Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

Título: Desarrollo de funcionalidades para el uso del módulo Admisión del alas HIS por usuarios discapacitados visuales.

Autores: Miguel Alejandro Nicao Cepeda

Yeni Roselló Legón

Tutor: Ing. Suleydis Suárez Serpa

Cotutores: Ing. Liuver Romel Sañudo Ortiz

Ing. Yanela Martínez Rodríguez

La Habana, julio 2013

"Año 55 de la Revolución

DATOS DE CONTACTO

Tutores:

Ing. Suleydis Suárez Serpa

Instructor graduado en el año 2007 de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Ha impartido las asignaturas Matemática Discreta, Álgebra lineal y Metodología de la Investigación Científica. Se encuentra vinculado al desarrollo del Sistema de Información Hospitalaria alas HIS, que se desarrolla en el Departamento de Gestión Hospitalaria del Centro Especializado en Soluciones de Informática Médica (CESIM). Ha tutorado varios trabajos de diploma hasta el momento.

Correo Electrónico: ssuarez@uci.cu

Ing. Liuver Romel Sañudo Ortiz

Graduado en la Universidad de Ciencias Informáticas (UCI) como Ingeniero en Ciencias Informáticas en el año 2011. Especialista del departamento Sistemas de gestión Hospitalaria del Centro de Informática Médica (CESIM). Ha tutorado 2 trabajos de diploma hasta el momento que han obtenido la calificación máxima de 5 puntos.

Correo Electrónico: lrsanudo@uci.cu

Ing. Yanela Martínez Rodríguez

Ingeniero en Ciencias Informáticas, graduado en el año 2012 de la Universidad de las Ciencias Informáticas (UCI). Vinculado al departamento de Sistemas de Gestión Hospitalaria (GEHOS) del Centro de Informática Médica (CESIM), donde actualmente desempeña los roles de vigilancia tecnológica y analista; este último en el proyecto Sistema de Información Hospitalaria alas HIS.

Correo Electrónico: ymrdguez@uci.cu

AGRADECIMIENTOS

Queremos agradecer a nuestros tutores, en especial a Suleydis por su entrega en este trabajo, al profesor Alain por su ayuda incondicional, a todos los profesores del proyecto que nos ayudaron y en general a todos los que a lo largo de la carrera contribuyeron a nuestra formación.

De Yeni:

Quiero agradecer a mis padres: a papi por ser mi guía y ejemplo y por depositar toda su confianza en mí , a mami por darme siempre amor, cariño, seguridad y sobre todo por ser la luz que me ilumina .

A mis abuelos: Cucha por ser la razón de mi vida y motor impulsor para mi formación profesional y personal, y por su esfuerzo y dedicación llegando a tener un lugar muy especial en mi corazón, a Mima por su amor y cariño y a mi abuelo Raimundo que siempre me ha estado apoyando desde cualquier lugar que se encuentre.

A mi hermano por quererme y estar siempre conmigo.

A mis primos por estar siempre presente y ser un pedacito de mi vida y en especial a Jorge por brindarme un apoyo incondicional.

A mis tíos: Yayi por su cariño y apoyo incansable, a Deisi por ser muy importante en mi vida y por todo su amor, a Ramón y el Chino por estar siempre junto a mí y a mi tío Ray porque ha sido muy especial y siempre guiarme por el camino correcto y sobre todo por ser un profesor ejemplar y brindarme amor y apoyo incondicional y a toda mi familia en general.

A mis amigos por estar siempre junto a mí y compartir alegrías y tristezas en estos largos años en especial a mi compañero de tesis.

De Miguel

Quiero agradecer principalmente y especialmente a mi madre por darme el cariño que solo una madre ejemplar puede dar, por ayudarme cuando más lo necesito, por ser la inspiración para lograr todas mis metas y por tantas cosas que me hacen sentir orgulloso de ser su hijo.

A mi abuelo Alejo que me cuida y me protege aunque ya no esté físicamente junto a mí y siempre desde chiquito se empeñó en guiarme y aconsejarme ante las distintas situaciones de la vida. A mi abue Celmi por darme amor, cariño y engendrar a esa persona maravillosa que es mi madre.

A todos mis tíos abuelos y tías abuelas, en particular a Mimi y Bienve las cuales perdí recientemente y las extraño todavía ya que fueron parte importante en mi vida.

A mis tíos Iván y Lázaro por quererme y cuidarme de forma magistral, poniendo dedicación en mi al igual que lo hace un padre con su hijo.

A mi pareja actualmente Dania la cual ha sido mi media mitad por más de 3 años y en la cual he encontrado una excelente compañera merecedora de todo mi amor. A mi familión entero de manera más general y a mis amigos por compartir todos los momentos.

DEDICATORIA

De Yeni:

A mis padres por hacer de mi alguien capaz de realizar sus metas y a toda mi familia y amigos por su apoyo, amor y sobre todo por la confianza que depositaron en mí.

De Miguel

Quiero dedicarle este trabajo a mi madre que sin ella no soy nadie, a mis abuelos, a mis tíos en general a esa maravillosa familia que tengo y a eso buenos amigos que me han ayudado.

RESUMEN

Actualmente a nivel mundial las nuevas tecnologías permiten que las personas con problemas visuales puedan integrarse al uso de sistemas informáticos, utilizando para ello programas o adaptaciones específicas que ayudan a manejar sin problemas un ordenador.

El módulo de Admisión del Sistema de Información Hospitalaria alas HIS no cuenta con la posibilidad de ser utilizado por cualquier tipo de usuario, existiendo ciertas barreras de accesibilidad en el mismo. Lo que trae consigo que personas con algún tipo de incapacidad como por ejemplo los discapacitados visuales no logren tener las mismas oportunidades que aquellos que no presentan ninguna limitación, a la hora de integrarse en el mundo laboral. El objetivo de este trabajo es desarrollar funcionalidades que faciliten el uso del módulo Admisión del sistema alas HIS a usuarios discapacitados visuales.

El desarrollo de la solución propuesta está guiado por el Proceso Unificado de Desarrollo. Se basa en tecnologías libres y utiliza Java como lenguaje de programación, como servidor de aplicaciones JBoss AS, Java Script cono lenguaje de programación del lado del cliente, JQuery para gestionar los eventos y Mespeak para la conversión de texto a voz.

Entre los beneficios que aporta la solución se encuentra mejorar las condiciones de trabajo en las instituciones hospitalarias del personal discapacitado visual, facilitándole una mejor interacción con el módulo de Admisión del sistema alas HIS.

Palabras clave:

admisión, alas HIS, discapacidad, discapacitado visual.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

El siglo XXI ha estado caracterizado por un enorme avance de las nuevas tecnologías. La investigación y el desarrollo de avances tecnológicos específicamente diseñados para personas con ceguera o deficiencia visual, han resultado hasta el momento en la creación de toda una serie de productos y dispositivos. Estos le permiten a dichas personas llevar una vida lo más parecida posible a la de aquellos que no presentan ningún tipo de deficiencia visual.

En la era moderna, la información y las comunicaciones componen elementos indispensables para el desarrollo del discapacitado visual, por lo que se amplía cada día la necesidad de perfeccionarlas en aras de lograr un mejor trabajo en sociedad. Contar con las tecnologías adecuadas y accesibles, proporciona las herramientas necesarias para que este tipo de persona logre desenvolverse en el entorno laboral convencional, y permite por tanto un elevado nivel de integración en el trabajo.

Las personas con déficit visual se encuentran vinculadas en los diferentes sectores de la sociedad como por ejemplo: educación, deporte, economía, cultura, salud, entre otros. El desarrollo vertiginoso de las Tecnologías de Información y las Comunicaciones (TIC), ha logrado un avance informático en las distintas esferas haciendo énfasis en la salud, permitiendo la obtención de sistemas de información que garantizan diagnósticos rápidos y confiables.

Los sistemas para la gestión de la información, son útiles para agilizar los procesos de atención médica y administrativos en las instituciones hospitalarias, lo que influye de forma directa en la calidad de los servicios brindados al paciente.

Para las instituciones de salud se han desarrollado soluciones que tienen como objetivo final informatizar y optimizar las actividades que se realizan en el proceso de atención al paciente, con el fin de incrementar su eficiencia; siendo una de estas los Sistemas de Información Hospitalaria (HIS por sus siglas en inglés).

Los HIS son aquellos sistemas de información orientados a satisfacer las necesidades de almacenamiento, procesamiento e interpretación de los datos médico-administrativos generados en una institución hospitalaria. Este tipo de sistema posibilita una mayor eficiencia en la gestión de los recursos humanos y

materiales y en los procesos que enfrentan los pacientes para obtener las acciones de salud que necesitan. A partir de un HIS se pueden obtener reportes e informes estadísticos, en dependencia del área o servicio que lo requiera, permitiendo la retroalimentación en el desempeño de la atención de salud y como consecuencia posibilitar un aumento de la calidad de los servicios (1). Estos constituyen un apoyo para las actividades y procesos de cualquier centro asistencial en todos sus niveles.

La Universidad de la Ciencias Informáticas (UCI), como institución de enseñanza superior de nuevo tipo, cuenta con varios centros de desarrollo de software, siendo el Centro de Informática Médica (CESIM) el encargado de desarrollar productos, servicios y soluciones informáticas para la optimización del trabajo y mejoramiento de la calidad de la atención médica, contribuyendo a la formación integral de profesionales y permitiendo un posicionamiento en el mercado nacional e internacional. El CESIM cuenta con varios departamentos, dentro de los que se encuentra el departamento de Sistemas de Gestión Hospitalaria. El mismo desarrolla varios productos, entre los cuales es importante señalar el Sistema de Información Hospitalaria alas HIS, el cual constituye un sistema integral para la gestión hospitalaria.

Dicho sistema está compuesto por diferentes módulos que interconectan las distintas áreas de una institución hospitalaria como son: Emergencia, Epidemiología, Laboratorio, Banco de Sangre, Farmacia, Consulta Externa, Hospitalización, Admisión y otros. El módulo de Admisión comprende la gestión de las historias clínicas, admisiones y reportes asociados al movimiento hospitalario. En la actualidad este módulo aún no cuenta con todas las prestaciones que permitan garantizar la accesibilidad total por todo tipo de usuario, sin importar qué limitaciones posean, entre las que se encuentra la discapacidad visual. Esto trae como consecuencia que este tipo de personas no pueda interactuar con la aplicación, privándolos de aumentar su desarrollo profesional.

Es deber de la sociedad el equiparar las oportunidades de las personas con discapacidad visual en la búsqueda, obtención, mantenimiento y superación de los puestos de trabajo con mejores condiciones, erradicando los actos de discriminación laboral en contra de estas personas.

Privar de una plaza de trabajo a un discapacitado visual por no poseer las condiciones necesarias para operar un sistema informático es una problemática en la cual se debe incidir para evitarlo. En ocasiones los

usuarios finales de los sistemas desarrollados son personas con algún tipo de déficit visual, por lo que se considera de gran importancia eliminar ciertas barreras arquitectónicas que impiden el uso de estas aplicaciones informáticas.

Para la inserción de estas personas en el mundo de las tecnologías informáticas se han creado varias aplicaciones que le brindan cierta ayuda; permitiendo que el horizonte de usuarios para los cuales pueden estar destinados la inmensa mayoría de las aplicaciones convencionales se amplíe. Con ello se logra también un aumento del mercado del software.

Basado en lo antes expuesto se plantea el siguiente **problema a resolver:** ¿Cómo lograr que usuarios con deficiencia visual puedan interactuar con el módulo de Admisión del sistema alas HIS?

En correspondencia con el problema el **objeto de estudio** lo constituye: usabilidad de Sistemas de Información por usuarios discapacitados visuales. Enmarcado en el **campo de acción:** usabilidad del módulo de Admisión del sistema alas HIS por usuarios con discapacidad visual.

Para resolver el problema identificado se propone el siguiente **objetivo general:** desarrollar funcionalidades que faciliten el uso del módulo Admisión del sistema alas HIS a usuarios discapacitados visuales.

Para dar cumplimiento al objetivo anteriormente planteado se definen las siguientes **tareas de investigación:**

- Valorar las tendencias actuales de las soluciones informáticas relacionadas con el uso de sistemas por usuarios discapacitados visuales así como las tecnologías y la arquitectura definida para el desarrollo del sistema alas HIS.
- Obtener los artefactos correspondientes a los flujos de trabajo “Modelado de Negocio” y “Requisitos”.
- Realizar el Diseño de la solución propuesta.
- Desarrollar las funcionalidades identificadas:
    - Funcionalidad que aumente el tamaño del contenido a visualizar en las interfaces y facilite la interacción con el sistema a los débiles visuales.

- Funcionalidad que permita a las personas con discapacidad visual leer la interfaz describiendo las opciones donde se posicione el cursor por un tiempo definido y las acciones que deben realizar para continuar.
- Funcionalidad que permita reutilizar texto obtenido de herramientas profesionales para el dictado, leyendo desde ficheros externos que generen estas herramientas con una estructura definida para la asociación correcta de datos.

Con el desarrollo de las funcionalidades para el uso del módulo Admisión del sistema alas HIS por usuarios discapacitados visuales se esperan obtener los siguientes ***beneficios***:

1. Posibilitar la interacción del personal discapacitado visual con el módulo Admisión del sistema alas HIS.
2. Lograr un mayor impacto social del sistema alas HIS.

El documento se encuentra estructurado en cuatro capítulos, estructurados de la siguiente manera:

Capítulo 1: Fundamentación teórica de las soluciones informáticas relacionadas con el uso de sistemas por usuarios discapacitados visuales y herramientas de desarrollo: se realiza un estudio preliminar de las soluciones informáticas existentes que brindan ayuda a personas discapacitadas visuales. Además, son fundamentadas las tecnologías, metodologías y herramientas de desarrollo a utilizar.

Capítulo 2: Características de las funcionalidades para discapacitados visuales en el módulo de Admisión del alas HIS: se describen las principales características de la solución y se presenta el Modelo de Dominio, conjuntamente con la especificación de los requerimientos funcionales, no funcionales y el Modelo de Casos de Usos del Sistema.

Capítulo 3: Diseño de las funcionalidades para discapacitados visuales en el módulo de Admisión del alas HIS: se realiza una descripción y análisis de la estructura de la solución que se propone para dar respuesta a la problemática planteada. Se fundamenta la arquitectura empleada, así como las estrategias de integración a tener en cuenta.

Capítulo 4: Implementación de las funcionalidades para discapacitados visuales en el módulo de Admisión del alas HIS: se introduce el flujo de trabajo de implementación, partiendo de los resultados obtenidos en el diseño y se exponen aspectos referentes a la seguridad del sistema, las estrategias de codificación, así como la forma en que se tratarán los errores.

CAPÍTULO 1. Fundamentación teórica de las soluciones informáticas relacionadas con el uso de sistemas por usuarios discapacitados visuales y herramientas de desarrollo

En este capítulo se identifican y describen los conceptos más importantes que tratan la usabilidad del módulo de Admisión del sistema alas HIS por usuarios con discapacidad visual. Se aborda el estudio del arte de los sistemas que brindan ayuda a personas discapacitadas visuales que poseen características semejantes al propuesto, analizándose a nivel internacional y nacional. Son expuestas las tecnologías, lenguajes, metodologías y herramientas que se utilizarán en el desarrollo de la solución propuesta.

1.1. Conceptos básicos relacionados con el dominio del problema

Discapacidad visual: se define como la alteración del sistema visual que trae como consecuencia dificultades en el desarrollo de actividades que requieran el uso de la visión. En el contexto de la discapacidad visual se encuentran las personas ciegas y con baja visión (2).

Existen diferentes grados de pérdida de visión, que abarcan desde las deficiencias visuales (pérdida parcial) a la ceguera (pérdida total de visión).

Amplificadores de pantalla (Zoom): lupa o lente que permite magnificar cualquier área de la pantalla de la computadora.

Sintetizador de voz: dispositivo que se encarga de la producción artificial del habla humana. La mayor parte de los dispositivos sintetizadores de voz tienen memorizados digitalmente cada uno de los fonemas o palabras que son capaces de emitir. Los datos que recibe un sintetizador procedente del ordenador corresponden a la identificación de los fonemas o palabras a emitir. Una vez que se analiza el dato, se activa una rutina encargada de generar el sonido correspondiente. Para lograr la más completa claridad en el habla, lo adecuado es almacenar pronunciaciones de palabras completas. Los sintetizadores de habla son muy útiles para ser usados por aquellas personas con discapacidades; por ejemplo, se utiliza un sintetizador de habla en programas de asistencia como los lectores de pantalla.

Narrador de Windows: conversor de texto a voz, diseñado pensando en los usuarios invidentes o con problemas de visión.

1.2. Sistemas automatizados existentes vinculados a la interacción de los usuarios discapacitados visuales con los Sistemas de Información

Zoom

Amplificador de pantalla que puede ser descargado de Internet en segundos e instalado muy fácilmente. Recomendable para personas que pueden aprovechar su resto visual. Puede amplificar cualquier área de la misma con varios niveles de magnificación. Para su uso se deben realizar adecuaciones a la configuración de la pantalla y distribución de los programas, menús y herramientas a fin de hacer el contenido a visualizar más asequible.

JAWS

Programa que lee la pantalla a las personas con discapacidad visual para que sepan la opción en la que están colocados y las acciones que deben realizar para continuar. Para navegar en los diferentes menús de la pantalla con esta aplicación la persona ciega utiliza el teclado de la computadora y prescinde del ratón. Da apoyo en español a las diferentes aplicaciones de Windows. Para optimizar su utilización se recomienda reducir los elementos que aparecen en los diferentes entornos visuales que sean utilizados para así eliminar elementos extras y se reduzca el tiempo de ejecución. Lo más interesante es que, además de mencionar lo que está en la interfaz lo explica, logrando así un funcionamiento de fácil acceso para las personas con discapacidad visual. Cabe mencionar que Jaws no es gratuito.

DSpeech

DSpeech es un programa para la Conversión de texto a voz o TTS (Text ToSpeech por sus siglas en inglés) y además hace uso de la función integrada de RAH (Reconocimiento automático del habla). Por tanto, DSpeech es capaz de decir en voz alta el texto que escribas. Para ello utilizará cualquiera de los modelos de vocalización que incluye Microsoft. Entre las utilidades de DSpeech se encuentra la posibilidad de reproducir el contenido de tu portapapeles y la opción de grabar en formato WAV o MP3.

Lazarux

Distribución GNU/Linux operando desde un Live-CD, adaptada a las necesidades informáticas de deficientes visuales de habla hispana, que incluye un amplio conjunto de aplicaciones accesibles y un motor de voz totalmente en español.

Narrador de Windows

Describe algunos eventos que se generan al usar el equipo. Es una herramienta destinada para los usuarios con ausencia total de la visión. Lee lo que aparece en la pantalla, los contenidos de la ventana activa, las opciones del menú o el texto que se ha escrito. El Narrador se ha creado para poder utilizarlo con Notepad, WordPad, los programas del panel de control, Internet Explorer y el escritorio de Windows además en algunas partes de la configuración de Windows. Ofrece una serie de opciones con las que puede personalizar el modo de lectura de los elementos de la pantalla. Es probable que el Narrador no lea algunas palabras de manera correcta cuando se utiliza en otros programas.

Lupa de Windows

Aumenta diferentes partes de la pantalla y forma parte del Centro de accesibilidad. También puede cambiar la resolución de pantalla, que ajusta la claridad, el tamaño y la cantidad de objetos que pueden aparecer en el monitor. Es un instrumento que permite ver más de cerca cosas que naturalmente no podemos. Existen personas que ya sea por estar lejos del monitor o por algún tipo de disfunción visual necesitan activar el ampliador para poder leer con normalidad.

HPR de IBM

Permite una navegación simple, rápida y eficiente al usuario por medio del teclado numérico o el teclado regular del computador. Viene con un sintetizador de voz integrado que utiliza la tarjeta de sonido de la computadora para leer en voz sintetizada el contenido de las páginas del Internet. El HPR puede navegar por los textos, imágenes, enlaces, formas, tablas y otros elementos de las páginas web.

Incluye opciones especiales para navegar tablas complejas. Para utilizar este programa, se debe tener instalado la versión 5 del navegador Internet Explorer.

Como parte de la investigación sobre los antecedentes de los sistemas que brindan ayuda a los usuarios discapacitados, se obtuvieron además otros resultados los cuales se muestran a continuación:

Nombre del software	Tipo de software	Tipo de licencia	Plataforma
Zoom V1.01	Desktop	Gratuito	Windows
JAWS	Desktop	Privativo	Windows
DSpeech	Desktop	Gratuito	Windows
Lazarux:	Desktop	Gratuito	Linux
Narrador de Windows	Desktop	Privativo	Windows
Lupa de Windows	Desktop	Privativo	Windows
HPR	Desktop	Privativo	Windows

Tabla 1.1 Comparación de los sistemas existentes que brindan ayuda a discapacitados visuales.

Muchas de las herramientas informáticas que existen en el mundo para discapacitados visuales son de uso libre lo que no quiere decir que sean de código abierto, imposibilitando con ello que se pueda realizar un estudio de su estructura y código para una posible reutilización en la solución deseada. Para otras se tiene que pagar una licencia, incurriendo en gastos adicionales para el desarrollo de la solución. Estas herramientas por ser aplicaciones de escritorio, no permiten la integración al sistema de gestión hospitalaria alas HIS y por ende tampoco permiten su uso en las diferentes plataformas para las cuales funciona este sistema.

Por las razones expuestas se hace evidente la necesidad de crear varias de estas funcionalidades en software libre, que brinden una integración completa al módulo de Admisión del Sistema de Información Hospitalaria alas HIS y que permitan la utilización de este por usuarios discapacitados visuales. No obstante,

el desarrollo de las funcionalidades de la presente investigación, se basa en características y funcionalidades que presentan estos sistemas estudiados.

1.3. Tecnologías, Metodologías, Herramientas y Lenguajes utilizados

Las tecnologías utilizadas en el proceso de desarrollo de las funcionalidades han sido agrupadas según el modelo arquitectónico escogido. Una adecuada elección de estas herramientas es de suma importancia para que la solución sea desarrollada con las tecnologías necesarias que permitan una integración completa al módulo de Admisión del sistema alas HIS, posibilitando una mayor organización y calidad en el presente trabajo. Las mismas se encuentran definidas por el departamento de Sistemas de Gestión Hospitalaria para el desarrollo de sus aplicaciones.

1.3.1. *Arquitectura de las funcionalidades*

Arquitectura basada en componentes

Consiste en una rama de la Ingeniería de software en la cual se trata con énfasis la descomposición del software en componentes funcionales. Esta arquitectura se enfoca en la descomposición del diseño en componentes funcionales o lógicos que expongan interfaces de comunicación bien definidas.

Es un acercamiento basado en la reutilización para definir, implementar, y componer, componentes débilmente acoplados en sistemas. Esta práctica apunta traer igualmente un amplio grado de beneficios tanto en el corto como el largo plazo, para el software en sí mismo (3).

Arquitectura Cliente-Servidor

Desde el punto de vista funcional, se puede definir la computación cliente-servidor como una arquitectura distribuida que permite a los usuarios finales obtener acceso a la información en forma transparente aún en entornos multiplataforma.

En el modelo cliente-servidor, el cliente envía un mensaje solicitando un determinado servicio a un servidor (hace una petición), y este envía uno o varios mensajes con la respuesta (provee el servicio). En un sistema distribuido cada máquina puede cumplir el rol de servidor para algunas tareas y el rol de cliente para otras (4).

1.3.2. *Tecnologías y lenguajes utilizados en el proceso de desarrollo*

Java

Otra característica que distingue al sistema alas HIS es que debe estar libre del costo relacionado con patentes de software**,** asociadas al servidor de aplicaciones, al servidor de base de datos, al sistema operativo huésped u otras herramientas o tecnologías utilizadas para su desarrollo. Para lograr este objetivo se propone el uso de un lenguaje de programación multiplataforma, como es el caso de Java.

Java es un lenguaje potente, seguro, universal y gratuito. Una de las principales características por las que se ha popularizado es su independencia de plataforma, por esto su uso extendido en Internet, ya que muchas personas deben tener acceso con ordenadores distintos. Con Java se pueden programar páginas web dinámicas, con acceso a base de datos, utilizando XML y con cualquier tipo de conexión de red entre cualquier sistema (5).

Java se volvió más popular a partir de la aparición de la especificación de Servlets y JSP (Java Server Pages) una tecnología orientada a crear páginas web. Los Servlets y las JSPs supusieron un importante avance ya que el API (Interfaz de Programación de Aplicaciones) es muy sencillo, flexible y extensible (6).

Java Server Faces (JSF)

Java Server Faces (JSF) es una plataforma de trabajo para desarrollo basado en el Modelo-Vista-Controlador (MVC). JSF está incluida en la plataforma Java Enterprise Edition, por lo que se pueden crear aplicaciones que la utilizan sin añadir bibliotecas adicionales en el proyecto. Esta plataforma de trabajo permite desarrollar de forma sencilla y eficaz interfaces de usuarios muy distintas a las acostumbradas en las aplicaciones Web (7).

RichFaces

RichFaces es una librería de componentes visuales para JSF. Posee un framework avanzado para la integración de funcionalidades Ajax en dichos componentes visuales, mediante el soporte de la librería Ajax4JSF.

RichFaces se integra perfectamente en el ciclo de vida de JSF, incluye funcionalidades Ajax, provee varias librerías de componentes como son Core, Ajax y UI, así como administración avanzada de recursos como

imágenes, código JavaScript y Hojas de Estilo en Cascada (CSS). Es posible crear interfaces de usuario de manera rápida y eficiente, basado en componentes que están listos para usar y son altamente configurables. RichFaces, además, es un proyecto que fue desarrollado con una arquitectura abierta para que fuera compatible con la mayor cantidad de entornos (8).

Ajax4JSF

Ajax es una tecnología asíncrona, en el sentido de que los datos adicionales se obtienen del servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página. JavaScript es el lenguaje interpretado en el que normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante XML Http Request, objeto disponible en los navegadores actuales. En cualquier caso, no es necesario que el contenido asíncrono esté formateado en XML. Ajax es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores, dado que está basado en estándares abiertos como JavaScript y Document Object Model (DOM) (9).

Es una librería de código abierto que se integra totalmente en la arquitectura de JSF y extiende la funcionalidad de sus etiquetas dotándolas con tecnología Ajax de forma limpia y sin añadir código JavaScript. Mediante este framework se puede variar el ciclo de vida de una petición JSF, recargar determinados componentes de la página sin necesidad de recargarla por completo, realizar peticiones automáticas al servidor, control de cualquier evento de usuario, etc. Esta librería permite dotar a la aplicación JSF de contenido mucho más profesional con muy poco esfuerzo (10).

XHTML

XHTML, acrónimo inglés de extensible Hypertext Markup Language (lenguaje extensible de marcado de hipertexto), es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web. Su objetivo es avanzar en el proyecto del World Wide Web Consortium (W3C) de lograr una web semántica, donde la información y la forma de presentarla estén claramente separadas (11).

Cascading Style Sheets (CSS)

Las hojas de estilo en cascada son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C (World Wide Web Consortium) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los

agentes de usuario o navegadores. Lo que se persigue con el desarrollo de CSS es separar la estructura de un documento de su presentación (12).

JBoss Seam

JBoss Seam es un framework que integra la capa de presentación (JSF) con la capa de negocios y persistencia (EJB), funcionando, según versa su significado en español, como una “costura” entre estos componentes. Seam también se integra perfectamente con otros frameworks como: RichFaces, ICE Faces, MyFaces, Hibernate y Spring (13).

JavaScript

Es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Al utilizar este lenguaje no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.

Utilizado para crear pequeños programitas encargados de realizar acciones dentro del ámbito de una página web. Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado. Es bastante sencillo y pensado para hacer las cosas con rapidez, a veces con ligereza. Incluso las personas que no tengan una experiencia previa en la programación podrán aprender este lenguaje con facilidad y utilizarlo en toda su potencia con sólo un poco de práctica (14).

JQuery

JQuery es considerado un Framework de JavaScript. Es decir, un conjunto de funciones que ya fueron desarrolladas y probadas, están listas para utilizarlas de una manera muy simplificada. En otras palabras, podremos lograr los mismos resultados, en menos tiempo sin necesidad de programar una funcionalidad completamente (15). Brinda la posibilidad de trabajar con Ajax, sin preocuparnos de los detalles complejos de la programación. Además cuenta con la posibilidad de agregar plugins, facilitando más aun nuestro trabajo.

Mespeak

Es una biblioteca de texto a voz JavaScript del lado del cliente 100% basado en el proyecto speak.js, añade soporte para Webkit y Safari e introduce módulos de voz se pueden cargar. Además, no hay necesidad de una incorporación de elementos HTML.

1.4. Tecnologías horizontales

Java Platform Enterprise Edition (JavaEE 5)

Java Platform Enterprise Edition o Java versión 5 es una plataforma de programación (parte de la Plataforma Java) para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura de N niveles distribuida. Se basa ampliamente en componentes de software modulares y se ejecuta sobre un servidor de aplicaciones (16).

Java Runtime Environment (JRE 6)

JRE es el acrónimo de Java Runtime Environment (entorno en tiempo de ejecución Java) y se corresponde con un conjunto de utilidades que permite la ejecución de programas Java sobre todas las plataformas soportadas. JVM (máquina virtual Java) es una instancia de JRE en tiempo de ejecución. Este interpreta el código Java y está compuesto además por las librerías de clases estándar que implementan el API de Java. Ambas JVM y API deben ser consistentes entre sí, de ahí que sean distribuidas de modo conjunto (17).

JBoss Application Server

JBoss Application Server es el servidor de aplicaciones de código abierto más ampliamente desarrollado del mercado. Por ser una plataforma certificada J2EE, soporta todas las funcionalidades de J2EE 1.4 e incluye servicios adicionales como clustering, caching y persistencia. JBoss es ideal para aplicaciones Java y aplicaciones basadas en la web. También soporta Enterprise Java Beans (EJB) 3.0, lo que hace el desarrollo de las aplicaciones mucho más simple. Además, al ser desarrollado con tecnología Java, es multiplataforma (18).

1.5. Metodologías de desarrollo de software

Proceso unificado de desarrollo (RUP)

RUP es el resultado de varios años de trabajo y uso práctico en el que se han unificado técnicas de desarrollo, a través del UML, y trabajo de muchas metodologías utilizadas por los clientes. En RUP se han agrupado las actividades en grupos lógicos en los que se definen nueve flujos de trabajo principales. Los seis primeros son conocidos como flujos de ingeniería y los tres últimos como flujos de apoyo. El ciclo de vida de RUP se caracteriza por ser dirigido por caso de uso, centrado en la arquitectura, iterativo e incremental (19).

Lenguaje Unificado de Modelado (UML)

UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. Permite la modelación de sistemas con tecnología orientada a objetos. Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como RUP), pero no especifica en sí mismo qué metodología o proceso utilizar.

Este lenguaje de modelado formal permite tener un mayor rigor en la especificación, realizar una verificación y validación del modelo desarrollado, automatizar determinados procesos y generar código a partir de los modelos y a la inversa. Esto último permite que el modelo y el código estén actualizados (20).

1.6. Herramientas

Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma típicamente ha sido usada para desarrollar entornos de desarrollo integrados (del inglés IDE), como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se entrega como parte del Eclipse.

El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la plataforma de los llamados clientes enriquecidos. Esto lo diferencia de

otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software, que adicionalmente permite al Eclipse extenderse, usando otros lenguajes de programación como C/C++, Python y Java (21).

JBoss Tools

Es un conjunto de plug-in para el Eclipse que permite el manejo de diferentes frameworks que facilitan el desarrollo de aplicaciones. Está constituido por varios módulos: RichFaces VE, Seam Tools, Hibernate Tools y JBoss AS Tools (22).

Visual Paradigm para UML

Visual Paradigm para UML (Lenguaje Unificado de Modelado) es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a la construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML (23).

Conclusiones

El estudio de las soluciones informáticas existentes que brindan ayuda a personas con deficiencias visuales, permitió concluir que no posibilitan su uso en las diferentes plataformas para las cuales funciona el alas HIS y no tienen una arquitectura compatible con dicho sistema que permita la integración al mismo. Como resultado del análisis llevado a cabo, se determina la necesidad de desarrollar las funcionalidades que posibilite el uso del módulo Admisión del Sistema de Información Hospitalaria alas HIS a usuarios con déficit visual.

Partiendo de la integración de las funcionalidades al alas HIS se acometió al estudio de la arquitectura y tecnologías definidas para el desarrollo del mismo así como de otras librerías necesarias que dan paso al cumplimiento del objetivo propuesto.

CAPÍTULO 2. Características de las funcionalidades para discapacitados visuales en el módulo de Admisión del alas HIS

En este capítulo se tiene como objetivo abordar los diferentes elementos que brindan la base teórica y conceptual para el desarrollo de las funcionalidades. Al no existir una definición clara de los procesos del negocio, se determina desarrollar un Modelo de Dominio, donde se abarcan las definiciones asociadas a la solución. Además, se enuncian los requisitos funcionales agrupados en los diagramas de casos de uso correspondientes.

2.1. Modelo de Dominio

El Modelo de Dominio o Modelo Conceptual es una representación visual de los conceptos u objetos que se manejan en el dominio de la solución propuesta. Los objetos o conceptos incluidos en el modelo de dominio no describen clases u objetos del software; sino entidades o conceptos del mundo real que están asociadas al problema en cuestión. Cuando se realiza la programación orientada a objetos, el funcionamiento interno del software va a imitar en alguna medida a la realidad, por lo que el mapa de conceptos del modelo de dominio constituye una primera versión para el desarrollo de las funcionalidades. Dicho modelo podrá ser utilizado como una base de las abstracciones relevantes en el proceso de construcción de las mismas.

2.2. Conceptos fundamentales del dominio

Con la finalidad de una mejor comprensión del Diagrama del Modelo de Dominio a continuación se dará una breve descripción de los conceptos encontrados en el ámbito del problema.

Lector Interfaz: es el que lee lo que se encuentra en la pantalla a medida que el usuario realice el recorrido con el cursor por encima de los textos.

Zoom: aumenta el tamaño del contenido de la interfaz en dependencia de la escala que el usuario seleccione.

Usuario: persona discapacitada que interactúa con el sistema durante la ejecución de las funcionalidades.

Lector Fichero: permite leer los ficheros externos que el usuario requiera.

Operaciones: son las acciones que el usuario realizará.

Fichero: fichero externo que el usuario necesite.

Interfaz: es la que tiene incluidas las funcionalidades que se llevarán a cabo.

2.2.1 Diagrama del Modelo de dominio

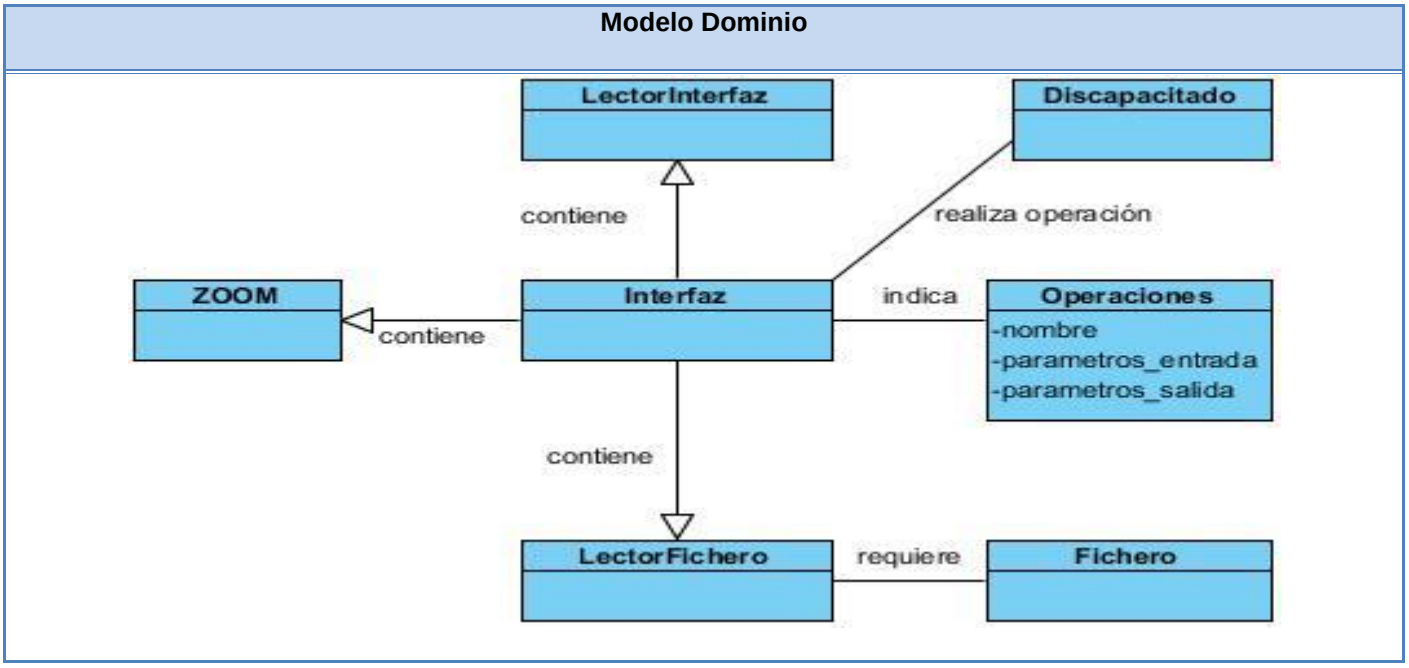

Figura 1.1 Diagrama Modelo de Dominio

2.3. Especificación de los requerimientos de software

Un requerimiento es una condición o capacidad que debe tener un sistema o un componente de este para satisfacer un contrato, norma, especificación u otro documento formal, facilitando el entendimiento entre clientes y desarrolladores.

2.3.1. *Requisitos funcionales del sistema*

Los requisitos funcionales especifican acciones que debe poder realizar un software, sin tener en cuenta las restricciones físicas, además definen su comportamiento de salida y entrada. Seguidamente se exponen los requisitos funcionales, por los cuales se rige el desarrollo de la aplicación.

RF1 Leer texto en pantalla.

- Permite leer todo el texto que aparece en pantalla a medida que el usuario se posicione encima de ellos con el cursor.

RF2 Leer fichero externo.

- Permite leer el contenido de un fichero externo seleccionado por el usuario.

RF3 Magnificar interfaz.

- Permite aumentar el tamaño del contenido de la interfaz.

RF4 Activar opción Leer texto en pantalla.

- Permite habilitar o deshabilitar la opción Leer texto en pantalla.

RF5 Activar opción Leer fichero externo.

- Permite habilitar o deshabilitar la opción Leer fichero externo.

RF6 Activar opción Magnificar interfaz.

- Permite habilitar o deshabilitar la opción Magnificar interfaz.

RF7 Cambiar Escala de Magnificación.

- Modifica el tamaño del contenido de la interfaz según el valor seleccionado en la escala.

2.3.2. *Requisitos no funcionales del sistema*

Los requisitos no funcionales describen aspectos del comportamiento de un sistema, capturando las propiedades y restricciones bajo las cuales deben operar. Son requerimientos que no se refieren específicamente a la funcionalidad de una aplicación. Se imponen restricciones sobre el producto que se está desarrollando y el proceso de desarrollo, y que especifican restricciones externas que el producto debe cumplir.

Seguridad

Las funcionalidades deben mantener seguridad y control a nivel de usuarios, garantizando el acceso de los mismos sólo a los niveles establecidos de acuerdo a la función o rol que desempeñan. Las contraseñas podrán cambiarse únicamente por el propio usuario o por el administrador del sistema. Toda entrada de información estará validada, de forma que la introducción de datos incorrectos se mostrará al usuario especificándole el tipo de error.

Hardware

Los requerimientos de hardware estarán dados por la plataforma específica que se utilice para la aplicación de las funcionalidades, en cuanto a sistema operativo y servidor de aplicaciones.

Estaciones de trabajo

Se necesitan estaciones de trabajo de 1GB de memoria RAM y un microprocesador Intel® Core-2 Duo o Intel® Dual-Core, con sistema operativo Windows o Linux.

Servidores

Para los servidores la solución estará conformada, fundamentalmente, por alta capacidad de procesamiento y redundancia, que permitan garantizar movilidad y residencia de la información y las aplicaciones bajo esquemas seguros y confiables.

Servidores de Base de datos: PowerEdge R910, Procesador Intel® Xeon® CPU E7- 8837 @ 2.67GHz, 16GB de memoria RAM, 1199GB de disco duro y sistema operativo Linux.

Servidores de Aplicaciones: PowerEdge R910, Procesador Intel® Xeon® CPU E7- 8837 @ 2.67GHz, 16GB de memoria RAM, 1199GB de disco duro y sistema operativo Linux.

Rendimiento

Los procesos generados por las nuevas funcionalidades en el momento de su ejecución deben ser lo suficientemente óptimo en cuanto al uso de memoria física y necesidad de procesamiento.

Software

Las funcionalidades estarán integradas al alas HIS, dicha aplicación debe poder ser desplegada en sistemas operativos Windows y Linux, utilizando la plataforma Java (Máquina Virtual de Java - Java Enterprise Edition), el servidor de aplicaciones JBoss AS y PostgreSQL para la gestión de la base de datos. Los usuarios deberán disponer de un navegador web, este puede ser Firefox 3.6, Google Chrome 14 o versiones superiores de ellos y deben tener habilitado JavaScript.

Portabilidad

La aplicación podrá ser desplegada sobre Sistemas Operativos Linux (Oracle Linux 5.5+, 6.x, Ubuntu 8.04 LTS (Long Term Support) Desktop Edition, Ubuntu Linux 10.04 y superior, entre otras versiones) y Windows (2000, XP, Server 2003, Server 2008, Server 2012, Vista, 7, 8).

Interfaz de usuario

Las ventanas de la aplicación que utilizará el usuario discapacitado visual deberán contener bien estructurados los datos, además de permitir la interpretación correcta de la información. La entrada de datos incorrecta será detectada e informada al usuario discapacitado.

Requerimientos de usabilidad

Las funcionalidades deben brindar, a través de una barra de navegación, un acceso fácil y rápido a todas las opciones de la aplicación. El significado de los íconos y los textos debe ser claro para la persona que interactúe con la aplicación. Las opciones que se proveen deben ser comprensibles, sin explicaciones excesivas sobre su uso, y deben admitir flujos alternativos, como cancelar la operación.

2.4. Modelo de casos de uso del sistema

El modelo de casos de uso del sistema documenta el comportamiento del sistema desde el punto de vista del usuario, permitiendo representar las funciones que se desean en el sistema (*casos de uso*), el entorno del sistema (*actores*), y las relaciones entre ellos. Aunque la parte más visible de dicho modelo son los diagramas de casos de uso, suele ir acompañado de una especificación textual de cada uno de los casos de uso (1).

Los actores del sistema no forman parte del mismo, sino que representan elementos que interactúan con él. Estos elementos son nombrados roles que pueden ser desempeñados por una o varias personas, un equipo o un sistema automatizado. Un actor puede introducir o recibir información del sistema (1).

Actor	Descripción
Usuario	Se encarga de agrupar el comportamiento común de todos los usuarios del sistema que interactúan con el módulo Admisión, es el que interactúa con este durante la ejecución de sus funcionalidades.

Tabla 2.1 Definición de los actores del sistema

2.4.1. *Definición de los actores del sistema*

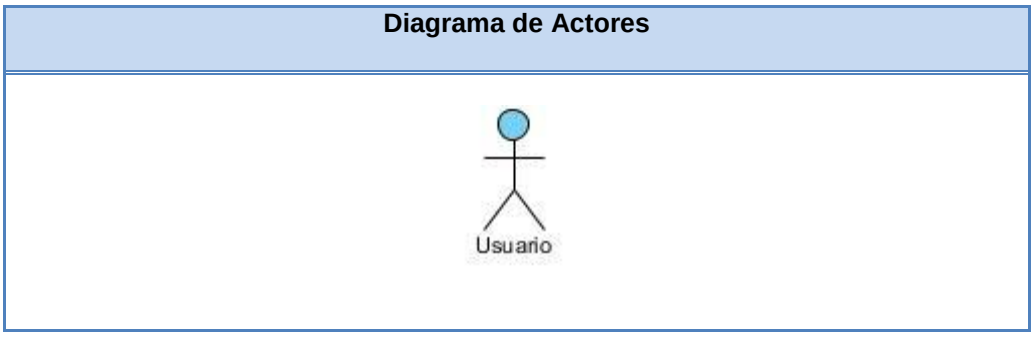

Figura 2.2 Diagrama de Actores

2.4.2. *Diagrama de Casos de Uso del Sistema*

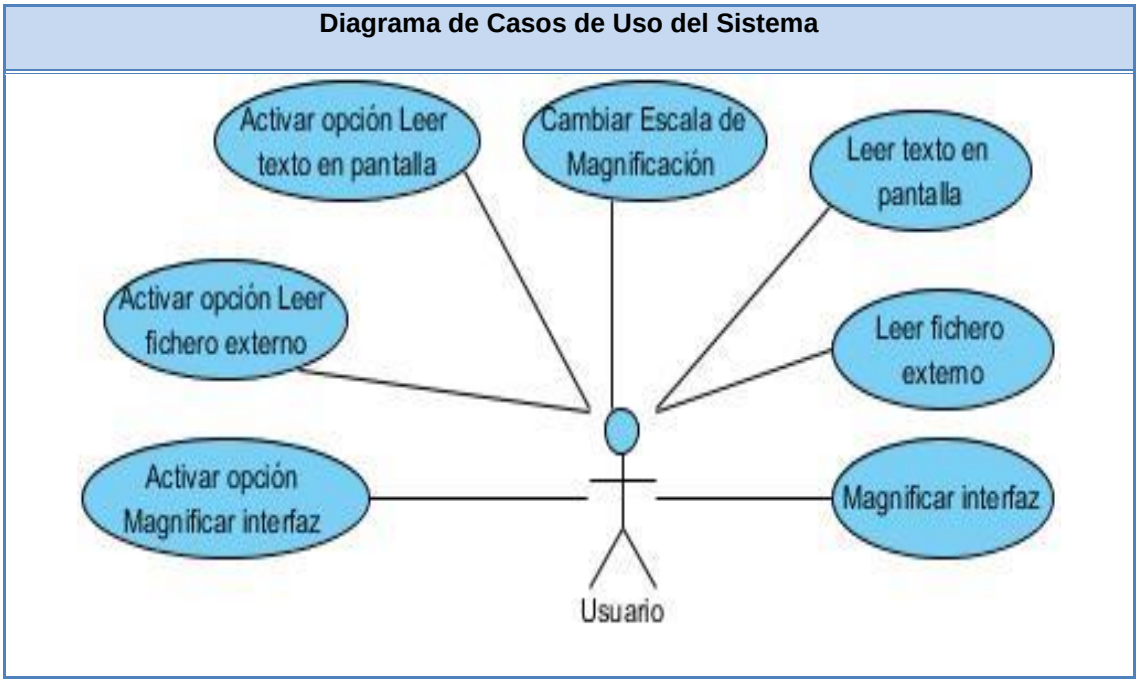

Figura 2.3 Diagrama de Caso de Uso del Sistema

2.4.3. *Descripción Textual de los Casos de Uso*

CASO DE USO:	Leer texto en pantalla
Propósito:	Permite leer todo el texto que aparece en pantalla.
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a la opción Leer texto en pantalla, luego de activarla la funcionalidad brinda la posibilidad de leer todo lo que aparece en pantalla a medida que usuario se posicione con el cursor encima de ellos, el caso de uso termina.
REFERENCIAS	
Actores:	Usuario
Requisitos:	RF4

Tabla 2.2 Descripción breve del caso de uso: Leer texto en pantalla

CASO DE USO:	Leer fichero externo
Propósito:	Permite leer el contenido de un fichero externo seleccionado por el usuario.
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a la opción Leer fichero externo, luego de activarla la funcionalidad brinda la posibilidad de adjuntar el fichero que desea, y permite leer todo el contenido del mismo, luego el caso de uso termina.
REFERENCIAS	
Actores:	Usuario

Requisitos:	RF5

Tabla 2.3 Descripción breve del caso de uso: Leer fichero externo

CASO DE USO:	Magnificar interfaz
Propósito:	Permite aumentar el tamaño del contenido de la interfaz.
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a la opción Magnificar interfaz, luego de activarla la funcionalidad brinda la posibilidad de ampliar el tamaño del contenido de la interfaz, el caso de uso termina.
REFERENCIAS	
Actores:	Usuario
Requisitos:	RF6

Figura 2.4 Descripción breve del caso de uso: Magnificar interfaz

CASO DE USO:	Cambiar Escala de Magnificación
Propósito:	Modifica el tamaño del contenido de la interfaz.
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a la opción Cambiar Escala de Magnificación, luego de activarla la funcionalidad brinda la posibilidad de seleccionar el tamaño de fuente que necesite a la hora de leer los textos, el caso de uso termina.

REFERENCIAS	
Actores:	Usuario
Requisitos:	RF7

Tabla 2.5 Descripción breve del caso de uso: Cambiar Escala de Magnificación

Conclusiones

En el presente capítulo se identificó el actor que interviene y el diagrama de casos de uso del sistema, logrando una representación detallada de cada proceso. A partir de los requerimientos funcionales fueron definidas las funcionalidades a implementar, posibilitando un mejor entendimiento de la solución propuesta. Para que la solución funcione adecuadamente debe cumplir con los requerimientos de software y hardware planteados.

CAPÍTULO 3. Diseño de las funcionalidades para discapacitados visuales en el módulo de Admisión del alas HIS

En este capítulo se realiza una descripción detallada de la solución propuesta y se explica la arquitectura de la misma. Se transforman los requisitos funcionales en un diseño de clases, donde la relación que se establece entre estas clases describe el funcionamiento de los casos de uso del sistema.

3.1. Descripción de la arquitectura, fundamentación

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema. Una Arquitectura de Software, también denominada Arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información. No es más que la estructura de un sistema informático, por lo que define los principales componentes del mismo y sus relaciones.

El componente propuesto presenta una arquitectura híbrida conformada por la arquitectura cliente-servidor y la basada en componentes.

Arquitectura Cliente-Servidor: consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta. Aunque esta idea se puede aplicar a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras. La interacción cliente-servidor es el soporte de la mayor parte de la comunicación por redes. Ayuda a comprender las bases sobre las que están construidos los algoritmos distribuidos.

Cliente: programa ejecutable que participa activamente en el establecimiento de las conexiones. Envía una petición al servidor y se queda esperando por una respuesta. Su tiempo de vida es finito una vez que son servidas sus solicitudes.

Servidor: es un programa que ofrece un servicio que se puede obtener en una red. Acepta la petición desde la red, realiza el servicio y devuelve el resultado al solicitante. Al ser posible implantarlo como aplicaciones de programas, puede ejecutarse en cualquier sistema donde exista TCP/IP y junto con otros programas de aplicación. El servidor comienza su ejecución antes de comenzar la interacción con el cliente (24).

Arquitectura Basada en Componentes

Una arquitectura basada en componentes describe una aproximación de ingeniería de software al diseño y desarrollo de un sistema. Esta arquitectura se enfoca en la descomposición del diseño en componentes funcionales o lógicos que expongan interfaces de comunicación bien definidas, lo cual provee un nivel de abstracción mayor que los principios de orientación por objetos. Además, es una especificación que define cómo crear y ensamblar los diversos componentes de negocio como componentes modulares, con la finalidad de incrementar la flexibilidad y facilidad de mantenimiento de los sistemas de información.

El objetivo de esta arquitectura es construir aplicaciones complejas mediante ensamblado de módulos (componentes) que han sido previamente diseñados por otras personas a fin de ser rehusados en múltiples aplicaciones. La arquitectura software de una aplicación basada en componentes consiste en uno o un número pequeño de componentes específicos de la aplicación (que se diseñan específicamente para ella), que hacen uso de otros muchos componentes prefabricados que se ensamblan entre sí para proporcionar los servicios que se necesitan en la aplicación.

El estilo de arquitectura basado en componentes tiene las siguientes características:

- Es un estilo de diseño para aplicaciones compuestas de componentes individuales.
- Pone énfasis en la descomposición del sistema en componentes lógicos o funcionales que tienen interfaces bien definidas.
- Define una aproximación de diseño que usa componentes discretos, los que se comunican a través de interfaces que contienen métodos, eventos y propiedades (3).

Aplicación de estas arquitecturas en la presente investigación:

La arquitectura cliente-servidor define como es la comunicación entre los elementos de la solución, ya que en el cliente es donde se muestran las interfaces que visualiza la información correspondiente y es en donde desempeñan sus labores las nuevas funcionalidades desarrolladas, brindando una mejor comprensión de los datos mostrados. En la parte del servidor se gestionan los estados de actividad para cada opción a utilizar además de la gestión de toda la información externa a mostrar. Esta última ayuda a mantener además los estados de actividad vigentes el tiempo que sea necesario ya que el cliente por sí solo no lo puede lograr.

En conjunto con lo anterior la presencia de la arquitectura basada en componentes permite la reutilización de los componentes por cada interfaz del módulo de Admisión, ya que cada funcionalidad puede trabajar por separado o en conjunto sin necesidad de hacer alguna alteración en el código implementado con anterioridad. Al utilizar componentes de terceros y tomar lo necesario de cada uno para conformar las nuevas funcionalidades, se pone en práctica uno de los principios planteados por esta arquitectura. La solución propuesta está conformada por varias funcionalidades encargadas de brindar los servicios de: magnificar el contenido de la interfaz, narrar el texto de cada elemento contenido en la misma, además de realizar la lectura de un fichero externo.

3.2. Estrategias de integración

Con el propósito de lograr una integración completa al módulo de Admisión del Sistema de Información Hospitalaria alas HIS se utilizan componentes que son de uso común para cualquier módulo que lo integran, entre los que se encuentran: las clases ActiveModule con el cual se puede conocer en que módulo está trabajando el usuario que interactúa con el sistema y la clase LocalSelector para saber el tema y el lenguaje en el que el usuario tiene configurado la aplicación.

El código que se genera para la realización de las funcionalidades fue pensado y adaptado a la estructura que hoy en día tienen las interfaces del módulo de Admisión, ya que al integrarse estas nuevas opciones no se alteran las formas en que se visualiza la información.

Las interfaces del módulo utilizan los ficheros JavaScript correspondientes a cada funcionalidad haciendo una referencia a la ubicación de estos, en dicha ubicación se encuentran además otros archivos JavaScript correspondientes a los distintos eventos generados en el módulo de Admisión.

3.3. Modelo de diseño

El Modelo de Diseño es un modelo de objetos que describe la realización de los casos de uso, es abarcador y está compuesto por artefactos que engloban todas las clases del diseño, subsistemas, paquetes, colaboraciones, y las relaciones entre ellos. Se centra principalmente en cómo los requisitos funcionales y no funcionales, junto a otras restricciones relacionadas con el entorno de implementación, tienen impacto en las funcionalidades a considerar. Es usado como una entrada inicial en las actividades de implementación y prueba.

Para la elaboración del modelo de diseño, se define una estructura de paquetes que permite dividir el sistema en fragmentos manejables para su futura implementación. Se emplea el criterio de empaquetamiento por proceso, siguiendo la estructura de procesos definidos en este.

Los paquetes son graficados mostrando la relación que guardan entre sí. Estos utilizan el paquete repositorio para su funcionamiento.

El paquete repositorio contiene dos subpaquetes, uno para las interfaces y otro para las clases Propias del Proceso. En el subpaquete de las interfaces se encuentran los ficheros JavaScript, los ficheros CSS y las páginas XHTML. Existen además 3 paquetes referentes a las funcionalidades que responden a las realizaciones de casos de uso.

Diagrama de paquetes

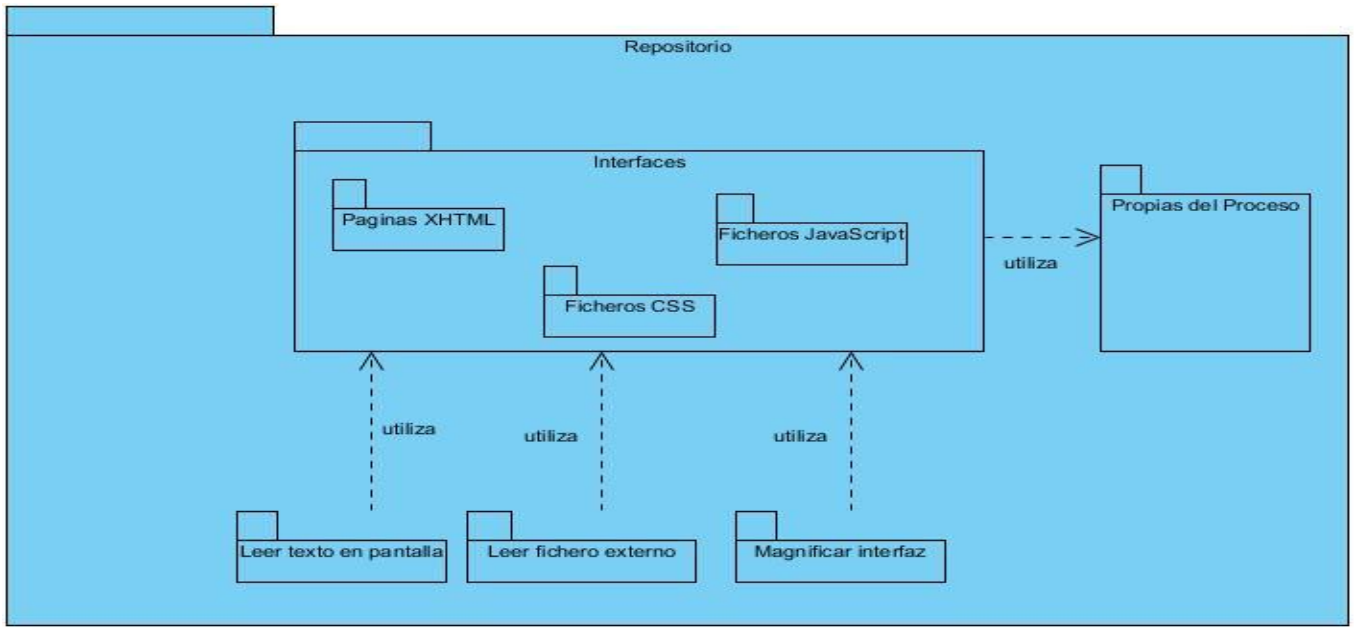

Figura 3.1 Diagrama de Paquetes

Como se evidencia en el diagrama de paquetes, en el sistema se implementan varias funcionalidades. Para este proceso se modela un diagrama de clases del diseño y por cada escenario es modelado un diagrama de interacción.

Como diagrama de interacción fue seleccionado el de secuencia. En los mismos se muestran gráficamente los eventos que origina el actor y las clases que forman parte de las funcionalidades además de las llamadas que se hacen en cada una de ellas para realizar una tarea determinada.

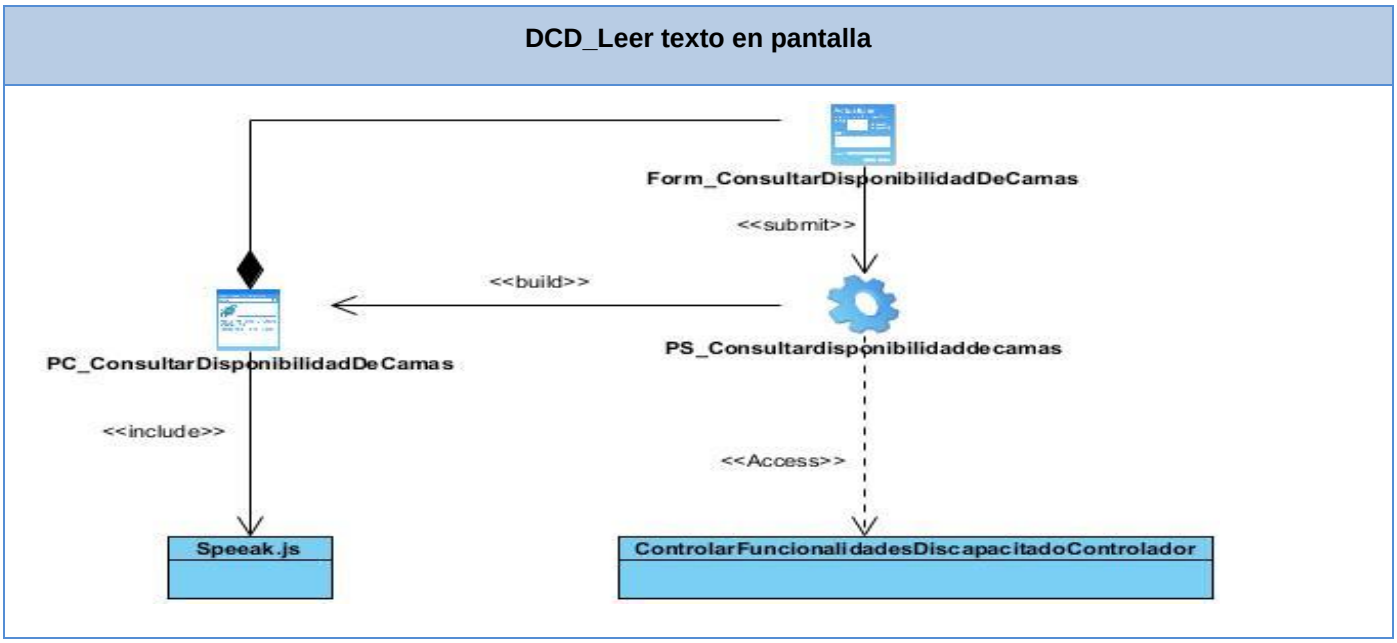

Figura 3.2 Diagrama de clases del diseño: DCD_Leer texto en pantalla

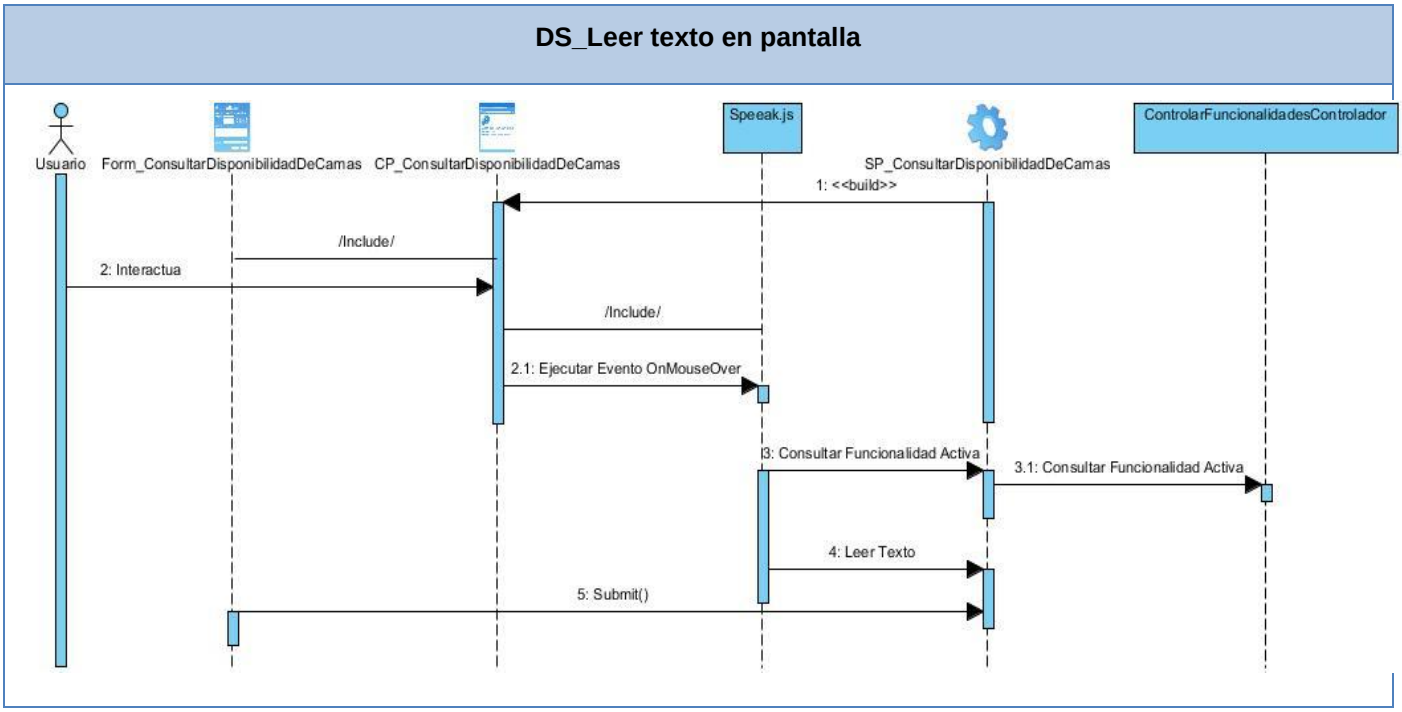

Figura 3.3 Diagrama de secuencia: DS_Leer texto en pantalla

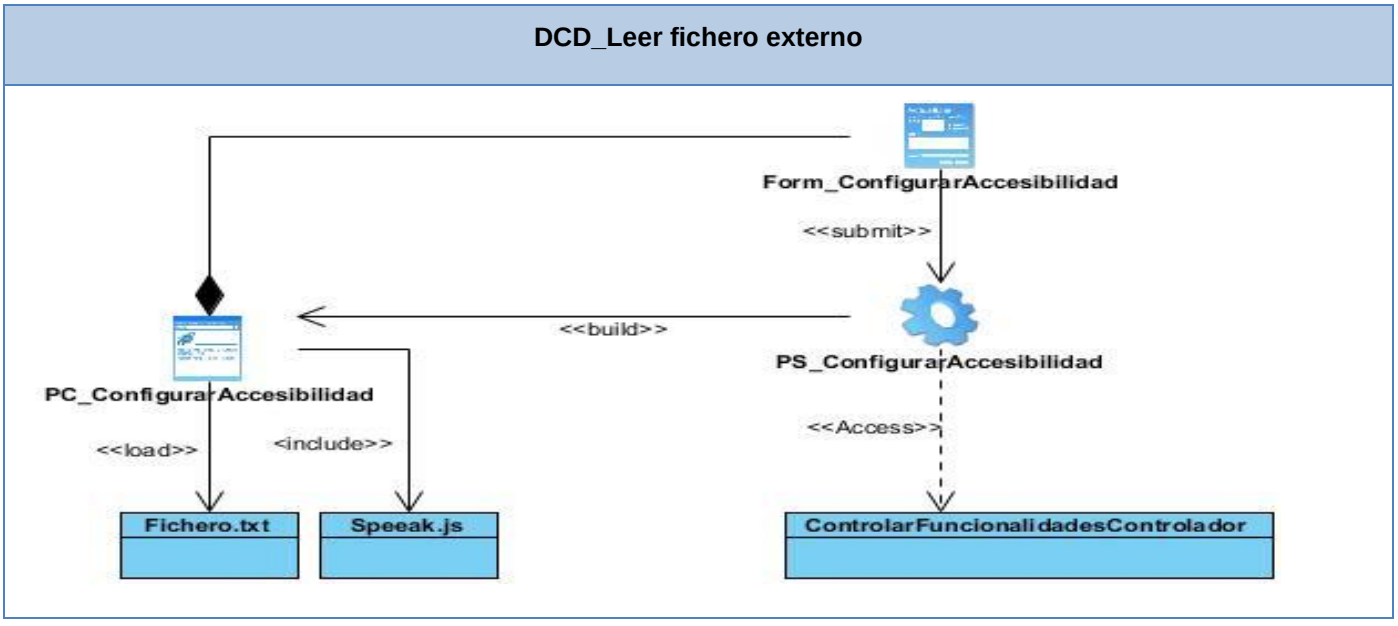

Figura 3.4 Diagrama de clases del diseño: DCD_Leer fichero externo

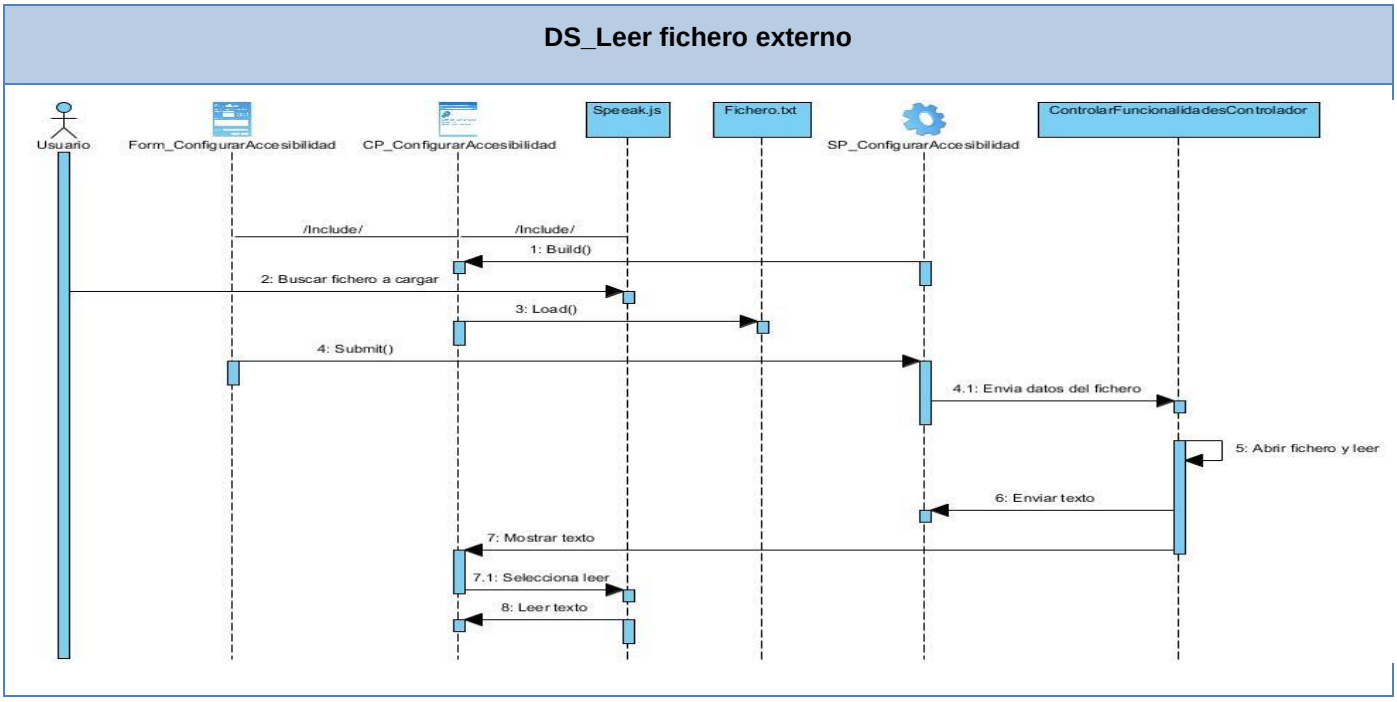

Figura 3.5 Diagrama de secuencia: DS_Leer fichero externo

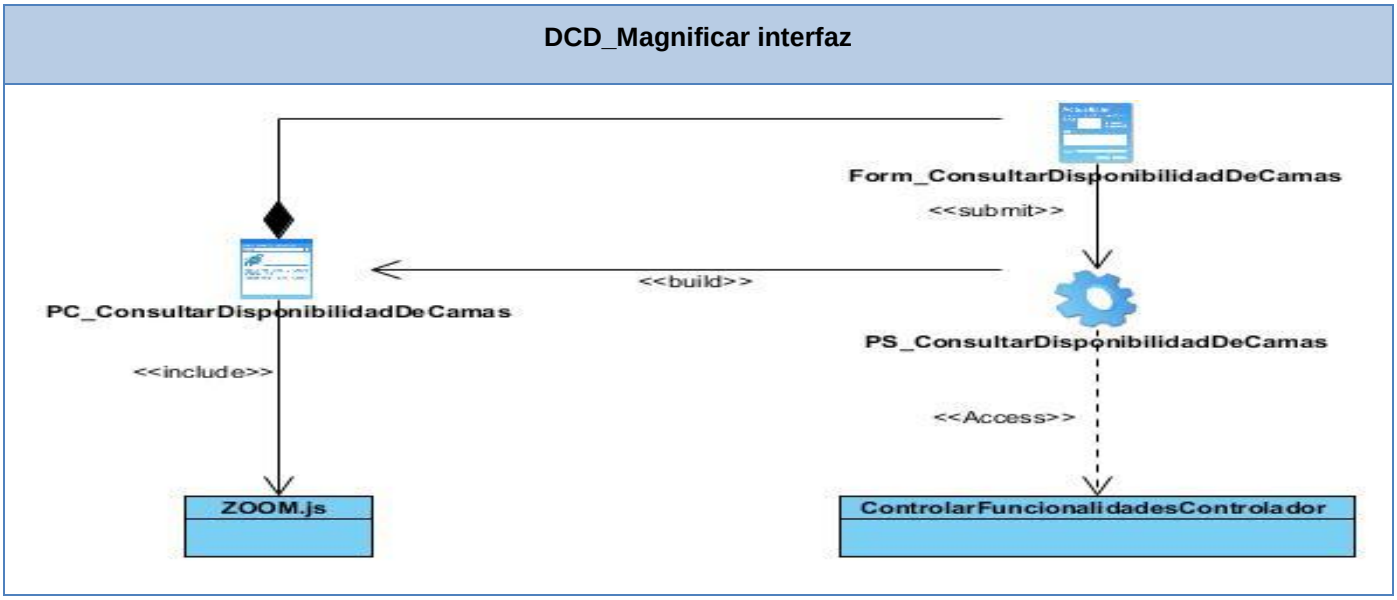

Figura 3.6 Diagrama de clases del diseño: DCD_Magnificar interfaz

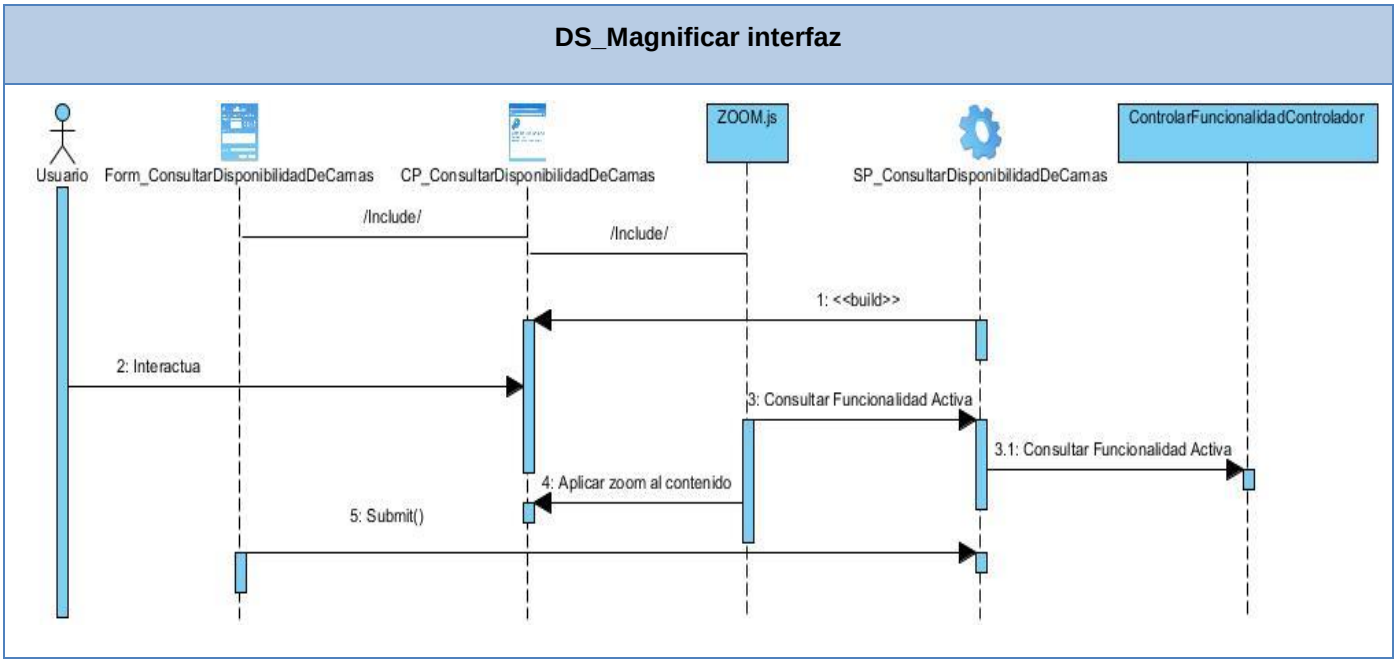

Figura 3.7 Diagrama de secuencia: DS_Magnificar interfaz

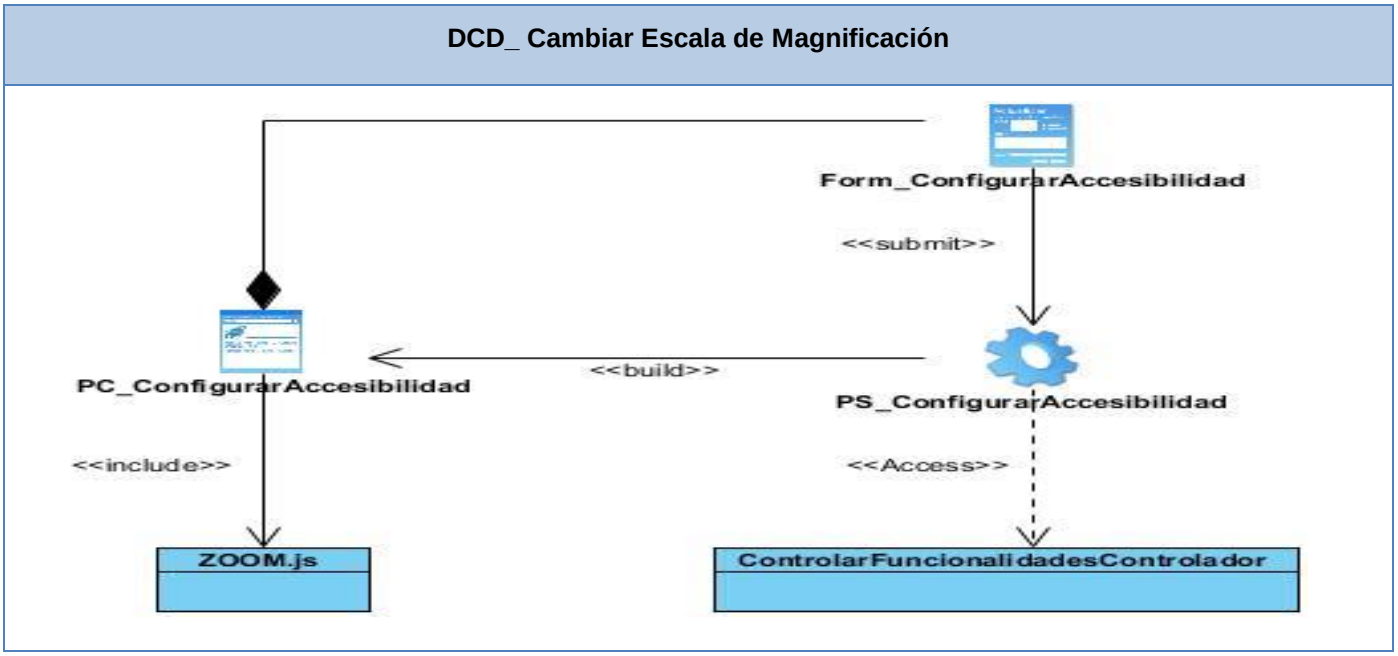

Figura 3.8 Diagrama de clases del diseño: DCD_ Cambiar Escala de Magnificación

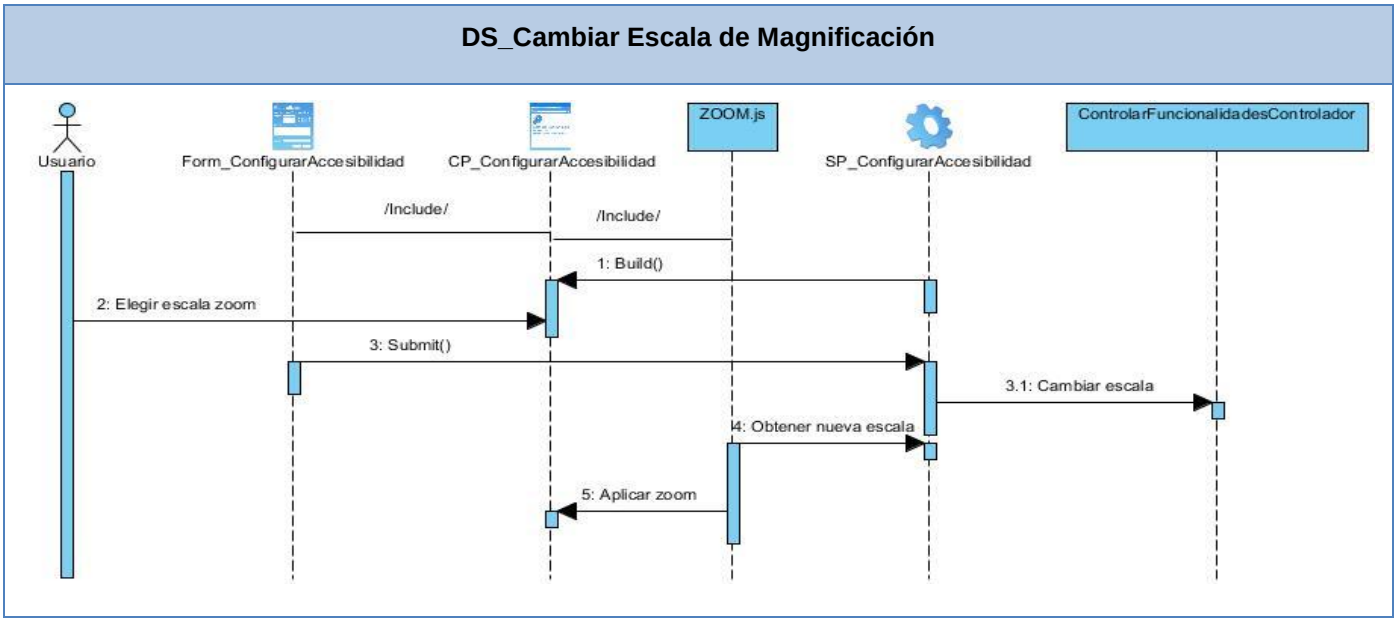

Figura 3.9 Diagrama de secuencia: DS_Cambiar Escala de Magnificación

Descripción de Clases:

Nombre: ControlarFuncionalidadesControlador	
Tipo de clase: Controladora.	
Atributo	**Tipo**
lectorPantalla	booleana
Zoom	booleana
Texto	String
textoSpeak	String
Data	bite[]
escalaString	String

Escala	Integer
EscalaZoomArray	List<String>
mostrarPanelLeer	booleana
mostrarPanelZoom	booleana
mostrarPanelFichero	booleana

Para cada responsabilidad:

Nombre:	inicializarControladoraFuncionalidades()
Descripción:	Inicializa las variables de la clase con los valores correspondientes.
Nombre:	cambiarEscalaZoom()
Descripción:	Permite asignarle a la variable escala el valor seleccionado por el usuario.
Nombre:	cargarFicheroExterno()
Descripción:	Permite procesar los datos cargados del fichero externo, analizarlos para convertirlos en textos y asignárselo a las variables texto y textoSpeak para poder mostrar la información y escucharla.
Nombre:	mostrarPanelFuncionalidadLeer()
Descripción:	Permite cambiar el valor de la variable mostrarPanelLeer de false a true para que se visualice el panel que permite activar o desactivar la funcionalidad Leer texto en pantalla.
Nombre:	mostrarPanelFuncionalidadZoom()

Descripción:	Permite cambiar el valor de la variable mostrarPanelZoom de false a true para que se visualice el panel que permite activar o desactivar y cambiar la escala de la funcionalidad Zoom.
Nombre:	mostrarPanelFuncionalidadFichero()
Descripción:	Permite cambiar el valor de la variable mostrarPanelFichero de false a true para que se visualice el panel que permite cargar el fichero del cual se quiere leer la información.

Tabla 3.1 Descripción de la clase controladora: ControlarFuncionalidadesControlador

Nombre: Speeak.js	
Tipo de clase: Fichero JavaScript.	
Variables Generales:	
speakBooleana	Variable que se encarga de almacenar e indicar a otras funciones si está activa o no la funcionalidad de leer la interfaz.
Para cada responsabilidad:	
Nombre:	Activar(bool)
Descripción:	En esta función se le asigna el valor tomado de la Clase ControlarFuncionalidades que indica el estado de activo o inactivo de la opción Leer Interfaz.
Nombre:	leerString(string)
Descripción:	Esta función permite leer el texto que se le pasa pro parámetro y hacerle las transformaciones necesarias para que se pueda entender mejor a la hora de escucharlo.

Nombre:	cambiarCaracteres(cadena)
Descripción:	En esta función se cambian aquellos caracteres que no son identificados por la librería JavaScript meSpeak por los que pueda leer correctamente y no pueda afectar que se entienda el contenido del texto.
Nombre:	pronunciarMesParaNumeros()
Descripción:	En esta función se obtiene el mes y el año mostrado por el componente rich:calendar.
Nombre:	botoneslistShuttle(elemento)
Descripción:	En esta función se obtiene un texto por cada botón del componente listShuttle.
Nombre:	cargar()
Descripción:	Función que se encarga de ejecutar todas las funciones necesarias a la hora de cargar la página a mostrar y de asignar por cada componente de la página el evento mouseover el cual posibilita que se lea el texto al cual se le pasa el puntero del mouse por encima.

Tabla 3.2 Descripción del fichero JavaScript: Speak.js

Nombre: ZOOM.js	
Tipo de clase: Fichero JavaScript.	
Variables Generales:	
zoomBooleana	Variable que se encarga de almacenar e indicar a otras funciones si está activa o no la funcionalidad aumentar el tamaño de los elementos de la interfaz.

escala	Variable que almacena el valor de la escala que se va a aplicar sobre el tamaño de los elementos de la interfaz
currFFZoom	Variable que almacena el valor de la escala natural para el navegador Firefox
currIEZoom	Variable que almacena el valor de la escala natural para el navegador Internet Explorer.
Para cada responsabilidad:	
Nombre:	activarZoom(bool,escalas)
Descripción:	En esta función se obtienen por parámetro los valores tomados de la clase ControlarFuncionalidades que indican si esta activa o no la funcionalidad de aumentar el tamaño de los elementos de la interfaz y el valor de la escala a la cual se desea aumentar el tamaño.
Nombre:	aplicarZoom()
Descripción:	Función que aplica el Zoom para aumentar o para disminuir según corresponda.
Nombre:	aumentarZoom(escala)
Descripción:	Esta función es la encargada de aumentar el tamaño de los elementos de la pantalla según el valor de la escala pasada por parámetro.
Nombre:	disminuirZoom()
Descripción:	Esta función es la encargada de regresar el tamaño de los elementos de la pantalla a su tamaño original.
Nombre:	llamarLista()

Descripción:	Función que se encarga de aplicarle el Zoom deseado a los elementos de la lista del componente rich:comboBox

Tabla 3.3 Descripción del fichero JavaScript: Zoom.js

Conclusiones

En el presente capítulo fueron analizados los elementos relacionados con la arquitectura, siendo identificados los objetos del diseño que sustentan el desarrollo de los componentes de implementación. Se especificaron además los atributos y métodos de las clases, los diagramas de clases del diseño para cada caso de uso, donde se representan las distintas relaciones entre cada una de estas y los diagramas de secuencia correspondientes a cada funcionalidad, obteniéndose de esta manera la estructura interna del funcionamiento de la solución propuesta. Con la generación de estos artefactos el equipo de desarrollo de las funcionalidades posee una idea común y una línea base a seguir para llevar a cabo la implementación exitosa de las mismas.

CAPÍTULO 4. Implementación de las funcionalidades para discapacitados visuales en el módulo de Admisión del alas HIS

En este capítulo se introduce el flujo de trabajo de implementación, partiendo de los resultados obtenidos en el diseño. Se muestra el modelo de implementación que está compuesto por el Diagrama de Despliegue y el Diagrama de Componentes. Estos diagramas describen las funcionalidades a construir, también se exponen los aspectos referentes a la seguridad del sistema, las estrategias de codificación, así como la forma en que se tratarán los errores.

4.1. Implementación

El modelo de implementación describe cómo los elementos del diseño se implementan en componentes. Entre los componentes se pueden encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe entre los paquetes y clases del modelo de diseño y los subsistemas y componentes físicos.

El propósito del modelo de implementación es definir la organización del código, planificar las integraciones de sistema necesarias en cada iteración e implementar las clases y subsistemas encontrados durante el Diseño (25).

Se debe proponer una estrategia de codificación que defina los formatos para la asignación de nombres a las variables, estilo de programación y métodos de documentación.

4.1.1. *Diagrama de Componentes*

Un diagrama de componentes modela los aspectos físicos de un aplicación, muestra las organizaciones y dependencias lógicas entre los componentes de un software, sean éstos componentes de código fuente, librerías, binarios o ejecutables.

De los estereotipos estándar que se aplican a los componentes según el lenguaje de modelado UML se emplean: library y file, los cuales representan, una biblioteca de objetos estática o dinámica y un documento que contiene código fuente, respectivamente.

Las relaciones de dependencia se utilizan en los diagramas de componentes para indicar que un componente se refiere a los servicios ofrecidos por otro componente. Los distintos componentes pueden

agruparse en paquetes según un criterio lógico y con vistas a simplificar la implementación. Estos paquetes son estereotipados como <<*subsistemas*>>.

Subsistemas de implementación por capas

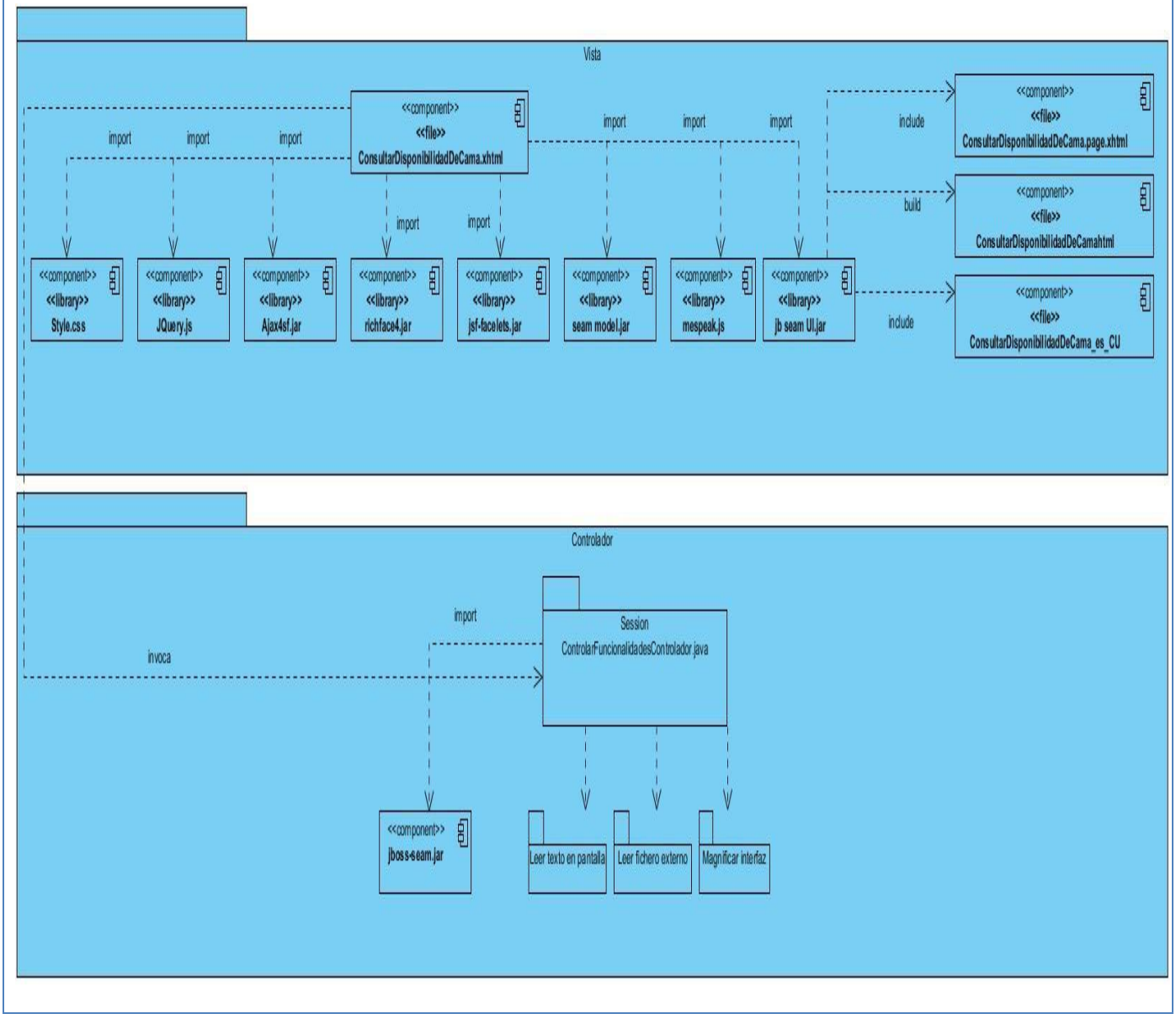

Figura 4.1 Subsistemas de implementación por capas

4.1.2. *Diagrama de Despliegue*

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema, muestra las relaciones físicas entre los componentes de hardware y software, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes de software (procesos y objetos que se ejecutan en ellos).

Para la implantación y la utilización de la aplicación en una institución hospitalaria el usuario debe conectarse a esta mediante una PC cliente utilizando un navegador web. Las peticiones por el protocolo HTTP serán procesadas por el servidor de aplicaciones que enviará la respuesta al cliente.

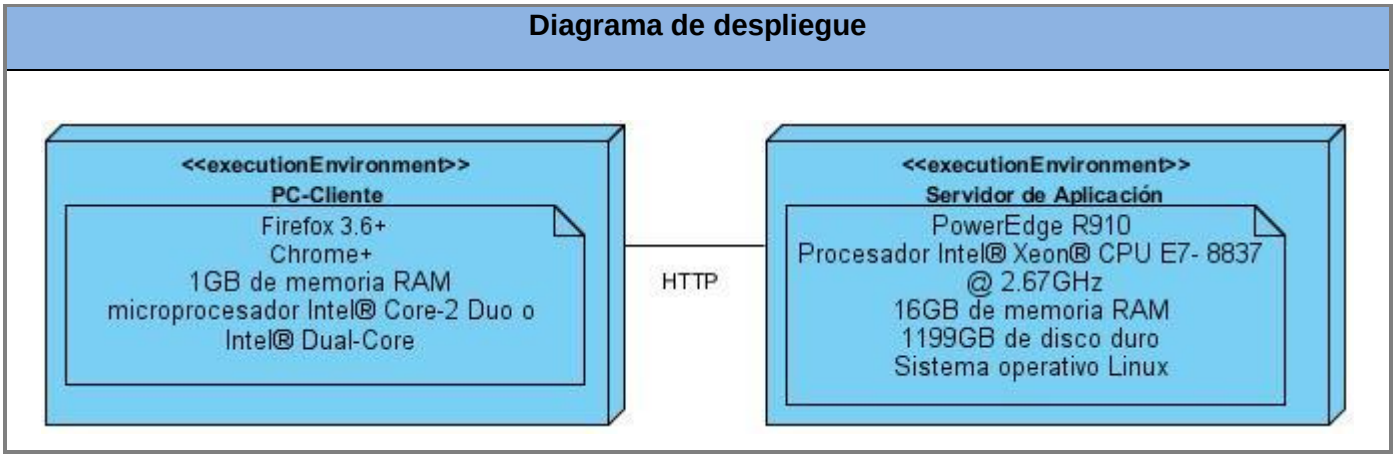

Figura 4.2 Diagrama de despliegue

4.2. Tratamiento de errores

Una excepción es un evento que ocurre durante la ejecución del programa que interrumpe el flujo normal de las sentencias. Las excepciones son el mecanismo recomendado para la propagación de errores que se produzcan durante la ejecución de las aplicaciones. Cuando dicho error ocurre dentro de un método Java, automáticamente se crea un objeto *'Excepcion'* el cual es tratado en el sistema de ejecución. Este objeto contiene información sobre la excepción, incluyendo su tipo y el estado del programa cuando ocurrió el error.

En el componente propuesto, el control de las excepciones se lleva a cabo a toda porción de código, donde pueda surgir alguna situación inesperada. También se controlan los errores que pueden surgir en la

validación de datos provenientes de la interfaz de usuario, puesto que encierran una lógica compleja en cierta medida.

Para el manejo de las excepciones o errores, en las clases controladoras de procesos, se utilizará el bloque **try** para detectar cuando ocurra algún fallo y un bloque **catch** donde se manejarán dichas excepciones, mediante mensajes que se muestran en la interfaz de usuario, por las facilidades que brinda el FacesMessages, componente del framework Seam.

4.3. Seguridad

La seguridad es un tema de gran importancia para cualquier Sistema de Información y toma mayor relevancia cuando se gestiona información médica. Para garantizar la seguridad de las funcionalidades desarrolladas solamente pueden acceder a ellas los usuarios que tienen permiso para trabajar en el módulo de Admisión.

Las funcionalidades dentro del módulo Admisión encargadas de la seguridad que se relacionan con las funcionalidades implementadas son: iniciar y cerrar sesión de trabajo.

Para iniciar la sesión de trabajo un usuario debe acceder al sistema e insertar su nombre de usuario y contraseña. Este se encarga de verifica que los datos introducidos sean válidos y dados los permisos de este usuario tendrá acceso al módulo al que desea entrar. Para acceder a dichas funcionalidades se accede a la opción seleccionar y luego se activa.

4.4. Estrategias de codificación. Estándares y estilos a utilizar

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez.

Usar técnicas de codificación sólidas y realizar buenas prácticas de programación, es de gran importancia para la calidad del software. La aplicación de estándares de codificación además posibilita que el software que se obtiene sea fácil de comprender y de mantener en el tiempo (26).

4.4.1. *Elementos de los estándares de codificación*

Notación Camello

Se emplea para denotar variables y parámetros. Especifica que la palabra de inicio del identificador comienza con minúscula. Si el identificador está compuesto por más de una palabra entonces estás deben comenzar con mayúsculas.

Como consecuencia se especifican algunas restricciones para la nomenclatura, al igual que se explican el uso de los estándares

Identación		
Inicio y fin de bloque	Se recomienda dejar dos espacios en blanco desde la instrucción anterior para el inicio y fin de bloque {}. Lo mismo sucede para el caso de las instrucciones if, else, for, while, do while, switch, foreach.	
Aspectos Generales	El identado debe ser de dos espacios por bloque de código. No se debe usar el tabulador; ya que este puede variar según la PC o la configuración de dicha tecla. Los inicios ({) y cierre (}) de ámbito deber estar alineados debajo de la declaración a la que pertenecen y deben evitarse si hay sólo una instrucción. Nunca colocar { en la línea de un código cualquiera, esto requiere una línea propia.	
Comentarios, separadores, líneas, espacios en blanco y márgenes		
Ubicación de comentarios	Al inicio de cada clase o función y al final de cada bloque de código.	Se recomienda comentar al inicio de la clase o función especificando el objetivo de la misma así como los parámetros que usa (especificar tipos de dato, y objetivo del parámetro) entre otras cosas.
Espacios en blanco	Entre operadores lógicos y aritméticos.	Se recomienda usar espacios en blanco entre estos operadores para lograr una mayor legibilidad en el código. Ejemplo: producto = nomproducto

	Sobre los espacios en blanco.	No se debe usar espacio en blanco: Después del corchete abierto y antes del cerrado. Después del paréntesis abierto y antes del cerrado. Antes de un punto y coma.
Variables y constantes		
Apariencia de variables		El nombre que se le da a las variables debe comenzar con la primera letra en minúscula, en caso de que sea un nombre compuesto se empleará notación Camello. Ejemplo: aspectoSolicitud
Clases y Objetos		
Apariencia de clases y objetos	Primera letra en mayúscula.	Los nombres de las clases deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación Pascal Ejemplo: MiClase (). Para el caso de las instancias se comenzara en minúscula.
Apariencia de atributos	Primera letra en minúscula.	El nombre que se le da a los atributos de las clases debe comenzar con la primera letra en minúscula, en caso de que sea un nombre compuesto se empleará notación Camello.
Apariencia de las funciones	Primera letra en mayúscula.	Para nombrar las funciones se debe tratar de utilizar verbos que denoten la acción que hace la función. Se empleará notación Pascal. Ejemplo: BuscarUnidad (). Si son funciones que obtienen un dato se emplea el prefijo get y si fijan algún valor se emplea el prefijo set.

Declaración de parámetro en funciones	Agrupados por tipos. Poner los string 1 numéricos 2, además, agrupar según valores por defecto.	Los parámetros que se le pasan a las funciones se recomienda sean declarados de forma tal que estén agrupados por el tipo de dato que contienen.

Tabla 4.1 Restricciones del código

Conclusiones

En el presente capítulo se realizó el Diagrama de Despliegue y de Componentes, con la concepción de ambos quedó conformado el modelo de implementación de las funcionalidades. Durante el proceso de codificación se cumplió con los estándares y estilos definidos, lo que permitió obtener un grupo de funcionalidades entendibles para todos los programadores y fácil de mantener en el transcurso del tiempo.

CONCLUSIONES GENERALES

Con el desarrollo de funcionalidades que permitan el uso del módulo de Admisión por usuarios discapacitados visuales se concluye lo siguiente:

1. El análisis de herramientas relacionadas con la usabilidad del módulo de Admisión del sistema alas HIS por usuarios con discapacidad visual evidenció que los mismos no cumplen con todos los requisitos funcionales deseados ni permiten su integración.
2. Las tecnologías, herramientas y la arquitectura definida, permitieron la construcción de funcionalidades para dar respuesta a los requisitos identificados para la solución del problema.
3. La implementación se basó en tecnologías de desarrollo disponibles y que aseguran el cumplimiento de los requerimientos y la construcción de funcionalidades completamente integradas al módulo de Admisión del sistema alas HIS.
4. Con la integración de las nuevas funcionalidades se espera aumentar la calidad funcional del módulo Admisión del sistema alas HIS en las entidades prestadoras de servicios de salud; permitiendo al personal discapacitado visual de esta área contar con nuevas herramientas para ejercer una tarea con mayor facilidad.

RECOMENDACIONES

Con el objetivo de enriquecer la solución propuesta se proponen las siguientes recomendaciones para el trabajo presentado:

- Mejorar el diseño de las interfaces acorde a los patrones de accesibilidad web.
- Mejorar la calidad de las voces generadas en el momento de lectura de los textos.

Bibliografía

- Alvarez, Miguel Angel. desarrolloweb.com.Qué es Java. [En línea] [Citado el: 12 de diciembre de 2012.] http://www.desarrolloweb.com/articulos/497.php.
- Blog de Juan Peláez en Geeks.ms. [En línea] [Citado el: 6 de noviembre de 2012.] http://geeks.ms/blogs/jkpelaez/archive/2009/04/18/arquitectura-basada-en-componentes.aspx.
- Capítulo 5. Cliente-Servidor. [En línea] [Citado el: 25 de enero de 2013.] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/marquez_a_bm/capitulo5.pdf.
- desarrolloweb.com.Que es Javascript. [En línea] [Citado el: 28 de marzo de 2013.] http://www.desarrolloweb.com/articulos/25.php.
- Diagramas UML. Software y Aplicaciones Web. [En línea] [Citado el: 2 de abril de 2013.] http://www.jtentor.com.ar/post/UML-Diagramas.aspx.
- Díaz, Frank Rodríguez. Desarrollo de una aplicación para la migración de datos del Sistema Automatizado de Microcirugía al sistema alas BQO. La Habana : s.n., 2012.
- EcuRed. [En línea] [Citado el: 13 de mayo de 2013.] http://www.ecured.cu/index.php/Arquitectura_Cliente_Servidor.
- EcuRed. Eclipse, entorno de desarrollo integrado. [En línea] [Citado el: 21 de enero de 2013.] http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado.
- ElektraInformatica.jQuery. que es y cómo se usa? [En línea] [Citado el: 28 de marzo de 2013.] http://www.elektrainformatica.com.ar/blog/item/110-jquery-que-es-y-como-se-usa?html.
- Expert Group.javaserverfaces.org. [En línea] [Citado el: 8 de enero de 2012.] http://www.javaserverfaces.org/specification/expert-group.
- Franky, María Consuelo. Java EE 5. [En línea] [Citado el: 2013 de enero de 8.] http://www.acis.org.co/fileadmin/Conferencias/ConfConsueloFranky_Abr19.pdf.

- Free Download Manager. Paradigma visual para UML (Plataforma Java). Visual Paradigm for UML [Java Platform] 6.0. [En línea] [Citado el: 22 de enero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.
- ignside.net. Cascade Style Sheets. [En línea] 2007. [Citado el: 8 de enero de 2013.] http://www.ignside.net/man/css/index.php.
- Itera. Rational Unified Process. [En línea] 2008. [Citado el: enero de 17 de 2013.] http://www.iteraprocess.com/index.php?option=com_content&task=view&id=18&Itemid=42.
- Jaramillo, Wilmer. Software Libre de Venezuela 777, C.A. [En línea] 2006. [Citado el: 15 de enero de 2013.] http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf.
- JBoss Ajax4jsf. [En línea] [Citado el: 8 de enero de 2012.] http://labs.jboss.com/portal/jbossajax4jsf.
- JBoss Community. JBoss Ajax4jsf. Introducción. [En línea] 2007. [Citado el: 15 de diciembre de 2012.] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.
- Larraga, Eduardo Falces. Ingeniería Informática.J2EE Framework capa de presentación. [En línea] 2008. [Citado el: 12 de diciembre de 2012.] http://openaccess.uoc.edu/webapps/o2/bitstream/10609/610/1/00752tfc.pdf.
- Lucifer, Prometeo. Java Runtime Environment – JRE. [En línea] [Citado el: enero de 16 de 2013.] http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre/.
- Maldonado, Daniel M. El CoDiGo K. Arquitectura de programación en 3 capas. [En línea] [Citado el: 14 de diciembre de 2012.] http://www.elcodigok.com.ar/2007/09/arquitectura-de-programacion-en-3-capas/.
- Maure, Amaya Álvarez Lorenzo y Mirelio Mora. Desarrollo de funcionalidades para la especialidad de Psicología del módulo Consulta Externa del sistema alas HIS. La Habana: s.n., 2012.
- Modelo de Implementación: Diagramas de Componentes y Despliegue. [En línea] [Citado el: 2013 de enero de 9.] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

- Mora, Francisco. UML: Lenguaje Unificado de Modelado. [En línea] 2003. [Citado el: 17 de enero de 2013.] http://www.dccia.ua.es/dccia/inf/asignaturas/GPS/archivos/Uml.PDF.
- Ottinger, Joseph. TheServerSide.com. JBoss releases JBoss Tools, Eclipse Plugins including Exadel. [En línea] [Citado el: 21 de enero de 2013.] http://www.theserverside.com/news/thread.tss?thread_id=45933.
- Pérez, Javier Egíluz. Librosweb.es. HTML y XHTML. Capítulo 1: Introducción. [En línea] [Citado el: 2013 de enero de 8.] http://www.librosweb.es/xhtml/capitulo1/html_y_xhtml.html.
- PgDBF.Sitio oficial de PgDBF. [En línea] [Citado el: 2013 de enero de 10.] http://pgdbf.sourceforge.net/.
- Rodríguez, Bertha de los Ángeles Vaillant Preval y Dáyron Aguila. Desarrollo del módulo Banco de Ojos del Sistema Información Hospitalaria alas HIS. La Habana: s.n., 2012.
- VIELKA ESCOBAR DE DONADO . slideshare.NATURALEZA Y CARACTERÍSTICAS DE LOS ALUMNOS/AS CON DISCAPACIDAD VISUAL. [En línea] [Citado el: 2013 de mayo de 8.] http://www.slideshare.net/Irene_Pringle/discapacidad-visualconceptos.
- w3schools.com.AJAX Tutorial. [En línea] [Citado el: 14 de diciembre de 2012.] http://www.w3schools.com/ajax/default.asp.
- Web Application. Plataforma J2EE. JBoss Seam Framework. [En línea] 2008. [Citado el: 8 de enero de 2013.] http://wilmanchamba.wordpress.com/2008/02/20/jboss-seam-framework/.

Referencias Bibliográficas

1. Maure, Amaya Álvarez Lorenzo y Mirelio Mora. *Desarrollo de funcionalidades para la especialidad de Psicología del módulo Consulta Externa del sistema alas HIS.* La Habana : s.n., 2012.

2. VIELKA ESCOBAR DE DONADO . slideshare.NATURALEZA Y CARACTERÍSTICAS DE LOS ALUMNOS/AS CON DISCAPACIDAD VISUAL. [En línea] [Citado el: 2013 de mayo de 8.] http://www.slideshare.net/Irene_Pringle/discapacidad-visualconceptos.

3. Blog de Juan Peláez en Geeks.ms. [En línea] [Citado el: 6 de noviembre de 2012.] http://geeks.ms/blogs/jkpelaez/archive/2009/04/18/arquitectura-basada-en-componentes.aspx.

4. Capítulo 5. Cliente-Servidor. [En línea] [Citado el: 25 de enero de 2013.] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/marquez_a_bm/capitulo5.pdf.

5. Alvarez, Miguel Angel. desarrolloweb.com.Qué es Java. [En línea] [Citado el: 12 de diciembre de 2012.] http://www.desarrolloweb.com/articulos/497.php.

6. Larraga, Eduardo Falces. Ingeniería Informática.J2EE Framework capa de presentación. [En línea] 2008. [Citado el: 12 de diciembre de 2012.] http://openaccess.uoc.edu/webapps/o2/bitstream/10609/610/1/00752tfc.pdf.

7. Expert Group.javaserverfaces.org. [En línea] [Citado el: 8 de enero de 2012.] http://www.javaserverfaces.org/specification/expert-group.

8. JBoss Ajax4jsf. [En línea] [Citado el: 8 de enero de 2012.] http://labs.jboss.com/portal/jbossajax4jsf.

9. w3schools.com.AJAX Tutorial. [En línea] [Citado el: 14 de dicembre de 2012.] http://www.w3schools.com/ajax/default.asp..

10. JBoss Community. JBoss Ajax4jsf. Introducción. [En línea] 2007. [Citado el: 15 de diciembre de 2012.] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.

11. Pérez, Javier Egíluz. Librosweb.es. HTML y XHTML. Capítulo 1: Introducción. [En línea] [Citado el: 2013 de enero de 8.] http://www.librosweb.es/xhtml/capitulo1/html_y_xhtml.html.

12. ignside.net. Cascade Style Sheets. [En línea] 2007. [Citado el: 8 de enero de 2013.] http://www.ignside.net/man/css/index.php..

13. Web Application. Plataforma J2EE. JBoss Seam Framework. [En línea] 2008. [Citado el: 8 de enero de 2013.] http://wilmanchamba.wordpress.com/2008/02/20/jboss-seam-framework/.

14. desarrolloweb.com.Que es Javascript. [En línea] [Citado el: 28 de marzo de 2013.] http://www.desarrolloweb.com/articulos/25.php.

15. ElektraInformatica.jQuery... que es y como se usa? . [En línea] [Citado el: 28 de marzo de 2013.] http://www.elektrainformatica.com.ar/blog/item/110-jquery-que-es-y-como-se-usa?.html.

16. Franky, María Consuelo. Java EE 5. [En línea] [Citado el: 2013 de enero de 8.] http://www.acis.org.co/fileadmin/Conferencias/ConfConsueloFranky_Abr19.pdf.

17. Lucifer, Prometeo. Java Runtime Environment – JRE. [En línea] [Citado el: enero de 16 de 2013.] http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre/.

18. Jaramillo, Wilmer. Software Libre de Venezuela 777, C.A. [En línea] 2006. [Citado el: 15 de enero de 2013.] http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf.

19. Itera. Rational Unified Process. [En línea] 2008. [Citado el: enero de 17 de 2013.] http://www.iteraprocess.com/index.php?option=com_content&task=view&id=18&Itemid=42.

20. Mora, Francisco. UML: Lenguaje Unificado de Modelado. [En línea] 2003. [Citado el: 17 de enero de 2013.] http://www.dccia.ua.es/dccia/inf/asignaturas/GPS/archivos/Uml.PDF.

21. EcuRed. Eclipse, entorno de desarrollo integrado. [En línea] [Citado el: 21 de enero de 2013.] http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado.

22. Ottinger, Joseph. TheServerSide.com. JBoss releases JBoss Tools, Eclipse Plugins including Exadel. [En línea] [Citado el: 21 de enero de 2013.] http://www.theserverside.com/news/thread.tss?thread_id=45933.

23. Free Download Manager. Paradigma visual para UML (Plataforma Java). Visual Paradigm for UML [Java Platform] 6.0. [En línea] [Citado el: 22 de enero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.

24. EcuRed. [En línea] [Citado el: 13 de mayo de 2013.] http://www.ecured.cu/index.php/Arquitectura_Cliente_Servidor.

25. Modelo de Implementación:Diagramas de Componentes y Despliegue. [En línea] [Citado el: 2013 de enero de 9 .] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

26. Rodríguez, Bertha de los Ángeles Vaillant Preval y Dáyron Aguila. *Desarrollo del módulo Banco de Ojos del Sistema Información Hospitalaria alas HIS.* La Habana : s.n., 2012.

ANEXOS

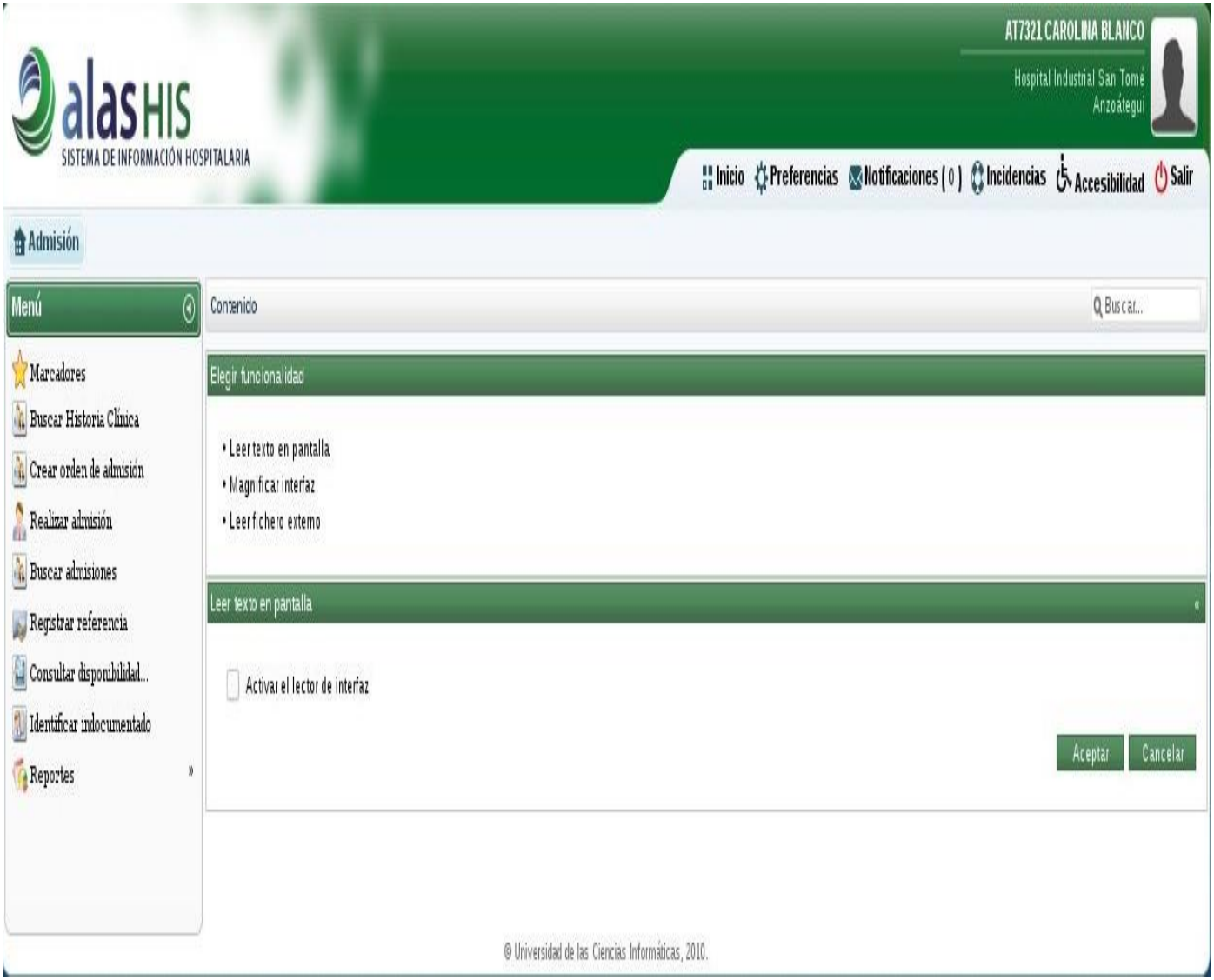

Figura A1 Activar opción Leer texto en pantalla

Figura A2 Activar opción Magnificar interfaz

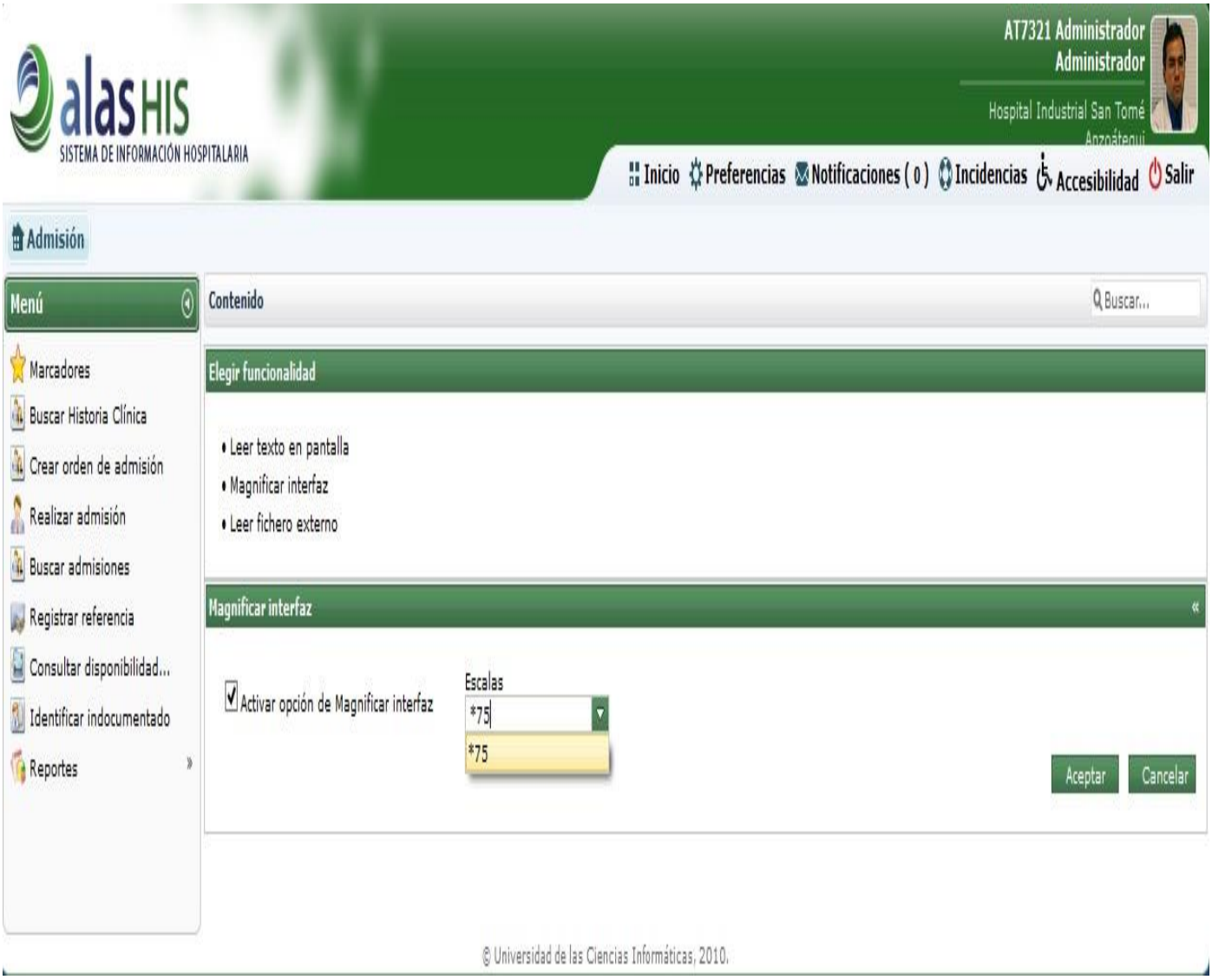

Figura A4 Cambiar Escala de Magnificación

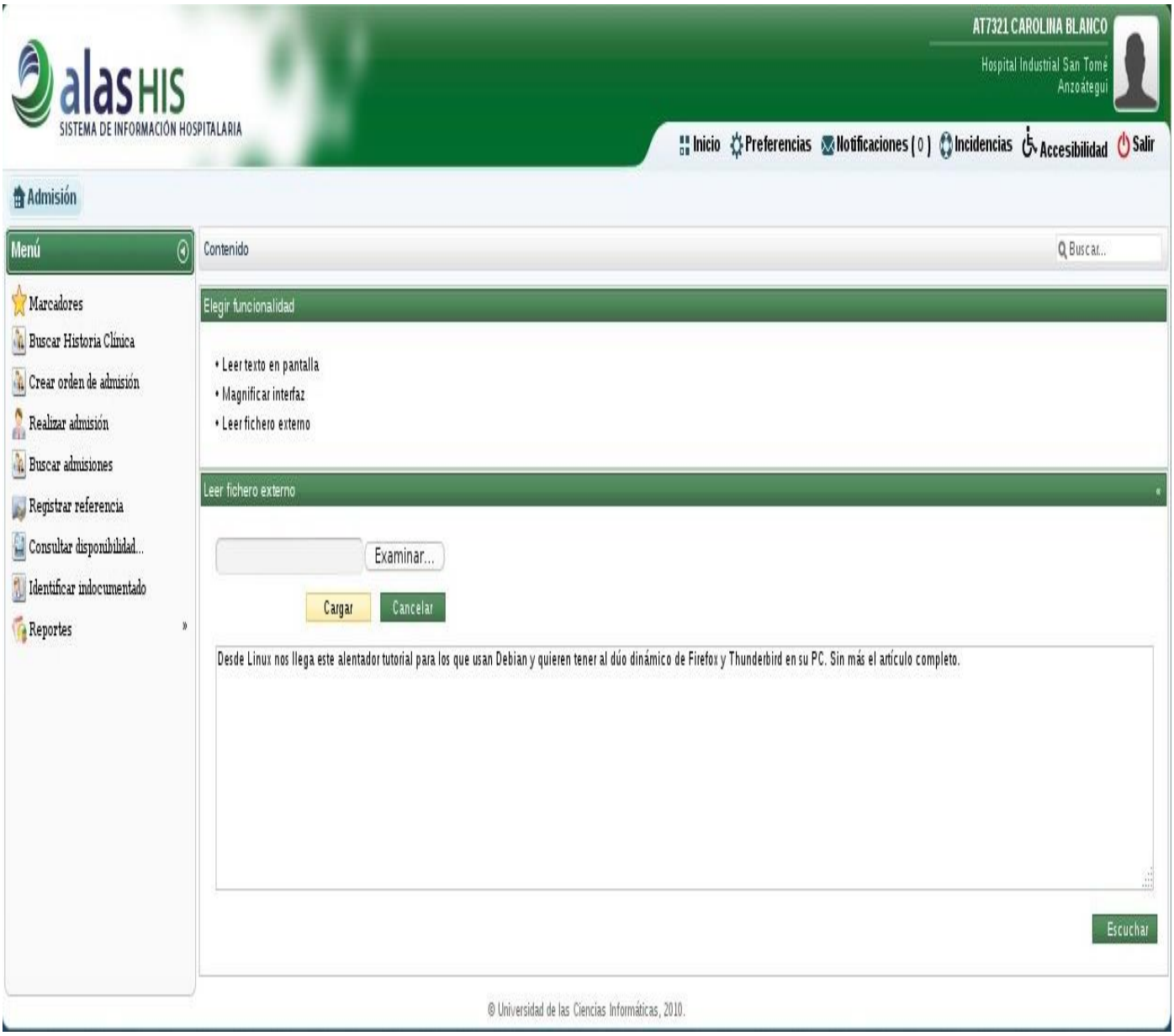

Figura A5 Cargar fichero externo

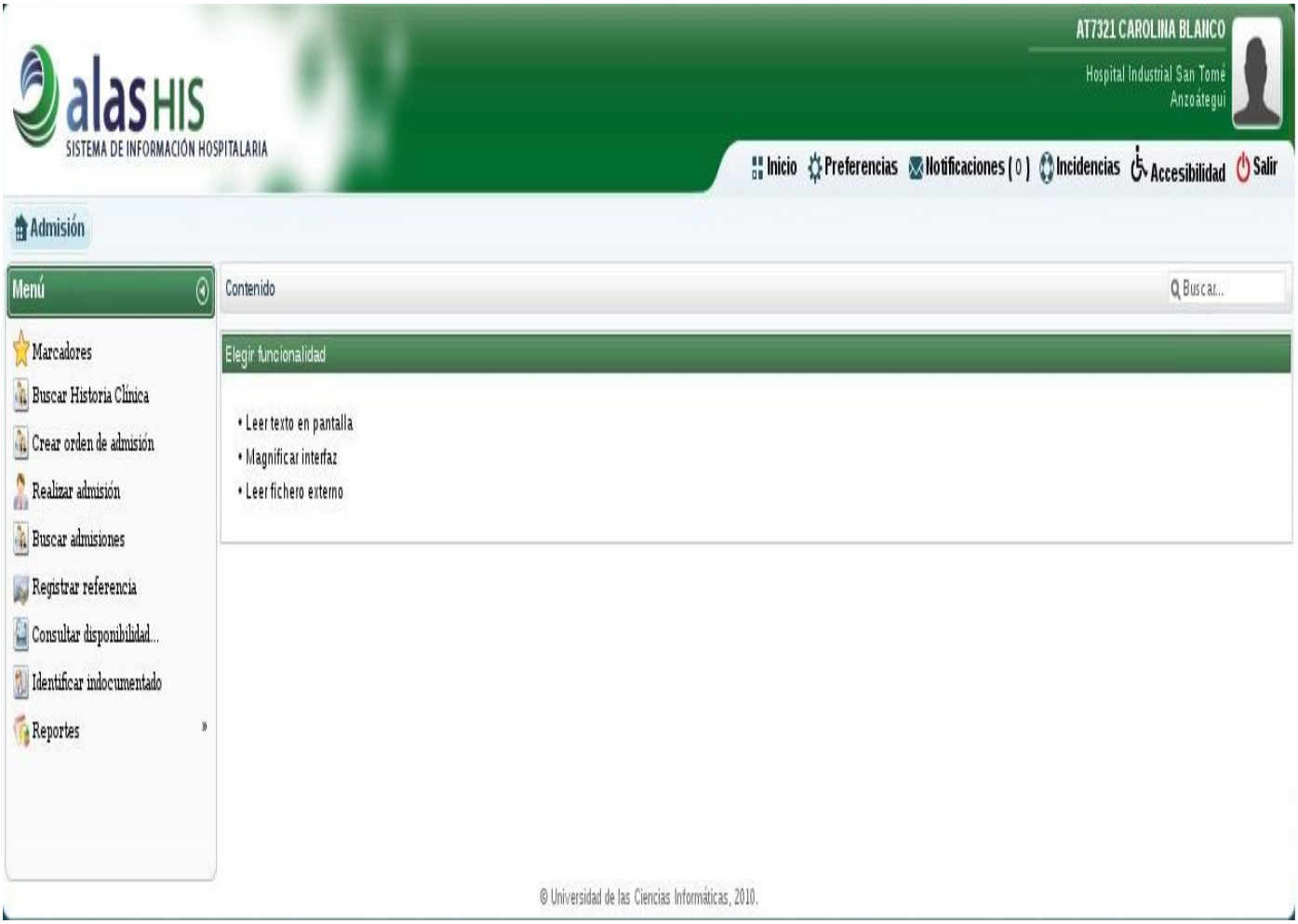

Figura A6 Interfaz de inicio de accesibilidad