Universidad de las Ciencias Informáticas

FACULTAD 6

Título: Extensión del NetBeans IDE para el diseño de Interfaces Gráficas de Usuario con Ext JS.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores:

Sheyla García Quintela

Félix González Martínez

Tutores:

Ing. Armando Robert Lobo

Ing. Adrián Rosales Cruz

Ing. Asdrubal A. Nicot García

La Habana, junio de 2013

"Año 55 de la Revolución"

...las ciencias no han de ser el patrimonio de unos pocos...

Félix Varela y Morales

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Firma del Autor	Firma del Tutor
Sheyla García Quintela	Ing. Armando Robert Lobo
Firma del Autor	Firma del Tutor
Félix González Martínez	Ing. Adrián Rosales Cruz
	Firma del Tutor
	Ing. Asdrubal Antonio Nicot García

DATOS DE CONTACTO

Autores:

Sheyla García Quintela

Universidad de las Ciencias Informáticas (UCI)

E-mail: squintela@estudiantes.uci.cu

Félix González Martínez

Universidad de las Ciencias Informáticas (UCI)

E-mail: fgonzalezm@estudiantes.uci.cu

Tutores:

Ing. Armando Robert Lobo

Centro de Tecnologías de Gestión de Datos (DATEC)

Universidad de las Ciencias Informáticas (UCI)

E-mail: arobert@uci.cu

Ing. Adrián Rosales Cruz

Centro de Tecnologías de Gestión de Datos (DATEC)

Universidad de las Ciencias Informáticas (UCI)

E-mail: adrianrc@uci.cu

Ing. Asdrubal Antonio Nicot García

Centro de Tecnologías de Gestión de Datos (DATEC)

Universidad de las Ciencias Informáticas (UCI)

E-mail: aangracia@uci.cu

AGRADECIMIENTOS

De Sheyla:

Durante este tiempo de universidad he vivido experiencias inolvidables que me han ayudado a superarme. Cada año fue un nuevo reto para mí, algunos con frutos buenos y otros no tanto, pero todos me permitieron llegar a donde estoy. Hoy quisiera agradecer a todas esas personas que me apoyaron con su amor incondicional y me ayudaron a enfrentar cada batalla.

A mis padres por estar siempre ahí y ser mis educadores principales, les doy las gracias por guiarme, sin ustedes no habría podido llegar hasta aquí.

A mi hermanita, que a pesar de ser la pequeña de las dos, siempre me ha ayudado a vencer mis obstáculos.

A mis abuelas por darme el apoyo material y espiritual que he necesitado.

A toda mi familia, especialmente a mi tía Lázara, mis primas Gretter y Laura y a mi tío Carlos.

A mi tío Juan, a Miriam y a Araí por protegerme siempre y apoyarme como una hija.

A Alfonso y Jenny por ser los mejores amigos del mundo, por escucharme, soportarme y siempre estar ahí.

A Dayana porque a pesar de que llegó a mi vida recientemente la considero una hermana más.

A todas mis amistades de dentro y fuera de la universidad, por apoyarme en todo momento, especialmente a Coqui, Thays, Damian, Pablo, Yeni, Yaisel, Frank Yasel, Raniel y a Viltre por ayudarme a salir de algunos apuros académicos.

A mis compañeros, con los que he compartido durante estos cinco años experiencias inolvidables y lamentablemente a muchos no los volveré a ver.

A todas mis amistades que ya no están en la universidad pero durante su estancia me apoyaron mucho, especialmente a Daniel, Javier, Freddy, Marlon y José Lázaro.

A todos los profesores que me apoyaron, especialmente a Yanelis, Yuleidys, Adonys, Omar y Edgar por confiar en mí y ayudarme siempre que los necesité.

A mis tutores Adrián, Asdrubal y Lobo, por orientarme en la realización de mi tesis.

Agradecimientos

A los todos profesores del departamento que nos ayudaron, especialmente Vania, Alberto, Yanet Parra, Yanet Rosales y Diana.

A mi compañero de tesis, por acompañarme durante esta batalla.

Pero en especial a Maikel, por ser un magnífico compañero y amigo. Lamentablemente la vida no te permitió lograr tu sueño y hoy quisiera compartir este logro contigo, siempre te recordaré.

De Félix:

A mi familia por el apoyo recibido y en especial a mis padres y a mi hermana.
A la Revolución por darme la posibilidad de estudiar y graduarme como ingeniero.
A mi compañera de tesis por asumir el reto de trabajar juntos.
A mis tutores por el apoyo recibido durante este periodo de tiempo.
A los especialistas del departamento que me permitieron evacuar dudas y en especial los profesores Diana, Alberto, Vania y Yanet Parra.
A mis amistades que me soportaron todo este tiempo como Karel, Adrian, Pablo, Dayana, Yudeily.
A profesores y estudiantes que me soportaron los 5 años y en especial en esta última etapa.
A todos ellos les debo mi formación como profesional y del resultado de este trabajo.

DEDICATORIA

De Sheyla:

Le dedico este Trabajo de Diploma a mis padres y a mi hermana, por acompañarme en todo momento y brindarme todo el amor y dedicación que he necesitado.

De Félix:

Le dedico este Trabajo de Diploma a mis padres y a mi hermana que son las personas más importantes en mi vida y a la revolución por darme esta oportunidad de estudios.

RESUMEN

La investigación surge en el departamento de Integración de Soluciones perteneciente al Centro de Tecnologías de Gestión de Datos (DATEC) de la Universidad de las Ciencias Informáticas (UCI), el cual promueve el desarrollo de soluciones que permiten el análisis, tratamiento y manipulación de la información capturada por un sistema informático. Aunque en el departamento se desarrollan las "Interfaces Gráficas de Usuario" o GUI (del inglés *Graphical User Interface*) con el Framework Ext JS que facilita el desarrollo orientado a componentes, el soporte que las herramientas de desarrollo ofrecen para su utilización bajo el concepto de "Desarrollo Rápido de Aplicaciones" o RAD (del inglés *Rapid Application Development*) es limitado. DATEC cuenta con herramientas como Ext Designer y Lycan GENESIS - Component Builder, pero estas se restringen al diseño de la GUI y a la obtención del código correspondiente, lo que disminuye la productividad de los usuarios respecto al desempeño que se alcanza con un Entorno de Desarrollo Integrado (IDE), donde se agrupan un mayor conjunto de funcionalidades. La investigación está orientada a desarrollar una extensión al NetBeans IDE que permita el diseño de GUI con Ext JS. Se realiza un estudio de la metodología a utilizar, los conceptos y herramientas asociados a las GUI. Se emplean los patrones de diseño en la confección e implementación de las clases necesarias, así como la reutilización de componentes presentes en la Plataforma NetBeans. Se realizan los casos de prueba donde se valida la implementación de los requisitos funcionales de la extensión desarrollada.

PALABRAS CLAVES
interfaces gráficas de usuario, ext js, componentes, extensión, netbeans ide.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE GRÁFICAS

INTRODUCCIÓN

El avance continuo de las técnicas de programación, potentes hardware, fuentes innovadoras de conocimientos y el empleo de metodologías de desarrollo, han logrado llevar a niveles diferentes los paradigmas sociales e informáticos. Ejemplo de ello lo constituye el avance de las tecnologías web, que ha permitido brindar diferentes servicios donde parte de su concepción teórica se funde en el concepto de "Desarrollo Rápido de Aplicaciones" o RAD (del inglés *Rapid Application Development*). Gracias a ello se enriquecen las librerías y Frameworks [1] empleadas para la creación de diferentes arquitecturas y estructuras lógicas de proyectos, soportadas por diferentes lenguajes de programación. Luego de adquirir madurez, el desarrollo de aplicaciones web, evoluciona bajo el nombre Aplicaciones Enriquecidas para la Internet o RIA (del inglés *Rich Internet Application*); donde se vincula al diseño una interfaz que relaciona la potencia de la arquitectura con el ambiente tradicional de escritorio. Dicho crecimiento ocupa parte esencial en el lenguaje de programación JavaScript y la tecnología AJAX, que fundan la base de varios Frameworks de presentación que por su potencia y flexibilidad han logrado una gran aceptación en el mercado internacional.

Entre estos Frameworks se encuentra Ext JS que ha transitado por varias versiones mejorando sus particularidades en calidad, documentación y arquitectura, permitiendo que el producto final posea mejor aceptación por los clientes debido a la originalidad de las "Interfaces Gráficas de Usuario" o GUI (del inglés *Graphical User Interface*) obtenidas. Sin embargo los calendarios propuestos pueden cumplirse cuando se desarrollan aplicaciones relativamente pequeñas, lo que dificulta en gran medida el desarrollo actual, caracterizado cada vez más por la complejidad de las aplicaciones. Para dar tratamiento a aplicaciones complejas en Ext JS fue necesaria la implementación de entornos de trabajo bajo el criterio de "Lo que ves es lo que obtienes" o WYSIWYG (del inglés *What You See Is What You Get*), facilitando el desarrollo de las GUI. Como ejemplos de estos entornos se pueden encontrar Ext Designer y Lycan GENESIS - Componet Builder, pero estos limitan sus funciones al diseño y la generación del código correspondiente en el desarrollo web lo cual es insuficiente con respecto al desempeño que se alcanza con un Entorno de Desarrollo Integrado o IDE (del inglés Integrated Development Environment).

[1] Framework: Estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, en base a la cual otro proyecto de software puede ser organizado y desarrollado.

Los mismos son casi imprescindibles para el desarrollo de proyectos con tecnologías web, los que permiten su integración con diferentes Frameworks y librerías tanto para el cliente como para el servidor, además de poseer un conjunto de funcionalidades mayor. Estos se encuentran entre las herramientas que cumplen con las características RAD, sin embargo se le adicionan nuevos lenguajes como IDE y no en todos los casos dan soporte para las GUI de los mismos.

La Universidad de las Ciencias Informáticas (UCI) como centro productivo, también hace uso de Frameworks de desarrollo en conjunto con los IDE. Ejemplo de ello lo constituye el Departamento de Integración de Soluciones del Centro de Tecnologías de Gestión de Datos (DATEC) perteneciente a la Facultad 6, el cual desarrolla soluciones informáticas con los Frameworks Symfony y Ext JS apoyados en el NetBeans IDE. Sin embargo, este último a pesar de dar soporte a varias tecnologías y lenguajes, no cuenta con facilidades para el diseño de GUI con Ext JS lo que conlleva al empleo de varias herramientas, un incremento adicional en el tiempo de desarrollo y un aumento en el consumo de recursos para desarrollar sistemas que en su estructura son complejos y técnicamente grandes.

Teniendo en cuenta lo antes expuesto se propone como **problema de la investigación:** ¿Cómo reducir el tiempo de desarrollo de Interfaces Gráficas de Usuario con Ext JS en el NetBeans IDE? El presente trabajo tiene como **objeto de estudio**: El diseño de GUI en los IDE, enmarcado en el **campo de acción**: El diseño de GUI con Ext JS para el NetBeans IDE. En búsqueda de dar solución al problema planteado se define como **objetivo general**: Desarrollar una extensión del NetBeans IDE para el diseño de GUI con Ext JS.

Objetivos Específicos

- Describir las funcionalidades para la extensión del NetBeans IDE que permita el diseño de GUI con Ext JS.
- Realizar el diseño de la extensión a partir de los requisitos identificados.
- Implementar y probar las funcionalidades en la extensión propuesta.

Tareas:

- Caracterización de las facilidades presentes en los IDEs para el diseño de GUI.
- Estudio de las herramientas que dan soporte al desarrollo con Ext JS.
- Identificación de los principios arquitectónicos, las Interfaces de Programación de Aplicaciones o APIs (del inglés Application Programming Interface) y componentes para desarrollar extensiones del NetBeans IDE.

- Definición de las funcionalidades y restricciones que debe cumplir la extensión.
- Confección del modelo de casos de uso del sistema de la extensión.
- Elaboración del modelo de diseño de la extensión.
- Elaboración del modelo de implementación.
- Confección del diseño para los casos de prueba definidos a partir de los requisitos funcionales.
- Realización de las pruebas funcionales a la extensión implementada.
- Validación de la solución propuesta.

Como **resultado** se espera obtener una extensión para el NetBeans IDE que permitirá el diseño de GUI con Ext JS siendo este un activo del departamento de Integración de Soluciones para desarrollar soluciones informáticas que lo requieran. Garantizará la obtención del código modelado y la vista previa del diseño, permitirá la técnica de arrastrar y soltar[2], además facilitará la vinculación de la GUI modelada con el proyecto web.

El trabajo de diploma está estructurado de la siguiente manera:

- **Capítulo 1** "Fundamentación teórica": Se brinda una descripción general de los conocimientos tratados, así como el estudio y análisis de algunas herramientas que existen para el diseño de GUI. Además, se brinda una descripción de las herramientas, lenguajes de programación y metodología de desarrollo de software empleadas para implementar la extensión del NetBeans IDE para el diseño de GUI con Ext JS.

- **Capítulo 2** "Diseño del Sistema": Se muestra el Modelo de Dominio para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo a la captura de requisitos y el diseño de un sistema. Se brinda una especificación de los Requisitos Funcionales (RF) y los Requisitos No Funcionales (RNF) de la aplicación. Se detallan los casos de uso representados en el diagrama de Casos de Uso del Sistema (CUS) para una mejor comprensión del funcionamiento del negocio. A través de los diagramas de clases de diseño y de secuencia se describe la realización de los CUS. Se definen los patrones de diseño y la arquitectura a utilizar.

[2] Técnica de arrastrar y soltar: Es una expresión informática que se refiere a la acción de mover con el ratón objetos de una ventana a otra o entre partes de una misma ventana.

- **Capítulo 3** “Implementación y Prueba”: Se enfoca en proveer una solución a los requisitos del sistema a través del flujo de trabajo Implementación. Se realiza el diagrama de componentes que contiene una descripción de las partes del sistema teniendo en cuenta su arquitectura y los ficheros utilizados. Finalmente se realizan pruebas al sistema para comprobar el funcionamiento de los requerimientos del sistema propuestos demostrando así el éxito de la solución propuesta.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En el presente capítulo se ofrecen varios conceptos asociados al origen y evolución de las GUI además de un análisis a las herramientas que las utilizan. Se muestra una breve descripción de los sistemas necesarios para el desarrollo de la solución propuesta en el trabajo de diploma, así como los lenguajes de programación, la metodología de desarrollo de software empleada, entre otros aspectos.

1.1 Conceptos asociados al dominio del problema

A continuación se presentan algunos conceptos fundamentales para una mejor comprensión de la investigación:

1.1.1 Desarrollo Rápido de Aplicaciones

Es un modelo de procesos del desarrollo del software lineal secuencial que enfatiza un ciclo de desarrollo extremadamente corto. Es una adaptación a “Alta velocidad” en el que se logra el desarrollo rápido utilizando un enfoque de construcción basado en componentes (1). Una vez delimitados los requisitos en el ámbito del proyecto, permite al equipo de desarrollo crear un sistema completamente funcional dentro de periodos cortos de tiempo.

1.1.2 Lo que ves es lo que obtienes

Es un editor o programa que permite al usuario ver el resultado final de su producto mientras que la interfaz o el documento se está creando. Permite su integración a cualquier tipo de editor cliente facilitando la inclusión de códigos. Se aplica además a los procesadores y otros editores de texto con formato (como los editores de HTML) que permiten escribir un documento viendo directamente el resultado final. (2)

1.1.3 Interfaz Gráfica de Usuario

Es un programa o entorno que gestiona la interacción con el usuario basándose en relaciones visuales como iconos, menús o un puntero. (3)

1.1.4 Aplicaciones enriquecidas para internet

Es un nuevo tipo de aplicaciones con grandes mejorías sobre las tradicionales aplicaciones web. Esta surge como una combinación de las ventajas que ofrecen las aplicaciones web y las aplicaciones tradicionales. (4)

1.1.5 Cliente enriquecido

En una arquitectura cliente-servidor, el término cliente enriquecido, se utiliza cuando el tratamiento de datos se produce principalmente en el lado del cliente. Además, son definidos como aplicaciones que son extensibles a través de extensiones y módulos, de esta manera son capaces de resolver más de un problema. Los clientes dinámicos son típicamente desarrollados en la parte superior de un marco. Este último ofrece un punto de partida básico sobre el que el usuario puede armar lógicamente partes relacionadas de la aplicación, que se llaman módulos. Idealmente, las soluciones no relacionadas (tales como las que están disponibles por proveedores diferentes) pueden trabajar juntos, de modo que todos los módulos parecen haber sido creados como un todo. (5)

1.1.6 Extensión

Es un módulo que le agrega características, servicios o funciones específicas a un sistema informático. Este se ejecuta desde el sistema principal e interactúan por medio de APIs. Permite a los usuarios colaborar con la aplicación principal, extendiendo sus funcionalidades y evitando que la misma se vea inmersa a cambios en sus dimensiones y funcionalidades, pero no pueden funcionar por sí mismos.

Existen múltiples sistemas que permiten ser extendidos, debido a que en su base no llevan consigo todas las funcionalidades que los usuarios puedan requerir. Un ejemplo de ello es la Plataforma NetBeans que posee una base modular y extensible, usada como una estructura de integración para crear aplicaciones de escritorio grandes. Las empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones. (6)

- **Elementos arquitectónicos para extender el NetBeans IDE**

EL NetBeans IDE posee ventajas genéricas de una plataforma de cliente enriquecido, ofrece numerosos marcos y varias características adicionales que pueden ser particularmente útiles para las aplicaciones, además posee gran adaptabilidad para el usuario final, presenta una arquitectura modular.

Este está construido sobre la Plataforma NetBeans, la misma se basa en normas internacionales y componentes reutilizables que pueden ser empleados en aplicaciones propias del usuario, brindando elementos de interfaz como ventanas, menús, barras de herramientas y otros componentes como APIs, sistemas de ayuda, internacionalización y representación de datos. Esto es logrado mediante la construcción de módulos que se acoplan gracias a la arquitectura modular.

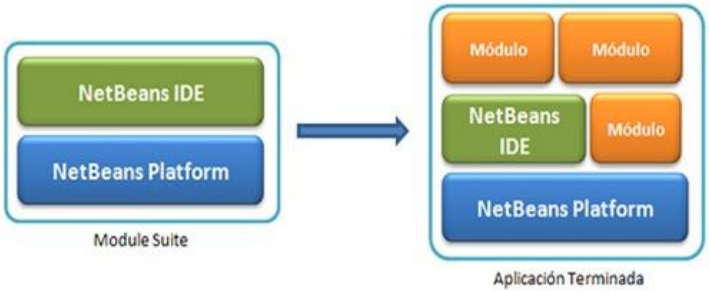


Figura 1: Arquitectura modular para el desarrollo de aplicaciones con NetBeans IDE y la plataforma NetBeans.

¿Cómo se puede extender mediante módulos el NetBeans IDE?

El NetBeans IDE soporta un acoplamiento flexible de los módulos, que es descrito por los datos de su fichero de manifiesto, junto con los datos especificados en los archivos relacionados con XML.

La estructura de un módulo es un archivo Requerimiento de Aplicación Conjunta o JAR (del inglés *Joint Application Requirement*) sencillo que por lo general consta de las siguientes partes:

- Manifiesto de archivo (manifest.mf).
- Capa de archivo (layer.xml).
- Los archivos de clase.
- Los recursos como iconos, Bundles.properties, etc.

Manifest.mf: Es una descripción textual del módulo y su entorno. Cuando se carga un módulo, este es el primer archivo leído por el sistema de módulos.

Layer.xml: Proporciona información específica del usuario y define la integración de un módulo en la plataforma. El mismo detalla todo lo que un módulo quiere añadir a la plataforma, que va desde las acciones de los elementos de menú a los servicios. Además, este formato es un sistema jerárquico que contiene las carpetas, archivos y atributos.

A continuación se muestra parte de la configuración del archivo layer.xml para el menú de una ventana.

```
<folder name="Menu">
    <folder name="Edit">
      <file name="MyAction.shadow">
          <attr name="originalFile"
            stringvalue="Actions/Edit/com-galileo-netbeans-module-MyAction.instance"/>
          </file>
    </folder>
</folder>
```

Otros aspectos de interés en la estructura de los módulos, es el diseño de clases que desarrolla el programador para garantizar el funcionamiento de la solución propuesta y por último se encuentra el archivo denominado Bundle.properties permite localizar la información registrada en el archivo de manifiesto.

Una vez comprendido el funcionamiento de los módulos se puede crear un proyecto para extender las funcionalidades del NetBeans IDE, el entorno provee de facilidades que permiten estructurar un proyecto que parte de una plantilla que se puede estructurar de la siguiente forma.

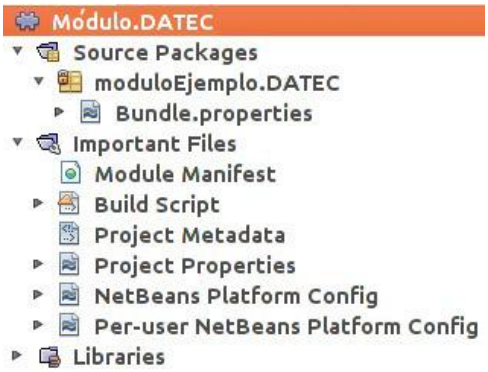


Figura 2: Estructura de un proyecto para extender al NetBeans IDE.

1.1.7 APIs

Son un conjunto de funciones que facilitan el intercambio de mensajes entre dos sistemas, es decir, que permiten la integración de diversos módulos a sus aplicaciones principales. Una API estable, permite que complementos de terceros funcionen como la versión original y amplíen el ciclo de vida en aplicaciones obsoletas, además, toma la información y hace que el trabajo de comunicación sea transparente. Ofrece al programador un cierto nivel de abstracción que enmascara la complejidad de acceso a un sistema o

aplicación, proponiéndole un conjunto de funciones de las cuales sólo se conocen los parámetros y los valores devueltos. La interfaz provee la ventaja de que no necesita preocuparse de cómo funciona una aplicación remota ni de la forma en que las funciones fueron implementadas para utilizar un programa. (7)

1.1.8 Componentes

Se comportan como entidades básicas de la programación, dentro de la Programación Orientada a Componentes (POC), que pueden ser interpretadas como cajas negras que encapsulan cierta funcionalidad y que son diseñadas sin saber quién las utilizará. En la POC los servicios de los componentes son conocidos mediante sus interfaces y requisitos; además, el objetivo de este tipo de programación es construir un mercado global de componentes de software, donde existen usuarios de aplicaciones que necesitan reutilizar componentes ya hechos y testeados para construir sus aplicaciones de forma más rápida y robusta.

Dentro del desarrollo de software, es una unidad reutilizable que puede interoperar con otros módulos de software por medio de sus interfaces, además, puede presentarse en forma de código fuente o código objeto. El desarrollo de software basado en componentes es uno de los mecanismos más efectivos para el diseño y construcción de grandes aplicaciones de software. (8)

1.2 Herramientas para el diseño de GUI con Ext JS

1.2.1 Antecedentes de las GUI con el Framework Ext JS

Ext JS es un Framework de presentación para desarrollar RIA. Inicia como proyecto formando parte de una extensión de la librería de Interfaz de Usuario de Yahoo o YUI (del inglés *Yohoo User Interface*), pero debido a su evolución se convierte en un producto sólido permitiendo la utilización de extensiones que soportan el desarrollo con las bibliotecas JQuery, YUI y Prototype sobre el núcleo de Ext JS. Permite al implementador hacer uso de componentes personalizables además de poseer un modelo de componentes extensibles, un API fácil de usar formando parte de la ayuda al usuario y provee una serie de componentes que se pueden incluir en las páginas web. A continuación, en la Tabla 1, se observan algunos de estos componentes y sus principales funcionalidades:

Componentes	**Funcionalidades**
Cuadros y áreas de texto	Introducir información alfanumérica.
Campo para fechas	Desplegar un panel de selección donde se muestran de forma interactiva los días, meses y años.
Campo numérico	Introducir datos numéricos.
Combo	Mostrar en una lista desplegable un conjunto de información.
Radiobutton y checkbox.	Seleccionar información a través de elección simple o múltiple.
Editor HTML	Gestionar código HTML
Árbol de datos	Mostrar información según la jerarquía declarada.
Pestañas	Permitir organizar la información de forma interactiva.
Barra de herramientas.	Mejorar la navegación en la aplicación desarrollada y la hace más cómoda al usuario.
Menú al estilo de Windows	Dividir las funcionalidades del producto a desarrollar.
Cuadro de diálogo	Establecer un diálogo entre la máquina y usuario, para notificaciones y entrada de información

Tabla 1: Ejemplo de componentes del Framework de desarrollo Ext JS

Los componentes del Framework Ext JS poseen atributos y métodos que los hacen personalizables e incluso extensibles, lo que favorece el desarrollo de las GUI por los implementadores, pues para el lanzamiento del Framework los creadores no brindaron una herramienta que facilitara el trabajo y permitiera obtener soluciones rápidas. Las GUI implementadas requerían un monto de tiempo considerable para su desarrollo y un posterior cambio en el diseño significaba comenzar a realizar las mismas desde cero o perder parte del trabajo realizado.

Para los productos de software desarrollados con Ext JS es necesario tener en cuenta aspectos que influyen en los clientes, elementos relacionados con la interfaz de la solución, la retroalimentación, tratamiento de posibles errores y disminuir la necesidad de memorización entre otros aspectos. El auge vertiginoso del Framework se debe al desarrollo bajo el concepto de RIA. La calidad de los productos, la seguridad que brinda entre otros aspectos de usabilidad, decidieron que se buscaran alternativas que favorecieran el desarrollo donde el avance se centra en herramientas que permiten el diseño de la GUI y la obtención de código modelado, lo que constituye un paso trascendental en esta vertiente.

- **Componentes de Ext JS para la extensión**

A partir de un estudio realizado a los productos del departamento Integración de Soluciones, se determinaron los componentes más utilizados para el diseño de las GUI. Para implementar la extensión fue necesaria la selección de algunos de estos, por lo cual se aplicó una encuesta de Muestreo Intencional (Ver Anexo 2) a 10 programadores del proyecto con al menos 1 año de experiencia de trabajo.
Los resultados de la investigación se muestran a continuación en la Tabla2:

Componentes de Ext JS para la extensión.	
➢ **Formulario**	➢ **Editor de Texto**
Label	EditorPanel
Button	
CheckBox	➢ **Tabla**
RadioButton	Table
RadioGroup	
ComboBox	➢ **Contenedores**
List	Panel
TextField	Form Panel
TextArea	
PasswordField	

Tabla 2: Componentes de Ext JS para la extensión al NetBeans IDE.

1.2.2 Extensiones de Ext JS para los IDEs

- **JCode**

La extensión JCode elaborada para la construcción de formularios con Ext JS para el NetBeans IDE según la definición del propietario es: *Una herramienta para el desarrollo de aplicaciones Java y J2EE formas web utilizando el Ext JS JavaScript Marco*. Esta herramienta reduce el tiempo de desarrollo al 90% del tiempo, sólo se tiene que conectar a la base de datos. (9) Está publicada bajo Licencia Pública General o GPL (del inglés *General Public License*) y se distribuye en la categoría de aplicaciones web, la extensión es propiedad de Vidal Pipa.

- **Extensión de Eclipse y Aptana**

Se utiliza en los archivos del proyecto embebidos en los script[3] para crear el código. Se incluyen opciones heredadas de las clases base y la documentación del mismo, no da posibilidades de editar el CSS[4] pero este se puede redefinir en las páginas donde se trabaje o crear nuevas reglas.

Aptana: Es un IDE para el desarrollo de JavaScript. Ext JS está basado en este lenguaje y garantiza funcionalidades como el completamiento de código y la detección de errores sintácticos. Aptana permite además la programación de CSS y HTML que facilitan el desarrollo de páginas web.

Eclipse: Es una potente herramienta de desarrollo de software libre que permite la implementación de aplicaciones en varios lenguajes de programación como JAVA, HTML, PHP entre otros. Se extiende para agregar las funcionalidades del Ext Designer de Sencha, garantizando un mejor rendimiento al trabajar con el Framework Ext JS.

1.2.3 Herramientas informáticas para el diseño de GUI con Ext JS

Con el devenir tecnológico se materializan ideas de empresas y usuarios de software que proveen soluciones informáticas para el diseño de GUI con Ext JS. Ejemplo de estas aplicaciones se pueden citar: Ext Designer, Lycan GENESIS - Component Builder, Sencha Architect.

- **Ext Designer**

Es una herramienta multiplataforma que requiere un servidor web para su ejecución. Ayuda a crear GUI para proyectos. Emplea la técnica de arrastrar y soltar, además, entre sus funcionalidades se encuentra la obtención del código modelado en la interfaz; presenta un inspector de propiedades sobre los componentes seleccionados en el área de diseño, cuenta con una paleta de herramientas y facilidades en cuanto a la visualización del código. Está basado en los siguientes lenguajes: Lenguaje de Marcado Hipertextual o HTML (del inglés *HyperText Markup Language*), HTML Dinámico o DHTML (del inglés *Dynamic HTML*), Hipertexto Pre-procesador o PHP (del inglés *Hypertext Pre-Processor*) y es en su

[3] Script: Es un guión o conjunto de instrucciones. Permiten la automatización de tareas creando pequeñas utilidades.
[4] CSS2: Las hojas de estilo en cascada (en inglés *Cascading Style Sheets*), es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML.

totalidad propietaria. Para garantizar el empleo sin litigios legales es necesaria la adquisición de su licencia.

- **Lycan GENESIS - Component Builder**

Es un entorno para el desarrollo de aplicaciones enriquecidas para la web sobre las tecnologías Ext JS, SQLite y Symfony soportado sobre el lenguaje PHP. Permite el desarrollo de componentes de una forma rápida y cómoda, se reutiliza y configura para desarrollos más especializados en otros dominios. Es una herramienta multiplataforma que posee el respaldo institucional del Centro DATEC de la UCI.

- **Sencha Architect**

La herramienta creada por Sencha[5] es el sucesor de Ext Designer, capacitado para la construcción de GUI con Ext JS. Añade funcionalidades que permiten la creación de aplicaciones para Sencha Touch[6]. Sencha Architect brinda flexibilidades de diseño como arrastrar y soltar elementos de la paleta de componentes, además, integra un navegador de componentes y un visor de propiedades. Ofrece almacenes de copias de seguridad para conexiones a orígenes de datos.

- **Análisis de las principales limitaciones**

Ext Designer y Sencha Architect son herramientas propietarias de Sencha, pues para hacer uso de las mismas, la UCI debería pagar por la adquisición de la licencia tanto así como los compradores estarían sujetos a las políticas de uso de estas. Además, el país está inmerso en la política de migración al software libre en busca de la soberanía tecnológica guiado por el proyecto de Servicios Integrales de Migración, Asesoría y Soporte (Simays) y con apoyo de la dirección de la Revolución, por lo que se incurriría en gastos excesivos y en contradicción con el desarrollo de dicho proceso.

- **Políticas de desarrollo de DATEC**

Existen herramientas que son de código abierto como Lycan GENESIS - Component Builder, la cual permite el diseño y la obtención del código modelado en la interfaz pero no responde a las necesidades actuales de DATEC, pues tiene como guía para el desarrollo el empleo de componentes que permitan ser

[5] Sencha: Empresa propietaria de Ext JS y de las herramientas Sencha Architect en sus variantes, de escritorio y para la web.
[6] Sencha Touch: Primer móvil HTML5 marco de JavaScript para aplicaciones web que se sienten nativa en Android, BlackBerry, iOS.

reutilizados en otras soluciones y para el desarrollo de estos es necesario vincular las capacidades de diseño de GUI con Ext JS a un IDE en aras de obtener mejores resultados en el desarrollo de soluciones web.

Las herramientas analizadas presentan limitaciones que podrían afectar el desarrollo productivo del departamento de Integración de Soluciones de DATEC; sin embargo poseen aspectos que ayudan al implementador. Entre estos se encuentran elementos como la interfaz, donde se destaca el uso de la paleta de componentes que permite la técnica de "Arrastrar y soltar", el uso de los inspectores de propiedades, el área de trabajo y capacidades de modificar el tamaño de los componentes. Además funcionales, como salvar el diseño realizado y exportar el código de la interfaz en código con estructura de componente o sin ella.

Las funcionalidades antes mencionadas son ejemplificadas en el Anexo 1: Aspectos positivos de las herramientas de diseño de Ext JS.

1.2.4 Tecnologías y Herramientas para el desarrollo

- **Proceso de Desarrollo.**

Las metodologías de desarrollo de software facilitan el uso de guías, técnicas y herramientas así como brindan soporte en el ciclo de vida del software, además de sincronizar los recursos humanos y materiales para aumentar las probabilidades de éxito, permitiendo que se describan las actividades y procedimientos. Para desarrollar la extensión del NetBeans IDE es necesario el empleo de alguna de estas metodologías que garanticen flexibilidad en la construcción de la herramienta, la obtención de los artefactos y que el producto final cumpla con los estándares de calidad. Luego de un minucioso estudio realizado sobre la base del diagrama de Boehm y Turner[7], coincidió la metodología de desarrollo con la propuesta por la línea base de la arquitectura del departamento de Integración de Soluciones de DATEC, OpenUP.

Esta se caracteriza principalmente por ser una metodología ágil con un enfoque centrado al cliente, se destaca por ser un proceso extensible dirigido principalmente a gestionar y desarrollar proyectos de software basados en un ciclo iterativo por lo que permite la detección temprana de errores. Se emplea mayormente en proyectos pequeños con pocos recursos, logrando disminuir las probabilidades de fracaso e incrementando las probabilidades de éxito. Permite ser aplicada en diferentes plataformas y aplicaciones

[7] Barry Boehm y Richard Turner: proponen un modelo de fácil comprensión capaz de evaluar cuantificar e identificar cinco variables críticas a la hora de decidir si el desarrollo de un sistema se aventura por metodologías ágiles o robustas.

de desarrollo, disminuye los artefactos generados con respecto a otras metodologías de desarrollo de software y presenta un desarrollo evolutivo centrado en la arquitectura desde edades tempranas.

➢ Lenguajes de programación

El avance continuo de la informática está potenciado por los lenguajes artificiales (lenguajes de programación) que permiten declarar sentencias o instrucciones para que estas sean procesadas por el ordenador, logrando comportamientos o cambios en parámetros físicos o lógicos denotados a través de reglas semánticas y sintácticas que enmarcan elementos, expresiones y además su estructura. Existen diferentes lenguajes de programación que varían su utilidad según sus características y estilos, permitiendo a los usuarios implementar soluciones informáticas con disímiles fines sociales. Para implementar la extensión del NetBeans IDE es necesario el empleo de varios lenguajes de programación entre los que se encuentran:

- **JAVA**

Fue desarrollado por Sun Microsystems basado en el paradigma Orientado a Objetos (OO) cumpliendo con características como encapsulación, herencia y polimorfismo. Es un lenguaje de alto nivel que junto a la Máquina Virtual de Java o JVM (del inglés *Java Virtual Machine*), permite que los proyectos sean ejecutados en diferentes entornos de hardware o software, además incluye en su concepción filosófica el colector de basura, facilitando que los implementadores se abstraigan del ciclo de vida de los componentes. Sus usos se extienden desde dispositivos móviles, sistemas empotrados, aplicaciones de servidor y clientes, entre otras. El uso de sentencias permite escribir programas de código abierto bajo la patente que rige la GPL de GNU. Para el desarrollo de proyectos con un grado de complejidad técnica según las necesidades de estos, se puede hacer uso de multihilos, que garantiza la realización de actividades simultáneas logrando un resultado en cuanto a rendimiento y respuestas del sistema. Se vinculan las técnicas de compilación (intérprete, compilador) las cuales se ejecutan sobre la JDK[8], además, permite la búsqueda y tratamiento de errores en el tiempo de ejecución por lo que se considera un lenguaje robusto.

[8] JDK: Es un software que provee herramientas de desarrollo para la creación de programas en Java.

El resultado de la investigación permitirá, mediante el diseño en el NetBeans IDE, la generación del código orientado a componentes con Ext JS de GUI utilizando la versión 1.6 de java. Esto se obtendrá a partir de escribir el código fuente en el lenguaje:

- **Java script**

Es interpretado por el navegador. Ha logrado mejoras en la interfaz y en la implementación de funcionalidades como la validación y el intercambio de datos con el servidor, la realización de cálculos sencillos y comunicación con el usuario de forma interactiva mediante diálogos. Permite el desarrollo OO basado en prototipos y el intercambio de información con la página web a través del Modelo de Objetos del Documento o DOM (del inglés *Document Object Model*), además de capturar los eventos generados y manipular las ventanas y tiene como característica que las variables son genéricas. Para la creación de la extensión se utilizará la versión 1.2.

➢ Entorno de desarrollo

Para la implementación de la extensión se utilizó el NetBeans IDE. Esta es una herramienta de código abierto bajo GPL que permite el desarrollo de aplicaciones con características modulares. Entre sus funcionalidades permite escribir, depurar, compilar y ejecutar programas, además, soporta otros lenguajes de programación. Se destacan funcionalidades como el auto-completamiento de código, capacidades para el diseño de GUI y la integración de Frameworks de desarrollo. Es un software multiplataforma y extensible que además posee abundante documentación referente a las clases y APIs que lo conforman.

La herramienta NetBeans IDE desarrollada sobre la Plataforma NetBeans[9] posee las ventajas de un cliente enriquecido, donde la mayoría de las aplicaciones tienen características semejantes visibles a través de menús, barras de herramientas, visualizaciones de progreso, representaciones de datos y otros componentes, potenciando el desarrollo de las interfaces de usuario. La plataforma permite hacer uso de varios elementos que facilitan la construcción de extensiones y aplicaciones con arquitectura modular. Entre estos se encuentra la internacionalización, el editor de datos para agregar funcionalidades, la personalización de los elementos de la pantalla y un generador de ayuda para el proyecto que se desarrolla en conjunto con las APIs y componentes reutilizables.

[9] La Plataforma NetBeans es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes.

- **Framework**

Ext JS en sus versiones 2 y 3, es un Framework de JavaScript bajo GPL y licencia comercial. Favorece la construcción de aplicaciones complejas para la web, donde se incluyen componentes de GUI personalizables, además de modelos de componentes extensibles, y como ayuda presenta un API donde declara la jerarquía de clases, atributos y métodos. Incluye la tecnología AJAX para garantizar la comunicación de forma asíncrona a través de la arquitectura Cliente-Servidor y ofrece ventajas en la creación de aplicaciones RIA empleando componentes predefinidos, permite compatibilidad con navegadores que soportan el lenguaje JavaScript.

- **Lenguaje de Modelado**

El Lenguaje Unificado de Modelado o UML (del inglés *Unified Modeling Language*), es un estándar industrial respaldado por la OMG[10] y un lenguaje que permite especificar, construir, documentar y visualizar el modelado de clases. Además de incluir los procesos de negocio y funciones del sistema, define las APIs o componentes reutilizables, la construcción de los diagramas de clases, los componentes del sistema, despliegue e interacción. Cuenta con capacidades para especificar comportamientos brindando apoyo a las metodologías de desarrollo de software abstrayéndose de cualquier sistema informático, soporta un diseño Orientado a Objetos, donde se denotan en la representación visual del modelo aspectos como la visibilidad de los atributos o métodos, las relaciones entre clases y la variedad existente de las mismas. La versión UML que se utilizará es la 2.0.

- **Herramientas CASE**

Visual Paradigm for UML es un programa de computador implementado en lenguaje Java que permite el modelado de UML, presenta licencia gratuita y comercial, es de tipo multiplataforma y brinda soporte al ciclo de vida del software. Permite el diseño de artefactos que complementan las metodologías de desarrollo, posee información en la ayuda respecto al lenguaje de modelado UML y al funcionamiento de la herramienta, cuenta con capacidad para la realización de ingeniería inversa, tiene soporte para ORM y la generación de código a varios lenguajes de programación. Puede ser extendida para agregar funcionalidades y favorece el desarrollo colaborativo. La versión que se utilizará para la creación de la extensión es la 8.0.

[10] Object Management Group (OMG) es un consorcio dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos.

1.3 Métodos para el análisis multicriterio

Durante el proceso investigativo se hace necesario verificar que la solución propuesta mejora el desarrollo de GUI con el Framework Ext JS en el NetBeans IDE. Para ello existen un grupo de procedimientos basados en el criterio de expertos con el objetivo de unificar la opinión de los mismos, estos aprueban la realización de un análisis multicriterio permitiendo evaluar proyectos y planes considerando simultáneamente diversos criterios heterogéneos. (13) Entre los métodos que permiten realizar este proceso se encuentran Macbeth, Scoring, Promethee y Delphi. Este último, está basado en la organización de un grupo de expertos de modo individual, a partir de la aplicación de un cuestionario y con el propósito de obtener un consenso general o los motivos discrepantes entre estos. Los expertos, seleccionados previamente, se someten a una serie de interrogantes sucesivas, cuyas respuestas se procesan estadísticamente para conocer la coincidencia o discrepancia que estos tienen en cuanto a lo consultado.

Una vez finalizada la extensión, se hace necesaria la utilización del método antes mencionado, con el fin de que un grupo de expertos, luego de evaluar diferentes criterios, expresen a través de un cuestionario, el tiempo que demora el desarrollo de GUI con Ext JS sólo a código y haciendo uso de la misma. El mismo permitirá en una última fase del proceso, determinar un promedio de tiempo basado en los resultados de los cuestionarios y comprobar a su vez si se reduce el tiempo de desarrollo.

Delphi plantea los siguientes principios:

- Anonimato: los expertos contestan las preguntas sin consultarse mutuamente.
- Retroalimentación controlada: después de cada ronda de preguntas, se tabulan las respuestas y se procesan, para que los participantes puedan evaluar los resultados.
- Respuesta estadística del grupo: el procesamiento de cada ronda se realiza con métodos estadísticos. Esto es la característica más importante que diferencia a este método de otros subjetivos.

Además está compuesto por 4 fases:

- Fase 1: Formulación del problema

En esta etapa es importante definir con exactitud el campo de investigación, siendo necesario estar seguros de que los expertos elegidos tengan la misma noción de este campo.

- Fase 2: Selección de expertos

En esta etapa el experto será elegido por tener conocimiento del tema consultado, con experiencia que garanticen la confiabilidad de los resultados y su capacidad de encarar el futuro. Las opiniones de los expertos pueden ser recogidas por vía postal o electrónica, así se obtiene una opinión real de cada uno.

- Fase 3: Elaboración y lanzamiento de los cuestionarios

Los cuestionarios se elaboran facilitando que los mismos puedan ser respondidos por los consultados, de manera que las respuestas puedan ser cuantificadas, donde serán relativas al grado de ocurrencia (probabilidad) y de importancia (prioridad).

- Fase 4: Desarrollo práctico y explotación de resultados

Se envía el cuestionario a los expertos elegidos, donde va acompañado de una presentación que incluye las finalidades del método Delphi, las condiciones, términos, plazo de la encuesta y la garantía del anonimato. El objetivo del cuestionario es disminuir la dispersión de opiniones y precisar la opinión media de los consultados.

Conclusiones

La investigación realizada sobre los conceptos principales para el análisis de la situación antes expuesta, fue el punto de partida para lograr una mejor comprensión durante el desarrollo de la extensión. A partir de un estudio realizado a las herramientas existentes para el desarrollo de las GUI, se detectaron algunas ventajas y limitaciones, sin embargo, siguiendo las políticas del centro de desarrollo, se hace necesaria la creación de una extensión que cubra las necesidades actuales del departamento. Además, fue seleccionada OpenUP como metodología de desarrollo para guiar el ciclo de vida de la extensión y el método Delphi con el fin de unificar los criterios de un grupo de expertos para comprobar si se reduce el tiempo de desarrollo de GUI con el Framework Ext JS en el departamento.

CAPÍTULO 2: DISEÑO DEL SISTEMA

Durante este capítulo se realiza un análisis que facilita la definición de los requisitos funcionales y no funcionales, representados a través de diagramas de CUS y su descripción, para una mejor comprensión del funcionamiento de la extensión. Además se representa el modelo de dominio, los diagramas de clases y diseño del caso de uso arquitectónicamente significativo.

2.1 Modelo de Dominio

El modelo de dominio es una representación de clases relacionadas entre sí, que permiten representar los conceptos más significativos del problema a resolver. Además, también se utiliza de guía para que los Usuarios tengan un mejor entendimiento de la posible solución.

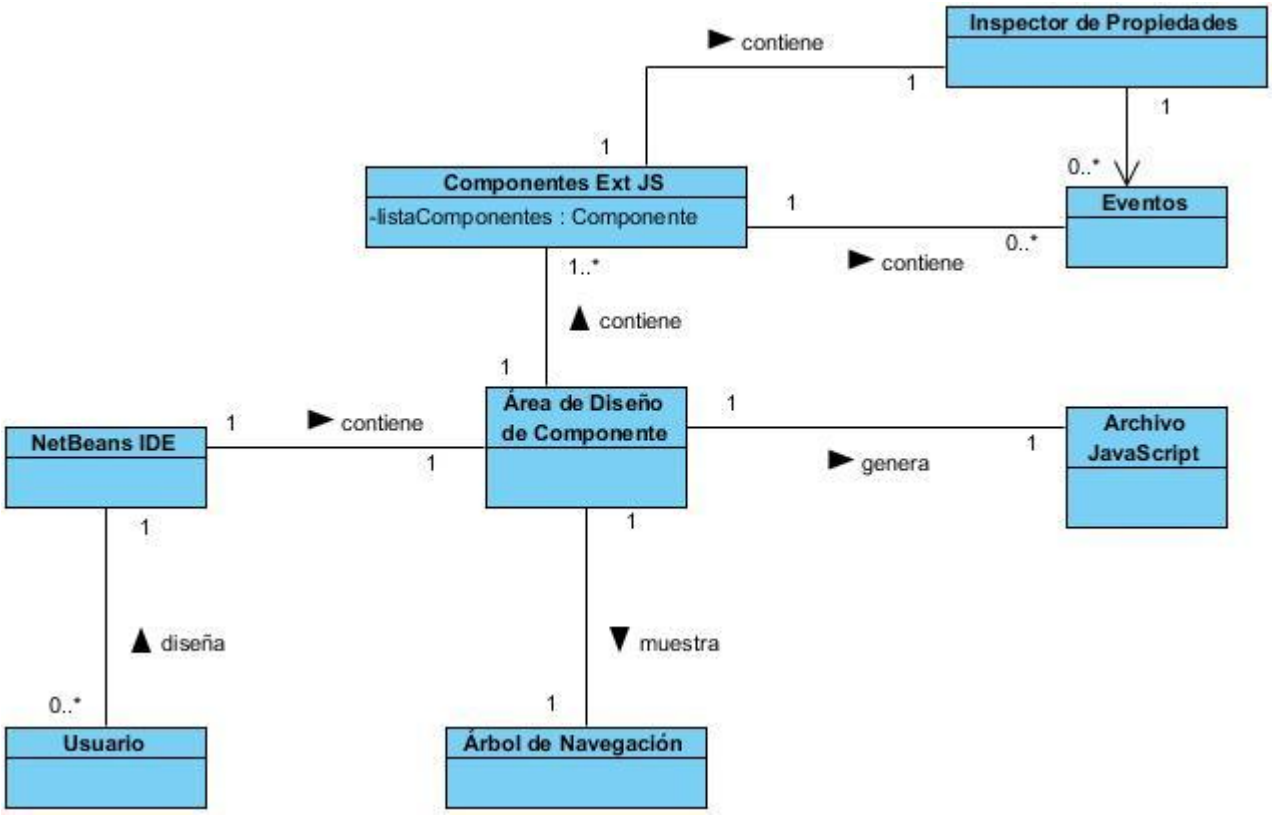


Figura 3: Modelo de Dominio

A través del diagrama de dominio representado anteriormente, se expresa el entendimiento ganado en el área bajo análisis como paso previo a la captura de requisitos y al diseño del sistema. Este modelo parte del área de diseño de componentes, que es donde el Usuario va a modelar su interfaz. Dicha área contiene componentes Ext JS que serán creados por el Usuario en su diseño, los cuales tienen asociados un inspector de propiedades que lo caracteriza y es administrado a través de eventos. Cuando el Usuario diseña una interfaz, se genera automáticamente el árbol de navegación que responde a la estructura del componente generado.
A continuación se describen explícitamente las clases que componen el diagrama de dominio:

- **Usuario**: Es el actor del sistema encargado de realizar el diseño de GUI con Ext JS en el NetBeans IDE.

- **NetBeans IDE**: Aplicación que será extendida para el diseño de GUI con Ext JS. A través de ella el Usuario puede trabajar en el área de diseño y aplicar las funcionalidades que le agrega la extensión.

- **Área de diseño de componente**: Constituye una parte fundamental para el desarrollo de la extensión. Permite la confección del diseño de componentes con el uso de la técnica arrastrar y soltar. Las modificaciones que se realicen en esta área influyen directamente en el árbol de navegación y en el Archivo JavaScript que se genera luego de creada la interfaz.

- **Componente Ext JS**: Clase que representa el componente del Framework Ext JS. A partir de su utilización se generará el código del archivo JavaScript correspondiente al mismo. Cada componente posee un conjunto de propiedades que varían en dependencia de las especificaciones del Usuario.

- **Eventos**: Controla el comportamiento de los componentes en el área de diseño y del inspector de propiedades garantizando la relación y actualización de la interfaz o área de trabajo.

- **Árbol de navegación**: Muestra los componentes del área de diseño en un orden lógico. Permite al Usuario ubicar aspectos referentes a la clasificación de los componentes. Controla componentes y

contenedores además de quién contiene a quién. Demuestra de forma rápida el nombre de acceso a determinados componentes evitando que sea necesario consultar al inspector de propiedades.

- **Inspector de propiedades**: Muestra las propiedades de los componentes permitiendo que sean editados y facilitando al Usuario actualizar los cambios realizados en el área de diseño. Gestiona la visualización de las propiedades de los componentes asociados a los Framework de Ext JS.

- **Archivo JavaScript**: Contiene el código JavaScript generado del diseño de interfaces gráficas creado por el Usuario y se actualiza mediante los cambios realizados en el área de diseño.

2.2 Requisitos

Los requisitos para un sistema son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Representan una condición o capacidad que necesita el usuario para resolver un problema o un objetivo (12). Además, se comportan como características que debe cumplir un sistema para satisfacer una norma o contrato.

2.2.1 Requisitos Funcionales

Especifican lo que el sistema debe hacer, además, dependen del tipo de software que se desarrolle, de los posibles usuarios del software y del enfoque general tomado por la organización al redactar los requisitos. Describen con detalle la función del sistema, sus entradas, salidas, excepciones, etcétera.
Los requisitos funcionales por casos de uso (CU) pertenecientes a la extensión a desarrollar son:

- **CU: Gestionar archivo**

RF1: Crear archivo

Descripción: El Usuario introduce los datos necesarios y crea un nuevo archivo, donde se guardará el código JavaScript correspondiente al diseño del componente realizado.

RF2: Modificar archivo

Descripción: El Usuario presiona clic derecho sobre el archivo JavaScript seleccionado y en la opción "propiedades" modifica el archivo.

RF3: Eliminar archivo

Descripción: El Usuario presiona clic derecho sobre el archivo JavaScript seleccionado y en la opción "eliminar" elimina el archivo seleccionado.

- **CU: Gestionar componente**

RF4: Crear componente

Descripción: El Usuario selecciona el componente a utilizar y lo arrastra hacia el área de diseño.

RF5: Modificar componente

Descripción: El Usuario especifica las propiedades del componente en el inspector de propiedades.

RF6: Eliminar componente

Descripción: El Usuario presiona clic derecho sobre el componente que desea eliminar y en la opción "eliminar" elimina el componente seleccionado.

RF7: Mostrar árbol de navegación

Descripción: Se muestra el árbol de navegación correspondiente a la estructura del diseño de componentes realizado por el Usuario en el área del diseño.

RF8: Generar archivo

Descripción: El sistema genera el código Ext JS correspondiente al diseño de componentes realizado en el área de diseño.

RF9: Exportar archivo

Descripción: El Usuario presiona el botón exportar habilitado en el área de diseño, luego selecciona la dirección hacia donde desea exportar el archivo creado.

- **CU: Seleccionar vista**

RF10: Seleccionar vista de diseño

Descripción: El Usuario selecciona la vista de diseño donde va a trabajar.

RF11: Seleccionar vista de código

Descripción: El Usuario puede seleccionar la vista de código para observar el código generado del diseño de componentes realizado.

2.2.2 Requisitos no Funcionales

Son requisitos que imponen restricciones en el diseño o la implementación así como estándares de calidad. Definen propiedades o cualidades que el producto debe tener. (14) Se refieren a las propiedades emergentes del sistema como la finalidad, software y hardware.

RNF de Software

- Se debe instalar la herramienta NetBeans IDE de la versión 7.2 o superior.
- Máquina Virtual de Java 1.6.0_27 y 1.7.0_15.

RNF de Hardware

- **RNF1 Prestaciones de la PC Cliente**

  Las PC clientes deben cumplir con los requerimientos mínimos que se especifican en dependencia del sistema operativo instalado como se muestra a continuación:

  - Microsoft Windows:
    - Procesador: Intel Pentium III o equivalente a 800 MHz
    - Memoria: 512 MB
    - Espacio en disco: 750 MB de espacio libre en el disco
  - Ubuntu:
    - Procesador: Intel Pentium III o equivalente a 800 MHz
    - Memoria: 512 MB
    - Espacio en disco: 650 MB de espacio libre en el disco

RNF de Diseño e Implementación

La herramienta para el desarrollo es NetBeans IDE de la versión 7.2, donde se utilizarán los lenguajes de programación para la implementación Java y Java Script de las versiones 1.6.0_27, 1.7.0_15 y 1.2 respectivamente; además del uso de la herramienta CASE Visual Paradigm 8.0 y el lenguaje de modelado UML 2.0.

RNF de Portabilidad

La extensión debe ser visible y usable desde los sistemas operativos Ubuntu y Microsoft Windows. Esto ocurre debido a que la aplicación que la porta, NetBeans IDE 7.2, es multiplataforma.

RNF de Usabilidad

- **RNF2 Tipo de aplicación informática**

  El Usuario que va a trabajar con la extensión debe tener experiencia en el rol de desarrollador o diseñador, así como haber trabajado en ocasiones anteriores con la herramienta de desarrollo NetBeans IDE.

- **RNF3 Finalidad**

  Contribuir al trabajo de los programadores del departamento, facilitando el diseño de componentes con Ext JS en el NetBeans IDE.

2.3 Diagramas de Caso de Uso del Sistema

Los diagramas de casos de uso documentan el comportamiento de un sistema desde el punto de vista del Usuario. Por lo tanto los casos de uso determinan los requisitos funcionales del sistema, es decir, representan las funciones que un sistema puede ejecutar. Se evidencia el patrón CRUD total en los casos de uso "Gestionar archivo" y "Gestionar componente".

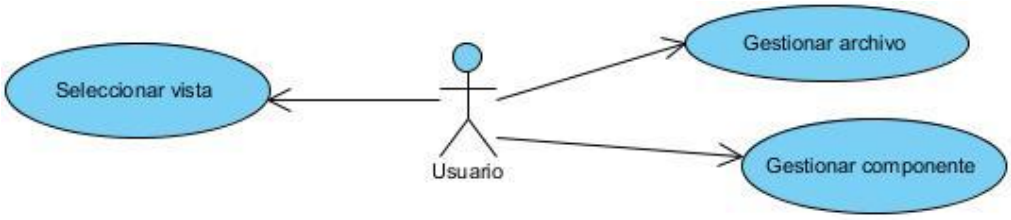


Figura 4: Diagrama de Casos de Uso del Sistema

A continuación se muestra la descripción del caso de uso Gestionar componente.

Caso de Uso:	Gestionar componente
Actores:	Usuario
Resumen:	El caso de uso comienza cuando al seleccionar la opción **"**Crear componente**"**, **"**Modificar componente**"**, **"**Eliminar componente**",** **"**Mostrar árbol de navegación**"**, **"**Generar archivo**"** o **"**Exportar archivo**"** en la sección Gestionar componente de la extensión, permite crear un nuevo diseño de componente. El sistema brinda la opción de modificar las propiedades del componente y eliminar el componente. Además, muestra el árbol de navegación del diseño realizado, actualiza y exporta el archivo JavaScript correspondiente al diseño de GUI creado.
Precondiciones:	Se debe ejecutar el Caso de Uso Gestionar archivo.
Referencias	RF4, RF5, RF6, RF7, RF8, RF9.
Prioridad	Crítico

Flujo Normal de Eventos	
Sección "Gestionar componente"	
Acción del Actor	**Respuesta del Sistema**
	1. Permite al Usuario realizar varias acciones: • Crear componente. • Modificar componente (*Ver en la Sección 1: Modificar componente*). • Eliminar componente (Ver en la Sección 2: Eliminar componente). • Mostrar el árbol de navegación (*Ver en la Sección 3: Mostrar árbol de navegación*) • Generar archivo (*Ver en la Sección 4: Generar archivo*) • Exportar archivo (*Ver en la Sección 5: Exportar archivo*)
2. Selecciona el componente de la paleta de componentes.	
3. Arrastra el componente seleccionado hacia el área de diseño.	
	4. Muestra el componente seleccionado en el área de diseño.

	5. Actualiza el árbol de navegación, las propiedades en el inspector de propiedades y termina así el caso de uso.

Prototipo de Interfaz

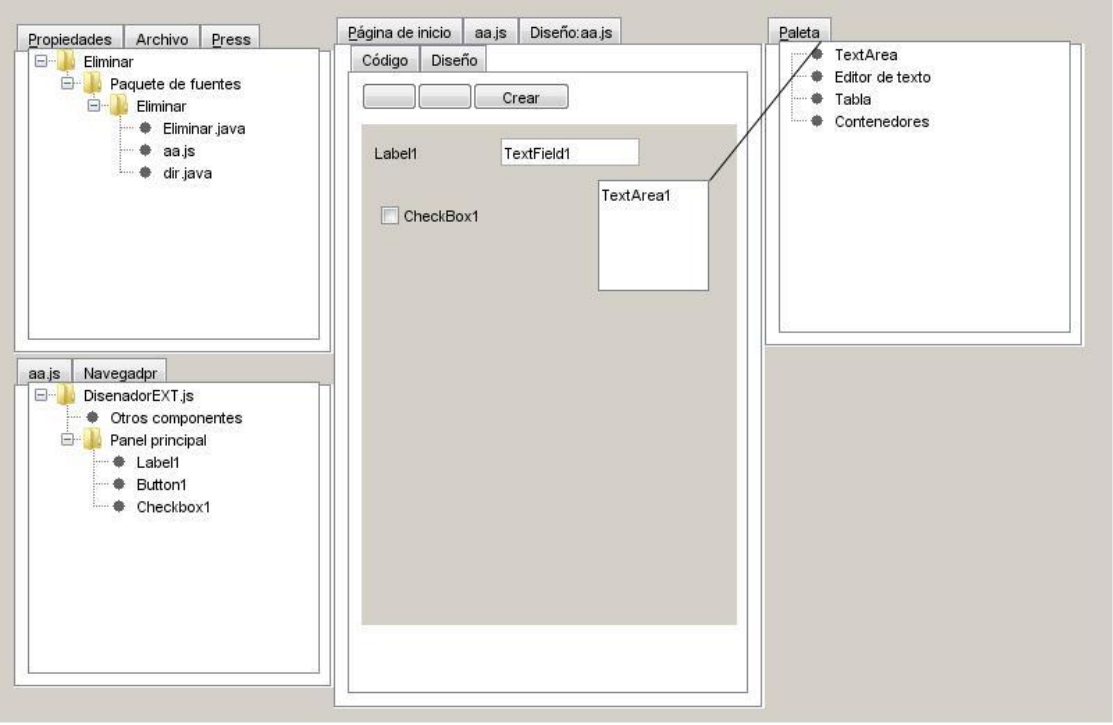


Figura 5: Crear componente

Sección 1 "Modificar componente"	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona el componente que será modificado en el área de diseño o en el árbol de navegación.	
2. Modifica propiedades del componente en el Inspector de Propiedades.	
	3. Muestra el componente modificado en el área de diseño.
	4. Actualiza el árbol de navegación.

Prototipo de Interfaz

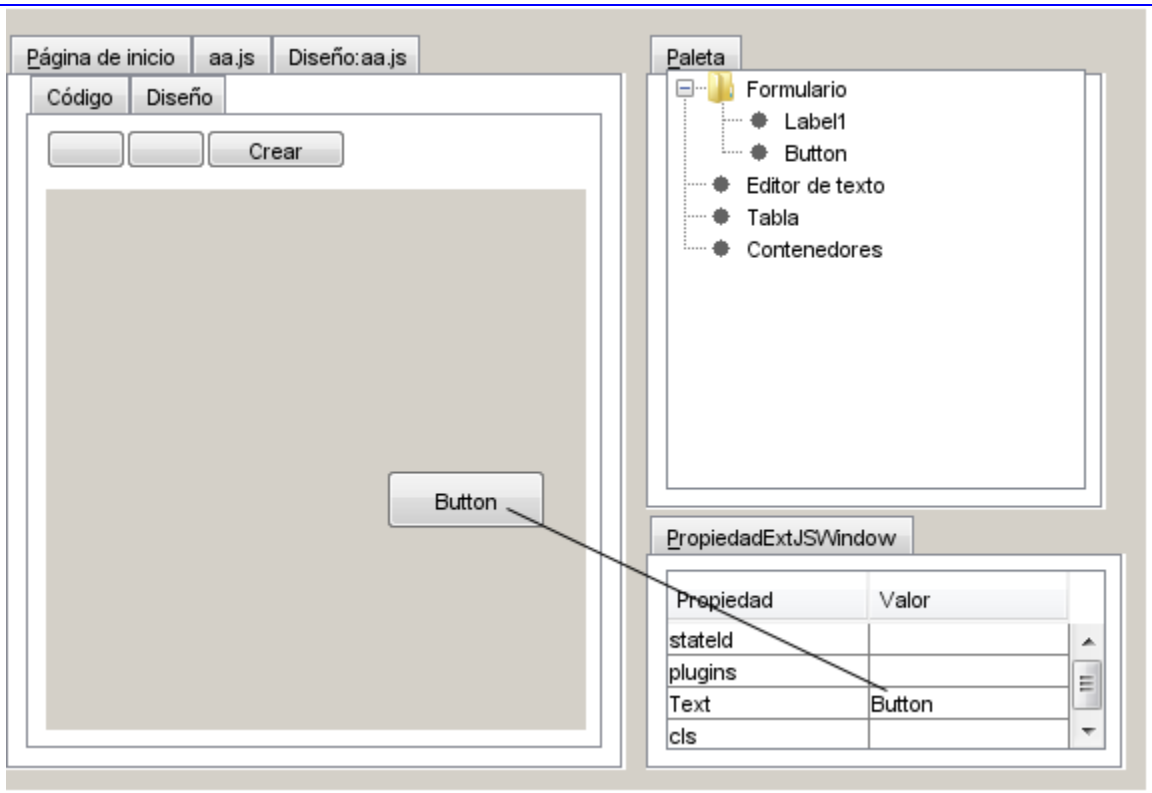


Figura 6: Modificar componente

Sección "Eliminar componente"	
1. Selecciona el componente que desea eliminar.	
2. Presiona clic derecho sobre el componente que desea eliminar.	
	3. Despliega un menú con la opción de **"**Eliminar**"**.
4. Selecciona la opción **"**Eliminar**"**.	
	5. Elimina el componente seleccionado.
	6. Actualiza el área de diseño.
	7. Actualiza el árbol de navegación.
Prototipo de Interfaz	

Figura 7:Eliminar componente

Sección "Mostrar árbol de navegación"	
1. Realiza el diseño de componente en el área de diseño.	
	2. Muestra el árbol de navegación correspondiente al diseño de componente realizado en el área de diseño.

Prototipo de Interfaz

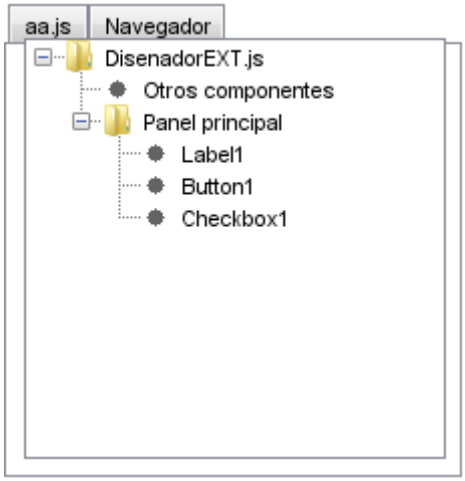
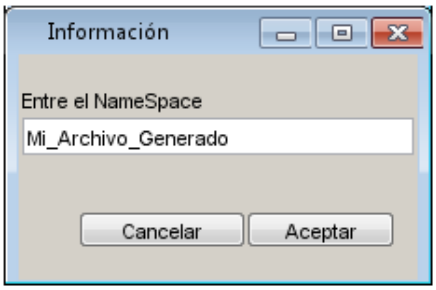


Figura 8: Mostrar árbol de navegación

Sección "Generar archivo"	
1. Presiona el icono correspondiente a la opción "Generar código" de la barra de herramientas de la vista de diseño.	
	2. Muestra una ventana para introducir el nombre del archivo.
3. Introduce el nombre del archivo que será generado y presiona la opción "Aceptar" o "Cancelar". En caso de que el usuario seleccione la opción "Cancelar", ver *Flujo Alterno 1.*	
	4. Si el usuario selecciona la opción "Aceptar" el sistema muestra un cuadro de diálogo notificando que el archivo fue actualizado y brinda al usuario la opción de quedarse en la vista de diseño o ver el archivo generado con las opciones "Aceptar" y "Cancelar". En caso de que el usuario seleccione la opción "Cancelar", ver *Flujo Alterno 2.*
5. Selecciona la opción "Aceptar".	
	6. Muestra el archivo JavaScript generado con las actualizaciones del diseño de componente realizadas.
Flujos Alternos	
Flujo Alterno 1	

	1. Cierra la ventana habilitada para introducir el nombre del archivo JavaScript.
Flujo Alterno 2	
	1. Se mantiene activa la vista de diseño.
Prototipo de Interfaz  Figura 9: Generar archivo	
Sección 6 "Exportar archivo"	
1. Presiona el ícono correspondiente a la opción "Exportar" de la barra de herramientas de la vista de diseño.	
	2. Muestra un cuadro de diálogo para que el Usuario busque la dirección donde desea guardar el archivo.
3. Selecciona la dirección hacia donde desea exportar el archivo y selecciona la opción "Guardar" o "Cancelar".	
	4. Si el usuario selecciona la opción "Exportar", el sistema guardar el archivo con extensión "js". Si no, ver Flujo Alterno.
5. Introduce el namespace [11] del archivo JavaScript y selecciona la opción "Aceptar" o "Cancelar".	
	6. Si el usuario selecciona la opción "Aceptar" el sistema guardar el archivo con extensión ".js". Sino ver Flujo Alterno2.
Flujo Alterno 1	

[11] Namespace: es un medio para organizar clases dentro de un entorno, agrupándolas de un modo más lógico y jerárquico.

	1. Cierra la ventana habilitada para introducir la dirección.
Flujo Alterno 2	
	2. Cierra la ventana habilitada para introducir el namespace y no exporta el archivo.

Prototipo de Interfaz

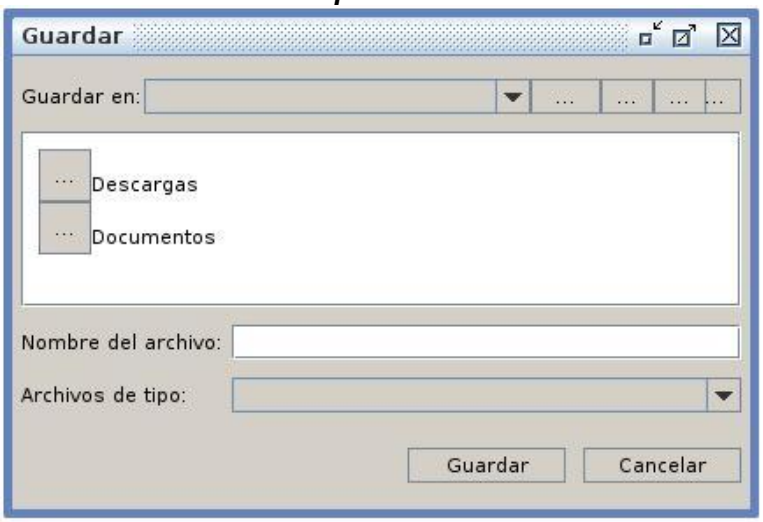


Figura 10: Exportar archivo

Postcondiciones:	Luego de ejecutarse el caso de uso el usuario puede crear, modificar y eliminar un componente para realizar el diseño de la GUI. A partir de este diseño se muestra el árbol de navegación y se actualiza el archivo JavaScript generado.

2.4 Modelo de Diseño

2.4.1 Arquitectura de la Extensión

Para la creación de la extensión, se tuvo en cuenta la arquitectura modular debido a las particularidades de la herramienta a extender. El NetBeans IDE, ofrece las características de una aplicación de cliente enriquecida en base a su arquitectura modular formada por un conjunto de APIs, en donde cada módulo se identifica como una nueva funcionalidad. La misma está estructurada por varias partes como se muestra en la Figura 11.

Figura 11: Arquitectura Modular del NetBeans IDE

Luego de aplicada la arquitectura modular, la extensión quedó estructurada como se muestra en la Figura 12.

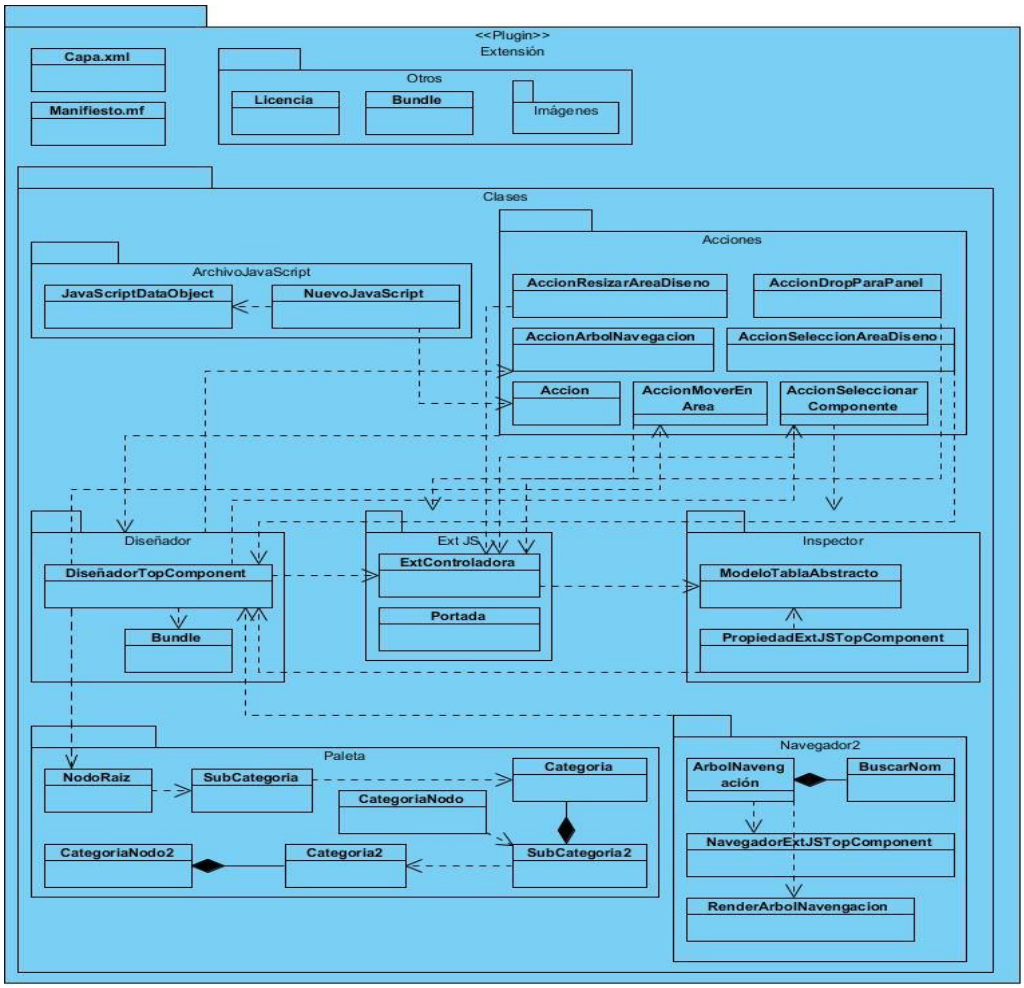


Figura 12: Paquete de la Extensión

2.4.2 Diagrama de clases del diseño

El diagrama de clase del diseño describe gráficamente las especificaciones de las clases del software y de las interfaces en una aplicación, conteniendo información como: clases, asociaciones y atributos, interfaces, con sus operaciones y constructores, métodos, tipos de atributos y dependencias. (15)

A continuación se muestra el diagrama de clases del diseño del CUS arquitectónicamente significativo Gestionar componente:

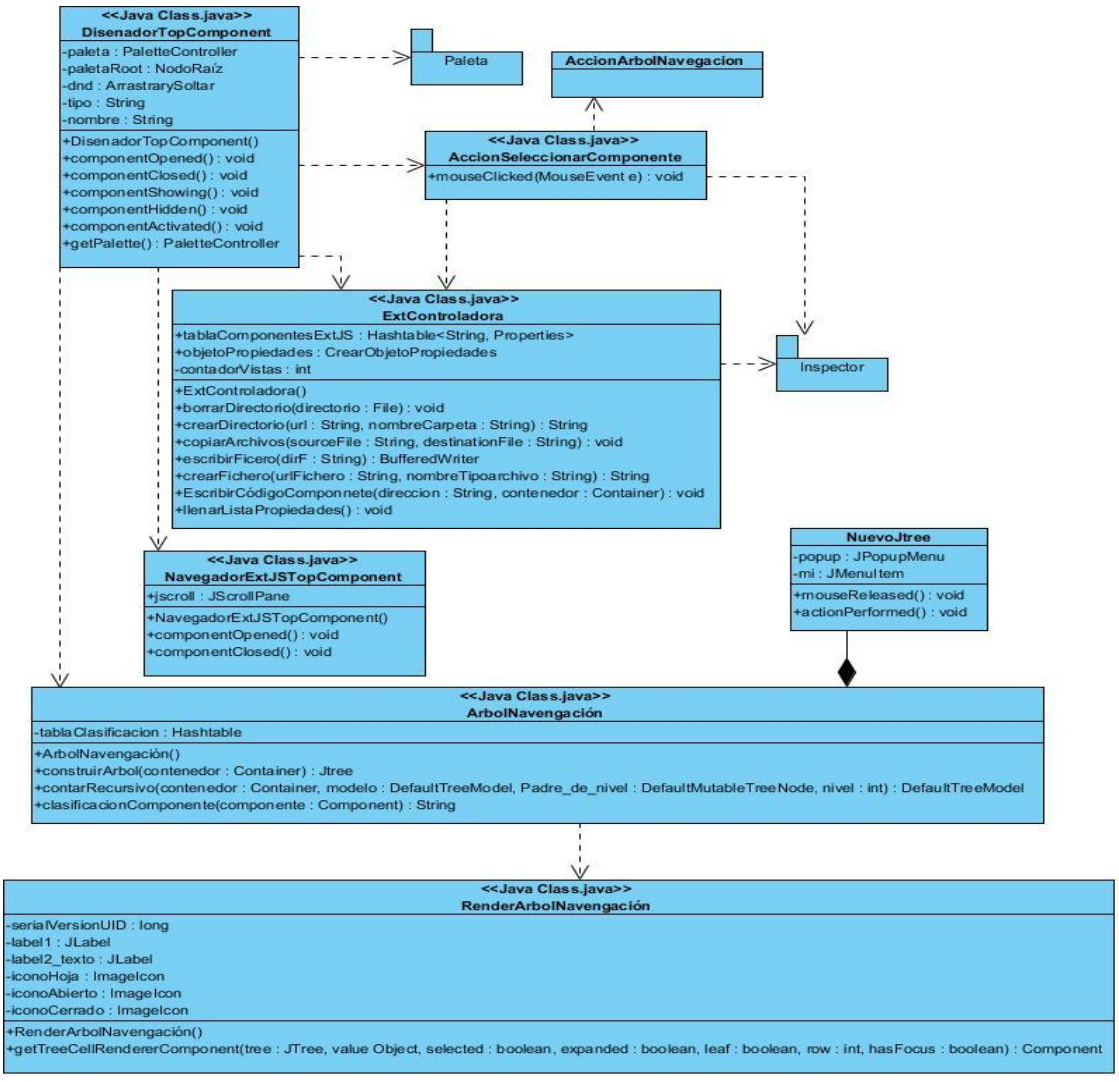


Figura 13: Diagrama de clases del diseño del CUS Gestionar componente

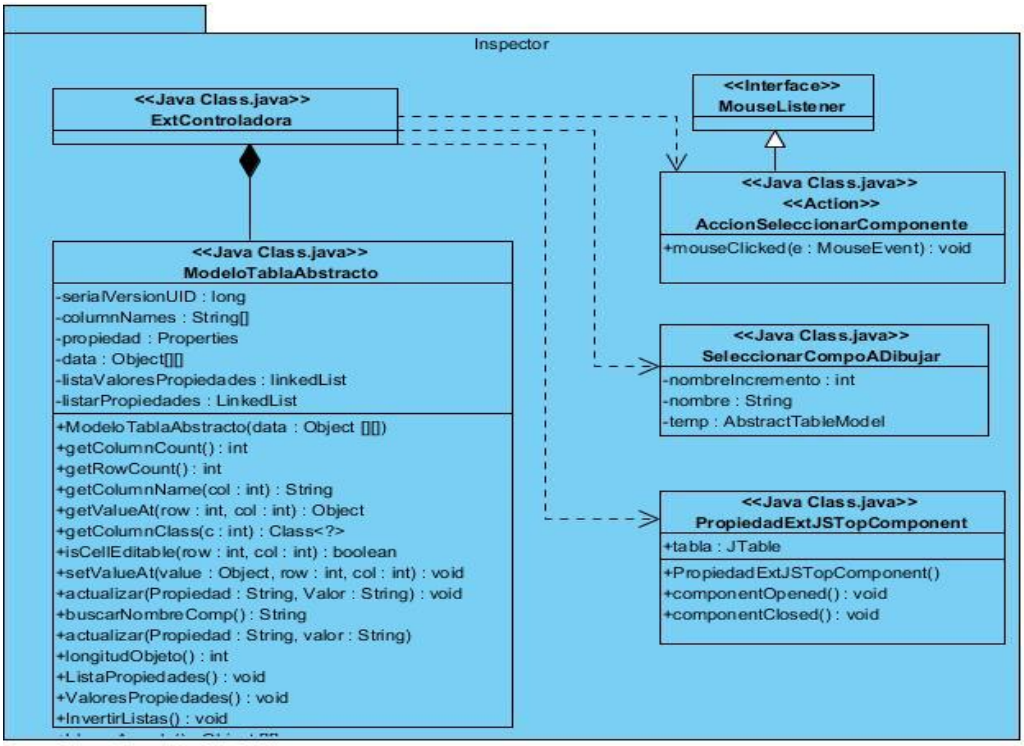


Figura 14: Diagrama de clases del diseño del CUS Gestionar componente- Paquete Inspector

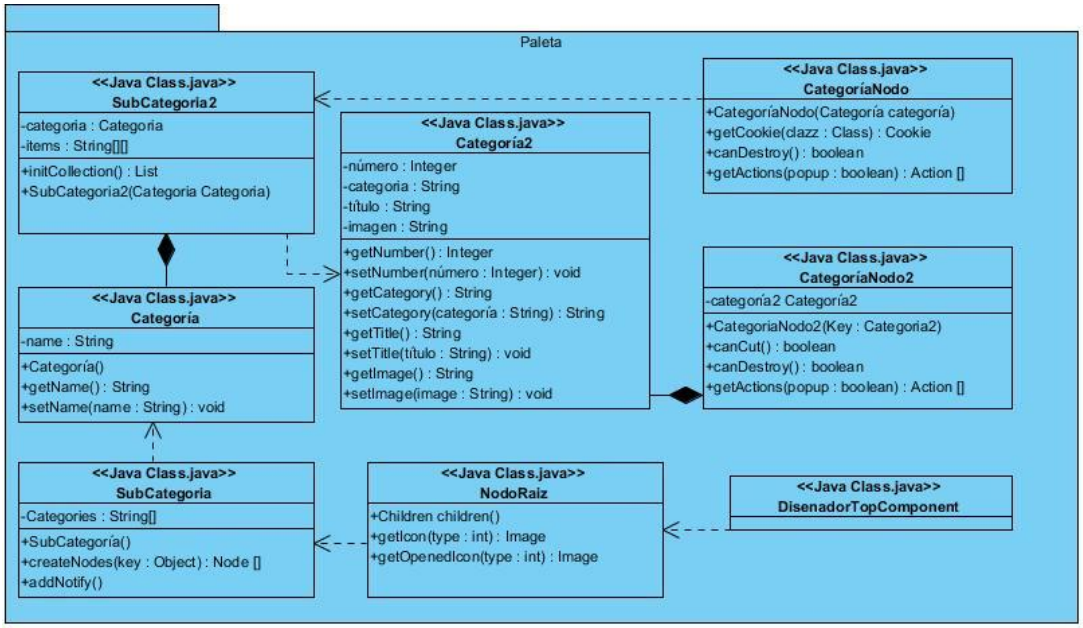


Figura 15: Diagrama de clases del diseño del CUS Gestionar componente- Paquete Paleta

Propósito: Permitir al Usuario gestionar los componentes que serán utilizados en el área de diseño.

Descripción: El diagrama provee varias funcionalidades que están unidas a los componentes que se manejan en el área de diseño de la clase DisenadorTopComponent, esta habilita una paleta de componentes que permite la funcionalidad de arrastrar y soltar componentes, además de mostrar las propiedades para un componente seleccionado en el área de diseño y añadirlo al navegador de componentes, provee funcionalidades como crear los componentes y editar sus propiedades, agrega además de la posibilidad de exportar el componente creado a un directorio seleccionado y actualizar el archivo JavaScript que habilitó el área de diseño imprimiendo la estructura de componente de Ext JS diseñado.

A continuación se describirán las clases contenidas en el diagrama Gestionar componente:

- **Clase:** DiseñadorTopComponent

Propósito: Permitir al Usuario el diseño de componentes.

Descripción: Clase que crea la paleta de componentes y los datos del inspector de propiedades correspondiente a los componentes creados. Posee el área de diseño y las opciones de ver las vistas de código y diseño, además de proveer el acceso a las funcionalidades de actualizar archivo y exportar componente.

- **Clase:** AccionSeleccionarComponente

Propósito: Visualizar las propiedades del componente seleccionado en el área de diseño.

Descripción: Es una clase que implementa la interfaz MouseListener y se implementa la solución en el método mouseClicked. Permite seleccionar las propiedades del componente sobre el cual se ha realizado el evento de clic, se tiene en cuenta el tipo de clic ejecutado para visualizar un pop-up menú del componente o actualizar la tabla de propiedades en el inspector de propiedades.

- **Clase:** ExtControladora

Propósito: Controlar el flujo de datos de la extensión.

Descripción: Clase que contiene atributos de tipo Hashtable, que guarda objetos de tipo Properties, correspondiente a los componentes del área de diseño, además posee los principales métodos para dar cumplimiento a los requisitos funcionales de la extensión.

- **Clase:** NavegadorExtJSTopComponent

Propósito: Mostrar el árbol de navegación con los componentes que se encuentran en el área de diseño.

Descripción: La clase se hace visible en el área izquierda inferior del Usuario y muestra los componentes que están en el área de diseño destacando los componentes y sus padres, emplea para su funcionamiento una barra que permite el desplazamiento por el árbol donde se visualizan los datos, estos son actualizados que muestra el árbol, los datos que muestra son actualizados desde la clase que permite el diseño DisenadorTopComponent.

- **Clase:** ArbolNavegacion

Propósito: Crear el árbol con que se visualiza en el NavegadorExt JSTopComponent.

Descripción: Posee un atributo de tipo Hashtable con la clasificación (padre, hijo) que se utiliza para construir el árbol de navegación. Este se construye apoyado en un objeto de tipo de contenedor que posee un conjunto de componentes. Esta clase es utilizada por DisenadorTopComponent que actualiza el árbol cuando se agrega o eliminan componentes.

- **Clase:** NuevoJTree

Propósito: Permitir al Usuario añadir acciones a los elementos del árbol de navegación.

Descripción: Clase que extiende de JTree e implementa la interfaz ActionListener, permite poner a los elementos del árbol la opción de visualizar un pop-up menú con la opción de eliminar.

- **Clase:** AcciónArbolNavegación

Propósito: Controlar las acciones realizadas sobre el árbol que se visualiza en el NavegadorExt JSTopComponent.

Descripción: La clase observa la ocurrencia de eventos en el árbol de navegación permitiendo seleccionar el componente al cual se le dio clic para representarlo en el área de diseño logrando ubicar al Usuario en el componente que busca, además muestra las propiedades para el componente seleccionado.

- **Clase**: AccionDropParaPanel

Propósito: Controlar la técnica de arrastrar y soltar los componentes que se encuentran dentro de un componente de tipo contenedor.

Descripción: La clase observa la ocurrencia de eventos ocurridos en los paneles agregados desde la paleta al área de diseño, este panel aceptará los componentes igual que el área de diseño redefiniendo la clase DropTargetListener.

- **Clase**: RenderArbolNavegación

Propósito: Definir comportamientos en el árbol de navegación.

Descripción: La clase permite definir cuando el árbol abre el nodo o elemento raíz principal, los hijos están cerrados o abiertos, además de mostrar el texto correspondiente al icono abierto, esta clase es usada para cambiar el setCellRenderer del árbol declarado en la clase ArbolNavengación.

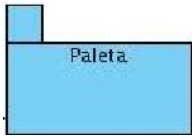


Figura 16: Paquete Paleta

Propósito: Agrupar el conjunto de clases perteneciente a la paleta contenida en el área de diseño.

Descripción: Paquete que contiene el conjunto de clases pertenecientes a la paleta contenida en el área de diseño. Estas clases serán descritas a continuación:

- **Clase:** Categoría

Propósito: Crear la estructura que contiene información para la paleta de componentes.

Descripción: La clase posee el atributo “name” que permite nombrar los nodos que serán creados usando esta clase. Además, implementa la interfaz TransferHandler que permitirá que los datos manejados en esta clase san transferibles mediante el uso de la técnica de arrastrar y soltar.

- **Clase:** Categoría2

Propósito: Crear la estructura que contiene información para los nodos a visualizar en la paleta de componentes.

Descripción: Implementa la interfaz TransferHandler que permitirá que los datos manejados en esta clase san transferibles mediante el uso de la técnica de arrastrar y soltar. Además, incluye atributos como título, imagen, categoría y número que permiten identificar el nodo.

- **Clase:** CategoríaNodo

Propósito: Crear una clase que permita crear diferentes clasificación de los nodos a mostrar en la paleta.

Descripción: Extiende de la clase AbstractNode y redefine comportamientos como las acciones, las cookie [12] y emplea la información de la clase Categoría redefiniendo el nombre a mostrar en el administrador de paleta definido por el NetBeans IDE. La clase CategoríaNodo permite establecer las bases para la creación de las clasificaciones de los componentes que se mostrarán en la paleta de componentes.

- **Clase:** CategoríaNodo2

Propósito: Una clase que permita crear Nodos a mostrar en las diferentes clasificaciones presentes en la paleta de componentes.

Descripción: Extiende de la clase AbstractNode y redefine comportamientos como las acciones, las cookie y emplea la información de la clase Categoría redefiniendo el nombre y el icono de a mostrar. La clase permite la creación de los nodos que se mostrarán en la paleta de componentes en las distintas clasificaciones creadas en la clase SubCategoría.

- **Clase:** SubCategoría

Propósito: Definir la clasificación de los componentes a visualizar en la paleta.

Descripción: Permite definir una lista de Categorías que se emplean para clasificar los componentes de la paleta a visualizar para el área de diseño. Además, permite vincular esta clasificación a los nodos de tipo Categorías mediante el método addNotify () y es la clase instanciada para pasar por parámetros a la clase NodoRaíz que construye la paleta.

- **Clase:** SubCategoría2

Propósito: Definir las componentes a mostrar según la clasificación definida en la clase SubCategoría2.

Descripción: Permite definir los componentes a visualizar mediante la declaración de un arreglo de elementos de tipo de dato String, además, permite asociar los componentes creados a la clase Categoría 2 redefiniendo el título a visualizar y la imagen a mostrar mediante el método initCollection () que retorna un arreglo de nodos.

[12] Cookie: Es una pequeña porción de información con una fecha de caducidad que se almacena en el fichero o directorio de cookies, referentes a informaciones de las preferencias de los usuarios.

- **Clase**: NodoRaíz

Propósito: Construir la paleta de componentes a utilizar en el área de diseño.

Descripción: La case extiende de la case AbstractNode y se le pasa por parámetros en el constructor un objeto de tipo Children. Es usada en la construcción de la paleta en la clase de diseño "DisenadorTopComponent" usado para la declaración de un atributo de este tipo llamado paleta.

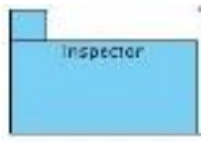


Figura 17: Paquete Inspector

Propósito: Agrupar el conjunto de clases perteneciente al navegador de contenido en el entorno de diseño del NetBeans IDE.

Descripción: Paquete que contiene el conjunto de clases pertenecientes al navegador que se habilita en el área de diseño del NetBeans IDE, luego de que el Usuario comienza el trabajo en el área de diseño. Estas clases serán descritas a continuación:

- **Clase**: ModeloTablaAbstracto

Propósito: Crear la estructura de filas y columnas de la tabla a mostrar en la clase PropiedadExtJSTopComponent.

Descripción: Se declara la estructura de las columnas y las filas se llenan de forma dinámica denotando propiedad y valor para el componente seleccionado en la interfaz, estos datos son actualizados a través de un arreglo bidimensional de Object, además, se define que la primera columna no será editable.

- **Clase**: PropiedadExtJSTopComponent

Propósito: Mostar la lista de propiedades de Ext JS para el componente seleccionado en el área de diseño.

Descripción: La clase se muestra en la parte inferior de la derecha del Usuario, debajo de la paleta de componentes, posee una tabla que actualiza las propiedades para el componente seleccionado en el área de diseño, permite modificar las propiedades de los componentes y visualizar estas en tiempo real, la tabla se actualiza desde la clase de de eventos de mouse AccionSeleccionarComponente.

- **Clase:** SeleccionarCompoADibujar

Propósito: Permite seleccionar el componente que se va a dibujar en el área de diseño.

Descripción: Permite seleccionar el componente que se va a dibujar el área de diseño además de crear el objeto de propiedades correspondiente al componente, se actualiza el componente con atributos como el nombre, texto y dimensiones.

Patrones de Diseño

Los patrones de diseño, son soluciones simples y elegantes a problemas específicos y comunes del diseño OO, basándose en la experiencia y habiéndose demostrado además que funcionan. Con el uso de patrones, los diseños serán mucho más flexibles, modulares y reutilizables.

Para la solución se aplicaron los siguientes patrones de diseño:

GRASP:

- **Experto:** Este patrón, está diseñado para que la responsabilidad de realizar una labor sea de la clase que tiene o puede tener los atributos involucrados. Está presente en las clases ExtContrladora, ArbolNavegacion y ModeloTablaAbstracto.

- **Creador**: Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Está presente en la clase ExtControladora y en la ComponentExt JSVisualElement para crear objetos.

- **Controlador:** La mayor parte de los sistemas reciben eventos de entrada externa, los cuales generalmente incluyen una interfaz gráfica operada por un Usuario. En este caso, si se está diseñando orientado a objetos, hay que elegir los controladores que manejen esos eventos de entrada. Este patrón ofrece una guía para tomar decisiones apropiadas que generalmente se aceptan. Se evidencia en la clase ExtControladora en el proceso de creación de los componentes, la selección de los mismos y en la asignación de los pop-up menú.

- **Alta cohesión**: Propone asignar la responsabilidad de manera que la complejidad se mantenga dentro de límites manejables asumiendo solamente las responsabilidades que deben manejar, evadiendo un

trabajo excesivo. Permite una mayor claridad en el entendimiento del sistema y una mayor capacidad de reutilización. Este patrón está presente en la clase AccionSeleccionAreaDiseno y AccionSelecMenu, se evidencia debido al bajo número de relaciones que poseen estas clases lo que favorece un bajo acoplamiento y por ello una alta cohesión.

- **Bajo Acoplamiento**: Debe haber pocas dependencias entre las clases. Si todas las clases dependen de todas, no se puede extraer software de forma independiente y reutilizarlo. Este patrón está presente en la clase NavegadorExtJSTopComponent que permite visualizar los componentes que se encuentran en el área de diseño.

GOF:

- **Singleton:** Asegura que una clase posea una sola instancia y proporciona un punto de acceso global a ella. Este patrón se ve evidenciado en la clase DisenadorTopComponent, debido a que siempre se hace una sola instancia de esta clase.

- **Facade:** Simplifica el acceso a un conjunto de clases proporcionando una única clase que todos utilizan para comunicarse con dicho conjunto. Se evidencia en la clase ExtControladora la cual se comporta como mediadora en la comunicación del resto de las clases que facilitan la creación de la solución.

- **Observer:** Permite definir la dependencia de uno o más objetos de forma que cuando un objeto cambie su estado, los restantes sean notificados y se actualicen. Se evidencia en los componentes del área de diseño que se encuentran en la clase DisenadorTopComponent, debido a que permite realizar la acción de mostrar la tabla cuando se presione clic sobre un componente, por lo que se comporta como un observador.

- **Strategy:** Permite encapsular algoritmos relacionados en clases y hacerlos intercambiables. Se implementa, debido a que todos los componentes del área de diseño poseen un algoritmo único y se ejecuta en dependencia del tipo de componente que se esté analizando, se evidencia en la clase ExtControladora cuando se asignan los pop-up menú o cuando se selecciona el componente a crear.

- **Iterator:** Proporciona una forma de acceder a los elementos de una colección de objetos de manera secuencial sin revelar su representación interna. Se evidencia cuando se recorren los objetos Properties utilizados en la clase ExtControladora.

2.4.3 Diagrama de Secuencia

Define acciones que puede realizar la aplicación en cuestión, además, representa la secuencia de mensajes entre las instancias de clases, componentes, subsistemas o actores. Enfatiza el orden de tiempo de los mensajes.

A continuación se muestra el diagrama de secuencias del CUS crítico Gestionar componente:

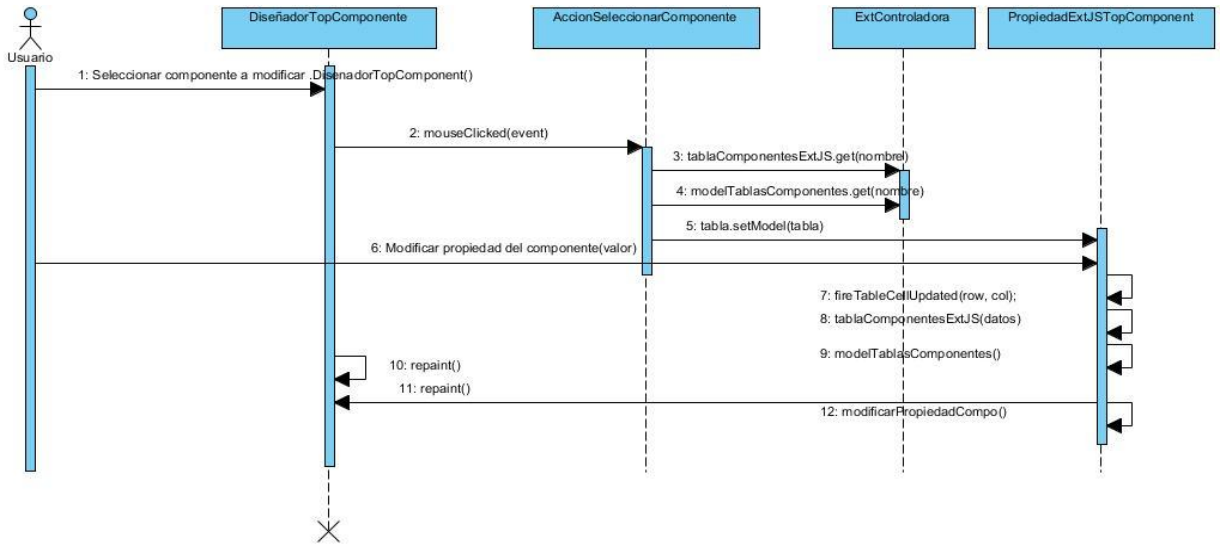


Figura 18: Diagrama de Secuencia del CUS Gestionar componente- Sección Modificar componente.

2.4.4 Escenario de despliegue

Es un tipo de escenario donde se muestra el diagrama del UML, el cual contiene las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. Representa los complementos de los diagramas de componentes que, unidos, proveen la vista de implementación del sistema. Describen la topología del sistema, la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos.

La extensión se debe desplegar en una PC cliente que posea instalado la herramienta a extender NetBeans IDE, por lo que no es necesario representar el diagrama.

Conclusiones

Mediante el modelo de dominio, se obtuvo una representación física del objetivo de la extensión, con el fin de que el Usuario, como actor del sistema, fundamentara las bases del funcionamiento de la misma.
A través de la identificación de los requisitos funcionales que intervienen en la extensión, se definió el punto de partida para la elaboración y representación de los diagramas de CUS y clases del diseño. Además los mismos, ayudaron a precisar los requerimientos técnicos necesarios para su correcto funcionamiento.
Luego de un estudio realizado de la arquitectura que utiliza la herramienta a extender, se determinó que para la estructuración del diagrama de paquete sería aplicada la Arquitectura Modular. Además, mediante el diagrama de clases del diseño quedó evidenciada la estructura interna de las clases y las relaciones entre ellas, transmitiendo a través del diagrama de secuencia el funcionamiento interno de las mismas.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

El presente capítulo se enfoca en proveer una solución a los requisitos del sistema a través del flujo de trabajo Implementación, además se analizan las pruebas relacionadas con la extensión. Se analizarán los estándares de codificación que ayudaron a la implementación, algunos ejemplos del código fuente y las interfaces principales de la extensión.

3.1 Modelo de implementación

El modelo de implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la estructura del sistema. Entre los componentes podemos encontrar datos, archivos, ejecutables, códigos fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos. Este artefacto describe cómo se implementan los componentes, congregándolos en subsistemas organizados en capas y jerarquías, y señala las dependencias entre éstos. (16)

3.1.1 Diagrama de Componentes

Los diagramas de componentes ilustran las piezas del software, controladores embebidos, etc. que conformarán un sistema. Un diagrama de componentes tiene un nivel más alto de abstracción que un diagrama de clase, usualmente un componente se implementa por una o más clases (u objetos) en tiempo de ejecución. Estos son bloques de construcción, como eventualmente un componente puede comprender una gran porción de un sistema. (17)

El diagrama de componentes que se muestra continuación pertenece al caso de uso arquitectónicamente significativo Gestionar componente. En el mismo se representa cada clase descrita en el diagrama de clases del diseño como un componente físico del sistema, pero además se muestra la relación del paquete que contiene a este conjunto de clases con el IDE. Dicha relación está dada por la necesidad de que algunas clases se integren al NetBeans IDE y puedan utilizar las funcionalidades nativas de la plataforma.

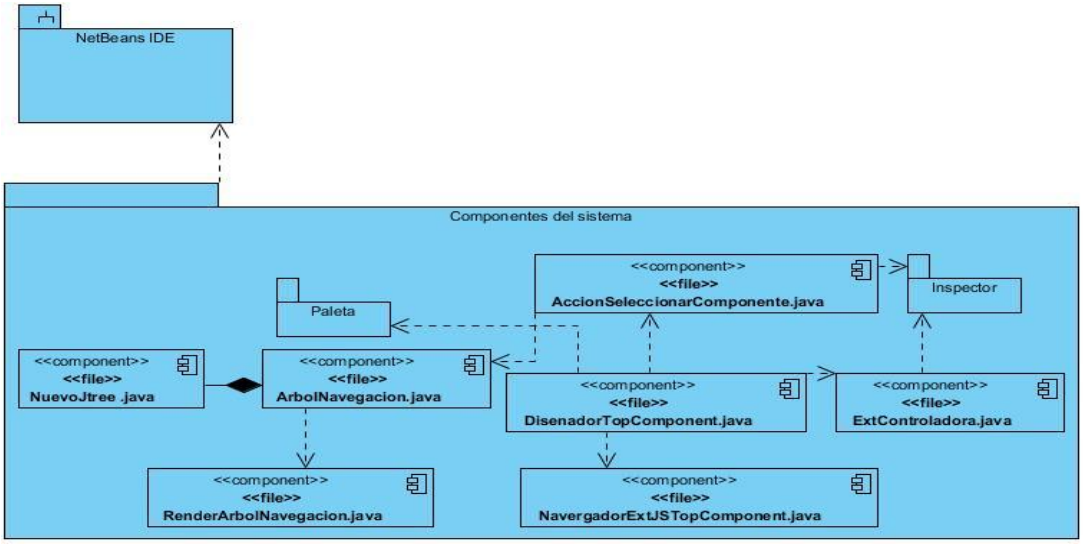


Figura 19: Diagrama de componente del CUS Gestionar componente.

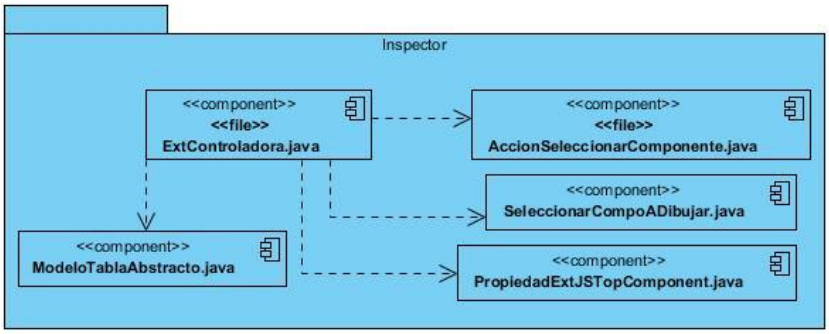


Figura 20: Diagrama de componente CUS Gestionar componente- Paquete Inspector.

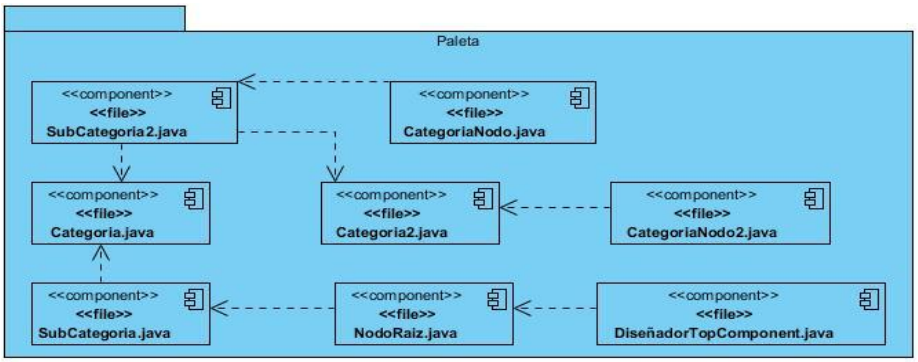


Figura 21: Diagrama de componente del CUS Gestionar componente- Paquete Paleta.

3.2 Código Fuente

En la programación, el código fuente, sigue un conjunto de reglas y normas propias de la aplicación que es utilizada para desarrollar, pero siendo siempre entendible por un Usuario que tenga estos conocimientos específicos.

3.2.1 Estándares de codificación

Un estándar de codificación comprende los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Además, si se aplica de forma continua un estándar de codificación bien definido, se utilizan técnicas de programación apropiadas y, posteriormente, se efectúan revisiones del código de rutinas, existen posibilidades de que un proyecto de software se convierta en un sistema de software fácil de comprender y de mantener.
Luego de aplicar el estilo de codificación definido, se logró que el código fuente cuente con las siguientes cualidades:

- **Extensibilidad**: Facilidad con que se adapta un sistema a cambios de especificación. Un buen estilo de código fomenta soluciones que no solo resuelven el problema de los programas, sino que también reflejan claramente la relación problema/solución. Esto tiene como efecto que muchos cambios simples en el problema reflejen de forma obvia los cambios a hacer en la extensión.

- **Reparabilidad**: Posibilidad de corregir errores sin demasiado esfuerzo.

- **Verificabilidad**: Facilidad con que pueden comprobarse propiedades de un sistema. Si el estilo de código hace obvia la estructura de la extensión, eso ayuda a verificar que el comportamiento sea el esperado.

- **Capacidad de evolución**: Capacidad de adaptarse a nuevas necesidades.

- **Comprensibilidad**: Facilidad con que la extensión puede ser comprendida.

Estilos de codificación empleados:

Durante la implementación de la extensión, fue necesario utilizar algunos estilos de codificación en busca de un estándar que aportara una mayor organización al código.

- **Identación**: Las normas de identación indican la posición en la que se deben colocar los diferentes elementos que se incluyen en el código fuente.
  El identado debe ser de cuatro espacios para:
    - Declaraciones dentro de las clases.
    - Enunciado dentro de los métodos y funciones.
    - Enunciados dentro de bloques y comandos.
    - Enunciados dentro de cuerpos switch/case.

- **Declaración de variables, parámetros, objetos y métodos:** El nombre de las variables y parámetros debe comenzar con la primera letra en minúscula, la cual identificará el tipo de datos al que se refiere, en caso de que sea un nombre compuesto, se empleará notación Camel Casing[13]. Además, los métodos están compuestos por múltiples palabras juntas iniciando cada palabra con letra mayúscula.

- **Clases**: El objetivo fundamental es nombrar las clases e instancias de forma estándar para la extensión. Los nombres de las clases deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación Pascal Casing[14].

- **Comentarios, separadores, líneas, espacios en blanco y márgenes**: Establecer un modo común para comentar el código, utilizar separadores, línea, espacios en blanco y márgenes, de forma tal que sea comprensible el código, siguiendo las pautas que a continuación se muestran:
    - Comentar al inicio de la clase o función especificando el objetivo de la misma.
    - Dejar una línea en blanco antes y después de la declaración de una clase o de una estructura y de la implementación de una función.

[13] Camel Casing: es un estilo de escritura que se aplica a frases o palabras compuestas. Se caracteriza por colocar en mayúscula la primera letra de cada palabra de un identificador.
[14] Pascal Casing: es un estilo de codificación que consiste en usar mayúscula al principio de un nombre y al inicio de cada palabra que contenga el identificador.

- Evitar comentar cada línea de código. Cuando el comentario se aplica a un grupo de instrucciones debe estar seguido de una línea en blanco. En caso de que se necesite comentar una sola instrucción se suprime la línea en blanco o se escribe a continuación de la instrucción. No se debe usar espacio en blanco:
  - Después del corchete abierto y antes del cerrado de un arreglo.
  - Después del paréntesis abierto y antes del cerrado.
  - Antes de un punto y coma.

Entre las funciones que posee la extensión se encuentra Actualizar archivo, la cual permite escribir en el archivo JavaScript el código con estructura de componente para el Framework Ext JS. Para este código generado se aplica el estilo de codificación empleado para el desarrollo de la extensión y promovido por el IDE. A continuación se muestra la estructura que el mismo posee:

- **Estructura del código del archivo JavaScript generado**:

```
Ext.namespace('vista');
vista.Estructura = Ext.extend(Ext.Panel, {
   constructor: function(config) {
      config = config || {};
      //declaración de componentes.
      Ext.apply(config, {
         //Propiedades del contenedor
         });
      vista.Estructura.superclass.constructor.call(this, config);
   }
});
```

3.2.2 Ejemplos de Código Fuente

El código implementado en su mayoría responde a funciones de creación, modificación, actualización, listado, entre otros, que se vuelven sencillos mediante el uso de las interfaces, eventos y librerías ofrecidas por la JDK.

Ejemplo de este código es el método **getPalette()**, encargado de proveer al área de diseño los componentes para emplear la técnica de arrastrar y soltar mediante la especificación del comportamiento en la clase DropTargetListener que usa en su implementación al método ComponenteADibujar() de la

clase ExtControladora. El mismo provee, además, el nombre identificativo para el componente capturado arrastrado desde la paleta.

```
private PaletteController getPalette() {
       if (null == paleta) {
           paletaRoot = new NodoRaiz(new SubCategoria());
           paletaRoot.setName("Paleta Root");
           paleta = PaletteFactory.createPalette(paletaRoot,
                   new PaletaAcciones(), null, null);
           try {
               paleta.addPropertyChangeListener(new PropertyChangeListener() {
                   @Override
                   public void propertyChange(PropertyChangeEvent evt) {
                       if (PaletteController.PROP_SELECTED_ITEM.
                               equals(evt.getPropertyName())) {
                           Lookup selItem = paleta.getSelectedItem();
                           if (null != selItem) {
                               Node selNode = (Node) selItem.
                                       lookup(Node.class);
                               if (null != selNode) {
                                   Image selImage = selNode.
                                           getIcon(BeanInfo.ICON_COLOR_32x32);
                                   String selName = selNode.getDisplayName();
                                   /**
                                    * Atributo nombre que permite ser empleado
                                    * para buscar el componente que se va a
                                    * agregar al área de diseño.
                                    */

                                   nombre = selName;
                                   AccionDropParaPanel.nombre = selName;
                               }
                           }
                       }
                   }
               });
           } catch (Exception ex) { System.out.println("Error en getPalette():"
                       + ex.getLocalizedMessage());
           }
       }
       return paleta;
   }
```

Figura 22: Método getPalette() de la clase DisenadorTopComponent.

3.2.3 Interfaces Principales de la Extensión

El diseño de la extensión se encaminó a lograr una interfaz agradable, sencilla y amigable correspondiente con la herramienta a extender. Las interfaces principales son las que componen el área de trabajo. Entre estas se encuentra el panel principal donde el usuario va a realizar su diseño de interfaces gráficas.

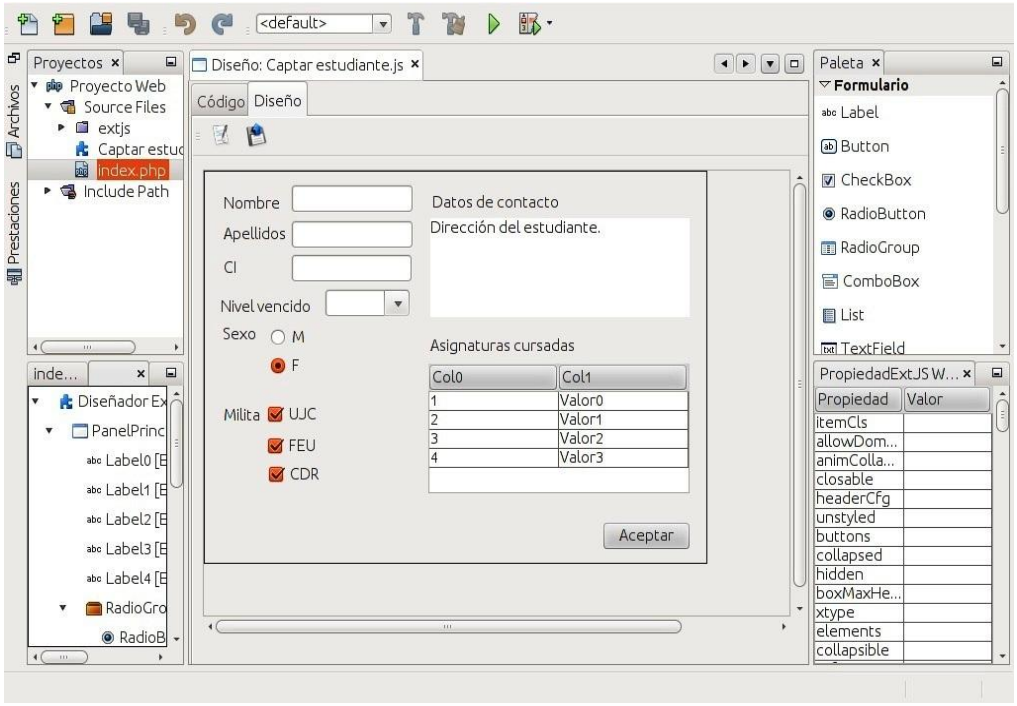


Figura 23: Área para el diseño de GUI con Ext JS.

Además también se encuentra la ventana habilitada para crear un nuevo archivo que contiene el nombre del archivo y otros elementos que son necesarios definir para su creación.

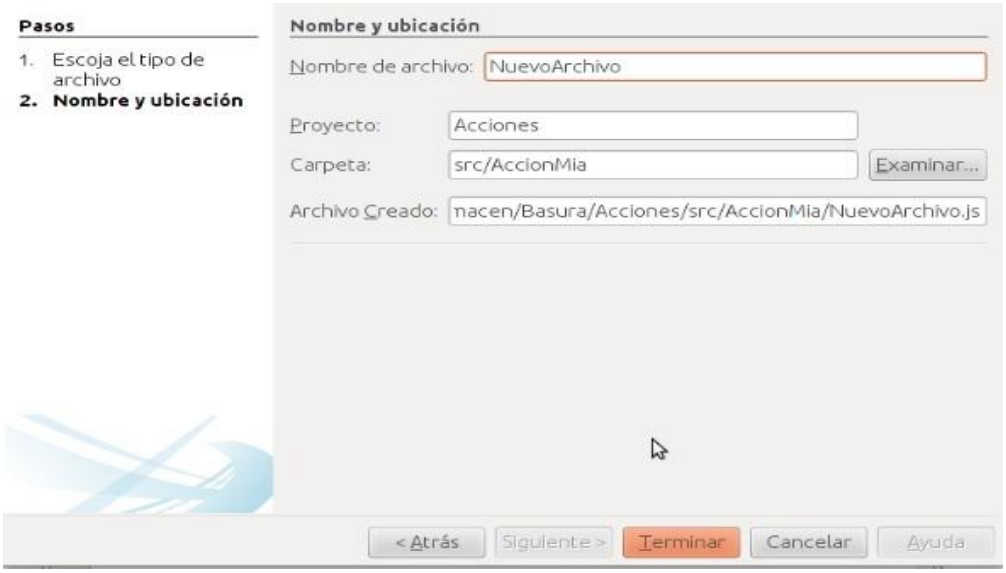


Figura 24: Ventana para crear archivo JavaScript.

3.3 Modelo de Pruebas

Una de las actividades más importantes desarrolladas durante el flujo de trabajo Implementación, son las pruebas del desarrollador. Al ejecutar pruebas al sistema durante su realización, se comprueba que el funcionamiento del mismo es correcto. También es importante destacar, cuando se trata de una extensión, la necesidad de realizar pruebas de integración, debido a que esta es una buena forma de verificar que las nuevas funcionalidades obtenidas se integran al sistema correctamente. Esto se realizó validando que las funcionalidades de dicha herramienta no fueron afectadas luego de la instalación de la extensión, lo cual se llevó a cabo exitosamente. Sin embargo, en ambos casos es importante determinar el tipo, el método y la técnica adecuada para implementar las pruebas.
Una forma de comprobar el funcionamiento de la herramienta tras las peticiones del usuario, es aplicando pruebas funcionales de Función, siguiendo el método de Caja Negra y la técnica de Partición de Equivalencia. Esta combinación permite al usuario asignarle valores correctos e incorrectos a las variables del sistema, para comprobar las posibles no conformidades existentes en cada una de sus funcionalidades.

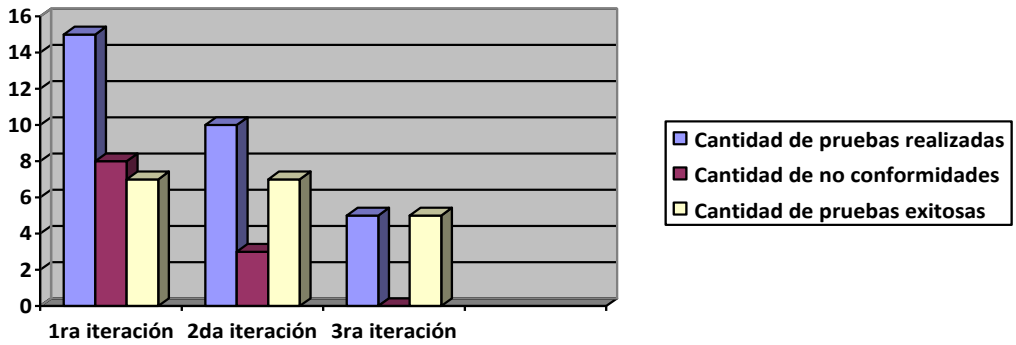
Los casos de prueba se definen a partir de las funcionalidades descritas en los casos de uso, validando el funcionamiento de cada uno de ellos. A través de ellos se especifican los escenarios, de ahí sus secciones y variables involucradas en cada proceso. De esta forma se detalla la respuesta del sistema ante cada valor introducido por el usuario, ya sea correcto o incorrecto.
Con el fin de representar los casos de prueba, se utiliza una tabla donde se desglosan las funcionalidades en secciones y a su vez en escenarios, para visualizar de manera fructífera la ejecución de las pruebas.
En una primera iteración se aplicaron pruebas al sistema comprobando el funcionamiento de los casos de uso "Gestionar archivo" y "Gestionar componente". Para el primer caso de uso se comprobó, entre otras funcionalidades, que los datos para crear un nuevo archivo fueran introducidos correctamente por el usuario y que el usuario no pudiera crear un archivo con campos vacíos. En el segundo caso de uso se comprobó que el usuario realizara un diseño de componentes, validando las propiedades de los mismos en el momento de modificarlos. En este último se validó además, que el árbol de navegación, el archivo generado y el archivo exportado, correspondieran al diseño de componentes realizado previamente por el usuario.

Para el caso de uso "Seleccionar vista", no existían variables involucradas por lo que no fue necesaria la validación a través del método partición de equivalencia. En consecuencia de esto se comprobó que el sistema permitiera al usuario seleccionar las vistas correspondientes al área de trabajo correctamente. A continuación se muestra el caso de prueba realizado al caso de uso "Gestionar archivo".

Escenario	Descripción	Nombre del Archivo	Dirección	Nombre	Extensión	Todos los archivos	Tamaño del archivo	Hora de modificación	Respuesta del sistema	Flujo central
EC 1.1 Modificar archivo con datos correctos.	Este escenario realiza la modificación de un archivo correctamente.	N/A (-)	N/A (-)	V (Mi_Archivo)	V (js)	N/A (-)	N/A (-)	N/A (-)	El sistema modifica el archivo.	1. Extensión al NetBeans IDE/Gestionar archivo/Modificar(Presiona clic derecho sobre el archivo JavaScript que desea modificar, muestra un menú desplegable, selecciona la opción Propiedades). 2. Se llenan los campos correctamente. 3. Selecciona la opción
EC 1.2 Modificar archivo con datos incorrectos.	Este escenario realiza la modificación de un archivo incorrectamente.	N/A (-)	N/A (-)	I (/Mi_Archivo)	V (js)	N/A (-)	N/A (-)	N/A (-)	El sistema valida que los campos estén llenos correctamente y se activa la opción "Cerrar ".	1. Extensión al NetBeans IDE/Gestionar archivo/Modificar(Presiona clic derecho sobre el archivo JavaScript que desea modificar, muestra un menú desplegable, selecciona la opción Propiedades). 2. Se llenan los campos con datos no válidos.
		N/A (-)	N/A (-)	V (Mi_Archivo)	I (lola)	N/A (-)	N/A (-)	N/A (-)		

Figura 25: Escenario "Modificar archivo" del caso de uso Gestionar archivo.

Luego de haber realizado las pruebas funcionales, para comprobar el funcionamiento de la extensión, se obtuvieron un total de 8 no conformidades en una primera iteración y 3 en una segunda iteración, las cuales fueron solucionadas satisfactoriamente en una última iteración. En la siguiente gráfica se muestra un resumen de los resultados obtenidos:



Gráfica 1: Resultado de las pruebas.

3.3.1 Método Delphi

Después de realizar las pruebas para comprobar el funcionamiento de la extensión, se hace necesario demostrar que el tiempo de desarrollo de GUI utilizando la extensión es reducido. Para esto se aplica el método Delphi a un grupo de expertos haciendo uso de la extensión.

Fases del método Delphi:

- Fase 1: Formulación del problema

La extensión desarrollada propone reducir el tiempo de desarrollo de GUI utilizando el Framework Ext JS. Para ello fue necesario definir los atributos (A) que serán evaluados por los expertos, teniendo en cuenta el método para el desarrollo de GUI con Ext JS utilizado actualmente en el departamento de Integración de Soluciones y el propuesto por la extensión del NetBeans IDE.

A1: Tiempo de realización de diseño de GUI con Ext JS sólo a código.

A2: Tiempo de realización de diseño de GUI con Ext JS utilizando la extensión propuesta.

Para la realización de la evaluación de los atributos vistos anteriormente, cada experto utilizará un prototipo de GUI definido previamente por el departamento de Integración de Soluciones. El mismo está conformado por los componentes definidos para la extensión, formando un diseño que responde a las necesidades del proyecto Sistema Integrado de Gestión y Estadística. A continuación se muestra el prototipo propuesto por el departamento:

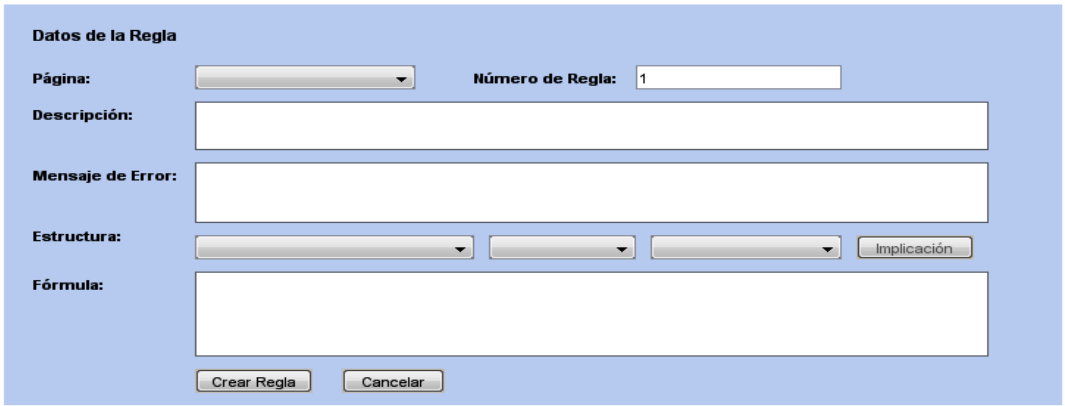


Figura 26: Prototipo de SIGE-Módulo Diseñador de Formularios.

- Fase 2: Selección de Expertos

Para conformar el grupo de expertos se midieron dos aspectos fundamentales: teniendo en cuenta que tuviera al menos 1 año de experiencia trabajando con el Framework de desarrollo Ext JS y que pertenecieran al departamento de Integración de Soluciones. Además, la selección se realizó siguiendo el principio de anonimato propuesto por el método Delphi y a personas experimentadas e interesadas en participar en el cuestionario. Estos aspectos permiten garantizar la confiabilidad de los resultados al contar con los especialistas en las áreas de conocimiento asociado a la solución del problema.

- Fase 3: Elaboración y realización del cuestionario

Para la elaboración de los cuestionarios (Ver Anexo 3), fue necesario medir los atributos determinados en la Fase 1. Esto llevó consigo la preparación de la herramienta donde los expertos pudieran realizar el diseño propuesto (Ver Figura 25), sólo a código y midiendo a su vez el tiempo de desarrollo de dicho procedimiento. Luego, realizaron el mismo diseño pero apoyándose en la extensión propuesta y midiendo nuevamente el tiempo de desarrollo. De esta forma cada experto obtuvo los resultados necesarios para responder posteriormente el cuestionario aplicado.

- Fase 4: Análisis de los resultados

Luego de aplicar el cuestionario a los 7 expertos seleccionados y evaluar los atributos A1 y A2 definidos previamente en la fase 1, se obtuvieron los siguientes resultados:

Expertos	**A1**	**A2**
Experto 1	60 min	10 min
Experto 2	90 min	15 min
Experto 3	75 min	10 min
Experto 4	90 min	14 min
Experto 5	85 min	16 min
Experto 6	65 min	12 min
Experto 7	60 min	10 min

Tabla 3: Tiempo de desarrollo de los expertos

A partir de los resultados se obtuvo un promedio de tiempo de desarrollo sólo a código de 75 minutos y utilizando la extensión 12 minutos, por lo que queda demostrado que el tiempo de desarrollo de GUI con Ext JS utilizando la extensión para el NetBeans IDE es reducido.

Conclusiones

Mediante la realización del flujo de trabajo Implementación se pusieron en práctica algunos estándares de codificación para una mejor comprensión del código, ayudando de esta forma a que futuros programadores se sientan familiarizados con el producto. Además para colaborar con el éxito de dicha comprensión se muestran algunas imágenes de la extensión como resultado final. Para demostrar que los requisitos propuestos fueron cumplidos satisfactoriamente, fue necesario llevar a cabo diferentes pruebas, que demostraron el funcionamiento de extensión.

CONCLUSIONES GENERALES

Luego de la realización del presente trabajo de diploma se concluye lo siguiente:

- La investigación realizada sobre los conceptos principales para el análisis de la situación problemática, fue el punto de partida para identificar las funcionalidades de la extensión.
- A partir del diagrama de clases del diseño, se estructuraron las clases de acuerdo con los patrones identificados, lo que permitió su posterior implementación.
- Se implementaron y probaron los componentes definidos y se obtuvo la primera versión de la extensión del NetBeans IDE para el diseño de GUI con Ext JS.

RECOMENDACIONES

A partir de los resultados obtenidos luego de la realización de la extensión se ofrece la siguiente recomendación:

- Continuar la investigación e implementación de otros componentes del Framework de desarrollo Ext JS, para su futura integración al NetBeans IDE para el diseño de GUI.

REFERENCIAS

1. EcuRed. Modelo RAD. [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 20, 2012.] http://www.ecured.cu/index.php/Modelo_de_desarrollo_rápido_de_aplicaciones.
2. Ecured. WYSIWYG. [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 12, 2012.] http://www.ecured.cu/index.php/WYSIWYG.
3. Ecured. Interfaz gráfica de Usuario (GUI). [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 12, 2012.] http://www.ecured.cu/index.php/GUI.
4. Ecured. RIA. [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 12, 2012.] http://www.ecured.cu/index.php/RIA.
5. Böck, Heiko. The Definitive Guide to Plataforma NetBeans 7. New York : Science+Business Media, 2012. 978-1-4302-4101-0.
6. EcuRed. Plugin. [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 26, 2012.] http://www.ecured.cu/index.php/Plugin.
7. Kevin Turner. Kioskea.net. [Internet] 11 2012. [Citado: 26 11, 2012.] http://es.kioskea.net/contents/langages/api.php3.
8. EcuRed. Desarrollo de software basado en componentes. [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 26, 2012.]

http://www.ecured.cu/index.php/Desarrollo_de_software_basado_en_componentes.

9. Oracle Corporation. NetBeans. Plugins catalogue. [Internet] Oracle, 2012. [Citado: 11 26, 2012.] http://plugins.NetBeans.org/plugin/41629/jcode-form-builder-with-Ext JS.
10. EcuRed. Los lenguajes de marcado. [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 12 6, 2012.]
11. Victoria Hernández. Buenas Tareas. [Internet] 5 12, 2011. [Citado: 12 7, 2012.] http://www.buenastareas.com/ensayos/Xml-Lenguaje-De-Marcas-Extensible/2165946.html.
12. Miguel A. Laguna. Ingeniería del Software I. [Internet] 2012. [2 12, 2013] http://www.infor.uva.es/~mlaguna/is1/apuntes/2-requisitos.pdf.
13. Garcia, José Luis Miralles i. Evaluación de impacto ambiental y evaluación estratégica ambiental. Habana : s.n., 2011.
14. EcuRed. Requisitos no Funcionales. [Internet] Enciclopedia colaborativa. [Citado: 2 12, 2013.]

15. Indira Ramírez Tamayo, Dainovy Rodríguez Marrero. *Sistema de edición de noticias.* La Habana : s.n., 2012.
16. MeRinde. [Internet] [Citado: 3 28, 2013.] http://merinde.net/index.php?option=com_content&task=view&id=495&Itemid=291.
17. Sparx System. [Internet] 2007. [Citado: 3 28, 2013.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_componentdiagram.html.

BIBLIOGRAFÍA

1. EcuRed. *XML.* [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 10, 2012.] http://www.ecured.cu/index.php/XML.
2. EcuRed. *Sencha Ext JS.* [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 9, 2012.] http://www.ecured.cu/index.php/Sencha_Ext_JS.
3. Gesen, MSc. Pedro Jesús Abreu. Portal Tunarte. *¿Qué es Ext JS?* [Internet] 6 28, 2011. [Citado: 11 9, 2012.] http://www.ltu.jovenclub.cu/index.php?option=com_content&task=view&id=2001&Itemid=187.
4. EcuRed. *Java.* [Internet] Enciclopedia Colaborativa, 12 14, 2010. [Citado: 11 10, 2012.] http://www.ecured.cu/index.php/Java.
5. EcuRed. *RIA.* [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 10, 2012.] http://www.ecured.cu/index.php/RIA.
6. Ecured. *GUI.* [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 10, 2012.] http://www.ecured.cu/index.php/GUI.
7. Ecured. *WYSIWYG.* [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 12, 2012.] http://www.ecured.cu/index.php/WYSIWYG.
8. Maner, Walter. Desarrollo Rápido de Aplicaiones. [Internet] 3 15, 1997. [Citado: 11 12, 2012.] http://mena.com.mx/gonzalo/maestria/ingsoft/presenta/rad/.
9. EcuRed. *Modelo RAD*. [Internet] Enciclopedia colaborativa, 12 14, 2010. [Citado: 11 20, 2012.] http://www.ecured.cu/index.php/Modelo_de_desarrollo_rápido_de_aplicaciones.
10. Raya Hernández, Mirian G and Zulueta Blanco, Maria Elena. Textos científico-técnicos. LA HABANA : CIENTIFICO-TECNICOS, 2011.
11. Böck, Heiko. The Definitive Guide to NetBeansTM Platform. New York : Springer-Verlag, 2009. 978-1-4302-2418-1.
12. Borudreau, Tim, Tulach, Jaroslav y Wielenga, Geertjan. Rich Client Programming Plugging into the Plataforma NetBeans. Stoughton, Massachusetts : Sun Microsystem, 2007. 0-13-235480-2.
13. Garcia, José Luis Miralles i. Evaluación de impacto ambiental y evaluación estratégica ambiental. Habana : s.n., 2011.

14. Almaguer, Armín González. El método Delphi y el procesamiento estadístico de los datos obtenidos de la consulta a los expertos. Habana, ISP “José de la Luz y Caballero” : s.n, 2013.
15. García, Fernando. LA TESIS Y EL TRABAJO DE TESIS: Recomendaciones metodológicas para la elaboración de los trabajos de tesis. México : Limusa S.A, 2004. 968 18-6235-X.
16. Petri, Jürgen. Plataforma NetBeans 6.9 Developer's Guide. BIRMINGHAM - MUMBAI : Packt Publishing, 2010. 978-1-849511-76-6.
17. Miguel A. Laguna. Ingeniería del Software I. [Internet] 2012. [2 12, 2013] http://www.infor.uva.es/~mlaguna/is1/apuntes/2-requisitos.pdf
18. EcuRed. *Requisitos no Funcionales*. [Internet] Enciclopedia colaborativa. [Citado: 2 12, 2013.]
19. Tello, Jesus Cáceres. Diagrama de casos de uso. [Internet] 2010. [Citado: 2 13, 2013.] http://www2.uah.es/jcaceres/capsulas/DiagramaCasosDeUso.pdf.
20. Curso de Arquitectura de Software. [Internet] 2010. [Citado: 2 16, 2013.] http://sophia.javeriana.edu.co/~metorres/Diplomado/5%20-%20Curso%20Arquitecturas%20-%20Clases.pdf.
21. EcuRed. *Diagrama de Despligue*. [Internet] Enciclopedia colaborativa. [Citado: 2 21, 2013.] http://www.ecured.cu/index.php/Diagrama_de_despliegue.
22. MeRinde. [Internet] [Citado: 3 28, 2013.] http://merinde.net/index.php?option=com_content&task=view&id=495&Itemid=291.
23. Sparx System. [Internet] 2007. [Citado: 3 28, 2013.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_componentdiagram.html.
24. Mastermagazine. [Online] [Cited: 3 28, 2013.] http://www.mastermagazine.info/termino/4328.php.
25. msdn. [Internet] 2003. [Citado: 3 28, 2013.] http://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx.
26. María Jesús Lamarca. Hipertexto. [Internet] [Citado: 4 10, 2013.] LapuenteHipertextohttp://www.hipertexto.info/documentos/namespaces.html.
27. EcuRed. *Método Delphi*. [Internet] Enciclopedia colaborativa. [Citado: 5 12, 2013.] http://www.ecured.cu/index.php/Metodo_delphi.
28. EcuRed. *Symfony*. [Internet] Enciclopedia colaborativa. [Citado: 5 15, 2013.] http://www.ecured.cu/index.php/Symfony.

ANEXOS

Anexo1. Aspectos positivos de las herramientas de diseño de Ext JS.

Aspectos	Lycan GENESIS	Ext Designer	Sencha Architec
Paleta de componentes	x	x	x
Navegador de componentes	x	x	x
Inspector de propiedades	x	x	x
Salvar diseño	x	x	x
Salvar plantilla	x		
Renderizar compontes	x	x	x
Arrastrar y soltar	x	x	x
Entorno escritorio			x
Generar código	x	x	x
Generar código con estructura de componente	x		

Tabla 4: Aspectos que se pueden emplear el desarrollo de la extensión

Anexo2. Encuesta realizada a los programadores del departamento de Integración de Soluciones.

Encuesta		
Dpto: Integración de Soluciones	**Centro: DATEC**	
Categoría Científica:	**Proyecto:**	**Experiencia:**
Componentes de Ext JS más utilizados en los proyectos		
Herramientas	**Contenedores:**	**Campos de Formulario**
◯ Barra de herramientas	◯ Contenedor	◯ Casilla de opción

◯ Separador	◯ Grupo de campos	◯ Campo de fecha
◯ Relleno	◯ Panel	◯ Editor de texto enriquecido
◯ Elemento de texto		◯ Campo numérico
◯ Agrupador de botones	**Estándar**	◯ Área de texto
	◯ Label	◯ Campo de hora
Gráficos	◯ Botón de acción	◯ Lista de opciones
◯ Gráficos de barras	◯ Componente rectangular	
◯ Gráfico de columnas	◯ Selector de desplazamiento	**Grilla de Datos**
◯ Gráfico de línea	◯ Barra de progreso	◯ Grilla
◯ Gráfico de pastel		

Tabla 5: Encuesta.

Anexo3. Cuestionario aplicado a los expertos utilizando el método Delphi.

Responda las siguientes preguntas:

1. ¿Qué tiempo se demoró en la realización del prototipo de GUI haciendo uso del Framework Ext JS sólo a código?
2. ¿Qué tiempo se demoró en la realización del diseño del propuesto de GUI utilizando la extensión del NetBeans IDE?

GLOSARIO

Requisitos: Condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo.

Web: Es un sistema de documentos (o páginas web) interconectados por enlaces de hipertexto, disponibles en Internet.

PHP: Lenguaje diseñado originalmente para la creación de páginas web dinámicas. Su utilización está basada principalmente en la interpretación del lado del servidor pero actualmente puede ser empleado desde una interfaz de línea de comandos o en la creación de otros tipos de aplicaiones.

AJAX: Es una tecnología asíncrona, en el sentido de que los datos adicionales se requieren al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página.

ORM: Es un componente de software que me permite trabajar con los datos persistentes simulando que lo mismos son parte de una base de datos OO.

GPL: La Licencia Pública General o GPL (del inglés *General Public License*), está orientada principalmente a proteger la libre distribución, modificación y uso de un software. Su propósito es declarar que el software cubierto por esta licencia es "Software libre" y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

SQLite: Es una biblioteca escrita en leguaje C que implementa un Sistema de gestión de bases de datos transaccionales SQL auto-contenido, sin servidor y sin configuración. El código de SQLite es de dominio público y libre para cualquier uso, ya sea comercial o privado. Actualmente es utilizado en gran cantidad de aplicaciones incluyendo algunas desarrolladas como proyectos de alto nivel.

Symfony: Es un framework para construir aplicaciones Web con PHP. En otras palabras, es un enorme conjunto de herramientas y utilidades que simplifican el desarrollo de las aplicaciones Web.