UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN

Genci II Gestor de Contenido Modular Para la Coordinación de Investigación De la Facultad de Ciencias de la UCV

Presentado ante la Ilustre
Universidad Central de Venezuela
Por el Bachiller
Br. Carlos Olivares
C.I: 15.487.459
Para optar al título de
Licenciado en Computación
Tutor: Profa. Mercy Ospina

Caracas, 28 de Mayo de 2012

Acta

Quienes suscriben, miembros del Jurado designado por el Consejo de Escuela de Computación, para examinar el Trabajo Especial de Grado presentado por el Bachiller Carlos Olivares Freytez C.I: 15.487.459, titulado:" **Genci-2 Gestor de Contenido Modular para la Coordinación de Investigación de la Facultad de Ciencias de la UCV**", a los fines de optar por el título de Licenciado en Computación, mención Aplicaciones en Tecnologías de Internet, dejen constancia de lo siguiente:

Leído como fue, dicho trabajo por cada uno de los miembros del jurado, se fijó el día 28 de Mayo de 2012 a las 9:00 am en la Sala PBIII de la Escuela de Computación, para que su autor lo defendiera en forma pública, mediante una presentación oral de su contenido, luego de lo cual respondieron las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobarlo con una nota de _____ puntos.

En fé de lo cual se levanta la presente Acta, en Caracas los 28 días del mes de Mayo del año dos mil doce.

_____

Profa. Mercy Ospina (Tutora)

_____ _____

Prof. Pio Arias (Jurado) Prof. Robinson Rivas (Jurado)

Agradecimientos y Dedicatorias

El primer agradecimiento es a Dios por haber creado la vida y permitirle al hombre equivocarse una y otra vez pero siempre dándole fuerzas para volverlo a intentar.

El segundo agradecimiento es a la Ciencia quien cuestiona todo lo que Dios creó y alimenta el ego humano, dándole herramientas para perfeccionar su búsqueda a la demostración de la verdad sobre las cosas.

Agradezco a las circunstancias de mi vida la cual colocó innumerables obstáculos, y me retó a tomar decisiones que cambiaron una y otra vez el rumbo de mi destino.

Agradezco a mi madre Carmen Alicia, quien con su incansable paciencia soportó mis más injustas respuestas, me apoyó en todo momento sin importar las decisiones que hubiese tomado, siempre colocando primero mis necesidades antes que la suya.

Agradezco a mi novia Mariana Simón quien me retó desde los inicios de mi carrera a superarme a mi mismo, apoyándome de día y de noche, ayudándome cuando más lo necesitaba y dándome fuerzas para continuar.

Agradezco a mi hermoso país Venezuela, en donde fui rodeado de un ambiente y una sociedad con la cual forjé cada aspecto de mi personalidad, lugares para inspirarme, lugares para aprender y para temer, pero por sobre todas las cosas me mostró la necesidad que existe para ayudarlo a mejorar.

Agradezco a los profesores y estudiantes de la Universidad Central de Venezuela, lugar en donde aprendí sobre diversos aspectos académicos, sociales, políticos, científicos y artísticos; Lugar con profesionales apasionados a la educación y a la mejora continua del estudiante, gracias profesores.

Para finalizar dedico mi pregrado y el éxito que hasta ahora he conseguido a todas aquellas personas que me han apoyado de alguna manera, quienes me han enseñado o me han permitido enseñar. También lo dedico a aquellos que me dieron la espalda y me empujaron para caer ya que gracias a ellos aprendí a volverme a levantar y a hacerlo mejor para superarlos en su mismo juego.

¡Gracias a la vida y a las oportunidades!

Resumen

La Coordinación de Investigación de la Facultad de Ciencias de la Universidad Central de Venezuela, brindado soporte a las diferentes dependencias que forman parte de ella, ha estado ofreciendo servicios informativos vía Web a través de una aplicación creada en el año 2006, denominada "GENCI" con el cual la ha creado y administrando las páginas Web de cada una de las dependencias.

El uso de esta aplicación durante varios años ha permitido conocer sus limitaciones funcionales y administrativas, entre ellas se han observado la inexistencia de funciones para la publicación de eventos y noticias de interés a la comunidad, así como para la creación de nuevos niveles de dependencias. A partir de este punto surge la necesidad crear una segunda versión de GENCI llamada GENCI-2 la cual permite la creación usuarios con diferentes roles para la administración de la aplicación a distintos niveles de seguridad, publicaciones dinámicas de noticias y artículos de cada una de las dependencias con contenido multimedia, calendarios de eventos, elaboración de menús personalizados para cada dependencia, con enlaces entre las distintas páginas Web.

Para permitir flexibilidad en el crecimiento y mantenimiento de la aplicación, GENCI-2 crea un sitio Web con una estructura jerárquica que refleja la estructura organizacional de la Coordinación de Investigación de la Facultad de Ciencias, la cual permitirá el crecimiento tanto en profundidad como en amplitud, agregando nuevas páginas Web para cada dependencia y subdependencia, las cuales pueden ser creadas usando temas predefinidos que permiten realizar un diseño homogéneo, rápido y fácil por personas con poco conocimiento técnico, y que luego serán administradas por usuarios con diferentes niveles de seguridad y privilegios acordes a la dependencia asociada.

Palabras Clave: Gestor de Páginas Web, Reingeniería de Software, Coordinación de Investigación, Aplicación Web, Base de Datos, Proceso Ágil Unificado, Ruby on Rails.

Índice

Índice de imágenes

Índice de Tablas

INTRODUCCIÓN

La Coordinación de Investigación de la Facultad de Ciencias de la Universidad Central de Venezuela, como grupo de investigadores que contribuyen con la solución de problemas en las aéreas prioritarias del Estado, tiene la necesidad de promover los trabajos realizados, la información institucional, proyectos presentes pasados y futuros, así como también divulgar información de interés científico y académico para toda la comunidad.

Por tal motivo en el año 2006 se desarrolla como trabajo especial de grado un generador de páginas Web denominado "GENCI", el cual fue desarrollado por el Lic. Alexander Rivero y el Lic. Clemente Borges. Ellos realizaron una aplicación Web la cual cubría las principales necesidades del centro de investigación en aquel momento, este generador de páginas Web cuenta con la posibilidad de crear páginas Web independientes y administrar el contenido de cada una de ellas por medio de usuarios predefinidos asignados para cada una de ellas.

GENCI se ha venido utilizando durante cuatro años y ha brindado la posibilidad a las diferentes dependencias que conforman la Coordinación de Investigación de tener cada una su página Web.

Debido al éxito que ha tenido la aplicación GENCI, surge la necesidad de crear una segunda versión de este Generador de páginas Web que amplié las funcionalidades. La idea se ha concebido como un Manejador de Contenidos en donde se cubren las necesidades principales que tiene la aplicación GENCI y además agregar otras funcionalidades que permitirán a la Coordinación de Investigación la gestión de la información de una forma más usable.

Esta aplicación Web que se denomina GENCI -2 se encuentra desarrollada en un lenguaje de programación de software libre llamado Ruby sobre el Framework Rails, el cual es la plataforma tecnológica en la que se están desarrollando las nuevas aplicaciones dentro de la Facultad de Ciencias de la Universidad Central de Venezuela. Fue desarrollado bajo este lenguaje de programación buscando obtener una mayor compatibilidad, unificación y crecimiento para los proyectos a futuros que serán propuestos en un futuro.

Este documento de Trabajo Especial de Grado tiene una estructura de tres capítulos, los cuales son:

Capítulo I – Planteamiento del Problema: En este capítulo se muestra cual es la función de la Coordinación de Investigación, cuales son las necesidades que existen actualmente a

solucionar en este Trabajo Especial de Grado, además de exponer algunos antecedentes de la aplicación Web.

Capítulo II – Marco Conceptual: Se presenta el marco teórico y las tecnologías utilizadas durante el desarrollo de este Trabajo Especial de Grado.

Capítulo III – Marco Aplicativo: Se describe el Marco Aplicativo en donde se expone la implementación del método de desarrollo de software Proceso Unificado Ágil y los diferentes entregables durante sus fases de Inicio, Elaboración, Construcción y Transición.

Para culminar se expone la conclusión y las referencias bibliográficas manejadas.

CAPÍTULO I – Planteamiento del Problema

A continuación se define el ambiente del problema comprendido en el presente Trabajo Especial de Grado, exponiendo cual es la función de la Coordinación de Investigación y cuáles son los problemas existentes actualmente.

1.1. Centros de Investigación

Un centro de Investigación en su forma más simple, es una agrupación de investigadores, profesores o profesionales, para realizar investigación en una determinada disciplina o área del conocimiento, contribuyendo con la solución de problemas en las aéreas prioritarias del Estado, en lo que concierne a las disciplinas que agrupa. Sus actividades principales son la investigación científica y/o tecnológica, incluyendo: Captación y entrenamiento de capital humano, transferencia de tecnología, difusión, divulgación científica y gestión, seguimiento y evaluación de procesos de ciencia y tecnología. Los Centros de Investigación permiten la difusión y adaptación del conocimiento científico y tecnológico necesario para aportar soluciones, contribuyendo a dinamizar el aparato productivo venezolano, con la finalidad de satisfacer los requerimientos de la población y por ende, favorecer el desarrollo del país y el bienestar de la sociedad venezolana.(Borges, H y Rivero, A. 2006)

Un centro de investigación puede ser independiente o estar adscrito a una institución universitaria o no universitaria, posee una organización formal, un cierto grado de autonomía administrativa y financiera, y puede o no tener personería jurídica propia.

1.1.1. Objetivos de un Centro de Investigación

A continuación se nombran algunos de los objetivos principales que persiguen los Centros de Investigación de un área en particular:

- Contribuir con la integración y al fortalecimiento de la investigación de un área en particular, promoviendo la cooperación entre los diferentes grupos de investigación del centro y/o entre otras unidades de investigación del país.
- Promover y realizar investigación básica y aplicada en cada una de las áreas de la investigación del Centro.
- Contribuir con la formación y permanente actualización en las distintas áreas de investigación del centro, mediante la organización y realización de cursos de especialización, de postgrado o pasantías, seminarios, asesoría, coloquios y otros eventos.

- Cooperar y participar activamente en programas y eventos nacionales e internacionales de investigación en las distintas áreas de investigación del Centro.
- El Centro de investigación puede servir de soporte al Pregrado y Postgrado del ente universitario o de ciencia y tecnología al que este adscrito, a través de la docencia y la dirección de Trabajos Especiales de Grado y de Tesis en el postgrado.

1.1.2. Actividades de un Centro de Investigación

Un Centro de Investigación tiene una estructura organizada para establecer, desarrollar, coordinar y dirigir actividades de investigación de un grupo de investigadores.

Una línea de investigación define los fines, objetivos y proposición de un investigador o de un conjunto de investigadores, bien sean personales, institucionales, de un colectivo o grupo. Dentro del ámbito académico, normalmente las finalidades, los objetivos, las políticas, las estrategias y las formas de evaluar el trabajo. Debe estar acompañado por el plan de acción o proyectos de investigación, que incluyen las actividades que se van a desarrollar y responder a las estrategias que éste se propone. A su vez, el plan de acción puede ir acompañado de un cronograma que indique el tiempo durante el cual se ejecutara el trabajo, los responsables del mismo y las formas de evaluación.

Un Centro de Investigación debe definir los siguientes aspectos: Fines del Centro de Investigación; misión del Centro de Investigación; objetivos generales y específicos del Centro de Investigación; políticas del Centro de Investigación; estructura que tendrá el Centro de acuerdo con el entorno social; el área de conocimientos y las líneas de investigación. Además, debe elaborar estrategias con las cuales enfrentara la ejecución del esquema con el fin de desarrollas, fomentar y difundir la investigación y de esta manera, desarrollar las políticas de investigación que sean trazadas para el Centro y construir el modelo con el cual se evaluara, tanto el desarrollo de las investigaciones propiamente dichas, como las demás actividades que en su hacer de fomento y difusión realice el Centro. (Borges, H y Rivero, A. 2006)

1.1.3. Coordinación de Investigación de la Facultad de Ciencias de la Universidad Central de Venezuela y sus dependencias.

La Coordinación de Investigación fue creada por la aprobación de la Facultad de Ciencias, en Diciembre de 1991. Inicialmente existía una Comisión de Investigación adscrita a la Coordinación de Postgrado formada por los Coordinadores de Investigación de las Escuelas, Centros e Institutos de la Facultad de Ciencias, la cual había sido creada por el Consejo de Facultad en Enero de 1986.

La creación de esta Coordinación constituyó un hecho muy significativo, ya que se le dio

a la "Investigación" la jerarquía que le corresponde dentro de una comunidad Científica que ofrece un gran y muy calificado aporte a la Ciencia y la Tecnología del País.

La Coordinación de Investigación de la Facultad de Ciencias está constituida por un Coordinador y el Consejo de Investigación, y tiene como propósitos: proteger y difundir las actividades de investigación.

El Consejo está constituido por el Coordinador de Investigación quien lo preside, los representantes de cada uno de los Consejos Técnicos de los Institutos, los representantes de las Comisiones de Investigación de cada una de las Escuelas y representantes de la Facultad ante el C.D.C.H.(Pio, A. 2006)

La Coordinación de Investigación de la Facultad de Ciencias tiene a su vez dependencias que se especializan en el área específica de cada escuela o institución. Entre las dependencias que maneja la están:

- Escuela de Biología.
- Escuela de Computación.
- Escuela de Física.
- Escuela de Matemáticas.
- Escuela de Química.
- Instituto de Ciencias de la Tierra.
- Instituto de Zoología Tropical.
- Instituto de Ciencias de Tecnología de Alimentos.
- Instituto de Biología Experimental.

1.2. Aplicación anterior GENCI.

La Coordinación de Investigación de la Facultad de Ciencias de la Universidad Central de Venezuela, haciendo uso de las tecnologías de información se ha estado apoyando en una plataforma Web que hasta ahora les ha brindado una serie de valores agregado a la comunidad, en este sentido, en el año 2006 se decide crear una aplicación Web llamada GENCI, la cual se puede observar en línea a través de la dirección www.coordinv.ciens.ucv.ve/investigacion y en la figura 1 del presente documento, esta aplicación le permitió a la coordinación, crear y administrar a cada una de sus dependencias un espacio para que puedan manejar el contenido independientemente unos de otros.

A pesar de que la aplicación Web GENCI actual ha resuelto varios problemas es muy limitado en sus funcionalidades, por ello surge el planteamiento del problema del presente Trabajo Especial de Grado.

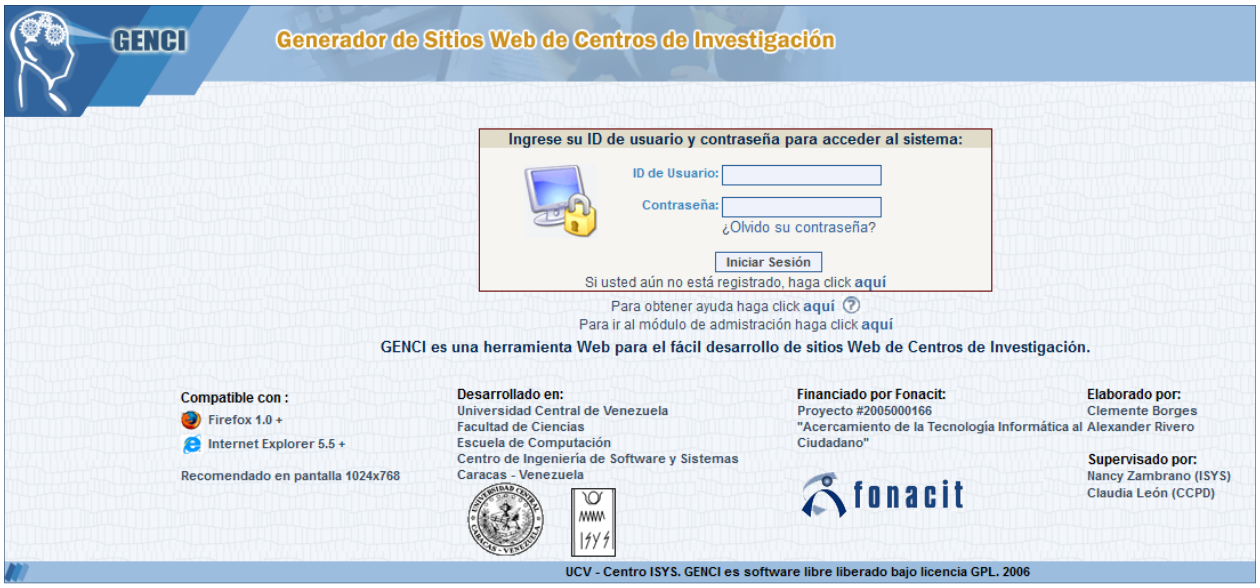


Figura 1 - Página de Inicio Aplicación Web GENCI

1.3. Planteamiento del Problema.

A medida que la tecnología va evolucionando y más posibilidades se le van sumando a las plataformas tecnologías, fueron surgiendo nuevos requerimientos para los usuarios y administradores de cada una de las dependencias, como también para la Coordinación de Investigación.

De esta forma tomando como base el éxito y uso que tuvo la aplicación Web GENCI, la Coordinación de Investigación busca desarrollar una segunda versión de este generador Web para agregarle las funcionalidades que han estado necesitando durante este tiempo, y con este proyecto agregar nuevas posibilidades eliminando las limitaciones más claras de la primera versión de GENCI, para así poder tener una nueva aplicación con la que la coordinación de Investigación y la comunidad de la Facultad de Ciencias pueda crecer y usar en diferentes áreas.

Las principales características que se quieren solucionar con esta nueva versión de GENCI son:

- El contenido que se maneja en GENCI es estático, por lo cual no permite agregar nuevas publicaciones dinámicamente.
- Las dependencias de la Coordinación no pueden contener sub-dependencias.
- La diagramación y manejo de estilos es poco usable y atractiva.
- Las posibilidades de escalabilidad son muy reducidas.
- Solo existe un usuario administrador que no permite el fácil manejo de contenido.

Habiendo descrito la aplicación anterior GENCI y las necesidades que presenta la Coordinación de Investigación de la Facultad de Ciencias, se procede a explicar algunos conceptos que son la base para la aplicación que se realizó en este Trabajo Especial de Grado.

1.4. Gestor de Contenido

Un Gestor de Contenido es una aplicación que da soporte para la creación y administración de contenidos, principalmente en formato Web por parte de los participantes.

Consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio. El sistema permite manejar de manera independiente el contenido y el diseño. Así, es posible manejar el contenido y darle en cualquier momento un diseño distinto al sitio sin tener que darle formato la estructura de nuevo, además de permitir la fácil y controlada publicación en el sitio a varios editores de manera controlada. Un ejemplo es el de editores que con jerarquía de usuarios en donde un tipo de usuario carga el contenido al sistema y otro de nivel superior que permite que estos contenidos sean visibles a todo el público.

El gestor de contenidos es una aplicación informática usada para crear, editar, gestionar y publicar contenido digital en diversos formatos. El gestor de contenidos genera páginas dinámicas interactuando con el servidor para generar la página web bajo petición del usuario, con el formato predefinido y el contenido extraído de la base de datos del servidor.

Esto permite gestionar, bajo un formato estandarizado, la información del servidor, reduciendo el tamaño de las páginas para descarga y reduciendo el coste de gestión del portal con respecto a una página estática, en la que cada cambio de diseño debe ser realizado en todas las páginas, de la misma forma que cada vez que se agrega contenido tiene que hacerse una nueva página HTML y subirla al servidor. (Juárez, R. 2005)

1.4.1. Objetivos de un Gestor de Contenido

A continuación nombro algunos de los objetivos principales que persiguen los Centro de Investigación de un área en particular:

- Permitir que sin conocimientos de programación ni construcción, cualquier usuario pueda indexar contenido en el portal.
- Permitir la gestión dinámica de usuarios y permisos, la colaboración de varios usuarios en el mismo trabajo, la interacción mediante herramientas de comunicación.
- Facilitar el acceso a la publicación de contenidos a un rango mayor de usuarios, funciones específicas para cada rol en particular.

- Canalizar la actualización, respaldos y reestructuración del sistema a través de la implementación correcta y del uso de una base de datos estructurada en el servidor.

Conociendo el concepto base de la aplicación desarrollada, se expone algunas aplicaciones Web de universidades re renombre internacional que se utilizaron como referencia para desarrollar GENCI-2.

1.5. Antecedentes

A continuación se exponen tres casos de estudio de departamentos de investigación de prestigiosas universidades de renombre internacional, como son Stanford, Berkeley y el MIT, en los siguientes puntos se mostrara una apreciación de las diferentes maneras como estas instituciones muestran el contenido relacionado con investigación, de tal manera de poder determinar alguna tendencia que nos ayude con la propuesta de solución al problema para cubrir las necesidades de la Coordinación de Investigación de la Facultad de Ciencias de la Universidad Central de Venezuela.

1.5.1. Universidad de Stanford

La universidad de Stanford en su Sitio Web nos muestra un diseño limpio, con una diagramación parecida a las usadas en sitios de noticias y revistas en formato web, en donde claramente muestra un enlace a la sección de investigación, ubicada en la barra de navegación como se muestra en la figura 2.

La sección de de investigación muestra una diagramación en donde proponen servir como directorio a las diferentes áreas en donde la universidad es efectiva, de tal forma que a su vez cada una de estas categorías enlazan a cada una de las facultades que componen la universidad.

Cada facultad es independiente a la hora de diagramar y publicar la información en sus sitios Web, y así la manera de mostrar sus respectivos sitios de investigación.

Tomando tres facultades para hacer la comparación (Facultad de Ingeniería, Facultad de Ciencias de la Tierra y Facultad de Humanidades y Ciencias) se ve claramente una tendencia en las tres en donde cada una de ella muestra información muy general sobre el departamento de investigación, mas no muestra proyectos en ejecución, profesores involucrados o futuros eventos.

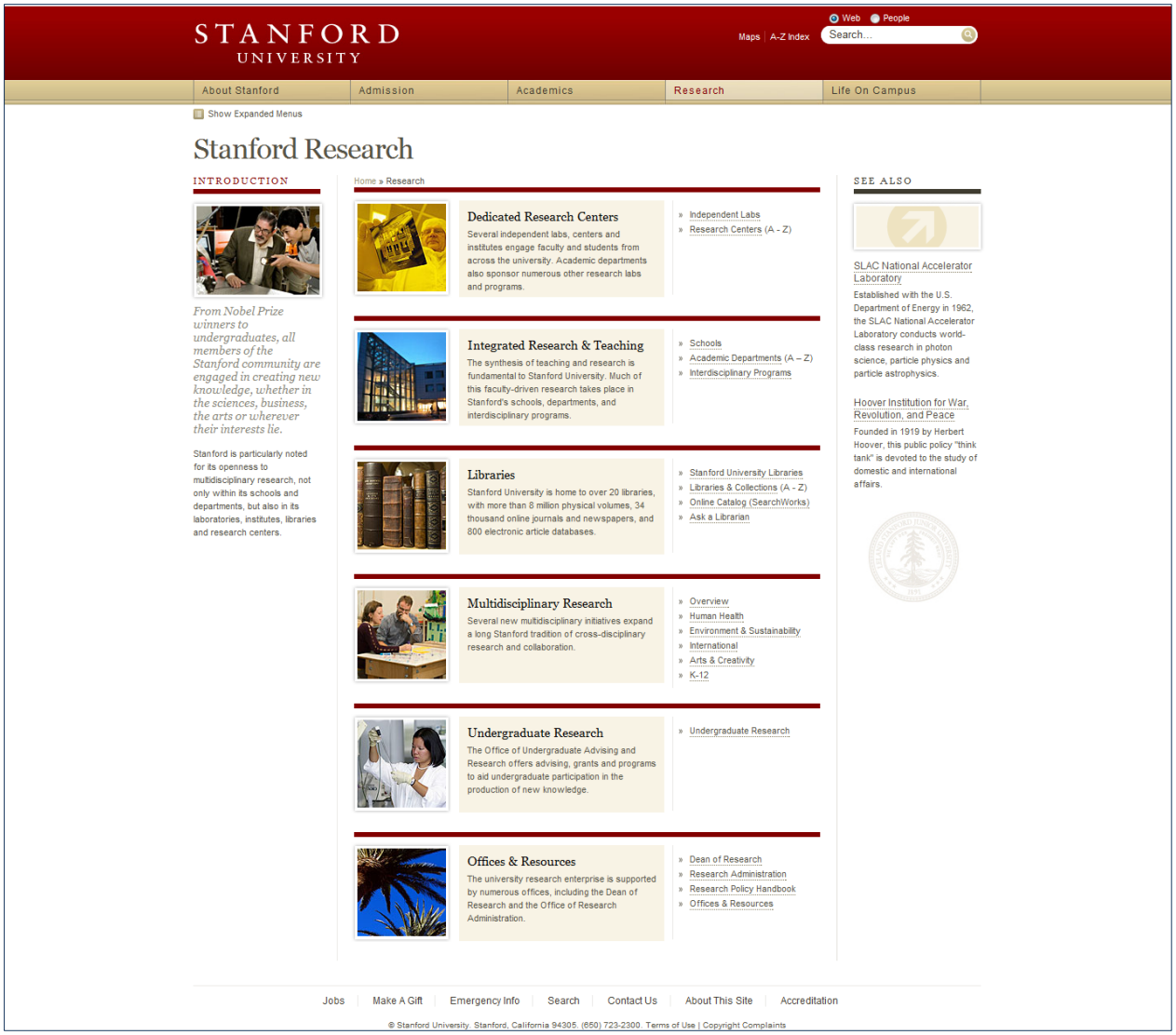


Figura 2 - Sitio Web Universidad de Stanford

1.5.2. Universidad de Berkeley

La universidad de Berkeley muestra un sitio web sumamente ágil, y ordenado en donde en su página de inicio tenemos la oportunidad de seleccionar el departamento de investigación, el cual es un sitio independiente que cuenta con información bastante útil y completa relacionada con los trabajos de investigación que hacen en la universidad. Existen secciones de noticias, facultades, industrias, fundaciones, unidades de investigación, entre otras.

Dentro de la opción de "Unidades de Investigación" se muestra una lista con las unidades que posee la universidad que tienen algún proyecto de investigación vigente, pero al igual que en la universidad de Stanford, esto funciona como directorio ya que al seleccionar alguna de las

unidades el enlace directo es hacia la facultad responsable por gestionar dicha investigación como se muestra en la figura 3.

Internamente en cada facultad existe la posibilidad de ir hasta la comisión de investigación de cada una de ellas, y a su vez cada una de ellas tiene una lista de investigaciones vigentes, las cuales son sitios web independientes, que no guardan ningún lineamiento relacionado con la pagina de la facultad o incluso la universidad.

Por este motivo hay muchos que poseen información completa relacionada a la investigación, docentes y alumnos involucrados, patrocinantes y de más, pero está de parte de cada departamento o unidad de investigación, publicar el contenido como mejor les parezca.

Figura 3 - Sitio Web de la Universidad de Berkeley

1.5.3. Instituto de Tecnología de Massachusetts (M.I.T)

Al igual que en los dos casos anteriores, el Instituto de Tecnología de Massachusetts cuenta con un fácil acceso a la sección de investigación, la cual muestra una lista de categorías relacionadas con las áreas en donde la universidad es efectiva en materia de investigación como se ve en la figura 4.

De esta manera trabaja al igual que los otros dos casos, como un directorio en donde al seleccionar alguna de las opciones que sugiere en la sección de investigación, el mismo enlaza al

sitio web perteneciente a la facultad responsable de seguir los temas de la investigación. Queda de parte de cada facultad que compone a la universidad, el hacer el énfasis necesario en la comisión de investigación.

Cada facultad muestra sus diferentes ramas de investigación y al navegar hasta los sitios web de estos, nos damos cuenta que cada uno de ellos es totalmente independiente de los sitios de las facultades y no guarda ninguna relación por los lineamientos tanto de las facultades como de la misma universidad.

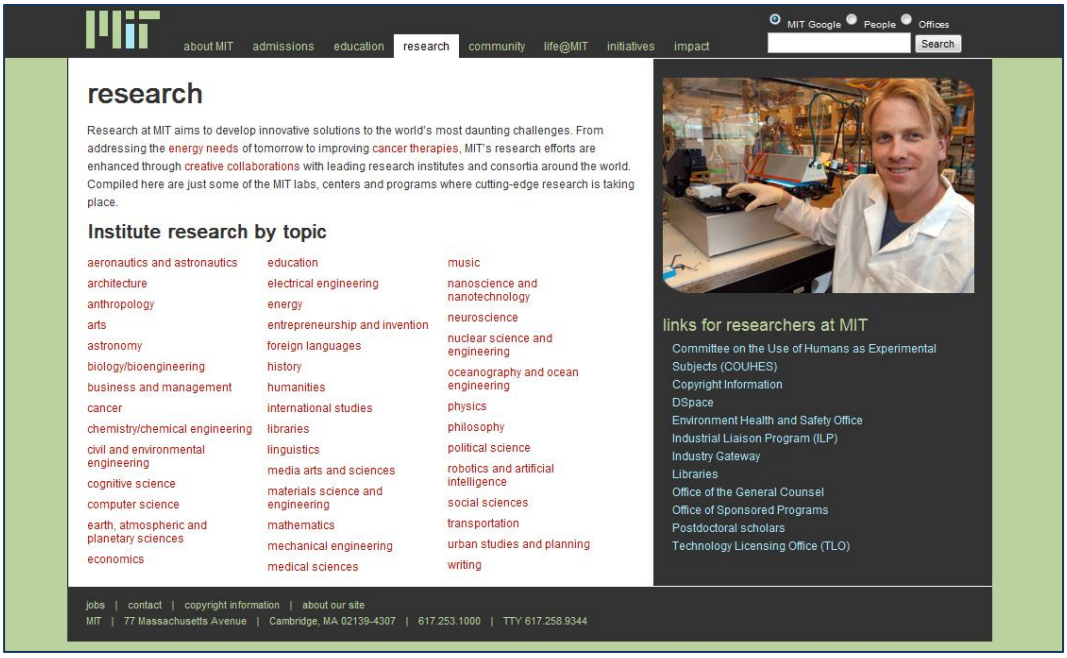


Figura 4 - Sitio Web del Instituto de Tecnología de Massachusetts

CAPÍTULO II – Marco Conceptual

El propósito de este capítulo es presentar las bases conceptuales de las tecnologías utilizadas durante el desarrollo de la aplicación del Gestor de Contenido Modular GENCI-2.

2.1. Aplicaciones Web

Las aplicaciones Web son todas aquellas aplicaciones que al ser desarrolladas y alojadas en un servidor en específico, pueden ser accedidas por parte de los usuarios a través de los diversos navegadores Web, para así utilizar sus funcionalidades. La ventaja de poder acceder a este tipo de aplicaciones vía Web, es que permite que un gran número de usuarios puedan hacer uso de estas, sin importar el momento o la ubicación en donde se encuentren. Además, estas aplicaciones al ser ejecutadas en navegadores de Internet, son desarrolladas en leguajes que puedan ser soportados por estos, entre algunos están ASP, PHP, HTML, entre otros.

Este tipo de aplicaciones gracias a las ventajas que ofrecen las comunicaciones y el flujo de la información a través de Internet, han hecho que se conviertan en una opción muy utilizada para el desarrollo de software. El bajo costo necesario para la implementación de este tipo de aplicaciones y la gran interactividad que se puede lograr con el usuario, son otros aspectos que han llevado a grandes compañías a nivel mundial a inclinarse hacia los software basados en la Web como vía de comunicación con sus clientes (Aplicaciones Web, 2010).

Debido al gran volumen de información y datos que viajan a través de la Web por medio de este tipo de aplicaciones, y sobre todo cuando manejan datos confidenciales de usuarios en distintos sitios de comercio electrónicos o bancarios, es necesario garantizar que estos viajen de forma segura a través de la red, para que no sean vulnerables a los ataques hechos por agentes maliciosos. Por esta razón, la seguridad es uno de los factores que más se deben tener en cuenta al momento del desarrollo de estas aplicaciones.

2.1.1. Ventajas de las Aplicaciones Web

Las Aplicaciones Web son de gran utilidad, ya que pueden ser accedidas desde cualquier parte del mundo, así como esta existen otras ventajas las cuales se mencionan a continuación. (Solcre, 2008)

- **Compatibilidad multiplataforma**

Las aplicaciones Web, tienen un camino mucho más sencillo para la compatibilidad multiplataforma que las aplicaciones de software descargables. Varias tecnologías incluyendo JavaScript, Flash, ASP y Ajax permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales.

- **Actualización**

Las aplicaciones basadas en Web están siempre actualizadas con el último lanzamiento, sin requerir que el usuario tome acciones Pro-activas, y sin necesitar llamar la atención del usuario o interferir con sus hábitos de trabajo, con la esperanza de que va a iniciar nuevas descargas y procedimientos de instalación.

- **Acceso Remoto**

Las aplicaciones basadas en Web no necesitan ser descargadas, instaladas y configuradas. Se accede vía online y están listas para trabajar sin importar cuál es su configuración o su hardware.

- **Eficiencia en consumo de recursos**

Las aplicaciones basadas en Web tienen muchas más demandas de memoria RAM por parte del usuario final que de los programas instalados localmente. Al residir y ejecutarse en los servidores del proveedor, esas aplicaciones basadas en Web usa en muchos casos la memoria de las computadoras, dejando más espacio para ejecutar múltiples aplicaciones al mismo tiempo sin incurrir en frustrantes deterioros del rendimiento.

- **Mayor Seguridad**

Las aplicaciones basadas en Web deberían ser más robustas, y menos propensas a crear problemas técnicos debido al software o conflictos de hardware. Estas aplicaciones permiten mantener actualizada la versión y es más fácil corregir algún virus una vez descubierto. Esta es la razón, por la cual, las aplicaciones basadas en Web deberían tener mucho menos virus que el software standalone.

- **Usuarios concurrentes**

Las aplicaciones basadas en Web pueden realmente ser utilizada por múltiples usuarios al mismo tiempo. No hay necesidad de compartir pantallas, o enviar mensajes instantáneos, cuando múltiples usuarios pueden ver e incluso editar al mismo tiempo un documento de manera

conjunta. Las compañías de conferencia Web y colaboración online, están involucradas en la evolución de las aplicaciones de colaboración en línea.

2.1.2. Desventajas de las Aplicaciones Web

- Los múltiples accesos realizados en forma simultánea degradan el rendimiento de las aplicaciones debido a la sobrecarga del servidor donde están alojadas.
- Como el navegador es la herramienta utilizada para ejecutar las aplicaciones, éstas dependen de la configuración de los mismos, para que puedan trabajar de forma óptima, ya sea en cuanto a los Cookies, Java Script o archivos Flash.
- Si el servidor donde esta almacenada la información y la aplicación Web no tienen los niveles de seguridad necesarios para reguardar los datos, estos pueden ser vulnerables a los ataques de agentes maliciosos.
- Para que las aplicaciones y la navegación en las mismas puedan ser ejecutadas de forma óptima es necesario poseer una buena velocidad de conexión a Internet.

2.2. Arquitectura Cliente/Servidor

Esta arquitectura consta principalmente de dos partes muy bien identificadas que se ejecutan de forma independiente, la primera conformada por un conjunto de clientes y la segunda por el servidor. El cliente es el encargado de iniciar la comunicación a través de la red con el servidor, emitiendo solicitudes de recursos o peticiones de servicios, los cuales pueden ser la consulta de una información en particular, un archivo, entre otros, y están ubicados en diferentes estaciones de trabajo. Por el otro lado, se encuentra el servidor, el cual se encarga de atender a las solicitudes realizadas por los clientes para responder con el recurso o la información solicitada.

En esta arquitectura, el servidor contiene la parte que debe ser compartida por varios usuarios (cliente), debido a esto, el servidor suele estar ubicado en equipos con grandes capacidades de procesamiento y recursos para atender a las peticiones realizadas por los clientes de forma eficiente. Algunos tipos específicos de servidores incluyen los servidores Web, los servidores de archivo, los servidores del correo, etc. Se puede observar en la figura 5 la representación de un cliente servidor. (Consejo Superior de Administración Electrónica, n.d).

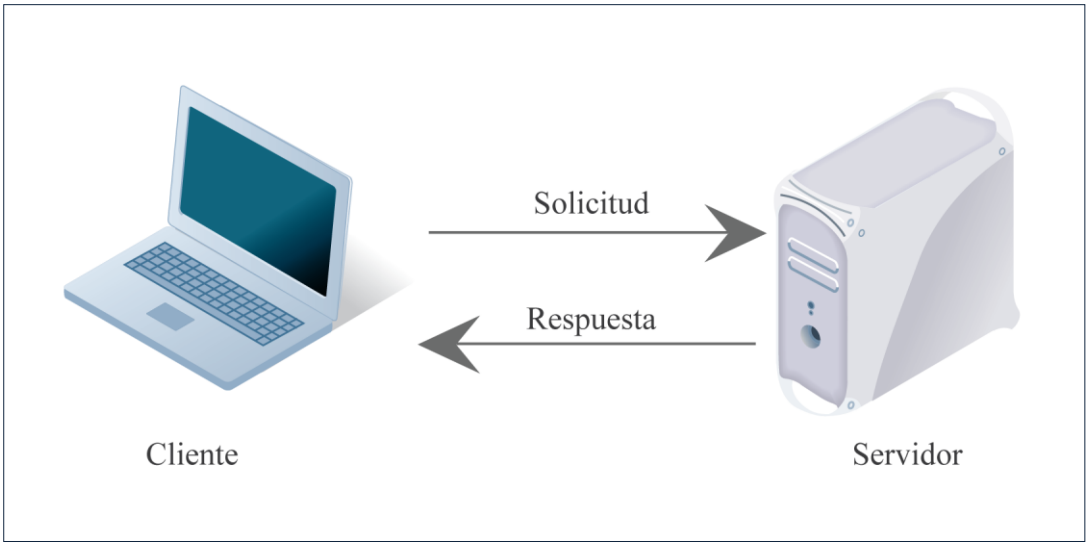


Figura 5 - Arquitectura Cliente / Servidor

2.3. Tecnologías del Cliente

Las tecnologías del lado del cliente son aquellas tienen las siguientes características:

- Inician solicitudes o peticiones, tienen por tanto un papel activo en la comunicación (dispositivo maestro o amo).
- Esperan y reciben las respuestas del servidor. Por lo general, puede conectarse a varios servidores a la vez.
- Normalmente interactúa directamente con los usuarios finales, mediante una interfaz gráfica de usuario.
- El fallo de un cliente no afecta el rendimiento del sistema.
- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.

Entre las tecnologías del lado del cliente utilizadas para el desarrollo de este Trabajo Especial de Grado se encuentran las expuestas a continuación.

2.3.1. Lenguaje de Marcas de Hipertexto HTML

Es el lenguaje más sencillo y utilizado para el desarrollo de páginas Web. Los documentos escritos en este formato son entendidos por los diversos navegadores, los cuales convierten esta información en las conocidas páginas Web que son mostradas a los usuarios finales cuando hacen unos de Internet.

Los documentos escritos en HTML, le indican al navegador donde colocar cada imagen, texto, video o enlace dentro de las páginas Web y la forma que estos elementos tendrán al momento de desplegarse. Para la referenciar a estos elementos y toda la información dentro de los documentos HTML, este hace uso de las llamadas etiquetas, las cuales son definidas mediante los símbolos "< >". En el medio de estos dos símbolos se define el elemento al que se refiere la etiqueta, por ejemplo, <p> para un párrafo de texto, <img> para una imagen o <a> para un enlace. Para la mayoría de estas de estas etiquetas existe su correspondiente etiqueta de cierre, lo cual le indica al navegador que a partir de ese punto dicha etiqueta no tendrá más efecto. Esta etiquetas de cierre son denotadas de la forma "< / >". (W3C, 2010)

En general HTML no es más que un conjunto de etiquetas lógicamente organizadas para dar forma a las páginas Web dependiendo a las necesidades que se requieran.

El HTML ha ido evolucionando en versiones, y hoy en día contamos con la versión 5, la cual brinda una serie de innovadoras características que nombraré a continuación.

2.3.2. JavaScript

Es un lenguaje interpretado muy parecido a los programas escritos en Java y trabajado de forma similar. Los JavaScript están presentes muy comúnmente dentro de las páginas Web e integrados a los distintos navegadores. Estos son pequeños programas que realizan acciones muy prácticas y útiles para el desarrollo de distintas aplicaciones en el ámbito de la Web. Al igual que HTML los JavaScript son ejecutados por el navegador y pueden estar incrustados dentro del documento HTML, o bien, separados en archivos independientes con la extensión js.

El uso que comúnmente se le da a estos pequeños programas se basa fundamentalmente en la creación de efectos especiales en las páginas y la definición de interactividades con el usuario. Validaciones a formularios, mensajes de errores o alertas, requerimiento de ciertas condiciones, entre otras, son algunas acciones muy comunes que son útiles al momento de el desarrollo de grandes aplicaciones con gran concurrencia de usuarios, en las que se necesitan enviar datos a un servidor para que este responda con la petición requerida, ya que el procesamiento previo de los datos evita que al llegar a un servidor en especifico, este desperdicie tiempo vital de procesamiento con datos inválidos que al final no generaran una respuesta satisfactoria (Mozilla. n.d).

El uso adecuado de esta herramienta puede brindarle una mejor experiencia y desempeño al usuario, así como también, grandes prestaciones en reducciones de costos de procesamiento en grandes aplicaciones.

La tecnología de JavaScript ayuda mucho en el comportamiento, manejo de eventos y lógica del lado del cliente, pero a esto le hace falta un lenguaje que maneje las interfaces de una menara más vistosa y eficiente, y es ahí cuando el CSS entra en acción.

2.3.3. Hojas en estilo cascada CSS

Es un mecanismo simple de descripción de estilos que se utiliza para mostrar un documento en la pantalla, cómo se va a imprimir, o incluso cómo va a ser presentada la información presente en ese documento. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.

CSS se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los Estilos definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores Web controlar el estilo y el formato de múltiples páginas Web al mismo tiempo. Cualquier cambio en el estilo de un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento, por esta razón proviene su definición de cascada.

CSS funciona a base de reglas, es decir, declaraciones sobre el estilo de uno o más elementos. Las hojas de estilo están compuestas por una o más de esas reglas aplicadas a un documento HTML o XML. Estas reglas tiene dos partes: un selector que indica el elemento y una declaración en la que se define el valor de uno de sus atributos (Guia Breve CSS, 2008).

Al igual que el HTML, el CSS también ha evolucionado en versiones y esta ya se encuentra en la versión 3, la cual ofrece una mayor gama de atributos para configurar los aspectos gráficos y de diagramación de una manera más eficiente que con HTML o imágenes, y este lenguaje lo explicaré a continuación.

2.3.4. Hojas en estilo cascada CSS3

CSS 3 contiene varias mejoras en cuanto a interfaz gráfica, posicionamiento y tamaño de los objetos, usando condiciones de alineación para cada uno. El objetivo es que sea más sencillo posicionar los controles dentro de la página y que cuenten con otras características como desplazamiento.

Las nuevas capacidades de CSS 3 permitirían entre otras cosas usar imágenes para los bordes, redondear y/o agregar sombras. Posicionamiento de elementos en pantalla: se podrá controlar de mejor manera los objetos y su dirección (horizontal o vertical).

Además, se quiere incluir el módulo de paginación para crear pies de página, referencias cruzadas y construir cabeceras para títulos de secciones. También se desea introducir una nueva propiedad para dividir secciones en columnas.

Nuevas funciones sobre todo encaminadas a brindar un mejor soporte a múltiples lenguajes. Se pretende volver a incluir el @font-face para utilizar fuentes externas. (Lemun, Juan. 2008)

2.4. Características del cliente

- Es quien inicia solicitudes o peticiones, tienen por tanto un papel activo en la comunicación (dispositivo maestro o amo).
- Espera y recibe las respuestas del servidor.
- Por lo general, puede conectarse a varios servidores a la vez.
- Normalmente interactúa directamente con los usuarios finales, mediante una interfaz gráfica de usuario.
- Por lo general, no comparte sus recursos con otros clientes.
- El fallo de un cliente no afecta el rendimiento del sistema.
- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.

2.5. Características del servidor

- Al iniciarse esperan a que lleguen las solicitudes de los clientes, desempeñan entonces un papel pasivo en la comunicación (dispositivo esclavo).
- Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.
- Por lo general, aceptan conexiones desde un gran número de clientes (en ciertos casos, el número máximo de peticiones puede estar limitado).
- No es frecuente que interactúen directamente con los usuarios finales.
- Gestiona y comparte sus recursos con los clientes que sirve.
- El servidor presenta a todos sus clientes una interfaz única y bien definida.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente.

2.6. Patrón Modelo Vista Controlador

El Modelo Vista Controlador (MVC), es un patrón de diseño muy utilizado para el desarrollo de aplicaciones Web, el cual tiene como idea fundamental, separar de dichas

aplicaciones sus principales componentes, entre ellos están el sistema de gestión de base de datos, las interfaces de usuario y las tareas de manejar los eventos producidos por los mismos. Para tratar cada uno de estos puntos, MVC provee de partes bien definidas, que permiten manejar de forma independiente estos tipos de componentes de las aplicaciones. Dichas partes son las siguientes (Camejo, Rafael, 2010):

2.6.1. Modelo

Realiza toda la gestión de datos dentro de las aplicaciones, representa su estructura y encapsula su información al igual que sus funcionalidades, manteniendo la lógica necesaria en cuanto al manejo de operaciones con los mismos, y las modificaciones a las que son sometidos durante las ejecuciones de los programas y acciones realizadas por los usuarios. En el modelo, es donde se realizan todas las operaciones en las que se ve involucrada la base de datos, bien sea por consultas, inserciones, actualizaciones o eliminación de datos.

2.6.2. Vista

Muestra la interfaz al usuario, en donde este puede generar acciones en las que se vean involucrados los datos, y se encarga de la presentación visual de los mismos representados por el modelo. Se pueden generar diferentes vistas de dicho modelo, pero cada una de estas, debe tener asociado un componente en el controlador.

2.6.3. Controlador

Atiende los eventos o acciones generadas por los usuarios desde la interfaz, estos pueden ser cambios en los datos manejados por el modelo, o por modificaciones en las interfaces mostradas por la vista. Estos eventos son traducidos en solicitudes de servicios.

Una vez separados los distintos componentes de las aplicaciones, en las partes antes descritas, problemas como cambios en la interfaz, que impliquen modificaciones trabajosas en otros componentes del programa son evitados, ya que al trabajar con MVC, estos son tratados de manera independiente, y una modificación en cualquiera de los componentes no representa ningún tipo de cambio en el resto de los módulos al que pertenezca.
Gracias a esto MVC trae consigo grandes beneficios entre los cuales están:

- El tener desacoplado los componentes del modelo, la vista y el controlador mejora de forma significativa la reusabilidad de la aplicación.
- Se facilitan las tareas de mantenimiento, escalabilidad y depuración del sistema.

- La independencia entre los módulos, permite la programación de cualquier componente dentro de los mismos de forma paralela, lo cual disminuye tiempo en la fase de desarrollo haciendo que grandes aplicaciones puedan ser terminadas con gran rapidez.
- La conexión entre el Modelo y sus Vistas es dinámica, esta se produce en tiempo de ejecución y no en tiempo de compilación.

2.7. Funcionamiento del patrón MVC

Aunque se pueden encontrar diferentes implementaciones de MVC, el flujo que sigue el control generalmente es el siguiente (Camejo, Rafael, 2010):

1. El usuario, interactúa con la interfaz de usuario de alguna forma (por ejemplo, pulsa un botón, enlace, etc.).
2. El controlador, recibe (por parte de los objetos de la interfaz-vista), la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos.
3. El controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario, (por ejemplo, el controlador actualiza el carro de la compra del usuario). Los controladores complejos, están a menudo estructurados, usando un patrón de comando que encapsula las acciones y simplifica su extensión.
4. El controlador delega a los objetos de la vista, la tarea de desplegar la interfaz de usuario. La vista, obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se refleja los cambios en el modelo (por ejemplo, produce un listado del contenido del carro de la compra). El modelo no debe tener conocimiento directo sobre la vista. Sin embargo, el patrón de observador puede ser utilizado para proveer una referencia indirecta entre el modelo y la vista, permitiendo al modelo notificar a los interesados de cualquier cambio. Un objeto vista, puede registrarse con el modelo y esperar a los cambios, pero aun así, el modelo en sí mismo sigue sin saber nada de la vista. El controlador no pasa objetos de dominio (el modelo) a la vista, aunque puede dar la orden a la vista para que se actualice. Nota: En algunas implementaciones, la vista no tiene acceso directo al modelo, dejando que el controlador envíe los datos del modelo a la vista.
5. La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

Las diferentes base de datos son vitales para el funcionamiento adecuado de las aplicaciones Web, ya que en ellas se almacena todo el contenido que la aplicación debe gestionar y mostrar de manera controlada, como también administrar los datos de control del sistema, por lo que se procederá a explicar las bases teóricas a continuación.

2.8. Base de Datos

Una base de datos (BD), es un conjunto de grandes volúmenes de datos almacenados de forma ordenada y relacionados entre sí, por medio de asociaciones específicas, de forma tal que los usuarios puedan actualizar y recuperar dicha información. Estos datos, desde el punto de vista computacional, pueden estar representados por tipos de archivo como: imágenes, videos, o documentos, pero también por una simple cadena de caracteres como usualmente se utilizan. Sin embargo, cualquiera que sea el tipo de dato que haya sido almacenado, este es perdurable en el tiempo y con la adecuada manipulación de los mismos se pueden acceder a ellos de una manera rápida, para así generar algún tipo de información o realizar una consulta al momento que se necesite. (Silberschatz, A y Korth, H y Sudarshan, S. 2006)

2.9. Modelos de Base de datos

Son una colección de herramientas conceptuales para describir los datos, sus relaciones, su semántica y las restricciones de consistencia. Los modelos de datos ofrecen un modo de describir el diseño de las bases de datos en los niveles físico, lógico y de visitas. (Silberschatz, A y Korth, H y Sudarshan, S. 2006)

Entre algunos de estos modelos de BD se encuentra las Bases de Datos Relacionales:

2.9.1. Base de datos relacionales

El modelo relacional usa una colección de tablas para representar tanto los datos como sus relaciones. Cada tabla tiene varias columnas, y cada columna tiene un nombre único. El modelo relacional es un ejemplo de un modelo basado en registros. Los modelos basados en registros se denominan así porque la base de datos se estructura en registros de formato fijo de varios tipos cada tabla contiene registros de un tipo dado. Cada tipo de registro define un número fijo de campos, o atributos. Las columnas de la tabla corresponden con los atributos del tipo de registro. El modelo de datos relacional es el modelo de datos más ampliamente usado, y una gran mayoría de sistemas de bases de datos actuales se basan en el modelo relacional. Estas usan un conjunto de tablas para representar tanto los datos como las relaciones entre ellos. (Silberschatz, A y Korth, H y Sudarshan, S. 2006)

2.10. Sistemas Manejador de Base de Datos (SMBD)

Es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos (DDL: Data

Definition Language), de un lenguaje de manipulación de datos (DML: Data Manipulation Language) y de un lenguaje de consulta (SQL: Structured Query Language). Un SMBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos. (Simón, M y Mata, J. 2010)

2.10.1. Objetivos de los SMBD

- Permitir el acceso concurrente: El SMBD, debe actualizar y mantener la consistencia de los datos de forma cuando haya un acceso concurrente sobre los mismos por parte de varios usuarios.
- Recuperar información: El SMBD debe proveer un mecanismo capaz de recuperar la BD en el caso de que ocurra alguna falla. El SMBD, debe dejar en cualquier circunstancia un estado consistente en la BD.
- Seguridad: Debe garantizar que solo los usuarios autorizados, tengan acceso a la data o parte de ella, así como también que tipo de privilegios tendrán sobre los datos mismos.
- Integridad y redundancia de datos: Se refiere a la calidad de los datos, el SMBD debe proporcionar medios necesarios para garantizar que los datos cumplan con ciertas reglas para que estos mantengan su consistencia y validez.
- Abstracción de la información: la BD contienen un gran volumen de información, los SMBD ocultan a los usuarios los detalles acerca del almacenamiento físico de los datos. (Simón, M y Mata, J.2010)

2.11. Ventajas de los SMBD

- Proveen facilidades para la manipulación de grandes volúmenes de datos. Simplifican la programación de chequeos de consistencia.
- Manejando las políticas de respaldo adecuadas, garantizan que los cambios de la base serán siempre consistentes sin importar si hay errores en el disco, o hay muchos usuarios accediendo simultáneamente a los mismos datos, o se ejecutaron programas que no terminaron su trabajo correctamente, etc.
- Permiten realizar modificaciones en la organización de los datos, con un impacto mínimo en el código de los programas.
- Permiten implementar un manejo centralizado de la seguridad de la información (acceso a usuarios autorizados), protección de información, de modificaciones, inclusiones, consulta.
- Las facilidades anteriores bajan drásticamente los tiempos de desarrollo y aumentan la calidad del sistema desarrollado si son bien explotados por los desarrolladores.
- Usualmente, proveen interfaces y lenguajes de consulta que simplifican la recuperación de los datos. (Simón, M y Mata, J. 2010)

2.12. Desventajas de los SMBD

- Recursos: Por lo general este tipo de software son de gran tamaño, ya que la complejidad del mismo y la gran cantidad de funciones que estos realizan, consumen un espacio considerable de memoria para ejecutarse y espacio en disco para su correcta instalación. Estos aspectos, son de vital importancia para que el software tenga un buen desempeño y no se degrade el rendimiento del sistema.
- Conocimiento previo: Para aprovechar al máximo las prestaciones que brindan los SMBD, es necesario, tener un conocimiento previo de los mismos, de lo contrario, se estará desaprovechando un gran número de funcionalidades que puedan ser de gran utilidad para las tareas a realizar, provocado así, gastos en costos y tiempos de operaciones.
- Costos: Algunos SMBD pueden llegar a ser bastantes costosos, así como también el hardware adicional necesario para el mismo. (Simón, Mariana y Mata, Jesús. 2010)

Como sistema manejador de Base de Datos, se propone hacer uso de MySQL, el cual se explicara a continuación.

2.13. Sistema de Manejador de Base de Datos MySQL

Debido a los altos costos de otras plataformas y de los beneficios que brinda el Sistema de Gestión de Base de Datos MySQL, además que es utilizado en las aplicaciones de la Facultad de Ciencias, se explica en detalle este manejador.

Es el SGBD relacional de código abierto más utilizado en la actualidad. El software MySQL utiliza una licencia pública general (GPL), para definir lo que se puede y no se puede hacer con el software en diferentes situaciones.

MySQL Server se desarrolló originalmente para tratar grandes bases de datos mucho más rápido que soluciones existentes, y ha sido utilizado con éxito en entornos de producción de alto rendimiento durante varios años. MySQL ofrece hoy en día una gran cantidad de funciones. Su conectividad, velocidad, seguridad y licencia pública hacen de MySQL altamente apropiado para acceder bases de datos en Internet y el SGBD más conveniente para el desarrollo de aplicaciones en las que se buscan soluciones rápidas y de bajos costos.

Por otra parte MySQL se basa en la tecnología cliente/servidor que consiste en un, servidor SQL multihilo que trabaja con diferentes programas, clientes, herramientas administrativas y un amplio abanico de interfaces de programación para permitir a muchos

lenguajes acceder a las bases de datos MySQL sin ningún problema, ya que para cada uno de estos existe una API especifica. (SGBD MySQL)

2.13.1. Características de MySQL

- Funciona en diferentes plataformas, entre las cuales se pueden mencionar: Windows, Solaris, GNU/Linux, Mac OS, SunOS, etc.
- Trabaja con multihilos, pueden usarse fácilmente varios procesadores si están disponibles.
- APIs disponibles para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, y Tcl.
- Diversos tipos de datos: enteros con/sin signo de 1, 2, 3, 4, y 8 bytes de longitud, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM, y tipos espaciales OpenGIS.
- Soporte completo para operadores y funciones en las cláusulas de consultas SELECT y WHERE.
- Soporte completo para las cláusulas SQL GROUP BY y ORDER BY. Soporte de funciones de agrupación (COUNT(), COUNT(DISTINCT ...), AVG(), STD(), SUM(), MAX(), MIN(), y GROUP_CONCAT()).
- Puede mezclar tablas de distintas bases de datos en la misma consulta.
- Un sistema de privilegios y contraseñas que es muy flexible y seguro. Las contraseñas son encriptadas cuando se conecta con un servidor.
- Soporta grandes cantidades de tablas y registros.
- Los clientes se pueden conectar con el servidor MySQL usando sockets TCP/IP en cualquier plataforma.
- Joins muy rápidos usando un multi-join de un paso optimizado.
- El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor.
- Soporte para alias en tablas y columnas como lo requiere el estándar SQL.
- Fácil de Instalar y Configurar.

Para gestionar los datos desde el lado del servidor se propone hacer uso del lenguaje de programación Ruby con el marco Rails, el cual se expone a continuación.

2.14. Ruby On Rails

Ruby on Rails es un entorno de desarrollo Web de código abierto que está optimizado para la satisfacción de los programadores y de la productividad. Esta realizado en lenguaje Ruby que permite escribir un buen código favoreciéndola convención antes que la configuración.

2.14.1. Ruby

Michael Fitzgerald define que, "Ruby es un lenguaje de programación orientado a objetos creado por primera vez en Japón en el año 1995 por Yukihiro Matz Matsumoto. Ruby fue adquirido y aceptado mundialmente como un lenguaje fácil de aprender, poderoso e interpretado. Ruby posee una licencia de software libre." (Fitzgerald, 2007)

"Su creador Yukihiro Matz Matsumoto mezcló partes de sus lenguajes favoritos (Perl, S y Eiffel, A y Lisp) para formar un nuevo lenguaje que incorporara tanto la programación funcional como la programación imperativa." (Ruby, n.d)

2.14.2. Rails

Bruce Tate, Curt Hibbs definen a Rails como el "framework mas productivo de desarrollo Web de todos los tiempos" (Tate, B.2006). Su base de datos se fundamenta en la estructura Modelo Vista Controlador.

Rails fue creado por David Heinemeier Hansson y publicado en el Julio del 2004 con una licencia de código abierto.

2.15. Patrón Modelo Vista Controlador en Rails

Dave Thomas y David Heinemeier mencionan que, "En 1979, Trygve Reenskaug muestra un nuevo patrón para el desarrollo interactivo de aplicaciones. En este diseño, las aplicaciones se dividen en tres tipos de componentes: modelos, vistas y controladores." (Thomas, D.2005)

2.15.1. Modelo

El Modelo consiste en las clases persistentes (que representan a las tablas de la base de datos). En Ruby on Rails, las clases del Modelo extienden la clase ActiveRecord.

Esta clase implementa el patrón de diseño ActiveRecord que permite corresponder objetos a relaciones, es decir, provee una interfaz entre las tablas de una base de datos relacional, y el código de la aplicación escrita en ruby que manipula los registros de la base de datos.

Rails añade automáticamente atributos y métodos a la clase basándose en las columnas de la tabla, por lo que automáticamente se corresponden las clases con las tablas y los atributos con las columnas.

Relaciones ActiveRecord

Trabajar con relaciones es una de las tareas más importantes de los marcos de trabajo de persistencia. La idea es manejar las relaciones con un alto rendimiento de cara al usuario final y gran sencillez de cara al desarrollador. ActiveRecord aprovecha las convenciones de Ruby para simplificar el acceso a datos relacionados, definiendo las siguientes relaciones (Ospina, M y Suarez, S. 2008):

- Belongs_to: Significa "pertenece a" y representa la relación "de muchos a uno" en el extremo de "muchos"
- Has_many: Representa el lado "uno" de la misma relación anterior, es decir es la otra cara de la relación belongs_to.
- Has_one: Representa la relación "uno a uno", se puede usar en conjunto con "belongs_to" o con "has_one"
- Has_and_belongs_to_many: Representa la relación de "muchos a muchos" donde hay una tabla adicional con las claves primarias de las tablas relacionadas, pero no hay más datos.

2.15.2. Vista

La vista es responsable de generar la interfaz de usuario, normalmente basada en la data del modelo.

Existe varias formas de gestionar las vistas, el método que aplica Rails es usar Ruby Embebido (archivos.html.erb), que son partes de código HTML mezclado con código en Ruby.

La gema encargada de controlar las vistas en rails es el ActiveView el cual es el responsable del despliegue de visualización de contenido en pantalla (Ospina, M y Suarez, S. 2008).

2.15.3. Controlador

El Controlador corresponde a la interacción del usuario e invocan a la lógica de la aplicación, que a su vez manipula los datos de las clases del Modelo y muestra los resultados usando las Vistas.

El ActionController se encuentra entre un ActiveRecord (interface para la base de datos) y un ActionView (motor de presentación de los datos). El ActionController provee facilidades para la manipulación y organización de la data que se extrae de la base de datos, y la que viene como

entrada en los formularios Web; para luego redireccionar a la plantilla de vista correspondiente con los datos resultantes del procesamiento previo.

Cada Controller es una clase de ruby, y cada método dentro de dicha clase representa un Action.

Los ActionControllers utilizan las rutas para saber qué controlador se usará, qué método del mismo se ejecutará y qué parámetros se le pasarán. Por defecto, las rutas presentan la siguiente estructura:

http://host/<controlador>/<método>[/<id>]

Al crear una nueva aplicación de Ruby on Rails se crearán una serie de directorios y archivos, los cuales representan una jerarquía organizada y predeterminada por Rails para estructurar los proyectos, llamada también Andamiaje (scaffold).

- Cada directorio cumple una función específica, a continuación se hará la descripción de algunos de estos.
- App: Almacena los componentes de la aplicación: vistas y helpers, controladores y modelos.
- Config: Contiene la configuración de la aplicación: configuración de la base de datos (dagtabase.yml), estructura del entorno Rails (environment.rb) y el enrutador de peticiones Web (routes.rb).
- Public: es similar al directorio public de un Servidor Web. Contiene los archivos JavaScript, imágenes, hojas de estilos y código html común para las interfaces.
- Script: Este directorio contiene scripts que inician y administran las diversas herramientas que se pueden utilizar en Rails; por ejemplo, se tienen los scripts que generan código (generate) e indican un Servidor Web (Server).

Rails sabe dónde encontrar todos los componentes que necesita dentro de la estructura de directorios presentada anteriormente de tal forma que no es necesario especificar detalles del despliegue de la aplicación. El trabajo de los desarrolladores se centra en crear y editar archivos dentro del directorio app, específicamente en los subdirectorios, controller, views y model (Ospina, M y Suarez, S. 2008).

Se puede ver la arquitectura modelo, vista y controlador en la figura 6.

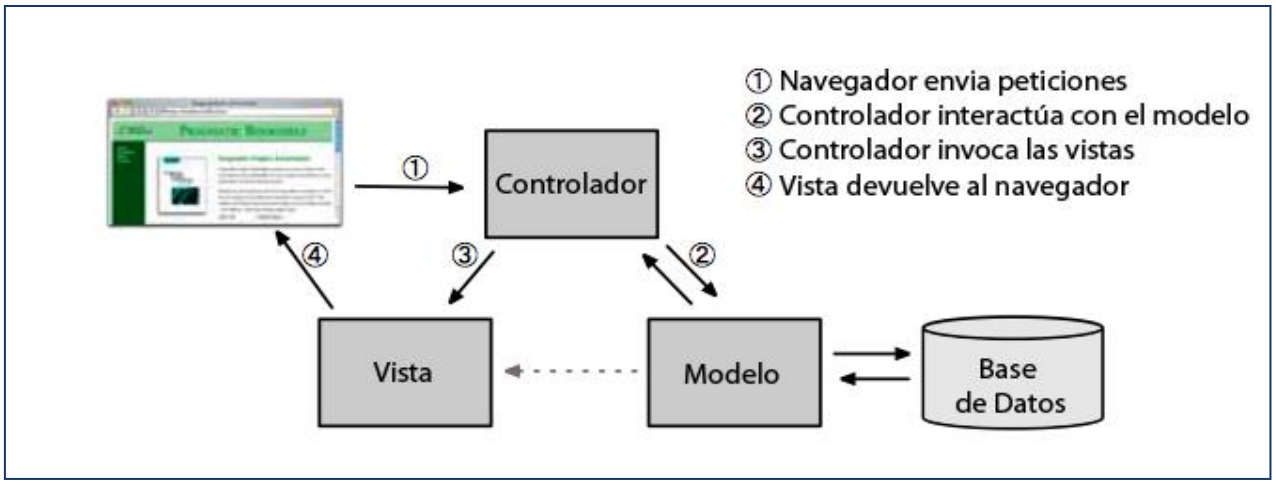


Figura 6 - Modelo Vista Controlador en RAILS.

2.16. Componentes

Los componentes son aquellas aplicaciones que le permiten a los desarrolladores agregarlas a su proyecto de una forma rápida y sencilla con la finalidad de disminuir el tiempo de desarrollo con aplicaciones funcionales que en muchos casos tienen un coste de tiempo alto desarrollarlas desde el comienzo.

Los componentes son aportados por diferentes desarrolladores alrededor del mundo y algunos son incluidos por rails.

CAPITULO III –Marco Aplicativo

En este capítulo se describirán los objetivos y la adaptación de la metodología de desarrollo de software Proceso Unificado Ágil, durante el desarrollo del Gestor de Contenido Modular "GENCI-2" para la Coordinación de Investigación de la Facultad de Ciencias de la Universidad Central de Venezuela.

3.1. Objetivo general de la aplicación

La aplicación desarrollada para el presente Trabajo Especial de Grado tiene como objetivo el de satisfacer las necesidades de la Coordinación de Investigación de la Facultad de Ciencias de la UCV, para que la misma tenga una herramienta que sea capaz de publicar contenido de interés para la comunidad universitaria, y a su vez pueda compartir la herramienta con cada una de las dependencias que la conforman como también sus sub-dependencias para que ellas a su vez puedan tener un sitio web en donde puedan administrar sus publicaciones siguiendo una estructura organizacional acorde con la establecida por la Facultad de Ciencias de la Universidad Central de Venezuela.

3.2. Objetivos específicos de la aplicación.

1. Crear un sitio web para que la Coordinación de Investigación pueda publicar contenido de interés tanto a la comunidad universitaria como al público en general.
2. Ofrecerle a las diferentes dependencias y a sus subdependencias la posibilidad de tener un sitio web en la que puedan publicar contenido de interés tanto a la comunidad universitaria como al público en general.
3. Ofrecer un método sencillo de acceso a los usuarios, mediante un URL asociado al recurso solicitado.
4. Contar con una estructura jerárquica tipo árbol, en la que se pueda crecer en al menos tres niveles para que la herramienta pueda utilizada en cada uno de los nodos.
5. Diseñar interfaces de usuario de acuerdo a los lineamientos de usabilidad aprobados por el cliente y el tutor académico.
6. Crear la Aplicación Web siguiendo los conceptos de gestor de contenido, para ello se debe:
    - Desarrollar el Sistema de Gestión de Particiones Virtuales y sus módulos principales.
    - Desarrollar el Sistema de Administración de Contenidos y sus módulos principales.

- Desarrollar el Sistema de Visualización de Contenido y sus módulos principales.
- Definir y llevar a cabo las pruebas funcionales y de aceptación del sistema.
- Poner en producción en un servidor dentro de la Facultad de Ciencias la aplicación Web para el uso de la Coordinación de Investigación.

3.3. Alcance de la aplicación

Debido a la proyección que puede tener esta aplicación web y en manera de garantizar una implementación de calidad y completamente funcional, se desarrollo una primera versión de la aplicación con las bases y las funcionalidades necesarias para cubrir las necesidades existentes, sin embargo se propone desarrollar módulos adicionales que puedan hacer de esta aplicación una herramienta más versátil y dinámica la cual se pueda ajustar a las necesidades de otros clientes.

Inicialmente durante la propuesta de trabajo especial de grado se planteó realizar los siguientes módulos:

- Sistema de Gestión de Particiones Virtuales:
  - Crear Partición.
  - Diseñar Partición.
  - Gestor de Usuarios.
  - Configuración Global.
- Sistema de Administración de Contenido:
  - Configuración de componentes.
  - Configuración de Web-Personalizada.
  - Crear Menús de navegación.
  - Contenido tipo Evento.
  - Configuración de Página de Inicio.
  - Crear componente de Contenido Tipo Texto.
  - Crear componente de Contenido Tipo Banner.
  - Configuración Parcial.
- Sistema de Visualización de Contenido.

Sin embargo debido a las necesidades de la Coordinación de Investigación y a las sugerencias planteadas durante el desarrollo de la aplicación se desarrollaron los módulos descritos a continuación:

- Modulo de Gestión de Usuarios:
  - Listar Usuarios.
  - Agregar Usuario validado contra el directorio LDAP de la Facultad de Ciencias de la UCV.
  - Editar Usuario.

- Modulo de Gestión de Dependencia / Particiones
    - Listar Hermanos.
    - Editar Hermanos.
    - Listar Hijos.
    - Editar Hijos.
    - Crear Dependencia.
    - Editar Dependencia.

- Modulo de Gestión de Paginas Personalizadas
    - Listar Páginas.
    - Editar Pagina.
    - Crear Pagina Secuencial.
    - Crear Pagina Clásica.
    - Crear Pagina 3 Columnas.
    - Crear Pagina HTML
    - Crear Pagina de Archivos.
    - Crear Página de Inicio.
    - Eliminar Pagina.
    - Construir Pagina.
    - Asociar Componentes.
    - Ordenar Componentes.

- Modulo de Gestión de Menús
    - Crear Menú Horizontal.
    - Crear Menús Verticales.
    - Listar Menús.
    - Editar Menús.
    - Eliminar Menús.

- Modulo de Gestión de Botones.
    - Crear Botón.
    - Listar Botones.
    - Editar Botón.
    - Eliminar Botón.

- Modulo de Gestión de Componente de Texto.
    - Crear Componente Texto.
    - Listar Componentes Texto.
    - Editar Componente Texto.
    - Eliminar Componente Texto.

- Modulo de Gestión de Componente de Banner.
    - Crear Componente Banner.
    - Listar Componentes Banner.
    - Editar Componente Banner.
    - Eliminar Componente Banner.

- Modulo de Gestión de Componente de Evento.
    - Crear Componente Evento.
    - Listar Componentes Evento.
    - Editar Componente Evento.
    - Eliminar Componente Evento.

- Modulo de Gestión de Contenido Tipo Texto.
    - Crear Contenido Tipo Texto.
    - Listar Contenido Tipo Texto.
    - Editar Contenido Tipo Texto.
    - Eliminar Contenido Tipo Texto.

- Modulo de Gestión de Contenido Tipo Evento.
    - Crear Contenido Tipo Evento.
    - Listar Contenido Tipo Evento.
    - Editar Contenido Tipo Evento.
    - Eliminar Contenido Tipo Evento.

- Modulo de Gestión de Contenido Tipo Banner.
    - Crear Contenido Tipo Banner.
    - Listar Contenido Tipo Banner.
    - Editar Contenido Tipo Banner.
    - Eliminar Contenido Tipo Banner.

- Modulo de Consultas de Log o Bitácora
    - Listar últimas entradas al Log (250).
    - Filtrar búsqueda mediante fechas.
    - Visualizar detalle de la entrada al Log.

- Modulo de Ayuda o Tutoriales
    - Listar Opciones de Ayuda.
    - Visualizar Tutorial.

- Modulo de Administración de Genci II
    - Iniciar Sesión de Administración.
    - Cerrar Sesión de Administración.
    - Cambiar de Dependencia a administrar.

- Modulo de Visualización de Contenido
    - Navegar por las Opciones.

3.4. Metodología de Desarrollo de Software

Según Kenneth Laudon una metodología de desarrollo es "un conjunto de métodos, uno o más para cada actividad dentro de cada fase de un proyecto de desarrollo de sistemas. La función primaria de una metodología de desarrollo es proporcionar una disciplina para todo el proceso de desarrollo" (Laudon,K. 2004). Ellos mencionan que, una buena metodología de desarrollo puede establecer la disciplina necesaria para el desarrollo de un software solo si establece estándares para los requerimientos, equipos de trabajo, diseño, programación y pruebas.

También afirman que para lograr un desarrollo de software de alta calidad, las organizaciones deben seleccionar una metodología apropiada que se adapte al proyecto y luego ponerla en producción. La metodología se basa en disciplina y una de ellas es llevar el control del proyecto desde sus comienzos, por lo que, es necesario que los documentos de requerimientos y especificaciones de los sistemas estén completamente detallados, exactos y completos siguiendo un protocolo entre las comunidades para una mejor comunicación.

"Las Metodologías de Desarrollo de Software son aquellas que ayudan a realizar el desarrollo de una aplicación disciplinadamente, haciendo proyecciones a futuro y marcando las pautas para la realización, para así tener control sobre el proyecto en cuanto a muchas características.

Las metodologías han solucionado problemas si son usadas correctamente, como lo pueden ser demoras en la entrega del proyecto y mala utilización de los recursos." (CMS 2008)

A continuación explicaré los dos tipos de metodologia que encontramos actualemente en el mercado como tambien sus subtipos.

3.5. Metodologías Ágiles

Son aquellas metodologías adaptables a las situaciones de los proyectos, "pero una adaptación continua sin un progreso logra muy poco. Por lo tanto, un proceso ágil de software debe adaptarse de forma incremental" (Pressman, 2006). Para que la metodología se pueda adaptar debe existir una retroalimentación con el cliente para lograr todos los requerimientos que el mismo desea en el tiempo necesario.

La metodología ágil tiene algunos principios como lo son:

- Los individuos y sus interacciones son más importantes que los procesos y las herramientas.
- El software que funciona es más importante que la documentación exhaustiva.
- La respuesta y soluciones del proyecto antes que seguir un plan específico y cerrado.

Algunas metodologías de desarrollo de software ágiles son:

- XP (Programación Extrema).
- AUP (Proceso Unificado Ágil)

3.6. Metodología de desarrollo AUP

Es una versión simplificada del *Rational Unified Process* (RUP). En él se describe un procedimiento sencillo y fácil de entender para el desarrollo de software de aplicación comercial ágil, utilizando las técnicas y los conceptos que aún permanecen todavía fieles a RUP.

El enfoque aplica técnicas ágiles e incluyen desarrollo basado en pruebas (TDD), Modelado basado en desarrollo ágil (AMDD), gestión de cambios ágil, y refactorización de base de datos para mejorar su productividad.

La Figura 7 muestra el ciclo de vida de la AUP. Haciendo una comparación de esta metodología con RUP se observa que las disciplinas han cambiado. En primer lugar, la disciplina Modelado de AUP abarca las disciplinas Modelado de Negocios, Requisitos y Análisis y Diseño de RUP. La disciplina Modelado es una parte importante de la AUP, como se puede ver, pero no dominan el proceso – se desea que el proceso permanezca ágil mediante la creación de solo los modelos y documentos estrictamente necesarios. En segundo lugar, la disciplina Configuración y Manejo del cambio es ahora la disciplina configuración de Gestión. En el desarrollo ágil las actividades de gestión de cambios suelen ser parte de los esfuerzos de manejo de requerimientos, que forma parte de la disciplina Modelado. (Ospina, M y Suarez, S. 2008)

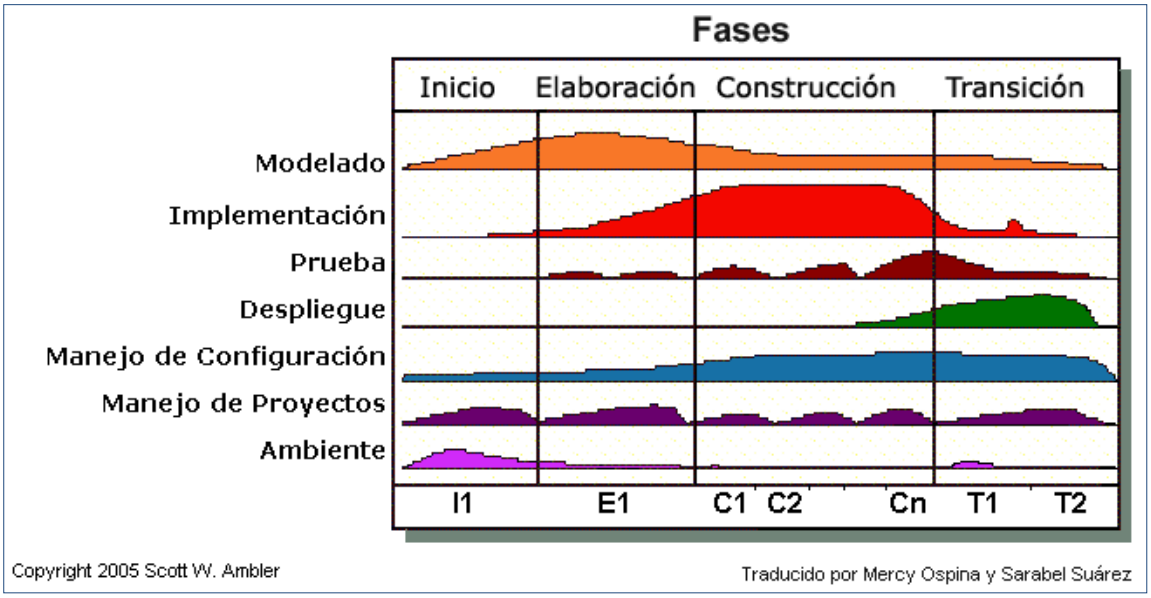


Figura 7 - Ciclo de Vida AUP

UP Ágil es formada por cuatro fases que se ejecutan en serie como se puede observar en la figura 7:

1. **Inicio.** El objetivo es identificar el alcance inicial del proyecto, una potencial arquitectura de la aplicación, y para obtener la financiación inicial del proyecto y la aceptación de los interesados.
2. **Elaboración.** El objetivo es probar la arquitectura de la aplicación. En esta fase el equipo de desarrollo se prepara para la fase de construcción creando el ambiente de desarrollo mediante la configuración del hardware, el software y las herramientas necesarias para la creación de la aplicación.
3. **Construcción**. El objetivo es construir software de trabajo, en una base regular e incremental que cumpla las necesidades altamente prioritarias de los interesados del proyecto.
4. **Transición.** El objetivo es validar y desplegar la aplicación en su entorno de producción. Se pueden llevar a cabo pruebas exhaustivas durante esta fase, incluidas las pruebas beta. El ajuste del producto se lleva a cabo aquí, así como la revisión para hacer frente a defectos importantes. El tiempo y esfuerzo que se puede invertir en esta fase varía de un proyecto a otro.

3.6.1. Disciplinas e Iteraciones

Cada fase se divide en disciplinas y estas a su vez se pueden dividir en varias iteraciones dependiendo de la complejidad del problema. Las disciplinas se realizan de manera sistemática, definiendo las actividades de desarrollo que realizan los miembros del equipo para construir, validar y entregar software de trabajo que satisfaga las necesidades de sus interlocutores. Las disciplinas son las siguientes (Ospina, M y Suarez, S. 2008):

Modelado: El objetivo de esta disciplina es entender el negocio de la organización, el dominio del problema hacía el que está dirigido el proyecto, e identificar una solución viable para resolver el dominio del problema. Ver figura 8.

Para esta disciplina no es necesario crear todos los modelos existentes, solo los más apropiados a la situación, ni es necesario modelar con muchos detalles en las fases de Inicio y Elaboración En la fase de Construcción se deben realizar tormentas de ideas, siguiendo la filosofía Justo a Tiempo (JIT por sus siglas en inglés) para obtener los detalles necesarios para cada iteración

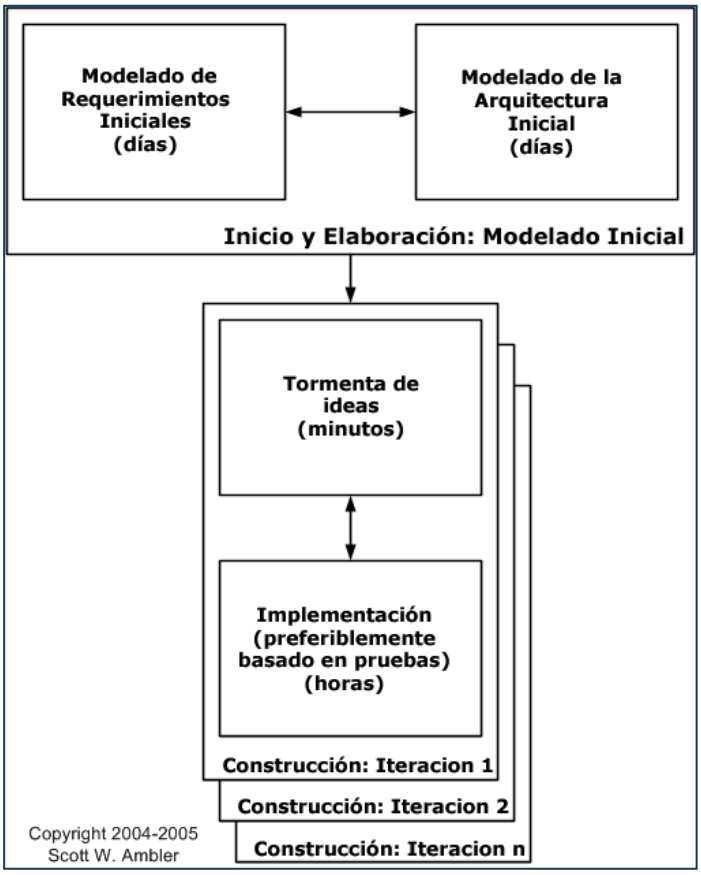


Figura 8 - Ciclo de vida del Desarrollo (AM)

- **Aplicación**: El objetivo de esta disciplina es transformar el (los) modelo(s) en código ejecutable y llevar a cabo un nivel básico pruebas, en particular, las pruebas unitarias.

En esta disciplina se recomienda trabajar en parejas y bajo el esquema de desarrollo basado en pruebas.

- **Pruebas:** El objetivo de esta disciplina es realizar una evaluación objetiva para garantizar la calidad. Esto incluye la búsqueda de defectos, la validación de que el sistema funciona tal como está establecido, y verificar que se cumplan los requisitos. Se puede ver en la figura 9.

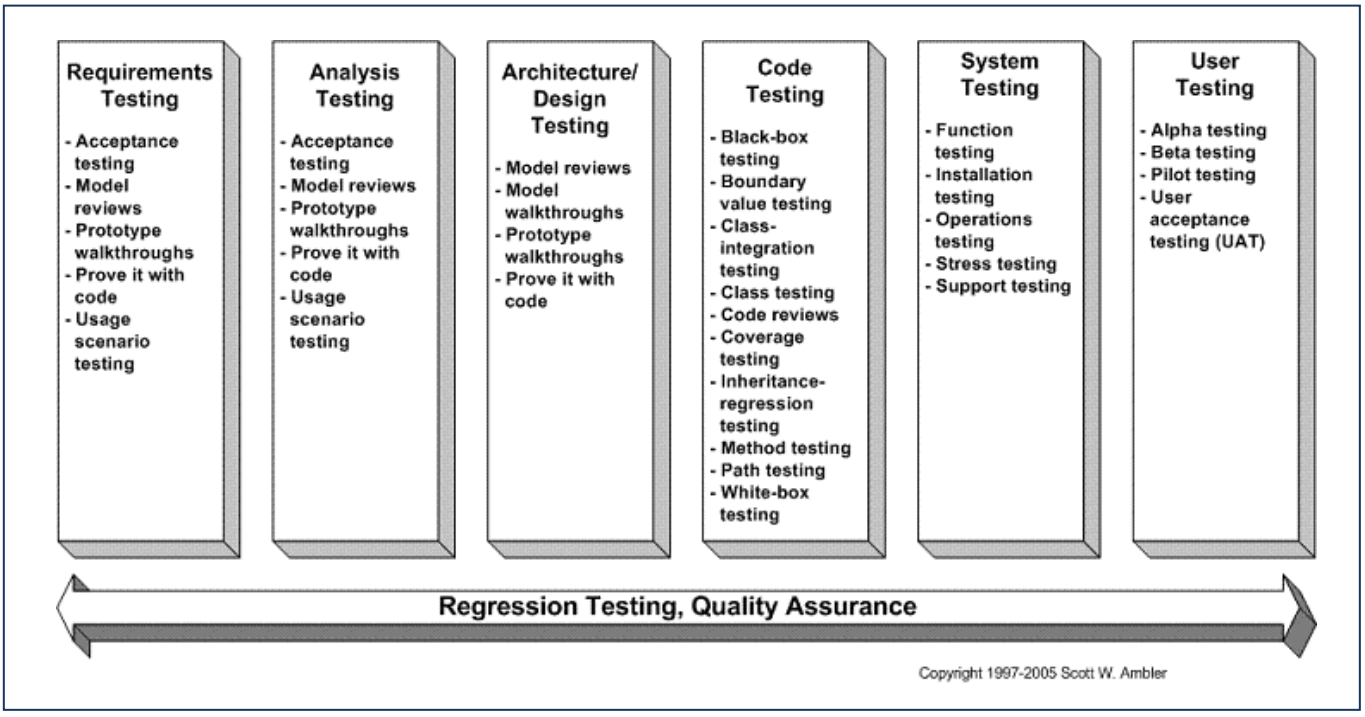


Figura 9 - Ciclo de vida de las pruebas orientadas a objetos (FLOOT)

- **Despliegue**: El objetivo de esta disciplina planificar para la entrega del sistema y ejecutar el plan para hacer que el sistema este a disposición de los usuarios finales.

- **Gestión de configuración**. El objetivo de esta disciplina es gestionar el acceso a los artefactos del proyecto. Esto incluye no sólo el seguimiento de versiones de los artefactos a lo largo del tiempo, sino también el control y la gestión de cambios en ellos.

- **Gestión de Proyectos**. El objetivo de esta disciplina es dirigir las actividades que lleva a cabo en el proyecto. Esto incluye la gestión de riesgos, dirección de personas (la asignación de tareas, seguimiento de los progresos, etc.), y coordinar con las personas y los sistemas fuera del alcance del proyecto para garantizar que se entrega a tiempo y dentro del presupuesto.

- **Medio ambiente**. El objetivo de esta disciplina es apoyar el resto del esfuerzo por garantizar que el proceso adecuado, la orientación (normas y directrices), y herramientas (hardware, software, etc.) están disponibles para el equipo según sea necesario.

3.6.2. Entrega de versiones incrementales en el tiempo

En lugar del enfoque de "big bang" donde se cumple con la entrega de todos los programas a la vez, los equipos de desarrollo AUP suelen emitir revisiones al final de cada iteración en escenarios de pre-producción. Una versión de desarrollo de una aplicación es algo que podría ser liberado en producción si pasa a través de un modulo de aseguramiento de calidad (QA), las pruebas, y procesos de despliegue, en pre -producción.

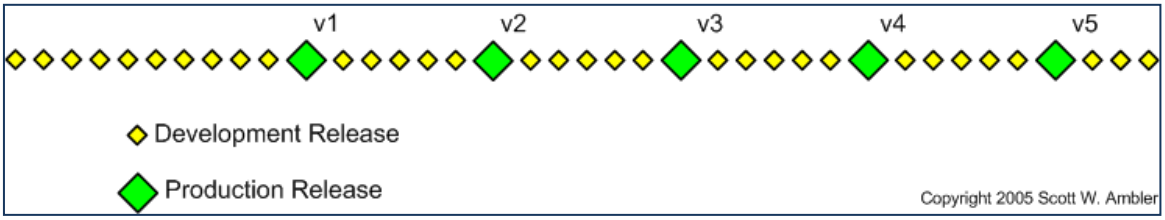


Figura 10 - Crecimiento de las versiones en el tiempo

En la figura 10, se puede ver que la primera versión de producción a menudo toma más tiempo que las versiones posteriores; en la primera versión de un sistema se debe tomar en cuenta el tiempo dedicado a crear el ambiente de trabajo y la adaptación del equipo de desarrollo, lo que permitirá tener una colaboración eficaz. La primera versión de producción podría tomar doce meses para entregar, la segunda versión de nueve meses, y luego otras versiones pueden ser entregadas cada seis meses (Ospina, Mercy y Suarez, Sarabel , 2008).

3.6.3. Filosofías de la AUP

El UP Ágil se basa en los siguientes principios:

1. **Confianza en las capacidades del personal.**
   No es necesaria una documentación detallada del proceso ya que casi nadie la lee, esto se sustituiría por guías de alto nivel y / o capacitación. El producto AUP ofrece flexibilidad en este aspecto dejando que los desarrolladores tomen la decisión de cuales documentos son necesarios.
2. **Simplicidad**
   Se busca que los documentos sean concisos y precisos, además de cortos.
3. **Agilidad**
   El Agile UP se ajusta a los valores y principios de la Agile Alliance.
4. **Centrado en las actividades de alto valor**
   La atención se centra en las actividades que realmente cuentan, no cada cosa que podría suceder en un proyecto.
5. **Independencia de herramientas**
   Se puede usar cualquier conjunto de herramientas con el Agile UP.
6. **Adaptación**
   Para satisfacer las necesidades del proyecto. La metodología AUP es fácilmente a las necesidades de cada proyecto.

3.7. Factores que implican la selección adecuada de una metodología

La selección de una metodología para el desarrollo de software implica varios factores como lo pueden ser: tiempo de desarrollo, calidad, reducción de costes, cumplir los requerimientos establecidos. Todos los aspectos mencionados anteriormente se deben tener en cuenta al momento de seleccionar la metodología, es decir, si las necesidades se adaptan a una solución rápida, ordenada y actualizable, es posible que la decisión se incline hacia una metodología ágil de desarrollo, en cambio si el proyecto es muy grande, donde todo debe quedar específicamente documentado, en el que se debe tener el control sobre todas las decisiones, es recomendada la utilización de una metodología tradicional o pesada.

Las metodologías definen el proceso que se debe cumplir para llevar a cabo el proyecto de principio a fin, pero también existen metodologías ligadas a cómo debemos hacer partes específicas del desarrollo, como lo es la interfaz con la que el usuario va a interactuar, por lo cual ahora procedemos a explicar los conceptos relacionados con la usabilidad y lineamientos de interfaces.

3.8. Adaptación de la metodología Proceso Unificado Ágil (AUP)

Para el desarrollo del Gestor de Contenido Modular "GENCI-2" para la Coordinación de Investigación de la Facultad de Ciencias de la Universidad Central de Venezuela se implementaron las diferentes fases de la metodología de desarrollo de software Proceso Unificado Ágil, la cual sigue los principios de la Modelación ágil, con la finalidad de que este se realizara de manera rápida, sencilla, eficiente y con la suficiente documentación. También se tomo esta decisión por estar familiarizados con los artefactos o entregables UML.

A continuación se expone la adaptación de las fases del Proceso Unificado Ágil (AUP) del presente trabajo especial de grado.

3.9. Inicio

En la fase de inicio de esta metodología se indican los requerimientos, objetivos y alcance de este trabajo especial de grado, los cuales son detallados en la sección 3.1, 3.2 y 3.3 de este documento.

3.9.1. Usuarios del Sistema

El sistema debe contar con funcionalidades que permitan la actualización de la información de los requerimientos mencionados anteriormente de manera sencilla, por tal motivo se presentan a dos grupos de usuarios:

Usuarios Públicos: los mismos corresponden al grupo de usuarios que solicitan información y navegan a través de las páginas consultando libremente su contenido.

Usuarios Administrativos: permite el acceso sólo a usuarios autorizados, los cuales tendrán podrán pertenecer a diferentes roles los cuales a su vez tendrán acceso o no a las herramientas de construcción de páginas, dependencias o contenido.

En la figura 11 se muestra una clasificación de los usuarios públicos, la cual se realizó con la finalidad de identificar los distintos tipos de usuarios del sistema y poder determinar las necesidades de cada uno, sin embargo el usuario que interactuará con el sistema es el usuario general.

En la figura 12 se muestran los usuarios administrativos, en este caso si cumplen distintos roles en el sistema.

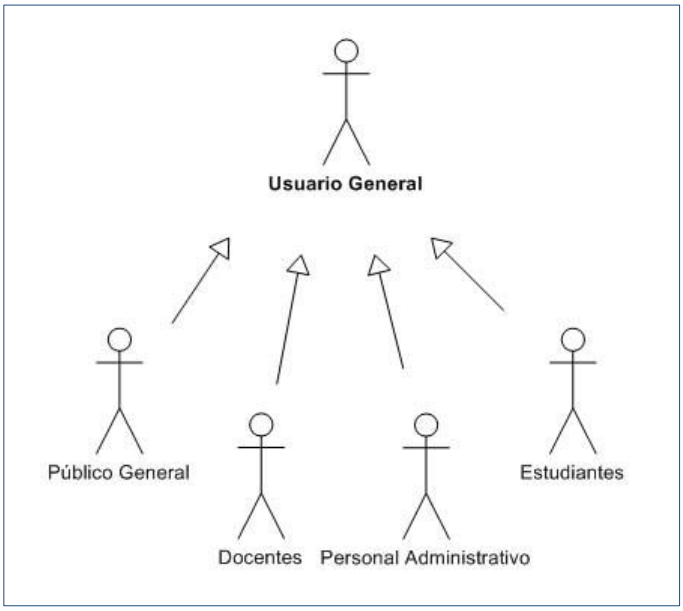


Figura 11 - Usuarios Públicos

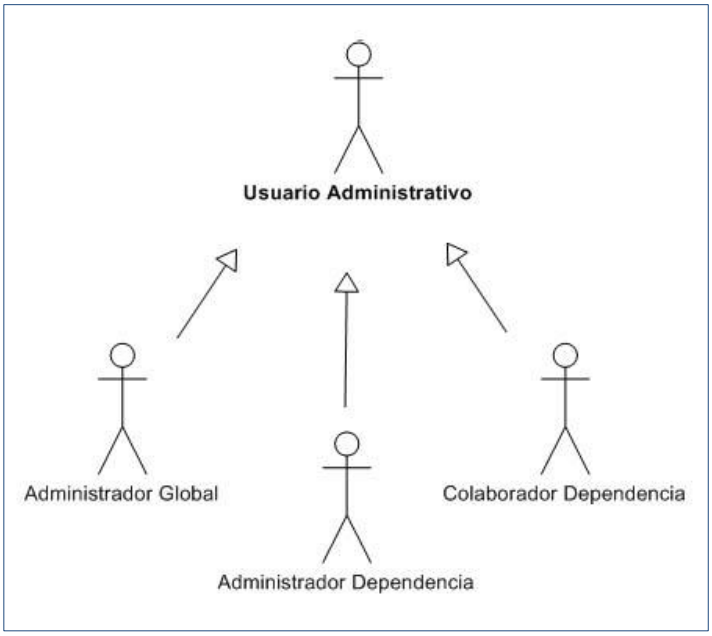


Figura 12 - Usuarios Administrativos

3.9.2. Perfiles de usuario

Usuario	Descripción	Necesidades
Usuarios Públicos (Publico General, Estudiante, Docente, Personal Administrativo)	Personas naturales o jurídicas que navegan en internet y consiguen artículos o información de interés para ellos.	• Saber navegar en Internet
Administrador Global	Figura perteneciente a la Facultad de Ciencias la cual tiene la capacidad de administrar todas las dependencias de la facultad, como también su contenido y estructura.	• El usuario debe pertenecer a la facultad de ciencias como un miembro activo ya que la autenticación se realiza directamente contra el directorio de LDAP de la Facultad de Ciencias. • El usuario debe ser responsable de sus acciones ya que con este acceso se le dará control total sobre el contenido, estructura y distribución de los elementos que componen las diferentes páginas creadas con Genci II.

Administrador Dependencia	Figura perteneciente a la Facultad de Ciencias la cual tiene la capacidad de administrar la dependencia la cual se le asigne, como también su contenido y estructura.	• El usuario debe pertenecer a la facultad de ciencias como un miembro activo ya que la autenticación se realiza directamente contra el directorio de LDAP de la Facultad de Ciencias. • El usuario debe ser responsable de sus acciones ya que con este acceso se le dará control total sobre el contenido, estructura y distribución de los elementos que componen la página de la dependencia a la cual se le está dando acceso.
Colaborador Dependencia	Figura perteneciente a la Facultad de Ciencias la cual tiene la capacidad de administrar el contenido de la dependencia a la cual se asigne.	• El usuario debe pertenecer a la facultad de ciencias como un miembro activo ya que la autenticación se realiza directamente contra el directorio de LDAP de la Facultad de Ciencias. • El usuario solo podrá administrar el contenido de dependencia a la cual está asignado.

Tabla 1 - Perfiles de Usuario.

3.9.3. Arquitectura de la aplicación

- **Arquitectura Física**

  Como se indico en el capítulo 2 las aplicaciones Web están basadas en arquitectura cliente-servidor, esta arquitectura se muestra en la figura 13

- **Arquitectura Lógica**

  El sistema desarrollado en el presente trabajo de investigación está basado en una arquitectura MVC con MySQL descritos en el capítulo 2 del presente documento.

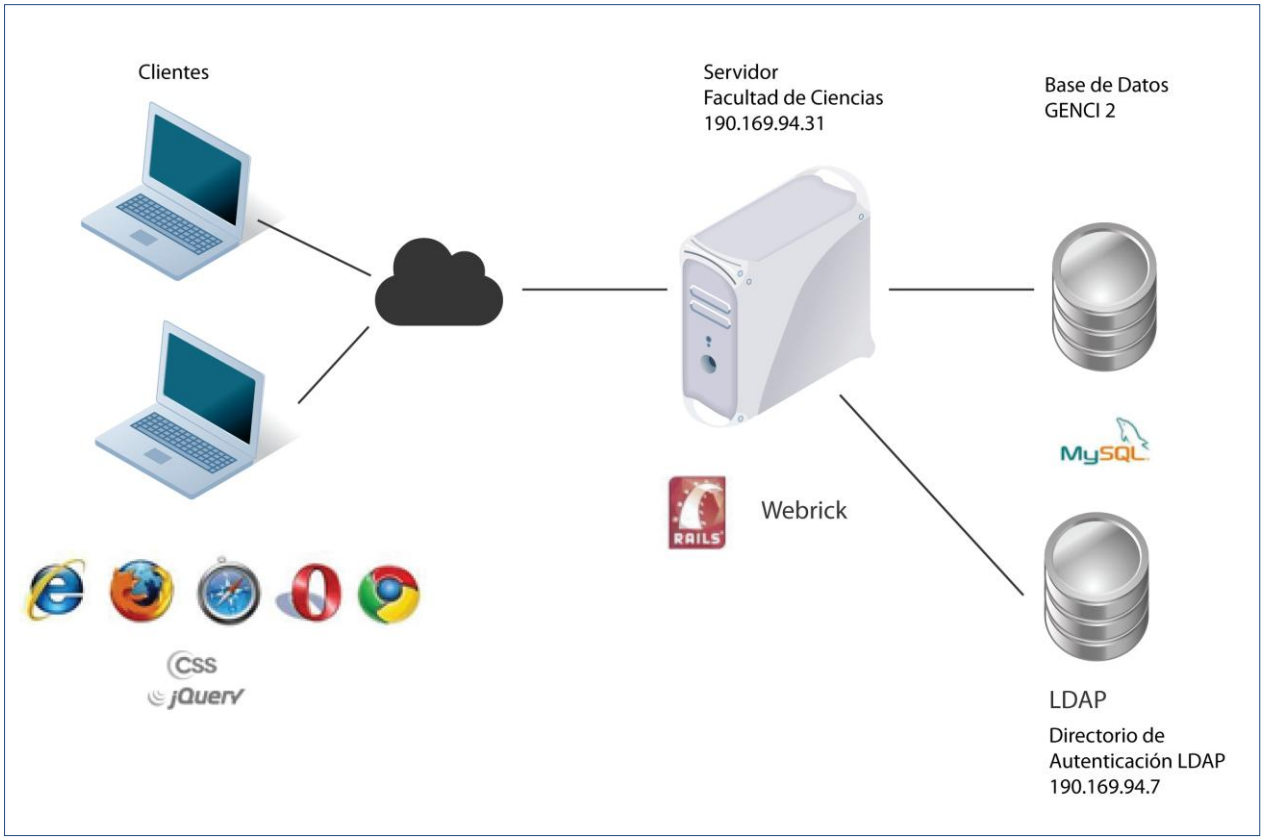


Figura 13 - Arquitectura Física

3.9.4. Prototipos de Diagramación de Interfaz

Durante la propuesta de trabajo especial de grado se presentaron una serie de bocetos los cuales fueron aprobados para continuar la aplicación, y a partir del mismo se realizaron prototipos de interfaz haciendo énfasis en la diagramación que iban a tener las posibles vistas del sistema Genci II.

Existen dos ambientes para los que se realizaron prototipos, por una parte tenemos las diagramaciones del lado de los usuarios públicos, los cuales se dividen en diagramación Clásica, Secuencial, 3 Columnas y HTML o Archivos, por otra parte tenemos la diagramación del administrador.

En la figura 14 se muestra el prototipo “Clásico” el cual es una de las opciones para las páginas resultantes creadas por el sistema.

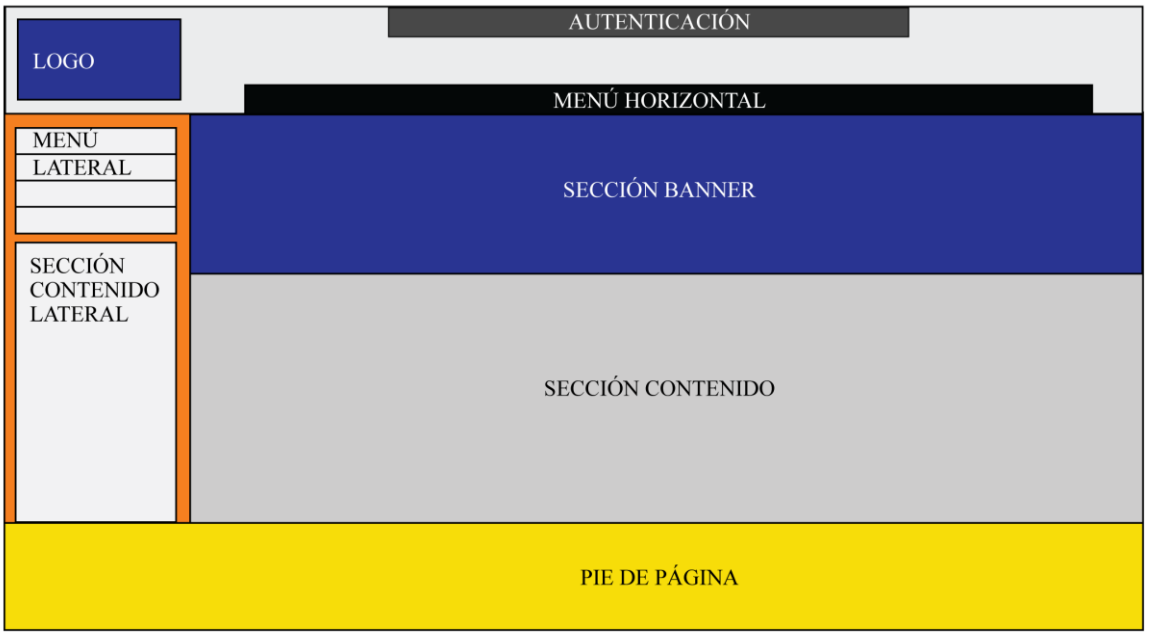


Figura 14 - Diagramación Clásica

En la figura 15 se muestra el prototipo “Secuencial” el cual es una de las opciones para las páginas resultantes creadas por el sistema.

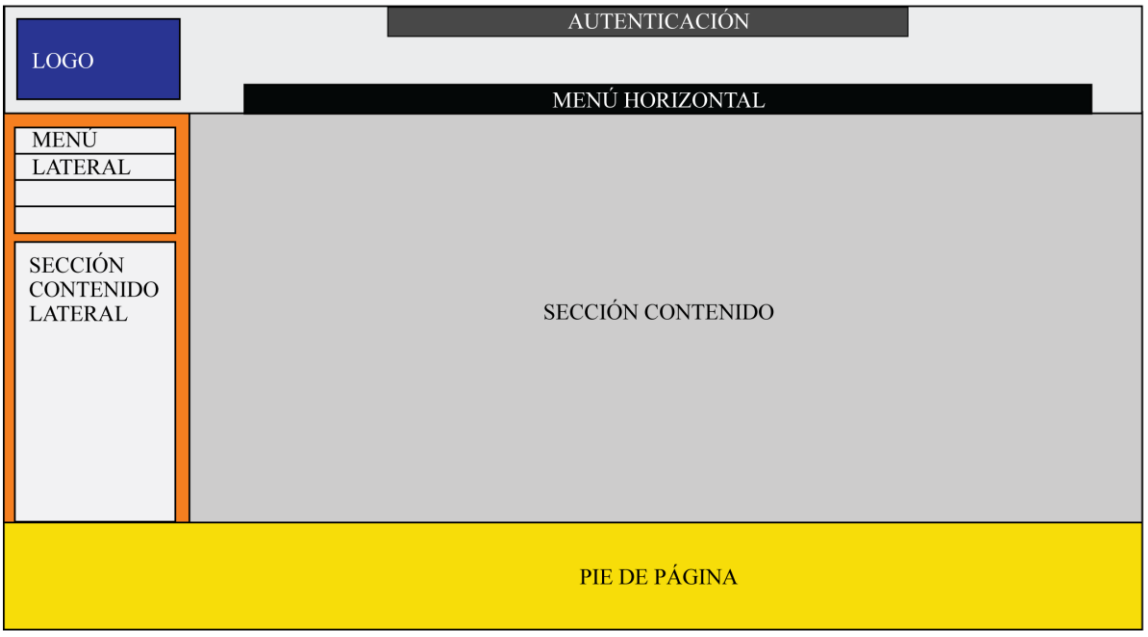


Figura 15 - Diagramación Secuencial

En la figura 16 se muestra el prototipo “3 Columnas” el cual es una de las opciones para las páginas resultantes creadas por el sistema.

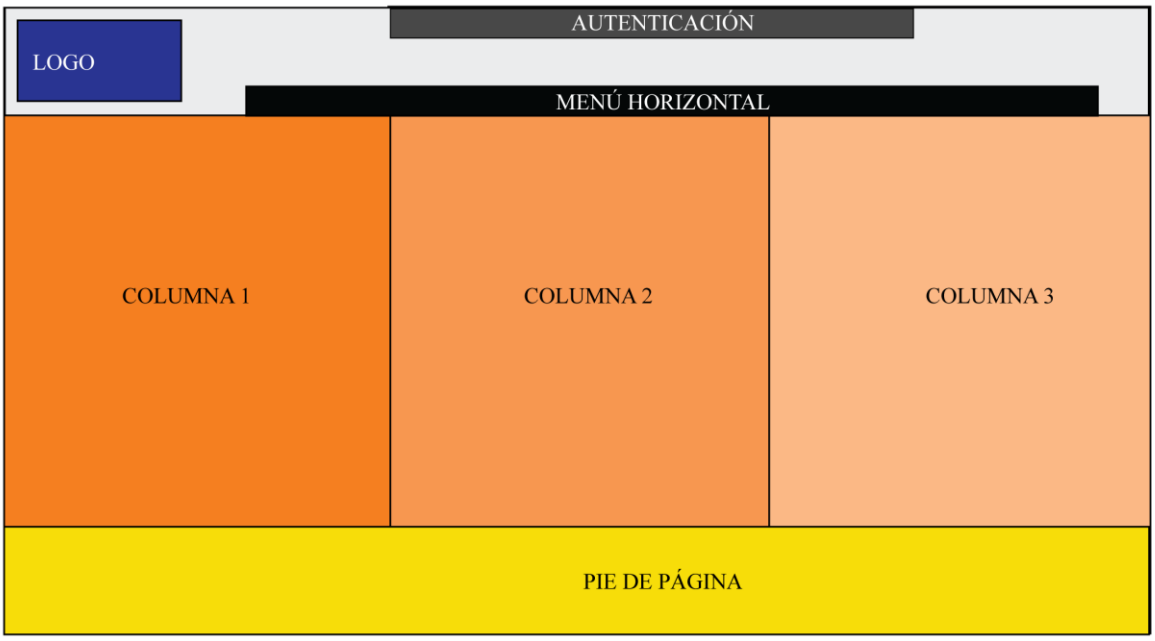


Figura 16 - Diagramación 3 Columnas

En la figura 17 se muestra el prototipo “HTML” ó “Archivos” el cual es una de las opciones para las páginas resultantes creadas por el sistema.

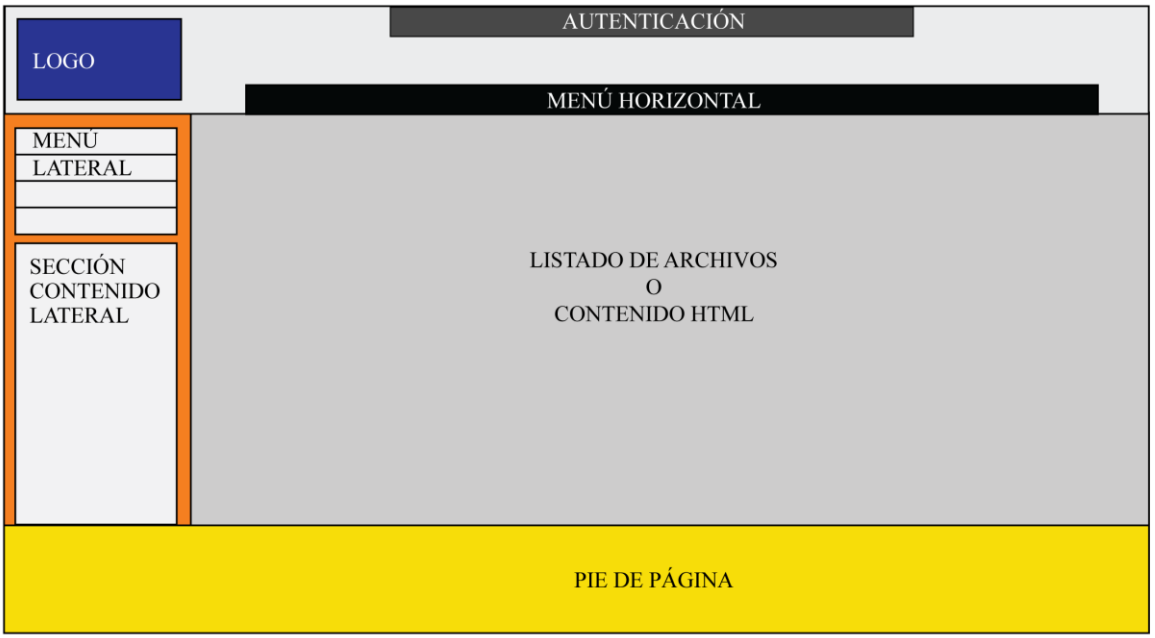


Figura 17 - Diagramación HTML

En la figura 18 se muestra el prototipo “Administrativo” cuya diagramación es igual para todos los módulos de administración.

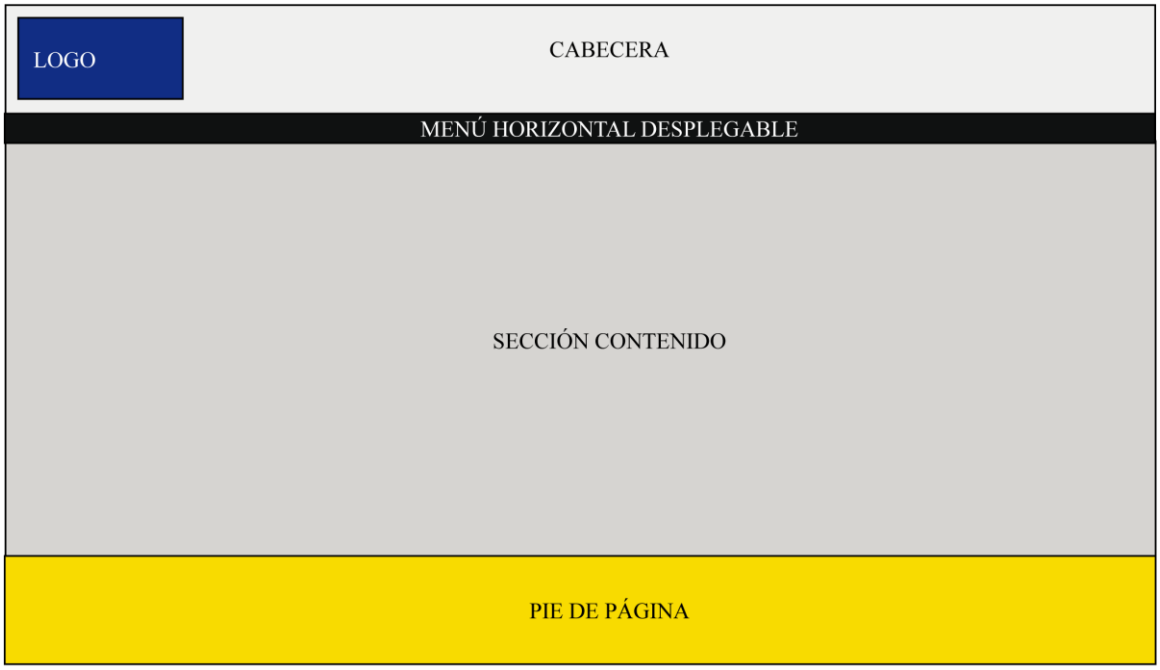


Figura 18 - Administración

3.9.5. Modelo de Datos

Un modelo de datos muestra de manera grafica la forma como los datos se relacionan entre sí, formando una estructura en la cual se soporta la aplicación que se creó.

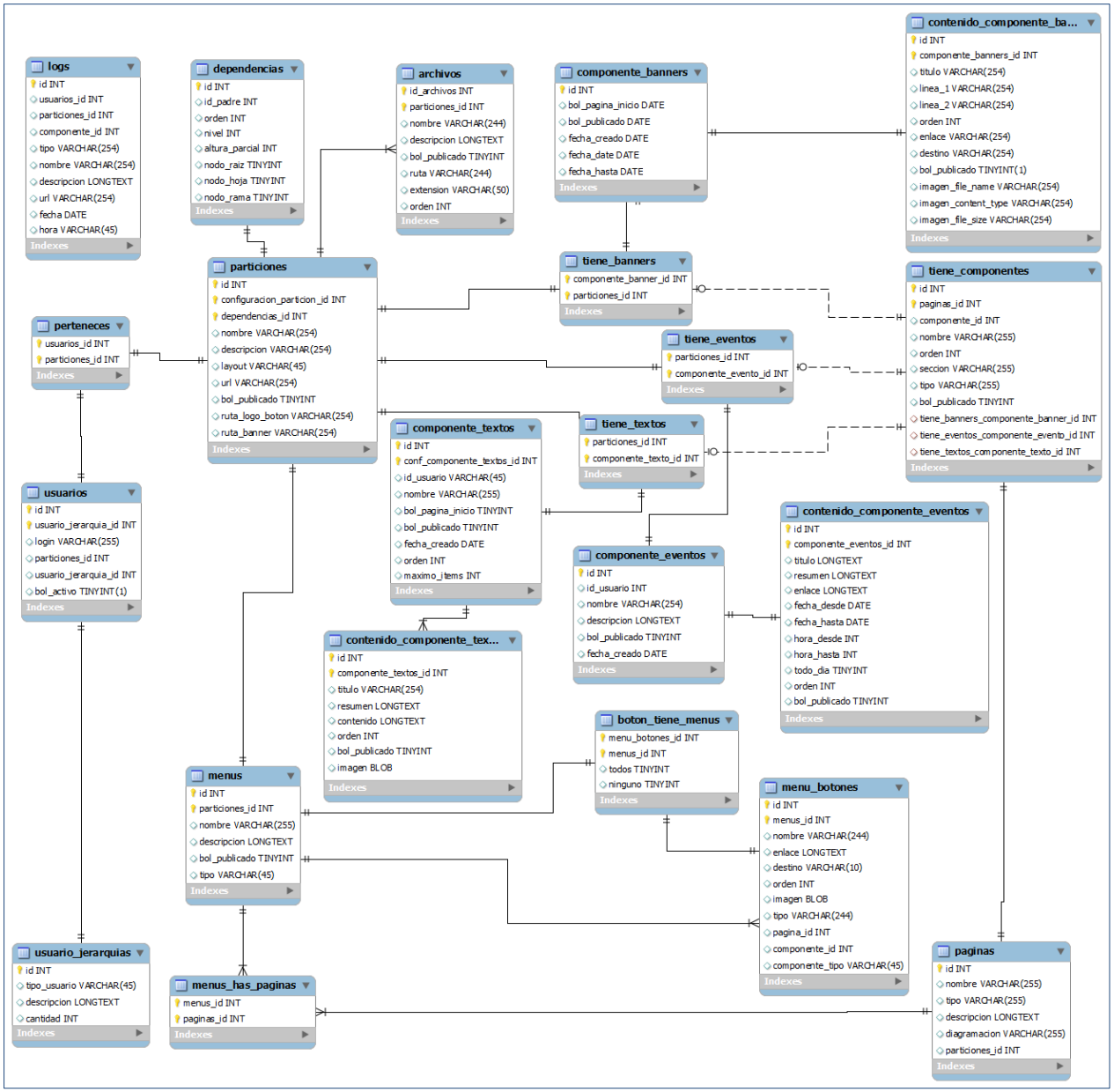


Figura 19 - Modelo de Datos

3.10. Elaboración

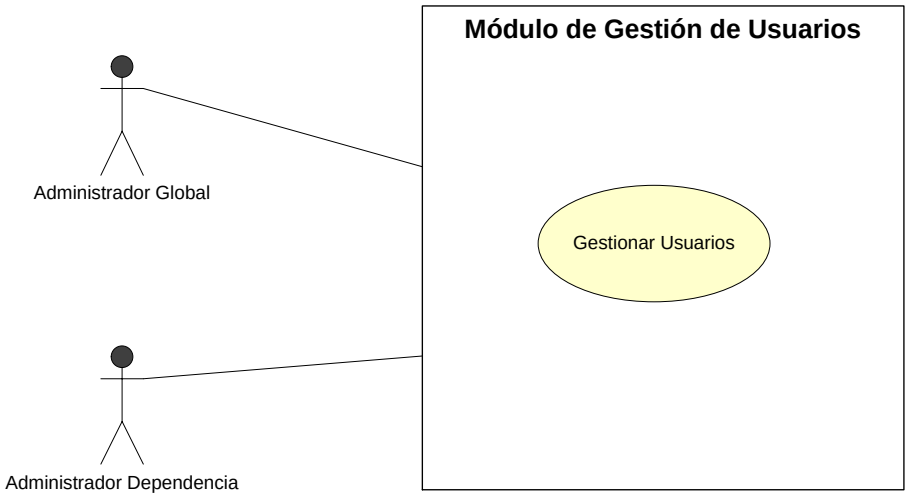
En esta etapa se determino la arquitectura del sistema, se realizaron el diagrama de clases inicial y el prototipo de interfaz de usuario.

3.10.1. Casos de uso

A continuación se exponen los diferentes diagramas de casos de uso realizados para la aplicación Web GENCI - 2, en los niveles 0 y 1.

- **Caso de Uso de Modulo Gestión de Usuarios**

Nivel 0



Nivel 1

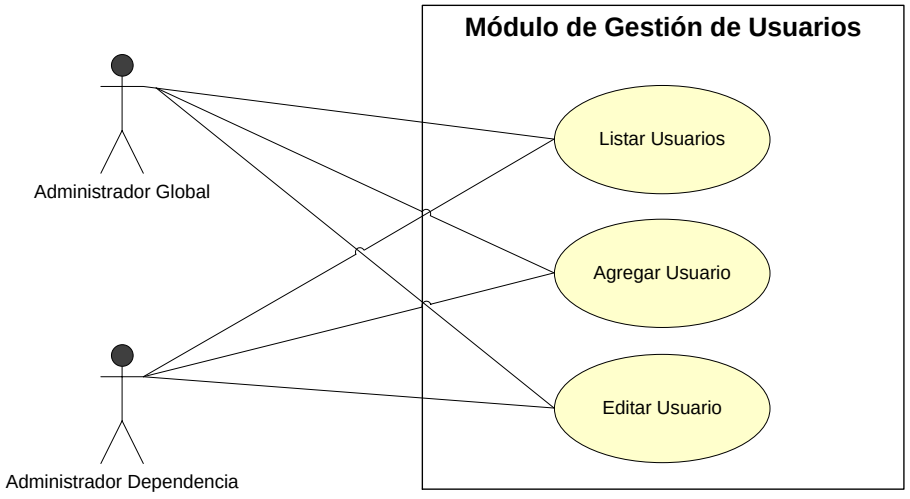


Figura 20 - Caso de Uso Gestión de Usuarios

- **Descripción de Modulo de Gestión de Usuarios.**

Caso de Uso	1. Gestionar Usuarios
Actores	Administrador Global Administrador Dependencias
Descripción	Administrar la creación de usuarios que se encuentren validados en el directorio del LDAP de la Facultad de Ciencias de la Universidad Central de Venezuela. Asignar los diferentes roles según los permisos que deba tener el usuario y de dependencias.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar los usuarios deberá seleccionar la opción del menú Gestor de Usuarios. 3. Luego se mostrara el listado de usuarios creados y la opción de editarlos o crear uno nuevo. 4. En ambos casos se puede asignar el rol y el estado de activación.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. En caso de que el usuario administrador global o de dependencias haya podido ingresar, al momento de crear un usuario este debe encontrarse en el directorio LDAP porque sino mostrara un error de creación, al igual de que si el usuario ya se encuentra creado no se podrá crear el mismo nuevamente.
Pre condiciones	El usuario debe estar autenticado en la aplicación y presionar el botón de Gestor de Usuario.
Post condiciones	El usuario pudo realizar la gestión de usuarios, bien sea listar los existentes, crear uno nuevo, editar los usuarios o eliminarlos del sistema.

Tabla 2 - Caso de Uso: Gestionar Usuarios.

Caso de Uso	2. Listar Usuarios
Actores	Administrador Global Administrador Dependencias
Descripción	Listar los usuarios que se encuentren creados y se mostrara el siguiente detalle de cada uno: ID, Nombre y Apellido, Usuario, Correo, Descripción, Rol y las acciones que en este caso sería la de editar al usuario.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 5. Se muestra el menú principal y para gestionar los usuarios deberá seleccionar la opción del menú Gestor de Usuarios. 6. Luego se mostrara el listado de usuarios creados.

Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario pudo autenticarse en la aplicación y selecciono en el menú la opción Gestor de Usuarios.
Post condiciones	El usuario puede observar el listado de usuarios creados y un breve detalle de cada uno.

Tabla 3 - Caso de Uso: Listar Usuarios.

Caso de Uso	3. Agregar Usuarios
Actores	Administrador Global Administrador Dependencias
Descripción	Gestionar la creación de los usuarios que se encuentren en el directorio LDAP de la Facultad de Ciencias de la Universidad Central de Venezuela, asignándoles el nombre de usuario que debe coincidir con el LDAP, Activación de Usuarios si se encuentra Activo o Inactivo y el Tipo de Usuario o rol que tendrá en la Aplicación GENCI-2 en este caso los diferentes tipos son los siguientes: Administrador Global, Administrador Dependencia Hereditaria, Administrador Dependencia y Colaborador Dependencia.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar los usuarios deberá seleccionar la opción del menú Gestor de Usuarios. 3. Luego debe seleccionar la opción de Crear Nuevo Usuario. 4. Debe completar los datos del usuario a crear en el formulario desplegado.
Flujos Alternos	3. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 4. En caso de que el usuario administrador global o de dependencias haya podido ingresar, al momento de crear un usuario este debe encontrarse en el directorio LDAP porque sino mostrara un error de creación, al igual de que si el usuario ya se encuentra creado no se podrá crear el mismo nuevamente.
Pre condiciones	El usuario debe estar autenticado en la aplicación y presiono el botón de Crear Nuevo Usuario. Además, el usuario debe encontrarse previamente creado en el directorio LDAP de la Facultad de Ciencias de la Universidad Central de Venezuela.
Post condiciones	El usuario pudo crear el usuario con los atributos deseados y se encuentra validado en el directorio LDAP de la Facultad de Ciencias de la Universidad Central de Venezuela.

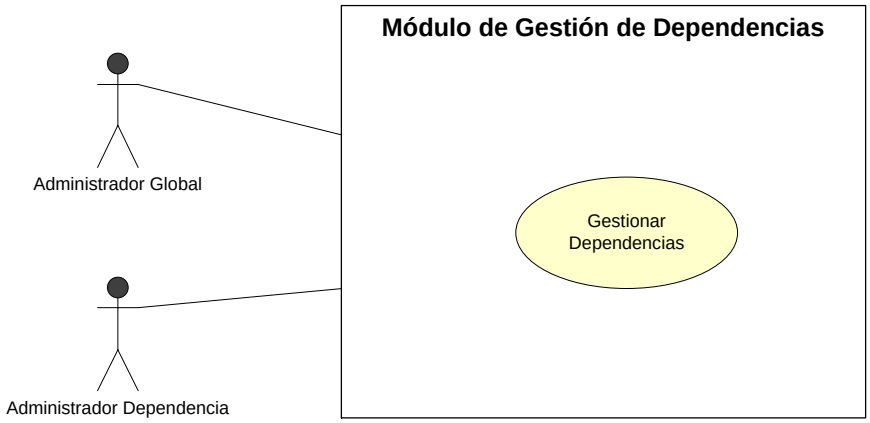
Tabla 4 - Caso de Uso: Agregar Usuarios.

Caso de Uso	4. Editar Usuarios
Actores	Administrador Global Administrador Dependencias
Descripción	Se podrá editar los usuarios que se encuentren listados, modificando solo el rol y la dependencia a la que pertenezcan.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar los usuarios deberá seleccionar la opción del menú Gestor de Usuarios. 3. Luego en el listado que se muestra debe seleccionar el icono en forma de lápiz que se encuentra en la columna de acciones y presionarlo. 4. El botón anterior lo llevara a un formulario con algunos datos precargados que no podrán ser modificados del usuario tales como: Usuario, Nombre y Apellido, Documento de Identificación, Descripción y otros datos a editar como la activación del usuario y el tipo de usuario. Dependendiendo del tipo de usuario podrá modificar la dependencia a la cual pertenece.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. En caso de que el usuario administrador global o de dependencias haya podido ingresar, al momento de editar un usuario debe seleccionar un tipo de usuario, ya que mostrara un error porque el usuario debe tener un rol en la aplicación.
Pre condiciones	El usuario debe estar autenticado en la aplicación y listo a los usuarios creados y seleccionó el botón en forma de lápiz del que se quiere editar.
Post condiciones	El usuario realizó el cambio de los atributos del usuario que selecciono.

Tabla 5 - Caso de Uso: Editar Usuarios.

- **Caso de Uso Modulo de Gestión de Dependencias.**

Nivel 0



Nivel 1

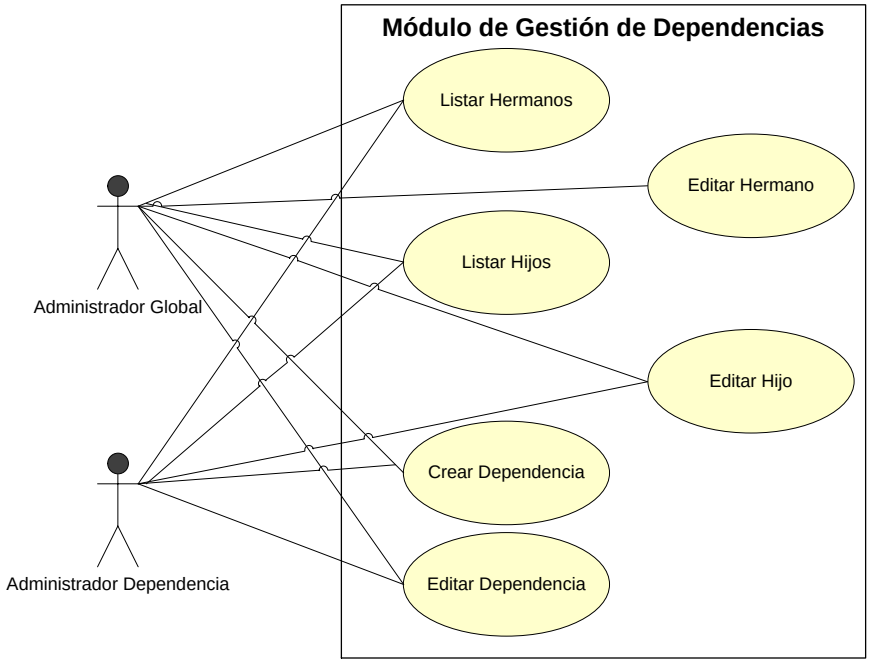


Figura 21 - Caso de Uso Gestión de Dependencias

- **Descripción del Modulo de Gestión de Dependencias.**

Caso de Uso	5. Gestionar Dependencias
Actores	Administrador Global Administrador Dependencias
Descripción	Se encarga de administrar las dependencias, es decir crear o editar dependencias de forma jerárquica establecida en un árbol de orden N y de altura H. En donde los creados o editados forman parte de dicha estructura.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar las dependencias, deberá seleccionar la opción del menú Particiones Dependencias. 3. Luego se mostrara la partición en la que se encuentra actualmente el usuario, las particiones hermanas creadas y las particiones hijas. Por ejemplo si se encuentra el usuario en Coordinación de Investigación sus particiones hijas podrían ser las diferentes escuelas de la Facultad de Ciencias. 4. Del lado derecho se encuentra un formulario en el cual pueden crear un hijo o un hermano de la partición en la que se encuentra el usuario actualmente.
Flujos Alternos	5. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 6. En caso de que el usuario administrador global o de dependencias haya podido ingresar, al momento de crear una dependencia que ya se encuentre creada mostrara un error ya que no se puede crear la misma dos veces.
Pre condiciones	El usuario pudo autenticarse en la aplicación y selecciono el botón de Particiones Dependencias en el que podrá realizar toda la gestión de dependencias.
Post condiciones	El usuario realizó la gestión en el módulo de dependencias necesaria.

Tabla 6 - Caso de Uso: Gestionar Dependencias.

Caso de Uso	6. Listar Hermanos
Actores	Administrador Global Administrador Dependencias
Descripción	Muestra el listado de hermanos de una dependencia creada y según el rol del usuario autenticado tendrá la opción de editar al hermano o no.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar los hermanos de una dependencia, deberá seleccionar la opción del menú Particiones Dependencias. 3. Luego se mostrara la lista de hermanos de esa partición y el botón de editar dependiendo del rol del usuario autenticado.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. En caso de que no existan hermanos de la dependencia se muestra un mensaje informativo de que la partición no tiene hermanos.
Pre condiciones	El usuario debe estar autenticado en la aplicación y selecciono la opción en el menú particiones dependencias.
Post condiciones	El usuario pudo observar el listado de los nodos hermanos que existen.

Tabla 7 - Caso de Uso: Listar Hermanos.

Caso de Uso	7. Editar Hermanos
Actores	Administrador Global
Descripción	El objetivo es poder editar un hermano de la partición en la que se encuentra el usuario.
Flujo Básico	1. El usuario administrador global debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar los hermanos de una dependencia, deberá seleccionar la opción del menú Particiones Dependencias. 3. Luego se mostrara la lista de hermanos de esa partición y dependiendo del rol del usuario autenticado se le mostrara la opción de editar a ese hermano o no. 4. Para editar al hermano debe seleccionar el botón editar y podrá cambiar el nombre, descripción, Layout y el estado de publicación en la que se encuentra.
Flujos Alternos	1. El usuario administrador global no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 3. En caso de que no existan hermanos de la dependencia se muestra un mensaje informativo de que la partición no tiene hermanos.

Pre condiciones	El usuario es de tipo Administrador Global y se encuentra autenticado en la aplicación. Presiono el botón editar que se encuentra en el listado de cada hermano.
Post condiciones	El usuario realizó las modificaciones necesarias en el nodo hermano.

Tabla 8 - Caso de Uso: Editar Hermanos.

Caso de Uso	8. Listar Hijos
Actores	Administrador Global Administrador Dependencias
Descripción	Expone el listado de hijos de una dependencia creada.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar los hijos de una dependencia, deberá seleccionar la opción del menú Particiones Dependencias. 3. Luego se mostrara la lista de hijos de esa partición y el botón de editar siempre y cuando el usuario autenticado sea un Administrador Global o un Administrador Dependencias.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. En caso de que no existan hijos de la dependencia se muestra un mensaje informativo de que la partición no tiene hijos.
Pre condiciones	El usuario debe estar autenticado en la aplicación y selecciono la opción Particiones Dependencias.
Post condiciones	El usuario pudo observar el listado de nodos hermanos de la dependencia en la que se encuentra.

Tabla 9 - Caso de Uso: Listar Hijos.

Caso de Uso	9. Editar Hijo
Actores	Administrador Global Administrador Dependencias
Descripción	El objetivo de editar hijo es poder cambiar los diferentes datos de un hijo de una partición tales como: nombre, descripción, Layout y estado de una publicación.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para editar a un hijo de una dependencia, deberá seleccionar la opción del menú Particiones Dependencias. 3. Luego se mostrara la lista de hijos de esa partición y el botón de editar siempre que se encuentra al lado del hijo. 4. Se desplegará un formulario con las opciones a editar.

Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. En caso de que no existan hijos de la dependencia se muestra un mensaje informativo de que la partición no tiene hijos y no se encontrara el botón editar.
Pre condiciones	El usuario presiono el botón de editar en el hijo que requiere hacer las modificaciones.
Post condiciones	El usuario realizó el cambio en los atributos del hijo necesarios.

Tabla 10 - Caso de Uso: Editar Hijo.

Caso de Uso	10. Crear Dependencias
Actores	Administrador Global Administrador Dependencias
Descripción	Crear dependencias hermanas o hijas de la partición actual en la que se encuentre el usuario autenticado.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para crear una dependencia hijo o hermano, deberá seleccionar la opción del menú Particiones Dependencias. 3. Luego al lado derecho se muestra un formulario para la creación de una dependencia en donde se requiere la siguiente información: nombre, descripción, URL, dependencia, layout y estado de publicación.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. En caso de que la dependencia se encuentre creada mostrara un mensaje de error, ya que no se puede crear una misma dependencia dos veces.
Pre condiciones	El usuario se encuentra autenticado y seleccionó la opción de Particiones Dependencias.
Post condiciones	El usuario completo el formulario de creación de dependencias y puede observar los cambios en el listado.

Tabla 11 - Caso de Uso: Crear Dependencias.

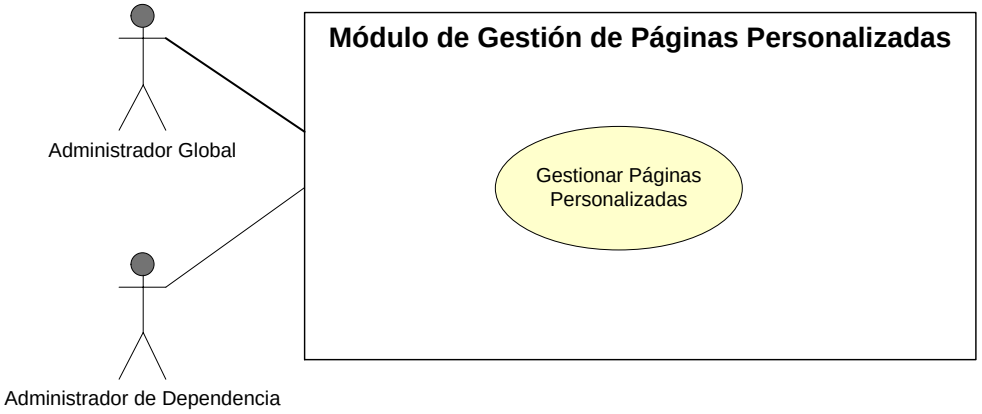
Caso de Uso	11. Editar Dependencias
Actores	Administrador Global Administrador Dependencias
Descripción	Editar la dependencia en la que se encuentra actualmente el usuario autenticado.

Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para editar la dependencia actual, deberá seleccionar la opción del menú Particiones Dependencias. 3. Luego seleccionar el botón de editar que se encuentra justo al lado del título Partición Actual. 4. Se le desplegara un formulario en el cual puede modificar los datos que desee el usuario.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado en la aplicación y presiono el botón de edición de dependencia actual.
Post condiciones	El usuario realizó los cambios de la dependencia actual.

Tabla 12 - Caso de Uso: Editar Dependencias.

- **Caso de Uso Modulo de Gestión de Paginas Personalizadas.**

Nivel 0



Nivel 1

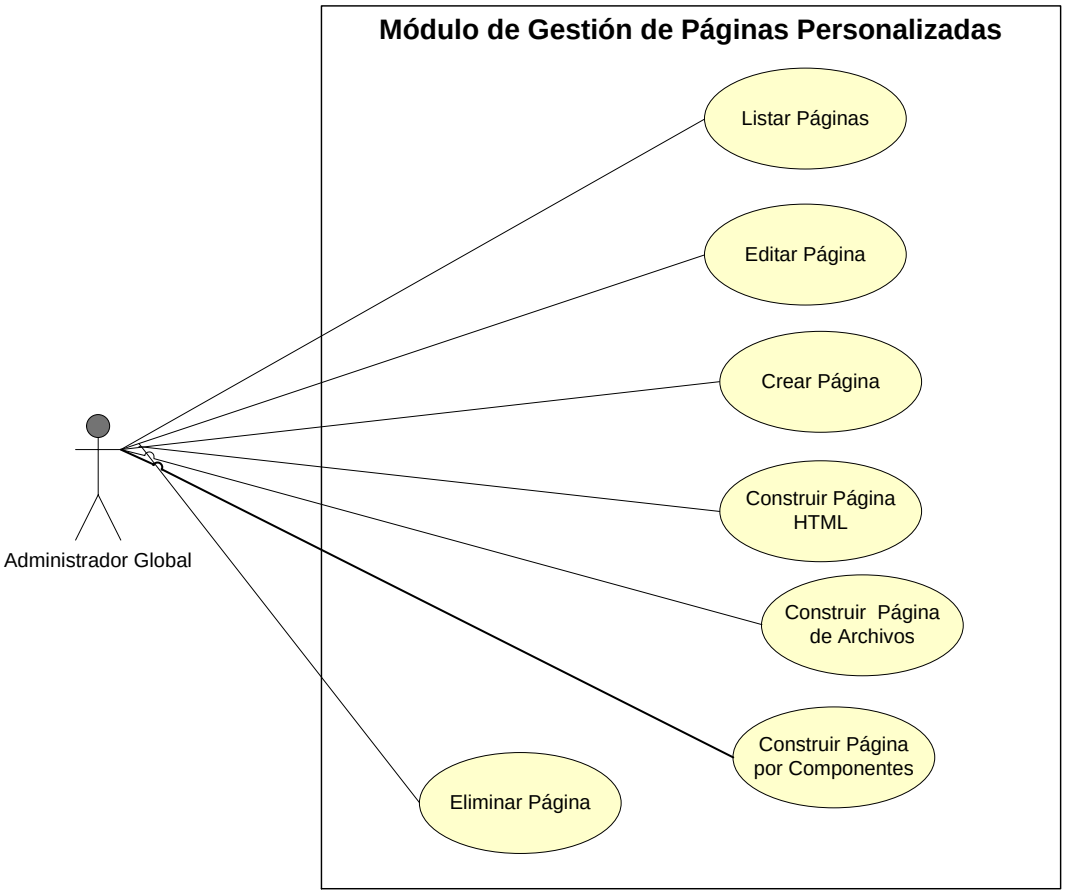


Figura 22 - Caso de Uso Gestión de Páginas Personalizadas

- **Descripción del Modulo de Gestión de Páginas Personalizadas.**

Caso de Uso	12. Gestionar Páginas Personalizadas.
Actores	Administrador Global Administrador Dependencias
Descripción	El modulo de gestión de paginas personalizadas, brinda la posibilidad al usuario de crear paginas ajustadas a las necesidades que se tengan, de tal manera que permite seleccionar entre las opciones de diagramación (Clásica, Secuencial o 3 Columnas), como también crear pagina de listado de archivos y para los usuarios más exigentes permite crear paginas en formato HTML asistido por una interfaz de ofimática en línea, la cual puede cambiarse con un botón por opción de codificación HTML. Adicional a esto el usuario puede editar las paginas, eliminarlas y construir el contenido de la misma jugando con componentes creados en el sistema y ordenándolos según su preferencia.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar las páginas personalizadas deberá seleccionar la opción del menú Páginas Personalizadas. 3. Luego se mostrara el listado de particiones creadas las cuales puede editar, seguir construyendo o eliminar, también existe la posibilidad de crear una nueva página personalizada.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario presiono el botón del menú Paginas Personalizadas.
Post condiciones	El usuario realizó la gestión dentro del módulo de las páginas personalizadas de la aplicación.

Tabla 12 - Caso de Uso: Gestionar Páginas Personalizadas.

Caso de Uso	13. Listar Páginas.
Actores	Administrador Global Administrador Dependencias
Descripción	Muestra el listado de páginas personalizadas creadas y cierta información tal como lo son: id, nombre, tipo, descripción, diagramación, elementos y acciones.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar las páginas personalizadas ya creadas deberá seleccionar la opción del menú Páginas Personalizadas. 3. Luego se mostrara el listado de páginas que se encuentran creadas con su breve detalle.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. En el caso de que no existan páginas personalizadas creadas se mostrara un mensaje informativo indicando que no existen páginas personalizadas para ese momento.
Pre condiciones	El usuario debe estar autenticado en la aplicación y debe seleccionar el botón de Páginas Personalizadas.
Post condiciones	El usuario pudo observar el listado de páginas personalizadas creadas y sus breves detalles de cada una.

Tabla 13 - Caso de Uso: Listar Páginas Personalizadas.

Caso de Uso	14. Editar Página.
Actores	Administrador Global Administrador Dependencias
Descripción	Editar una página consiste en poder cambiar el titulo de la página y la descripción.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para editar una página personalizadas ya creadas deberá seleccionar la opción del menú Páginas Personalizadas. 3. Luego se mostrara el listado de páginas que se encuentran creadas con su breve detalle y en la columna de acciones se encuentra el botón en forma de lápiz, el cual al presionarlo llevara a un formulario en el cual se puede editar el titulo de la página personalizada y la descripción.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.

Pre condiciones	El usuario debe encontrarse autenticado en la aplicación y haber seleccionado el botón de Páginas Personalizadas, para luego en el listado en la columna de acciones seleccionar el botón en forma de lápiz.
Post condiciones	El usuario realizó la edición de los atributos de la página que requería.

Tabla 14 - Caso de Uso: Editar Páginas Personalizadas.

Caso de Uso	15. Crear página.
Actores	Administrador Global Administrador Dependencias
Descripción	Consiste en poder crear una página la cual puede se diagramada de diferentes formas tales como: HTML, secuencial, tres columnas, clásica y archivo.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para crear una página personalizada deberá seleccionar la opción del menú Páginas Personalizadas. 3. Luego puede seleccionar el botón Crear Página Nueva y se desplegara un formulario que requiere el nombre, la descripción y el tipo de página a crear.
Flujos Alternos	2. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario se encuentra autenticado en la aplicación y presiono el botón de Paginas Personalizadas y luego presiono el botón de Crear Página Nueva.
Post condiciones	El usuario realizó la creación de la página y puede observarla en el listado.

Tabla 15 - Caso de Uso: Crear Páginas Personalizadas.

Caso de Uso	16. Construir página HTML.
Actores	Administrador Global Administrador Dependencias
Descripción	El objetivo de construir una página HTML es diagramar a través de una herramienta sencilla de edición el formato de la página y también se tiene la opción de colocar código HTML para usuarios con un poco más de experiencia.
Flujo Básico	4. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 5. Se muestra el menú principal y para construir una página HTML se deberá seleccionar la opción del menú Páginas Personalizadas. 6. Después se mostrara un listado de las páginas personalizadas, y para construir una página HTML se debe seleccionar solo aquellas páginas que en la columna Diagramación tengan el

	detalle Diagramación HTML. 7. Para construir se debe seleccionar el botón de herramienta para poder construir la página HTML bien sea por código o por diseño.
Flujos Alternos	3. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado en la aplicación y seleccionar la opción páginas personalizadas. Además, debió previamente haber creado una página personalizadas de tipo HTML.
Post condiciones	El usuario pudo construir la página HTML y podrá observarse en el mismo editor y para ver el resultado final debe navegarse hasta la dirección de creación.

Tabla 16 - Caso de Uso: Construir Páginas Personalizadas HTML.

Caso de Uso	17. Construir página de Archivos.
Actores	Administrador Global Administrador Dependencias
Descripción	Construir página de archivos consiste en crear una página personalizada en la que puedas cargar diferentes archivos, para que el cliente pueda descargarlos a través de la aplicación.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para construir una página de archivos se deberá seleccionar la opción del menú Páginas Personalizadas. 3. Después se mostrara un listado de las páginas personalizadas, y para construir una página de archivos se debe seleccionar solo aquellas páginas que en la columna Diagramación tengan el detalle Lista de Archivos. 4. Para construir se debe seleccionar el botón de herramienta para poder construir la página de archivos.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. No se puede cargar un archivo con un peso mayor a 5Mb.
Pre condiciones	El usuario debe estar autenticado en la aplicación y seleccionar la opción páginas personalizadas. Además, debió previamente haber creado una página personalizadas de tipo página de archivos.
Post condiciones	El usuario pudo construir la página de archivos y para poder descargar los archivos se debe navegar hasta la ruta en la que se encuentre, también tiene la opción de ver los que se encuentran cargados debajo del formulario de carga de archivos.

Tabla 17 - Caso de Uso: Construir Páginas Personalizadas de Archivos.

Caso de Uso	18. Construir página por componente.
Actores	Administrador Global Administrador Dependencias
Descripción	La opción de construcción de pagina tiene por defecto tres tipos de diagramación disponible (Clásica, Tres Columnas y Secuencial) las cuales pueden ser alimentadas por los componentes que han sido creados con anterioridad en la dependencia en cuestión. En la opción de construcción se puede agregar el componente, colocarle el nombre, la cantidad de elementos que va a mostrar, y ordenarlos según las necesidades.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para construir una página de de componente se deberá seleccionar la opción del menú Páginas Personalizadas. 3. Después se mostrara un listado de las páginas personalizadas, y para construir una página de componentes se debe seleccionar solo aquellas páginas que en la columna Diagramación tengan el detalle de Diagramación 3 columnas, Diagramación Secuencial y Diagramación clásica. 4. Para construir se debe seleccionar el botón de herramienta, el cual desplegara la forma de la diagramación que se selecciono y la edición de los componentes creados con anterioridad.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado en la aplicación y seleccionar la opción páginas personalizadas. Además, debió previamente haber creado una página personalizadas de tipo secuencial, clásico o tres columnas y haber creado al menos uno o más componentes en el módulo de gestión de componentes que incluirá en alguna de las diagramaciones mencionadas anteriormente.
Post condiciones	El usuario pudo construir la página por componente y puede ver cómo va construyéndose y luego el diseño final a través de la navegación hacia en donde se construyo la página.

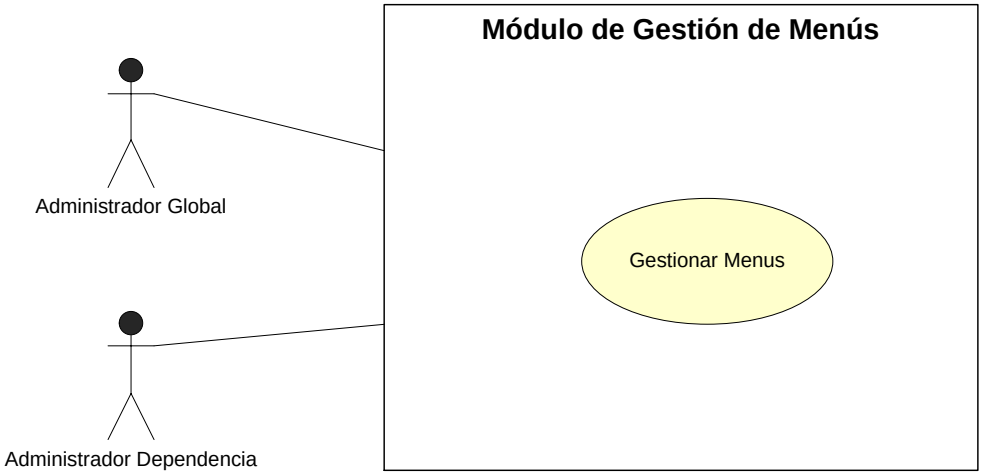
Tabla 18 - Caso de Uso: Construir Páginas Personalizadas por Componentes.

Caso de Uso	19. Eliminar página.
Actores	Administrador Global Administrador Dependencias
Descripción	Si se quiere eliminar alguna página personalizada creada se puede realizar por medio de esta opción.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para eliminar una página personalizada se deberá seleccionar la opción del menú Páginas Personalizadas. 3. Después se mostrara un listado de las páginas personalizadas, y para eliminar una página en la columna de acciones se debe presionar el botón en forma de equis.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. Si una página personalizada contiene componentes internamente, primero deben eliminarse antes de eliminar la página ya que si no mostrara un mensaje de error.
Pre condiciones	El usuario se encuentra autenticado y selecciono la opción Paginas Personalizadas y luego presiono el botón en forma de equis que se encuentra en la columna de acciones del listado de páginas personalizadas.
Post condiciones	El usuario pudo eliminar la página que selecciono y que no tenía contenido relacionado.

Tabla 19 - Caso de Uso: Eliminar Páginas Personalizadas.

- **Caso de Uso Modulo de Gestión de Menús.**

Nivel 0



Nivel 1

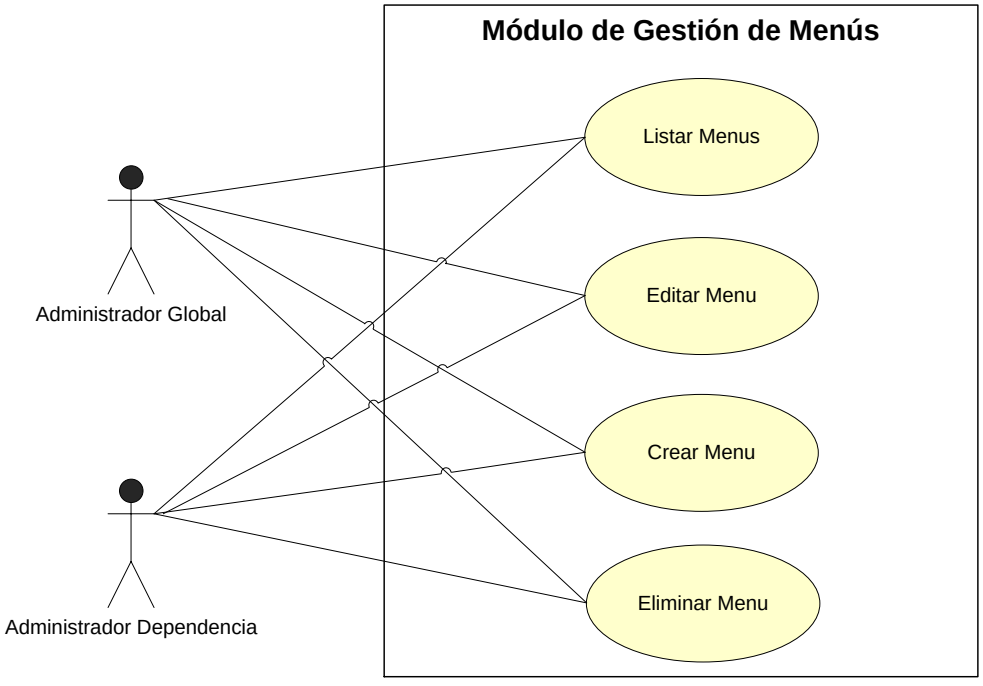


Figura 23 - Caso de Uso Gestión de Menús

- **Descripción del Modulo de Gestión de Menús.**

Caso de Uso	20. Gestionar Menús.
Actores	Administrador Global Administrador Dependencias
Descripción	El objetivo de gestionar menús consiste es brindarle la posibilidad al usuario de crear, editar o eliminar los diferentes menús que puede utilizar en las páginas de GENCI-2.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar menús deberá seleccionar la opción del menú Manejador de Menú. 3. Después se desplegara el listado de menús creados y se tendrá la opción de crear un nuevo menú, editar o eliminar algún menú existente.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario se encuentra autenticado en la aplicación y selecciono en el menú principal la opción de Manejador de Menú.
Post condiciones	El usuario realizó la gestión de menús que requería, bien sea listar, editar o eliminar.

Tabla 20 - Caso de Uso: Gestionar Menús.

Caso de Uso	21. Listar Menús.
Actores	Administrador Global Administrador Dependencias
Descripción	Se podrá listar todos los menús creados y se puede observar un pequeño detalle tal y como lo son: id, nombre, tipo, descripción, partición, cantidad de botones, estado de publicación y las acciones que se pueden realizar con dicho menú.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar menús deberá seleccionar la opción del menú Manejador de Menú. 3. Se desplegara el listado de menús creados.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario se encuentra autenticado en la aplicación y seleccionó la opción de Manejador de Menú.
Post condiciones	El usuario pudo observar el listado de los diferentes menús creados.

Tabla 21 - Caso de Uso: Listar Menús.

Caso de Uso	22. Editar Menús.
Actores	Administrador Global Administrador Dependencias
Descripción	Una vez creado un menú este puede ser editado en la cantidad de botones, descripción, tipo.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar menús deberá seleccionar la opción del menú Manejador de Menú. 3. Se desplegara el listado de menús creados y para poder editar se debe seleccionar el botón en forma de lápiz que llevara a un formulario en el cual se puede editar el nombre, descripción, tipo de menú y estado de la publicación.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. Si ya se encuentra un menú horizontal creado y se desea crear otro el sistema muestra un mensaje de dialogo en donde al aceptar el menú creado sobrescribirá al creado anteriormente, en el caso de no aceptar se cambiara el menú a una forma vertical.
Pre condiciones	El usuario se encuentra autenticado en la aplicación y debió haber creado al menos un menú para poder editarlo. Además de seleccionar el botón en forma de lápiz que se encuentra en la columna de acciones del listado de menús.
Post condiciones	El usuario pudo editar el menú que selecciono.

Tabla 22 - Caso de Uso: Editar Menús.

Caso de Uso	23. Crear Menús.
Actores	Administrador Global Administrador Dependencias
Descripción	El objetivo de crear menús es mostrar un contenedor en el cual se incluyen los enlaces internos y externos de navegación a través de botones.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para crear menús deberá seleccionar la opción del menú Manejador de Menú. 3. Luego se mostrara un botón que dice Crear Nuevo Menú, el cual llevara a un formulario que se debe completar. 4. Los datos a completar en el formulario de creación son: nombre, descripción, tipo de menú, estado de publicación de menú y la página en la que se visualizara el menú a crear.

Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. Si ya se encuentra un menú horizontal creado y se desea crear otro el sistema muestra un mensaje de dialogo en donde al aceptar el menú creado sobrescribirá al creado anteriormente, en el caso de no aceptar se cambiara el menú a una forma vertical.
Pre condiciones	El usuario debe estar autenticado y debe estar creada la página personalizada en la que se quiere mostrar un nuevo menú.
Post condiciones	Se puede observar el cambio navegando hasta la página en la que se creó el menú.

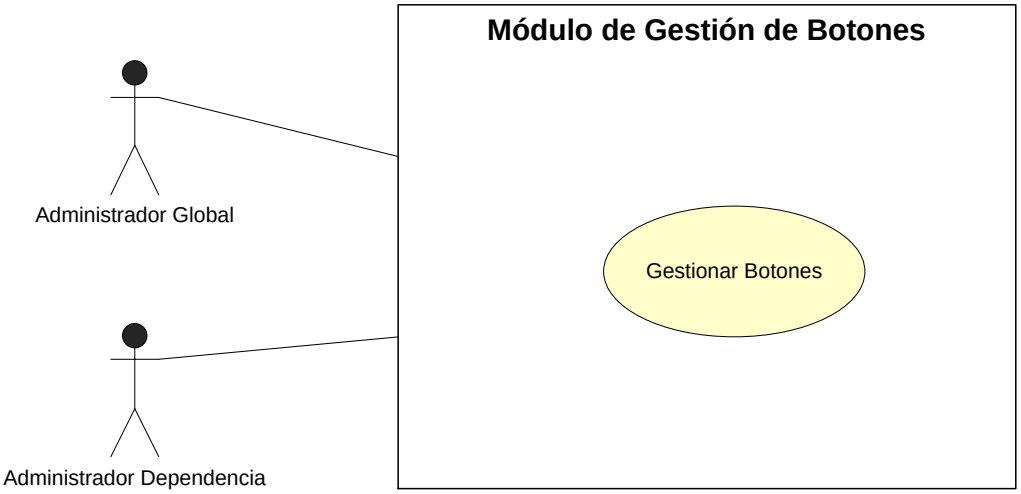
Tabla 23 - Caso de Uso: Crear Menús.

Caso de Uso	24. Eliminar Menús.
Actores	Administrador Global Administrador Dependencias
Descripción	Si no se quiere tener en el listado algún menú que haya sido creado se puede eliminar a través de la opción eliminar menús.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para eliminar menús deberá seleccionar la opción del menú Manejador de Menú. 3. Luego se desplegara un listado de menús creados el cual contiene en la columna de acciones un botón en forma de equis que permitirá la eliminación del menú seleccionado.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado y debe existir al menos un menú en el listado de menús creados.
Post condiciones	Se puede observar el cambio navegando hasta la página en la que se elimino el menú.

Tabla 24 - Caso de Uso: Eliminar Menús.

- **Caso de Uso Modulo de Gestión de Botones.**

Nivel 0



Nivel 1

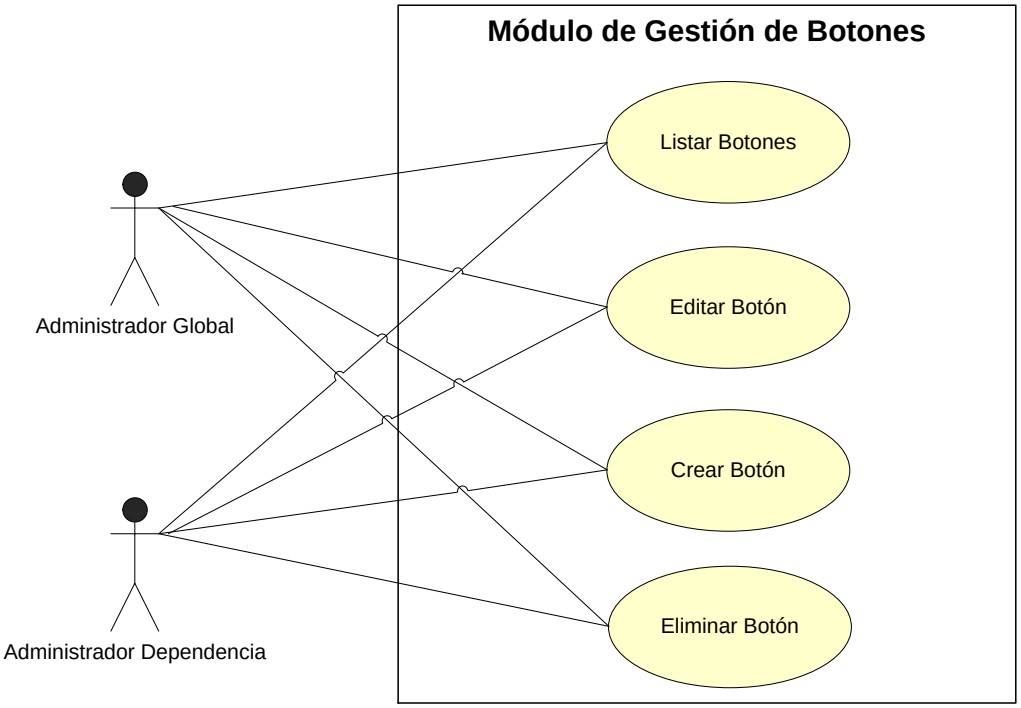


Figura 24 - Caso de Uso Gestión de Botones

- **Descripción del Modulo de Gestión de Botones.**

Caso de Uso	25. Gestionar Botones.
Actores	Administrador Global Administrador Dependencias
Descripción	Gestionar botones permite al usuario crear, editar o eliminar un botón que se encuentre en un menú.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar botones deberá seleccionar la opción del menú Manejador de Menú. 3. Una vez que se listen todos los menús creados se selecciona el botón de herramienta de la columna de acciones. 4. Luego se expondrán las opciones para crear un botón, editarlo o eliminarlo.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. Debe encontrarse creado al menos un menú al que agregarle botones, sino no se dejara crear botones.
Pre condiciones	El usuario esta autenticado y debe haber creado al menos un menú para agregarle botones.
Post condiciones	El usuario pudo autenticarse en el sistema y podrá asignarle botones a los menús.

Tabla 25 - Caso de Uso: Gestionar Botones.

Caso de Uso	26. Listar Botones.
Actores	Administrador Global Administrador Dependencias
Descripción	Se pueden observar el listado de botones que se encuentran en el menú creado previamente y seleccionado.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar los botones deberá seleccionar la opción del menú Manejador de Menú. 3. Una vez que se listen todos los menús creados se selecciona el botón de herramienta de la columna de acciones. 4. Luego se mostrara la lista de botones creados.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. Debe encontrarse creado al menos un menú que contenga botones para observar el listado de botones.

Pre condiciones	El usuario esta autenticado y debe haber creado al menos un menú con botones para poder ver el listado de botones.
Post condiciones	El usuario pudo autenticarse en el sistema y podrá ver la lista de botones por menú.

Tabla 26 - Caso de Uso: Listar Botones.

Caso de Uso	27. Editar Botones.
Actores	Administrador Global Administrador Dependencias
Descripción	Editar botones tiene la finalidad de poder editar diferentes atributos que tiene un botón creado.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para editar un botón deberá seleccionar la opción del menú Manejador de Menú. 3. Una vez que se listen todos los menús creados se selecciona el botón de herramienta de la columna de acciones. 4. Luego se mostrara la lista de botones creados y en la columna de acciones se encuentra un botón en forma de lápiz el cual direccionara a la edición del botón. 5. Los atributos que pueden ser editados son: nombre, mostrar o no la imagen en botón, el orden en el menú, el estado de publicación, el tipo de enlace y con cual pagina interna quiere realizar el enlace.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario esta autenticado y debe haber creado al menos un menú con botones para poder editar alguno de ellos.
Post condiciones	El usuario pudo autenticarse en el sistema y podrá editar el botón del menú que selecciono.

Tabla 27 - Caso de Uso: Editar Botones.

Caso de Uso	28. Crear Botones.
Actores	Administrador Global Administrador Dependencias
Descripción	Se puede crear botones a un menú previamente creado permitiéndole al usuario realizar enlaces internos o externos en la aplicación GENCI-2.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para editar un botón deberá seleccionar la opción del menú Manejador de Menú. 3. Una vez que se listen todos los menús creados se selecciona el botón de herramienta de la columna de acciones. 4. Luego se mostrara un botón llamado Crear Nuevo Botón. 5. Una vez seleccionado se desplegara un formulario en el cual se puede crear el botón seleccionando el icono del botón, el nombre, el orden, el tipo de enlace.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario esta autenticado y debe haber creado al menos un menú.
Post condiciones	El usuario pudo autenticarse en el sistema y podrá crear el botón en el menú creado previamente.

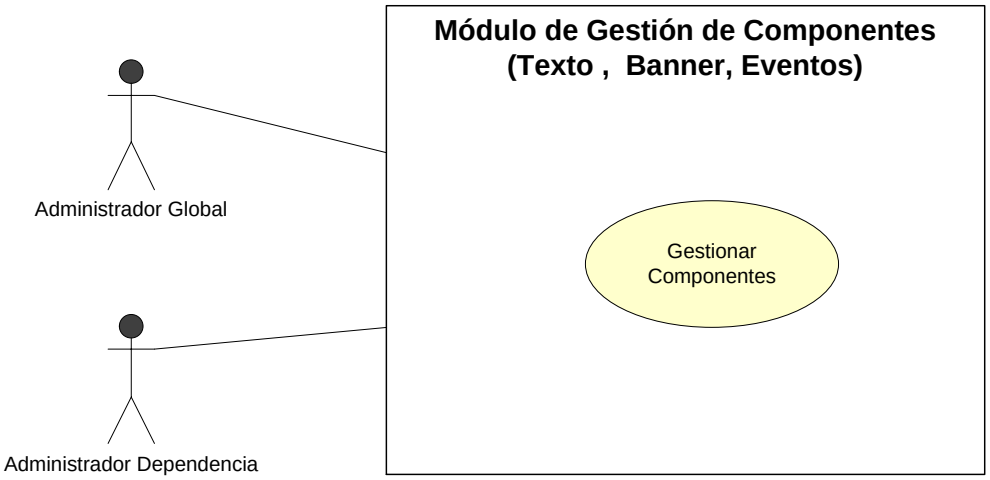
Tabla 28 - Caso de Uso: Crear Botones.

Caso de Uso	29. Eliminar Botones.
Actores	Administrador Global Administrador Dependencias
Descripción	Se puede eliminar un botón de un menú previamente creado.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para eliminar un botón deberá seleccionar la opción del menú Manejador de Menú. 3. Una vez que se listen todos los menús creados se selecciona el botón con forma de equis de la columna de acciones, el cual debe presionarse para eliminar un botón.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario esta autenticado y debe haber creado al menos un botón en un menú.
Post condiciones	El usuario pudo autenticarse en el sistema y podrá eliminar el botón en el menú creado previamente.

Tabla 29 - Caso de Uso: Eliminar Botones.

- **Caso de Uso Modulo de Gestión de Componentes.**

Nivel 0



Nivel 1

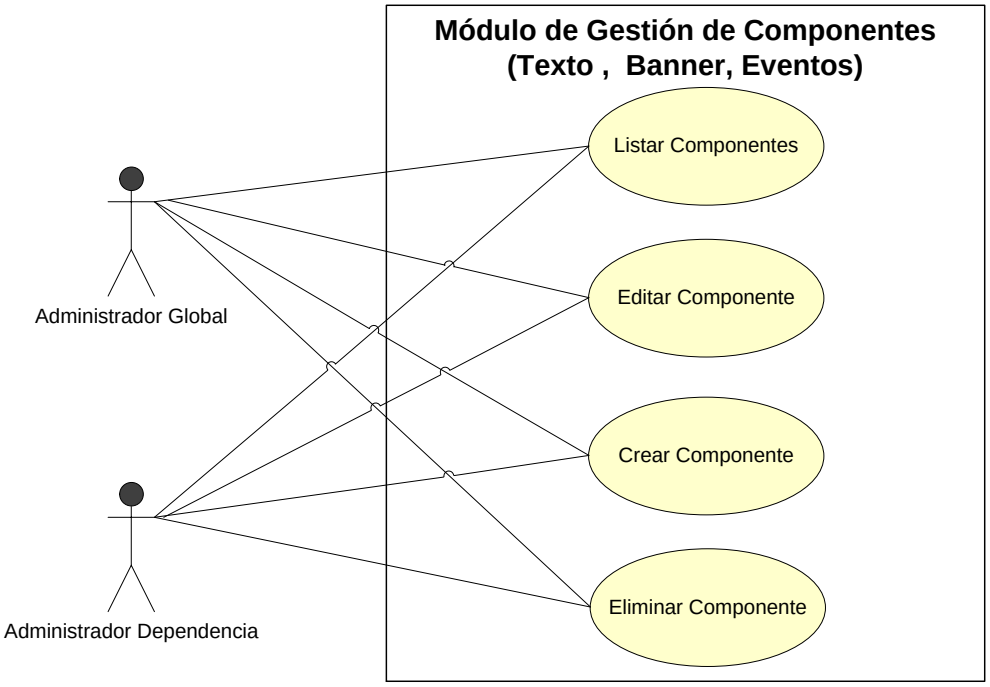


Figura 25 - Caso de Uso Gestión de Componentes.

- **Descripción del Modulo de Gestión de Componentes (Texto, Banner, Evento).**

Caso de Uso	30. Gestionar Componentes.
Actores	Administrador Global Administrador Dependencias
Descripción	La administración de componentes puede ser para diferentes tipos de elementos como de Texto, Banner y Eventos. La gestión se refiere a crear, editar y eliminar componentes que funcionan como repositorio de información los cuales posteriormente pueden ser utilizados en la creación de páginas personalizadas.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar componentes deberá seleccionar alguna de las siguientes opciones: Componentes Tipo "Texto", Componentes Tipo "Evento" y Componentes Tipo "Banner". 3. En los cuales se podrá crear, editar o eliminar componente de acuerdo a las necesidades del usuario.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado haber seleccionado del menú alguna opción de componentes.
Post condiciones	El usuario pudo gestionar el componente del tipo que desee.

Tabla 30 - Caso de Uso: Gestionar Componentes.

Caso de Uso	31. Listar Componentes.
Actores	Administrador Global Administrador Dependencias
Descripción	Según el componente seleccionado se pueden observar todos los componentes creados.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar los componentes deberá seleccionar la alguna de las opciones del menú: Componentes Tipo "Texto", Componentes Tipo "Evento" y Componentes Tipo "Banner". 3. Una vez que seleccione alguna de las opciones anteriores se listara de acuerdo el tipo los componentes creados.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe encontrarse autenticado y haber seleccionado en el menú principal alguna opción de Componentes.

Post condiciones	El usuario pudo ver la lista de los componentes según el tipo seleccionado.

Tabla 31 - Caso de Uso: Listar Componentes.

Caso de Uso	32. Crear Componentes.
Actores	Administrador Global Administrador Dependencias
Descripción	Crear un componente permitirá almacenar información en un contenedor que podrá ser expuesto en la página personalizada que se quiera.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para crear componentes deberá seleccionar alguna de las siguientes opciones: Componentes Tipo “Texto”, Componentes Tipo “Evento” y Componentes Tipo “Banner”. 3. Para crear el componente se selecciona el botón Crear Nuevo Componente, para luego completar el formulario de creación en el que se deben completar los siguientes datos: nombre, descripción y estado de la publicación del componente.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario se encuentra autenticado y selecciono en el menú principal alguna opción de tipo componente en la cual creara uno nuevo.
Post condiciones	El usuario pudo agregar el componente creado en la página personalizada que necesite.

Tabla 32 - Caso de Uso: Crear Componentes.

Caso de Uso	33. Editar Componente.
Actores	Administrador Global Administrador Dependencias
Descripción	Una vez creado un componente se puede editar el contenido, nombre y estado de la publicación.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para editar componentes deberá seleccionar alguna de las siguientes opciones: Componentes Tipo “Texto”, Componentes Tipo “Evento” y Componentes Tipo “Banner”. 3. Una vez que se listen todos los componentes creados se selecciona el botón de lápiz de la columna de acciones, que permitirá la edición del componente seleccionado.

Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario esta autenticado y creo previamente algún tipo de componente.
Post condiciones	Se pudo editar el contenido de los componentes que se encuentran en las páginas personalizadas.

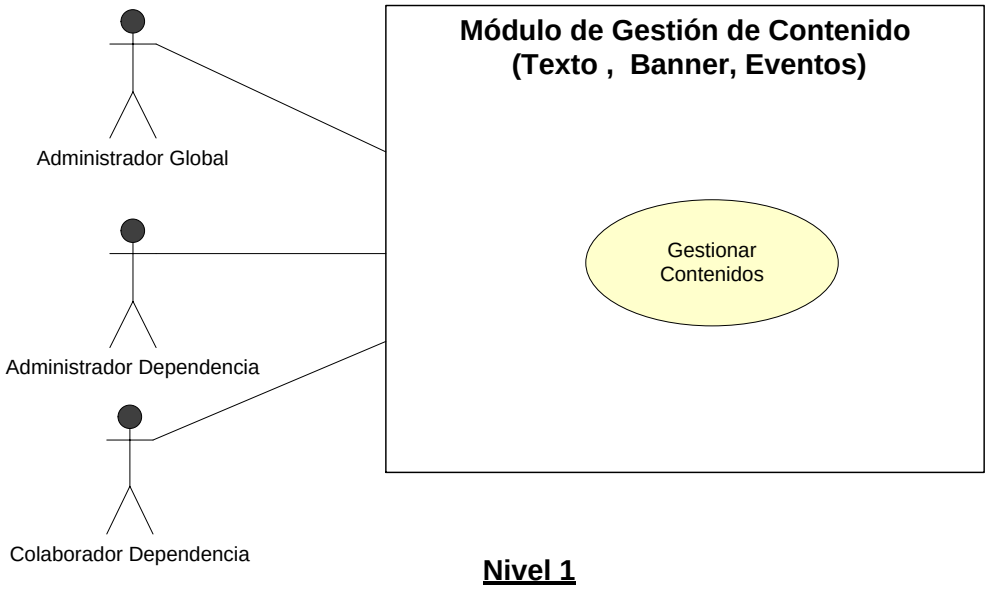
Tabla 33 - Caso de Uso: Editar Componentes.

Caso de Uso	34. Eliminar Componente.
Actores	Administrador Global Administrador Dependencias
Descripción	Se puede eliminar un componente que no se encuentre en una página personalizada.
Flujo Básico	1. El usuario administrador global o de dependencias debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para eliminar componentes deberá seleccionar alguna de las siguientes opciones: Componentes Tipo “Texto”, Componentes Tipo “Evento” y Componentes Tipo “Banner”. 3. Una vez que se listen todos los menús creados se selecciona el botón con forma de equis de la columna de acciones, el cual debe presionarse para eliminar un componente.
Flujos Alternos	1. El usuario administrador global o de dependencias no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial. 2. En el caso de que el componente este siendo utilizado en alguna página personalizada mostrara un mensaje de error al querer eliminarlo.
Pre condiciones	El usuario esta autenticado y el componente creado que se quiere eliminar no deben estar siendo utilizado en ninguna página personalizada.
Post condiciones	El usuario pudo eliminar un componente creado.

Tabla 34 - Caso de Uso: Eliminar Componentes.

- **Caso de Uso Modulo de Gestión de Contenido.**

Nivel 0



Nivel 1

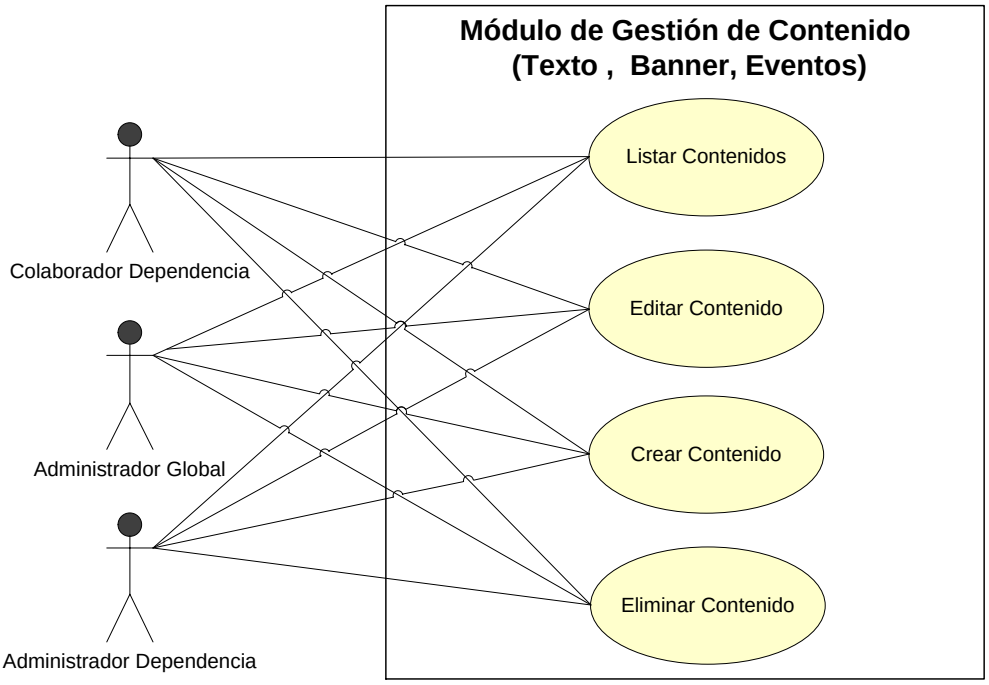


Figura 26 - Caso de Uso Gestión de Contenido

- **Descripción del Modulo de Gestión de Contenido (Texto, Banner, Evento).**

Caso de Uso	35. Gestionar Contenidos.
Actores	Administrador Global Administrador Dependencias Colaborador Dependencia
Descripción	La gestión de contenido permite agregar, editar y eliminar contenido a un modulo de componente.
Flujo Básico	1. El usuario administrador global, de dependencias o el colaborador dependencia debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para gestionar el contenido deberá seleccionar alguna de las siguientes opciones: Contenido Tipo "Artículos", Contenido Tipo "Evento" y Contenido Tipo "Banner". 3. En los cuales se podrá crear, editar o eliminar contenido de acuerdo a las necesidades del usuario.
Flujos Alternos	1. El usuario administrador global, de dependencias o colaborador de dependencia no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado y previamente debió crear componentes en los cuales se encontrara el contenido a gestionar.
Post condiciones	El usuario pudo autenticarse en el sistema y puede observar el contenido que se encuentra en los componentes de las páginas personalizadas.

Tabla 35 - Caso de Uso: Gestionar Contenidos.

Caso de Uso	36. Listar Contenidos.
Actores	Administrador Global Administrador Dependencias Colaborador Dependencia
Descripción	Se podrá listar los contenidos creados dependiendo del tipo que el usuario haya seleccionado.
Flujo Básico	1. El usuario administrador global, de dependencias o el colaborador dependencia debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para listar el contenido deberá seleccionar alguna de las siguientes opciones: Contenido Tipo "Artículos", Contenido Tipo "Evento" y Contenido Tipo "Banner". 3. Una vez seleccionada alguna de las opciones anteriores se mostrara un listado de los contenidos creados con una breve descripción tales como: id, titulo, componente en el que se encuentra, resumen, contenido, imagen, orden, publicado y la barra de acciones.
Flujos Alternos	1. El usuario administrador global, de dependencias o colaborador de dependencia no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado y previamente debió crear contenidos y componentes.
Post condiciones	El usuario pudo autenticarse en el sistema y pudo listar los contenidos creados.

Tabla 36 - Caso de Uso: Listar Contenidos.

Caso de Uso	37. Crear Contenidos.
Actores	Administrador Global Administrador Dependencias Colaborador Dependencia
Descripción	Cuando se quiera tener en los componentes algún contenido se deben crear en esta opción.
Flujo Básico	1. El usuario administrador global, de dependencias o colaborador de dependencia debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para crear el contenido deberá seleccionar alguna de las siguientes opciones: Contenido Tipo "Artículos", Contenido Tipo "Evento" y Contenido Tipo "Banner". 3. Una vez seleccionada alguna de las opciones anteriores se mostrara un botón llamado Crear Nuevo Contenido. 4. El cual desplegara un formulario con los siguientes campos necesarios: titulo, resumen, componente padre, imagen del artículo, orden de contenido, estado de publicación del artículo y el contenido del artículo con la herramienta que permite darle formato o escribir código HTML.
Flujos Alternos	1. El usuario administrador global, de dependencias o colaborador de dependencia no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado y previamente debió crear componentes en los que irán los contenidos creados.
Post condiciones	El usuario pudo autenticarse en el sistema y público el contenido creado en el componente de la página personalizada.

Tabla 37 - Caso de Uso: Crear Contenidos.

Caso de Uso	38. Editar Contenidos.
Actores	Administrador Global Administrador Dependencias Colaborador Dependencia
Descripción	Cuando se quiera tener en los componentes algún contenido se deben crear en esta opción.
Flujo Básico	1. El usuario administrador global, de dependencias o colaborador de dependencia debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para editar el contenido deberá seleccionar alguna de las siguientes opciones: Contenido Tipo "Artículos", Contenido Tipo "Evento" y Contenido Tipo "Banner". 3. Una vez seleccionada alguna de las opciones anteriores se mostrara el listado de contenidos creados. 4. Para poder editar el contenido se selecciona el botón en forma

	de lápiz que llevara al formulario en donde se podrá realizar la edición.
Flujos Alternos	1. El usuario administrador global, de dependencias o colaborador de dependencia no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado y previamente debió crear componentes y contenidos para poder editar el contenido.
Post condiciones	El usuario pudo autenticarse en el sistema y edito el contenido creado en el componente de la página personalizada.

Tabla 38 - Caso de Uso: Editar Contenidos.

Caso de Uso	39. Eliminar Contenidos.
Actores	Administrador Global Administrador Dependencias Colaborador Dependencia
Descripción	Cuando se ha creado un contenido que se quiera eliminar se utiliza esta opción.
Flujo Básico	1. El usuario administrador global, de dependencias o colaborador de dependencia debe autenticarse en la aplicación, con su respectivo usuario y contraseña y luego presionar el botón Login. 2. Se muestra el menú principal y para eliminar el contenido deberá seleccionar alguna de las siguientes opciones: Contenido Tipo “Artículos”, Contenido Tipo “Evento” y Contenido Tipo “Banner”. 3. Una vez seleccionada alguna de las opciones anteriores se mostrara el listado de contenidos creados. 4. Para poder eliminar el contenido se selecciona el botón en forma de equis que me permitirá eliminar el contenido de un componente.
Flujos Alternos	1. El usuario administrador global, de dependencias o colaborador de dependencia no se encuentra en el directorio LDAP o introdujo mal el usuario o contraseña y se le mostrara un error en la pantalla inicial.
Pre condiciones	El usuario debe estar autenticado y previamente debió crear contenido para poder eliminarlo.
Post condiciones	El usuario pudo autenticarse en el sistema y elimino el contenido creado.

Tabla 39 - Caso de Uso: Eliminar Contenidos.

3.10.2. Prototipos de Interfaz

En esta fase se definen los estilos, estructuras y se construyo en HTML, Jquery y CSS3 las vistas de la aplicación, tanto para el ambiente de administración como para las interfaces a usuarios públicos, y parte de las mismas se pueden ver a continuación en las figuras 27 y 28:

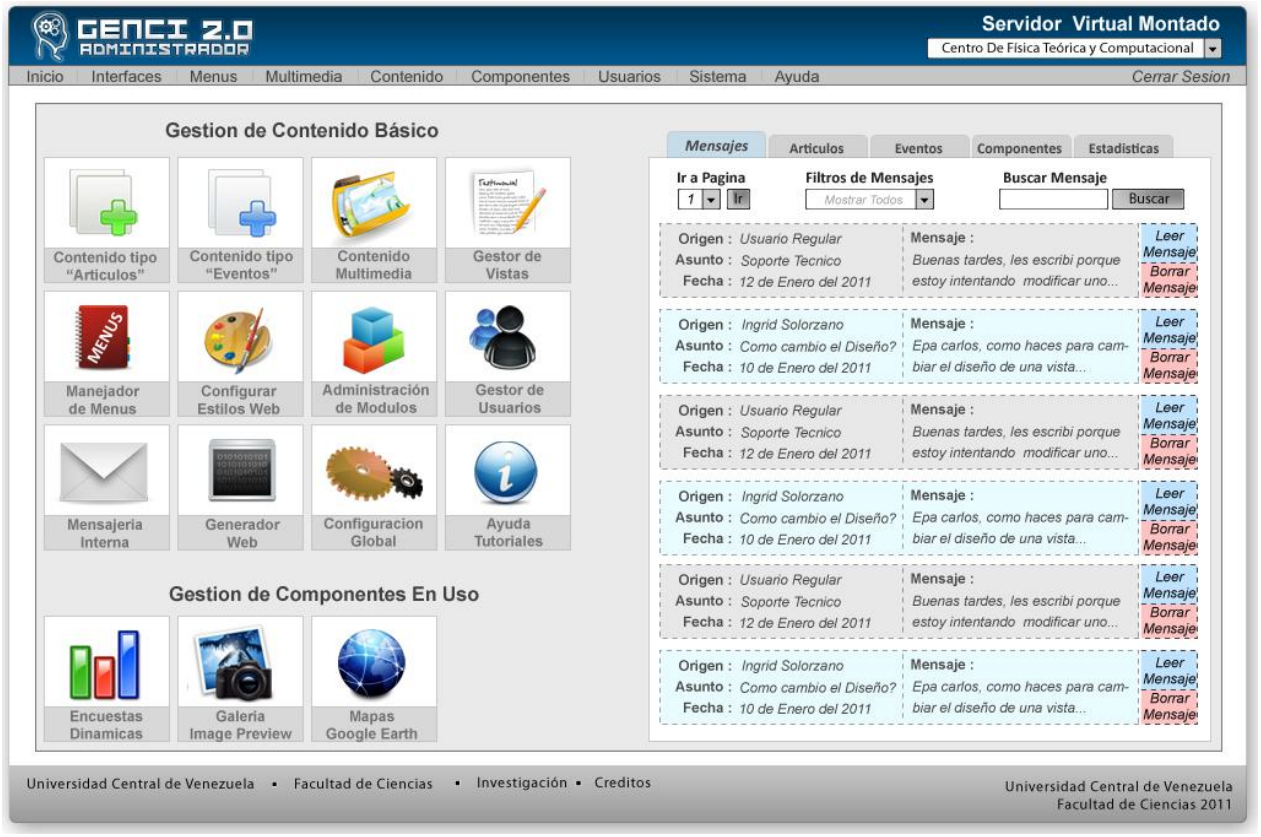


Figura 27 - Interfaz Cliente

Figura 28 - Interfaz Cliente

3.10.3. Plataforma de Desarrollo

Hardware (Características mínimas):

- Pentium 4 o equivalente.
- Memoria RAM 512Mb.
- Disco Duro 40Gb.
- Monitor SVGA.

Software:

- Ruby 1.9.2.
- Rails 3.0.3
- MySql 5.0.
- XHTML
- JavaScript 1.3.
- CSS 3.
- Webrick como servidor Web de desarrollo.
- Aptana Studio 3 como entorno de programación.
- Varios tipos de navegadores: Chrome, Mozilla, Opera, Explorer y Safari.

3.11. Construcción

La fase de construcción se realizo de manera modular, se fueron realizando las pruebas y ajustes necesarios a medida que se desarrollaron las funcionalidades del sistema, a medida que se iban integrando nuevos módulos se realizaron ajustes a los módulos listos para garantizar que el funcionamiento de cada uno de ellos, y no se afectara de ninguna forma el comportamiento en conjunto.

Durante el inicio del desarrollo se llevaron a cabo la elaboración y puesta en producción de las interfaces, tanto la de administración de contenido y construcción de componentes y paginas, como la de navegación del usuario público, y se fue probando de manera simultánea los resultados obtenidos.

Por necesidades académicas y laborales las iteraciones de construcción del trabajo especial de grado se realizaron en función a metas u objetivos en vez de hacerlos en función a tiempo, de esta manera me pude organizar mejor y exigirme según lo que esa iteración requería.

En la tabla 40 se puede observar el Cronograma de desarrollo de las iteraciones correspondientes a esta fase:

Iteración	Objetivos
1	• Diseño de Modelo de Datos. • Diseño de Casos de Uso. • Modelado de Base de Datos. • Configuración de ambiente de desarrollo. • Diseño y desarrollo de interfaces estáticas. • Pruebas de despliegue con datos extraídos de la Base de Datos
2	• Configuración y desarrollo de método de autenticación mediante la gema Authologic. • Configuración de archivo "route" y manejo de parámetros por URL. • Desarrollo de modulo de usuarios. • Desarrollo de manejo de sesiones y filtros. • Desarrollo de modulo de gestión de Particiones. • Desarrollo de lógica de estructura de árbol N-ario, recorridos y modelado. • Pruebas de módulos desarrollados.
3	• Desarrollo de modulo de Gestión de Menús. • Desarrollo de modulo de Gestión de Botones. • Pruebas de módulos desarrollados
4	• Desarrollo de modulo de gestión de pagina personalizada (Fase 1) Diagramación Clásica. • Desarrollo de modulo de gestor de componente texto. • Desarrollo de modulo de gestor de contenido de texto. • Pruebas de módulos desarrollados.
5	• Desarrollo de modulo de gestión de pagina personalizada (Fase final) Diagramación clásica. • Desarrollo de modulo de gestión de pagina personalizada (Fase final) Diagramación Secuencial. • Desarrollo de modulo de gestión de pagina personalizada (Fase final) Diagramación 3 Columnas. • Depuración de modulo de gestión de componentes de texto. • Depuración de modulo de gestión de contenido de texto. • Desarrollo de modulo de gestión de componentes de eventos. • Desarrollo de modulo de gestión de contenido tipo eventos. • Desarrollo de modulo de gestión de componente banner. • Desarrollo de modulo de gestión de contenido tipo banner. • Ajustes de modulo de gestión de pagina personalizada. • Depuración de modulo de gestión de menús. • Depuración de modulo de gestión de botones. • Pruebas de módulos desarrollados. • Presentación parcial de aplicación a cliente y tutor.
6	• Desarrollo de modulo de gestión de pagina personalizada Pagina HTML. • Desarrollo de modulo de gestión de pagina personalizada Pagina de Archivos. • Ajustes al modulo de usuarios, creando diferentes roles y perisologías.

	• Desarrollo de lógica de recorrido en árbol N-ario para perisologías. • Desarrollo de módulos de control de particiones para súper-usuarios. • Pruebas de módulos desarrollados. • Presentación parcial de aplicación a tutor.
7	• Desarrollo de log de operaciones. • Desarrollo de validaciones en formularios y Bases de Datos. • Depuración y mejoras en interfaz, estilos y carga de datos. • Depuración de errores de la aplicación. • Carga de datos de prueba y para la puesta en producción. • Pruebas de módulos desarrollados. • Prueba de usuarios. • Presentación parcial de aplicación a tutor y jurado.
8	• Desarrollo de modulo de autenticación LDAP. • Ajustes en el modulo de usuarios. • Ajustes en permisos, sesiones y manejo de usuarios. • Puesta en producción de la aplicación en maquina virtual del centro de computación. • Carga de video tutoriales. • Prueba Funcional en Ambiente de Producción. • Realización de encuestas a usuarios.

Tabla 40 - Cronograma de desarrollo

3.11.1. Iteración 1

Objetivos Planteados.

- Diseño de Modelo de Datos.
- Diseño de Casos de Uso.
- Modelado de Base de Datos.
- Configuración de ambiente de desarrollo.
- Diseño y desarrollo de interfaces estáticas.
- Pruebas de despliegue con datos extraídos de la Base de Datos

Se inició con la primera iteración analizando con más detalle las necesidades y requerimientos planteados en la propuesta de trabajo especial de grado, y las modificaciones solicitadas durante la presentación, por lo que se llevo a papel las necesidades iniciales para poder visualizar la magnitud de la necesidad requerida y poder buscar una solución que se ajuste.

Se diseñó un modelo de datos inicial el cual soportaba las necesidades iniciales, sin embargo debido al desarrollo que se llevo a cabo el modelo sufrió 7 versiones hasta llegar a la final la cual es la expuesta en el “Modelo de Datos”, la primera versión se veía como la muestra la figura 29.

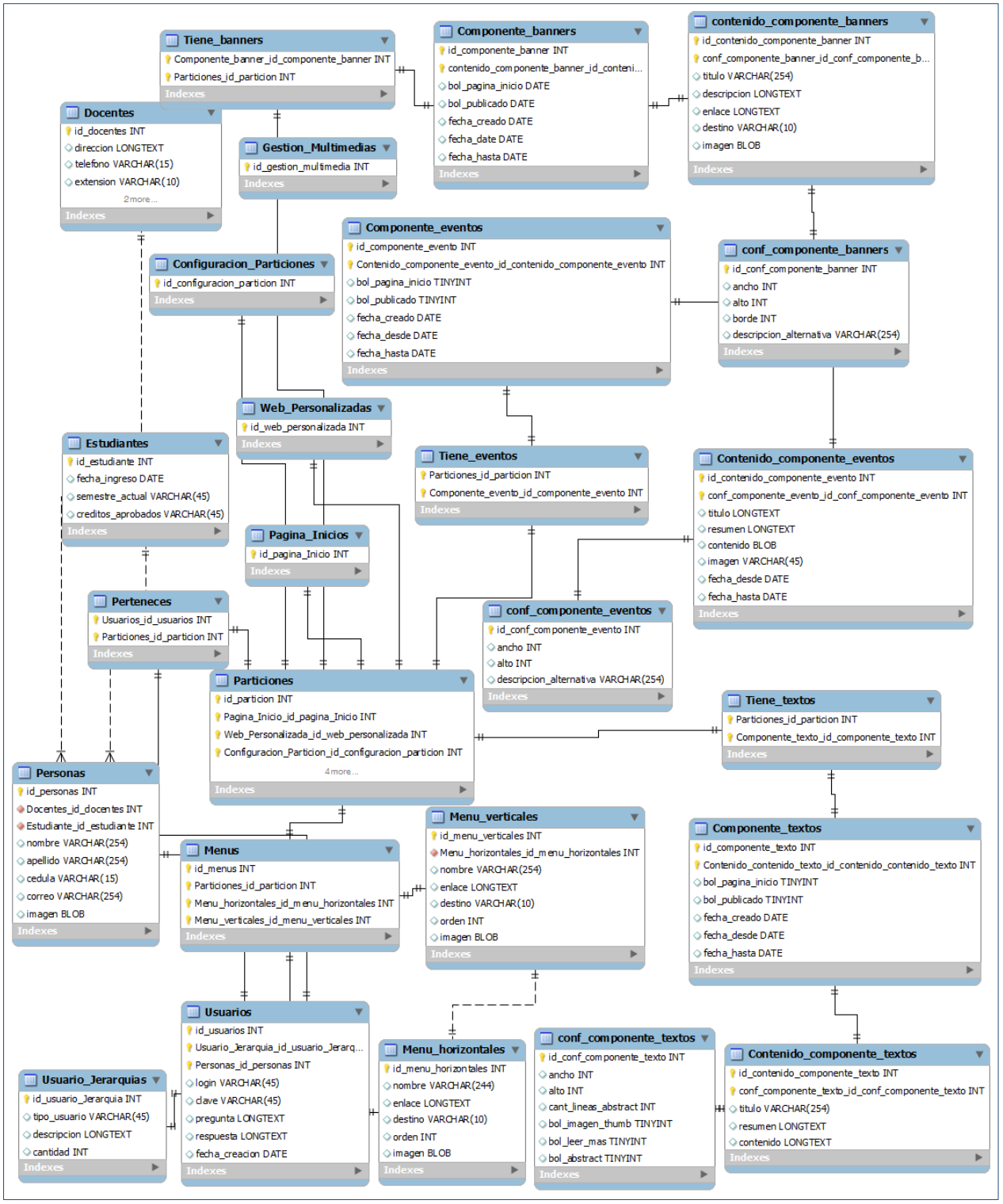


Figura 29 - Modelo de Datos - Versión 1

Se diseño una serie de casos de usos hasta el nivel 1 para tener una idea de lo que se

necesitaban en los módulos principales de la aplicación, los cuales están incluidos en los diagramas de casos de uso expuestos anteriormente.

Mediante la herramienta de MySQL Workbrench se modelo el diseño de datos que se proponía crear para la aplicación, y posteriormente se genero un script de SQL el cual tras algunos ajustes se convirtió en el repositorio de datos de MySQL de "genci2".

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';


-- -----------------------------------------------------
-- Table `genci2`.`Usuario_Jerarquias`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `genci2`.`Usuario_Jerarquias` ;

CREATE  TABLE IF NOT EXISTS `genci2`.`Usuario_Jerarquias` (
  `id_usuario_Jerarquia` INT NOT NULL ,
  `tipo_usuario` VARCHAR(45) NULL ,
  `descripcion` LONGTEXT NULL ,
  `cantidad` INT NULL ,
  PRIMARY KEY (`id_usuario_Jerarquia`) )
ENGINE = InnoDB;
```

Figura 30 - SQL Versión 1

El ambiente de desarrollo usado fue en Ubuntu 11, en el cual se instalo el ambiente de desarrollo de Ruby 1.9.2, Rails 3.0.3 y MySQL 5.0. La instalación de este ambiente se realizo en una maquina portátil y en una computadora de escritorio en las cuales se desarrollo toda la aplicación.

Usando herramientas gráficas de la casa de software Adobe, se realizaron los diseños y maquetación inicial en donde gracias a los vectores creados en Adobe Illustrator, las imágenes retocadas en Adobe Photoshop y usando CSS 3 y Jquery se realizaron las vistas de toda la aplicación, como se muestra en las figuras 31 y 32.

Figura 31 - Interfaz Usuarios Públicos

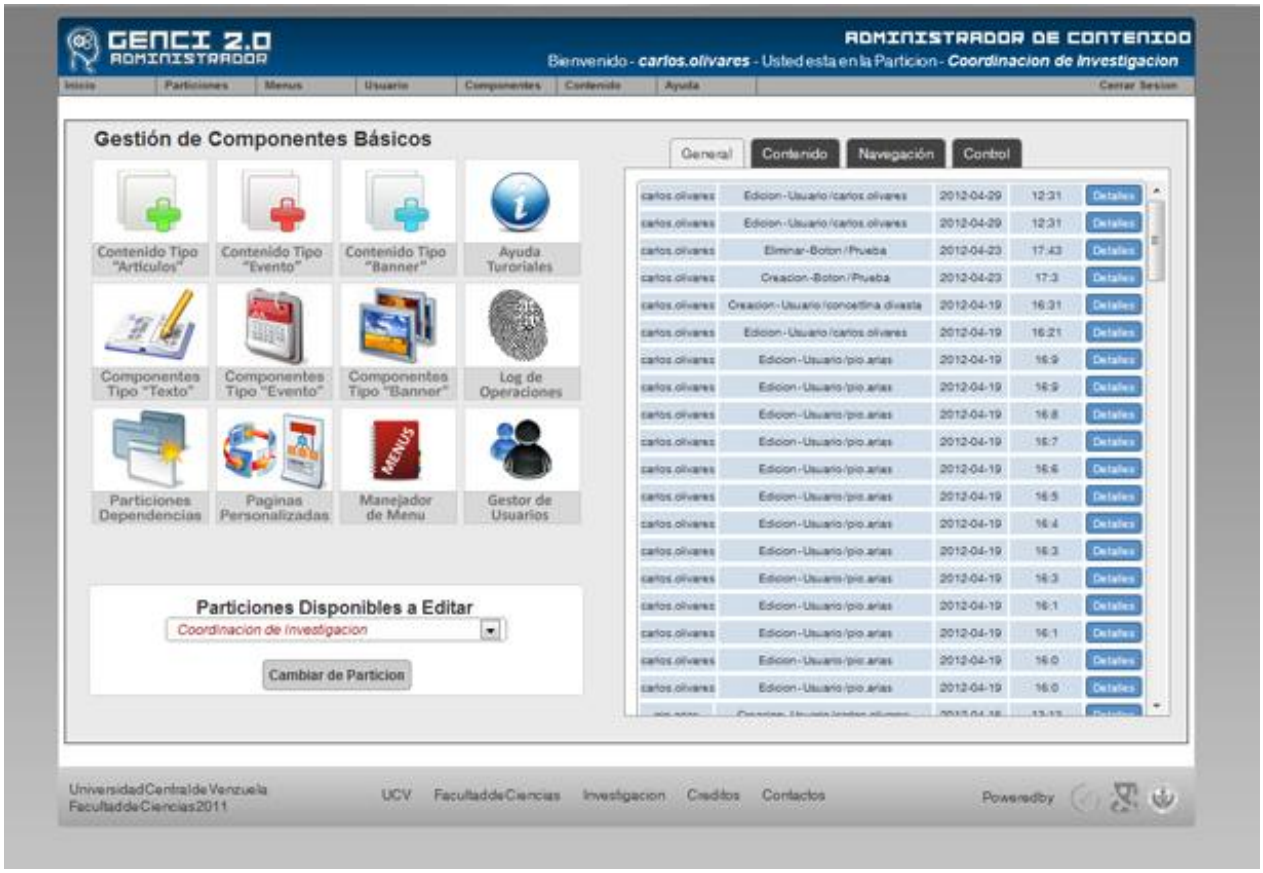


Figura 32 - Interfaz Administrador

Una vez realizadas las vistas, teniendo la base de datos y el ambiente de producción se cargaron datos temporales para realizar las pruebas necesarias de los elementos creados hasta el momento, logrando el objetivo y finalizando con éxito la primera iteración.

3.11.2. Iteración 2

Objetivos Planteados.

- Configuración y desarrollo de método de autenticación mediante la gema Authologic.
- Configuración de archivo “route” y manejo de parámetros por URL.
- Desarrollo de modulo de usuarios.
- Desarrollo de manejo de sesiones y filtros.
- Desarrollo de modulo de gestión de Particiones.
- Desarrollo de lógica de estructura de árbol N-ario, recorridos y modelado.
- Pruebas de módulos desarrollados.

En esta iteración luego de tener el ambiente de desarrollo listo y probado, se continuó la fase de construcción utilizando la gema de Authologic la cual gestiona todos los métodos de

autenticación, sesión, encriptación y demás métodos que se necesitaron para poder manejar usuarios, al mismo se le realizaron múltiples modificaciones para poder guardar en la base de datos la información relacionada al usuario el cual estaba estructurado en el modelo de objeto de "persona" el cual se ve en la primera versión del modelo de datos presentado en la iteración 1.

Tras múltiples pruebas y ajustes, se logro realizar la configuración óptima y operativa de la gema dentro de la aplicación "Genci II" y a partir de allí se crearon usuarios para poder iniciar sesión y realizar trabajo en el ambiente de "Administración" y en el de "Usuarios Públicos"

Figura 33 - Panel de Autenticación Lado Público

Lo siguiente en que se trabajo fue en manejar las rutas de manera que fuese sencillo navegar a través del URL y que de manera dinámica se pudieran definir las rutas de acceso a la aplicación, de tal modo que cuando se creara una dependencia, se pueda publicar y decirle al público que la dirección de esa página web era por ejemplo: www.ciens.ucv.ve/investigacion, o www.ciens.ucv.ve/computacion a diferencia de lo que existe hoy en día en donde la ruta de una página creada por Genci I es complicada ya que tiene diferentes niveles y nombres lógicos en el medio.

En este sentido se trabajo con el archivo "routes.rb" de configuración de la aplicación de Genci II, en donde se programó para que se tomara la primera palabra como un parámetro y no como un controlador, ya que por defecto Rails toma la primera palabra como el controlador, la segunda como método y la tercera como parámetro, quedando de la forma:
http://www.pagina.com/<controlador>/<metodo>/<atributo>

Sin embargo se trabajo para que las rutas fueran tomadas de la forma:
http://www.pagina.com/<parametro>/<identificador>

Y de esta manera al recibir un parámetro lo recibe un modelo que valida la existencia del mismo en la tabla de dependencias, y dependiendo de la respuesta y el identificador el controlador redirecciona hacia un controlador y un método especifico.

routes.rb

```
95
96  #******* RUTAS DE ADMINISTRACION DE PAGINAS PERSONALIZADAS  ******
97  match "(:id)/listar_pagina", :controller => 'admin_conf_pagina', :action => 'index'
98  match "/data_pagina", :controller => 'admin_conf_pagina', :action => 'data'
99  match "(:id)/nueva_pagina", :controller => 'admin_conf_pagina', :action => 'nuevo'
100 match "(:id)/eliminar_pagina", :controller => 'admin_conf_pagina', :action => 'eliminar'
101 match "(:id)/editar_pagina", :controller => 'admin_conf_pagina', :action => 'editar'
102 match "(:id)/editar_archivo", :controller => 'admin_conf_pagina', :action => 'diagramacion_editar_archivos'
103 match "(:id)/borrar_archivo", :controller => 'admin_conf_pagina', :action => 'borrar_archivo'
104 match "(:id)/conf_diagramacion_archivos", :controller => 'admin_conf_pagina', :action => 'diagramacion_archivos'
105 match "(:id)/conf_diagramacion_html", :controller => 'admin_conf_pagina', :action => 'diagramacion_html'
106 match "(:id)/conf_diagramacion_clasica", :controller => 'admin_conf_pagina', :action => 'diagramacion_clasica'
107 match "(:id)/conf_diagramacion_columnas", :controller => 'admin_conf_pagina', :action => 'diagramacion_columnas'
108 match "(:id)/conf_diagramacion_secuencial", :controller => 'admin_conf_pagina', :action => 'diagramacion_secuencial'
109
110 #******* RUTAS DE ADMINISTRACION DE BOTONES DE MENUS  ******
111 match "(:id)/listar_botones", :controller => 'admin_menu_botones', :action => 'listar'
112 match "/data_botones", :controller => 'admin_menu_botones', :action => 'data'
113 match "(:id)/nuevo_botones", :controller => 'admin_menu_botones', :action => 'index'
114 match "(:id)/eliminar_boton", :controller => 'admin_menu_botones', :action => 'eliminar'
115 match "(:id)/editar_boton", :controller => 'admin_menu_botones', :action => 'editar'
```

Figura 34 - Rutas Genci II

Se desarrollo el modulo de gestión de usuarios agregando los métodos de agregar, editar, listar y eliminar de manera que complementándolo con la gema Autologic se pudiera manejar el perfil completo de los usuarios de una menara sin sencilla siguiendo los estándares de un gestor de contenido.

Al finalizar ese modulo se relaciono con los métodos de creación de sesiones y filtros necesarios para los métodos de otros módulos y se procedió a probar con diferentes usuarios en múltiples sesiones para probar el comportamiento de los mismos.

Luego que se logró conseguir el funcionamiento correcto de los usuarios en ambas interfaces (administración y usuarios públicos), validación de sesiones y lectura de parámetros en URL, se procedió a crear particiones o dependencias lógicas en función de poder probar las relaciones entre usuarios y dependencias lógicas creadas desde el administrador de Genci II. En este sentido se desarrollo el modulo de dependencia, el cual es la raíz de la aplicación, por la que esto tuvo un análisis y detalle en la construcción del mismo, aplicando un modelado de asentamiento de datos para poder construir en cualquier momento un árbol N-ario y poder determinar la posición de la dependencia en cualquier momento. Se desarrollaron métodos de creación, edición y listado de dependencias hermanas e hijas.

Figura 35 - Vista de Particiones y Dependencias

La estructura deseada para este modulo debía simular una estructura de nodos en un árbol N-Ario de altura H (en donde H inicialmente era igual a 3, sin embargo H se definió como una variable no limitativa). De esta manera la aplicación permitiría crear una dependencia modelada como un nodo, y de este modo desde la misma se podían crear hijos o hermanos para poder crecer en la estructura de árbol tal como se muestra en la figura 36.

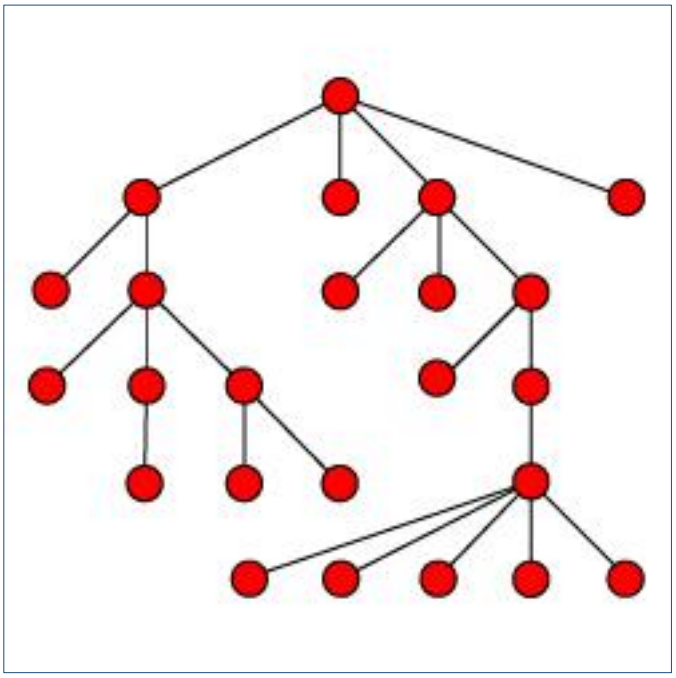


Figura 36 - Árbol N-Ario

3.11.3. Iteración 3

Objetivos Planteados.

- Desarrollo de modulo de Gestión de Menús.
- Desarrollo de modulo de Gestión de Botones.
- Pruebas de módulos desarrollados

Para la tercera iteración ya se tenía listo el modulo de usuarios, la creación de dependencias, el manejo de rutas y validaciones de sesión por lo que el paso siguiente se pensó que era la navegación entre paginas por lo que se inicio la construcción del modulo de gestión de menús, el cual tuvo un alto nivel de dificultad debido al dinamismo de la pagina.

Los enlaces eran relativos, los sitios en donde se posicionaban eran relativos incluso la paginas en donde se iban a incluir era relativo y para ese momento era intangible porque no se tenían paginas, por lo que se inicio el desarrollo con dos tipos de menús; menú horizontal y menú vertical, los cuales en teoría deberían tener la capacidad de aparecer en una sección específica y en algunas páginas seleccionadas por el constructor del sitio. Se pensó que los menús eran dependientes entre ellos ya que un menú podía llevar a una página con otro(s) menú(s), o incluso el mismo menú.

```
#********  METODO QUE CREA EL NUEVO MENU EN LA BD ***********
def nuevo
  if session[:sesion_administracion].to_s == "true"
  usuario_actual
   fecha = Time.now
   @particion = Particione.actual(session[:dependencia])
   @menu = Menu.new(params[:menu])
   @menu.particiones_id = @particion.id
   items = Pagina.where(:particiones_id => @particion.id)
   if @menu.save!
           if !items.blank?
              items.each do |check|
                  if params[:"#{check.id}"] == "ok"
                     @menu_tiene_pagina = MenusTienePagina.new()
                     @menu_tiene_pagina.menus_id =  @menu.id
                     @menu_tiene_pagina.paginas_id = check.id
                     @menu_tiene_pagina.save!
                  end
              end
           end
#********** CREACION DE LOG DE OPERACIONES  ************
```

Figura 37 - Parte del Código de Controlador del Menú

En este sentido de analizó y se empezó la construcción de un contenedor de enlaces relativos, los cuales llamamos “menús”, quienes son responsables de ubicar a una serie de “botones”.

Se tomo como regla que cada dependencia podía tener un solo menú horizontal el cual se comporta como menú principal de navegación y sería un estándar para todas las páginas, mientras que los menús verticales podían variar de visibilidad dependiendo de la configuración de cada página.

Al desarrollar el modulo de gestión de botones se pensó en los posibles casos que existían de enlace los cuales son:

- Enlaces Externos
  - Paginas Externas a Genci II
- Enlaces Internos
  - Paginas Internas de Genci II
  - Componentes de Genci II
  - Dependencias de Genci II

Y las mismas tenían la posibilidad de tener ciertos atributos los cuales debían ser definidos por el administrador del sistema para así no limitar su uso y maximizar el potencial de navegación mediante estos menús dinámicos.

Entre las propiedades que se pueden personalizar están la de agregar una imagen al botón, tener un enlace de cualquier tipo, decidir si se abre en la misma página o en otra, el orden dentro del menú, el nombre, la descripción y la posibilidad de no publicarlo.

Figura 38 - Menú Horizontal (rojo) y Menú Vertical Resultante (morado)

3.11.4. Iteración 4

Objetivos Planteados.

- Desarrollo de modulo de gestión de pagina personalizada (Fase 1) Diagramación Clásica.
- Desarrollo de modulo de gestor de componente texto.
- Desarrollo de modulo de gestor de contenido de texto.
- Pruebas de módulos desarrollados.

Luego de finalizar la tercera iteración era necesario poder navegar haciendo uso de las herramientas que hasta el momento se habían creado, ya que se tenía modelado parte importante de la aplicación pero el contenedor dinámico mayor de todo aun no estaba creado, lo cual fue un reto más el poder crearlo de manera totalmente dinámica, modular y relativa desde su creación hasta su despliegue.

En este sentido se procedió a crear el contenedor en una "diagramación clásica" para hacer las pruebas de cómo distribuir de manera dinámica los elementos dentro de ella, por lo que se investigo la función nativa de rails 3 ***"<%= content_for :identificador do%>"*** el cual se programaba en la vista y podía ser utilizada para ubicarla dinámicamente en las secciones del layout identificado como ***"<%= yield :identificador %>"*** como se muestra en la Figura 39.

```
<!------------------------------------------------------>
<!--      VISTA QUE MUESTRA EL CONTENIDO DEL BANNER     -->
<!------------------------------------------------------>


<%= content_for :contenido_banner do%>
    <%if !@componentes.blank?
            @componentes.each do |comp|
                if comp.seccion.include?'col_banner'
                    if comp.tipo.include?'texto'
                        @componente_elemento = ComponenteTexto.find(comp.componente_id)
                        @contenido_elemento = ContenidoComponenteTexto.where(:componente_textos_id => @componente_elemento.id, :bol
                        %>
                        <div class="modulo">
                        <div class="titulo_modulo"> <%=comp.nombre.to_s%></div>
                    <%
                    elsif comp.tipo.include?'evento'
                        @componente_elemento = ComponenteEvento.find(comp.componente_id)
                        @contenido_elemento = ContenidoComponenteEvento.where(:componente_eventos_id => @componente_elemento.id, :bo
                        %>
                        <div class="modulo">
                        <div class="titulo_modulo"> <%=comp.nombre.to_s%></div>
                    <%
                    elsif comp.tipo.include?'banner'
                        @componente_elemento = ComponenteBanner.find(comp.componente_id)
                        @contenido_elemento = ContenidoComponenteBanner.where(:componente_banners_id => @componente_elemento.id, :bo
                    end
            %>
```

Figura 39 - Parte del código de la vista dinámica del contenido en la sección Banner

Los retos se llevaron tanto del lado administrador en donde se debían ofrecer las opciones de configuración que se habían creado dentro de la aplicación como componentes, contenidos, menús y demás elementos que constituyen una página como también del lado de usuarios

públicos, los cuales solicitan una respuesta y se debía construir en un contenedor general, dependiendo de los elementos solicitados.

Para esta iteración se definió un contenedor o pagina de diagramación clásica con cinco secciones:

- Navegación Horizontal.
- Navegación Vertical.
- Contenido Lateral
- Contenido Banner
- Contenido Central

Posterior a la creación del modulo de gestión de pagina clásica se desarrollo el modulo de creación de componente tipo texto, el cual cuenta con las opciones de crear, editar, listar y eliminar componentes los cuales son contenedores menores de contenido de este tipo.

Seguido a este desarrollo se creó el modulo de gestión de contenido de texto que incluye los métodos de crear, editar, listar y eliminar contenido los cuales a su vez forman parte de un componente, el cual a su vez se puede ubicar dentro de una o más secciones dentro de la pagina de diagramación clásica, como se muestra en la figura 40.

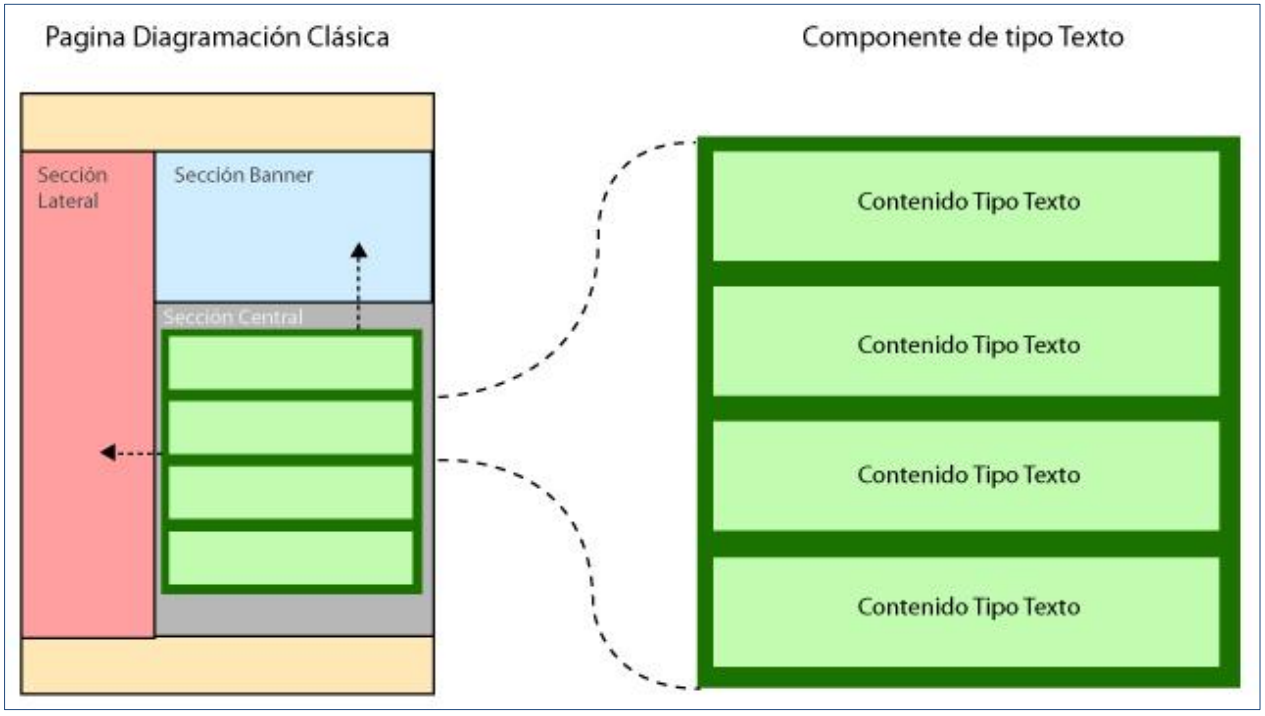


Figura 40 - Componente Tipo Texto en Diagramación Clásica

De esta manera se procedió a crear múltiples módulos con contenidos independientes en páginas separadas y así poderlo acoplar a los demás módulos.

Esta iteración fue sumamente compleja ya que el nivel de abstracción que se tuvo que aplicar a la solución tomo muchas pruebas y ajustes para que el sistema funcionara correctamente, el modelo de datos sufrió varias modificaciones, la lógica a implementar también sufrió modificaciones a medida que se acercaba al resultado deseado.

Al finalizar el desarrollo del modulo de gestión de Pagina Personalizadas en Diagramación Clásica, el modulo digestión de componente tipo texto y el modulo de gestión de contenido tipo texto se procedió a acoplar los módulos ya existentes a la aplicación, lo cual dio como resultado el ajuste en los módulos de menús y manejo de URL, para que se pudiera trabajar de manera fluida y sin causar respuestas indeseadas en el sistema.

3.11.5. Iteración 5

Objetivos Planteados.

- Desarrollo de modulo de gestión de pagina personalizada (Fase final) Diagramación clásica.
- Desarrollo de modulo de gestión de pagina personalizada (Fase final) Diagramación Secuencial.
- Desarrollo de modulo de gestión de pagina personalizada (Fase final) Diagramación 3 Columnas.
- Depuración de modulo de gestión de componentes de texto.
- Depuración de modulo de gestión de contenido de texto.
- Desarrollo de modulo de gestión de componentes de eventos.
- Desarrollo de modulo de gestión de contenido tipo eventos.
- Desarrollo de modulo de gestión de componente banner.
- Desarrollo de modulo de gestión de contenido tipo banner.
- Ajustes de modulo de gestión de pagina personalizada.
- Depuración de modulo de gestión de menús.
- Depuración de modulo de gestión de botones.
- Pruebas de módulos desarrollados.
- Presentación parcial de aplicación a cliente y tutor.

Al finalizar la cuarta iteración se podía navegar entre páginas dinámicas con componentes de tipo texto, se tenía manejo de dependencias, usuarios, rutas, menús y botones. Sin embargo aun hacía falta realizar ajustes al modulo de edición y construcción de pagina clásica.

Para esta modificación se desarrollo una interfaz de “arrastras y soltar” (Drag & Drop) implementando Jquery y Css, de manera que la construcción de las paginas se pudiera hacer de una forma sencilla para usuarios regulares, esto trajo como consecuencia un trabajo fuerte a nivel de interfaces y manejo dinámico de variables, el cual se uso mediante la librería de JavaScript de JSON para poder controlar las múltiples variables dinámicas producidas por la interacción de los usuarios dentro de la interfaz grafica, parte del código se puede ver en la figura 41.

```
#*** titulo_   -> Tiene el titulo del componente          **********
#*** numero_items_  -> Tiene la cantidad de Items permitidos   *****
#*** tipo_componenete_   -> Tien l tipo de componente        *******
#*****************************************************************
def guardar_diagramacion
 fecha = Time.now
 @particion = Particione.actual(session[:dependencia])
 @pagina = Pagina.find(params[:id_pagina])
 bloque = "bloque_"
 orden = "orden_"
 titulo = "titulo_"
 numero = "numero_items_"
 tipo = "tipo_componente_"
 columna = "columna_"
 contador_not_in = 0
 arreglo_not_in = []
 @arr = ActiveSupport::JSON.decode(params[:arreglo_componentes])
 if !@arr.blank?
     @arr.each do |componente|
     arreglo_not_in[contador_not_in] =   componente.to_s
     contador_not_in +=1
       handle_tiene_componente = TieneComponente.existe(componente)
       cambiar = "false"
# Si el elemento esta en la BD reviso si algun valor fue modificado para actualizarlo
        if !handle_tiene_componente.blank?
            handle_tiene_componente.each do |valores|
               if params["elemento_tipo_"+"#{componente.to_s}"].to_s.downcase.include?('texto')
                 tipo_elemento = "texto"
```

Figura 41 - Manejo de JSON en el controlador de Paginas

Una vez que la interfaz de construcción se concreto correctamente para la diagramación clásica y los componentes de tipo texto, se procedió a desarrollar los módulos de gestión de componentes de tipo eventos, y tipo banner.

Para estos dos módulos se implemento la misma lógica que se aplico para el desarrollo del modulo de gestión de texto, ya que como son contenedores y los cambios significativos esta en los datos del contenido, lo único que cambiaba eran los tipos de componentes a nivel del modelo de datos.

En este sentido se crearon los módulos de gestión tipo evento y tipo banner y para ambos casos se implementaron los métodos de crear, listar, editar y eliminar componente.

Posterior a este desarrollo se procedió a desarrollar el modulo de gestión de contenido tipo evento, para lo cual se analizaron diferentes plugins, librerías o módulos que ofrecieran esta herramienta, y se selecciono un desarrollo de Jquery el cual cubre con todas las necesidades propuestas para este modulo. Se analizaron los atributos necesarios para su funcionamiento, se modelaron y se construyeron los métodos de creación, edición, listado y eliminación de este tipo de contenido. Luego del lado del usuario público se implementó la visualización del calendario en todas sus formas y se modelaron los datos extraído de la base de datos.

Para el caso del contenido de Banner se realizo el mismo procedimiento, en donde se analizo y busco la herramienta de Jquery que mejor se ajustara a las necesidades de la aplicación, logrando implementar un desarrollo libre que existía basado en Jquery. Se procedió a modelar los atributos necesarios y a crear los métodos de crear, listar, editar y eliminar componente, en este caso fue necesario el uso de la gema de PaperClip para la carga de imágenes. Luego del lado del usuario público se implementó la visualización del banner rotatorio modelado según los datos asentados en la base de datos.

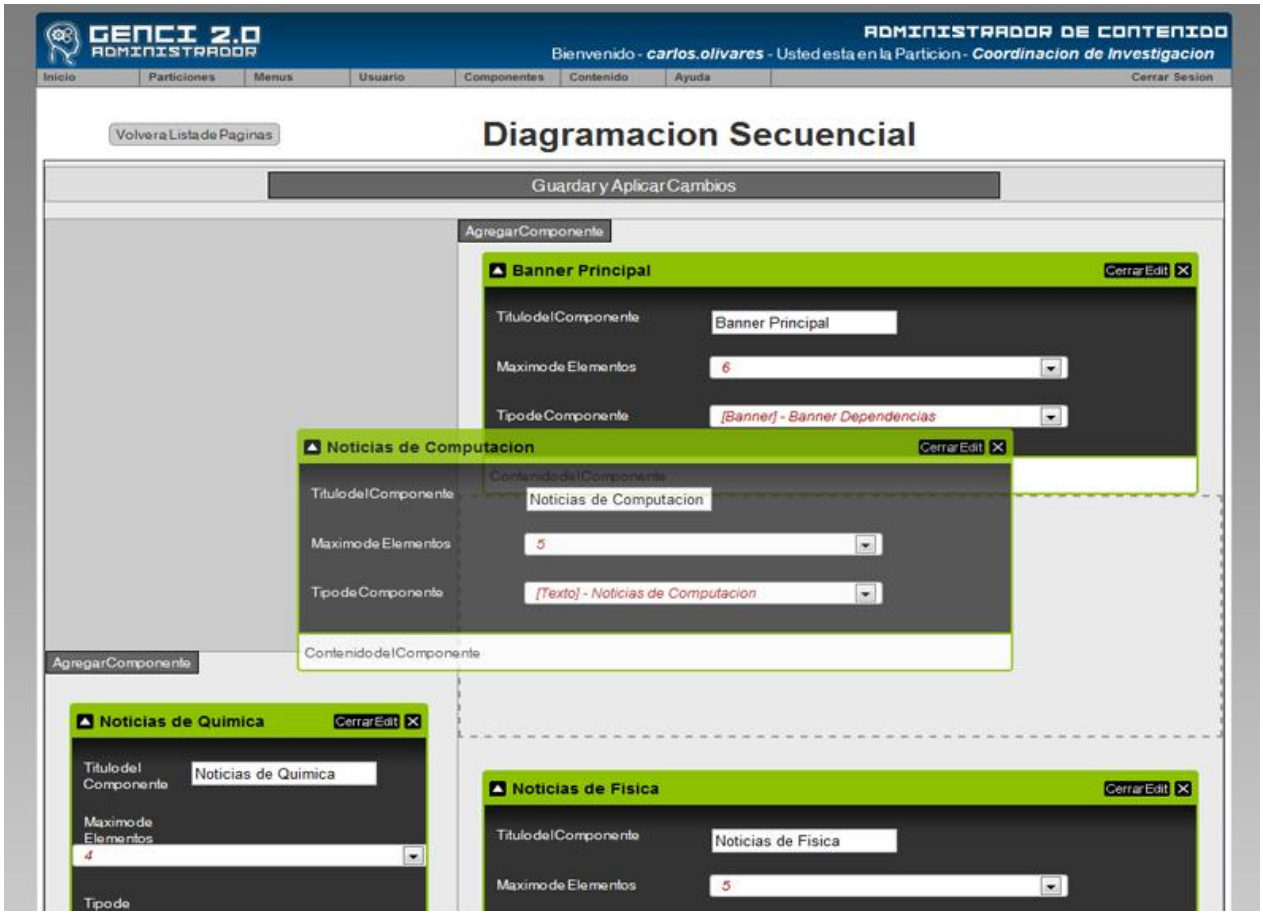


Figura 42 - Construcción Drag And Drop de Página Personalizada

En este punto se realizaron pruebas parciales sobre el funcionamiento de los tres tipos de componentes y tres tipos de contenido en la página de diagramación clásica.

Luego de ajustes de acoplamiento entre los módulos creados se procedió a crear los dos tipos de diagramación faltantes, los cuales fueron la diagramación secuencial y la diagramación 3 columnas.

El desarrollo de estas opciones gracias a las bondades de ruby y de rails, se hizo bastante rápido ya que con los métodos de content_for y yield, se crearon las vistas tanto del lado del administrador como en los layout del lado del usuario público, se ajustó la configuración para estos casos en las librerías de JSON, Jquery y Css para finalizar sus desarrollos.

Una vez que se logro el funcionamiento correcto de los módulos de componentes, contenido y paginas personalizadas de los tres tipos, fue necesario realizar un ajuste en los módulos desarrollados hasta el momento y realizar pruebas unitarias de cada uno para ver el comportamiento y despliegue de cada uno trabajando en conjunto.

En este sentido se modifico el modulo de Menús y Botones para poder navegar entre paginas internas y externas, se modifico el archivo de rutas para evitar errores o posibles entradas invalidas mediante el URL.

Se crearon diferentes dependencias con páginas personalizadas, contenidos especiales y menús personalizados, menús y botones particulares y se probó la navegación de todos los casos que hasta el momento se podían crear y se tomaron los correctivos necesarios para finalizar esta iteración.

El día 19 de Marzo del año 2012 a las 10 de la mañana se presento ante el coordinador de investigación de la facultad de ciencias, el Dr. Pio Arias y la Profa. Mercy Ospina tutor académico, de manera parcial la aplicación para que los revisaran los avances y sugirieran cambios necesarios al sistema.

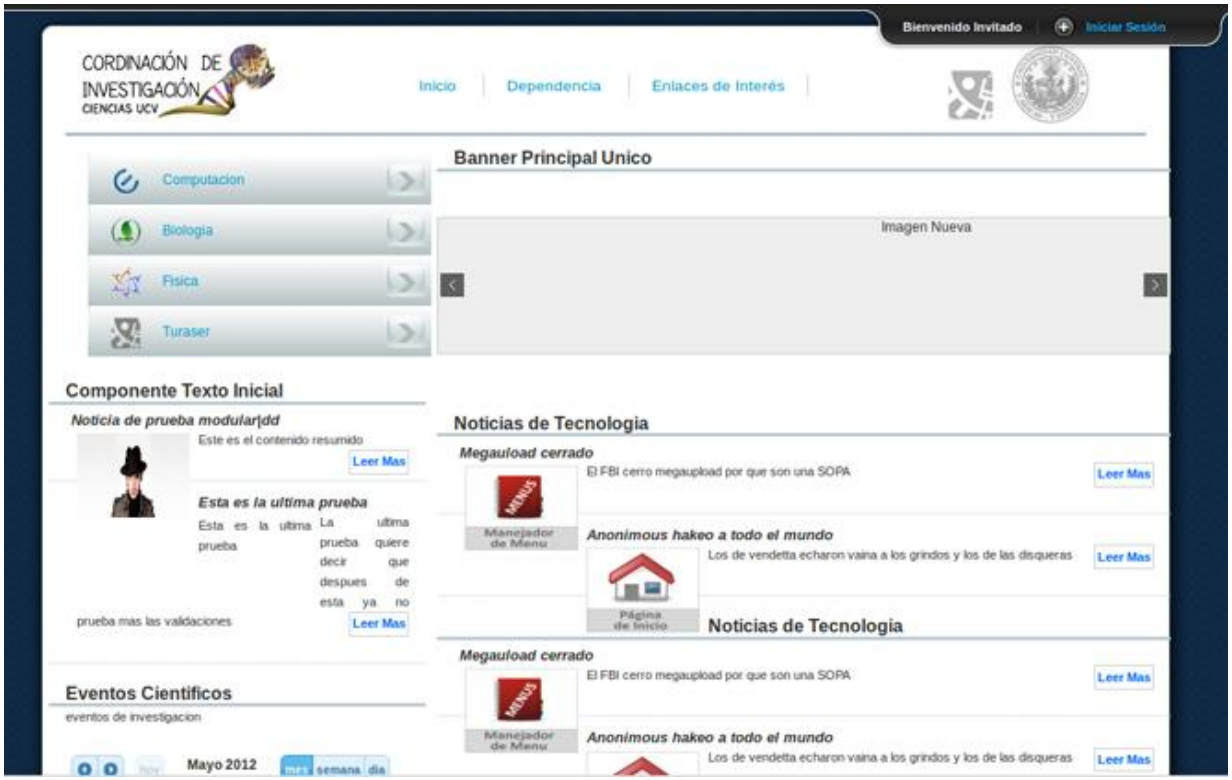


Figura 43 - Vista publica Presentada

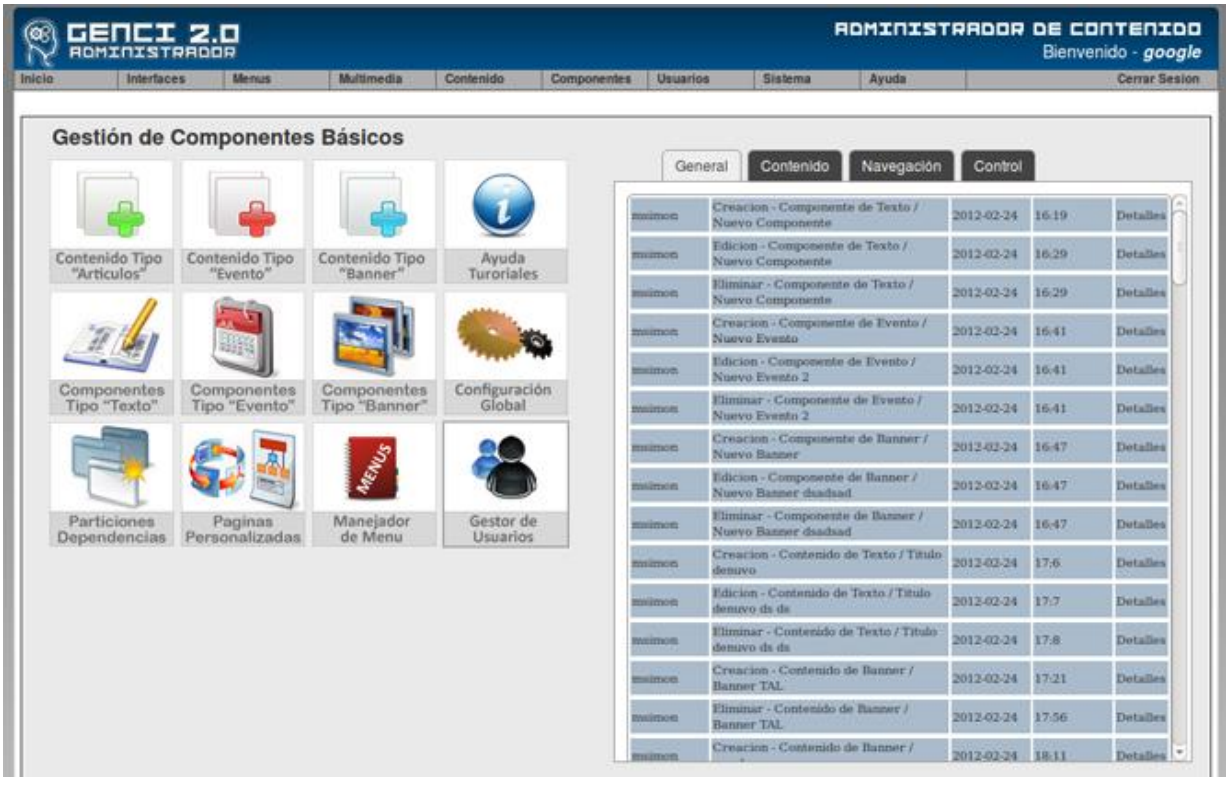


Figura 44 - Vista Administrativa Presentada

3.11.6. Iteración 6

Objetivos Planteados.

- Desarrollo de modulo de gestión de pagina personalizada Pagina HTML.
- Desarrollo de modulo de gestión de pagina personalizada Pagina de Archivos.
- Ajustes al modulo de usuarios, creando diferentes roles y perisologías.
- Desarrollo de lógica de recorrido en árbol N-ario para perisologías.
- Desarrollo de módulos de control de particiones para súper-usuarios.
- Pruebas de módulos desarrollados.
- Presentación parcial de aplicación a tutor.

Tras la reunión mantenida con el Dr. Arias y la Profa. Ospina, sugirieron modificaciones las cuales garantizaban la implementación de la aplicación de manera inmediata ya que las propuestas harían el sistema más flexible y con un grado de administración mas controlable, ya que se podrían delegar diferentes roles para cargar datos de prueba y puesta en producción.

Entre las modificaciones sugeridas se planteó:

- Agregar tipo de página personalizada tipo “Archivo” pensada para las páginas de descarga de recursos, los cuales son usados en la facultad en cada materia.
- Agregar tipo de página personalizada tipo “HTML” pensada para satisfacer cualquier otra necesidad que no pudiese ser modelada con componentes, dándole así la libertad al usuario de escribir HTML y que el navegador lo pudiese interpretar.
- Definir roles de administración con diferentes permisos, incluyendo un supe usuario capaz de navegar desde el administrador a todas las dependencias sin necesidad de salir y volver a iniciar sesión, y un colaborador con privilegios limitados.
- Modificaciones generales en la forma como se despliegan los datos, como se alimentan los datos y permisos a algunas funcionalidades para tipos de usuario.

A partir de estas necesidades se inicio la modificación de la aplicación para ajustarse a las sugerencias planteadas.

Se desarrolló el modulo de construcción de pagina HTML usando un desarrollo de Jquery, ajustado a las necesidades del cliente en donde por medio de una interfaz grafica, se puede crear contenido con herramientas de ofimática, el cual traduce lo requerido a código HTML, permitiendo también al usuario experto poder colocar dicha interfaz en "modo código" y así poder editar o escribir código HTML con toda liberta, y cambiando al modo asistido puede visualizar de manera inmediata el resultado obtenido, tal como se muestra en la figura 45.

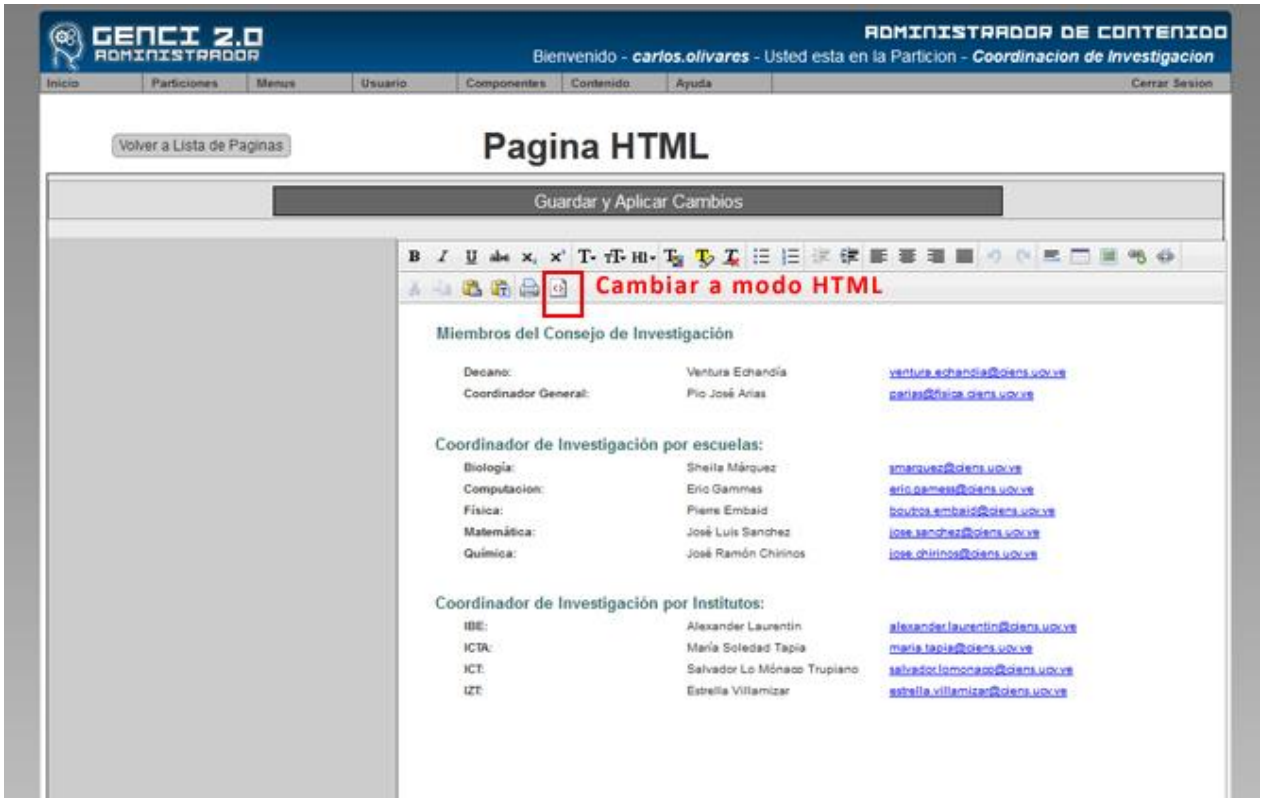


Figura 45 - Vista pagina HTML

Luego se desarrollo el modulo de construcción de pagina de Archivos, la cual da la posibilidad al administrador del sistema a crear un repositorio de archivos el cual pueda ser visualizados por los usuarios públicos y puedan descargar diferentes tipo de archivos desde el servidor.

Para este modulo se utilizo el método nativo de rails de ActiveRecord de DataFile el cual permite gestionar archivos en un repositorio definido por el desarrollador, parte del código se puede visualizar en la figura 46.

```
class Datafile < ActiveRecord::Base

  def self.save(upload,directorio)
   if !File.directory?('public/archivos/'+directorio)
      Dir.mkdir('public/archivos/'+directorio, 777)
   end

    name =  upload['datafile'].original_filename
    directory = "public/archivos/"+directorio+"/"
    # create the file path
    path = File.join(directory, name)
    # write the file
    File.open(path, "wb") { |f| f.write(upload['datafile'].read) }
  end

end
```

Figura 46 - Modelo del Manejador de Archivos

De esta manera se concluiría el desarrollo de paginas personalizadas permitiendo de esta manera crear al administrador del sitio paginas personalizadas entre las que podría construir paginas de componentes de texto, eventos y banner de imágenes, en tres tipos de diagramación posibles; a su vez podía crear una página de lista de archivos a descargar y en caso que el sistema con estas opciones no le brindara la posibilidad de crear el contenido como se desea, se creó una vista que permite crear paginas HTML ajustadas a las necesidades del administrador.

Una vez que se concluyo esta fase de desarrollo se modifico el modulo de usuarios, agregando nuevos roles y definiendo permisologías para que de esta manera se pudieran distribuir las responsabilidades entre los actores de la coordinación de investigación. Se definieron tres tipos de usuarios administrativos:

- **Administrador Global:** Capaz de administrar todos los elementos de todas las dependencias, capaz de cambiar de dependencia sin necesidad de volver a iniciar sesión.
- **Administrar de Dependencia:** Capaz de administrar todos los elementos de la dependencia a la cual se le fue asignado.

- **Colaborador de Dependencia:** Capaz de administrar únicamente el contenido de las estructuras creadas por los administradores.

De esta manera en el modelo de datos se realizo una modificación para ajustarse a estos requerimientos, y las vistas modelos y controladores involucrados en estos requerimientos fueron ajustados para que se comportaran según las necesidades, ya que el rol del usuario permite ejecutar ciertas funcionalidades que otros roles no pueden, repercutiendo esto el árbol N-Ario ya que el usuario al pertenecer a un nodo especifico, es necesario poder ubicar parámetros del mismo para poderle asignar ciertos privilegios.

De esta manera se muestra en la figura 47 como quedo el cambio de dependencia a administrar en la interfaz de administración, sugerida por el Dr. Arias.

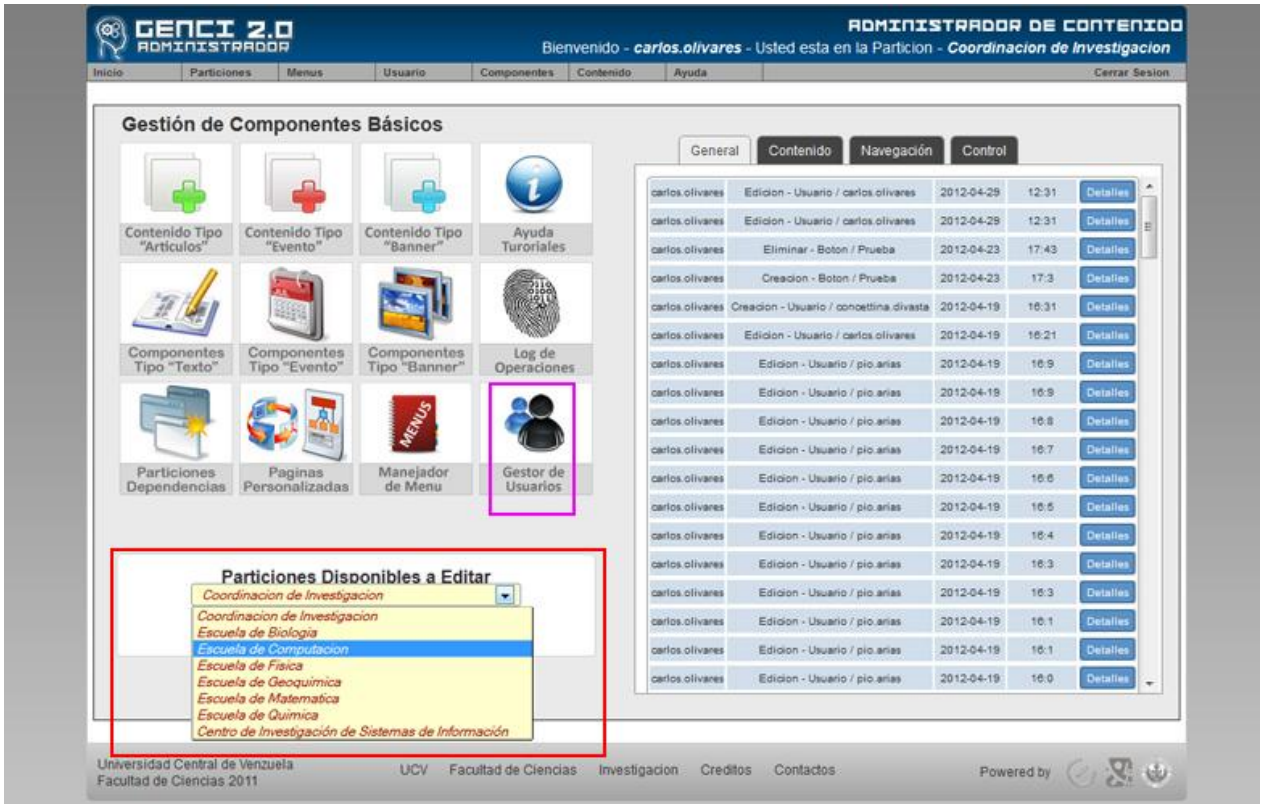


Figura 47 - Cambio de Dependencia a Administrar

Luego de esto se realizo una presentación a la Profa. Ospina la cual valido las modificaciones sugeridas y aprobó el desarrollo de los módulos que se desarrollaron, quedando ajustes menores en la aplicación antes de montarla en el servidor y ponerlo en producción.

3.11.7. Iteración 7

Objetivos Planteados.

- Desarrollo de log de operaciones.
- Desarrollo de validaciones en formularios y Bases de Datos.
- Depuración y mejoras en interfaz, estilos y carga de datos.
- Depuración de errores de la aplicación.
- Carga de datos de prueba y para la puesta en producción.
- Pruebas de módulos desarrollados.
- Prueba de usuarios.
- Presentación parcial de aplicación a tutor y jurado.

Al finalizar la sexta iteración luego de la aprobación de la tutora académica, se realizo una revisión general de toda la aplicación en donde se tomaron las correcciones necesarias y se mejoraron los aspectos que se consideró suficiente para la presentación de este trabajo especial de grado.

Se desarrollo un modulo de bitácora, o log de operaciones en donde el administrador puede visualizar en primera lista las entradas de los usuarios y las acciones que ejecutan en el sistema, como también ver los detalles de cada entrada. Se agrego un buscador de entradas filtrada por fecha, para que de esta manera se pudiera controlar el acceso a la aplicación y los cambios realizados por todos los usuarios que interaccionan con el tal como se ve en la figura 48.

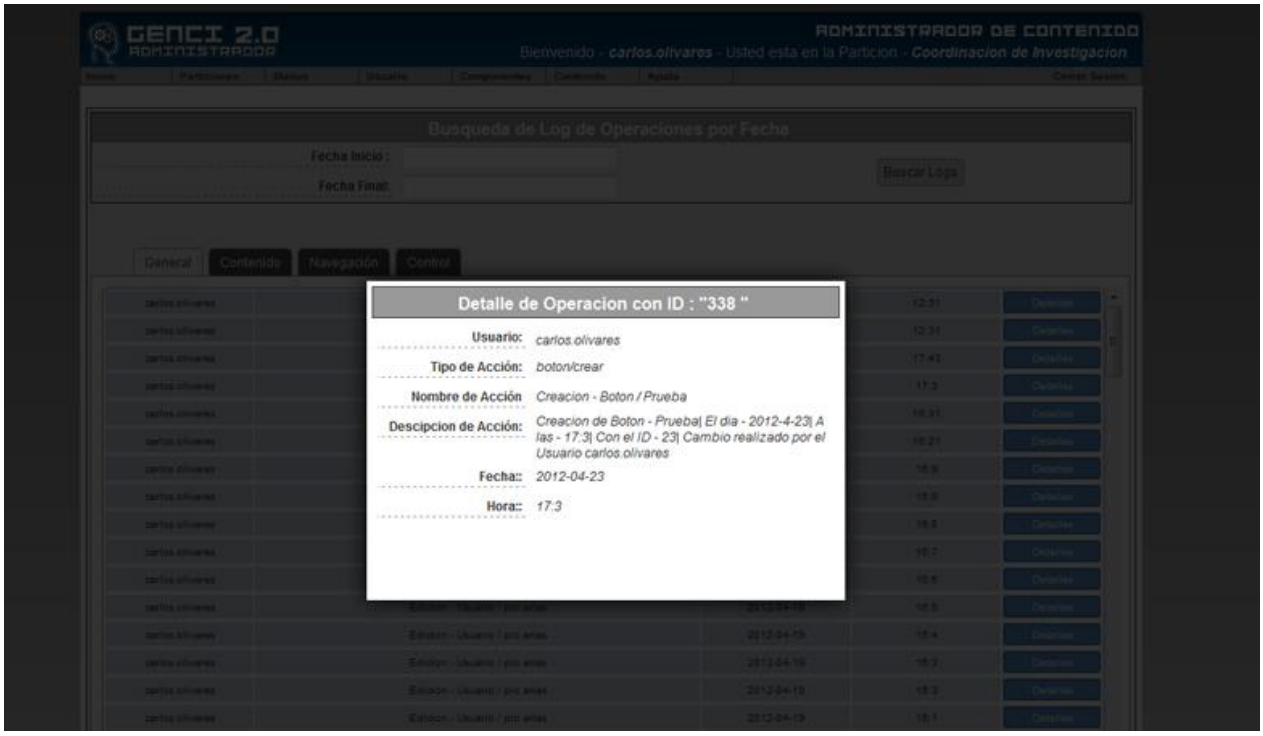


Figura 48 - Detalle de la entrada en el log

Posterior a este ultimo desarrollo modular se realizo una inspección de casos bordes validando mediante JavaScript y Rails los múltiples formularios y entradas del usuario, para de esta manera poderle guiar en su trabajo con la aplicación, como también prevenir que puedan existir errores en la ejecución de algún modulo de la aplicación.

Se realizaron mejoras en la interfaz ya que esto ayudaría a que el usuario tenga mayor confianza con el sistema y le sea más intuitivo el uso de cada una de las herramientas, utilizando imágenes que sirvieran como metáforas, Css3, estilos de Jquery.

Se limpió el código para no dejar impresiones de prueba “puts”, indentación necesaria y documentación extra a la que se fue colocando a medida que se desarrollo la aplicación

Para finalizar esta iteración, y como prueba final se vació toda la base de datos y se inicio la carga de todos los datos desde cero, pero esta vez tomando datos reales para depurar cualquier error y tener información que pueda ser utilizada en la primera fase de producción de la aplicación.

Se hicieron diseños especiales para los tipos de banner y los botones de cada sección y se cargaron al sistema.

Una vez que se valido que el sistema no presento fallas y estaba completo, se pidió la colaboración de dos personas ajenas al desarrollador para que utilizaran el sistema y dieran sus sugerencias tanto en la parte administrativa como en la parte publica, logrando de esta forma ajustar aun más la experiencia al usuario tomando las recomendaciones de ellas.

Una vez finalizado este proceso se termino la carga de la información y se fue con la versión final para presentársela al tutor académico, Profa. Mercy Ospina, y el Jurado. Prof. Robinson Rivas el viernes 13 de Abril.

3.11.8. Iteración 8

Objetivos Planteados.

- Desarrollo de modulo de autenticación LDAP.
- Ajustes en el modulo de usuarios.
- Ajustes en permisos, sesiones y manejo de usuarios.
- Puesta en producción de la aplicación en maquina virtual del centro de computación.
- Carga de video tutoriales.
- Prueba Funcional en Ambiente de Producción.
- Realización de encuestas a usuarios.

Tras la presentación ofrecida al Prof. Rivas durante la entrega parcial de la aplicación surgieron preguntas de implementación de cómo se construyeron los módulos y en el momento en el que se discutió sobre el funcionamiento del modulo de usuarios y privilegios, se planteo una modificación completa del modulo dejando deshabilitado el que se había desarrollado en la segunda iteración, para poner en producción un nuevo modulo con requerimientos específicos de cómo debe ser el correcto uso del mismo para garantizar que la puesta en producción dentro de la facultad de ciencias fuese de una mera más homogénea con los sistemas existentes hasta el momento.

Entre los requerimientos solicitados estaba el de utilizar el directorio de usuarios activos de la facultad de ciencias, los cuales se encuentran en un repositorio centralizado en una estructura de datos bajo el modelo de LDAP, el cual es un directorio que contiene el perfil de los usuarios de la facultad de ciencias, tanto estudiantes como docentes y personal administrativo.

De esta manera los datos de los usuarios utilizados por el sistema Genci II, son los existentes dentro del directorio centralizado y se evita duplicar datos de control que ya se encuentran normalizados por los entes pertinentes dentro de la facultad de ciencias.

Para utilizar el sistema LDAP, se utilizo una gema de Ruby llamada “ruby-ldap” en su versión 0.9.19 la cual tiene un api con métodos definidos los cuales hicieron que el desarrollo de este requerimiento se pudiera hacer en una tarde.

En este punto se encontraron limitaciones debido a que por razones de configuración y seguridad, los datos de LDAP de la faculta de ciencias, solo puede ser consultado desde la red interna de la facultad, por lo que las pruebas y desarrollos se debieron hacer desde el laboratorio de Base de Datos, o en una sala la cual se dispuso a través del Prof. Rivas dentro del centro de computación, en donde se contó con todos los recursos y apoyo para poder lograr el objetivo de implementar dichas funcionalidades, parte del código en donde se implementa LDAP se muestra en la figura

```
class ApplicationController < ActionController::Base
  protect_from_forgery


def create

#*******************************************
#  DATOS DE CONEXION Y CONFIGURACION DE LDAP
#*******************************************

require 'rubygems'
require 'net/ldap'
ldap = Net::LDAP.new
ldap.host = "190.169.94.7"
ldap.port = "389"

#*******************************************
#        USUARIO DE AUTENTICACION A LDAP
#*******************************************

username = params[:user_session][:login]
password = params[:user_session][:password]

#*******************************************
#  METODO DE AUTENTICACION DE USUARIO BIN
#*******************************************

ldap.auth "cn=#{username_root},dc=ciens,dc=ucv,dc=ve", password_root
sesion_conectada = ldap.bind

ldap.auth "uid=#{username},ou=users,dc=ciens,dc=ucv,dc=ve", password
usuario_valido = ldap.bind
```

Figura 49 – Código de LDAP en Método Create

Una vez finalizado el nuevo modulo de gestión de usuarios, se redefinieron los métodos afectados en las perisologías, sesiones y filtros que antes se manejaban a través de la gema Authologic, utilizando de esta manera los métodos de autenticación y validaciones de LDAP, en donde con el usuario y contraseña del usuario registrado en el sistema se puede entrar a la sesión de administración.

Existen las limitaciones de que un administrador debe activar el usuario que se desea utilizar en el sistema de Genci II, y el único dato que debe utilizar es su nombre de usuario, enseguida se le asigna el rol que se desea, la dependencia a la pertenece y se decide activar o no activar. El resto de los datos son extraídos directamente del directorio de ldap.

El sistema no permite crear usuarios que no estén activos dentro del directorio ldap y la validación principal es la del nombre de usuario, quedando la edición limitada a solo cambiar el rol que tiene el usuario creado, la dependencia y la activación del mismo, tal como se muestra en la figura 50 .

Figura 50 – Gestión de Usuarios “Edición”

Luego de hacer los ajustes y pruebas en conjunto con todos los demás módulos se procedió a configurar la maquina virtual que se asigno para esta aplicación, a la cual solo se tuvo acceso al puerto 22 mediante SSH, y a través de la consola se realizaron todas las instalaciones y configuraciones necesarias para que el sistema trabajara de manera correcta y pudiera resolver las peticiones de clientes web, parte de la configuración del ambiente de producción puede verse en la figura 51.

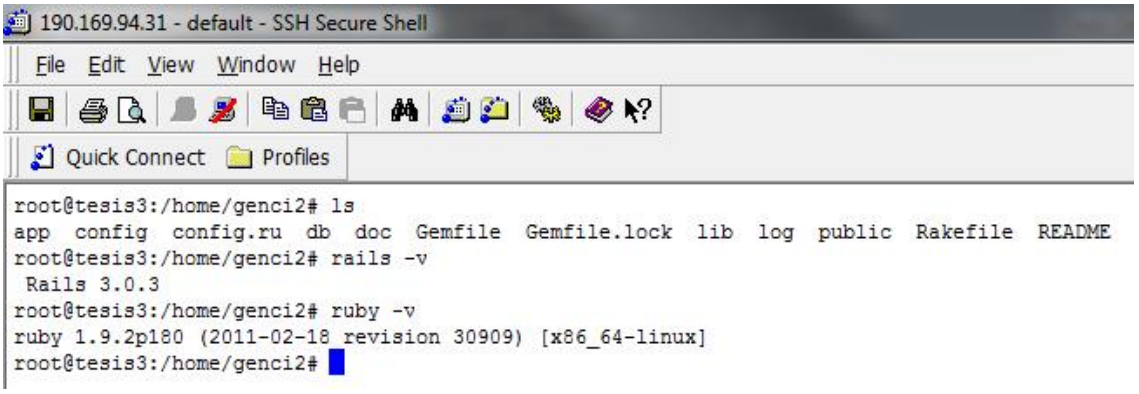


Figura 51 – Ambiente de Producción SSH

Una vez que todos los módulos en ambiente de producción se probaron y funcionaron correctamente, se procedió a realizar las encuestas de usuarios públicos y usuarios administrativos, cuya información más detallada se encuentra en el punto 3.12, fase de transición.

3.12. Transición

En la fase de transición se realizaron los trabajos necesarios para que la aplicación quedara funcionando en un ambiente de producción, y pudiese ser accedido desde un URL en cualquier parte del mundo, adicional a eso se realizaron pruebas de usabilidad y encuestas al usuario las cuales fueron el feedback directo de clientes reales de la aplicación.

A continuación se presentan a mayor detalle las partes involucradas en esta fase.

3.12.1. Ambiente de Producción

En esta fase se colocó una versión de la aplicación en un servidor del centro de computación, el cual tiene las siguientes características.

Actualmente está ejecutándose en un computador que no tiene arquitectura de servidor debido a que está siendo usado para migrar el servidor Web actual, además a la máquina física se le han definido varias máquinas virtuales, y la aplicación Web de la escuela corre en una de estas máquinas virtuales. Las características de este computador son:

Máquina Física:
Hardware:

- 8 GB de RAM
- Disco Duro 1 Tera Raid-1 GB
- Procesador QuadCore 3.0 GHZ

Software:

- Linux Debian Squeeze
- XEN Software
- SSH server

Máquina Virtual donde se ejecuta la aplicación de la Escuela de Computación:
Hardware:

- IP Publico 190.169.94.31
- 512 MB RAM
- Disco Duro 30 GB
- Procesador DualCore 3.0 GHZ

Software (Máquina Virtual):

- Linux Debian Squeeze
- Ruby 1.9.2
- Rails 3.0.3
- Webrik
- Authologic
- Paperclip
- MySQL2
- SSH server
- MySQL Server 5

Dentro de la máquina virtual se creó una carpeta donde está alojado el sistema en el directorio /home/genci2, el cual se encuentra en el IP público 190.196.94.31 dentro del centro de computación.

3.12.2. Diagrama de despliegue

A continuaciones se observa en la figura 45 el diagrama que muestra el despliegue de la aplicación

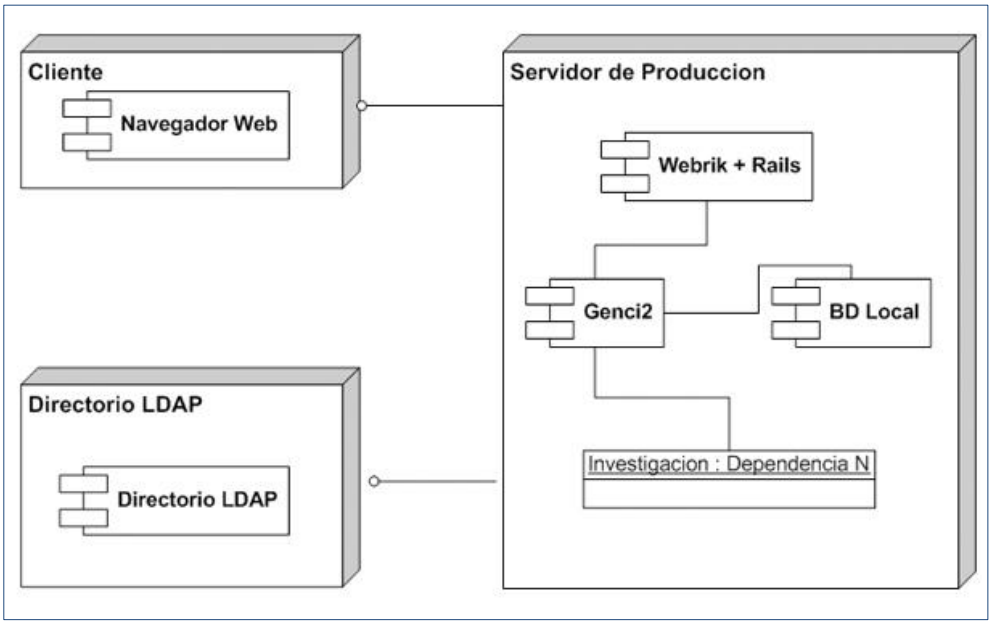


Figura 45 - Diagrama de Despliegue

3.12.3. Encuestas

Las encuestas fueron realizadas en línea por medio de una herramienta llamada Survey Monkey cuyo enlace es www.surveymonkey.com, la cual ofrece una herramienta de realización de encuestas y medición de resultados de forma gratuita.

Se realizaron dos encuestas diferentes, la primera orientada al módulo de administración de GENCI-2 y la segunda hacia el público en general que se mantendrá informado a través de la aplicación.

Usuarios Público General

En la encuesta para el público en general que se puede observar a continuación, se realizaron preguntas orientadas a la usabilidad de la aplicación, interfaz gráfica, colores utilizados y un puntaje general sobre el manejo de la aplicación GENCI-2.

Encuesta para público en general

1. ¿Cómo le parece la implementación de colores en el diseño de la aplicación Web Genci - 2?

- Excelente
- Bueno
- Regular
- Malo

2. ¿Qué opina con respecto al componente de texto?

- Excelente
- Bueno
- Regular
- Malo

3. ¿Qué opina con respecto al componente de eventos?

- Excelente
- Bueno
- Regular
- Malo

4. ¿Qué opina con respecto al componente de banner?

- Excelente
- Bueno
- Regular
- Malo

5. ¿Qué le parece el estilo del menú lateral?

- Excelente
- Bueno
- Regular
- Malo

6. ¿Qué le parece el menú horizontal?

- Excelente
- Bueno
- Regular
- Malo

7. ¿Cómo le parece el acceso a la dependencia de su interés?

- Sencillo
- Regular
- Complicado

8. En la escala del 1 al 5, siendo el 5 la puntuación máxima. ¿Qué le parece el resultado del desarrollo de la aplicación Web Genci - 2?

- 5
- 4
- 3
- 2
- 1

Resultados de la encuesta hacia el público en general

La encuesta hacia el público en general fue realizada a 21 personas, las cuales seleccionaron las siguientes opciones para la obtención de los diferentes resultados.

1. ¿Cómo le parece la implementación de colores en el diseño de la aplicación Web Genci - 2?

Crear gráfico Descargar

	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**76,2%**	**16**
Bueno	23,8%	5
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**21**
	pregunta omitida	**0**

2. ¿Qué opina con respecto al componente de texto?

Crear gráfico Descargar

	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**95,2%**	**20**
Bueno	4,8%	1
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**21**
	pregunta omitida	**0**

3. ¿Qué opina con respecto al componente de eventos?

Crear gráfico Descargar

	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**85,7%**	**18**
Bueno	14,3%	3
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**21**
	pregunta omitida	**0**

4. ¿Qué opina con respecto al componente de banner?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**71,4%**	**15**
Bueno	28,6%	6
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**21**
	pregunta omitida	**0**

5. ¿Qué le parece el estilo del menú lateral?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**71,4%**	**15**
Bueno	28,6%	6
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**21**
	pregunta omitida	**0**

6. ¿Qué le parece el menú horizontal?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**65,0%**	**13**
Bueno	35,0%	7
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**20**
	pregunta omitida	**1**

7. ¿Cómo le parece el acceso a la dependencia de su interés?		Crear gráfico	Descargar
		Porcentaje de respuestas	Cantidad de respuestas
Sencillo		**90,5%**	**19**
Regular		9,5%	2
Complicado		0,0%	0
		pregunta respondida	**21**
		pregunta omitida	**0**

8. En la escala del 1 al 5, siendo el 5 la puntuación máxima. ¿Qué le parece el resultado del desarrollo de la aplicación Web Genci - 2?		Crear gráfico	Descargar
		Porcentaje de respuestas	Cantidad de respuestas
5		**76,2%**	**16**
4		23,8%	5
3		0,0%	0
2		0,0%	0
1		0,0%	0
		pregunta respondida	**21**
		pregunta omitida	**0**

Usuarios Administradores de la Aplicación

Para el módulo de administración de la aplicación se realizaron preguntas también orientadas a la usabilidad de la aplicación, creación de páginas personalizadas, banners, entre otros. A continuación se pueden observar las preguntas realizadas.

Encuesta para administradores de GENCI - 2

1. ¿Cómo le parece la diagramación y uso de de los colores en el panel de control de la aplicación Web GENCI - 2?

- Excelente
- Bueno
- Regular
- Malo

2. ¿Qué opina sobre el módulo de particiones y dependencias?

- Excelente
- Bueno
- Regular
- Malo

3. ¿Qué le parece el módulo de páginas personalizadas?

- Excelente
- Bueno
- Regular
- Malo

4. ¿Qué opina sobre la construcción de páginas personalizadas?

- Sencilla
- Regular
- Complicada

5. ¿Cómo le parece el módulo de gestión de usuarios?

- Excelente
- Bueno
- Regular

- Malo

6. ¿Qué le parece el módulo de componentes y contenido de texto?

- Excelente
- Bueno
- Regular
- Malo

7. ¿Cómo le parece el módulo de componentes y contenido de banner?

- Excelente
- Bueno
- Regular
- Malo

8. ¿Qué opina sobre el módulo de componentes y contenido de eventos?

- Excelente
- Bueno
- Regular
- Malo

9. ¿Qué opina sobre la vista de la bitácora de usuarios?

- Excelente
- Bueno
- Regular
- Malo

10. En la escala del 1 al 5, siendo el 5 la mayor puntuación. ¿Cómo le parece el desarrollo de la aplicación Web GENCI - 2?

- 5
- 4
- 3
- 2
- 1

Resultados de Administradores de la Aplicación

La encuesta para el módulo de administración fue realizada a 9 personas y de sus votaciones se obtuvieron los siguientes resultados. La tendencia indica que existe una aceptación bastante buena de la aplicación GENCI -2.

1. ¿Cómo le parece la diagramación y uso de de los colores en el panel de control de la aplicación Web GENCI - 2?

Crear gráfico Descargar

	Porcentaje de respuestas	Cantidad de respuestas
Excelente	88,9%	8
Bueno	11,1%	1
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	9
	pregunta omitida	0

2. ¿Qué opina sobre el módulo de particiones y dependencias?

Crear gráfico Descargar

	Porcentaje de respuestas	Cantidad de respuestas
Excelente	88,9%	8
Bueno	11,1%	1
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	9
	pregunta omitida	0

3. ¿Qué le parece el módulo de páginas personalizadas?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Excelente	88,9%	8
Bueno	11,1%	1
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	9
	pregunta omitida	0

4. ¿Qué opina sobre la construcción de páginas personalizadas?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Sencilla	100,0%	9
Regular	0,0%	0
Complicada	0,0%	0
	pregunta respondida	9
	pregunta omitida	0

5. ¿Cómo le parece el módulo de gestión de usuarios?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Excelente	77,8%	7
Bueno	22,2%	2
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	9
	pregunta omitida	0

6. ¿Qué le parece el módulo de componentes y contenido de texto?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**77,8%**	**7**
Bueno	22,2%	2
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**9**
	pregunta omitida	**0**

7. ¿Cómo le parece el módulo de componentes y contenido de banner?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**77,8%**	**7**
Bueno	22,2%	2
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**9**
	pregunta omitida	**0**

8. ¿Qué opina sobre el módulo de componentes y contenido de eventos?	Crear gráfico	Descargar
	Porcentaje de respuestas	Cantidad de respuestas
Excelente	**88,9%**	**8**
Bueno	11,1%	1
Regular	0,0%	0
Malo	0,0%	0
	pregunta respondida	**9**
	pregunta omitida	**0**

9. ¿Qué opina sobre la vista de la bitácora de usuarios?		Crear gráfico	Descargar
		Porcentaje de respuestas	Cantidad de respuestas
Excelente		88,9%	8
Bueno		11,1%	1
Regular		0,0%	0
Malo		0,0%	0
		pregunta respondida	9
		pregunta omitida	0

Conclusiones

Al concluir la realización del trabajo especial de grado denominado "*Genci-2 Gestor de Contenido Modular para la Coordinación de Investigación de la Facultad de Ciencias de la UCV*" la cual está compuesta por una serie de módulos explicados en este documento, brindan la posibilidad a los usuarios administrativos designados por la Coordinación de Investigación la libertad de crear sitios web utilizando las herramientas que ofrece este sistema, como también gestionar el contenido publicado en la misma, definiendo una estructura jerárquica entre las dependencias creadas, con un acceso personalizado a través del URL, de tal manera que tanto la Coordinación de Investigación como las dependencias que la conforman cuentan con un sistema capaz de crecer según sus necesidades y mantener contenido sin necesidad de tener conocimientos o manejo de terminología técnica de programación para estas tareas.

Las necesidades planteadas por el coordinador de investigación el Dr. Pio Arias al momento del levantamiento de información, fueron cubiertas completamente, incluso en la solución entregada a la Coordinación de Investigación se ofrecen más herramientas y posibilidades de las que fueron solicitadas en un principio, ya que la idea de realizar este trabajo especial de grado no era solo solucionar una necesidad puntual, sino crear una herramienta con la que puedan crecer y se pueda seguir desarrollando a partir de la entrega que quedó en producción, por lo que el esfuerzo invertido en esta aplicación fue grande, en donde la principal motivación venia dada desde un sentimiento interno pertenencia al grupo de profesionales egresados que enorgullecen a la facultad de ciencias.

El haber utilizado una metodología de desarrollo ágil (AUP) fue vital para este trabajo, ya que gracias a la flexibilidad que propone dicha metodología, se pudieron realizar modificaciones y desarrollos rápidos sin necesidad de iterar nuevamente sobre labores que ya estaban concluidas. Debido a que el desarrollo fue realizado por una sola persona esta metodología también fue útil ya que los artefactos que se fueron realizando iban encajando de manera homogénea en la documentación de la aplicación, los cuales sin dejar la formalidad y rigurosidad, permitían ajustarse a las necesidades que surgían modulo a modulo.

El manejo de Ruby on Rails como tecnología del lado del servidor fue clave para este desarrollo, ya que la simplicidad de codificación, uso del patrón MVC nativo, y estructura del framework permitió realizar un desarrollo modular, rápido, reutilizable y escalable. El uso de gemas, el archivo de rutas y configuración aportan versatilidad a la hora de crear soluciones puntuales.

En general se puede concluir que con esta entrega se otorgan una serie de beneficios a la coordinación de investigación entre las que se pueden nombrar:

- Gestión de particiones para que cada dependencia, sub-dependencia y cuantos niveles se deseen, puedan utilizar esta herramienta de manera independiente.
- Gestión de páginas personalizadas, diagramadas y distribuidas a criterio del administrador.
- Gestión de menús de navegación independientes y enlaces personalizados tanto internos, como externos.
- Gestión de módulos de texto, banner y eventos.
- Gestión de contenidos de tipo texto con maquetación HTML, tipo banner y manejo de eventos con calendario de fecha hora y lugar.
- Gestión de usuarios validados contra el directorio central de LDAP de la facultad.
- Manejo de bitácora o log de operaciones en todas las acciones del sistema.
- Gestión de rutas del URL de manera personalizada.
- Estructura lógica jerárquica de las dependencias.
- Desarrollo modular y reutilizable.
- Desarrollo creado en lenguaje Ruby on Rails, el cual es conocido por el personal académico de la facultad de ciencias.

Sin embargo se considera que todo software y sistema puede mejorarse, por lo que el ofrecer esta herramienta abre las posibilidades de crear nuevos módulos o mejorar técnicas implementadas en este trabajo especial de grado, por lo que este software puede servir de base para continuar construyendo un gestor de paginas dinámicas y gestor de contenido modular propio de la Facultad de Ciencias de la Universidad Central de Venezuela, creado por un miembro de la Escuela de Computación.

Recomendaciones

Como recomendaciones se sugiere utilizar la aplicación en entorno de producción por un tiempo no menor a 6 meses, con la finalidad de conocer la capacidad y el alcance que puede tener el sistema, para posteriormente realizar un análisis de posibles mejoras al sistema.

Es necesario adiestrar al personal administrativo que manejará la estructura y la información de la página, para que sepan como explotar al máximo las bondades del sistema y puedan producir resultados óptimos por esta aplicación, por lo que se propone una jornada especial en donde se les expliquen las sugerencias de uso del sistema.

Debido a las diferentes necesidades que tiene la comunidad universitaria, y las ventajas que ofrece el sistema de Genci II, seguramente van a surgir nuevos requerimientos funcionales que podría ser agregados al sistema siguiendo la línea modular de desarrollo que se implementó, para así poder solucionar las necesidades que tengan los diferentes actores que interactúen con el Genci II.

Entre las recomendaciones a futuros desarrollo se pueden listar los siguientes:

- Desarrollo de Layout o constructor guiado de formatos para tipos de página. (Ejemplo: Pagina tipo Investigación).
- Desarrollo de gestor de perfiles centralizados de publicaciones del personal administrativo y docente.
- Mejoras en componentes tipo HTML y listas de archivos en el constructor de páginas personalizadas.
- Desarrollo de más estilos gráficos aprobados por la comisión pertinente de la Facultad de Ciencias.

Referencias Bibliográficas

Arias, Pio (2006). ¿Quiénes Somos?. Recuperado el 7 de Noviembre de 2010, de http://www.coordinv.ciens.ucv.ve/investigacion/quienes.php

Borges, Clemente y Rivero, Alexander (2006). *GENCI: Una Herramienta para la Presencia en Internet de Centros de Investigación*. Caracas: Trabajo Especial de Grado Universidad Central de Venezuela.

Bruce Tate, C. H. (2006). *Ruby on rails: up and running. California*, USA: O´Reilly Media.

Camejo, Rafael. (2010).*Qué es MVC? (Modelo – Vista – Controlador)*. Recuperado el 7 de Noviembre de 2010, de http://www.joserafael.com/blog/ique-es-mvc-modelo-vista-controlador/

Consejo Superior de Administración Electrónica (n.d), Arquitectura Cliente / Servidor. Recuperado el 7 de Noviembre de 2010, de http://www.csi.map.es/csi/silice/Global71.html

CMS (2008). *Selecting a Development Approach (Traducido al Español) [Documento PDF]*, Recuperado el 15 de Septiembre de 2010, de http://www.cms.gov/SystemLifecycleFramework/Downloads/SelectingDevelopmentApproach.pdf

David Thomas, D. H. (2005*). Agile Web Development with Rails*. (T. p. LLC, Ed.) USA.

Desarrollo Web. (2009). *Que es HTML 5*. Recuperado el 7 de Noviembre de 2010, de http://www.desarrolloweb.com/articulos/que-es-html5.html

Fitzgerald, M. (2007). *Ruby Pocket Reference (Primera Edición ed.).* (O. Media, Ed.) USA.

Juarez, Rafael (2005) Manejador de Contenidos. Recuperado el 7 de Noviembre de 2010, de http://www.proyecto-internet.com/diseno-de-paginas-web-glosario.php

Kenneth E Kendall, J. E. (2005). *Análisis y Diseño de Sistemas* (Sexta Edición ed.). México: Pearson Educación.

Laudon Kenneth C, L. J. (2004). *Sistemas de Información Gerencial* (Octava Edición ed.). México: Prentice Hall.

Lemus, Juan (2008), *CSS 3: más social que nunca*. Recuperado el 7 de Noviembre de 2010, de http://www.maestrosdelweb.com/editorial/css-3-mas-social-que-nunca/

Ospina, Mercy y Suarez, Sarabel (2008). *Diseño y Desarrollo de una nueva Aplicación Web para la Escuela de Computación de la Universidad Central de Venezuela con Tecnología Ruby On Rails*. Caracas: Trabajo Especial de Grado Universidad Central de Venezuela.

Pressman, R. S. (2006*). Ingeniería del Software* (Sexta Edición ed.). Mc Graw Hill.

Mozilla. (n.d). JavaScript. Recuperado el 20 de Junio de 2010, de https://developer.mozilla.org/es/JavaScript

Ruby, (n.d). *Acerca de Ruby.*Recuperado el 7 de Noviembre de 2007 de http://www.ruby-lang.org/es/about/

Silberschatz, Abraham y Korth, Henry y Sudarshan, S (2006). *Fundamentos de Bases de Datos*. Madrid: Aravaca

Simón, Mariana y Mata, Jesús (2010). *Desarrollo de una Aplicación Web para la automatización de la Unidad de Servicio Comunitario de la Facultad de Ciencias.* Caracas: Trabajo Especial de Grado Universidad Central de Venezuela.

SGBD MySQL. (n.d.). *Panorámica del sistema de gestión de base de datos MySQL*. Recuperado el 7 de Noviembre de 2010, de http://dev.mysql.com/doc/refman/5.0/es/what-is.html

Solcre. (2008), Ventajas de las aplicaciones Web[Documento PDF]. Recuperado el 7 de Noviembre de 2010, de http://www.solcre.com/files/ventajas_de_las_aplicaciones_web.pdf

Shklar & Rosen. (2009). *Web Aplication Architecture (Traducido)*. Wiley

W3C. (n.d*). Extensible Markup Language XML*. Recuperado el 7 de Noviembre de 2010, de http://www.w3.org/XML

W3C. (n,d). *Guia Breve de Servicios Web*. Recuperado el 7 de Noviembre de 2010, de http://www.w3c.es/divulgacion/guiasbreves/serviciosweb

W3C (2008). *Guia Breve de CSS*. Recuperado el 7 de Noviembre de 2010, de http://www.w3c.es/divulgacion/guiasbreves/hojasestilo

W3C. (2010). What is HTML (Traducido del Ingles). Recuperado el 7 de Noviembre de 2010, de http://www.w3.org/MarkUp/

Wikipedia. (2010). *Cliente Servidor*. Recuperado el 7 de Noviembre de 2010, de http://es.wikipedia.org/wiki/Cliente-servidor