción, distinto de los otros objetos de la misma clase. Esta visión proporciona una vista integral del comportamiento de un sistema, es decir muestra el flujo de control a través de muchos objetos. La vista de interacción se representa mediante dos diagramas centrados en distintos aspectos: el diagrama de secuencia y el diagrama de colaboración.

Un diagrama de secuencia muestra un conjunto de mensajes, dispuestos en una secuencia temporal. Cada rol en la secuencia se muestra como una línea de vida, es decir, una línea vertical que representa el rol durante cierto plazo de tiempo, con la interacción completa. Los mensajes se muestran como flechas entre las líneas de vida. Un diagrama de secuencia puede mostrar un escenario, es decir una historia individual de una transacción. La figura 2.4 muestra el diagrama de secuencia para el caso de uso mover actuador.

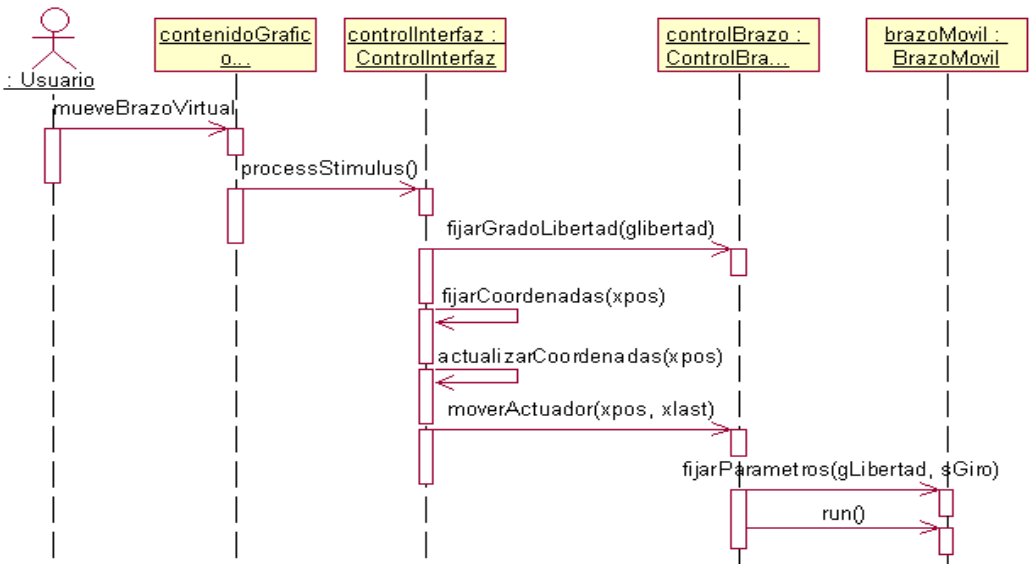

Figura 2.4. Diagrama de secuencia de mensajes

Un diagrama de colaboración modela los objetos y los enlaces significativos dentro de una interacción. Los objetos y los enlaces son significativos solamente en el contexto proporcionado por la interacción. Un rol describe un objeto, y un rol en la asociación describe un enlace dentro de una colaboración. Un diagrama de colaboración muestra los roles en la interacción en una disposición geométrica (figura 2.5). Los mensajes se muestran como flechas, ligadas a las líneas de la relación, que conectan a los roles. La secuencia de mensajes, se indica con los números secuenciales que preceden a las descripciones del mensaje.

Un diagrama de colaboración muestra la implementación de una operación. La colaboración muestra los parámetros y las variables locales de la operación, así como asociaciones más permanentes. Cuando se implementa el comportamiento, la secuencia de los mensajes corresponde a la estructura de llamadas anidadas el paso de señales al programa. Sin embargo, también puede ser utilizado para mostrar un caso de uso. La figura 2.5 muestra un diagrama de colaboración para el caso de uso mover actuador.

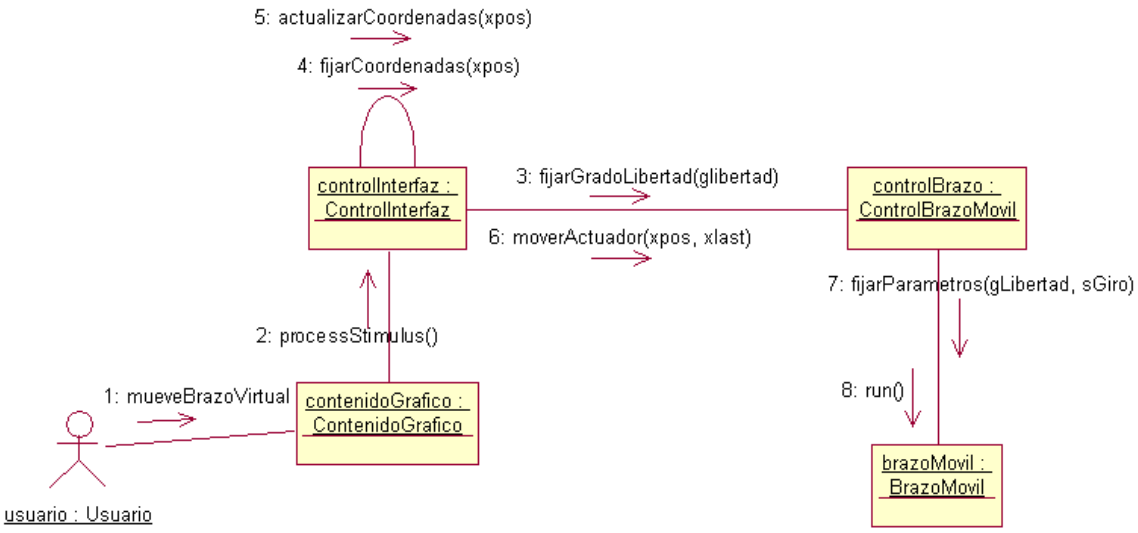

Figura 2.5. Diagrama de colaboración

2.1.4. Vista de máquina de estados

Una máquina de estados modela las posibles historias de vida de un objeto de una clase. Una máquina de estados contiene los estados conectados por transiciones. Cada estado modela un período de tiempo, durante la vida de un objeto, en el que satisface ciertas condiciones. Cuando ocurre un evento, se puede desencadenar una transición que lleve el objeto a un nuevo estado. Cuando se dispara una transición, se puede ejecutar una acción unida a la transición. Las máquinas de estados se muestran como diagramas de estados, como el de la figura 2.6.

Las máquinas de estados se pueden utilizar para describir interfaces de usuario, controladores de dispositivo, y otros subsistemas reactivos. También pueden usarse para describir los objetos pasivos que pasan por varias fases cualitativas distintas, durante su tiempo de vida, cada una de las cuales tiene su propio comportamiento especial.

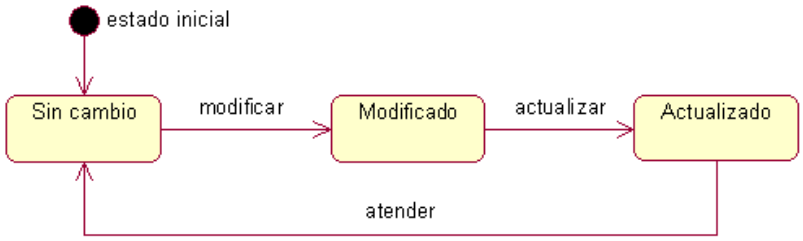

Figura 2.6. Diagrama de estados

2.1.5. Vista de Actividades

Un grafo de actividades es una variante de una máquina de estados, que muestra las actividades de computación implicadas en la ejecución de un cálculo. Un estado de actividad representa una actividad: un paso en el flujo de trabajo o la ejecución de una operación. Un grafo de actividades describe grupos secuenciales y concurrentes de actividades. Los grafos de actividades se

muestran en diagramas de actividades. La figura 2.7 muestra un diagrama de actividades para el caso de uso mover actuador.

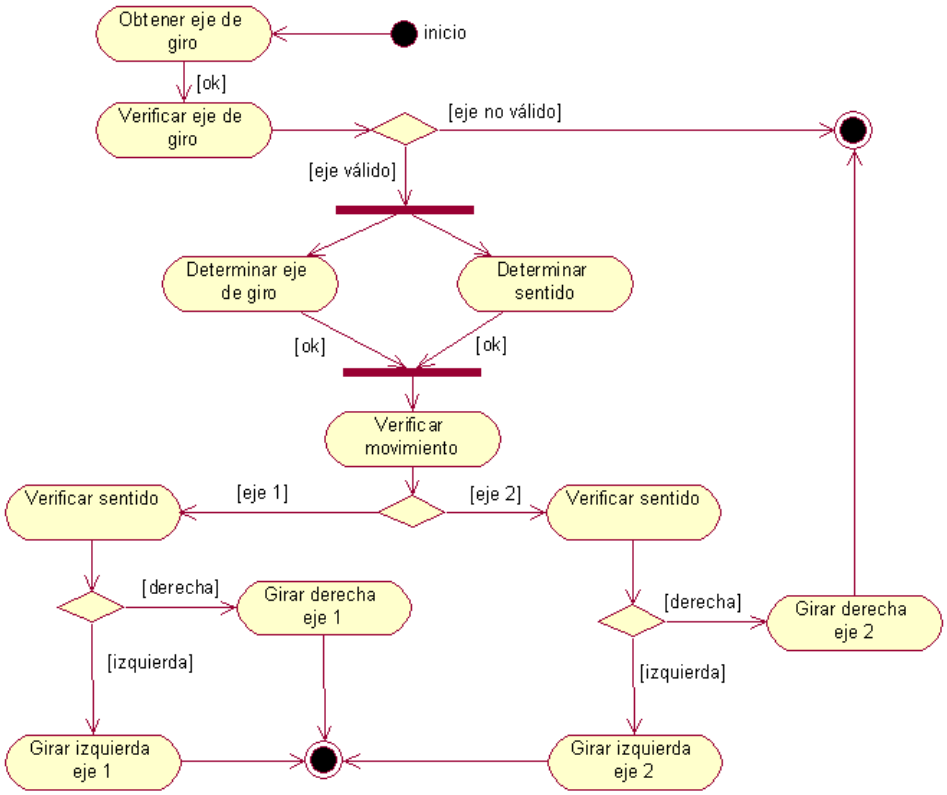

Figura 2.7. Diagrama de actividades

2.1.6. Vista física

En estas vista se modelan la estructura de la implementación de la aplicación, su organización en componentes, y su despliegue en nodos de ejecución. Estas vista proporcionan una oportunidad de establecer correspondencia entre las clases, los componentes de implementación y los nodos. Se identifican dos tipos de vistas físicas, la vista de implementación y la vista de despliegue.

La vista de implementación modela los componentes de un sistema, a partir de los cuales se construye la aplicación, así como las dependencias entre los componentes, para poder determinar el impacto de un cambio propuesto. También modela la asignación de clases y de otros elementos del modelo a los componentes. La vista de implementación se representa en diagramas de componentes. La figura 2.8 muestra el diagrama de componentes de la aplicación de simulación quirúrgica.

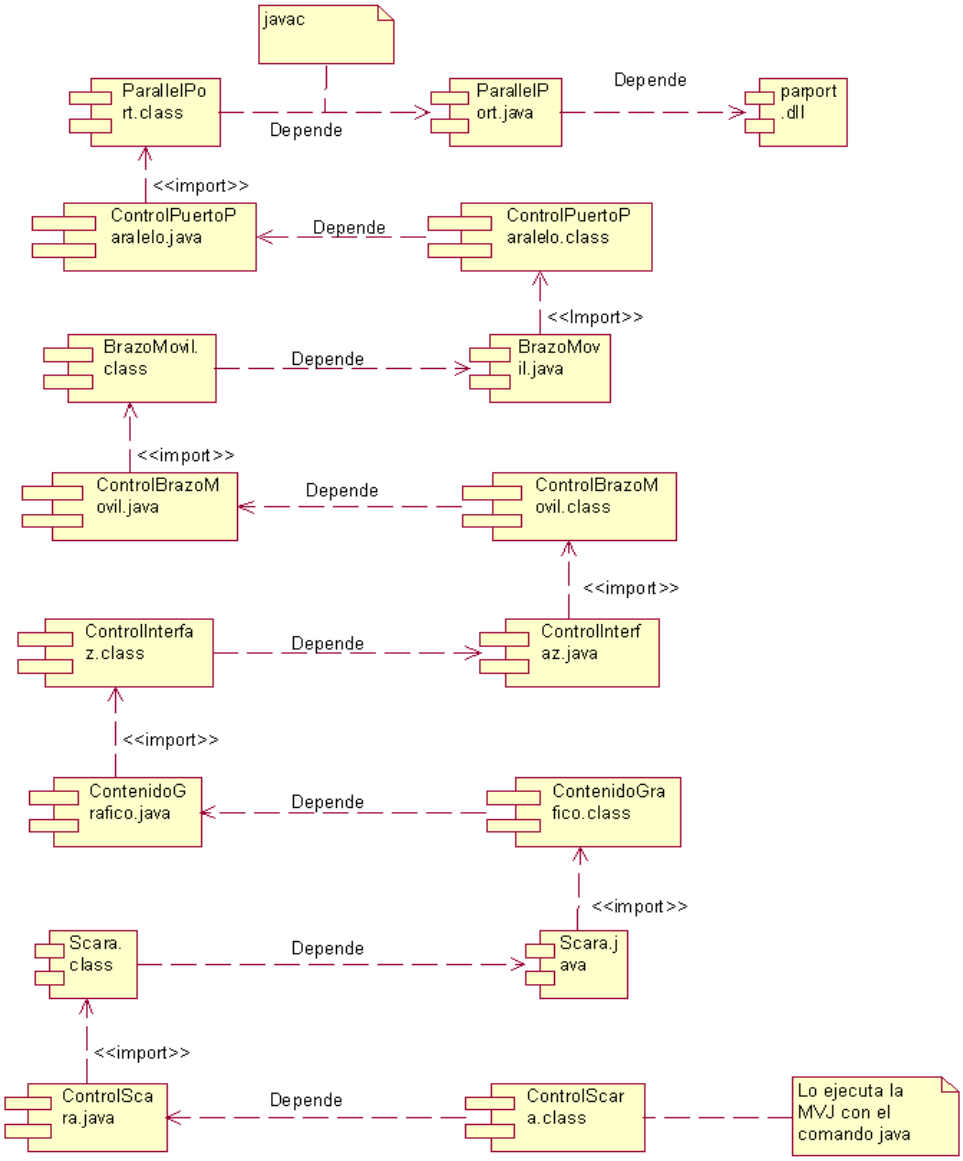

Figura 2.8. Diagrama de componentes

El diagrama de componentes muestra los tipos de componentes del sistema; una configuración particular de la aplicación puede tener más de una copia de un componente. Un círculo pequeño con un nombre es una interfaz, un conjunto coherente de servicio. Una línea sólida que va desde un componente a una interfaz, indica que el componente proporciona los servicios de la interfaz. Una flecha con guiones de un componente a una interfaz indica que el componente requiere los servicios proporcionados por la interfaz.

La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos. Un nodo es un recurso de ejecución, tal como una computadora, un dispositivo o memoria. Esta vista permite determinar las consecuencias de distribución y de asignación de recursos. La vista de despliegue se representa en diagramas de despliegue. La figura 2.9 muestra un diagrama de despliegue para la aplicación de ejemplo.

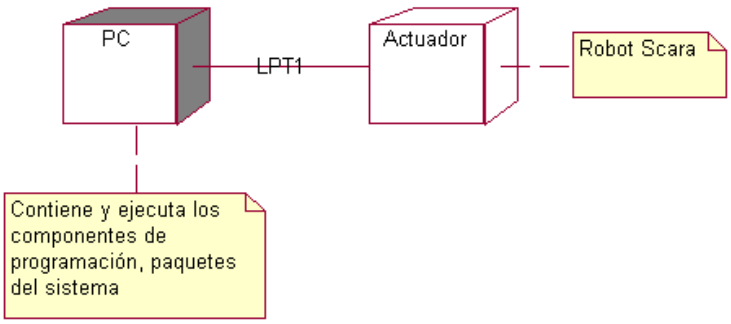

Figura 2.9. Diagrama de despliegue

2.1.7. Vista de gestión del modelo

Esta vista modela la organización del modelo en sí mismo. Un modelo abarca un conjunto de paquetes que contienen los elementos del modelo, tales como clases, máquinas de estados, y casos de uso. Los paquetes pueden contener otros paquetes: por lo tanto, un modelo señala un paquete raíz que contiene indirectamente todo el contenido del modelo. Los paquetes son

unidades para manipular el contenido de un modelo, así como unidades para el control de acceso y el control de configuración. Cada elemento del modelo pertenece a un paquete o a otro elemento.

Un modelo es una descripción completa de un sistema, con una determinada precisión, desde un punto de vista. Puede haber varios modelos de un sistema desde distintos puntos de vista; por ejemplo, un modelo de análisis y un modelo de diseño. Un modelo se representa como una clase especial de paquete. Un subsistema es otro paquete especial. Representa una porción de un sistema, con una interfaz perfectamente determinada, que puede ser implementado como un componente distinto. Generalmente, la información de gestión del modelo se representa mediante diagramas de clases.

La figura 2.10 muestra la descomposición de la totalidad del sistema de ejemplo en paquetes y sus relaciones de dependencia.

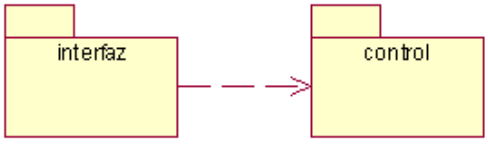

Figura 2.10. Diagrama de paquetes

3. SERVICIO DE MEDICINA PREVENTIVA Y EDUCACIÓN A LA POBLACIÓN INFANTIL

La realidad virtual es un área poco explorada en Latinoamérica que ofrece capacidades innovadoras no soportadas por ninguna otra tecnología y que representa un aspecto revolucionario en la percepción de los servicios de telemedicina. La tecnología de realidad virtual tiene el potencial para mejorar la forma en que las personas interactúan con los servicios de telemedicina, representa una interfaz de comunicación avanzada que permite una forma intuitiva de interactuar con la información y constituye un ambiente flexible que mejora la percepción de presencia física durante dicha interacción. Así, la VR puede ser utilizada en diversos servicios de telemedicina como son: asistencia remota, educación a distancia, tratamiento de pacientes con desordenes psiquiátricos, visualización biológica, etc.

Este anexo ilustra la aplicación del modelo en la creación de un servicio de medicina preventiva y educación a la población infantil, utilizando realidad virtual. Este servicio consiste en un entorno virtual (objetos tridimensionales localizados en un espacio tridimensional) del tipo inmersivo, gracias a la posibilidad de navegación (desplazamiento) del usuario y experiencia de interacción con los objetos virtuales que buscan ser representaciones tridimensionales del mundo real, así como de la incorporación de luces y sonidos que dan la sensación de realismo. Este entorno virtual tiene como objetivo primordial captar la atención de los usuarios, que en primera instancia se ha escogido sean niños. Los cuales, en la acción de explorar el entorno virtual descubran presentaciones audio-visuales que desplieguen contenidos de aprendizaje relacionados con temas de prevención en salud o cuidado personal. En este caso se ha escogido impartir dos presentaciones didácticas acerca de higiene oral y nutrición.

3.1. PLANO DE SERVICIOS

3.1.1. Caracterización del servicio de telemedicina

a) Información general del servicio de telemedicina:

Servicio: Medicina preventiva y educación a pacientes.

Objetivo: Servir como herramienta educativa a la población infantil en temas de salud y auto-cuidado.

Actores: Población infantil.

Descripción:

- El servicio ofrece un entorno virtual que puede ser recorrido de forma natural e intuitiva.
- El entorno virtual está compuesto por un conjunto de objetos visuales tridimensionales.
- Los objetos visuales tridimensionales se caracterizan por poseer atributos como color, textura, luminosidad, sonido y comportamiento.
- En el entorno virtual se disponen de pantallas de proyección donde se muestra en forma de presentación imágenes relacionadas con temas de salud y auto-cuidado.
- Los temas de salud y auto-cuidado son explicados mediante un mecanismo de audio.
- La proyección de las imágenes y la ejecución del audio se inician cuando los usuarios interceptan un área de influencia.
- La proyección de las imágenes y la ejecución del audio se detiene cuando los usuarios salen del área de influencia.
- En el entorno virtual se cuenta con objetos visuales que representan animaciones y lo hacen más atractivo.
- La interacción de los usuarios, la cual permite recorrer el entorno virtual, se realiza mediante el teclado del PC.

Gráfico: Desde la figura 3.1 hasta la figura 3.6 se observa el entorno tridimensional del servicio de medicina preventiva y educación a la población infantil.

Figura 3.1. Entorno tridimensional del servicio de caminata virtual

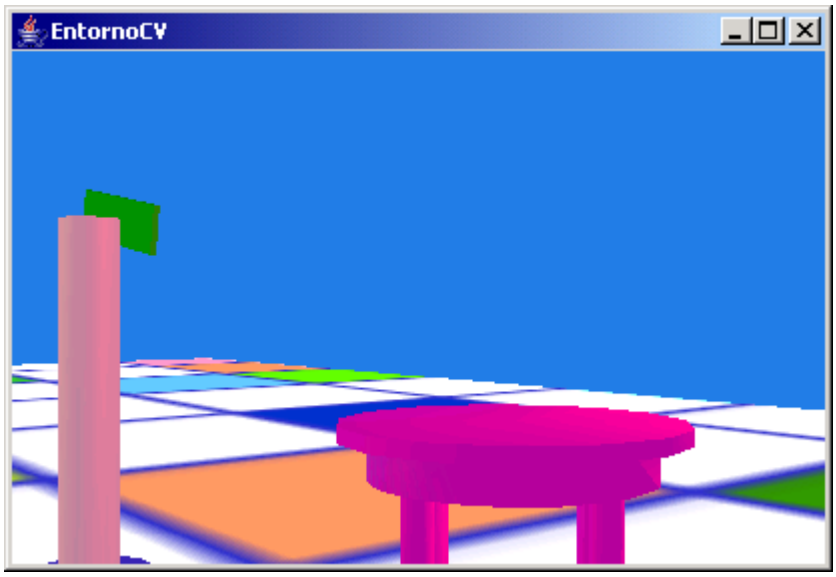

Figura 3.2. Exploración de objetos visuales

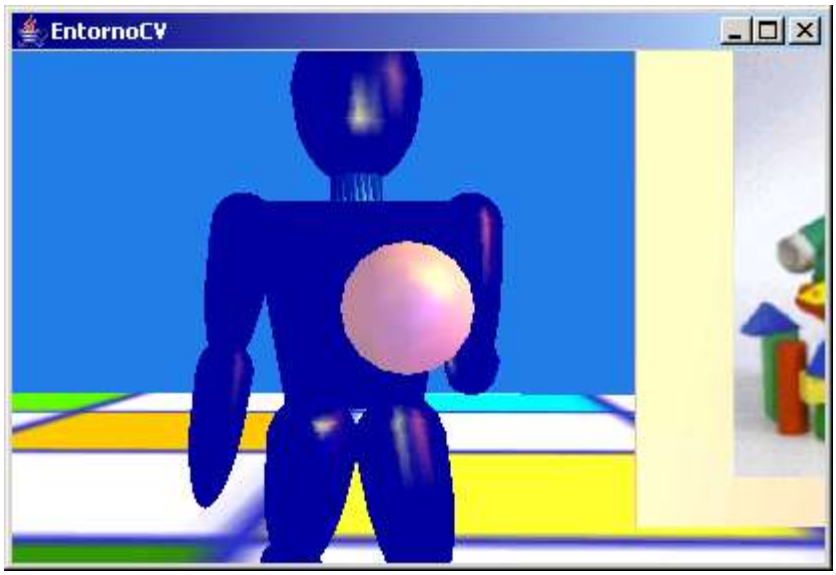

Figura 3.3. Percepción de luces en el entorno virtual

Figura 3.4. Animación de objetos visuales – vista 1

Figura 3.5. Animación de objetos visuales – vista 2

Figura 3.6. Presentación audio-visual sobre salud oral

b) Diagrama de casos de uso:

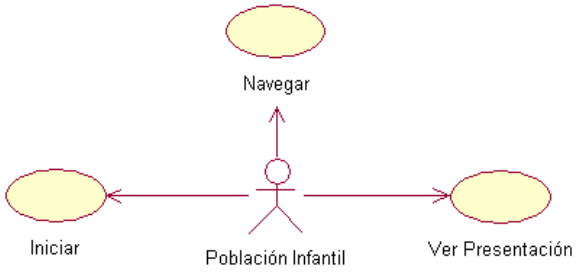

Figura 3.7. Diagrama de casos de uso

c) Descripción de casos de uso:

Caso de uso No. 1: Iniciar.

Objetivo: Iniciar la ejecución del servicio.

Descripción funcional:

- El usuario inicia la ejecución del servicio mediante la solicitud del mismo desde un navegador web.
- El servicio incluye en el applet los objetos visuales y configura sus propiedades y comportamiento.
- El servicio muestra el applet, como en la figura 3.1.

Descripción extra-funcional:

- El throughput de la red debe ser aceptable.
- El PC de los usuarios debe tener buenas capacidades de procesamiento.

Caso de uso No. 2: Navegar.

Objetivo: Interactuar con el entorno tridimensional mediante el desplazamiento de los usuarios a través del entorno utilizando el teclado del PC.

Descripción funcional:

- Para obtener un desplazamiento hacia adelante, el usuario presiona la tecla flecha hacia arriba.

- Para obtener un desplazamiento hacia atrás, el usuario presiona la tecla flecha hacia abajo.
- Para obtener una rotación hacia la derecha, el usuario presiona la tecla flecha hacia la derecha.
- Para obtener una rotación hacia la izquierda, el usuario presiona la tecla flecha hacia la izquierda.
- La velocidad de desplazamiento y rotación pueden incrementarse al presionar la techa shift y luego las techas anteriormente descritas.

Descripción extra-funcional:

Ninguna.

Caso de uso No. 3: Ver Presentación.

Objetivo: Iniciar una presentación audio-visual relacionada con temas de salud y auto-cuidado.

Descripción funcional:

- El entorno virtual contiene dos objetos visuales en forma de pantallas de presentación o tableros.
- Cada tablero posee un sensor de proximidad el cual se activa cuando los usuarios se encuentran dentro de una zona de influencia.
- Al activarse el sensor se inicia la presentación de un tema de salud preventiva y auto-cuidado.
- La presentación continúa si el usuario se mantiene en la zona de influencia y termina si sale de ella.

Descripción extra-funcional:

Es necesario que los tableros posean capacidades para detectar intercepciones o colisiones de tal forma que no puedan ser traspasados por los usuarios.

3.1.2. Calidad de servicio del servicio

Los atributos de calidad de servicio que se consideran para este servicio son los siguientes:

- Disponibilidad: 99%.
- Confianza: alta por ser un servicio de educación.
- Latencia: 10 segundos para el inicio del servicio.
- Prioridad: media.

3.2. PLANO DE COMPONENTES

Para dar soporte al servicio se requiere de un componente web denominado Entorno3D.

3.2.1. Arquitectura del servicio

La arquitectura del servicio está conformada por el componente web Entorno3D identificado, y se presenta en la figura 3.8.

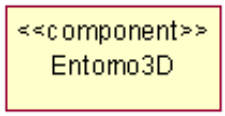

Figura 3.8. Arquitectura del servicio

Entorno3D: Componente web, ofrece los recursos necesarios para iniciar la ejecución del applet.

3.2.2. Interacción de componentes

La funcionalidad del servicio, especificada para cada caso de uso, se muestra en las figuras 3.9, 3.10 y 3.11.

Interacción caso de uso iniciar:

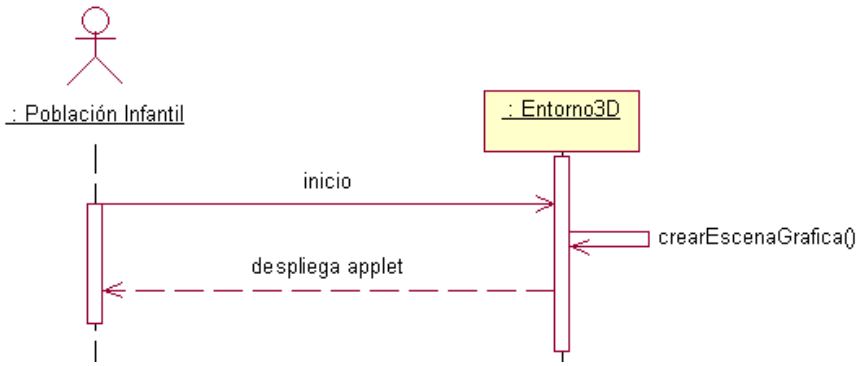

Figura 3.9. Interacción con el componente Entorno3D para el caso de uso iniciar

Descripción: Los usuarios solicitan la ejecución del servicio desde un navegador web, esta petición es recibida por el contenedor web del servidor de aplicaciones y es redireccionada hacia la página principal, esta página, que contiene información del applet, inicia la descarga del mismo. El applet crea la escena gráfica la cual representa el entorno virtual y éste se despliega en el navegador del usuario.

Interacción caso de uso navegar:

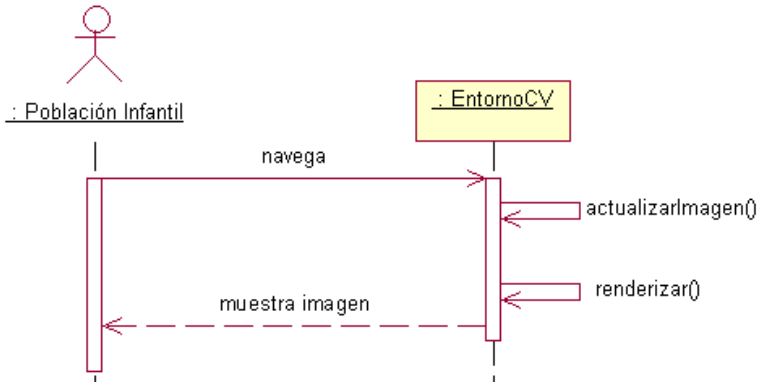

Figura 3.10. Interacción con el componente Entorno3D para el caso de uso Navegar

Descripción: La interacción de los usuarios con el entorno virtual se realiza sobre el applet EntornoCV y se utiliza el teclado del PC. Así, para moverse hacia adelante el usuario presiona la tecla flecha hacia arriba, para moverse hacia atrás el usuario presiona la tecla flecha hacia abajo, para girar hacia la derecha el usuario presiona la tecla flecha hacia la derecha y para girar hacia la izquierda el usuario presiona la tecla flecha hacia la izquierda. Si el usuario desea aumentar la velocidad de navegación, presiona la tecla shift de forma simultánea a las teclas anteriormente descritas. Los cambios generados en la escena tridimensional son gestionados por clases de comportamiento que adicionalmente actualizan y renderizan la pantalla de proyección del entorno virtual.

Interacción caso de uso ver presentación:

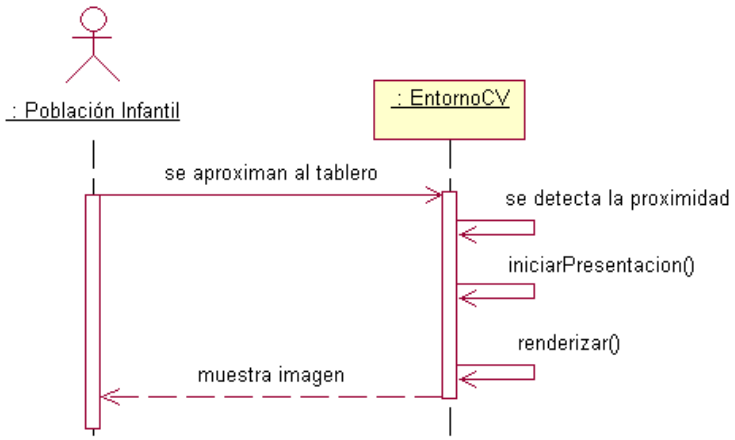

Figura 3.11. Interacción con el componente Entorno3D para el caso de uso Ver Presentación

Descripción: Los usuarios tienen la capacidad de iniciar la presentación audio-visual de educación en salud y auto-cuidado, para ello es necesario desplazarse hasta los tableros de presentación. La proximidad de los usuarios a los tableros es detectada por un sensor el cual inicia la presentación de la información audio-visual. Si los usuarios continúan en la zona de influencia del tablero, la presentación continúa, pero si se alejan, la presentación finaliza.

3.2.3. Especificación de componentes

Componente No. 1: *Entorno3D.*

a) Información general:

Versión: 3.1.
Fecha: 2 de Mayo de 2004.
Descripción: Es un componente de presentación que inicia la descarga y ejecución del applet que contiene el entorno tridimensional con el cual el usuario interactúa.
Tipo: Componente web.
Ámbito: Educación en salud y auto-cuidado.
Interfaces Requeridas: Ninguna.
Interfaces Ofrecidas: Ninguna.
Licencia: Open Source.
Nombre del fabricante: Claudia Milena Hernández, Carlos Hernán Tobar.
Pagina web: www.unicauca.edu.co/~claudiah, www.unicauca.edu.co/~carlost
Contacto: claudiah@unicauca.edu.co, carlost@unicauca.edu.co
Dirección: FIET – Unicauca.

b) Interfaces:

Interfaces ofrecidas: Ninguna.
Interfaces requeridas: Ninguna.

c) Propiedades:

Propiedad No. 1: Portabilidad.

Descripción: El componente esta realizado en Java y por lo tanto puede ser utilizado en cualquier sistema operativo que tenga la máquina virtual de Java.
Valor: Alto.

d) *Requisitos:*

Requisitos de red
Ancho de banda: Mínimo 10 Kbps.
Protocolos de comunicación: HTTP, IP.
Calidad de servicio: Indiferente.

Requisitos Software
Plataforma de soporte: Servidor J2EE.
Librerías: Java3D.
Sistema operativo: Independiente.
Programas: Direct X y Plugin Java3D en el PC del cliente.

3.3. PLANO DE REALIZACIÓN

3.3.1. Clasificación de componentes

Para el servicio los componentes se clasifican como se describe a continuación.

Número de componentes: 1.
Componentes web: Entorno3D.
Componentes empresariales: Ninguno.
Componentes clientes: Ninguno.

3.3.2. Descripción de realización

Para el servicio la información de realización es la siguiente:

Componente No. 1: *Entorno3D.*

Tipo: Componente web.

Páginas jsp:

- index.jsp: Esta página se contacta cuando el usuario solicita el servicio mediante la dirección URL, y tiene como función redireccionar hacia la página principal.
- principal.jsp: Contiene el applet que carga el ambiente tridimensional.

Servlets: Ninguno.

Applets:

- EntornoCV.java: Contiene los objetos visuales del ambiente tridimensional.

JavaBeans: Ninguno.

Intefaces requeridas: Ninguna.

Descriptores de despliegue: web.xml.

Archivo war: entorno3d.war.

3.3.3. Implementación de los componentes

El componente se implementó en el lenguaje de programación Java y con especial utilización del API Java3D, y se realizó el proceso de codificación, compilación y empaquetamiento. Este componente se incluye en el CD de este trabajo y puede ser consultado.

3.3.4. Especificación de despliegue

La información de despliegue del servicio es la siguiente:

Servicio: Medicina preventiva y educación a la población infantil.

Descripción: Este servicio consiste en un entorno virtual (objetos tridimensionales localizados en un espacio tridimensional) del tipo inmersivo, gracias a la posibilidad de navegación (desplazamiento) del usuario y experiencia de interacción con los objetos virtuales que buscan ser representaciones tridimensionales del mundo real, así como de la incorporación de luces y sonidos que dan la sensación de realismo. Este entorno virtual tiene como objetivo primordial captar la atención de los usuarios, que en primera instancia se ha escogido sean niños. Los cuales, en la acción de explorar el entorno virtual descubran presentaciones audio-visuales que desplieguen contenidos de aprendizaje relacionados con temas de prevención en salud o cuidado personal. En este caso se ha escogido impartir dos presentaciones didácticas acerca de higiene oral y nutrición.

Componentes Utilizados:

- Componentes web: Entorno3D.

Descriptor de despliegue:

- web.xml: Este servicio se despliega como un componente web por lo tanto el descriptor de despliegue esencial es web.xml.

Archivo ear: No se considera.

PCs que alojan los componentes:

El servicio (archivo entorno3d.war) se desplegó en el servidor de aplicaciones residente en el PC con dirección IP 192.168.0.45.

Fecha de instalación: 30 de Junio de 2004.

3.4. PLANO DE SOPORTE

A continuación se describe el framework de componentes utilizado.

Plataforma: Java Open Application Server –JOnAS.

Versión: 4.0.0.

Descripción: JOnAS es una implementación de código abierto de un servidor de aplicaciones conforme a la especificación J2EE, y está desarrollado en Java completamente. Permite desplegar y ejecutar aplicaciones compuestas por Enterprise JavaBeans y componentes web.
Fabricante: ObjectWeb consortium.
Licencia: Open Source.
Modelo de componentes base: Enterprise JavaBeans 2.0.
Servicios de soporte: Comunicación y nombrado, contenedor EJB, contenedor Web, ear, transacción, bases de datos, seguridad, mensajería, gestión, mail y webservices.
Protocolos de comunicación: RMI, RMI sobre IIOP, CMI.
Requisitos de sistema: JDK 1.3 o mayor, Sistema operativo (Linux, AIX, Windows, Solaris, HP-UX, etc.), bases de datos (Oracle, PostgreSQL, MySQL, SQL server, Access, DB2, Versant, Informix, Interbase, etc.).
Características adicionales: Última versión - 4.1.2.del 7 de Julio de 2004.

3.5. RESULTADOS

Este servicio está compuesto únicamente por un componente web y por lo tanto la arquitectura del servicio se define con éste componente. Sin embargo puede verse que el servicio puede ser fácilmente extendido para satisfacer nuevos requisitos de servicio mediante la incorporación de componentes empresariales, tanto de sesión como de entidad.

Es importante notar que el entorno virtual fue realizado completamente en Java utilizando el API Java3D.

Este servicio y su documentación se incluye en el CD.

4. SERVICIO DE AFILIACIÓN DE USUARIOS

Este anexo ilustra la aplicación del modelo de creación de servicios de telemedicina en la concepción, desarrollo, implementación y documentación de un servicio de afiliación de usuarios. Este servicio esta enmarcado dentro de los servicios de gestión de información médica, tiene un enfoque de creación orientada a componentes y se describe de acuerdo a los cuatro planos del modelo de creación de servicios definido en el trabajo de grado.

4.1. PLANO DE SERVICIOS

4.1.1. Caracterización del servicio de telemedicina

a) Información general del servicio de telemedicina:

Servicio: Afiliación de usuarios.
Objetivo: Servir como herramienta de gestión en la afiliación y registro de usuarios.
Actores: Administrador.
Descripción:

- Este servicio puede ser utilizado por el administrador del sistema o la persona encargada de la afiliación y registro de usuarios.
- El servicio permite el registro, actualización o eliminación de los datos de afiliación de un usuario.
- El servicio permite ver la lista de los usuarios afiliados.
- El servicio puede ser utilizado a través de la web.

Gráfico: En la figura 4.1. se observa la interfaz gráfica del servicio, donde se muestra la página de registro de nuevos afiliados.

Figura 4.1. Interfaz gráfica del registro de usuarios

b) Diagrama de casos de uso:

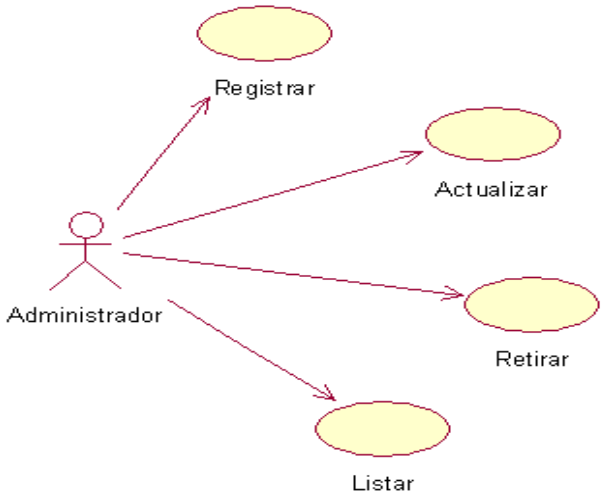

Figura 4.2. Diagrama de casos de uso

c) Descripción de casos de uso:

Caso de uso No. 1: Registrar.

Objetivo: Permitir el ingreso de los datos de un nuevo afiliado.

Descripción funcional:

- El usuario accede a una pagina web de registro, donde se solicitan el nombre, apellido, tipo y número de identificación, dirección, teléfono, e-mail e ips.
- El usuario suministra los datos y solicita registrarlos.
- La aplicación verifica si el número de identificación del usuario ya esta registrado en el servicio, de ser así no se permite un nuevo registro con el mismo número de identificación y se muestra una página de error.
- Si no existe el número de identificación, se realiza el registro de los datos del afiliado y se muestra una página de información.

Descripción extra-funcional:

- Los datos deben guardarse en una base de datos.

Caso de uso No. 2: Actualizar.

Objetivo: Modificar los datos de un afiliado.

Descripción funcional:

- El usuario accede a una pagina web de actualización donde se solicita el número de identificación del afiliado al que se le modificarán sus datos.
- El usuario suministra un número de identificación y solicita buscar los datos del afiliado.
- La aplicación verifica si el número de identificación existe, de ser así muestra una página con los datos del afiliado, donde es posible modificarlos todos a excepción del número y el tipo de identificación.
- Si no existe el número de identificación, se muestra una página donde se informa que no se encontró el número de identificación.
- El usuario modifica los datos de un afiliado.
- La aplicación guarda los nuevos datos del afiliado.

Descripción extra-funcional:

- Los datos deben modificarse en la base de datos.

Caso de uso No. 3: Retirar.

Objetivo: Eliminar el registro de un afiliado.

Descripción funcional:

- El usuario accede a una pagina web de retiro de afiliados donde se solicita el número de identificación del afiliado que será retirado.
- El usuario suministra un número de identificación y solicita retirar al afiliado.
- La aplicación verifica si el número de identificación existe, de ser así retira el afiliado y muestra una página donde informa que el registro del afiliado fue retirado.
- Si no existe el número de identificación, se muestra una página donde se informa que no se encontró el número de identificación.

Descripción extra-funcional:

- Los datos deben modificarse en la base de datos

Caso de uso No. 4: Listar.

Objetivo: Ver la lista de los afiliados registrados.

Descripción funcional:

- El usuario accede a la página web del servicio de afiliación donde tiene la opción de ver la lista de afiliados.
- El usuario selecciona ver la lista de afiliados.
- La aplicación busca los afiliados registrados y muestra una página con la lista de afiliados, donde se incluye el nombre, apellido y número de identificación de cada afiliado.

Descripción extra-funcional:

- La aplicación debe cargar la lista de los afiliados en forma dinámica desde la base de datos.

4.1.2. Calidad de servicio del servicio

Los atributos de calidad de servicio que se consideran para este servicio son los siguientes:

- Disponibilidad: 99%.
- Latencia: 10 segundos.
- Consistencia: se debe garantizar la integridad de la información.
- Prioridad: media.
- Seguridad: los datos solo se pueden modificar por usuarios autorizados.

4.2. PLANO DE COMPONENTES

Los componentes identificados para dar soporte al servicio son:

- Registro: Componente de presentación que contiene las páginas web del servicio.
- Afiliación: Componente que permite gestionar la información de los afiliados.
- BDAfiliación: Componente de persistencia de los datos de los afiliados.

4.2.1. Arquitectura del servicio

La arquitectura del servicio conformada a partir de los componentes identificados es la siguiente:

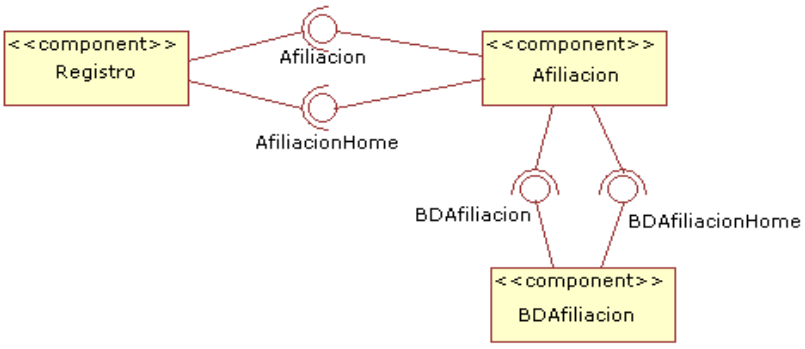

Figura 4.3. Arquitectura del servicio

Registro: Componente web, ofrece las páginas web necesarias para el ingreso, actualización y retiro de un afiliado. También ofrece las páginas de error, información y listado de afiliados.
Afiliación: Componente de sesión, realiza la lógica correspondiente a la gestión de la información de los afiliados.
BDAfiliación: Gestiona la persistencia de los datos correspondientes a los afiliados.

4.2.2. Interacción de componentes

La funcionalidad del servicio, especificada para cada caso de uso, se determina mediante la interacción de sus componentes como se muestra en las figuras 4.4, 4.5, 4.6 y 4.7.

Interacción Caso de Uso Registrar:

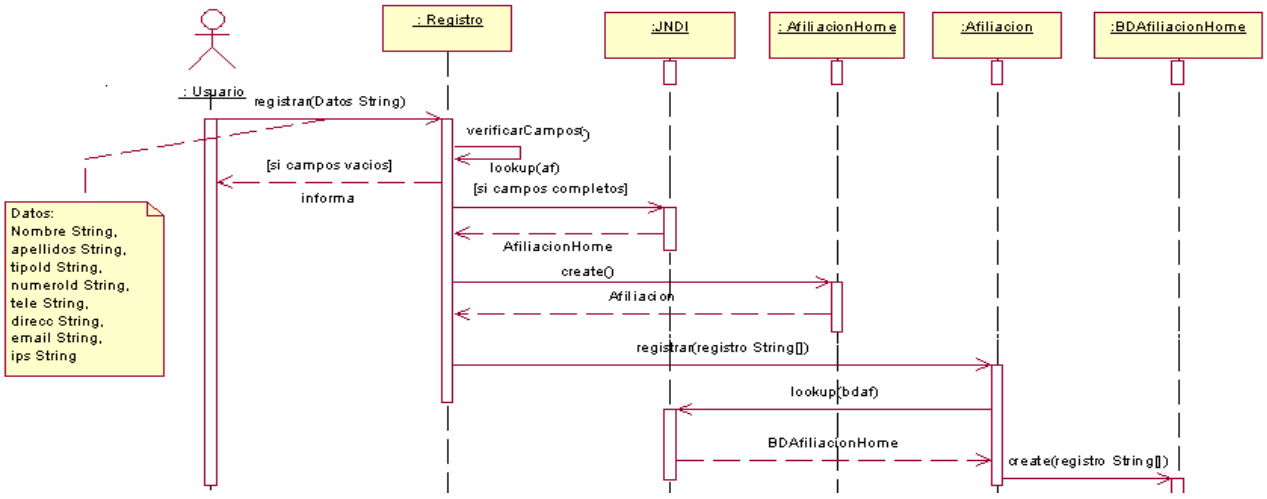

Figura 4.4. Interacción de componentes para el caso de uso registrar

Descripción: Al usuario se le presenta una página de registro donde se solicitan los datos del afiliado: nombre, apellidos, tipo de identificación, número de identificación, teléfono, dirección, email e ips. Cuando el usuario solicita el registro de los datos, y suministra adecuadamente los datos se obtiene la referencia a la interfaz AfiliacionHome a través de JNDI. Utilizando la

operación create de AfiliacionHome se obtiene la interfaz Afiliación, a la cual se le solicita registrar un afiliado mediante la operación registrar y se le entregan todos los datos en un arreglo. En esta operación se obtiene la interfaz Home del componente BDAfiliacion. Luego se utiliza la operación create para crear un nuevo registro de afiliado con los datos suministrados, si se puede realizar el registro se retorna el String "succcess" y de lo contrario "failure". Finalmente el componente registro le informa al usuario que el afiliado ha sido registrado mediante una página de información, o de lo contrario se despliega un mensaje de error.

Interacción Caso de uso Actualizar:

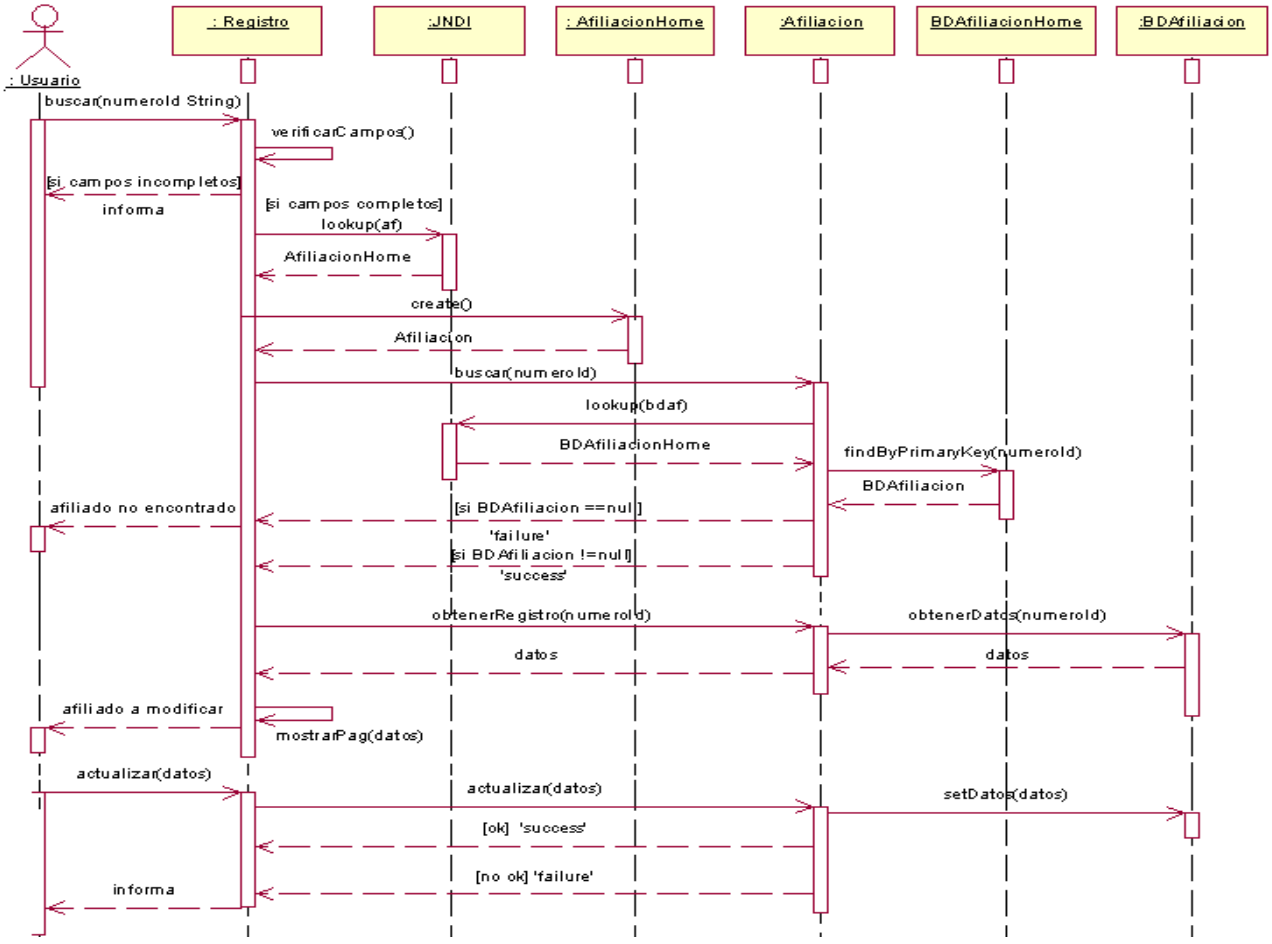

Figura 4.5. Interacción de componentes para el caso de uso actualizar

Interacción Caso de uso Retirar:

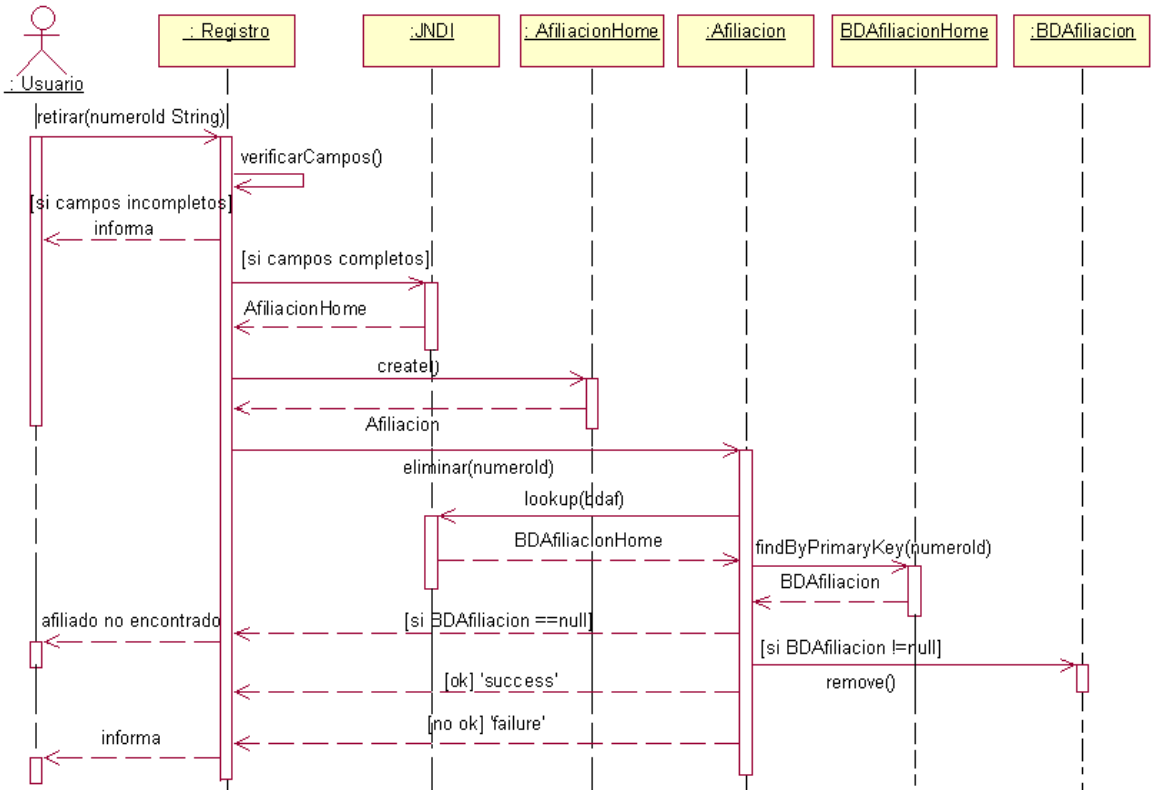

Figura 4.6. Interacción de componentes para el caso de uso retirar

Descripción: Al usuario se le presenta una página de retiro donde se solicita el número de identificación del afiliado a retirar. Cuando el usuario ingreso el número de identificación del afiliado a retirar y suministra adecuadamente los datos se obtiene la referencia a la interfaz AfiliacionHome a través de JNDI. Utilizando la operación create de AfiliacionHome se obtiene la interfaz Afiliación, de la cual se utiliza la operación eliminar que tiene como parámetro el número de identificación del afiliado, en esta operación se obtiene la interfaz Home del componente BDAfiliacion. Luego se utiliza la operación findByPrimaryKey para buscar si existe un afiliado con el número de identificación indicado, si no existe retorna "failure". Si el afiliado no se encuentra se le informa al usuario y si existe se retira al afiliado y se retorna "success"

Interacción Caso de uso Listar:

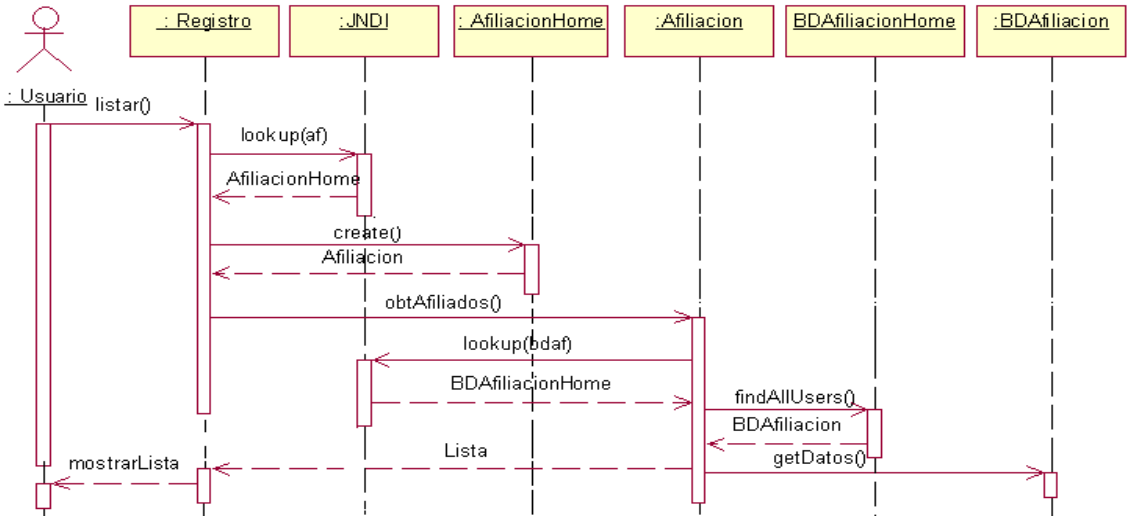

Figura 4.7. Interacción de componentes para el caso de uso listar

Descripción: Al usuario se le presenta una página donde tiene la opción de ver la lista de afiliados, cuando el usuario solicita ver la lista se obtiene la referencia a la interfaz AfiliacionHome a través de JNDI. Utilizando la operación create de AfiliacionHome se obtiene la interfaz Afiliación, de la cual se utiliza la operación obtAfiliados, en la cual se obtiene la interfaz Home del componente BDAfiliacion. Luego se utiliza la operación findAllUsers para recuperar todos los afiliados. Después se obtienen los datos de los afiliados y se forma una lista que es devuelta al componente registro para que sea presentada al usuario.

4.2.3. Especificación de componentes

Componente No. 1: *Registro.*

a) Información general:

Versión: 1.2.

Fecha: 2 de Julio de 2004.

Descripción: Es un componente de presentación que ofrece las páginas web necesarias para la interacción de los usuarios con el servicio.

Tipo: Componente web.

Ámbito: Gestión de información de afiliados.

Interfaces Requeridas:

- AfiliaciónHome.
- Afiliación.

Interfaces Ofrecidas: Ninguna.

Licencia: Open Source.

Nombre del fabricante: Claudia Milena Hernández, Carlos Hernán Tobar.

Pagina web: www.unicauca.edu.co/~claudiah, www.unicauca.edu.co/~carlost

Contacto: claudiah@unicauca.edu.co, carlost@unicauca.edu.co

Dirección: FIET – Unicauca.

b) Interfaces:

Interfaces ofrecidas: Ninguna.

Interfaces requeridas: estas se muestran en la figura 4.8.

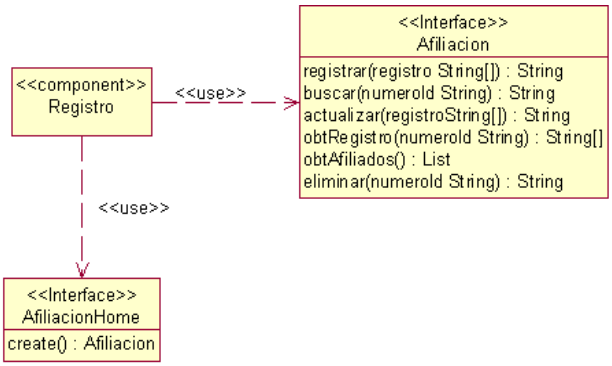

Figura 4.8. Interfaces requeridas por el componente Registro

c) Propiedades:

Propiedad No. 1: Portabilidad.
Descripción: El componente esta realizado en Java y por lo tanto puede ser utilizado en cualquier sistema operativo que tenga la máquina virtual de Java.
Valor: Alto.

Propiedad No. 2: Seguridad.
Descripción:

- El componente oculta los datos de los afiliados y envía la información de forma cifrada.

Valor: Alto.

d) Requisitos:

Requisitos de red
Ancho de banda: Mínimo 10 Kbps.
Protocolos de comunicación: HTTP, IP.
Calidad de servicio: Soporte para transmisión segura.

Requisitos Software
Plataforma de soporte: Servidor J2EE.
Librerías: ninguna.
Sistema operativo: Independiente.
Programas: ninguno.

Componente No. 2: *Afiliación.*

a) Información general:

Versión: 1.7.

Fecha: 2 de Julio de 2004.

Descripción: Es un componente que realiza la lógica correspondiente la gestión de la información de los afiliados.

Tipo: Componente de sesión.

Ámbito: Gestión de afiliados.

Interfaces Requeridas:

- BDAfiliacionHome.
- BDAfiliacion.

Interfaces Ofrecidas:

- AfiliaciónHome.
- Afiliación.

Licencia: Open Source.

Nombre del fabricante: Claudia Milena Hernández, Carlos Hernán Tobar.

Pagina web: www.unicauca.edu.co/~claudiah, www.unicauca.edu.co/~carlost

Contacto: claudiah@unicauca.edu.co, carlost@unicauca.edu.co

Dirección: FIET – Unicauca.

b) Interfaces:

Las interfaces que ofrece el componente son:

Interfaz No. 1: AfiliaciónHome.

Operaciones:

- Afiliación create(): Esta operación crea la interfaz Afiliación.

Interfaz No. 2: Afiliación.

Operaciones:

- String registrar(String[] registro): Esta operación almacena los datos de un afiliado en una base de datos. Si se realiza la operación satisfactoriamente el valor de retorno es "success", de lo contrario se retorna "failure".
- String buscar(String numeroId): Esta operación permite determinar si una persona se encuentra o no afiliada en el registro. Si la persona a buscar se encuentra registrada se retorna "success", de lo contrario se retorna "failure". La búsqueda se realiza por el número de identificación.
- String actualizar(String[] registro): Esta operación permite modificar los datos de registro de un afiliado. Se permite cambiar toda la información de registro excepto el tipo y el número de identificación.
- String[] obtRegistro(String numeroId): Esta operación permite obtener todos los datos de registro de un afiliado, para ello es necesario suministrar el número de identificación del mismo.
- List obtAfiliados(): Esta operación permite obtener una lista con todos los afiliados en el servicio, esta lista puede ser utilizada para ser desplegada en una página en forma de tabla.
- String eliminar(String numeroId): Esta operación retira el registro de un afiliado determinado, para esto es necesario suministrar el número de identificación del afiliado. Si esta operación es exitosa se retorna "success" o de lo contrario se retorna "failure".

Las interfaces ofrecidas y requeridas se muestran en la figura 4.9.

c) Propiedades:

Propiedad No. 1: Portabilidad.
Descripción: El componente esta realizado en Java y por lo tanto puede ser utilizado en cualquier sistema operativo que tenga la máquina virtual de Java.
Valor: Alto.

Propiedad No. 2: Interoperabilidad.

Descripción: El componente interopera con componentes Corba.

Valor: Alto.

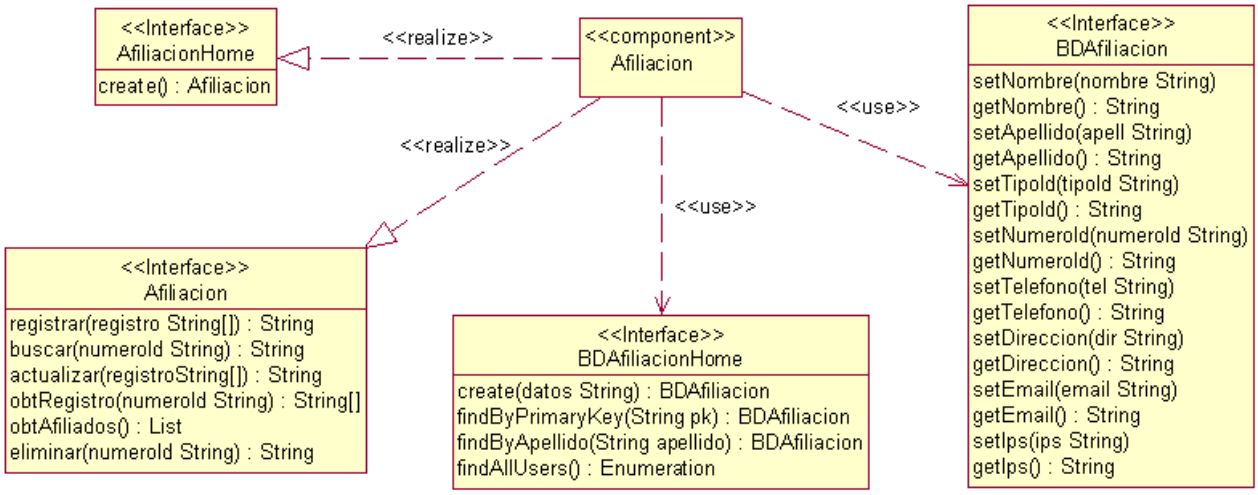

Figura 4.9. Interfaces del componente Afiliación

d) Requisitos:

Requisitos de red

Ancho de banda: Indiferente.

Protocolos de comunicación: RMI, RMI/IIOP.

Calidad de servicio: Indiferente.

Requisitos Software

Plataforma de soporte: Servidor J2EE.

Librerías: Ninguna.

Sistema operativo: Independiente.

Programas: Ninguno.

Componente No. 3: *BDAfiliacion.*

a) Información general:

Versión: 2.1.

Fecha: 2 de Julio de 2004.

Descripción: Es un componente de persistencia de los datos de los usuarios registrados (afiliados).

Tipo: Componente de entidad.

Ámbito: Almacenamiento, recuperación, actualización y eliminación de información de afiliados.

Interfaces Requeridas: Ninguna.

Interfaces Ofrecidas:

- BDAfiliacionHome.
- BDAfiliacion.

Licencia: Open Source.

Nombre del fabricante: Claudia Milena Hernández, Carlos Hernán Tobar.

Pagina web: www.unicauca.edu.co/~claudiah, www.unicauca.edu.co/~carlost

Contacto: claudiah@unicauca.edu.co, carlost@unicauca.edu.co

Dirección: FIET – Unicauca.

b) Interfaces:

Las interfaces que ofrece el componente son:

Interfaz No. 1: BDAfiliacionHome.

Operaciones:

- BDAfiliacion create(String nombre, String apellido, String tipoId, String numeroId, String telefono, String direccion, String email, String ips): Esta operación crea la interfaz

BDAfiliacion, y almacena los datos suministrados en una tabla llamada usuarios en una base de datos relacional.

- BDAfiliacion findByPrimaryKey(String pk): Retorna la interfaz BDAfiliacion de acuerdo a la clave primaria, en este caso el número de identificación del afiliado.
- BDAfiliacion findByApellido(String apellido): Retorna la interfaz BDAfiliacion de acuerdo al apellido suministrado.
- Enumeration findAllUsers(): Retorna todos los usuarios registrados.

Interfaz No. 2: BDAfiliacion.

Operaciones:

- void setNombre(String nombre): Almacena el atributo suministrado en la columna nombre de la tabla usuarios.
- String getNombre(): Recupera el valor de la columna nombre de un afiliado.
- void setApellido(String apellido): Almacena el atributo suministrado en la columna apellido de la tabla usuarios.
- String getApellido(): Recupera el valor de la columna apellido de un afiliado.
- void setTipoId(String tipoId): Almacena el atributo suministrado en la columna tipoid de la tabla usuarios.
- String getTipoId(): Recupera el valor de la columna tipoid de un afiliado.
- void setNumeroId(String numeroId): Almacena el atributo suministrado en la columna numeroid de la tabla usuarios.
- String getNumeroId(): Recupera el valor de la columna numeroid de un afiliado.
- void setTelefono(String telefono): Almacena el atributo suministrado en la columna telefono de la tabla usuarios.
- String getTelefono(): Recupera el valor de la columna telefono de un afiliado.
- void setDireccion(String direccion): Almacena el atributo suministrado en la columna direccion de la tabla usuarios.
- String getDireccion(): Recupera el valor de la columna direccion de un afiliado.

- void setEmail(String email): Almacena el atributo suministrado en la columna email de la tabla usuarios.
- String getEmail(): Recupera el valor de la columna email de un afiliado.
- void setIps(String ips): Almacena el atributo suministrado en la columna ips de la tabla usuarios.
- String getIps(): Recupera el valor de la columna ips de un afiliado.

Las interfaces ofrecidas se muestran en la figura 4.10.

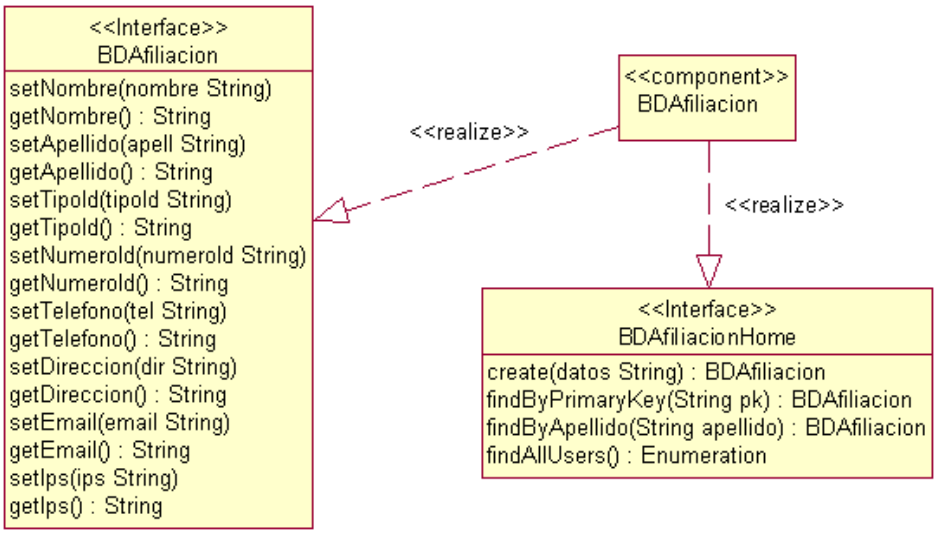

Figura 4.10. Interfaces del componente BDAfiliacion

c) Propiedades:

Propiedad No. 1: Portabilidad.

Descripción: El componente esta realizado en Java y por lo tanto puede ser utilizado en cualquier sistema operativo que tenga la máquina virtual de Java.

Valor: Alto.

Propiedad No. 2: Interoperabilidad.

Descripción: El componente interopera con componentes Corba.

Valor: Alto.

Propiedad No. 3: Persistencia.

Descripción: El componente garantiza la persistencia de la información de los afiliados.

Valor: Alto.

Propiedad No. 4: Transaccionalidad.

Descripción: El componente garantiza la integridad de la información en la realización de las operaciones.

Valor: Alto.

d) Requisitos:

Requisitos de red

Ancho de banda: Indiferente.

Protocolos de comunicación: RMI, RMI/IIOP.

Calidad de servicio: Indiferente.

Requisitos Software

Plataforma de soporte: Servidor J2EE.

Librerías: Driver JDBC.

Sistema operativo: Independiente.

Programas: MySQL.

4.3. PLANO DE REALIZACIÓN

4.3.1. Clasificación de componentes

Para el servicio los componentes se clasifican como se describe a continuación.

Número de componentes: 3.

Componentes web: Registro.

Componentes empresariales:

- Componentes de sesión: Afiliación.
- Componentes de entidad: BDAfiliación.
- Componentes manejados por mensajes: Ninguno.

Componentes clientes: Ninguno.

4.3.2. Descripción de realización

Para los componentes del servicio la información de realización es la siguiente.

Componente No. 1: *Registro.*

Tipo: Componente web.

Páginas jsp:

- index.jsp: Esta página se contacta cuando el usuario solicita el servicio mediante la dirección URL, y tiene como función redireccionar hacia la página de registro.
- registro.jsp: Esta página permite registrar los datos correspondientes a un afiliado, contiene los campos del nombre, apellido, tipo de identificación, número de identificación, teléfono, dirección, e-mail e ips.

- busqueda.jsp: Es la página de búsqueda del servicio y permite ingresar el número de identificación de un afiliado.
- actualizacion.jsp: Esta página permite ingresar los datos correspondientes a la actualización de un afiliado.
- eliminación.jsp: Esta página permite ingresar el número de identificación del afiliado que desea retirarse.
- afiliados.jsp: Esta página despliega la información de los afiliados registrados en forma de una tabla.
- información.jsp: Esta página muestra información de las operaciones realizadas.

Servlets: Ninguno.

Applets: Ninguno.

JavaBeans:

- AfiliadoBean.java: Gestiona la información de las páginas jsp e inicia las interacciones entre los componentes con el fin de satisfacer la funcionalidad definida en los casos de uso.

Intefaces requeridas:

- AfiliacionHome.
- Afiliacion.

Descriptores de despliegue:

- web.xml: Define las referencias de los componentes que interactúan con este componente.
- jonas-web.xml: No se emplea para este componente en particular.
- faces-config.xml: Define las reglas de navegaciones para las páginas del componente, y declara el JavaBean utilizado.

Archivo war: registro.war.

Componente No. 2: *Afiliación.*

Tipo: Componente EJB de sesión.

Interfaces ofrecidas:

- AfiliacionHome.java.

- Afiliacion.java.

Implementación:

- AfiliacionBean.java.

Interfaces requeridas:

- BDAfiliacionHome.java.
- Afiliacion.java.

Descriptores de despliegue:

- afiliacion-ejb-jar.xml: Define el nombre del componente EJB, la interfaz Home, las interfaces local o remota, y las referencias a los componentes asociados.
- afiliacion-jonas-ejb-jar.xml: Define el nombre JNDI del componente a través del cual será contactado.

Archivo jar: afiliación.jar.

Componente No. 3: *BDAfiliación*

Tipo: Componente EJB de entidad.

Interfaces ofrecidas:

- BDAfiliacionHome.java.
- BDAfiliacion.java.

Implementación:

- BDAfiliacionBean.java.

Descriptores de despliegue:

- bdafiliacion-ejb-jar.xml: Define el nombre del componente EJB, la interfaz Home, las interfaces local o remota, el nombre de la tabla en la base de datos y el nombre y tipo de las columnas de la tabla.
- bdafiliacion-jonas-ejb-jar.xml: Define el nombre JNDI del componente a través del cual será contactado.

Archivo jar: bdafiliacion.jar.

4.3.3. Implementación de los componentes

Los componentes se implementaron en el lenguaje de programación Java, y se realizó el proceso de codificación, compilación y empaquetamiento. Estos componentes se incluyen en el CD de este trabajo y pueden ser consultados.

4.3.4. Especificación de despliegue

La información de despliegue del servicio es la siguiente:

Servicio: Servicio de afiliación de usuarios.
Descripción: Este servicio permite la gestión de los datos correspondientes a los afiliados de una IPS. La gestión corresponde a las labores de registro, actualización, retiro y visualización de afiliados.
Componentes Utilizados:

- Componentes web: Registro.
- Componentes de sesión: Afiliación.
- Componentes de entidad: BDAfiliación

Descriptor de despliegue:

- application.xml: relaciona los componentes que conforman el servicio en forma de módulos, además para el componente web define el contexto raiz.

Archivo ear: ingreso.ear.
PCs que alojan los componentes:
El servicio (archivo .ear) se desplegaron en el servidor de aplicaciones residente en el PC con dirección IP 192.168.0.45.
Fecha de instalación: 20 de Julio de 2004.

4.4. PLANO DE SOPORTE

A continuación se describe el framework de componentes utilizado.

Plataforma: Java Open Application Server –JOnAS.
Versión: 4.0.0.
Descripción: JOnAS es una implementación de código abierto de un servidor de aplicaciones conforme a la especificación J2EE, y está desarrollado en Java completamente. Permite desplegar y ejecutar aplicaciones compuestas por Enterprise JavaBeans y componentes web.
Fabricante: ObjectWeb consortium.
Licencia: Open Source.
Modelo de componentes base: Enterprise JavaBeans 2.0.
Servicios de soporte: Comunicación y nombrado, contenedor EJB, contenedor Web, ear, transacción, bases de datos, seguridad, mensajería, gestión, mail y webservices.
Protocolos de comunicación: RMI, RMI sobre IIOP, CMI.
Requisitos de sistema: JDK 1.3 o mayor, Sistema operativo (Linux, AIX, Windows, Solaris, HP-UX, etc.), bases de datos (Oracle, PostgreSQL, MySQL, SQL server, Access, DB2, Versant, Informix, Interbase, etc.).
Características adicionales: Última versión - 4.1.2.del 7 de Julio de 2004.

4.5. RESULTADOS

Los resultados del desarrollo de este prototipo fueron satisfactorios ya que se cumplió con los requisitos de servicio y se verificó una vez más la validez del modelo en la creación de servicios de telemedicina. Este servicio y su documentación también están incluidos en el CD.

5. MANUAL DE USUARIO

Para validar el modelo de creación de servicios de telemedicina se desarrollaron los servicios de: visualización de imágenes médicas tridimensionales, afiliación y registro de usuarios, comunicación en tiempo real y, medicina preventiva y educación a la población infantil. A continuación se presentan los requisitos para el correcto funcionamiento de los servicios, su instalación y manual de operación.

5.1. REQUISITOS

Los requisitos software necesarios para la aplicación son los siguientes:

- Java Development Kit JDK 1.4 ó mayor.
- Java Server Faces 1.0.
- Driver JDBC 2.0.
- Servidor de aplicaciones Java Open Application Server JOnAS versión 4.0 o mayor.
- Servidor web Tomcat 5.0 ó mayor.
- Java 3D 1.3 ó mayor.
- DirectX 8.0 ó mayor.
- MySQL 4.0 ó mayor.

Los instaladores de estos programas pueden ser encontrados en el CD adjunto.

5.2. INSTALACIÓN DE LOS SERVICIOS

Para instalar los servicios es necesario realizar las actividades mencionadas a continuación:

- Configuración de la base de datos.
- Configuración del servidor JOnAS.
- Inicio de JOnAS.
- Despliegue del servicio.

Configuración de la base de datos:

En MySQL se debe crear una base de datos llamada tesis.

***Configuración de JOnAS*:**

Siga los pasos indicados a continuación para garantizar el correcto funcionamiento del servidor de aplicaciones.

1- Una vez instalado el servidor es necesario configurar las variables de entorno.
JONAS_ROOT = directorio de instalación de JONAS
PATH = $JONAS_ROOT/bin/unix en Unix ó
PATH = %JONAS_ROOT%\bin\nt en Windows.

2- En el directorio JONAS_ROOT escriba *ant install* para compilar el servidor.

3- Los archivos de configuración de JOnAS se encuentran en el directorio JONAS_ROOT, pero estos pueden ser editados para cambiar la configuración por defecto del servidor. Sin embargo es recomendable que los archivos necesarios por una aplicación especifica sean ubicados en un directorio separado. Para esto se crea un directorio llamado JONAS_BASE y la variable de entorno JONAS_BASE.

Para inicializar el directorio de trabajo JONAS_BASE, escriba *ant create_jonasbase* en el directorio JONAS_ROOT.

4- Para configurar los servicios de JOnAS se utiliza el archivo jonas.properties, localizado en JONAS_BASE/conf/jonas.properties. Para los servicios presentados se debe configurar el servicio de base de datos, así:

En el archivo jonas.properties, en el servicio de bases de datos adicione MySQL:

jonas.service.dbm.datasources MySQL

Cree el archivo MySQL.properties en JONAS_BASE/conf/ con la siguiente información:

datasource.name	*jdbc_1*
datasource.url	*jdbc:mysql://localhost/tesis*
datasource.classname	*(clase Driver JDBC)*
datasource.username	*(NombreUsuario para acceder a la base de datos)*
datasource.password	*(Password para acceder a la base de datos)*
datasource.mapper	*rdb*

Inicio de JOnAS:

En una ventana de comandos escriba *jonas start.*

Despliegue de los servicios:

Copie los archivos visor3d.ear y registro.ear localizados en la carpeta *servicios* del CD al directorio *JONAS_BASE/apps* del servidor de aplicaciones. También copie el archivo *comunicación.war* localizado en la carpeta *servicios* del CD al directorio *JONAS_BASE/webapps.* Posteriormente desde una ventana de comandos ejecute *jonas admin. –a visor3d.ear, jonas admin. –a registro.ear, y jonas admin. –a comunicación.war*.

5.3. MANUAL DE USUARIO

En esta sección se presenta el manual de usuario del servicio de visualización de imágenes médicas tridimensionales. Este servicio se accede a través de un navegador en la dirección: http://localhost:9000/visor3d, allí aparece la página de ingreso al servicio, figura 5.1. Para ingresar al servicio se debe suministrar un nombre de usuario, una contraseña, elegir un perfil ya sea médico o estudiante y presionar el botón ingresar.

Figura 5.1. Página de Ingreso al servicio

Los usuarios registrados en el servicio se muestran en la figura 5.2.

Nombre de Usuario	Contraseña	Perfil
Pedro	12_llave	Estudiante
Eduardo	85city	Médico
Marcela	Kl85	Estudiante

Figura 5.2. Usuarios registrados en el servicio

Cuando un usuario solicita ingresar al servicio, se verifican los datos suministrados y si no son válidos se niega el acceso al servicio como se muestra en la figura 5.3., si los datos son válidos se muestra la página principal del servicio mostrada en la figura 5.4.

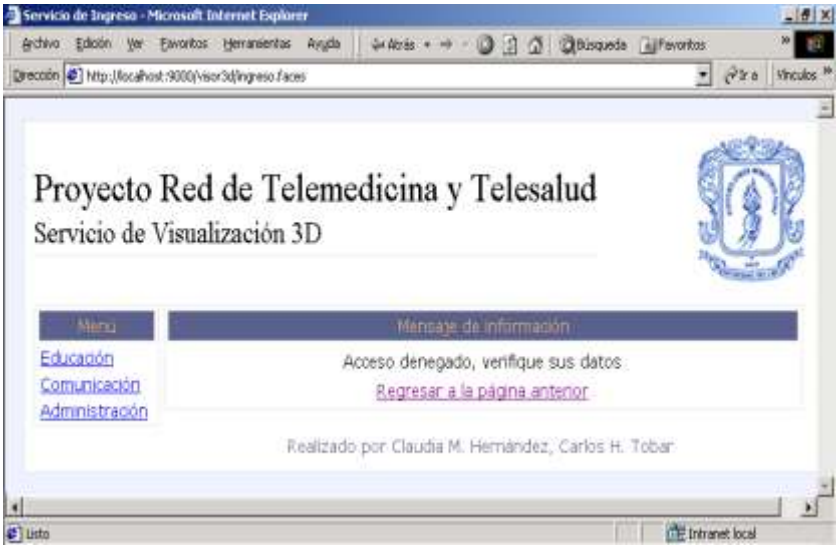

Figura 5.3. Página de acceso denegado del servicio

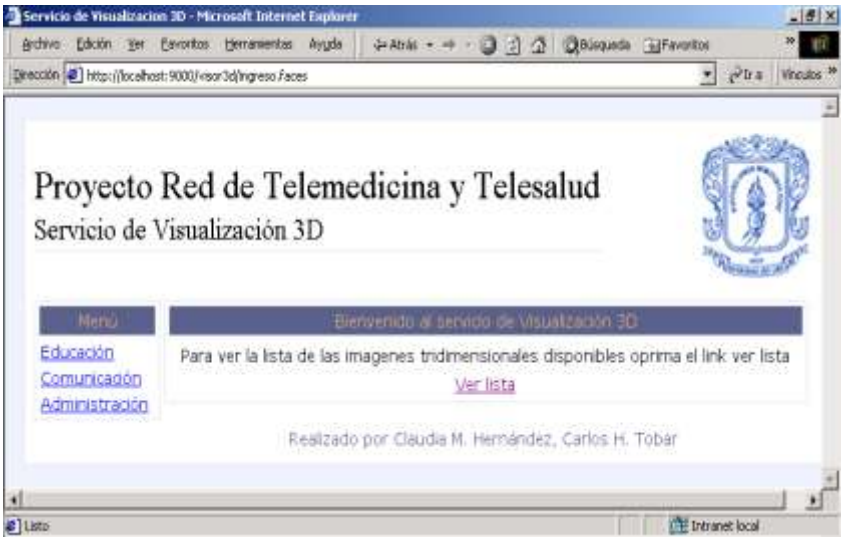

Figura 5.4. Página principal del servicio

En la página principal del servicio se ofrece la opción de ver la lista de las imágenes médicas tridimensionales disponibles, para ver la lista presione el botón ver lista, se mostrará la página de la figura 5.5.

Figura 5.5. Página de la lista de las imágenes 3D disponibles del servicio

En esta página se pueden observar las imágenes 3D disponibles y una descripción de ellas, para visualizar una presione el botón ver imagen 3D. Suponiendo que selecciona la segunda imagen, se mostrará la página de la figura 5.6.

La imagen tridimensional puede ser rotada, trasladada, acercada o alejada mediante la manipulación del mouse:

- Clic derecho y arrastre para rotar la imagen.
- Clic izquierdo y arrastre para trasladar la imagen.
- Clic central y arrastre para acercar la imagen.

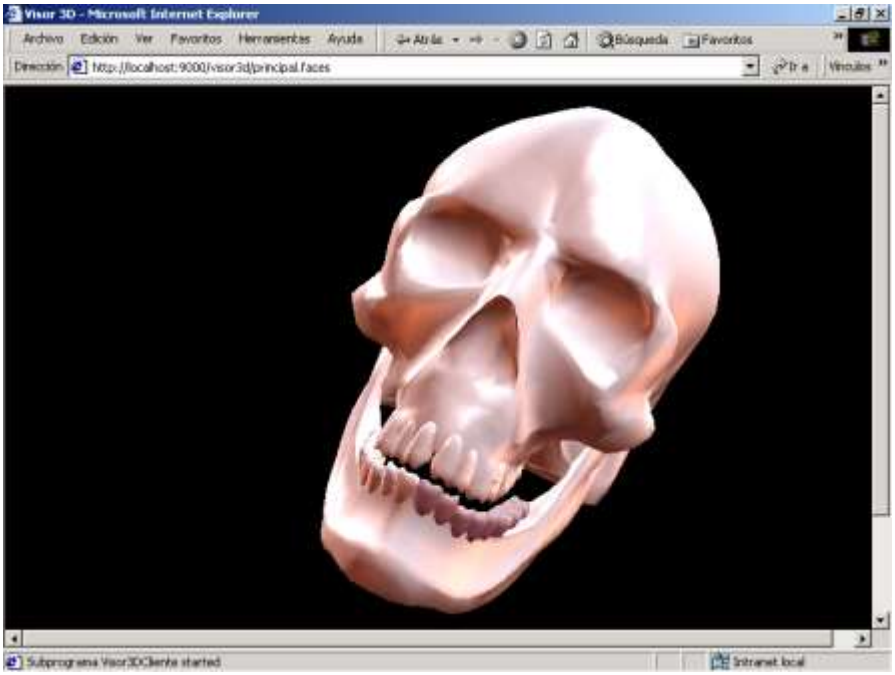

Figura 5.6. Página de visualización y manipulación de una imagen 3D

Estos movimientos pueden realizarse cuantas veces se quiera, a escogencia del usuario del servicio. Si se quiere cargar otra imagen 3D regrese a la página anterior mediante el botón regresar del navegador y escoja otra imagen tridimensional de la lista.

Redespliegue del servicio

Cuando se quiere sacar el servicio de operación se dice que sé redespliega en el servidor de aplicaciones. Para redesplegar el servicio, en una ventana de comandos escriba:
jonas admin. –r visor3d.ear

De forma similar se utilizan los servicios de afiliación y registro de usuarios, de comunicación, y de medicina preventiva y educación a la población infantil mediante su solicitud en el menú de la izquierda de las páginas web.