Universidad de las Ciencias Informáticas

Facultad 4

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Título: Diseño e implementación de la interfaz de usuario del Módulo Trabajo Educativo, para el Sistema de gestión de Residencia.

Autor:

Yuniel Rodríguez Bello

Tutor:

Ing. Yosvany David Medina Hernandez

Ciudad de La Habana, Junio del 2009

"Año del 50 Aniversario del Triunfo de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco 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
Yuniel Rodríguez Bello	Ing. Yosvany David Medina Hernández

"No se vive celebrando victorias, sino superando derrotas."

Ernesto Che Guevara

DATOS DEL CONTACTO

Ing. Yosvany David Medina Hernandez

Profesor Adiestrado. Graduado en el 2007 de Ingeniero en Ciencias Informáticas en la Universidad de Ciencias Informáticas (UCI), Título de Oro.

Ha impartido asignaturas del departamento de Ingeniería y Gestión de Software y del departamento Técnicas de Programación, dentro del mismo se desempeño como Jefe de Asignatura de Grafico x Computadoras durante el curso 2008-2009. Forma parte del equipo de desarrollo del Sistema de Gestión de Residencia. Se desempeña como Líder de Proyecto. Ha participado en varios eventos a nivel de universidad e internacionales al igual que ha realizado varias publicaciones dentro de la maestría de Gestión de proyectos Informáticos, la cual cursa.

E-mail: ymedina@uci.cu

AGRADECIMIENTOS

A mis padres por su amor infinito, por hacer de mí un hombre de bien, por la educación que en mí inculcaron... por todo.

A mi hermana por apoyarme tanto, no solo aquí en la universidad sino a lo largo toda mi vida
A mi familia por estar día a día brindándome todo su apoyo.
Agradezco a la Revolución por darme la oportunidad de estudiar en esta Univecidad.
A Fidel, máximo guía de la Revolución cubana y forjador de esta universidad en la que he realizado mis estudios y he vivido mis mejores momentos de mi vida.
A la Universidad de las Ciencias Informáticas.
A mi tutor Yosvany D.Medina Hernández por tanta dedicación y por enceñarme a esforzarme cada vez más.
A Yoinet por darme el apoyo que me a dado todos estos años.
A Yandi Fernandez Martínez por ser un amigo incondicional y soportarme todos estos años a pesar de ser como soy.
A Yuslin Morales Pino por ser una gran amiga y ayudarme todos estos años, brindándome su mano en lo que necesite.
A mi novia por tener la dedicación y comprensión durante todos estos años y que a pesar de la distancia de este último año me a seguido apoyando sin alejarse de mi lado.
A todos los que de una manera u otra contribuyeron al desarrollo de esta investigación lleguen mis agradecimientos más sinceros.
A todos mis sinceros y fieles amigos lo que han estado ahí cuando siempre los necesito.
A todos los que en el transcurso de estos cortos pero fructíferos años han dejado su huella en mi vida.

A todos...

DEDICATORIA

A mis padres Pedro Rodríguez Pajón y Zenaida Bello Sánchez cada letra de este Trabajo de Diploma.

RESUMEN

Dado que actualmente en la Universidad de las Ciencias Informáticas(UCI) se está desarrollando un sistema que informatice todos los procesos de la residencia, donde uno de los principales es el de “Trabajo Educativo” y el mismo no tiene una interfaz de usuarios que facilite dicha gestión, bajo esta necesidad se decidió, desarrollar el diseño e implementación de una interfaz web para gestionar, facilitar y mejorar el acceso a la información manejada en el Sistema, cumpliendo con los estándares web internacionales, haciendo uso de diferente patrones de diseño, así como con los requerimientos del usuario; mostrando el resultado del trabajo del programador de interfaz, acorde con la planificación para el cumplimiento de la implementación de cada interfaz de usuario de este sistema.

Una interfaz de usuario no es más que una mediación, entre hombre y máquina, que facilita la comunicación, la interacción, accesibilidad, navegabilidad y eficiencia, entre el ser humano y la computadora. El diseño de interfaces de usuario es un aspecto que ha adquirido relevancia en el desarrollo de sistemas y la calidad del mismo puede influir de manera positiva o negativa en el criterio de los stakeholder sobre el producto, ya que es este aspecto una de los más representativo a la vista de los usuarios finales.

En el Sistema de Gestión de Residencia consta dentro de sus principales módulos con “Trabajo Educativo” donde las interfaces de usuario desarrolladas para el mismo deben ser:

Funcionales, amigables y rápidas.

- Que cumplan con los requerimientos del sistema así como con los requerimientos del usuario.
- Que cumpla con los estándares de calidad y diseño internacionales.
- Que utilicen las tecnologías más novedosas y que hayan sido aprobadas a su vez por la dirección de informatización.

Por tal motivo pretendemos con este Trabajo de Diploma una implementación eficiente de estas interfaces de usuario, que va a contribuir en gran medida, al éxito del software que garantizará la gestión de información con todos los usuarios que interactúen con el Sistema y principalmente para los que trabajen en el módulo “Trabajo Educativo” del Sistema de Gestión de Residencia.

PALABRAS CLAVE

Interfaz, Capa de presentación, Pantallas, Formularios,

INDICE DE CONTENIDOS

INTRODUCCION

Hoy en día los sistemas informáticos están presentes en todas las esferas de la sociedad, en la política, en lo social, en la cultura, en la economía, en fin en todas sin excepción. El uso de aplicaciones informáticas ha sido de carácter vital para realizar trabajos que se tornan complejos, tediosos, que requieren de mucho tiempo y detalle, los cuales si fueran realizados por el hombre estarían expuestos a cometer grandes errores y muchas veces, pues simplemente conllevaría a la pérdida de tiempo, todo esto ha dado lugar a la necesidad de informatizarnos cada vez más, automatizar los procesos que enmarcan un trabajo o alguna situación determinada, los cuales garantizan un aprovechamiento óptimo del tiempo de trabajo, reducción de fallas en un gran margen, permitiendo muchas facilidades, logrando una interacción más amena y cómoda entre los usuarios y su puesto de trabajo. En la actualidad se presentan disimiles problemas difíciles de resolver de forma manual, sin la utilización de computadoras, pudiéndose solucionar de una forma más rápida y eficiente, a través de aplicaciones informáticas. El avance de las tecnologías en el mundo se ha convertido en una herramienta al servicio de las estrategias de cada institución. Esto trae consigo que la creación de sistemas automatizados se haga cada vez más engorrosa y que las necesidades del cliente sean cada vez mayor y a la vez más complejas de satisfacer.

Todos los productos de software deben contar con una interfaz válida, amigable, accesible y flexible para todos los usuarios, además de cumplir con los requerimientos. Una interfaz bien definida es diseñada específicamente según los principios de cada usuario final. Pueden existir interfaces con propósitos comunes, apariencias diferentes y resultados exitosos para el cliente. Diseñada específicamente para la persona que la usará o para cualquier usuario, pues la mayoría de los programas y sistemas operativos ofrecen varias formas de interacción al usuario. Cada vez se hace más imperativo la mejor interacción hombre-computadora a través de una adecuada interfaz de Usuario.

Nuestro país lucha por convertirse en uno de los mayores productores de software, para esto se creó la Universidad de las Ciencias Informáticas (UCI), en la cual se desarrollan una serie de proyectos productivos, entre los que podemos encontrar el Sistema de Gestión de Residencia, el cual fue creado a solicitud de la Vicerrectoría de Residencia de informatizar esta área en busca de agilizar los procesos, actualmente se desarrolla en la facultad 4 un sistema que automatice los mismos. Para este se

identificaron un conjunto de requerimientos no funcionales de interfaz, necesarios para crear una interacción amigable y a la vez funcional de los usuarios con el mismo, por lo que es necesario el diseño e implementación de una interfaz de usuario adaptada a las necesidades del cliente y compatible con la arquitectura del proyecto. Entiéndase como interfaz de usuarios el canal visual que permite la comunicación entre hombre y máquina, facilitando interacción, accesibilidad, navegabilidad y eficiencia, y como diseño e implementación de la misma al desarrollo que garantizara el cumplimiento de estas características. Dentro de las áreas enfrascadas en la automatización de la residencia de la Universidad podemos mencionar específico el área de Trabajo Educativo.

Planteándose como **problema** de este trabajo la inexistencia de una interfaz de usuario amigable y funcional, que complemente la arquitectura definida para el módulo de Trabajo Educativo del proyecto de Residencia de la UCI.

El **objeto de estudio** de este proyecto lo componen las capas de presentación de los proyectos productivos en la UCI y como **campo de acción** la interfaz de usuario para el Módulo de Trabajo Educativo del proyecto Residencia de la UCI.

Este trabajo tiene como **objetivo general** diseñar e implementar una interfaz amigable y funcional, que complemente la arquitectura definida para el módulo de Trabajo Educativo del proyecto de Residencia de la UCI.

La tesis está estructurada en 3 Capítulos. En el capítulo 1 “Fundamentación Teórica” se realiza un estudio de las interfaces existentes, de las diferentes librerías utilizadas en el mundo para el desarrollo de las mismas así como las herramientas o IDEs de para la creación de estas. En el capítulo 2 “Propuesta de solución” se describen los elementos fundamentales a tener en cuenta en el desarrollo de la solución, además de describir algunos componentes que brinda el framework Extjs y de algunos estándares de codificación empleados para la solución. En el capítulo 3 “Solución del problema” quedan establecidas las interfaces del sistema para módulo “Trabajo Educativo” así como las expresiones regulares utilizadas para las validaciones de los campos de textos de cada interfaz de dicho módulo.

CAPÍTULO 1: ESTADO DEL ARTE

1.1 Introducción

En este capítulo se realizará un estudio acerca del surgimiento y desarrollo de la interfaz gráfica de usuario, así como un estudio de los diferentes estándares internacionales y patrones empleados en el diseño de la capa de presentación de todo sistema de gestión. También el mismo se describirá los Frameworks más utilizados tanto en el mundo como en la universidad para la implementación de interfaces web amigables, rápidas y funcionales; además de algunos IDEs más utilizados en el desarrollo de las mismas.

Por tanto se manejan un conjunto de conceptos y terminologías, que se incorporan al desarrollo de las interfaces de usuario, realizándose a la vez breves valoraciones de las mismas que ayuden a alcanzar los objetivos propuestos por el trabajo de diploma.

1.2 Interfaces de Usuarios

La creación de las interfaces de usuario ha sido un área del desarrollo de software que ha evolucionado dramáticamente a partir de la década de los setentas. La interfaz de usuario es el vínculo entre el usuario y el programa de computadora. Una interfaz es un conjunto de comandos o menúes a través de los cuales el usuario se comunica con el programa, **(Myers, 1996)**.

¨Las interfaces son los factores individuales más importantes en la aceptación de un sistema automatizado" **(Tsichritzis, 1985)**.

"Tienen un impacto decisivo en la eficiencia y la efectividad con la que el usuario podrá aprender y utilizar un sistema" **(Hammer, 1983)**.
El objetivo de las interfaces de usuario es el de la adaptación y comprensión entre la complejidad de los sistemas y las capacidades del ser humano. Por esto las tendencias son hacia interfaces más potentes considerando:
El aumento de la capacidad de los sistemas.
El aumento de la complejidad (según la ley de la tecnología, "a mayor potencia, mayor complejidad" **(Hammer, 1983)**: surge la necesidad por parte del usuario de contar con algo que se la oculte.

La interfaz debe cumplir con algunas características:

• Naturalidad. El nuevo sistema automatizado debe tender a ser lo más similar al antiguo.

• Facilidad de aprendizaje y uso, dos aspectos que no siempre van unidos.

• Consistencia. La interfaz debe mantener uniformidad en cuanto a estilo, Vocabulario, etc. **(Hammer, 1983)**

1.3 Clasificación de interfaces

1.3.1 Según la forma de interactuar del usuario

- Interfaces alfanuméricas (intérpretes de mandatos) que solo presentan texto.
- Interfaces gráficas de usuario (GUI, *Graphics User Interfaces*), las que permiten comunicarse con el ordenador de una forma muy rápida e intuitiva representando gráficamente los elementos de control y medida.
- Interfaces táctiles, que representan gráficamente un "panel de control" en una pantalla sensible que permite interaccionar con el dedo de forma similar a si se accionara un control físico. **(Toledo, 2008)**

1.3.2 Según su construcción

Pueden ser de hardware o de software:

- Interfaces hardware.- Se trata de un conjunto de controles o dispositivos que permiten la interacción hombre-máquina, de modo que permiten introducir o leer datos del equipo, mediante pulsadores, reguladores e instrumentos.
- Interfaces software.- Son programas o parte de ellos, que permiten expresar nuestros deseos al ordenador o visualizar su respuesta **(Toledo, 2008)**

La evolución de las interfaces de usuario corre en paralelo con la de los sistemas Operativos; de hecho, la interfaz constituye actualmente uno de los principales elementos de los mismos. A continuación se muestran las distintas interfaces que históricamente han ido apareciendo, ejemplificándolas con las sucesivas versiones de los sistemas operativos más populares.

1.4 Tipos de Interfaces

Las interfaces no siempre son intuitivas; ya que existen varios tipos de interfaces dentro de las que podemos encontrar:

1.4.1 Interfaces de línea de mandatos:

Esta interfaz se encuentra por ejemplo en algunos sistemas operativos como los NOS de los Reuters o algunos Shell de Unix, DOS, etc.; es la primera que utilizaron los ordenadores, El diseño de esta interfaz es crítico para el manejo del equipo, hay algunas muy bien diseñadas que incorporan controles intuitivos y de fácil manejo, en cambio existen otras que no se entienden bien y el usuario no acierta a manejarlas correctamente sin estudiar un manual o recibir formación del experto. Esta interfaz tiene como inconveniente que carga la memoria del usuario (debe memorizar los mandatos; incluso la ayuda es difícil de leer); nombres no siempre adecuados a las funciones, significado de los mandatos mal comprendido a veces (varios mandatos con el mismo o parecido significado, como DEL y ERASE); inflexible en los nombres (DEL y no DELETE), estas interfaces también tiene sus ventajas entre ellas se pude decir que son potente, flexible y controlado por el usuario, aunque esto es una ventaja para usuarios experimentados. La sintaxis es estricta, y los errores pueden ser graves. **(Toledo, 2008)**

1.4.2 Interfaces de menús.

Estas interfaces es una lista de opciones que se muestran en la pantalla o en una ventana de la pantalla para que los usuarios elijan la opción que deseen, los menús permiten dos cosas: navegar dentro de un sistema, presentando rutas que llevan de un sitio a otro, y seleccionar elementos de una lista, que representan propiedades o acciones que los usuarios desean realizar sobre algún objeto. Las interfaces de menús aparecen cuando el ordenador se vuelve una herramienta de usuario y no sólo de programadores. Las actuales interfaces gráficas u orientadas a objetos siguen utilizando este tipo de interfaces. **(Toledo, 2008)**

1.4.3 Interfaces gráficas.

La GUI es definida por Lewis y Rieman de la siguiente forma:

"Las interfaces gráficas de usuario son aquellas que incluyen cosas como menús, ventanas, teclado, ratón y algunos otros sonidos que la computadora hace, en general,

todos aquellos canales por los cuales se permite la comunicación entre el hombre y la computadora." **(Lewis-Rieman, 1993)**

También existen otras definiciones tales como:

"…tipo de entorno que permite al usuario elegir comandos, iniciar programas, ver listas de archivos y otras opciones utilizando las representaciones visuales (iconos) y las listas de elementos del menú…" **(Stephanidis, 2001)**

"…es aquella parte de un programa que comunica al usuario con el programa mediante representaciones gráficas…" **(Stephanidis, 2001)**

La definición más acertada para esta terminología, según la opinión del autor, acorde con la esfera informática y el desarrollo de software, es la siguiente:

"La Interfaz Gráfica de Usuario es un tipo de interfaz que mediante el uso del lenguaje visual, sustentado por un conjunto de imágenes y componentes gráficos, muestra en la pantalla una información específica así como un conjunto de acciones posibles, posibilitando a los usuarios una interacción con un sistema informático." **(Stephanidis, 2001)**

Un GUI es una representación gráfica en la pantalla del ordenador de los programas, datos y objetos, así como de la interacción con ellos. Un GUI proporciona al usuario las herramientas para realizar sus operaciones, más que una lista de las posibles operaciones que el ordenador es capaz de hacer.

1.5 Surgimiento y desarrollo de las GUIs

La idea de una interfaz gráfica de usuario para un computador u ordenador, surge incluso, mucho antes de que la tecnología estuviera disponible para ello. El Memex (Memory Expander) era una máquina que Vannevar Bush había ideado, pero nadie pudo materializarlo y en síntesis, posibilitaba la búsqueda y reproducción de archivos microfilmados y daba además la posibilidad a los usuarios de tomar anotaciones en los márgenes. Luego, en 1945 Bush retoma sus ideas que inspiraron a Douglas Engelbart, quien construyó la máquina. **(Shneiderman, 2005)**

Douglas recibe el apoyo por parte de las Fuerzas Aéreas de los Estados Unidos e inventa el ratón del ordenador y la primera interfaz gráfica en 1960. Culminó su trabajo en 1968 con una demostración ante miles de personas y es considerado como el padre de la Interfaz Gráfica de Usuario. **(Reimer, 2005)**

Esa interfaz gráfica creada por Douglas en los laboratorios de Xerox, fue introducida en las computadoras de Apple - Macintosh en 1984 y no fue hasta el 1993 y dado por la necesidad de eliminar muchas de las barreras existentes hombre – ordenador, que

se llevó a la población, con la primera versión popular de Windows 3.0 como sistema operativo. **(Reimer, 2005)**

En la actualidad la GUI es un elemento fundamental para la producción de software, porque refleja el rostro del mismo y en cierto grado, delimita su carácter competitivo. Las GUIs son interfaces muy amigables, rápidas, funcionales y flexibles, porque su misión fundamental es brindar mayores facilidades a los usuarios en el trabajo con el sistema.

1.6 Características de una GUI

En la actualidad, una buena interfaz gráfica de usuario debe contar con un conjunto de características que podrían sintetizarse en:

Características de un GUI:

1. Facilidad de comprensión, aprendizaje y uso.
2. Facilidad de identificación del objeto de interés
3. Sigue el paradigma de la interacción objeto-acción.
4. Permite la transferencia de información entre programas.
5. Se puede manipular en la pantalla directamente los objetos y la información.
6. Existe una muestra visual de la información y los objetos (iconos y ventanas).
7. Proporciona respuesta visual a las acciones del usuario.
8. Existe información visual de las acciones y modos del usuario/sistema (menús, paletas).
9. Existen controles gráficos (widgets) para la selección e introducción de la información.
10. Permite a los usuarios personalizar la interfaz y las interacciones.
11. Proporciona flexibilidad en el uso de dispositivos de entrada (teclado/ratón).
12. Las operaciones serán rápidas, incrementales y reversibles, con efectos inmediatos. **(Toledo, 2008)**

Una característica importante es que el GUI permite manipular los objetos e información de la pantalla, no sólo presentarla.

Para usar un GUI, los usuarios deben conocer (o aprender) una serie de conceptos:

Organización del sistema diferentes tipos de iconos y efecto de las acciones sobre ellos, elementos básicos de una ventana, uso de los controles del GUI, uso del ratón.
Con el desarrollo de las tecnologías los usuarios se han acostumbrado a encontrar cierto nivel de sofisticación en el diseño gráfico, incluyendo el de las páginas Web.

1.7 Interfaces GUI Web

Cuando se produjo un fenómeno sorprendente, llamado a revolucionar la comunicación entre seres humanos: Internet y la WWW. Con la aparición de la web se hizo posible que cualquier persona pudiera ofrecer información particularizada a los demás y encontrar documentos interactivos sobre cualquier tema, relacionados unos con otros mediante enlaces que permitían saltar de página en página alrededor del mundo. Las páginas web supusieron la aparición de las interfaces web, interfaces gráficas de usuario con unos elementos comunes de presentación y navegación que pronto se convirtieron en estándares de facto estos no son más que normas o estándares en los que se hace referencia a documentos o herramientas que son seguidas y usadas por la industria como si hubieran sido aprobadas por organismos oficiales, pero que en realidad no lo son. Este tipo de interfaces deben servir de intermediarias entre unos usuarios genéricos, no acostumbrados generalmente al uso de aplicaciones informáticas. El diseño de las páginas web ha evolucionado con el tiempo hacia un esquema general perfectamente definido, ofreciendo unas interfaces bien definidas, con un conjunto de componentes gráficos, funcionales y similares que hacen posible que sea cual sea el usuario que accede a un sitio web cualquiera la comunicación entre ellos sea posible y efectiva. De esta forma se han definido elementos y agrupaciones de estos que han demostrado su utilidad y su comprensión por los usuarios, entre los que podemos destacar los sistemas de navegación, los dinteles, los pies de página, los formularios de entrada de datos, etc., que normalmente encontraremos en todas las páginas web y cuyo diseño y funcionalidad son similares en todas ellas.

1.7.1 Estándares Web

Estos no son más que un conjunto de recomendaciones dadas por el Word Wide Web Consortium (W3C) y otras organizaciones internacionales acerca de cómo crear e interpretar documentos basados en el Web. Son un conjunto de tecnologías orientadas a brindar beneficios a la mayor cantidad de usuarios, asegurando la vigencia de todo documento publicado en el Web. El objetivo es crear una Web que trabaje mejor para todos, con sitios accesibles a más personas y que funcionen en cualquier dispositivo de acceso a Internet. Existen varios estándares que garantizan una mejor calidad, accesibilidad y rapidez de las aplicaciones web, "un estándar no es más que un modelo a seguir al hacer algo. Son documentos que dan los detalles técnicos y las reglas necesarias para que un producto o tecnología se use correctamente". **(W3C, 2008)**

1.7.2 Estándares Web Del W3C

- **HTML 4.01 Specification**

Es un estándar que permite la estructuración entre diversos lenguajes. Desde sus inicios, el HTML ha incorporado nuevas versiones y es justamente la última de ellas, el HTML 4, el que será estudiado a fondo por este grupo de trabajo para lograr que este formato sea interoperable y compatible con otras tecnologías, adaptándose así a las necesidades del mercado. **(W3C, 2008)**

- **XHTML Modularization Specification**

XHTML es una "reformulación de los tres tipos de documento definidos por HTML 4, pero como aplicaciones de XML **(W3C, 2008)**

- **XML Base**

Es un formato que permite la lectura de datos sin importar el hardware, software o sistema operativo que estén utilizando en las organizaciones.

Es un Lenguaje de Etiquetado Extensible simple, pero estricto, que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML (lenguaje en que se escriben las páginas WEB) pero su función principal es describir datos y no mostrarlos como es el caso de HTML. Las tecnologías

XML son un conjunto de módulos que ofrecen servicios útiles a las demandas más frecuentes por parte de los usuarios. XML sirve para estructurar, almacenar e intercambiar información. **(W3C, 2007)**

- **CSS level 2 revisión 1**

Las Hojas de Estilo (CSS) describen como es que los documentos son presentados en pantalla, de forma impresa, o quizás como son pronunciados. Sirve para añadir estilo a documentos HTML y XML. **(W3C, 2007)**

1.7.3 Patrones de diseño

Para lograr una capa de interfaz más amigable para los usuarios y a su vez más robusta se pusieron en práctica varios patrones de diseño dentro de los que tenemos:

Module Tabs: Es utilizado para que lo usuarios puedan navegar en el sitio sin tener que actualizarlo. También es usado cuando no hay suficiente espacio en el sitio web para mostrar todo el contenido dentro de todas la pestaña.
Vertical Dropdown Menú: Se utiliza cuando hay más de 2 secciones principales de un sitio. Es utilizado para mantener más desahogada la aplicación y contenga más espacios, y no luzca muy cargada.
Input Prompt: Es utilizado cuando él o los usuarios deben introducir datos en un sistema.
Good Defaults: Se utiliza generalmente para cuando el usuario no puede escribir nada solo marcar o seleccionar el dato que quiera.
Structured Format: Se utiliza cuando la entrada de la que desea cobrar es un tipo de datos. Por ejemplo un código postal, una fecha o la hora, un número de teléfono. También es utilizado cuando se necesita seleccionar un campo, una casilla de verificación o botones. **(Toxboe, 2009)**

1.7.4 Framework

En el desarrollo de software, un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un frameworks puede incluir soporte de programas, bibliotecas y un lenguaje de scripting entre otros softwares para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Un frameworks representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio. Los Framework son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional. Framework es un concepto sumamente genérico, se refiere a “ambiente de trabajo, y ejecución”, por ejemplo “.Net” es considerado un “frameworks” para desarrollar aplicaciones (Aplicaciones sobre Windows). En general los frameworks son soluciones completas que contemplan herramientas de apoyo a la construcción (ambiente de trabajo o desarrollo) y motores de ejecución (ambiente de ejecución). Un frameworks agrega funcionalidad extendida a un lenguaje de programación; ésta, automatiza muchos de los patrones de programación para orientarlos a un determinado propósito. Un framework proporciona una estructura al código y hace que los desarrolladores escriban código mejor, más entendible y manejable. Además hace la programación más fácil, convirtiendo complejas funciones en sencillas instrucciones. Está usualmente escrito en el lenguaje que extiende.

También se puede definir que un " Framework" es un diseño reusable de un sistema (o subsistema). Está expresado por un conjunto de clases abstractas y el modo en que sus instancias colaboran para un tipo específico de software.

Los framework web no son más que una estructura definida, reusable en el que sus componentes facilitan la creación de aplicaciones web. En cierto sentido podemos afirmar que nos proveen una capa de abstracción sobre la arquitectura original ocultándola o adaptándola para no tener que utilizar el protocolo http de manera nativa y así acelerar los tiempos de desarrollo y mantenimiento. A medida que la Internet fue evolucionando también lo hicieron con ella los frameworks webs, y con ello los desarrollos abandonaban el modelo cliente servidor para pasarse a los clientes livianos se puso más énfasis en la construcción de marcos de desarrollo más potentes. Con todos estos sucesos fueron apareciendo diferentes tipos de frameworks de

javascript para facilitar estandarización, el cross-browsing además de su eficiente trabajo y tiempo de programación. Estas librerías suelen permitir encadenar código, por lo que nos facilita mejor mantenimiento y menos tiempo de desarrollo, suelen ser rápidas a la hora de realizar acciones así como ayuda a la creación de interfaces de usuario avanzadas necesarias para hacer proyectos de la web 2.0. Entre los más conocidos se encuentran:

1.7.4.1 Frameworks de JavaScript más usados

MooTools:

Mootools es una librería de JavaScript orientado a objetos, compacto, modular y diseñado para escribir código compatible y extensible de forma rápida y sencilla. MooTools permite realizar el trabajo de forma eficaz y eficiente. **(Team, 2009)**

Características:

- **Core:** Funciones principales, basado en prototype.
- **Class:** Permite la creación de clases reutilizables.
- **Native:** Simplifica el trabajo y definición de arrays, funciones, tipos de datos (numéricos y cadena) y elementos DOM.
- **Element:** Funciones adicionales para trabajar con elementos DOM (selección, filtros, formularios, eventos, dimensiones).
- **Window:** funciones especificas para el objeto Window (DomReady y cambio de dimensiones).
- **Remote:** Simplifica el uso de AJAX, Cookies, Json.
- **Effects:** Diferentes efectos de animación. Es parte de otro proyecto moo.fx (una librería de efectos gráficos realmente ligera, 30Kb, compatible tanto con mootolols como con prototype).
- **Drag:** Funciones de drag and drop.
- **Plugins:** Unas 10 utilidades adicionales (hash, scroller, sortables, tips,…)

MooTools como Moo.fx son Open Source (MIT license).

JQuery:

Según indican en la Web, jQuery es una librería JavaScript rápida y concisa que simplifica la forma de trabajar con documentos HTML, manejando eventos, realizando animaciones y añadiendo interacción AJAX en tus páginas Web. JQuery ha sido diseñado para cambiar tu forma de escribir JavaScript. Exiten un gran número de aplicaciones Web 2.0 que usan esté FrameWork, por ejemplo, Wordpress se ha pasado a JQuery en su última versión (versión 2.2). **(Team, 2009)**

Características:

- Selectors: jQuery selectors son una combinación de CSS 1-3, XPath, junto con funciones de código especiales que permiten que funcionen conjuntamente. De forma realmente sencilla podemos seleccionar diferentes elementos del árbol DOM mediante XPath.
- Attributes: Permiten seleccionar, cambiar, leer, añadir y borrar los atributos de los elementos DOM de forma sencilla.
- Traversing: Funciones de filtrado y búsqueda.
- Manipulation: Funciones de inserción y modificación.
- CSS: Funciones para el cambio de estilo.
- Events: Control de eventos para elementos DOM.
- Effects: Sencillos efectos visuales. Si no fuera por los plugins que hay desarrollados para jQuery quedaría muy por debajo de otros FrameWorks en este punto.
- Utilities: Otras funciones genéricas adicionales, permiten referenciar objetos sin problemas de espacios de nombres.

Otras de las características importantes de jQuery que merece la pena mencionar son:
Mejora la gestión del espacio de nombres, evitando el uso de variables globales.
Multitud de plugins disponibles que se integran y amplían las funciones de jQuery.

Dojo:

El FrameWork JavaScript Dojo permite añadir características dinámicas fácilmente a las páginas Web. Puedes utilizar los componentes que incorpora Dojo para mejorar la usabilidad y funcionabilidad. Puedes construir interfaces con degradados de color, widges y transacciones animadas de forma rápida y sencilla. También puedes usar el API de bajo nivel y capas compatibles para escribir scripts portables y simplificar aquellos que son complejos.

Dojo también proporciona un potente entorno de programación incorporando un sistema de control de eventos, un API E/S y un lenguaje estándar mejorado. Puedes usar las herramientas de Dojo para construir unidades de test para tu código Java Script y optimizar el desarrollo. **(Team, 2009)**

Características:

- **Múltiple Points Of Entry:** Es un concepto fundamental en el diseño de Dojo. Este término significa que los usuarios pueden empezar a utilizar Dojo al nivel que ellos deseen.
- **Independencia del interpretador:** Asegurar soporte para el mayor número de plataformas.
- **Unifica varios codebases:** incorpora en varios FrameWorks (nWidgets, Burstlib).

Qooxdoo:

Qooxdoo es uno de los FrameWorks JavaScript AJAX más sencillos e innovadores. Es Open Source bajo licencia dual LGPL/EPL. Incluye soporte para desarrollo profesional Java Script, un GUI toolkit y comunicaciones cliente-servidor avanzadas. **(Team, 2009)**

Características:

- **Detección del cliente:** qooxdoo reconoce y permite acceder a la información del navegador está siendo utilizado.
- **Capa de abstracción de navegador:** Incluye una capa de abstracción que intenta abstraer las características especificas de cada navegador en una capa común. Esta característica simplifica el desarrollo en JavaScript, permitiendo acceder a estilos, posiciones (relativas a página, cliente y pantalla) y otras funciones básicas.

- **Implementación de propiedades avanzadas:** qooxdoo soporta "verdaderas" propiedades para objetos. Cada clase puede definir propiedades que tendrán las instancias creadas. Con addProperty se añadirán de forma automática las funciones get y set.
- **Gestión de eventos:** Integra su propia interfaz de eventos, incluyendo funciones para dar de alto y de baja eventos. En adición, incorpora la posibilidad de llamar a la función desde cualquier contexto y es independiente del navegador utilizado.

Ext JS:

ExtJS empezó siendo un conjunto de librerías y extensiones para YUI (Yahoo! User Interface). Con el tiempo se convirtió en un FrameWork independiente y a principios de 2007 se creó una compañía para comercializar y dar soporte del FrameWork Ext. De esta forma Ext tiene dos tipos de licencias, LGPL y comercial. Básicamente, se pude usar EXT para los proyectos que se deseen; pero solo obtener soporte si se tiene licencia comercial. Este FrameWork tiene algunas características muy importantes como son: **(Frederick, 2008)**

Características:

- **Ext.Element:** Representa un elemento del árbol DOM. Muchas de las funciones de manipulación de los elementos tienen un parámetro opcional que permite realizar el cambio mediante un efecto de animación. El parámetro de animación puede ser un dato boleano o un objeto que incluye las opciones de la animación.
- **Ext.BorderLayout:** Esta clase representa un diseño común para ser usado en aplicaciones de escritorio.
- **Ext.DomHelper:** Utilidades para trabajar plantillas o DOM. Suporta el uso de DOM o fragmentos de HTML de forma transparente.
- **Ext.TabPanel:** Un ligero contenedor de tabs.
- **Ext.UpdateManager:** Proporciona soporte para actualización AJAX de los objetos Element.

Es una librería que nos permite potenciar la capa UI de Javascript en nuestras aplicaciones. Para ello nos ofrece una serie de librerías (compatibles con PHP 4 y 5) para integrar Ext JS en nuestro sistema. Funciona como un mapeado en clases de la librería JS. Entre las posibilidades que ofrece nos encontramos con la creación de formularios, combos, grids o menus. A parte ayuda a la comunicación entre el cliente y el servidor mediante JSON y XML.

Entre las bondades de este framework tenemos:

Modelo de Componentes.
Modelo de Contenedores.
Capas.
Grid.
PlantillasX.
Vistas de Datos.

Sirve de puente entre las librerías JS más usadas (Prototype, JQuery, YUI)
Tiene Componentes UI del alto rendimiento y personalizables.
Modelo de componentes extensibles.
Un API fácil de usar.
Se puede apreciar una reducción de consumo de memoria en el navegador considerable. En aplicaciones normales el consumo baja hasta casi la mitad aunque se calcula que puede llegar en aplicaciones pesadas hasta un 300% menos.
Usar un motor de render como ExtJS nos permite tener además estos beneficios:
Existe un balance entre Cliente – Servidor. La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo.
Comunicación asincrónica. En este tipo de aplicación el motor de render puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente se dé cuenta. El tráfico de red puede disminuir al permitir que la aplicación elija que información desea transmitir al servidor y viceversa, sin embargo la aplicación que haga uso de la pre-carga de datos puede que revierta este beneficio por el incremento del tráfico.

Es una librería de JavaScript para la construcción de aplicaciones ricas de Internet, admite todos los principales navegadores web, incluyendo: Internet Explorer 6 +, Firefox 1.5 + (PC, Mac), Safari 3+ Opera 9 + (PC, Mac).
Después de conocer las características de los framework analizados anteriormente, se llega a la conclusión de que no existe uno que aporte una solución a todos los problemas, sino que cada uno tiene puntos en los que son superiores a los demás. Para nuestra solución en particular, la mayoría cumplen con las características necesarias de requerimientos, pero principalmente ExtJS y Dojo.

Entre ambos en el momento de definir uno para el desarrollo, nos enfocamos en cuál de los dos existe una mayor facilidad para acceder a la documentación y detalles de su implementación interior como es el API del framework. Además de revisar la cantidad de soluciones informáticas que se desarrollan en nuestra universidad en la que nos puedan brindar capacitación. Analizando todos estos puntos se define que ExtJS es el framework que se acerca de una forma más directa a todo el análisis realizado.

1.8 IDE de Desarrollos

➢ Spket IDE

Spket IDE es una excelente aplicación que ofrece la posibilidad de editar en lenguaje de programación JavaScript, XUL/XBL, Laszlo, SVG, Silverlight y Yahoo! Desarrollo de Widgets (componentes), dentro de las numerosas características de Spket IDE, podemos destacar algunas como el autocompletado de comandos, diferenciación por colores de la sintaxis, resumen de contenido que ayuda a los desarrolladores a crear código JavaScript eficiente de forma productiva, además la comunidad de Extjs está utilizando este IDE para el desarrollo de los temas del FrameWork, etc. **(Team, 2008)**

Pude funcionar como una aplicación independiente o como un plugin de Eclipse.
Nuevas características que incluyen:
1. Toggle Mark Occurrences para javascript añadido
2. Adición/Eliminación de comentarios para javascript.

Cuenta con un funcionamiento totalmente sencillo para todo aquel programador profesional o aficionado y posee una interfaz gráfica verdaderamente eficiente y completa para la edición de nuestras aplicaciones.

Requerimientos necesarios:

- Java 1.5.

Tipo de licencia

- Programas de dominio público

Sistema Operativo

- Win 98,2000,XP

➢ Cómodo IDE

Komodo IDE (Integrated Development Environment) es un editor de código escrito en XUL capaz de trabajar con JavaScript, Perl, Python, Ruby, PHP, XSLT y muchos otros lenguajes más tanto para Windows como para Linux.

Komodo IDE cuenta con sintaxis coloreada para remarcar errores al escribir y dispone de un servidor FTP integrado desde el que se puede sincronizar tus archivos locales (mediante la implementación de plug-ins).

Otro aspecto a destacar de este cómodo editor, es la función de autocompletado que acelera el trabajo cada vez que necesites volver a escribir una misma línea de código.

Komodo IDE integra un editor avanzado, debugger gráfico y multitud de preferencias configurables. Al soportar tantos lenguajes y plataformas, Komodo IDE elimina la necesidad de utilizar variadas herramientas de programación específicas, simplificando tu trabajo desde un único programa. Además su tipo de licencia es de programas de libre evaluación. **(Copeland, 2009)**

Licencia:

- Comercial

Plataforma:

- Multiplataforma

Sistema Operativo

- Windows, Linux y Mac OS

Plataforma:

- Multiplataforma

Después de conocer algunas de las características de las herramientas analizadas anteriormente se decidió utilizar la herramienta Spket IDE para darle solución al problema planteado ya que la misma posee múltiples cualidades que hacen el desempeño del trabajo más sencillo y a la vez con la calidad requerida.

1.9 Conclusiones del Capítulo

En el presente capítulo se hizo una descripción de los conceptos asociados a las principales actividades que se deben realizar para el desarrollo de la capa de presentación y se valoraron los frameworks más utilizados en el mundo así como un estudio de los diferentes patrones de diseño y de estándares web empleados para la implementación de interfaces de usuario, además se hizo valoraciones de diferentes herramientas utiliza en el desarrollo de la capa de Presentación del módulo “Trabajo Educativo” del Sistema de Gestión de Residencia para concluir con la necesidad del desarrollo en esta capa, una interfaz amigable, funcional y rápida, teniendo en cuenta el empleo eficiente de estas herramientas y tecnologías.

CAPITULO 2: DESCRIPCION DE LA SOLUCION PROPUESTA

2.1 Introducción al capítulo

En este capítulo se dará una panorámica de cómo la solución propuesta dio respuesta al problema y se describirán elementos fundamentales a tener en cuenta en el desarrollo de la solución.

Además se realizará una descripción de la capa de Presentación que le permiten al usuario intercambiar con el sistema. Incluye también la descripción de algunos componentes que nos brinda la librería Extjs.

Se describirán todos los estándares de codificación utilizados para el desarrollo de las interfaces del módulo “Trabajo Educativo” del Sistema de Gestión de la Residencia.

2.2 Solución Propuesta

2.2.1 Pautas Generales para el diseño de los prototipos de interfaz

En el mundo se ha definido un conjunto de pautas generales necesarias para lograr un mejor diseño de interfaces de usuarios, más amigables accesibles y funcionales, así como una mayor interacción entre el usuario y el producto a desarrollar. A continuación se describen algunas de las utilizadas en la solución propuesta.

- Pida solo la información absolutamente necesaria
- Infiera información a partir de otra disponible
  Por ejemplo, la provincia se puede inferir del C.P.
- Reutilice los campos cuando sea posible.

  Por ejemplo, el email puede servirnos en ocasiones como nombre de usuario.
- No pida la información dos veces.

  Por ejemplo, si el usuario ha rellenado la dirección de facturación, no le obligue a volver a rellenar la dirección de envió si no es necesario, pregúntele si quiere que sea la misma. **(Martín, 2009)**
- Proporcione un titulo al formulario que exprese claramente su función
- Utilice una nomenclatura clara y familiar, sin tecnicismos ni extranjerismos.
- No utilice preguntas complejas ni haga pensar al usuario.

- Redacte siempre las opciones de forma afirmativa. Por ejemplo, junto a un *check* escriba “Si Deseo recibir el boletín" en vez de "No deseo recibir el boletín".
- Organice los campos en una sola columna de datos.

Sin entrar en las razones de accesibilidad que lo justifican me cuesta muchas discusiones hacer comprender que, apelotonando los datos en la misma línea o colocándolos en varias columnas, se pierde tanto a nivel de usabilidad que no merece la pena el espacio vertical que se gana.

Como siempre, hay muchos contextos de uso y excepciones justificables, como los formularios que se rellenan de forma repetitiva y constante, pero la excepción nunca puede convertirse en norma.
Una sola columna funciona mejor. Los formularios con dos columnas tienen más probabilidades de que los usuarios pasen por alto algunos campos, dado que crean un orden ambiguo de lectura. Sus ojos se moverán hacia donde espera encontrar el próximo campo, que será habitualmente hacia abajo, en vertical. No esperan a que se les indique mediante el parpadeo del cursor donde mirar.

- Organice los campos en grupos lógicos, utilizando para ello la mínima cantidad de elementos visuales (evitando así ruido visual).
- Agrupe, si es posible, los campos obligatorios al comienzo del formulario.
- Proporcione un diseño ordenado, alineando verticalmente todas las etiquetas y todos los campos entre sí.

Todos los campos deben estar verticalmente alineados entre sí a la izquierda.

¿Como alinear las etiquetas entre sí: a la derecha, a la izquierda o las colocamos encima del campo?

- Si tenemos que rellenar datos que son familiares (y no son muchos): Etiquetas en vertical encima del campo.
- Cuando necesitemos ajustar el espacio vertical: Etiquetas a la izquierda del campo, alineadas a la derecha.
- Hay que ajustar el espacio vertical, y los datos no nos son familiares o son complejos: Etiquetas al lado del campo, alineadas a la izquierda. **(Best Practices for Form Design, 2008)**

- Situé las respuestas de los campos ***radio buttons*** y ***check box*** después de los mismos**.**

  De esta manera se favorece la alineación vertical de todos los controles.
- Utilice etiquetas estándar para agrupar campos y hacer más manejable la información(OPTGROUP, FIELDSET)
- Si se utilizan ***radio buttons*** o ***checks box*** agrupe visualmente de forma clara y unívoca los distintos grupos de opciones.
- Homogeneíce los anchos de los campos de texto cuando estos sean similares (evitando así ruido visual).
- Dote a los campos de texto de flexibilidad para que admitan los datos en cualquier formato.

  Por ejemplo, un campo para introducir el número teléfono debería admitir paréntesis, guiones, espacios; un campo para introducir importes debería admitir decimales con punto o con coma, etc. **(Padilla, 2008)**
- Si se utilizan *radio buttons* asegúrese de que todas las opciones son claramente excluyentes.

  - No los utilice cuando las respuestas sean más de tres y complejas, o más de cinco y simples.
  - Siempre que se cumpla la regla anterior, utilice *radio buttons* en vez de combos
- Si un *radio button* tiene más de dos respuestas, colóquelas en vertical, unas debajo de otras alineadas a la izquierda. **(Alertbox, 2004)**

- Valore la posibilidad de evitar, mediante Java Script, que en determinados campos se pueda introducir determinados caracteres.

  Por ejemplo, que en el campo DNI solo se puedan introducir números y letras, haciendo que el resto de caracteres no se puedan teclear en el campo. **(Padilla, 2008)**

- No implemente saltos automáticos del foco del formulario.

  Por ejemplo, en los campos de cuenta, no haga que el foco se mueva sólo al siguiente campo cuando se ha rellenado el anterior. **(Manchón, 2003)**

- Un error típico es introducir el salto automático entre campos de texto consecutivos y hacer innecesario el uso del tabulador.

  Aunque este comportamiento puede parecer que facilita la tarea de introducción de datos, no es adecuado porque quita control a los usuarios, no es un funcionamiento estándar y es necesario mirar la pantalla para saber en qué campo se está.

  Todo ello puede provocar fácilmente errores, como por ejemplo, introducir datos pertenecientes a un campo en el siguiente cuando no se introduce el formato esperado por el salto automático.
- Asegurase de que la tecla "Enter" realiza la acción para los cual este destinado el formulario
- Evite, mediante JavaScript, que el usuario pueda impacientarse y enviar dos veces el formulario. **(Padilla, 2008)**
- Identifique claramente los campos obligatorios y los opcionales mediante el literal (Obligatorio) u (Opcional), según si se van a marcar los obligatorios o los opcionales, colocando dicho literal detrás de la etiqueta del campo y por tanto antes del campo.

  Para saber si marcar los obligatorios o los opcionales.
  - Indique los campos obligatorios cuando sean menos que los opcionales.
  - Indique los campos opcionales cuando sean menos que los obligatorios.

    (Best Practices for Form Design, 2008)
- Incluir ayudas breves o ejemplos junto a los campos, pero solo cuando sea realmente necesario para saber cómo ingresar un dato.
- Asegúrese de que al leer en línea estas ayudas o ejemplos se lean antes que el campo**,** por ello, un buen lugar para colocarlas es encima del campo.
- No incluya un botón "Reset" (es decir, de Limpiar o Borrar el formulario).
- En los formularios de un solo paso evite tener un botón "Cancelar" cuya función sea en realidad volver a la página anterior.
- Coloque los botones o enlaces que realizan las acciones primarias.

Por ejemplo el botón "Enviar" lo más cerca posible del último campo del formulario. No los separe del formulario mediante, por ejemplo, una línea.

- De un nombre adecuado a los botones del formulario, relacionado con su acción y no de carácter general.
- Por ejemplo, use "Buscar" en vez de un genérico "Aceptar".

(Alertbox, 2000)

- Destaque los campos que han dado error pero no se base para ello únicamente en el color.

  Acompáñelos de un icono de error que aparezca también en el resumen del comienzo de la página.

Repita el mensaje de error al lado del campo para no tener que volver a la lista inicial

- Cuando se produzca un error, el formulario no debe restearse**,** es decir, los campos no erróneos deben seguir manteniendo la información en ellos introducida.
- Redactar claramente los textos de error mediante términos claros, sencillos y no técnicos.

  No utilizar mensajes genéricos del tipo “No se ha podido enviar el formulario... ”
- Evite validar los campos uno a uno, cuando pierden el foco**,** mostrando inmediatamente un mensaje de error al usuario. A los usuarios les incomoda esta práctica.
- Cuando el usuario envié el formulario, infórmele del resultado de su acción: Indíquele si se ha realizado correctamente, qué datos se han enviado, como puede ponerse en contacto con los responsables del sitio si ha habido problemas o para hacer un seguimiento del mismo, o como puede modificar los datos enviados.
- Si el proceso de envío es lento, incluya en la página un mensaje de "enviando datos".
- Informe a los usuarios de por qué deben rellenar el formulario y cuándo y a través de qué medio recibirán una respuesta. **(Cabrera, 2004)**
- Asocie explícitamente las etiquetas con sus controles mediante LABEL y su atributo "for".
- Compruebe que el tabulador permite acceder a todos los campos en el mismo orden que el visual.
- Mejore la experiencia del usuario mediante JavaScript y AJAX pero asegúrese que el formulario funcione correctamente sin ellos. **(Padilla, 2008)**
- No establezca un límite de tiempo ajustado para complementar el formulario.
- Si los formularios son muy extensos la solución no son las columnas, sino la división en páginas bien rotuladas que indiquen al usuario en que paso está del proceso (por ejemplo Paso 3 de 4).

- Si el formulario se presenta en varias páginas hay que seguir el lema 1 tema = 1 página.
- El usuario debe poder volver a los pasos anteriores**.**
- **No solicite información externa** en medio del proceso mediante la abertura de una ventana nueva del navegador. **(Starling, 2001)**

2.2.2 Pautas Específicas para el diseño de los Prototipos de interfaz

Para el diseño del las diferentes interfaces a desarrollar, se consibieron un conjunto de pautas especificas para nuestras interfaces, dentro de las que podemos destacar:

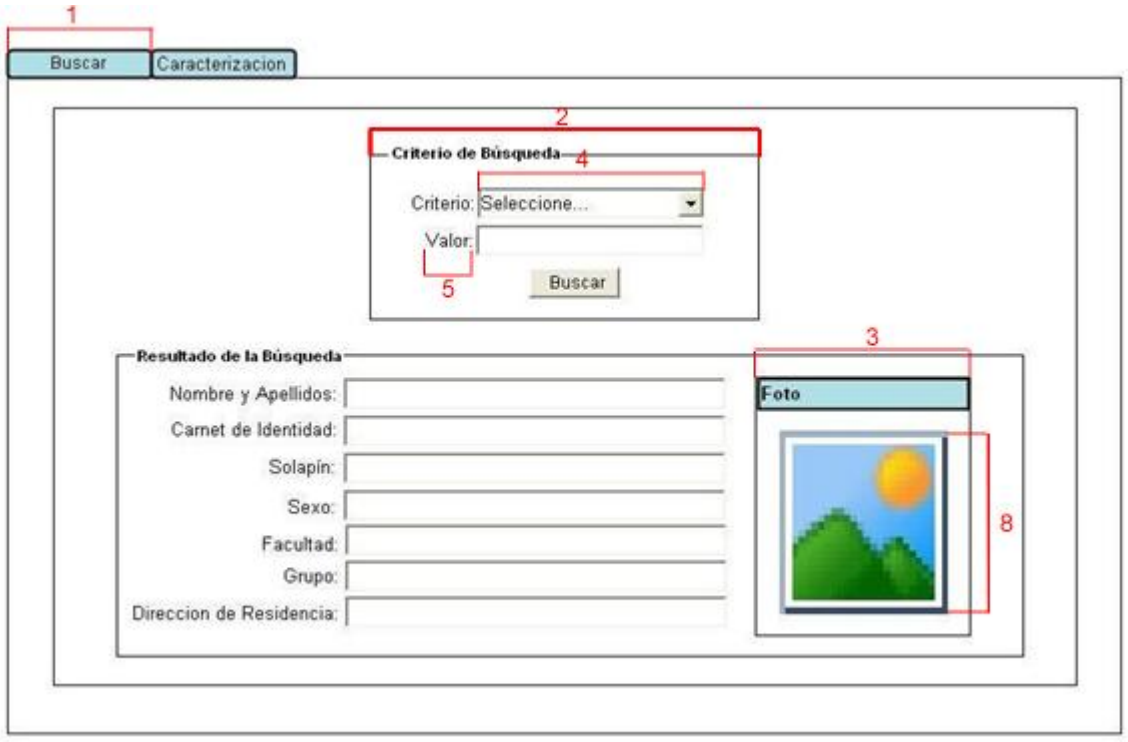


Fig. 2.2.2.1 Buscar

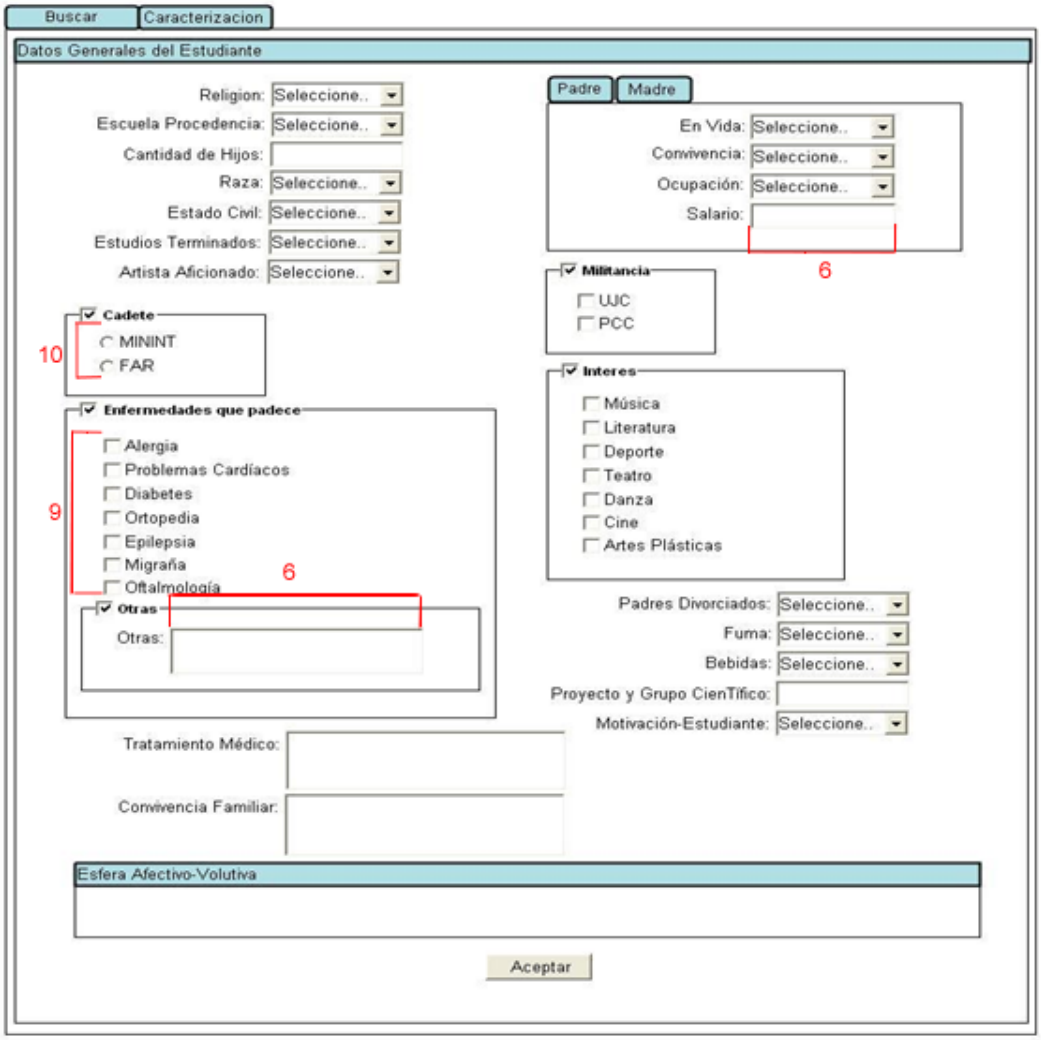


Fig. 2.2.2.2 Realizar Caracterización

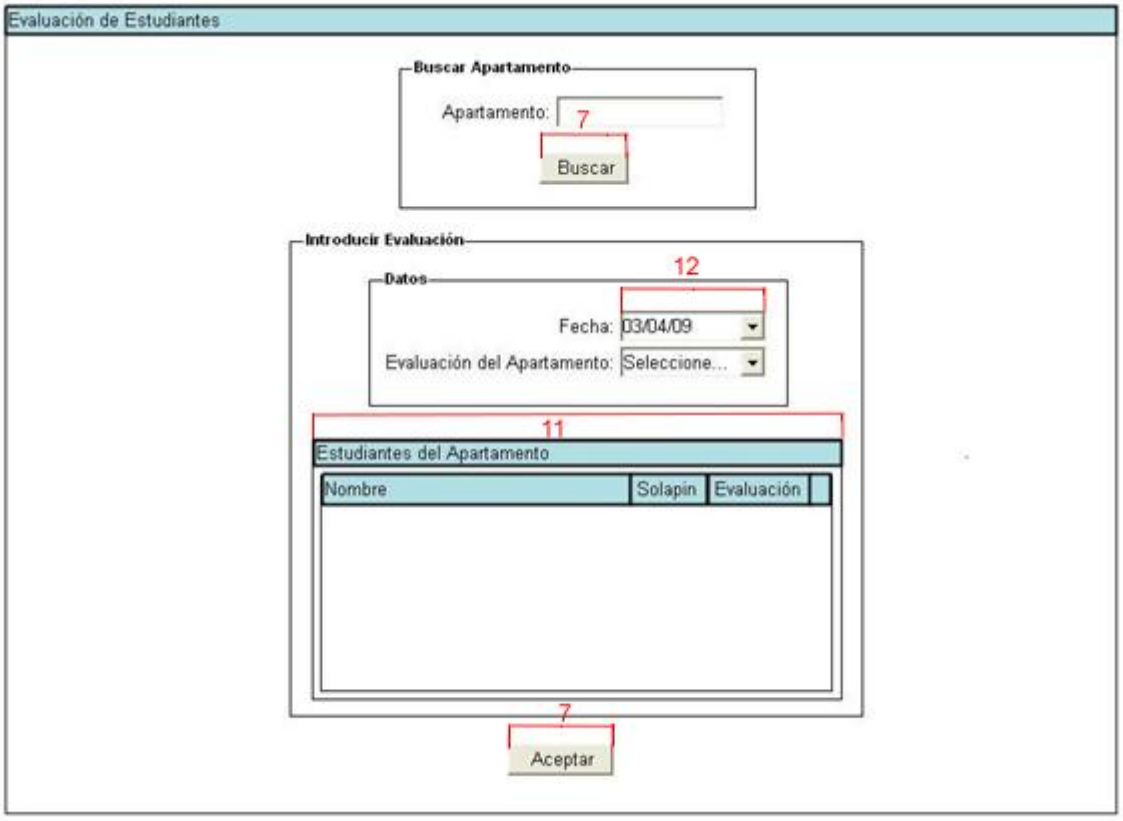


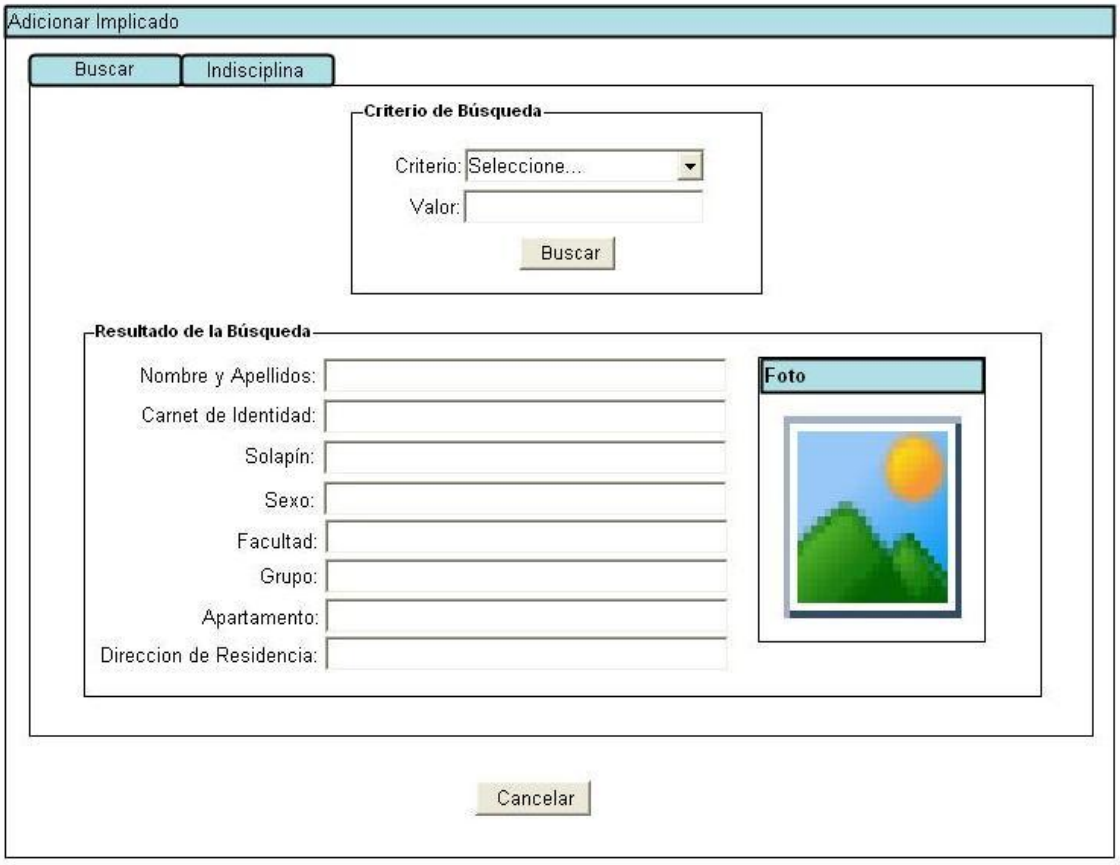
Fig. 2.2.2.3 Realizar Evaluación

La tipografía utilizada para todos los textos es Arial 10 ptos.

1. Los Tab tiene un título que representa el contenido que se encuentra dentro de cada tab, se encuentran alineados al centro a una separación de 10px, la letra es negra y resaltada y la separación entre un tab y otro es de 1px.
2. Se utilizará fieldset para agrupar información específica, el título del mismo representa lo que se va a tratar dentro del, tiene un color de letras azul oscuro y resaltada, el borde del mismo tiene que ser sencillo, el espacio entre el titulo y el borde es de 1px, el margen interior del fieldSet es de 29px, el espacio entre varios fieldSet es de 29px,
3. Los form tienen un titulo que representa lo que se va a tratar dentro del, tiene color de letra negra y resaltada, la separación del título y al borde del encabezado del los formPanel es de 4px.
4. Se utilizará selectores en todos los casos que se desee mostrar información que no se pueda modificar, ya sea por defecto o por otro medio, estos mostrarán la palabra “Seleccione” para brindar mayor información.
5. Los fieldLabel se utilizará para describir la información relacionada al componente asociado, se encuentra alineado a la derecha con un espacio es de 8px, su letra es normal y de color negro.

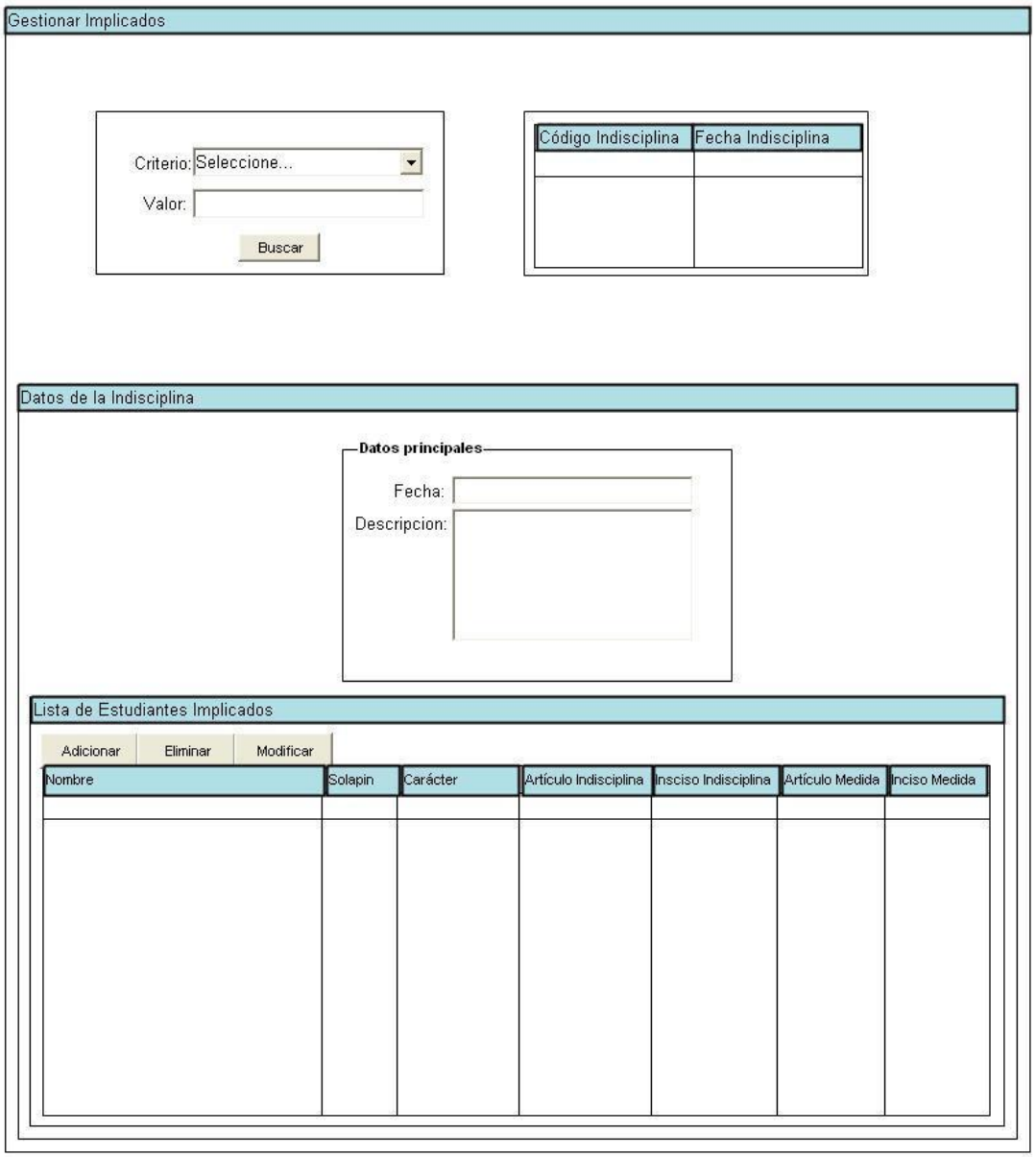
6. Se utilizará campos de texto de hasta 50 caracteres para escribir o mostrar datos no muy extensos y campos de más de 50 caracteres para datos relativamente largos.
7. Se utilizará botones aceptar para cuando se vaya a guardar información, botones buscar cuando se desea buscar información estos estarán situados al centro del formulario y botones cancelar cuando se desee deshacer algo. Los botones tendrán un icono que representa la lo que debe hacer el mismo, la letra será normal y alineada al centro a una separación de 10px.
8. Las fotos tiene que estar alineadas al centro y a una separación del borde de 10px.
9. En las enfermedades que padece el estudiante se utilizará chexbox para seleccionar varias enfermedades a la vez, estos tendrán una letra normal de color negra.
10. En el fieldset de cadete se utilizará radio button para marcar solo una de las variantes, estos tendrán una letra normal de color negra.
11. Los grid se utilizará para mostrar varios datos a la vez en forma de tabla, el titulo de cada columna del grid representa los datos que se van a mostrar en el mismo.
12. Se utilizará selectores de fechas para mostrar el formato fecha.

2.2.3 Breve descripción de las Interfaces que contiene la capa de Presentación



Prototipo. 2.2.3.1 Buscar

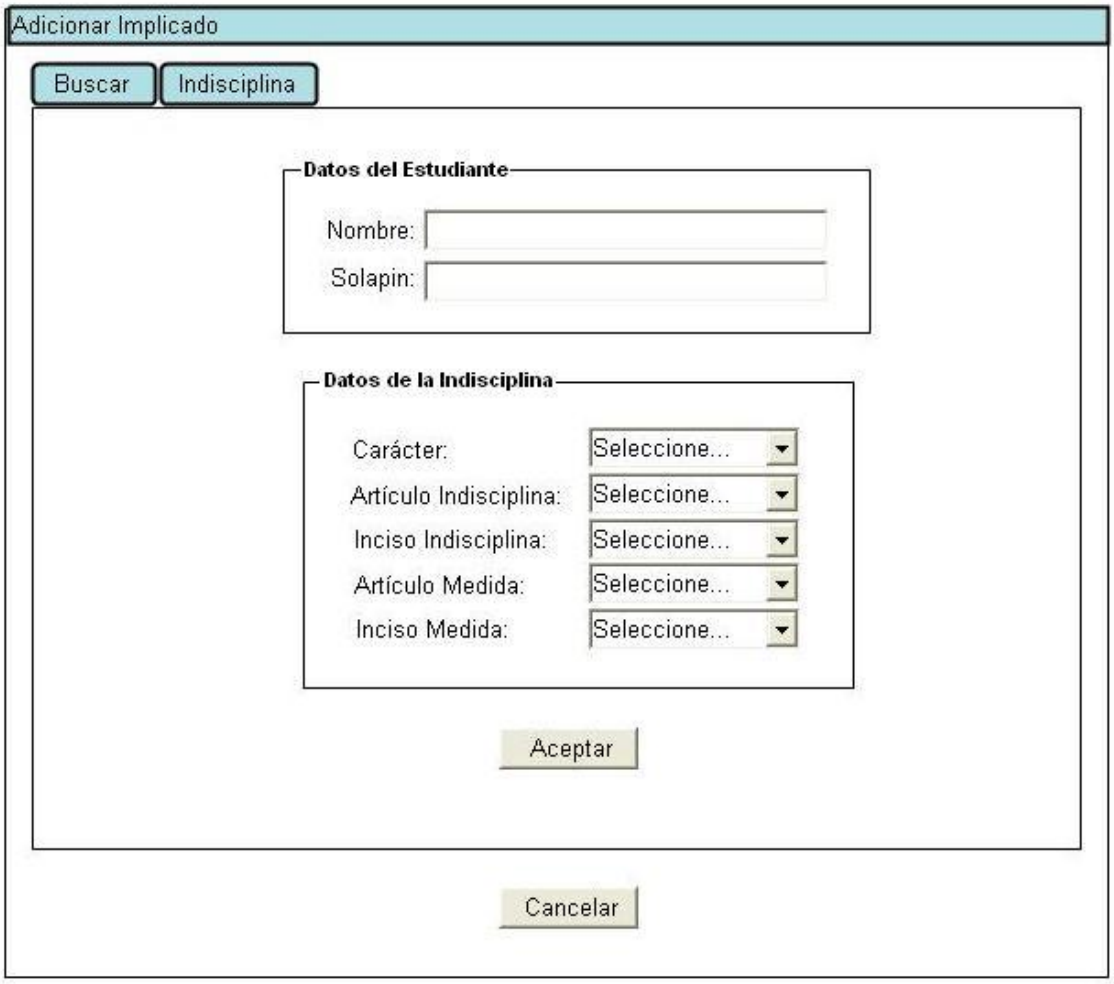
Este prototipo se utilizará para realizar la búsqueda de estudiantes, para esto solo se seleccionará en el selector “Criterio” la opción por la cual se va a buscar y en el campo de texto “Valor” se escribe el dato referente al criterio seleccionado, después se da clic en el botón para que se realice la búsqueda, si se devuelve algún resultado se completarán todo los demás campos de textos existentes en la pantalla con sus respectivos datos encontrados así como la foto de ese estudiante.



Prototipo. 2.2.3.2 Gestionar Implicado

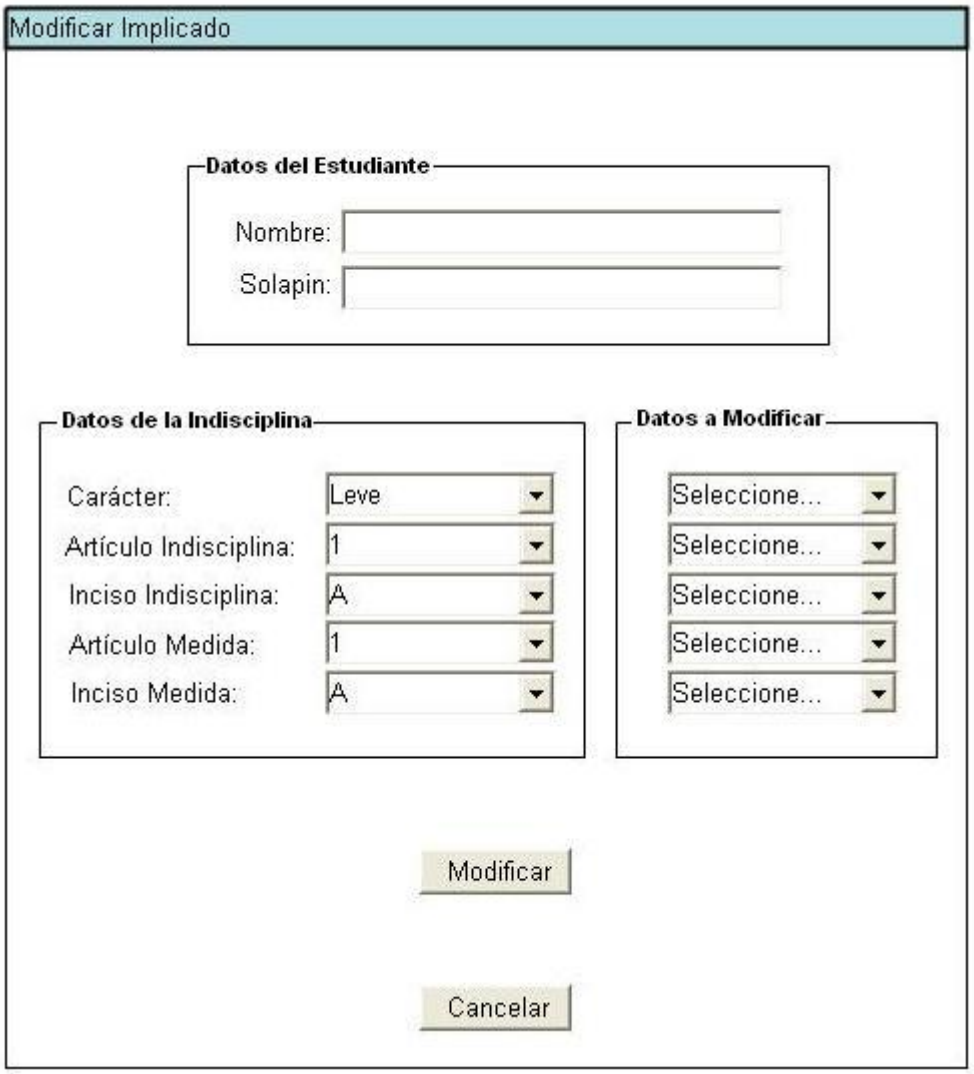
Este prototipo se utilizará para gestionar los implicados a una indisciplina donde para esto se selecciona un criterio por lo cual se deberá realizar la búsqueda para encontrar una indisciplina y en el campo valor se escribe el dato que corresponda con la opción que se seleccionó. Posteriormente se da clic en el botón y después se mostrará en la tabla el código de la indisciplina y la fecha en la cual fue realizada la misma. Luego se selecciona la indisciplina y se mostrará en el campo de texto "Fecha", fecha en que ocurrió esa indisciplina así como en el campo de texto "Descripción" una breve descripción de la misma. Se mostrara el listado de todos los estudiantes implicados, para agregar a un implicado en la indisciplina solo se da clic en el botón "Adicionar" y se adicionará el mismo en la tabla "Lista de Estudiantes Implicados", Para eliminar alguno de los que se encuentran ya se marcará uno y se da

clic en el botón “Eliminar” y para modificar alguno de los implicados se deberá de marcar un estudiante y se da clic en el botón “Modificar”.



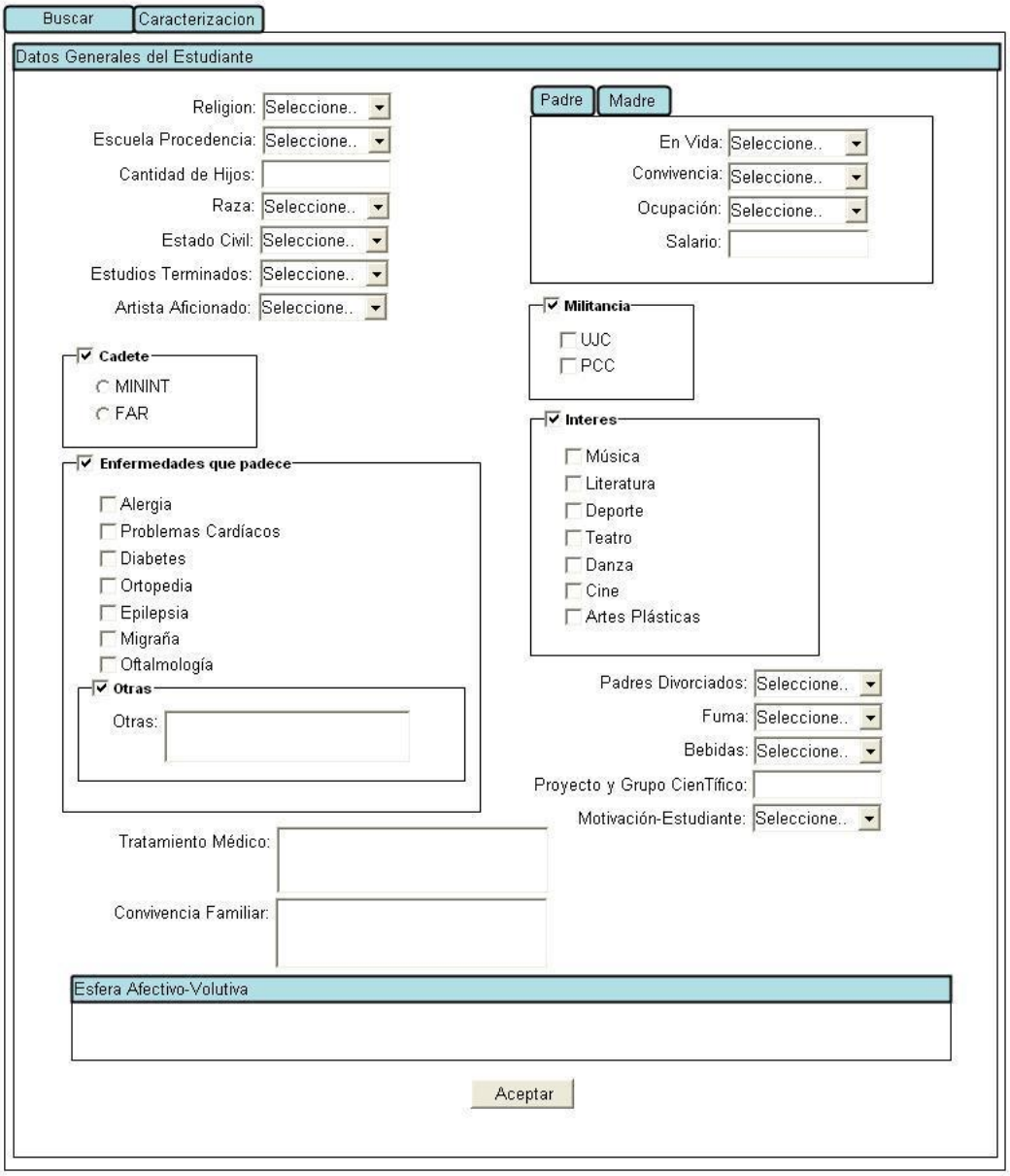
Prototipo. 2.2.3.3 Adicionar Implicado

Este prototipo se utilizará para adicionar implicados a una indisciplina, donde se mostrará algunos datos del estudiante encontrado después de realizar una búsqueda, en el campo de texto “Nombre” se mostrará el nombre del estudiante encontrado y en el campo de texto “Solapín” el solapín del mismo. Después se completará la indisciplina en la que fue implicado donde solo se seleccionará en el selector “Carácter “ el carácter de la indisciplina en el de “Artículo Indisciplina ” el artículo de la misma , en el de “Inciso Indisciplina “ el inciso de esa indisciplina , en el de “Artículo Medida ” el artículo relacionado con la medida que es tomada respecto a esa indisciplina y en el de “Inciso Medida ” el inciso de esa medida tomada, una vez llenada la indisciplina se da clic en el botón “Aceptar” para registrar esa información, y para deshacer todo lo realizado anteriormente solo se da clic en el botón “Cancelar”.



Prototipo. 2.2.3.1 Modificar Implicados

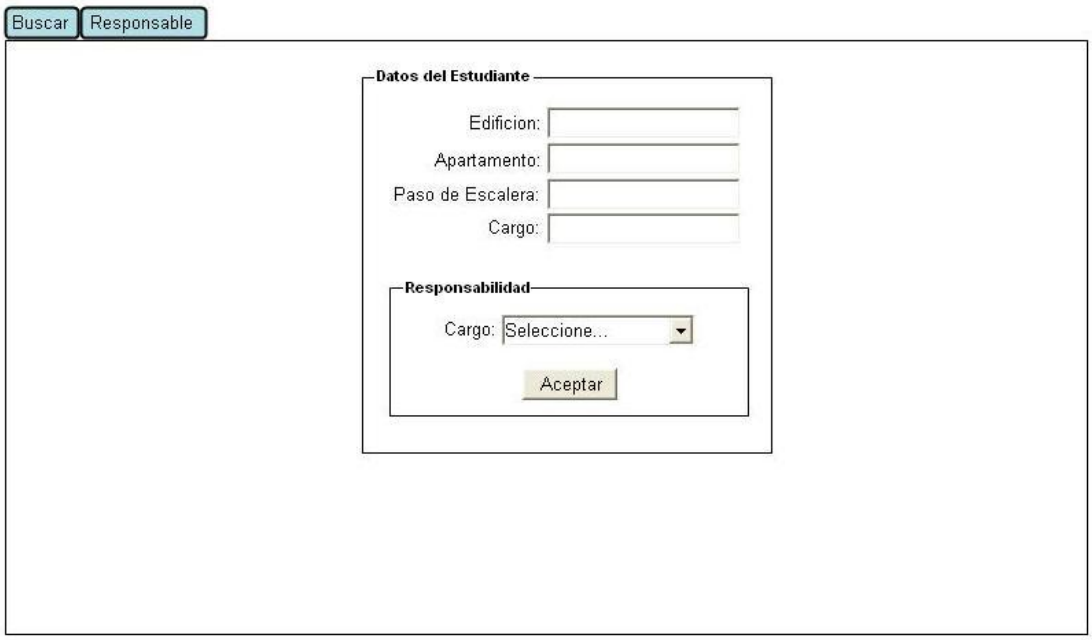
Este prototipo se utilizará para modificar los implicados de una indisciplina, donde se mostrará algunos datos del estudiante encontrado después de realizar una búsqueda, en el campo de texto “Nombre ” se mostrará el nombre del estudiante encontrado y en el campo de texto “Solapín ” el solapín del mismo, también se obtendrá los datos de la Indisciplina en la que ese estudiante fue implicado y para modificar esos datos en caso de que haya que hacerlo solo se seleccionara las diferentes opciones que aparecen en los selectores que se encuentran al lado de los datos anteriores de esa indisciplina, posteriormente se da clic en el botón “Modificar” para registrar esa información y para deshacer todo lo antes hecho solo se da clic en el botón “Cancelar” para realizar operación.



Prototipo. 2.2.3.2 Realizar Caracterización

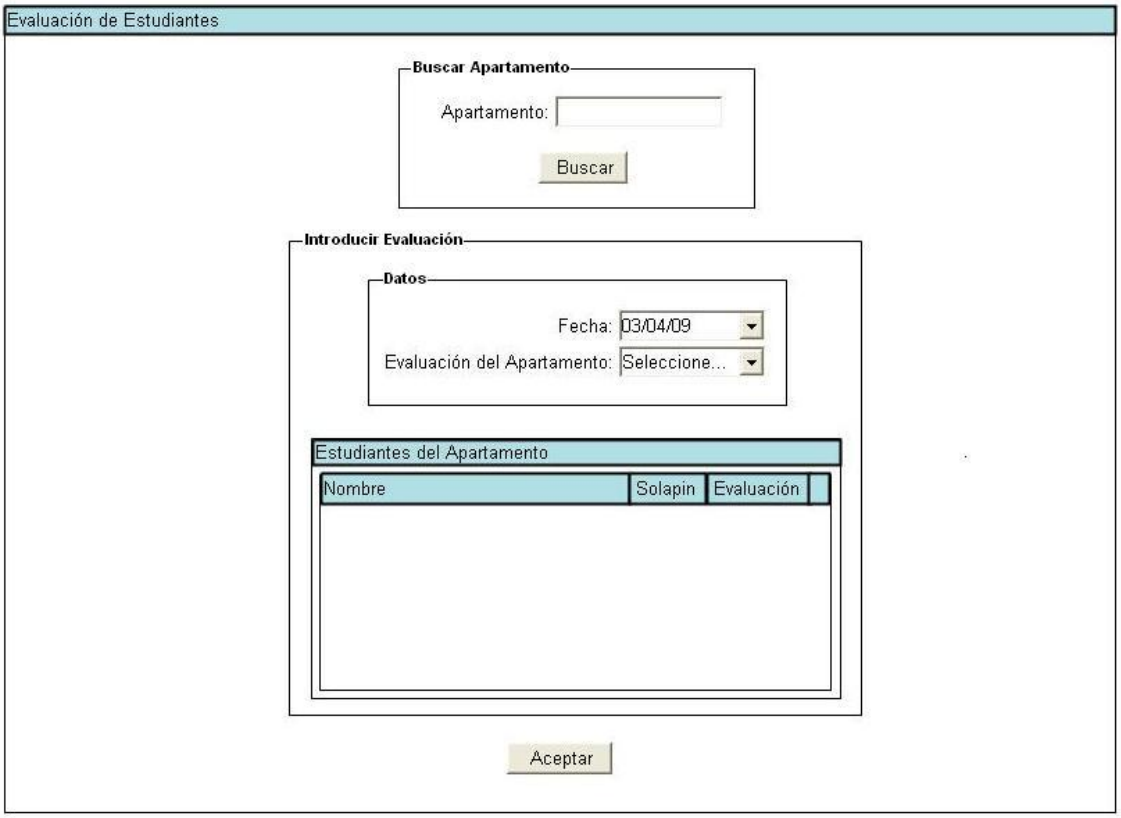
Este prototipo se utilizará para realizar la caracterización de cada estudiante para esto se realiza una búsqueda del estudiante al cual se le hará la caracterización y una vez hecho esto se activara el otro tab de al lado del buscar y se continua a realizar la misma. En los selectores “Religión”, “Escuela Procedencia”, “Raza”, “Estado civil”, “Estudios terminados” y “Artista Aficionado” se selecciona la opción que cumpla con ese estudiante y el campo de texto “Cantidad de hijos “se completa ese dato, también se llenaran los datos referente a los padres en los tab “Padre” , “Madre”, simplemente se selecciona en los selectores “En vida”,” Convivencia” y “Ocupación” lo referente a esa información así como en el campo de texto “Salario”. Posteriormente se continúa

completando información referente al estudiante donde en el chexbox “Cadete” después de marcado se despliega y aparecen 2 radiobutton en el cual se marcará solo uno, después se prosigue a completar lo referente a las enfermedades que padece desplegándose un conjunto de chexbox donde se marcarán todas las referentes con el mismo y en el caso que la enfermedad que posee dicho estudiante no se encuentra en las mostradas simplemente se escribe en el campo de texto “Otras” las enfermedades que tiene el mismo. Luego se continúa con la militancia de la persona y se marca la misma en los chexbox relacionados así como el interés que posee ese estudiante ya sea por la música, deporte, cultura etc. Para esto se marca cada uno de los chexbox relacionados con ese interés y se prosigue al completamiento de esa caracterización, donde se selecciona en los selectores “Padres Divorciados”, “Fuma”, “Bebidas” y “Motivación-Estudiante” la informacion referente a ese tema y en el campo de texto “Proyecto y Grupo Científico” se escribe la información relacionada, después se completa si el estudiante posee tratamiento médico y como es su convivencia familiar para esto se escriben en los campos de textos “Tratamiento Médico” y “Convivencia Familiar” toda la informacion sobre ese tema. Una vez completada toda esa información se da clic en el botón “Aceptar” para almacenar todos esos datos.



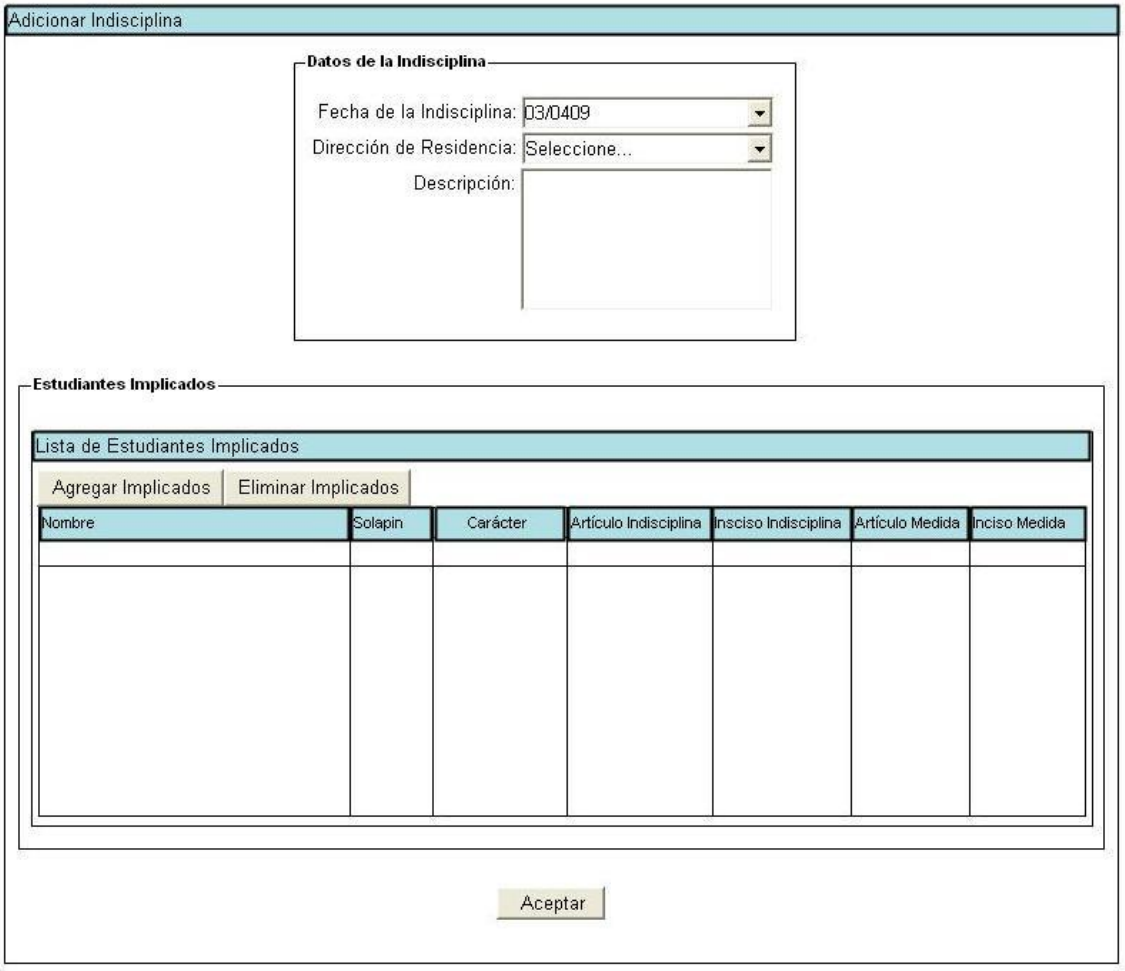
Prototipo. 2.2.3.3 Estructura FEU

Este prototipo se utilizará para asignarle responsabilidades a cada estudiante en la residencia para esto se mostrarán algunos datos del estudiante después de realizar una búsqueda y se seleccionar en el selector “Cargo” la opción que se le dará para su responsabilidad luego se dará clic en el botón “Aceptar” para registrar esa información.



Prototipo. 2.2.3.4 Realizar Evaluación

Este prototipo se utilizará para realizar las evaluaciones tanto de estudiantes como del apartamento de forma general para esto en el campo de texto “Apartamento” se escribirá el número del mismo y se da clic en el botón “Buscar” con el fin de que se realice la búsqueda de ese apartamento, si se obtiene algún resultado se mostrarán en una tabla todo los estudiantes que viven en ese local y de ellos algunos de sus datos, en la última columna de la tabla se encuentra por cada fila que tenga un selector donde se seleccionará la evaluación que se le asignara a cada uno de esos estudiantes, en el selector “Fecha” se escogerá la fecha en que se realizará la evaluación y en el de “Evaluación del Apartamento ” se seleccionará una evaluación para dicho local, luego se dará clic en el botón “Aceptar” para registrar toda la información.



Prototipo. 2.2.3.5 Adicionar Indisciplina

Este prototipo se utilizará para adicionar una nueva indisciplina donde en el selector fecha se seleccionará la fecha en que se realizó, en el selector “Dirección de Residencia” se seleccionará la dirección de residencia donde fue cometida la indisciplina y en el campo de texto “Descripción” se escribirá una breve descripción de la indisciplina cometida. Para agregar implicados a una indisciplina solo se tiene que dar clic en el botón “Agregar Implicados” y se adicionará dicho estudiante en la tabla “Lista de Estudiantes Implicados” con algunos datos, en la misma aparecerán varias columnas que se tienen en su interior selectores para seleccionar los datos referentes a la indisciplina. Para eliminar implicados del incidente solo se tiene que marcar en la tabla la fila que no es de interés y dar clic en el botón “Eliminar” para así quitarlo de la lista de todos los estudiantes que se encuentran implicados, después de todo se da en el botón “Aceptar” para guardar toda la información.

2.3 Descripción de los Componentes

- **FormPanel:** Los formularios actúan como contenedores para todos los elementos de entrada, toda esta información recolectada por él puede ser entregada a un agente procesador que es usualmente especificado en el atributo “url”.
- **TabPanel:** Como la mayoría de los contenedores, Ext.Tabpanel deriva de Ext.Panel, con lo que hereda todas sus funcionalidades básicas. Realmente se comporta como un panel en el que el contenido se puede dividir en varios tabs, donde cada uno de ellos es un panel con su correspondiente layout.
- **FieldSet:** Agrupa controles en un formulario. Esto permite a los autores agrupar temáticamente los controles de un formulario, haciéndolo más fácil de entender y rellenar para el usuario.
- **FieldLabel:** Puede ser asociado a un control y proveer información acerca del mismo. Como su nombre lo dice, asigna una etiqueta al control. Nota que cuando la etiqueta recibe el enfoque, lo pasa automáticamente a su control asociado.
- **Textarea:** Los elementos textarea son los campos que permiten introducir varias líneas de texto.
- **TextField:** Se utiliza para crear áreas para ver e introducir texto.
- **DateField:** Permite seleccionar una fecha dentro de un calendario que es mostrado cuando se hace click en este componente. En este componente nunca se podrá realizar una fecha incorrecta.
- **GridPanel:** Es un componente que permite únicamente listar información dentro de ellos en forma de tabla.
- **EditorGridPanel:** Es un componente que permite listar la información dentro de ellos en forma de tabla y además permite editar los valores
- Buttons:
- **Checkbox:** Es un componente que permite hacer selecciones múltiples en un conjunto de opciones
- **Combo:** El control ComboBox sustituye el obsoleto OptionMenu con un control potente que utiliza un TreeModel (generalmente un ListStore) que proporciona los elementos de la lista que se mostrarán. El ComboBox implementa la interfaz CellLayout, que proporciona diversos métodos para gestionar la visualización de los elementos de la lista.
- **Numberfield:** Este componente permite introducir

- **Radio:** Es un componente que permite seleccionar un elemento de un conjunto de opciones.

2.4 Estándares de Codificación

Para la implementación de las interfaces haciendo uso de la librería de java script Extjs se hace necesario para su mejor comprensión en desarrollos posteriores o de soporte la definición y utilización de algunos estándares de codificación. Donde se definen de varias formas:

El nombre de cada una de ellas será el nombre del módulo Ted seguido por un "_" y después el nombre que realizará la interfaz.

Ejemplo: Ted_realizarEvaluacion.js

Los componentes seguirán los siguientes estándares:

Componentes	**Prefijo**	**Ejemplo**
FormPanel	Frm_	Frm_nombre
TextField	TxF_	TxF_nombre
Textarea	TxA_	TxA_nombre
Checkbox	Chk_	Chk_funcion
Button	Bt_	Cmb_enviar
Combo	Cbo_	Cbo_nombre
Radio	Rad_	Rad_opcion
GridPanel	Gd_	Gd_nombre
EditorGridPanel	Gedi_	Gedi_nombre
Numberfield	Nb_	Nb_nombre
DateField	Dte_	Dte_nombre
FieldLabel	Lb_	Lb_nombre
TabPanel	Tb_	Tb_nombre

Para los atributos de cada componente se seguirán los estándares de codificaciones siguientes:

- En los id de cada componente se pondrá la palabra "id" y seguido una palabra que especifique el uso de ese componente.
- En los id de los store se pondrá la palabra "store" y seguido una palabra que especifique el uso del componente.
- En el root de los componentes que tengan ese atributo se pondrá una palabra que identifique el uso de ese componente.
- En la url se pondrá el nombre de la función que realizará esa interfaz.

2.5 Conclusión del Capítulo

En este capítulo se dio una panorámica de cómo la solución propuesta dio respuesta al problema, para ello se abordó temas, sin los cuales no hubiese sido posible la realización del sistema. Además se realizó una breve descripción de las interfaces de usuario que contiene la capa de Presentación, cumpliendo los requisitos de forma correcta. También se hizo una descripción de algunos de los componentes que nos brinda la librería Extjs así como la definición de estándares de codificación utilizados para el desarrollo de las interfaces del módulo “Trabajo Educativo” del Sistema de Gestión de Residencia.

Capítulo 3: SOLUCION

3.1 Introducción

En el presente capítulo se exponen la solución del problema planteado la cual se dará una breve descripción de la estructura de la capa de presentación y de las expresiones regulares utilizadas para las validaciones de los campos, además se muestran las pantallas finales implementadas para el módulo de Trabajo Educativo del Sistema.

3.2 Estructura de la capa de presentación

Teniendo en cuenta la arquitectura definida para el proyecto basada en el Framework de PHP Symfony, podemos identificar los subdirectorios que se usaran en la capa de presentación y que contendrá la solución implementada.

En la figura 3.2.1 se muestra dicha estructura.

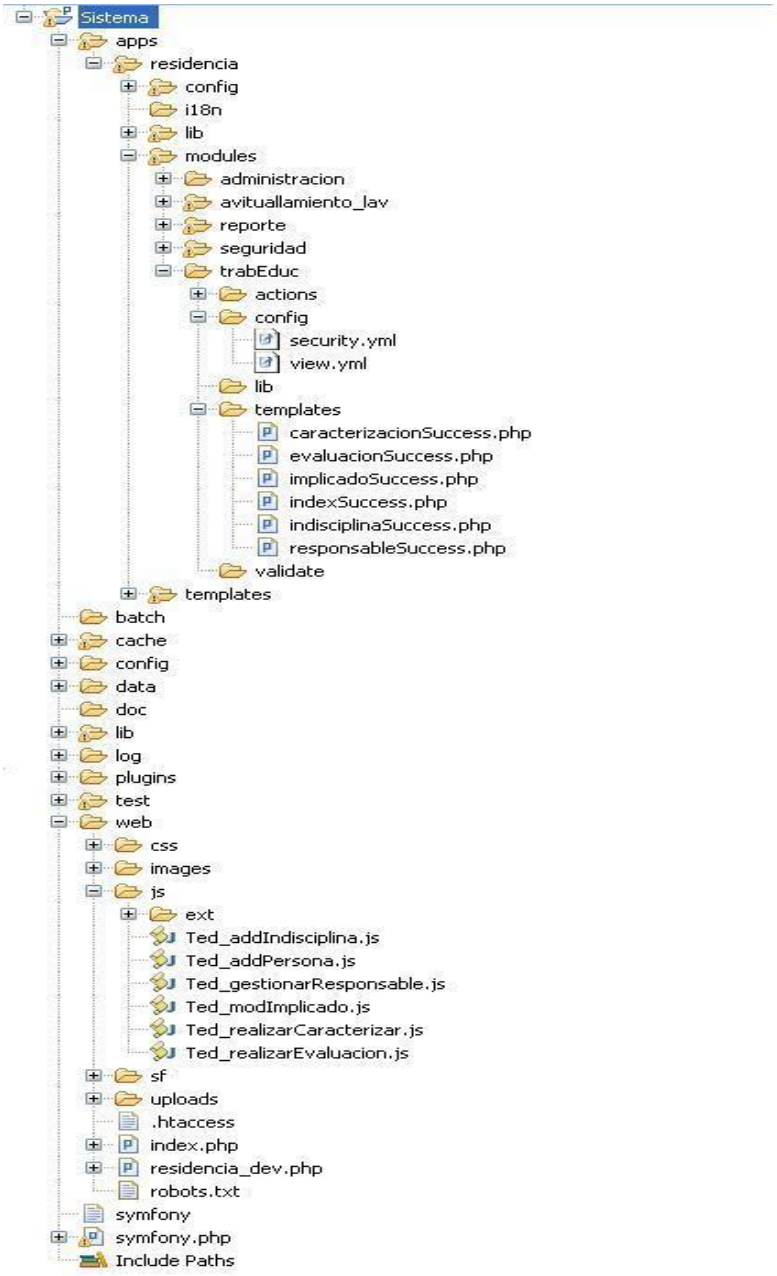


Fig. 3.2.1 Estructura de la capa de presentación

Todos los módulos de una aplicación constan de cinco carpetas estas son: *actions*, *config*, *lib*, *templates* y *valídate* donde dentro de ellas las más importantes para la capa de presentación son la carpeta “config” y “templates”. En la *templates* se encuentran todas las plantillas correspondientes a las acciones del módulo, es donde se definen los ficheros “Success.php” estos no son más que plantillas en blanco, ya que con la utilización del frameword de java script Ext-js no es necesario crear las vistas en ese fichero y solo se utiliza como base para mostrar la pantalla que es ejecutada, obteniéndose por cada interfaz una plantilla. La configuración de las vistas se realiza mediante un fichero “*yml*” que se encuentra en la carpeta *config,* donde la misma puede contener archivos de configuración adicionales con parámetros exclusivos del módulo, dentro de los cuales encontramos el *view.yml*. Cada módulo contiene un *view.yml* que define las opciones de su propia vista, de esta forma, es posible definir en un único archivo las opciones de la vista para todo el módulo entero y las opciones para cada vista. Además en la carpeta Web, se encuentran los archivos que se pueden acceder de forma pública.

La siguiente es una estructura típica del directorio web:

web/

css/

images/

js/

uploads/

La carpeta ***css*** contiene los archivos de hojas de estilo creados con CSS (archivos con extensión .css) propios del framework de php y también los estilos de la librería de java script Extjs. En la de ***images*** se encuentran las imágenes utilizadas en la construcción del sitio con formato .jpg, .png o .gif y en la de ***js*** se encuentran todos los ficheros .js utilizados en la creación de las interfaces del sistema.

3.3 Validación por Expresiones Regulares

Las pruebas de validación de las interfaces de usuario básicamente chequean si los tipos y rangos de los valores de un campo son correctos. Por ejemplo, si un campo numérico es realmente un número o si una fecha es realmente una fecha válida, etc. La validación de datos es cuando necesitamos chequear si el valor dado es válido entre, digamos, una lista de posibles elecciones, quizás requiriendo una búsqueda en una base de datos. **(Vegas, 2006)**

A continuación se muestran las diferentes expresiones regulares utilizadas para lograr las validaciones de los campos de texto de cada interfaz de usuario del modulo “Trabajo Educativo” y lograr con esto la calidad de las interfaces.

/[0-9 a-z A-Z]/: Esta expresión regular valida que en los campos de textos solo se puedan escribir letras ya sean mayúsculas o minúsculas y números.

/[0-9]/: Esta expresión regular se utiliza para validar que en campos de texto solo se puedan escribir números.

/[0-9 a-z A-Z .]/: Esta expresión regular se utiliza para validar que en los campos de texto solo se puedan escribir números, letras ya sea minúsculas o mayúsculas y un punto.

/[0-9 .]/: Esta expresión regular se utiliza para validar que en los campos de texto solo se puedan escribir números y un punto.

/[0-9 a-z A-Z . -]/: Esta expresión regular se utiliza para validar que en los campos de texto solo se puedan escribir números, letras ya sea minúsculas o mayúsculas, puntos y símbolo de menos.

/[a-z]/: Esta expresión regular se utiliza para validar que en los campos de texto solo se puedan escribir letras minúsculas.

3.4 Implementación de la Solución

En este epígrafe se mostrarán todas las pantallas implementadas para darle solución al problema planteado, así como el código fuente de una de las interfaces seleccionada como la más general, los demás códigos se podrán encontrar en los anexos.

3.4.1 Interfaz Realizar Caracterización

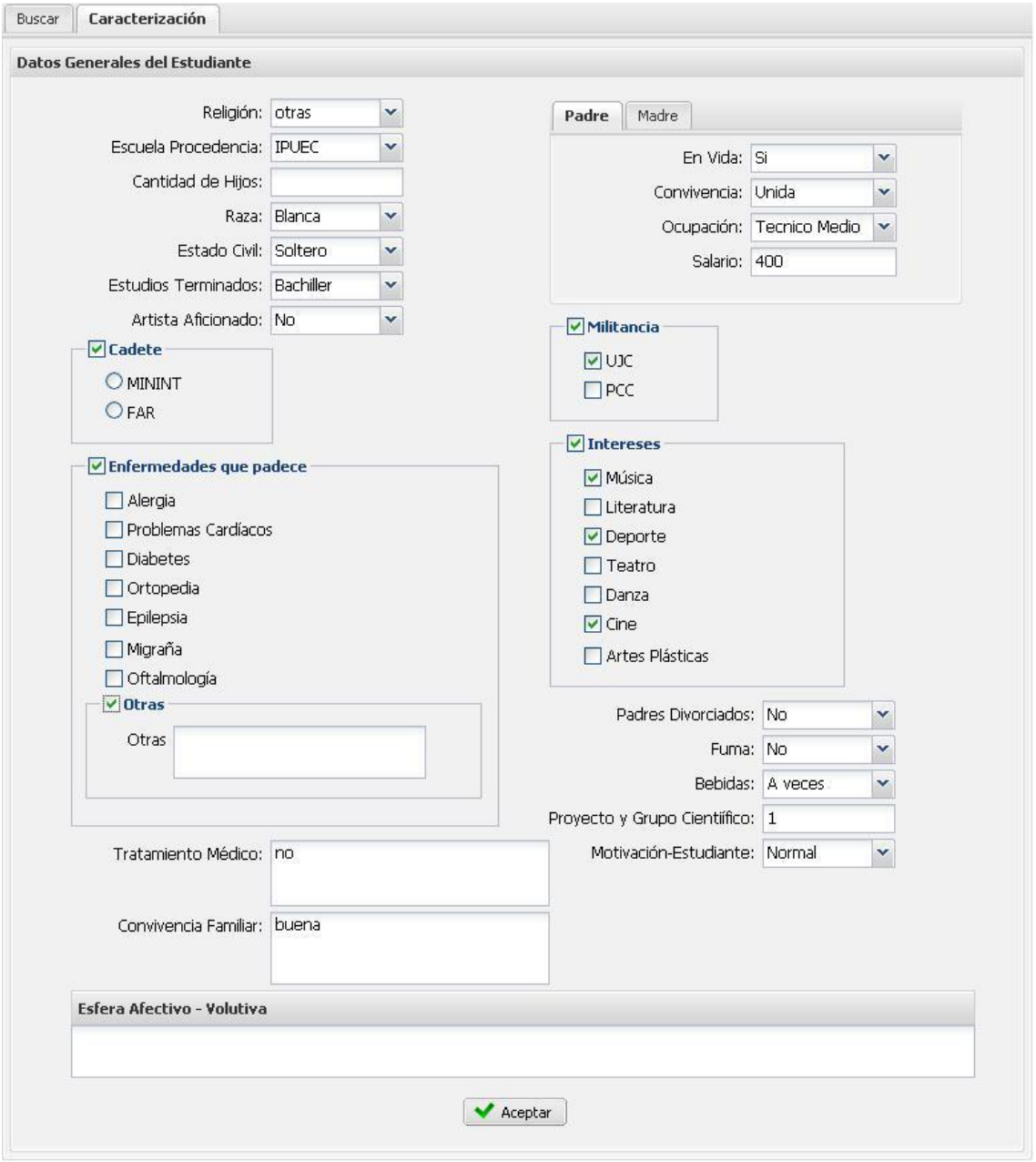


Fig 3.4.1Interfaz Realizar Caracterización

3.4.1.1 Código Fuente

```
Ext.onReady(function(){
    Ext.QuickTips.init();
    Ext.MessageBox.buttonText.yes = "Si";
    Ext.MessageBox.buttonText.cancel = "Cancelar";
    var tabs = new Ext.TabPanel({
        renderTo: document.body,
        title: 'Buscar',
        id: 'idtab',
        activeTab: 0,
        border: false,
        frame:true,
        style: ' margin:0 auto',
        width:775,
        autoHeigth: true,
        listeners: {
            'tabchange': function(tabpanel, tab){
                tab.doLayout();
            }
        },
        defaults: {
            frame: true,
            autoHeigth: true
        },
        items: [{
            title: 'Buscar',
            name: 'TbP_primero',
            id:'idbuscar',
            autoWidth: true,
            autoHeight: true,
            bodyStyle:'padding:10px 4px 10px 30px',
            items: [busqueda]
        }, {
            title: 'Caracterizaci&oacute;n',
            name: 'TbP_segundo',
            autoWidth: true,
            id:'idCaract',
            autoHeight: true,
            disabled:true,
            items: [top]
        }]
    });
});
var top = new Ext.form.FormPanel({
    id: 'idtop',
    title: 'Datos Generales del Estudiante',
    bodyStyle: 'padding:10px 4px 10px 40px',

    autoHeight: true,
    buttonAlign: 'center',
    frame: true,
    autoWidth: true,
    items: [{
        title: '',
        buttonAlign: 'center',
        defaults: {
            columnWidth: 0.5
        },
```

```
        width: 720,
        layout: 'column',
        //**********************************Aqui el Panel de
dos*********************************************
        items: [{
            title: '',
            buttonAlign: 'center',
            layout: 'form',
            labelAlign: 'right',
            labelWidth: 145,

            //**********************************Aqui el la parte
Izquierda****************
            items: [{
                xtype: 'combo',
                fieldLabel: 'Religi&oacute;n',
                name: 'Cbo_religion',
                id: 'idReligion',
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                store: new Ext.data.SimpleStore({
                    data: [['Catolico', 'Catolico'], ['Cristiano',
'Cristiano'], ['Afro', 'Afro'], ['otras', 'otras'],['ninguna',
'ninguna']],
                    fields: ['id', 'texto']
                }),
                valueField: 'id',
                displayField: 'texto',
                hiddenName: 'Cbo_religion',
                triggerAction: 'all',
                mode: 'local',
                emptyText: 'Seleccione...',
                readOnly: true,
                width: 100
            }, {
                xtype: 'combo',
                fieldLabel: 'Escuela Procedencia',
                name: 'Cbo_procedencia',
                id:'idEscuela',
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                store: new Ext.data.SimpleStore({
                    data: [['Camilitos', 'Camilitos'], ['IPUEC',
'IPUEC'], ['IPVC', 'IPVCE'], ['Politecnico', 'Politecnico']],
                    fields: ['id', 'texto']
                }),
                valueField: 'id',
                allowBlank: false,
                displayField: 'texto',
                hiddenName: 'Cbo_procedencia',
                triggerAction: 'all',
                mode: 'local',
                emptyText: 'Seleccione...',
                readOnly: true,
                width: 100
            }, {
                xtype: 'numberfield',
                fieldLabel: 'Cantidad de Hijos',
                id: 'idcanthijos',
                maskRe : /[0-9]/,
                name: 'Nb_canthijos',
```

```
                width: 100,
                maxLength: 2
            }, {
                xtype: 'combo',
                fieldLabel: 'Raza',
                name: 'Cbo_raza',
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                id:'idRaza',
                store: new Ext.data.SimpleStore({
                    data: [['B', 'Blanca'], ['N', 'Negra'], ['M',
'Mestiza']],
                    fields: ['id', 'texto']
                }),
                valueField: 'id',
                displayField: 'texto',
                hiddenName: 'Cbo_raza',
                triggerAction: 'all',
                mode: 'local',
                emptyText: 'Seleccione...',
                readOnly: true,
                width: 100
            }, {
                xtype: 'combo',
                fieldLabel: 'Estado Civil',
                name: 'Cbo_estcivil',
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                store: new Ext.data.SimpleStore({
                    data: [['Casado', 'Casado'], ['Soltero',
'Soltero']],
                    fields: ['id', 'texto']
                }),
                valueField: 'id',
                displayField: 'texto',
                hiddenName: 'Cbo_estcivil',
                triggerAction: 'all',
                mode: 'local',
                emptyText: 'Seleccione...',
                readOnly: true,
                width: 100
            }, {
                xtype: 'combo',
                fieldLabel: 'Estudios Terminados',
                name: 'Cbo_estterminados',
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                store: new Ext.data.SimpleStore({
                    data: [['Bachiller', 'Bachiller'], ['TecMedio',
'Tec. Medio']],
                    fields: ['id', 'texto']
                }),
                valueField: 'id',
                displayField: 'texto',
                hiddenName: 'Cbo_estterminados',
                triggerAction: 'all',
                mode: 'local',
                emptyText: 'Seleccione...',
                readOnly: true,
                width: 100
            }, {
```

```
                xtype: 'combo',
                fieldLabel: 'Artista Aficionado',
                name: 'Cbo_artista',
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                store: new Ext.data.SimpleStore({
                    data: [['S', 'Si'], ['N', 'No']],
                    fields: ['id', 'texto']
                }),
                valueField: 'id',
                displayField: 'texto',
                hiddenName: 'Cbo_artista',
                triggerAction: 'all',
                mode: 'local',
                emptyText: 'Seleccione...',
                readOnly: true,
                width: 100
            }, {}, {
                xtype: 'fieldset',
                checkboxToggle: true,
                collapsed: true,
                title: 'Cadete',
                labelWidth: 10,
                width: 150,
                autoHeight: true,
                height: 100,
                items: [{
                    layout: 'form',
                    heigth: 50,
                    border: false,
                    items: [{
                        labelSeparator: ' ',
                        xtype: 'radio',
                        width: 70,
                        boxLabel: 'MININT',
                        name: 'Rad_radio1',
                        id: 'idmin',
                        inputValue: 'MININT'

                    }, {
                        labelSeparator: ' ',
                        xtype: 'radio',
                        width: 70,
                        boxLabel: 'FAR',
                        id: 'idfar',
                        name: 'Rad_radio1',
                        inputValue: 'FAR'
                    }]
                }]
            }, {
                xtype: 'fieldset',
                checkboxToggle: true,
                collapsed: true,
                title: 'Enfermedades que padece',
                labelWidth: 10,
                width: 320,
                autoHeight: true,
                id: 'idenfermedad',
                height: 100,

                items: [{
```

```
                    layout: 'form',
                    heigth: 50,
                    border: false,
                    items: [{
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Alergia',
                        id: 'idalergia',
                        inputValue: 'Alergia'


                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 180,
                        boxLabel: 'Problemas Card&iacute;acos',
                        id: 'idprCorazon',
                        inputValue: 'Prob. Cardiacos'

                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Diabetes',
                        id: 'iddiabetis',
                        inputValue: 'Diabetis'

                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Ortopedia',
                        id: 'idortopedia',
                        inputValue: 'Ortopedia'
                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Epilepsia',
                        id: 'idepilecia',
                        inputValue: 'Epilepsia'
                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Migra&ntilde;a',
                        id: 'idmigranna',
                        inputValue: 'Migraña'
                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Oftalmolog&iacute;a',
                        id: 'idofta',
                        inputValue: 'Message Contents'
                    }, {

                        xtype: 'fieldset',
                        checkboxToggle: true,
                        title: 'Otras',
```

```
                        autoHeight: true,
                        defaults: {
                            width: 80
                        },
                        defaultType: 'textfield',
                        collapsed: true,
                        labelWidth: 50,
                        items: [{
                            labelSeparator: ' ',
                            xtype: 'textarea',
                            id: 'idmas',
                            fieldLabel: 'Otras',
                            width: 190,
                            height: 40
                        }]
                    }]
                }]


            }, {
                xtype: 'textarea',
                fieldLabel: 'Tratamiento M&eacute;dico',
                labelAlign: 'center',
                maskRe : /[0-9 a-z A-Z .]/,
                width: 210,
                height: 50,
                name: 'TxA_tratmedico',
                allowBlank: true,
                id: 'idtratmedico'
            }, {}, {
                xtype: 'textarea',
                fieldLabel: 'Convivencia Familiar',
                width: 210,
                height: 55,
                maskRe : /[0-9 a-z A-Z .]/,
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                name: 'TxA_convivecon',
                id: 'idconvivecia'

            }]

        }, {
            title: '',
            width: 10
        }, //**********************************Aqui el Panel de
Derecho***********************************************
        {
            layout: 'form',
            autoHeigth: true,
            items: [{
                title: '',
                autoHeigth: true,
                layout: 'form',
                labelAlign: 'right',
                labelWidth: 155,
                width: 480,
                items: [new Ext.TabPanel({
                    activeTab: 0,
                    border: false,
                    width: 310,
```

```
                    height: 155,
                    listeners: {
                        'tabchange': function(tabpanel, tab){
                            tab.doLayout();
                        }
                    },
                    defaults: {
                        frame: true,
                        autoHeigth: true
                    },
                    items: [{
                        title: 'Padre',
                        layout: 'form',
                        labelWidth: 140,
                        items: [{
                            xtype: 'combo',
                            allowBlank: false,
                            blankText: 'Este campo es obligatorio',
                            fieldLabel: 'En Vida',
                            name: 'Cbo_pvivo',
                            store: new Ext.data.SimpleStore({
                                data: [['S', 'Si'], ['N', 'No']],
                                fields: ['id', 'texto']
                            }),
                            valueField: 'id',
                            displayField: 'texto',
                            hiddenName: 'Cbo_pvivo',
                            triggerAction: 'all',
                            mode: 'local',
                            emptyText: 'Seleccione...',
                            readOnly: true,
                            width: 110
                        }, {
                            xtype: 'combo',
                            allowBlank: true,

                            fieldLabel: 'Convivencia',
                            name: 'Cbo_pcvive',
                            store: new Ext.data.SimpleStore({
                                data: [['Unida', 'Unida'], ['No
Unida', 'No Unida']],

                                fields: ['id', 'texto']
                            }),
                            valueField: 'id',
                            displayField: 'texto',
                            hiddenName: 'Cbo_pcvive',
                            triggerAction: 'all',
                            mode: 'local',
                            emptyText: 'Seleccione...',
                            readOnly: true,
                            width: 110
                        }, {
                            xtype: 'combo',
                            allowBlank: true,

                            fieldLabel: 'Ocupaci&oacute;n',
                            name: 'Cbo_pocupacion',
                            store: new Ext.data.SimpleStore({
                                data: [['Obrero', 'Obrero'],
['Campesino', 'Campesino'], ['Profesional', 'Profesional'], ['Técnico
Medio', 'Técnico Medio'], ['Ama de casa', 'Ama de casa'], ['Cuenta
```

```
propia', 'Cuenta propia'], ['Cuadro', 'Cuadro'], ['Jubilado',
'Jubilado'], ['Militar', 'Militar']],
                                fields: ['id', 'texto']
                            }),
                            valueField: 'id',
                            displayField: 'texto',
                            hiddenName: 'Cbo_pocupacion',
                            triggerAction: 'all',
                            mode: 'local',
                            emptyText: 'Seleccione...',
                            readOnly: true,
                            width: 110
                        }, {
                            xtype: 'numberfield',
                            allowBlank: true,
                             id: 'idpsalario',
                            maskRe : /[0-9 .]/,
                            nanText:'N&uacute;mero inv&aacute;lido',
                            fieldLabel: 'Salario',
                            width: 110,
                            name: 'Nb_psalario'
                        }]
                    }, {
                        title: 'Madre',
                        autoHeigth: true,
                        labelWidth: 140,
                        layout: 'form',
                        items: [{
                            xtype: 'combo',
                            allowBlank: false,
                            blankText: 'Este campo es obligatorio',
                            fieldLabel: 'En Vida',
                            name: 'Cbo_mvivo',
                            store: new Ext.data.SimpleStore({
                                data: [['S', 'Si'], ['N', 'No']],
                                fields: ['id', 'texto']
                            }),
                            valueField: 'id',
                            displayField: 'texto',
                            hiddenName: 'Cbo_mvivo',
                            triggerAction: 'all',
                            mode: 'local',
                            emptyText: 'Seleccione...',
                            readOnly: true,
                            width: 110
                        }, {
                            xtype: 'combo',
                            allowBlank: true,
                            fieldLabel: 'Convivencia',
                            name: 'Cbo_mconvive',
                            store: new Ext.data.SimpleStore({
                                data: [['Unida', 'Unida'], ['No
unida', 'No Unida']],
                                fields: ['id', 'texto']
                            }),
                            valueField: 'id',
                            displayField: 'texto',
                            hiddenName: 'Cbo_mconvive',
                            triggerAction: 'all',
                            mode: 'local',
                            emptyText: 'Seleccione...',
```

```
                            readOnly: true,
                            width: 110
                        }, {
                            xtype: 'combo',
                            allowBlank: true,
                            fieldLabel: 'Ocupaci&oacute;n',
                            name: 'Cbo_mocupacion',
                            store: new Ext.data.SimpleStore({
                               data: [['Obrero', 'Obrero'],
['Campesino', 'Campesino'], ['Profesional', 'Profesional'], ['Técnico
Medio', 'Técnico Medio'], ['Ama de casa', 'Ama de casa'], ['Cuenta
propia', 'Cuenta propia'], ['Cuadro', 'Cuadro'], ['Jubilado',
'Jubilado']],
                                fields: ['id', 'texto']
                            }),
                            valueField: 'id',
                            displayField: 'texto',
                            hiddenName: 'Cbo_mocupacion',
                            triggerAction: 'all',
                            mode: 'local',
                            emptyText: 'Seleccione...',
                            readOnly: true,
                            width: 110
                        }, {
                            xtype: 'numberfield',
                            allowBlank: true,
                            fieldLabel: 'Salario',
                            id: 'idmsalario',
                            width: 110,
                            maskRe : /[0-9 .]/,
                            nanText:'N&uacute;mero inv&aacute;lido',
                            name: 'Nb_msalario'
                        }]
                    }]
                }), {
                    title: '',
                    frame: false,
                    height: 10
                }, {

                    xtype: 'fieldset',
                    checkboxToggle: true,
                    collapsed: true,
                    title: 'Militancia',
                    labelWidth: 10,
                    width: 125,
                    autoHeight: true,
                    height: 100,
                    items: [{
                        layout: 'form',
                        heigth: 50,
                        border: false,
                        items: [{
                            labelSeparator: ' ',
                            xtype: 'checkbox',
                            width: 50,
                            boxLabel: 'UJC',
                            id: 'idujc',
                            inputValue: 'Si'

                        }, {
```

```
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 50,
                        boxLabel: 'PCC',
                        id: 'idpcc',
                        inputValue: 'Si'

                    }]
                }]
            }, {
                xtype: 'fieldset',
                checkboxToggle: true,
                collapsed: true,
                title: 'Intereses',
                labelWidth: 10,
                width: 220,
                autoHeight: true,
                height: 100,
                items: [{
                    layout: 'form',
                    heigth: 45,
                    border: false,
                    items: [{
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'M&uacute;sica',
                        id: 'idmusica',
                        inputValue: 'Música'

                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Literatura',
                        id: 'idlitera',
                        inputValue: 'Literatura'

                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Deporte',
                        id: 'iddeporte',
                        inputValue: 'Deporte'
                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Teatro',
                        id: 'idteatro',
                        inputValue: 'Teatro'
                    }, {
                        labelSeparator: ' ',
                        xtype: 'checkbox',
                        width: 150,
                        boxLabel: 'Danza',
                        id: 'iddansa',
                        inputValue: 'Dansa'
                    }, {
                        labelSeparator: ' ',
```

```
                            xtype: 'checkbox',
                            width: 150,
                            boxLabel: 'Cine',
                            id: 'idcine',
                            inputValue: 'Cine'
                        }, {
                            labelSeparator: ' ',
                            xtype: 'checkbox',
                            width: 150,
                            boxLabel: 'Artes Pl&aacute;sticas',
                            id: 'idaPlast',
                            inputValue: 'Artes Plásticas'
                        }]
                    }]
                }, {
                    xtype: 'combo',
                    labelWidth: 200,
                    fieldLabel: 'Padres Divorciados',
                    name: 'Cbo_pdivorciados',
                    allowBlank: false,
                    blankText: 'Este campo es obligatorio',
                    store: new Ext.data.SimpleStore({
                        data: [['S', 'Si'], ['N', 'No']],
                        fields: ['id', 'texto']
                    }),
                    valueField: 'id',
                    displayField: 'texto',
                    hiddenName: 'Cbo_pdivorciados',
                    triggerAction: 'all',
                    mode: 'local',
                    emptyText: 'Seleccione...',
                    readOnly: true,
                    width: 100
                }, {
                    xtype: 'combo',
                    labelWidth: 80,
                    fieldLabel: 'Fuma',
                    labelWidth: 100,
                    name: 'Cbo_fuma',
                    allowBlank: false,
                    blankText: 'Este campo es obligatorio',
                    store: new Ext.data.SimpleStore({
                        data: [['S', 'Si'], ['N', 'No'], ['A veces',
'A veces']],
                        fields: ['id', 'texto']
                    }),
                    valueField: 'id',
                    displayField: 'texto',
                    hiddenName: 'Cbo_fuma',
                    triggerAction: 'all',
                    mode: 'local',
                    emptyText: 'Seleccione...',
                    readOnly: true,
                    width: 100
                }, {
                    xtype: 'combo',
                    labelWidth: 80,
                    fieldLabel: 'Bebidas',
                    name: 'Cbo_bebidas',
                    allowBlank: false,
                    blankText: 'Este campo es obligatorio',
```

```
                    store: new Ext.data.SimpleStore({
                        data: [['S', 'Si'], ['N', 'No'], ['A veces',
'A veces']],
                        fields: ['id', 'texto']
                    }),
                    valueField: 'id',
                    displayField: 'texto',
                    hiddenName: 'Cbo_bebidas',
                    triggerAction: 'all',
                    mode: 'local',
                    emptyText: 'Seleccione...',
                    readOnly: true,
                    width: 100
                }, {
                    xtype: 'textfield',
                    labelWidth: 80,
                    maskRe : /[0-9 a-z A-Z .]/,
                    fieldLabel: 'Proyecto y Grupo Cienti&iacute;fico',
                    id: 'idproyectoygrupo',
                    name: 'TxF_proyectoygrupo',
                    allowBlank: true,
                    width: 100
                }, {
                    xtype: 'combo',
                    labelWidth: 80,
                    fieldLabel: 'Motivaci&oacute;n-Estudiante',
                    name: 'Cbo_motivo',
                    allowBlank: false,
                    blankText: 'Este campo es obligatorio',
                    store: new Ext.data.SimpleStore({
                        data: [['Mucho', 'Mucho'], ['Normal',
'Normal'], ['Regula', 'Regular'], ['Ninguna', 'Ninguna']],
                        fields: ['id', 'texto']
                    }),
                    valueField: 'id',
                    displayField: 'texto',
                    hiddenName: 'Cbo_motivo',
                    triggerAction: 'all',
                    mode: 'local',
                    emptyText: 'Seleccione...',
                    readOnly: true,
                    width: 100
                }

                ]
            }]
        }]
    }, {
        title: 'Esfera  Afectivo - Volutiva',
        width: 680,
        autoHeigth: true,
        items: [{
            xtype: 'textarea',
            width: 680,
            maskRe : /[0-9 a-z A-Z . -]/,
            allowBlank: true,
            blankText: 'Este campo es obligatorio',
            id: 'idafetVolutiva',
            height: 40
        }]
    }, {
```

```
        tite: '',
        height: 1,
        frame: false,
        buttonAlign:'center',
        style: 'margin:0 auto',
        bodyStyle: 'padding:15px 15px 25px 295px',
        items: [{
            items: [{
                xtype: 'button',
                text: 'Aceptar',
                 icon : '../../images/ok.ico',
                 cls:"x-btn-text-icon",

                handler: function(){
                    top.form.submit({
                        url: 'guardarCaraterizacion',
                        waitMsg: 'Guardando...',
                        waitTitle: 'Espere por favor!!!',
                                  params: {
                                        facultadEst:
Ext.getCmp('facultadEstudiante').getValue(),
                                        solapinEst:
Ext.getCmp('solapinEstudiante').getValue()
                                  },
                        success: function(){
                            Ext.Msg.show({
                                msg: 'Se guardaron los datos.',
                                title: 'Informaci&oacute;n',
                                icon: Ext.Msg.INFO,
                                buttons: Ext.Msg.OK
                            });
                            Ext.getCmp('idCaract').setDisabled(true);
                            busqueda.form.reset();
                            top.form.reset();
                            Ext.getDom('foto').setAttribute('src',
'../../images/individuo.JPG');
                            Ext.getCmp('idtab').activate('a');
                        },
                        failure: function(){
                           if(!top.form.isValid()){
                            Ext.Msg.show({
                                msg: 'Existen campos incorrectos',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                        }
                        }

                    });
                }
            }]
        }]
    }]
});
```

3.4.2 Interfaz Buscar Estudiantes

Fig. 3.4.2.1 Interfaz Buscar Estudiantes

3.4.3 Interfaz Gestionar Implicados

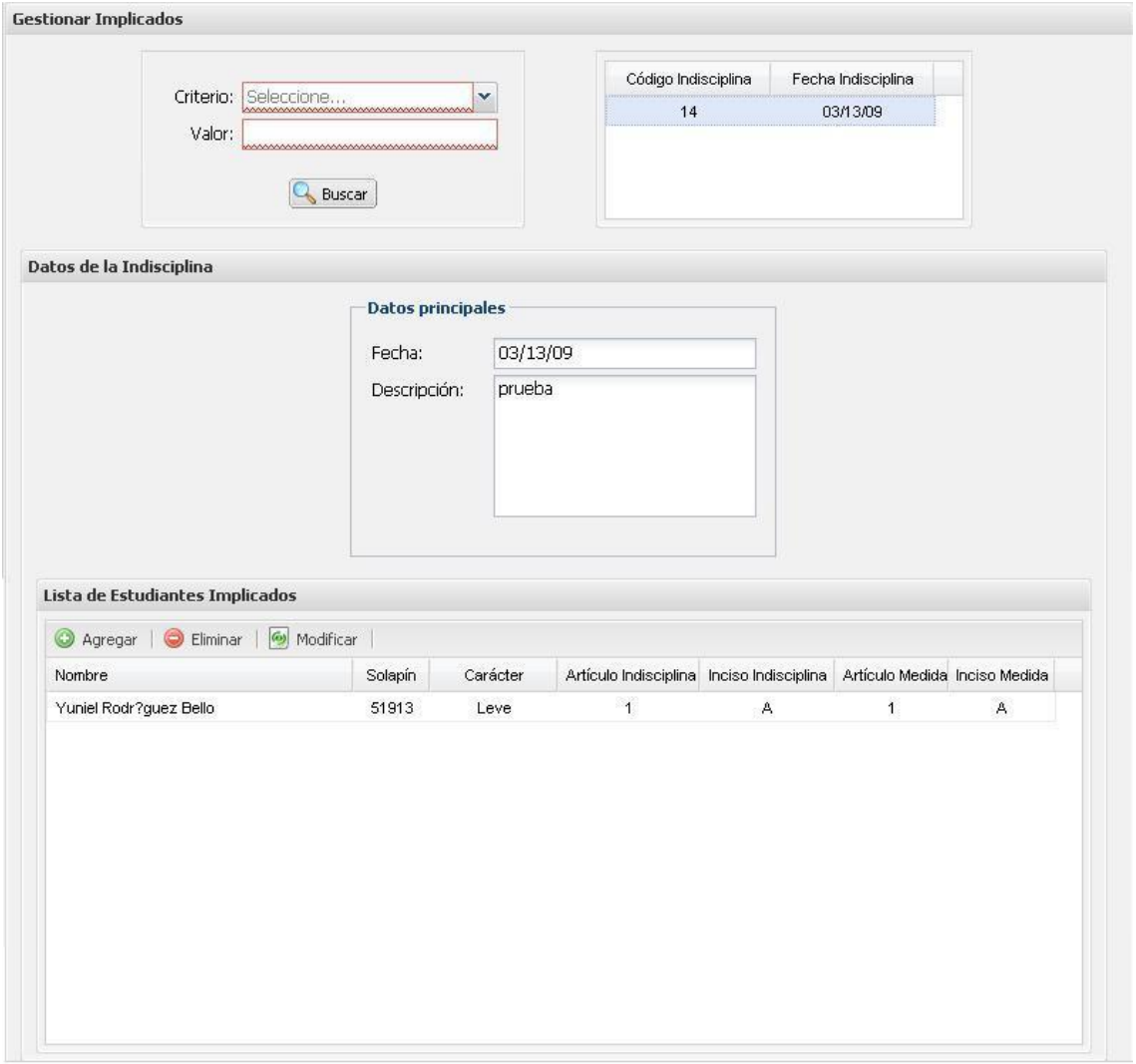


Fig. 3.4.3.1Interfaz Gestionar Implicados

3.4.4 Interfaz Adicionar Implicados

Fig. 3.4.4.1Interfaz Adicionar Implicados

3.4.5 Interfaz Modificar Implicados

Fig. 3.4.5.1 Interfaz Modificar Implicados

3.4.6 Interfaz Estructura FEU

Fig. 3.4.6.1 Interfaz Estructura FEU

3.4.7 Interfaz Realizar Evaluación

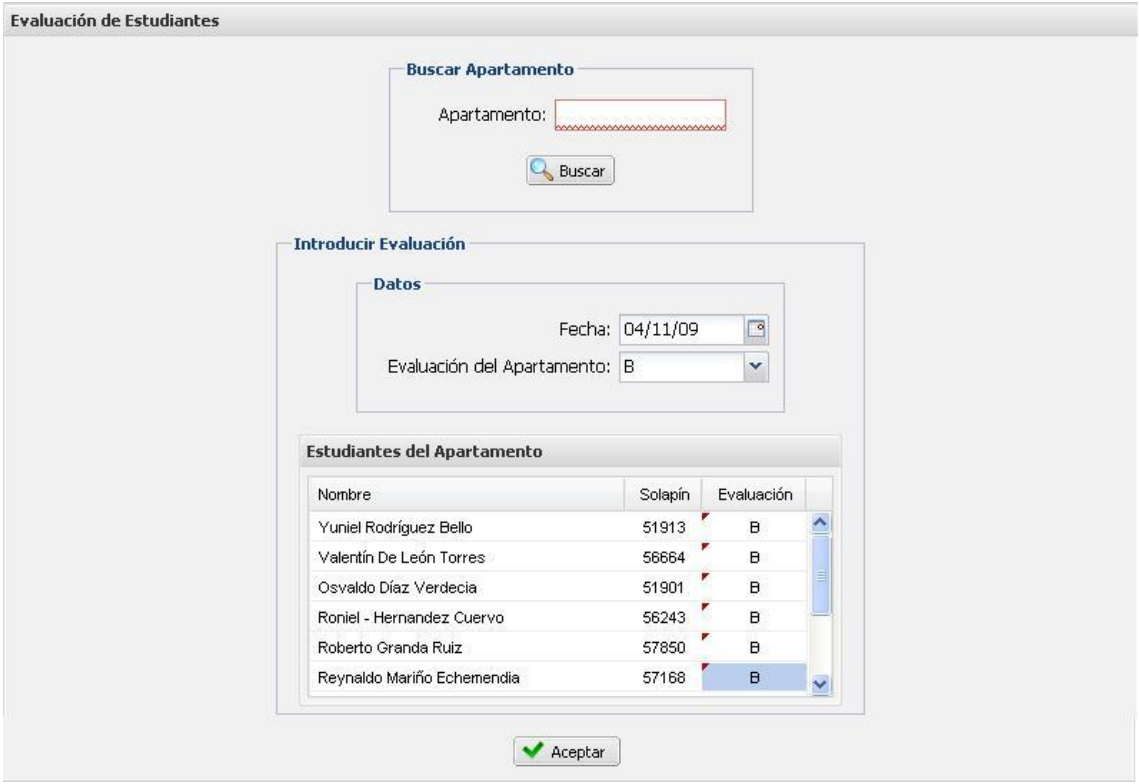


Fig. 3.4.7.1 Interfaz Realizar Evaluación

3.4.8 Interfaz Adicionar Indisciplina

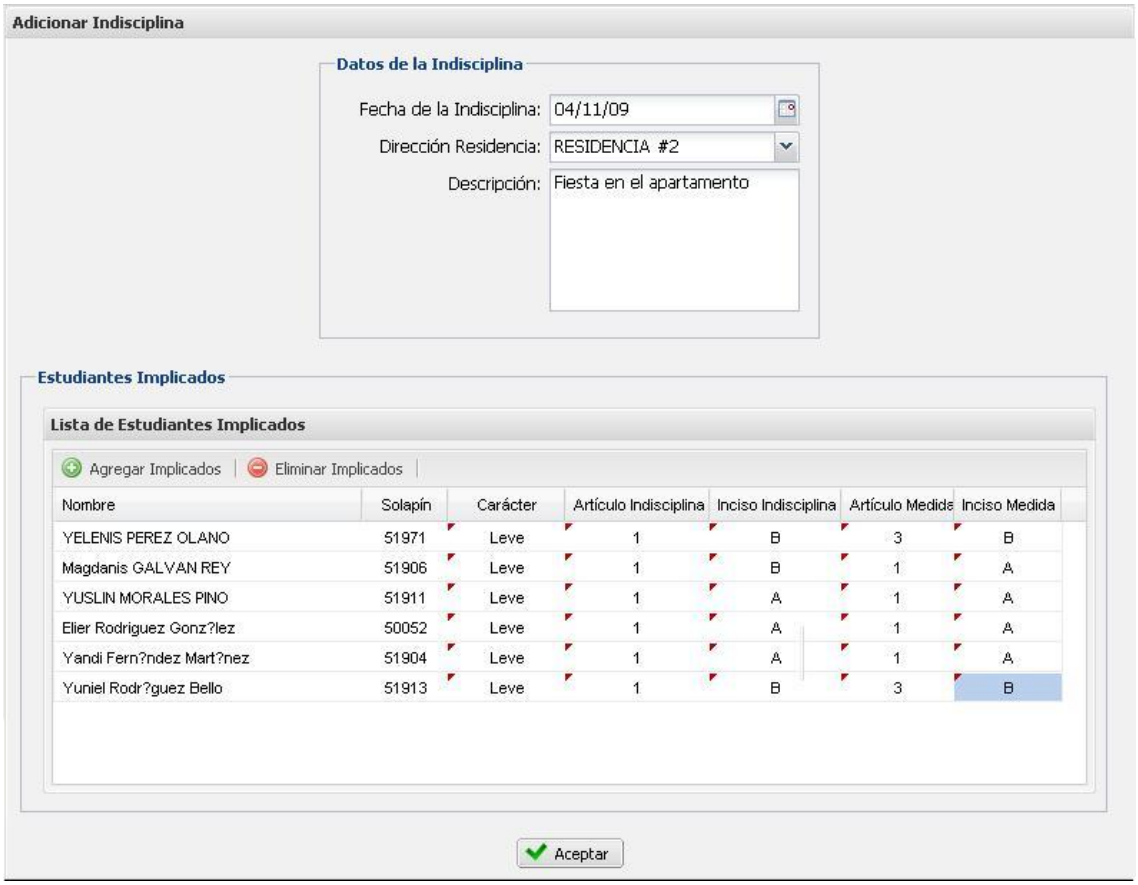


Fig. 3.4.8.1 Interfaz Adicionar Indisciplina

3.5 Concluciones del Capítulo

En este capítulo se analizaron las interfaces de usuario relacionadas con cada uno de los casos de usos que conforman el modulo “Trabajo Educativo” del Sistema de gestión de la residencia, también se analizaron las diferentes expresiones regulares utilizadas para las validaciones de los campos de textos utilizados. Mediante estos análisis se puede certificar cada una de estas interfaces que, de forma amigable, funcional y rápida, se integra con la capa de presentación según el Framework Symfony y Extjs donde a través de toda esta vinculación se puede garantizar la culminación de todo un proceso de desarrollo.

CONCLUSIONES GENERALES

- Se identificaron los patrones, estándares y pautas necesarios para garantizar la calidad del desarrollo.
- Se implementaron todas las interfaces de acuerdo a los prototipos validados por el cliente.
- Se desarrollo e implementó la capa de presentación para el módulo de trabajo educativo haciendo uso del Framework Extjs e integrándolo con la arquitectura definida para el proyecto.
- Se implementó utilizando el estándar de codificación definido para el uso del Framework de interfaz
- Se diseñaron los prototipos de interfaz de acuerdo a los requerimientos.

RECOMENDACIONES

Los objetivos generales de este trabajo han sido logrados, pero a lo largo de su desarrollo, han ido surgiendo ideas que podrían considerarse en un futuro, para lo cual se recomienda:

- Migrar a una versión superior de Extjs
- La implementación de la capa de Presentación, para los módulos restantes del Sistema de gestión de Residencia, teniendo en cuenta el estándar de codificación y la estructura de la capa de Presentación definida

REFERENCIAS BIBLIOGRAFICAS

Alertbox, Jakob Nielsen's. 2000. Useit. [En línea] 16 de abril de 2000. http://www.useit.com/alertbox/20000416.html.

—. 2004. Useit. [En línea] 27 de septiembre de 2004. http://www.useit.com/alertbox/20040927.html.

Best Practices for Form Design. Wroblewski, Luke. 2008. 2008.

Cabrera, Javier. 2004. Webexperto. [En línea] 23 de abril de 2004. http://www.webexperto.com/articulos/art/196/formularios-usables/.

Hammer. 1983. 1983.

Junoy, Josep M. 2004. alzado. [En línea] 22 de diciembre de 2004. http://www.alzado.org/articulo.php?id_art=388.

Lewis-Rieman. 1993. 1993.

Manchón, Eduardo. 2003. alzado. [En línea] 9 de febrero de 2003. http://www.alzado.org/articulo.php?id_art=57.

—. 2003. alzado. [En línea] 9 de febrero de 2003. http://www.alzado.org/articulo.php?id_art=50.

Martín, César. 2009. Desarrollo web. [En línea] 2009. http://www.desarrolloweb.com/articulos/224.php.

Myers. 1996. 1996.

Padilla, Reinier Matos. 2008. Maestros del Web. [En línea] 25 de enero de 2008. http://www.maestrosdelweb.com/editorial/usabilidad-al-disenar-formularios-de-contacto/.

Reimer. 2005. 2005.

Shneiderman. 2005. *Diseño de interfaces de usuario.* s.l. : Prentice Hall, 2005.

Starling, Andrew. 2001. ECommerce-Guide. [En línea] 11 de diciembre de 2001. http://www.ecommerce-guide.com/solutions/building/article.php/10362_938071.

Stephanidis. 2001. 2001.

Team, developer. 2008. Spket IDE. [En línea] 2008. http://www.spket.com.

Copeland,Bart.2009. ActiveState. [En línea] 2009. http://www.activestate.com/komodo/.

Team, Developers. 2009. Qooxdoo the new era of web development. [En línea] 2009. http://qooxdoo.org.

Team, Dojo Development. 2009. Dojo. [En línea] 2009. http://www.dojotoolkit.org.

Team, Ext Management. 2009. Extjs. [En línea] 2009. http://extjs.com.

Team, jQuery Development. 2009. jQuery. [En línea] 2009. http://jquery.com/.

Team, The MooTools Dev. 2009. MooTools. [En línea] 2009. http://mootools.net/.

Toledo, Carlos Aimacaña. 2008. Wikiciencia. [En línea] 2008. http://www.wikiciencia.org/informatica/programacion/iusuario/index.php.

Toxboe, Anders. 2009. ui-patterns. [En línea] 2009. http://ui-patterns.com.

Tsichritzis. 85. 85.

Vegas, Juristo - Moreno. 2006. 2006.

W3C. 2008. W3C. [En línea] 2008. http://www.w3.org.

—. 2008. W3C. [En línea] 2008. http://www.w3.org/TR/xhtml1/#xhtml.

—. 2002. W3C. [En línea] 1 de agosto de 2002. http://www.w3.org/TR/xhtml1/#xhtml.

—. 2007. World Wide Web Consortium W3C. [En línea] 2007. http://www.w3.org.

BIBLIOGRAFIA

Casanovas, Josep. 1999. Wikilearning. [En línea] 30 de noviembre de 1999. http://www.wikilearning.com/tutorial/diseno_de_formularios_conceptos_basicos-directrices_de_usabilidad_para_el_diseno_de_formularios/3953-3.

Designing Reusable Classes. Footer, Johnson y. 1988. 1988.

Frederick, Shea. 2008. *Learning Ext JS.* 2008.

Gonzalez, Carlos D. 2009. Usabilidadweb. [En línea] febrero de 2009. http://www.usabilidadweb.com.ar.

Jarrett, Caroline. 2005. Usabilitynews. [En línea] 7 de abril de 2005. http://www.usabilitynews.com/news/article2352.asp.

Fabien Potencier, François Zaninotto. *Guia definitiva de Symfony.*

GLOSARIO DE TERMINOS

- **IDE:** Integrated Development Environment (Ambiente de desarrollo integrado).

- **UCI:** Universidad de las Ciencias Informáticas.

- **Navegador:** Programa utilizado para ubicar y ver páginas Web. Por ejemplo Netscape, Mosaic, Microsoft Internet Explorer, FireFox, Opera y otros.

- **PHP:** Es un acrónimo recursivo que significa “Hypertext Pre-processor”. Es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas Web dinámicas.

- **Hardware:** Conjunto de componentes materiales de un sistema informático. Cada una de las partes físicas que forman un ordenador, incluidos sus periféricos.
- **Librerías:** En Inglés library. Cuando se habla de ordenadores, se refiere al conjunto de rutinas que realizan las operaciones usualmente requeridas por los programas. Las librerías pueden ser compartidas, lo que quiere decir que las rutinas de la librería residen en un fichero distinto de los programas que las utilizan. Los programas enlazados con bibliotecas compartidas no funcionarán a menos que se instalen las bibliotecas o librerías necesarias.
- **Software:** blando-duro, en referencia a la intangibilidad de los programas y corporeidad de la máquina. Software es un término genérico que designa al conjunto de programas de distinto tipo (sistema operativo y aplicaciones diversas) que hacen posible operar con el ordenador.
- **Usuario:** Es el que recibe los estímulos de parte del sistema y a su vez se encarga de retroalimentarlo y definir su comportamiento.
- **Framework:** es un diseño reusable de un sistema (o subsistema). Está expresado por un conjunto de clases abstractas y el modo en que sus instancias colaboran para un tipo específico de software.
- **GUI:** Interfaz Gráfica de Usuario.
- **Estándares Web:** Estos no son más que un conjunto de recomendaciones dadas por el Word Wide Web Consortium (W3C) y otras organizaciones internacionales acerca de cómo crear e interpretar documentos basados en el Web.

- **Patrones de diseño:** Es aquellos que se utiliza definir las mejores formas de construir interfaces hombre-máquina (GUI).

ANEXOS

Anexo 1: Código fuente de la Interfaz Buscar Estudiantes

```
function borrarFormBusqueda(){
    Ext.getCmp('idnombreEstudiante').setValue('');
    Ext.getCmp('idciEstudiante').setValue('');
    Ext.getCmp('idsolapinEstudiante').setValue('');
    Ext.getCmp('idfacultadEstudiante').setValue('');
    Ext.getCmp('idsexoEstudiante').setValue('');
    Ext.getCmp('idgrupoEstudiante').setValue('');
    Ext.getCmp('idaptoEstudiante').setValue('');
    Ext.getCmp('idComboCriterio').setValue('');
    Ext.getCmp('idValor').setValue('');
    Ext.getCmp('idDirecResid').setValue('');
    Ext.getDom('foto').setAttribute('src',
'../../images/individuo.JPG');
};

function comprobarImplicado(){
    var solapinEst = Ext.getCmp('solapinEstudiante').getValue();
    Ext.getCmp('listaEstImp').stopEditing();
    Ext.getCmp('listaEstImp').getStore().each(function(item){
        if (item.data.solapin == solapinEst) {
            Ext.Msg.show({
                msg: 'Este estudiante ya esta insertado en al Lista',
                title: 'Informaci&oacute;n!!!',
                icon: Ext.Msg.INFO,
                buttons: Ext.Msg.OK
            });
            Ext.getCmp('idTabIndisciplina').setDisabled(true);
        }
    });
};


panelBuscar = new Ext.FormPanel({
    width: 570,
    height: 400,
    items: [{
        xtype: 'fieldset',
        title: 'Criterio de B&uacute;squeda',
        style: 'margin:0 auto',
        labelAlign: 'right',
        defaultType: 'textfield',
        labelWidth: 60,
        width: 250,
        autoHeight: true,
        keys:[{
                key:Ext.EventObject.ENTER,
                scope:this,
                fn:function(key, e){
                        Ext.getCmp('idAceptar').focus();
                }}],
        buttonAlign: 'center',
        defaults: {
            width: 150
        },
```

```
        items: [{
            fieldLabel: 'Criterio',
            id: 'idComboCriterio',
            name: 'Cbo_criterio',
            allowBlank: false,
            blankText: 'Este campo es obligatorio',
            xtype: 'combo',
            store: new Ext.data.SimpleStore({
                fields: ['id', 'texto'],
                data: [['solapin', 'Solapin'], ['user', 'Usuario'],
['ci', 'Carnet de Identidad']]
            }),
            valueField: 'id',
            displayField: 'texto',
            hiddenName: 'Cbo_criterio ',
            triggerAction: 'all',
            mode: 'local',
            readOnly: true,
            emptyText: 'Seleccione...',
            listeners:{
               'select':function(a, b, index){
                    Ext.getCmp('idVal').setDisabled(false)
                    Ext.getCmp('idVal').setValue('')
                    if(index==0){
                       Ext.getCmp('idVal').maskRe=/[0-9 a-z A-Z]/
                       Ext.getCmp('idVal').minLength=5
                       Ext.getCmp('idVal').markInvalid('Debe escribir
5 caracteres como m&iacute;nimo');
                       Ext.getCmp('idVal').focus(true, 2)
                    }
                    if(index==1){
                        Ext.getCmp('idVal').maskRe=/[a-z]/
                        Ext.getCmp('idVal').minLength=2
                        Ext.getCmp('idVal').markInvalid('Debe escribir
2 caracteres como m&iacute;nimo');
                        Ext.getCmp('idVal').focus(true, 2)
                    }
                    if(index==2){
                        Ext.getCmp('idVal').maskRe=/[0-9]/
                        Ext.getCmp('idVal').minLength=11
                        Ext.getCmp('idVal').markInvalid('Debe escribir
11 caracteres como m&iacute;nimo');
                        Ext.getCmp('idVal').focus(true, 2)
                    }
                }
            }
        }, {
            fieldLabel: 'Valor',
            id: 'idVal',
            minLengthText:'Cantidad de caracteres incorrecta',
            maskRe : /[0-9 a-z A-Z]/,
            disabled:true,
            allowBlank: false,
            blankText: 'Este campo es obligatorio',
            name: ' FLb_valor',
            listeners:{
               'valid':function(a){
                   if(Ext.getCmp('idComboCriterio').getValue()=='ci'){

Ext.getCmp('idVal').minLength=11
                    var n = Ext.getCmp('idVal').getValue().length;
```

```
                    var t = 11;
                    if (n > t) {

Ext.getCmp('idVal').setValue(Ext.getCmp('idVal').getValue().substring(
0, t));
                      }
                   }
                }
            }
        }],
        buttons: [{
            text: 'Buscar',
            icon: '../../images/buscar1.ico',
            cls:"x-btn-text-icon",
            id: 'idAceptar',
            minWidth: 30,
            handler: function(){
                panelBuscar.form.submit({
                    url: 'buscarEstudiante',
                    waitMsg: 'Buscando...',
                    waitTitle: 'Informaci&oacute;n',
                    success: function(response, options){
                        if (options.result.id == '0') {
                            Ext.Msg.show({
                                msg: 'Este estudiante no existe en las
Base Datos UCI',
                                title: 'INFORMACION!!!',
                                icon: Ext.Msg.INFO,
                                buttons: Ext.Msg.OK
                            });

Ext.getCmp('idTabIndisciplina').setDisabled(true);
                        }
                        else {

Ext.getCmp('nombreEstudiante').setValue(options.result.nombre);

Ext.getCmp('ciEstudiante').setValue(options.result.ci);

Ext.getCmp('solapinEstudiante').setValue(options.result.solapin);

Ext.getCmp('facultadEstudiante').setValue(options.result.facultad);

Ext.getCmp('sexoEstudiante').setValue(options.result.sexo);

Ext.getCmp('grupoEstudiante').setValue(options.result.grupo);

Ext.getCmp('aptoEstudiante').setValue(options.result.apto);

Ext.getCmp('idDirecResid').setValue(options.result.direcResid);
                            Ext.getDom('foto').setAttribute('src',
options.result.foto);

Ext.getCmp('idNombreEst').setValue(options.result.nombre);

Ext.getCmp('idSolapinEst').setValue(options.result.solapin);

Ext.getCmp('idTabIndisciplina').setDisabled(false);
                            comprobarImplicado();
                        }
                    },
```

```
                    failure: function(){
                        if (Ext.getCmp('idVal').getValue() == '' ||
Ext.getCmp('idComboCriterio').getValue() == '') {
                            Ext.Msg.show({
                                msg: 'Error !!!. No pueden existir
campos vacios',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                        }
                        else {
                        if(!panelBuscar.form.isValid()){
                        Ext.Msg.show({
                                msg: 'Debe escribir el
par&aacute;metro de b&uacute;squeda correctamente',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                        }
                        else{
                            Ext.Msg.show({
                                msg: 'Error en la conexi&oacute;n',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                           }
                        }
                    }
                });
            }
        }]
    }, {
        xtype: 'fieldset',
        title: 'Resultado de la B&uacute;squeda',
        style: 'margin-top: 10px',
        labelAlign: 'right',
        labelWidth: 135,
        autoHeight: true,
        width: 560,
        layout: 'column',
        items: [{
            columnWidth: 0.72,
            layout: 'form',
            defaultType: 'textfield',
            defaults: {
                width: 230,
                readOnly: true
            },
            items: [{
                fieldLabel: 'Nombre y Apellidos',
                name: 'FLb_nombre',
                id: 'idnombreEstudiante'
            }, {
                fieldLabel: 'Carnet de Identidad',
                name: 'FLb_ci',
                id: 'idciEstudiante'
            }, {
                fieldLabel: 'Solap&iacute;n',
```

```
                name: 'FLb_solapin',
                id: 'idsolapinEstudiante'
            }, {
                fieldLabel: 'Sexo',
                name: 'FLb_sexo',
                id: 'idsexoEstudiante'
            }, {
                fieldLabel: 'Facultad',
                name: 'FLb_facultad',
                id: 'idfacultadEstudiante'
            }, {
                fieldLabel: 'Grupo',
                name: 'FLb_grupo',
                id: 'idgrupoEstudiante'
            }, {
                fieldLabel: 'Apartamento',
                name: 'FLb_apto',
                id: 'idaptoEstudiante'
            }, {
                fieldLabel: 'Direcci&oacute;n de Residencia',
                name: 'FLb_direcResid',
                id: 'idDirecResid'
            }]
        },{
            columnWidth: 0.28,
            items: [{
                autoHeight: true,
                items: [{
                    title: 'Foto',
                    width: 150,
                    autoHeight: true,
                    frame: true,
                    bodyStyle: 'padding: 5px',
                    html: '<img src="../../images/individuo.JPG"
id="foto" class="foto"/>'
                }]
            }]
        }]
    }]

});

tabs = new Ext.TabPanel ({
    width: 570,
    activeTab: 0,
    id:'idTab',
    defaults: {
        frame: true,
        autoHeight: true
    },
    listeners: {
        'tabchange': function(tab1, tab){
            tab.doLayout ();
        }
    },
    items: [{
        title: 'B&uacute;squeda',
        id: 'idTabBusqueda',
        items: [panelBuscar]
    }, {
        title: 'Indisciplina',
```

```
        id: 'idTabIndisciplina',
        disabled: true,
        items: [panelAdicionar]
    }]
});
formAgregar = new Ext.Window ({
    title: 'Adicionar Implicado',
    width: 600,
    height: 500,
    x: 100,
    y:10,
    closable: false,
    resizable: false,
    layout: 'fit',
    modal: true,
    plain: true,
    bodyStyle: 'padding:5px;',
    buttonAlign: 'center',
    items: [tabs],
    buttons: [{
        text: 'Cancelar',
        icon : '../../images/editdelete.ico',
        cls:"x-btn-text-icon",
        handler: function (){
            borrarFormBusqueda ();
            panelAdicionar.form.reset();
            Ext.getCmp('idTabIndisciplina').setDisabled(true);
            formAgregar.hide();
        }
    }]

});
```

Anexo 2: Código fuente de la Interfaz Gestionar Implicados

```
var formBuscarIndisciplina = new Ext.FormPanel({
    frame: true,
    bodyStyle: 'padding:15px 15px 0',
    defaultType: 'textfield',
    labelWidth: 45,
    labelAlign:'right',
    keys:[{
                key:Ext.EventObject.ENTER,
                scope:this,
                fn:function(key, e){
                        Ext.getCmp('idBuscar').focus();
                }}],
    width: 270,
    height: 125,
    buttonAlign: 'center',
    defaults: {
        width: 180
    },
    items: [{
        fieldLabel: 'Criterio',
        id: 'idComboCriterioInd',
```

```
        xtype: 'combo',
        name: 'Cbo_criterioInd',
        allowBlank: false,
        blankText: 'Este campo es obligatorio',
        valueField: 'id',
        displayField: 'texto',
        hiddenName: 'criterioInd',
        triggerAction: 'all',

        mode: 'local',
        readOnly: true,
        emptyText: 'Seleccione...',
        listeners:{
               'select':function(a, b, index){
                    Ext.getCmp('idValor').setDisabled(false)
                    Ext.getCmp('idValor').setValue('')
                    if(index==0){
                       Ext.getCmp('idValor').maskRe=/[0-9 a-z A-Z]/
                       Ext.getCmp('idValor').minLength=1
                       Ext.getCmp('idValor').focus(true, 2)

                    }
                    if(index==1){
                        Ext.getCmp('idValor').maskRe=/[a-z A-Z 0-9]/
                        Ext.getCmp('idValor').minLength=5
                        Ext.getCmp('idValor').focus(true, 2)
                        Ext.getCmp('idValor').markInvalid('Debe
escribir 5 caracteres como m&iacute;nimo');

                    }
                }
            },
        store: new Ext.data.SimpleStore({
            fields: ['id', 'texto'],
            data: [['idIndisciplina', 'CODIGO INSDISCIPLINA'],
['solapin', 'SOLAPIN RESPONSABLE']]
        })
    }, {
        name: ' FLb_valor',
        fieldLabel: 'Valor',
        id: 'idValor',
        minLengthText:'Cantidad de caracteres incorrecta',
        allowBlank: false,
        maskRe : /[0-9 a-z A-Z]/,
        disabled:true,
        blankText: 'Este campo es obligatorio'
    }],
    buttons: [{
        text: 'Buscar',
        icon : '../../images/buscar1.ico',
        cls:"x-btn-text-icon",
        minWidth: 30,
        minLengthText:'Cantidad de caracteres incorrecta',
        id: 'idBuscar',
        handler: function(){
            formBuscarIndisciplina.form.submit({
                url: 'buscarIndisciplina',
                waitMsg: 'Buscando...',
                waitTitle: 'Informaci&oacute;n',
                success: function(response, options){
                    if (options.result.id == 0) {
```

```
                        Ext.Msg.show({
                            msg: 'No existe ninguna indisciplina con
ese c&oacute;digo',
                            title: 'Informaci&oacute;n!!!',
                            icon: Ext.Msg.INFO,
                            buttons: Ext.Msg.OK
                        });

Ext.getCmp('idpanelIndisciplina').setVisible(false);

Ext.StoreMgr.get('storeIndisciplina').removeAll();
                    }
                    else {
                        if (options.result.id == 1) {
                            Ext.Msg.show({
                                msg: 'No existe ning&uacute;n
responsable con ese solap&iacute;n',
                                title: 'Informaci&oacute;n!!!',
                                icon: Ext.Msg.INFO,
                                buttons: Ext.Msg.OK
                            });

Ext.getCmp('idpanelIndisciplina').setVisible(false);

Ext.StoreMgr.get('storeIndisciplina').removeAll();
                        }
                        else {

Ext.getCmp('idListaIndisciplinas').getStore().reload({
                                params: {
                                    criterio:
Ext.getCmp('idComboCriterioInd').getValue(),
                                    valor:
Ext.getCmp('idValor').getValue()
                                }
                            });

                            Ext.Msg.show({
                                msg: 'Se encontraron los siguientes
datos.',
                                title: 'Informaci&oacute;n',
                                icon: Ext.Msg.INFO,
                                buttons: Ext.Msg.OK
                            });

Ext.getCmp('idpanelIndisciplina').setVisible(false);
                            formBuscarIndisciplina.form.reset();
                        }
                    }

                },
                failure: function(){
                    if (Ext.getCmp('idComboCriterioInd').getValue() ==
'' || Ext.getCmp('idValor').getValue() == '') {
                        Ext.Msg.show({
                            msg: 'Error !!!. No pueden existir campos
vacios',
                            title: 'ERROR!!!',
                            icon: Ext.Msg.ERROR,
                            buttons: Ext.Msg.OK
                        });
```

```
Ext.getCmp('idpanelIndisciplina').setVisible(false);

Ext.StoreMgr.get('storeIndisciplina').removeAll();
                    }
                    else {

                    if(!formBuscarIndisciplina.form.isValid()){
                    Ext.Msg.show({
                            msg: 'Debe escribir el par&aacute;metro de
b&uacute;squeda correctamente',
                            title: 'ERROR!!!',
                            icon: Ext.Msg.ERROR,
                            buttons: Ext.Msg.OK
                        });
                    }
                    else{
                        Ext.Msg.show({
                            msg: 'Error en la conexi&oacute;n',
                            title: 'ERROR!!!',
                            icon: Ext.Msg.ERROR,
                            buttons: Ext.Msg.OK
                        });

Ext.getCmp('idpanelIndisciplina').setVisible(false);

Ext.StoreMgr.get('storeIndisciplina').removeAll();
                    }
                    }
                }
            });
        }

    }]
});
var idInd = 0;
var gridIndisciplinas = new Ext.grid.EditorGridPanel({
    enableColumnResize: false,
    id: 'idListaIndisciplinas',
    clicksToEdit: 2,
    collapsible: true,
    animCollapse: false,
    autoScroll: true,
    width: 265,
    iconCls: 'icon-grid',
    height: 125,
    frame: true,
    store: new Ext.data.JsonStore({
        id: 'storeIndisciplina',
        root: 'indisciplinas',
        url: 'listaIndisciplina',
        fields: ['idIndisciplina', 'fecha', 'descripcion']
    }),
    cm: new Ext.grid.ColumnModel([{
        header: "C&oacute;digo Indisciplina",
        align: 'center',
        width: 115,
        dataIndex: 'idIndisciplina'
    }, {
        header: "Fecha Indisciplina",
        width: 115,
```

```
        align: 'center',
        dataIndex: 'fecha'
    },
    {
        header: "Descripci&oacute;n",
        width: 8000000,
        hidden: true,
        dataIndex: 'descripcion'
    }
    ]),
    sm: new Ext.grid.RowSelectionModel({
        singleSelect: true,
        listeners: {
            rowselect: function(sm, row, rec){
                //optienes el primer registro seleccionado
                var record = sm.getSelected().data; //es el primer
registro selecionado
                //en record tienes los valor del registro
                Ext.getCmp('idFechaInd').setValue(record['fecha']);

Ext.getCmp('idDescipcionInd').setValue(record['descripcion']);
                Ext.getCmp('idpanelIndisciplina').setVisible(true);
                idInd = record['idIndisciplina'];
                Ext.getCmp('idlistaEstImp').getStore().reload({
                    params: {
                        idIndisciplina: idInd
                    }
                });
            }
        }
    })

});



Ext.onReady(function(){
    Ext.QuickTips.init();
    var boleT = new Ext.Panel({
        renderTo: document.body,
        id: 'idpanelPrincipal',
        frame: true,
        title: 'Gestionar Implicados',
        bodyStyle: 'padding:5px 5px 0',
        buttonAlign: 'center',
        style: ' margin:0 auto',
        width:795,
        autoHeight: true,
        layout: 'column',
        items: [{
            header: false,
            width: 85,
            height: 140,
            bodyStyle: 'padding:5px 5px 5px 170px'
        }, formBuscarIndisciplina, {
            title: '',
            header: false,
            width: 50,
            bodyStyle: 'padding:5px 5px 5px 170px'
        }, gridIndisciplinas, {
```

```
            xtype: 'form',

            title: 'Datos de la Indisciplina',
            id: 'idpanelIndisciplina',
            hidden: true,
            frame: true,
            bodyStyle: 'padding:5px 5px 0',
            style: 'margin:0 auto',
            autoHeight: true,
            width: 765,
            items: [{
                xtype: 'fieldset',
                title: 'Datos principales',
                bodyStyle: 'padding:5px 5px 0',
                labelWidth: 80,
                style: 'margin:0 auto',
                height: 170,
                width: 300,
                defaults: {
                    width: 185,
                    readOnly: true
                },
                items: [{
                    xtype: 'textfield',
                    fieldLabel: 'Fecha',
                    name: 'TxF_fechaInd',
                    id: 'idFechaInd'
                }, {
                    xtype: 'textarea',
                    fieldLabel: 'Descripci&oacute;n',
                    name: 'TxA_descripcionInd',
                    id: 'idDescipcionInd',
                    height: 100
                }]
            }, {
                xtype: 'editorgrid',
                enableColumnResize: false,
                id: 'idlistaEstImp',
                clicksToEdit: 2,
                style: 'margin:15px 0px 0px 0px',
                title: 'Lista de Estudiantes Implicados',
                autoScroll: true,
                width: 743,
                height: 250,
                frame: true,
                store: new Ext.data.JsonStore({
                    id: 'storeImplicado',
                    root: 'implicados',
                    url: 'listaImplicado',
                    fields: ['nombre', 'solapin', 'caracter',
'articuloInd', 'incisoInd', 'articuloMed', 'incisoMed']
                }),
                colModel: new Ext.grid.ColumnModel([{
                    header: "Nombre",
                    width: 215,
                    dataIndex: 'nombre'
                }, {
                    header: "Solap&iacute;n",
                    width: 54,
                    align: 'center',
                    dataIndex: 'solapin'
```

```
                }, {
                    header: "Car&aacute;cter",
                    width: 90,
                    align: 'center',
                    dataIndex: 'caracter'
                }, {
                    header: "Art&iacute;culo Indisciplina",
                    width: 100,
                    align: 'center',
                    dataIndex: 'articuloInd'
                }, {
                    header: "Inciso Indisciplina",
                    width: 95,
                    align: 'center',
                    dataIndex: 'incisoInd'
                }, {
                    header: "Art&iacute;culo Medida",
                    width: 80,
                    align: 'center',
                    dataIndex: 'articuloMed'
                }, {
                    header: "Inciso Medida",
                    width: 75,
                    align: 'center',
                    dataIndex: 'incisoMed'
                }]),
                sm: new Ext.grid.RowSelectionModel({
                    singleSelect: true,
                    listeners: {
                        rowselect: function(sm, row, rec){
                            //optienes el primer registro seleccionado
                            var record = sm.getSelected().data; //es
el primer registro selecionado
                            //en record tienes los valor del registro

                            nombreImplicado = record['nombre'];
                            solapinImplicado = record['solapin'];
                            caracterIndisciplina = record['caracter'];
                            articuloIndis = record['articuloInd'];
                            incisoIndis = record['incisoInd'];
                            articuloMedida = record['articuloMed'];
                            incisoMedida = record['incisoMed'];
                        }
                    }
                }),
                tbar: [{
                    text: 'Agregar ',
                    icon : '../../images/edit_add.ico',
                    cls:"x-btn-text-icon",
                    handler: function(){
                        formAgregar.show();
                    }
                }, '-', {
                    text: 'Eliminar ',
                     icon : '../../images/delete.ico',
                     cls:"x-btn-text-icon",
                    handler: function(){
                        if
(!Ext.getCmp('idlistaEstImp').getSelectionModel().getSelected()) {
                            Ext.Msg.show({
                                msg: 'Informaci&oacute;n !!!. Debe
```

```
marcar el implicado a eliminar',
                                    title: 'Informaci&oacute;n!!!',
                                    icon: Ext.Msg.INFO,
                                    buttons: Ext.Msg.OK
                                });
                            }
                            else {
                                Ext.Ajax.request({
                                    url: 'eliminarImplicado',
                                    params: {
                                        idIndis: idInd,
                                        solapinImp: solapinImplicado
                                    },
                                    success: function(response){
                                        Ext.Msg.show({
                                            msg: 'Se ha eliminado el
Implicado correctamente',
                                            title: 'Informaci&oacute;n',
                                            icon: Ext.Msg.INFO,
                                            buttons: Ext.Msg.OK
                                        });

Ext.getCmp('idlistaEstImp').getStore().reload();
                                    }
                                })
                            }
                        }
                    }, '-', {
                        text: 'Modificar',
                         icon : '../../images/modificar.ico',
                        cls:"x-btn-text-icon",
                        handler: function(){
                            if
(!Ext.getCmp('idlistaEstImp').getSelectionModel().getSelected()) {
                                Ext.Msg.show({
                                    msg: 'Informaci&oacute;n !!!. Debe
marcar el implicado a modificar',
                                    title: 'Informaci&oacute;n!!!',
                                    icon: Ext.Msg.INFO,
                                    buttons: Ext.Msg.OK
                                });
                            }
                            else {

Ext.getCmp('idNombreImplicado').setValue(nombreImplicado);

Ext.getCmp('idSolapinImplicado').setValue(solapinImplicado);

Ext.getCmp('idCarMed').setValue(caracterIndisciplina);

Ext.getCmp('idArtInd').setValue(articuloIndis);

Ext.getCmp('idIncInd').setValue(incisoIndis);

Ext.getCmp('idArtMed').setValue(articuloMedida);

Ext.getCmp('idIncMed').setValue(incisoMedida);
                                formModificar.show();
                            }
                        }
                    }, '-']
```

```
            }]
        }]
    });
});
```

Anexo 3: Código fuente de la Interfaz Adicionar Implicados

```
panelAdicionar = new Ext.FormPanel({
    width: 560,
    height: 380,
    buttonAlign: 'center',
    bodyStyle: 'padding:15px 15px 0',
    items: [{
        xtype: 'fieldset',
        title: 'Datos del Estudiante',
        bodyStyle: 'padding:5px 10px 5px 10px',
        style: 'margin:0 auto',
        defaultType: 'textfield',
        labelWidth: 50,
        width: 320,
        autoHeight: true,
        defaults: {
            width: 220,
            readOnly: true
        },
        items: [{
            fieldLabel: 'Nombre',
            id: 'idNombreEst',
            name: 'FLb_nombre'
        }, {
            fieldLabel: 'Solap&iacute;n',
            id: 'idSolapinEst',
            name: 'FLb_solapin'
        }]
    }, {
        xtype: 'fieldset',
        title: 'Datos de la Indisciplina',
        bodyStyle: 'padding:5px 10px 5px 10px',
        style: 'margin: 15px 0px 0px 118px',
        defaultType: 'textfield',
        labelWidth: 130,
        width: 300,
        autoHeight: true,
        defaults: {
            width: 115
        },
        items: [{
            xtype: 'combo',
            fieldLabel: 'Car&aacute;cter',
            id: 'idCaracter',
            name: 'Cbo_caracterInd',
            store: new Ext.data.SimpleStore({
                storeId: 'idStoreCaracter',
                fields: ['id', 'texto'],
                data: [['Leve', 'Leve'], ['Menos Grave', 'Menos
Grave'], ['Grave', 'Grave'], ['Muy Grave', 'Muy Grave']]
```

```
            }),
            valueField: 'id',
            displayField: 'texto',
            hiddenName: 'Cbo_caracterInd',
            triggerAction: 'all',
            mode: 'local',
            readOnly: true,
            allowBlank: false,
            blankText: 'Este campo es obligatorio',
            emptyText: 'Seleccione...'
        }, {
            xtype: 'combo',
            fieldLabel: 'Art&iacute;culo Indisciplina',
            id: 'idArticuloInd',
            name: 'Cbo_articuloInd',
            store: new Ext.data.SimpleStore({
                storeId: 'idStoreArticuloInd',
                fields: ['id', 'texto'],
                data: [[1, 1], [2, 2], [3, 3], [4, 4], [5, 5]]
            }),
            valueField: 'id',
            displayField: 'texto',
            hiddenName: 'Cbo_articuloInd',
            triggerAction: 'all',
            mode: 'local',
            readOnly: true,
            allowBlank: false,
            blankText: 'Este campo es obligatorio',
            emptyText: 'Seleccione...'
        }, {
            xtype: 'combo',
            fieldLabel: 'Inciso Indisciplina',
            id: 'idIncisoInd',
            name: 'Cbo_incisoInd',
            store: new Ext.data.SimpleStore({
                storeId: 'idStoreIncisoInd',
                fields: ['id', 'texto'],
                data: [['A', 'A'], ['B', 'B'], ['C', 'C'], ['D', 'D']]
            }),
            valueField: 'id',
            displayField: 'texto',
            hiddenName: 'Cbo_incisoInd',
            triggerAction: 'all',
            mode: 'local',
            readOnly: true,
            allowBlank: false,
            blankText: 'Este campo es obligatorio',
            emptyText: 'Seleccione...'
        }, {
            xtype: 'combo',
            fieldLabel: 'Art&iacute;culo Medida',
            id: 'idArticuloMed',
            name: 'Cbo_articuloMed',
            store: new Ext.data.SimpleStore({
                storeId: 'idStoreArticuloMed',
                fields: ['id', 'texto'],
                data: [['1', '1'], ['2', '2'], ['3', '3'], ['4', '4']]
            }),
            valueField: 'id',
            displayField: 'texto',
            hiddenName: 'Cbo_articuloMed',
```

```
            triggerAction: 'all',
            mode: 'local',
            readOnly: true,
            emptyText: 'Seleccione...'
        }, {
            xtype: 'combo',
            fieldLabel: 'Inciso Medida',
            id: 'idIncisoMed',
            name: 'Cbo_incisoMed',
            store: new Ext.data.SimpleStore({
                storeId: 'idStoreIncisoInd',
                fields: ['id', 'texto'],
                data: [['A', 'A'], ['B', 'B'], ['C', 'C'], ['D', 'D']]
            }),
            valueField: 'id',
            displayField: 'texto',
            hiddenName: 'Cbo_incisoMed',
            triggerAction: 'all',
            mode: 'local',
            readOnly: true,
            emptyText: 'Seleccione...'
        }]
    }],
    buttons: [{
        text: 'Aceptar',
        icon : '../../images/ok.ico',
        cls:"x-btn-text-icon",
        style: 'margin: 0px 0px 25px 5px',
        handler: function(){
            var valido = true;
            if (Ext.getCmp('idCaracter').getValue() == 'Leve' &&
(Ext.getCmp('idArticuloMed').getValue() == '' ||
Ext.getCmp('idIncisoMed').getValue() == '')) {
                valido = false;
            }
            if (valido == true) {
                panelAdicionar.form.submit({
                    url: 'insertarImplicado',
                    waitMsg: 'Guardando...',
                    waitTitle: 'Espere por favor',
                    params: {
                        idIndis: idInd
                    },
                    success: function(response){
                        Ext.Msg.show({
                            msg: 'Se ha registrado el implicado
correctamente',
                            title: 'Informaci&oacute;n',
                            icon: Ext.Msg.INFO,
                            buttons: Ext.Msg.OK
                        });

Ext.getCmp('idlistaEstImp').getStore().reload();
                        panelAdicionar.form.reset();
                        Ext.getDom('foto').setAttribute('src',
'../../images/individuo.JPG');
                        panelBuscar.form.reset();

Ext.getCmp('idTabIndisciplina').setDisabled(true);

Ext.getCmp('idTab').activate('idTabBusqueda');
```

```
                    },
                    failure: function(){
                        if (Ext.getCmp('idCaracter').getValue() == ''
|| Ext.getCmp('idArticuloInd').getValue() == '' ||
Ext.getCmp('idIncisoInd').getValue() == '') {
                            Ext.Msg.show({
                                msg: 'Error !!!. No pueden existir
campos vacios',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                        }
                        else {
                            Ext.Msg.show({
                                msg: 'Error en la conexi&oacute;n',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                        }
                    }
                });
            }
            else {
                Ext.Msg.show({
                    msg: 'Error !!!. No pueden existir campos vacios',
                    title: 'ERROR!!!',
                    icon: Ext.Msg.ERROR,
                    buttons: Ext.Msg.OK
                });
            }

        }
    }]
});
```

Anexo 4: Código fuente de la Interfaz Modificar Implicados

```
function comprobarQueSeModifiqueAlgo(){
    if (Ext.getCmp('idCarMed').getValue() != '' &&
(Ext.getCmp('idCaracterMod').getValue() ==
Ext.getCmp('idCarMed').getValue())) {
        return 1;
    }
    else {
        if (Ext.getCmp('idArtInd').getValue() != '' &&
(Ext.getCmp('idArtIndMod').getValue() ==
Ext.getCmp('idArtInd').getValue())) {
            return 1;
        }
        else {
            if (Ext.getCmp('idIncInd').getValue() != '' &&
(Ext.getCmp('idIncIndMod').getValue() ==
Ext.getCmp('idIncInd').getValue())) {
                return 1;
```

```
            }
            else {
                if (Ext.getCmp('idArtMed').getValue() != '' &&
(Ext.getCmp('idArtMedMod').getValue() ==
Ext.getCmp('idArtMed').getValue())) {
                    return 1;
                }
                else {
                    if (Ext.getCmp('idIncMed').getValue() != '' &&
(Ext.getCmp('idIncMedMod').getValue() ==
Ext.getCmp('idIncMed').getValue())) {
                        return 1;
                    }
                }
            }
        }
    }
      return 0;
};

panelModificar = new Ext.Panel({
    frame: true,
    items: [{
        xtype: 'form',
        id: 'idFormDatosEst',
        bodyStyle: 'padding: 10px 10px 5px 10px',
        autoHeight: true,
        autoWidth: true,
        items: [{
            xtype: 'fieldset',
            title: 'Datos del Estudiante',
            bodyStyle: 'padding:5px 10px 5px 10px',
            style: 'margin:0 auto',
            defaultType: 'textfield',
            labelWidth: 50,
            width: 328,
            autoHeight: true,
            defaults: {
                width: 230,
                readOnly: true
            },
            items: [{
                fieldLabel: 'Nombre',
                name: 'FLb_nombreImplicado',
                id: 'idNombreImplicado'
            }, {
                fieldLabel: 'Solap&iacute;n',
                name: 'FLb_solapinImplicado',
                id: 'idSolapinImplicado'
            }]
        }]
    }, {
        xtype: 'panel',
        width: 455,
        autoHeight: true,

        layout: 'column',
        items: [{
            columnWidth: 0.65,
            layout: 'form',
            items: [{
```

```
                xtype: 'form',
                id: 'idFormDatosInd',
                style: 'margin: 15px 0px 0px 5px',
                height: 200,
                width: 290,
                items: [{
                    xtype: 'fieldset',
                    title: 'Datos de la Indisciplina',
                    bodyStyle: 'padding:5px 5px 5px 5px',
                    defaultType: 'textfield',
                    labelWidth: 125,
                    width: 277,
                    autoHeight: true,
                    defaults: {
                        width: 115,
                        readOnly: true
                    },
                    items: [{
                        fieldLabel: 'Car&aacute;cter',
                        name: 'FLb_carMed',
                        id: 'idCarMed'
                    }, {
                        fieldLabel: 'Art&iacute;culo Indisciplina',
                        name: 'FLb_artInd',
                        id: 'idArtInd'
                    }, {
                        fieldLabel: 'Inciso Indisciplina',
                        name: 'FLb_incInd',
                        id: 'idIncInd'
                    }, {
                        fieldLabel: 'Art&iacute;culo Medida',
                        name: 'FLb_artMed',
                        id: 'idArtMed'
                    }, {
                        fieldLabel: 'Inciso Medida',
                        name: 'FLb_incMed',
                        id: 'idIncMed'
                    }]
                }]
            }]
        }, {
            columnWidth: 0.35,
            layout: 'form',
            items: [{
                xtype: 'form',
                id: 'idFormModificarImp',
                style: 'margin: 15px 5px 0px 0px',
                buttonAlign: 'center',
                height: 220,
                width: 170,
                items: [{
                    xtype: 'fieldset',
                    title: 'Datos a Modificar',
                    bodyStyle: 'padding: 5px 2px 5px 2px',
                    defaultType: 'combo',
                    labelWidth: 0.2,
                    width: 152,
                    autoHeight: true,
                    defaults: {
                        width: 115,
                        labelSeparator: '',
```

```
                        valueField: 'id',
                        displayField: 'texto',
                        triggerAction: 'all',
                        mode: 'local',
                        readOnly: true,
                        emptyText: 'Seleccione...'
                    },
                    items: [{
                        id: 'idCaracterMod',
                        name: 'Cbo_caracterMod',
                        store: new Ext.data.SimpleStore({
                            storeId: 'idStoreCaracter',
                            fields: ['id', 'texto'],
                            data: [['Leve', 'Leve'], ['Menos Grave',
'Menos Grave'], ['Grave', 'Grave'], ['Muy Grave', 'Muy Grave']]
                        }),
                        hiddenName: 'Cbo_caracterMod'
                    }, {
                        id: 'idArtIndMod',
                        name: 'Cbo_ArtIndMod',
                        store: new Ext.data.SimpleStore({
                            storeId: 'idStoreArticuloInd',
                            fields: ['id', 'texto'],
                            data: [[1, 1], [2, 2], [3, 3], [4, 4], [5,
5]]
                        }),
                        hiddenName: 'Cbo_ArtIndMod'
                    }, {
                        id: 'idIncIndMod',
                        name: 'Cbo_IncIndMod',
                        store: new Ext.data.SimpleStore({
                            storeId: 'idStoreIncisoInd',
                            fields: ['id', 'texto'],
                            data: [['A', 'A'], ['B', 'B'], ['C', 'C'],
['D', 'D']]
                        }),
                        hiddenName: 'Cbo_IncIndMod'
                    }, {
                        id: 'idArtMedMod',
                        name: 'Cbo_ArtMedMod',
                        store: new Ext.data.SimpleStore({
                            storeId: 'idStoreArticuloMed',
                            fields: ['id', 'texto'],
                            data: [[1, 1], [2, 2], [3, 3], [4, 4]]
                        }),
                        hiddenName: 'Cbo_ArtMedMod'
                    }, {
                        id: 'idIncMedMod',
                        name: 'Cbo_IncMedMod',
                        store: new Ext.data.SimpleStore({
                            storeId: 'idStoreIncisoMed',
                            fields: ['id', 'texto'],
                            data: [['A', 'A'], ['B', 'B'], ['C', 'C'],
['D', 'D']]
                        }),
                        hiddenName: 'Cbo IncMedMod'
                    }]
                }],
                buttons: [{
                    text: 'Modificar',
                    icon : '../../images/modificar.ico',
```

```
                    cls:"x-btn-text-icon",
                    handler: function(){
                        if (Ext.getCmp('idCaracterMod').getValue() ==
'' && Ext.getCmp('idArtIndMod').getValue() == '' &&
Ext.getCmp('idIncIndMod').getValue() == '' &&
Ext.getCmp('idArtMedMod').getValue() == '' &&
Ext.getCmp('idIncMedMod').getValue() == '') {
                            Ext.Msg.show({
                                msg: 'Debe modificar al menos un valor
del implicado',
                                title: 'Informacion',
                                icon: Ext.Msg.INFO,
                                buttons: Ext.Msg.OK
                            });
                        }
                        else {
                            var valor = comprobarQueSeModifiqueAlgo();
                            if (valor == 1) {
                                Ext.Msg.show({
                                    msg: 'Los datos modificados son
los mismo que tiene el implicado',
                                    title: 'Informaci&oacute;n',
                                    icon: Ext.Msg.INFO,
                                    buttons: Ext.Msg.OK
                                });
                            }
                            else {

Ext.getCmp('idFormModificarImp').form.submit({
                                    url: 'modificarImplicado',
                                    waitMsg: 'Guardando...',
                                    waitTitle: 'Espere por favor',
                                    params: {
                                        idIndis: idInd,
                                        solapinImp: solapinImplicado
                                    },
                                    success: function(response){
                                        Ext.Msg.show({
                                            msg: 'Se han modificado
los datos ',

                                            title:
'Informaci&oacute;n',

                                            icon: Ext.Msg.INFO,
                                            buttons: Ext.Msg.OK
                                        });

Ext.getCmp('idlistaEstImp').getStore().reload();
                                        formModificar.hide();

Ext.getCmp('idFormModificarImp').form.reset();
                                    },
                                    failure: function(){

                                        Ext.Msg.show({
                                            msg: 'Error en la
conexi&oacute;n',

                                            title: 'ERROR!!!',
                                            icon: Ext.Msg.ERROR,
                                            buttons: Ext.Msg.OK
                                        });
                                    }
```

```
                                    });
                                }
                            }

                        }
                    }]
                }]

            }]
        }]
});

formModificar = new Ext.Window({
    title: 'Modificar Implicado',
    width: 493,
    height: 445,
    closable: false,
    resizable: false,
    layout: 'fit',
    modal: true,
    plain: true,
    bodyStyle: 'padding:5px;',
    buttonAlign: 'center',
    items: [panelModificar],
    buttons: [{
        text: 'Cancelar',
        icon : '../../images/editdelete.ico',
         cls:"x-btn-text-icon",
         handler: function(){
            formModificar.hide();
            Ext.getCmp('idFormModificarImp').form.reset();
        }
    }]
});
```

Anexo 5: Código fuente de la Interfaz Estructura FEU

```
panelResponsable = new Ext.FormPanel({
    frame: true,
    width: 768,
    keys:[{
                key:Ext.EventObject.ENTER,
                scope:this,
                fn:function(key, e){
                        Ext.getCmp('idacep').focus();
                }}],
    buttonAlign: 'center',
    height: 423,
    items: [{
        xtype: 'fieldset',
        title: 'Datos del Estudiante',
        style: 'margin: 10px 0px 0px 230px',
        width: 298,
        height: 280,
        labelWidth: 120,
        labelAlign: 'right',
        defaultType: 'textfield',
        defaults: {
            width: 140,
```

```
            readOnly: true
        },
        items: [{
            fieldLabel: 'Edificio',
            name: 'FLb_edificio',
            id: 'idEdificio'
        }, {
            fieldLabel: 'Apartamento',
            name: 'FLb_apartamento',
            id: 'idApto'
        }, {
            fieldLabel: 'Paso de Escalera',
            name: 'FLb_pasoEscalera',
            id: 'idPasoEscalera'
        },{
                fieldLabel: 'Cargos',
            name: 'FLb_cargos',
            id: 'idcargo'
            }, {
            name: 'FLb_solapin',
            id: 'idSolapin',
            labelSeparator: '',
            hidden: true
        }, {
            xtype: 'fieldset',
            style: 'margin: 20px 0px 0px 6px',
            title: 'Responsabilidad',
            defaultType: 'textfield',
            labelAlign: 'right',
            buttonAlign: 'center',
            labelWidth: 60,
            width: 260,
            height: 100,
            items: [{
                fieldLabel: 'Cargo',
                id: 'idComboCargo',
                width: 140,
                name: 'Cbo_cargo',
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                xtype: 'combo',
                store: new Ext.data.SimpleStore({
                    fields: ['id', 'texto'],
                    data: [['Jefe de Edificio', 'Jefe de Edificio'],
['Jefe Apartamento', 'Jefe Apartamento'], ['Jefe Paso Escalera', 'Jefe
Paso Escalera'], ['Sin cargo','Sin cargo']]
                }),
                valueField: 'id',
                displayField: 'texto',
                triggerAction: 'all',
                mode: 'local',
                readOnly: true,
                emptyText: 'Seleccione...'
            }],
            buttons: [{
                text: 'Aceptar',
                icon : '../../images/ok.ico',
                cls:"x-btn-text-icon",
                minWidth: 30,
                id:'idacep',
                handler: function(){
```

```
                    var valido = false;
                    var seleccion =
Ext.getCmp('idComboCargo').getValue();
                    var valor = Ext.getCmp('idEdificio').getValue();
                          if (valor != '' && seleccion != ''&&
seleccion=='Sin cargo') {
                        panelResponsable.form.submit({
                            url: 'sincargo',
                            waitMsg: 'Quitando el cargo...',
                            waitTitle: 'Espere por favor!!!',
                            success: function(){
                                Ext.Msg.show({
                                    msg: 'Cambios guardados
correctamente',
                                    title: 'Informaci&oacute;n',
                                    icon: Ext.Msg.INFO,
                                    buttons: Ext.Msg.OK
                                });
                                panelBuscar.form.reset();

      panelResponsable.form.reset();

Ext.getDom('foto').setAttribute('src', '../../images/individuo.JPG');

      Ext.getCmp('idEdificio').setValue('');
                                Ext.getCmp('idApto').setValue('');

Ext.getCmp('idPasoEscalera').setValue('');

      Ext.getCmp('idcargo').setValue('');

      Ext.getCmp('idResponsable').setDisabled(true);

Ext.getCmp('idmitab').activate('idBuscar');
                            },
                            failure: function(){
                                Ext.Msg.show({
                                    msg: 'Error en la
conexi&oacute;n',
                                    title: 'ERROR!!!',
                                    icon: Ext.Msg.ERROR,
                                    buttons: Ext.Msg.OK
                                });
                            }
                        });
                    }
                          else if (valor != '' && seleccion != '')
{
                        panelResponsable.form.submit({
                            url: 'insertarResponsable',
                            waitMsg: 'Guardando...',
                            waitTitle: 'Espere por favor!!!',
                            success: function(){
                                Ext.Msg.show({
                                    msg: 'Cambios guardados
correctamente',
                                    title: 'Informaci&oacute;n',
                                    icon: Ext.Msg.INFO,
                                    buttons: Ext.Msg.OK
                                });
                                panelBuscar.form.reset();
```

```
      panelResponsable.form.reset();

Ext.getDom('foto').setAttribute('src', '../../images/individuo.JPG');

      Ext.getCmp('idEdificio').setValue('');
                                  Ext.getCmp('idApto').setValue('');

Ext.getCmp('idPasoEscalera').setValue('');

      Ext.getCmp('idcargo').setValue('');

      Ext.getCmp('idResponsable').setDisabled(true);

      Ext.getCmp('idmitab').activate('idBuscar');
                                },
                                failure: function(){
                                    Ext.Msg.show({
                                        msg: 'Error en la
conexi&oacute;n',
                                        title: 'ERROR!!!',
                                        icon: Ext.Msg.ERROR,
                                        buttons: Ext.Msg.OK
                                    });
                                }
                            });
                        }
                        else {
                            Ext.Msg.show({
                                msg: 'Error !!!. No pueden existir campos
vacios',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                        }
                    }
                }]
            }]
    }]
});

Ext.onReady(function(){
    Ext.QuickTips.init();
    var tabs = new Ext.TabPanel({
        renderTo: document.body,
        activeTab: 0,
        id:'idmitab',
        border: false,
        style: ' margin:0 auto',
        buttonAlign: 'center',
        width: 788,
        height: 510,

        defaults: {
            frame: true
        },
        listeners: {
            'tabchange': function(tabpanel, tab){
                tab.doLayout();
            }
```

```
        },
        items: [{
            border: false,
            title: 'Buscar',
            id: 'idBuscar',
            items: [panelBuscar]
        }, {

 title: 'Responsable',
 id: 'idResponsable',
 disabled:true,
  listeners:{
   'activate': function(){
     Ext.Ajax.request({
    url:'cargo',
    params:{
    solapin: Ext.getCmp('idSolapin').getValue()
    },
     success:function(resp){
            Ext.getCmp('idcargo').setValue(resp.responseText);
      }
        });
}
   },
            items: [panelResponsable]
        }]
    });
});
```

Anexo 6: Código fuente de la Interfaz Realizar Evaluación

```
formAgregarEvaluacion = new Ext.FormPanel({
    bodyStyle: 'padding:5px 5px 0',
    style: ' margin:0 auto',
    id: 'idformAgregarEvaluacion',
    height: 385,
    width: 470,
    items: [{
        xtype: 'fieldset',
        title: 'Introducir Evaluaci&oacute;n',
        bodyStyle: 'padding:5px 5px 0',
        style: 'margin:0 auto',
        buttonAlign: 'center',
        labelAlign: 'left',
        labelWidth: 130,
        width: 412,
        height: 337,
        items: [{
            xtype: 'fieldset',
            title:'Datos',
            bodyStyle: 'padding:5px 5px 0',
            style: 'margin:0 auto',
            buttonAlign: 'center',
            labelAlign: 'right',
            labelWidth: 163,
            width: 300,
            height: 80,
```

```
            items: [{
                xtype: 'datefield',
                id: 'idFecha',
                allowBlank: false,
                readOnly: true,
                width: 105,
                blankText: 'Este campo es obligatorio',
                fieldLabel: 'Fecha',
                name: 'Dte_fecha'
            }, {
                xtype: 'combo',
                fieldLabel: 'Evaluaci&oacute;n del Apartamento',
                id: 'idComboEvaluacionApto',
                name: 'Cbo_evaluacionApto',
                width: 105,
                allowBlank: false,
                blankText: 'Este campo es obligatorio',
                store: new Ext.data.SimpleStore({
                    fields: ['id', 'texto'],
                    data: [['B', 'B'], ['R', 'R'], ['M', 'M']]
                }),
                valueField: 'id',
                displayField: 'texto',
                hiddenName: 'Cbo_evaluacionApto',
                triggerAction: 'all',
                mode: 'local',
                readOnly: true,
                emptyText: 'Seleccione...'
            }, {
                        xtype: 'textfield',
                        name: 'TxF_apartamento',
                id: 'idApartamento',
                labelSeparator: '',
                hidden: true
                  }]
        }, {
            xtype: 'editorgrid',
            id: 'idlistaEstApto',
            style: 'margin: 15px 0px 0px 0px',
            clicksToEdit: 2,
            enableColumnResize: false,
            title: 'Estudiantes del Apartamento',
            autoScroll: true,
            width: 380,
            height: 190,
            frame: true,
            store: new Ext.data.JsonStore({
                id: 'storeEstApto',
                root: 'integrantes',
                url: 'listaIntegrantes',
                fields: ['nombre', 'solapin', 'evaluacion']
            }),
            colModel: new Ext.grid.ColumnModel([{
                header: "Nombre",
                width: 220,
                dataIndex: 'nombre'
            }, {
                header: "Solap&iacute;n",
                width: 54,
                align: 'center',
                dataIndex: 'solapin'
```

```
            }, {
                header: "Evaluaci&oacute;n",
                align: 'center',
                width: 73,
                dataIndex: 'evaluacion',
                editor: new Ext.form.ComboBox({
                    valueField: 'id',
                    id: 'idcomboEvaluacion',
                    displayField: 'evaluacion',
                    readOnly: true,
                    triggerAction: 'all',
                    mode: 'local',
                    store: new Ext.data.SimpleStore({
                        storeId: 'storecomboCaracter',
                        fields: ['id', 'evaluacion'],
                        data: [['B', 'B'], ['R', 'R'], ['M', 'M']]
                    })
                })
            }])
        }]
    }],
    buttons: [{
        text: 'Aceptar',
         icon : '../../images/ok.ico',
         cls:"x-btn-text-icon",
        handler: function(){
            var valido = false;
            var cantidad =
Ext.StoreMgr.get('storeEstApto').getCount();
            if (cantidad != 0) {
                valido = true;
                Ext.StoreMgr.get('storeEstApto').each(function(value){
                    if (value.data.evaluacion == '') {
                        valido = false;
                    }
                });
            }
            if (valido) {
                var array = [];
                Ext.StoreMgr.get('storeEstApto').each(function(){
                    array.push(this.data);
                });
                formAgregarEvaluacion.form.submit({
                    url: 'insertarEvaluacion',
                    waitMsg: 'Guardando...',
                    waitTitle: 'Espere por favor!!!',
                    params: {
                        integrantes: Ext.encode(array)
                    },
                    success: function(param){
                        Ext.Msg.show({
                            msg: 'Se guardaron los datos',
                            title: 'Informaci&oacute;n',
                            icon: Ext.Msg.INFO,
                            buttons: Ext.Msg.OK
                        });
                                    formAgregarEvaluacion.form.reset();
                                    formBuscar.form.reset();

      Ext.StoreMgr.get('storeEstApto').removeAll();
                    },
```

```
                    failure: function(){
                        if (Ext.getCmp('idFecha').getValue() == '' ||
Ext.getCmp('idComboEvaluacionApto').getValue() == '') {
                            Ext.Msg.show({
                                msg: 'Error !!!. No pueden existir
campos vacios',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                        }
                        else {
                            Ext.Msg.show({
                                msg: 'Error en la conexi&oacute;n',
                                title: 'ERROR!!!',
                                icon: Ext.Msg.ERROR,
                                buttons: Ext.Msg.OK
                            });
                        }
                    }
                });
            }
            else {
                Ext.Msg.show({
                    msg: 'Error !!!. No pueden existir campos vacios',
                    title: 'ERROR!!!',
                    icon: Ext.Msg.ERROR,
                    buttons: Ext.Msg.OK
                });
            }
        }
    }]
});

formBuscar = new Ext.FormPanel({
    bodyStyle: 'padding:5px 5px 0',
    style: ' margin:0 auto',
    id: 'idformBuscar',
    height: 122,
    width: 300,
    items: [{
        xtype: 'fieldset',
        title: 'Buscar Apartamento',
        bodyStyle: 'padding:5px 5px 0',
        style: 'margin:0 auto',
        buttonAlign: 'center',
        keys:[{
                key:Ext.EventObject.ENTER,
                scope:this,
                fn:function(key, e){
                        Ext.getCmp('idAceptar').focus();
                }}],
        labelAlign: 'right',
        labelWidth: 95,
        width: 255,
        height: 108,
        items: [{
            xtype: 'textfield',
            allowBlank: false,
            id: 'idApto',
            blankText: 'Introduzca el apartamento',
```

```
            fieldLabel: 'Apartamento',
            width: 120,
            minLength:4,
            listeners:{
                  'valid':function(e){
                   if(Ext.getCmp('idApto').getValue()==0){
                   Ext.getCmp('idApto').setValue('');
                    }
                    }
                    },
            minLengthText:'Debe escribir entre 4 y 6 caracteres',
            allowDecimals:false,
            allowNegative : false,
            maskRe : /[0-9]/,
            name: 'TxF_AptoBuscar',
            listeners:{
        'valid':function(a){
    var n = Ext.getCmp('idApto').getValue().length;
    var t = 6;
    if (n > t) {

Ext.getCmp('idApto').setValue(Ext.getCmp('idApto').getValue().substrin
g(0, t));
    }
}
            }
        }],
        buttons: [{
            text: 'Buscar',
            icon : '../../images/buscar1.ico',
            cls:"x-btn-text-icon",
            id:'idAceptar',
            minWidth: 30,
            handler: function(){
                formBuscar.form.submit({
                    url: 'buscarIntegrantesApto',
                    waitMsg: 'Buscando...',
                    waitTitle: 'Informaci&oacute;n',
                    success: function(response, options){
                        if (options.result.id == 0) {
                            Ext.Msg.show({
                                msg: 'Este apartamento no existe en la
UCI',

                                title: 'INFORMACION!!!',
                                icon: Ext.Msg.INFO,
                                buttons: Ext.Msg.OK
                            });

Ext.StoreMgr.get('storeEstApto').removeAll();
                            Ext.getCmp('idApto').setValue('')
                        }
                        else {

Ext.getCmp('idlistaEstApto').getStore().reload({
                                params: {
                                    apto:
Ext.getCmp('idApto').getValue()
                                }
                            });
```

```
      Ext.getCmp('idApartamento').setValue(Ext.getCmp('idApto').getVal
ue());

                                Ext.Msg.show({
                                    msg: 'Se encontraron los siguientes
datos.',
                                    title: 'Informaci&oacute;n',
                                    icon: Ext.Msg.INFO,
                                    buttons: Ext.Msg.OK
                                });
                                formBuscar.form.reset();
                            }
                        },
                        failure: function(){
                            if (Ext.getCmp('idApto').getValue() == '') {
                                Ext.Msg.show({
                                    msg: 'Error !!!. No pueden existir
campos vacios',
                                    title: 'ERROR!!!',
                                    icon: Ext.Msg.ERROR,
                                    buttons: Ext.Msg.OK
                                });
                            }
                            else {
                            if (!formBuscar.form.isValid()){
                                Ext.Msg.show({
                                    msg: 'Debe escribir un apartamento
correctamente',
                                    title: 'ERROR!!!',
                                    icon: Ext.Msg.ERROR,
                                    buttons: Ext.Msg.OK
                                });
                            }
                            else
                            {
                                Ext.Msg.show({
                                    msg: 'Error en la conexi&oacute;n',
                                    title: 'ERROR!!!',
                                    icon: Ext.Msg.ERROR,
                                    buttons: Ext.Msg.OK
                                });
                                }
                            }
                        }
                    });
                }
            }]
    }]
});

Ext.onReady(function(){
    Ext.QuickTips.init();
    var formAddEvaluacion = new Ext.Panel({
        renderTo: document.body,
        id: 'idtop',
        listeners:{
        'render':function(a){
          Ext.getCmp('idApto').focus(true,1)
          }},
        title: 'Evaluaci&oacute;n de Estudiantes',
        frame: true,
```

```
        layout: 'form',
        bodyStyle: 'padding:5px 5px 0',
        buttonAlign: 'center',
        style: ' margin:0 auto',
        width:795,
            height: 550,
        items: [formBuscar, formAgregarEvaluacion]
    });
});
```

Anexo 7: Código fuente de la Interfaz Adicionar Indisciplina

```
function borrarFormBusqueda(){
    Ext.getCmp('nombreEstudiante').setValue('');
    Ext.getCmp('ciEstudiante').setValue('');
    Ext.getCmp('solapinEstudiante').setValue('');
    Ext.getCmp('facultadEstudiante').setValue('');
    Ext.getCmp('sexoEstudiante').setValue('');
    Ext.getCmp('grupoEstudiante').setValue('');
    Ext.getCmp('aptoEstudiante').setValue('');
    Ext.getCmp('idComboCriterio').setValue('');
    Ext.getCmp('idValor').setValue('');
    Ext.getCmp('idDirecResid').setValue('');
    Ext.getDom('foto').setAttribute('src',
'../../images/individuo.JPG');
};

function insertarImplicado(valor){
    var prueba = 0;
    var solapinEst = Ext.getCmp('solapinEstudiante').getValue();
    if (solapinEst == '') {
        Ext.Msg.show({
            msg: 'Error !!!. No puede insertar un estudiante vacio',
            title: 'ERROR!!!',
            icon: Ext.Msg.ERROR,
            buttons: Ext.Msg.OK
        });
    }
    else {
        Ext.getCmp('idlistaEstImp').stopEditing();
        Ext.getCmp('idlistaEstImp').getStore().each(function(item){
            if (item.data.solapin == solapinEst) {
                prueba = 1;
            }
        });
        if (prueba != 1) {
            Ext.getCmp('idlistaEstImp').getStore().insert(0, valor);
            Ext.getCmp('idlistaEstImp').startEditing(0, 2);
            borrarFormBusqueda();
        }
        else {
            Ext.Msg.show({
                msg: 'Error !!!. Este estudiante ya esta insertado en
al Lista',
                title: 'ERROR!!!',
                icon: Ext.Msg.ERROR,
                buttons: Ext.Msg.OK
```

```
            });
        }
    }
};

formAgregar = new Ext.Window({
    title: 'Buscar Estudiante',
    width: 610,
    height: 470,
    y:30,
      closable : false,
    resizable: false,
    layout: 'fit',
    modal: true,
    plain: true,
    bodyStyle: 'padding:5px;',
    buttonAlign: 'center',
    items: [panelBuscar],
    buttons: [{
        text: 'Aceptar',
        icon : '../../images/ok.ico',
         cls:"x-btn-text-icon",
        handler: function(){
            var record = new Ext.data.Record.create([{
                name: 'nombre'
            }, {
                name: 'solapin'
            }, {
                name: 'caracter'
            }, {
                name: 'articuloInd'
            }, {
                name: 'incisoInd'
            }, {
                name: 'articuloMed'
            }, {
                name: 'incisoMed'
            }]);
            var tupla = new record({
                nombre: Ext.getCmp('nombreEstudiante').getValue(),
                solapin: Ext.getCmp('solapinEstudiante').getValue(),
                caracter: '',
                articuloInd: '',
                incisoInd: '',
                articuloMed: '',
                incisoMed: ''
            });
            insertarImplicado(tupla);
            panelBuscar.form.reset();
                  Ext.getDom('foto').setAttribute('src',
'../../images/individuo.JPG');
        }
    },{
        text: 'Cancelar',
        icon : '../../images/editdelete.ico',
         cls:"x-btn-text-icon",
        handler: function(){
            borrarFormBusqueda();
            formAgregar.hide();
        }
    }]
```

```
});

Ext.onReady(function(){

    Ext.QuickTips.init();

    var formAddIndisciplina = new Ext.FormPanel({
        renderTo: document.body,
        id: 'idformulario',
        layout: 'form',
        title: 'Adicionar Indisciplina',
        frame: true,
        bodyStyle: 'padding:5px 5px 0',
        buttonAlign: 'center',
        style: ' margin:0 auto',
        width:790,
            //height: 510,
        items: [{
            xtype: 'fieldset',
            title: 'Datos de la Indisciplina',
            bodyStyle: 'padding:5px 5px 0',
            style: 'margin:0 auto',
            labelAlign: 'right',
            defaultType: 'textfield',
            labelWidth: 140,
            width: 350,
            height: 200,
            defaults: {
                width: 175,
                allowBlank: false,
                blankText: 'Este campo es obligatorio'
            },
            items: [{
                xtype: 'datefield',
                fieldLabel: 'Fecha de la Indisciplina',
                name: 'Dte_fecha',
                id: 'idFecha',
                readOnly: true
            }, {
                xtype: 'combo',
                fieldLabel: 'Direcci&oacute;n Residencia',
                id: 'idComboDirecResid',
                listeners:{
               'select':function(a, b, index){
                        Ext.getCmp('idDescripcion').focus(true, 2)
                }
            },
                name: 'Cbo_direcResid',
                store: new Ext.data.SimpleStore({
                    fields: ['id', 'texto'],
                    data: [['RESIDENCIA #1', 'RESIDENCIA #1'],
['RESIDENCIA #2', 'RESIDENCIA #2'], ['RESIDENCIA #3', 'RESIDENCIA
#3']]
                }),
                valueField: 'id',
                displayField: 'texto',
                hiddenName: 'Cbo_direcResid',
                triggerAction: 'all',
                mode: 'local',
                readOnly: true,
```

```
                emptyText: 'Seleccione...'
            }, {
                xtype: 'textarea',
                fieldLabel: 'Descripci&oacute;n',
                name: 'TxA_descripcion',
                id: 'idDescripcion',
                height: 100
            }]
        }, {
            xtype: 'fieldset',
            title: 'Estudiantes Implicados',
            bodyStyle: 'padding:5px 5px 0',
            style: 'margin-top: 20px',
            labelAlign: 'left',
            labelWidth: 145,
            width: 760,
            height: 312,
            items: [{
                xtype: 'editorgrid',
                enableColumnResize: false,
                id: 'idlistaEstImp',
                clicksToEdit: 2,
                title: 'Lista de Estudiantes Implicados',
                autoScroll: true,
                width: 736,
                height: 270,
                frame: true,
                store: new Ext.data.SimpleStore({
                    id: 'storeImplicado',
                    root: 'implicados',
                    fields: ['nombre', 'solapin', 'caracter',
'articuloInd', 'incisoInd', 'articuloMed', 'incisoMed'],
                    data: []
                }),
                colModel: new Ext.grid.ColumnModel([{
                    header: "Nombre",
                    width: 215,
                    dataIndex: 'nombre'
                }, {
                    header: "Solap&iacute;n",
                    width: 60,
                    align: 'center',
                    dataIndex: 'solapin'
                }, {
                    header: "Car&aacute;cter",
                    width: 82,
                    align: 'center',
                    dataIndex: 'caracter',
                    editor: new Ext.form.ComboBox({
                        valueField: 'id',
                        id: 'idcomboCaracter',
                        displayField: 'caracter',
                        readOnly: true,
                        triggerAction: 'all',
                        mode: 'local',
                        store: new Ext.data.SimpleStore({
                            storeId: 'storeCaracter',
                            fields: ['id', 'caracter'],
                            data: [['Leve', 'Leve'], ['Menos Grave',
'Menos Grave'], ['Grave', 'Grave'], ['Muy Grave', 'Muy Grave']]
                        })
```

```
                    })
                }, {
                    header: "Art&iacute;culo Indisciplina",
                    width: 101,
                    align: 'center',
                    dataIndex: 'articuloInd',
                    editor: new Ext.form.ComboBox({
                        valueField: 'id',
                        displayField: 'articuloInd',
                        readOnly: true,
                        triggerAction: 'all',
                        mode: 'local',
                        store: new Ext.data.SimpleStore({
                            storeId: 'storecomboArticuloInd',
                            fields: ['id', 'articuloInd'],
                            data: [[1, 1], [2, 2], [3, 3], [4, 4], [5,
5], [6, 6], [7, 7], [8, 8], [9, 9], [10, 10], [11, 11], [12, 12], [13,
13], [14, 14], [15, 15], [16, 16], [17, 17], [18, 18], [19, 19], [20,
20], [21, 21], [22, 22], [23, 23], [24, 24], [25, 25], [26, 26], [27,
27], [28, 28], [29, 29], [30, 30], [31, 31], [32, 32], [33, 33], [34,
34], [35, 35], [36, 36], [37, 37], [38, 38], [39, 39], [40, 40], [41,
41], [42, 42], [43, 43], [44, 44], [45, 45], [46, 46], [47, 47], [48,
48], [49, 49], [50, 50], [51, 51], [52, 52], [54, 54], [55, 55], [56,
56], [57, 57], [58, 58], [59, 59], [60, 60], [61, 61], [62, 62], [63,
63], [64, 64], [65, 65], [66, 66], [67, 67], [68, 68], [69, 69], [70,
70], [71, 71], [72, 72], [73, 73]]
                        })
                    })

                }, {
                    header: "Inciso Indisciplina",
                    width: 92,
                    align: 'center',
                    dataIndex: 'incisoInd',
                    editor: new Ext.form.ComboBox({
                        valueField: 'id',
                        displayField: 'incisoInd',
                        readOnly: true,
                        triggerAction: 'all',
                        mode: 'local',
                        store: new Ext.data.SimpleStore({
                            storeId: 'storecomboIncisoInd',
                            fields: ['id', 'incisoInd'],
                            data: [['A', 'A'], ['B', 'B'], ['C', 'C'],
['D', 'D'], ['E', 'E'], ['F', 'F']]
                        })
                    })
                }, {
                    header: "Art&iacute;culo Medida",
                    width: 79,
                    align: 'center',
                    dataIndex: 'articuloMed',
                    editor: new Ext.form.ComboBox({
                        valueField: 'id',
                        displayField: 'articuloMed',
                        readOnly: true,
                        triggerAction: 'all',
                        mode: 'local',
                        store: new Ext.data.SimpleStore({
                            storeId: 'storecomboArticuloMed',
                            fields: ['id', 'articuloMed'],
```

```
                            data: [[1, 1], [2, 2], [3, 3], [4, 4], [5,
5], [6, 6], [7, 7], [8, 8], [9, 9], [10, 10], [11, 11], [12, 12], [13,
13], [14, 14], [15, 15], [16, 16], [17, 17], [18, 18], [19, 19], [20,
20], [21, 21], [22, 22], [23, 23], [24, 24], [25, 25], [26, 26], [27,
27], [28, 28], [29, 29], [30, 30], [31, 31], [32, 32], [33, 33], [34,
34], [35, 35], [36, 36], [37, 37], [38, 38], [39, 39], [40, 40], [41,
41], [42, 42], [43, 43], [44, 44], [45, 45], [46, 46], [47, 47], [48,
48], [49, 49], [50, 50], [51, 51], [52, 52], [54, 54], [55, 55], [56,
56], [57, 57], [58, 58], [59, 59], [60, 60], [61, 61], [62, 62], [63,
63], [64, 64], [65, 65], [66, 66], [67, 67], [68, 68], [69, 69], [70,
70], [71, 71], [72, 72], [73, 73]]
                        })
                    })
                }, {
                    header: "Inciso Medida",
                    width: 75,
                    align: 'center',
                    dataIndex: 'incisoMed',
                    editor: new Ext.form.ComboBox({
                        valueField: 'id',
                        displayField: 'incisoMed',
                        readOnly: true,
                        triggerAction: 'all',
                        mode: 'local',
                        store: new Ext.data.SimpleStore({
                            storeId: 'storecomboIncisoMed',
                            fields: ['id', 'incisoMed'],
                            data: [['A', 'A'], ['B', 'B'], ['C', 'C'],
['D', 'D'], ['E', 'E'], ['F', 'F']]
                        })
                    })
                }]),
                tbar: [{
                    text:'Agregar Implicados',
                    icon : '../../images/edit_add.ico',
                    cls:"x-btn-text-icon",
                    //disabled:true,
                    handler: function(){
                    if(formAddIndisciplina.form.isValid()){
                        formAgregar.show();
                    }
                    else{
                    Ext.Msg.show({
                               msg: 'No pueden existir campos
vacios',
                               title: 'ERROR!!!',
                               icon: Ext.Msg.ERROR,
                               buttons: Ext.Msg.OK
                           });
                    }
                    }
                }, '-', {
                    text: 'Eliminar Implicados',
                    //disabled:true,
                    icon : '../../images/delete.ico',
                    cls:"x-btn-text-icon",
                    handler: function(){
                        var seleccion =
Ext.getCmp('idlistaEstImp').getSelectionModel().getSelectedCell();
                        if (seleccion != null) {
                            var seleccionado =
```

```
Ext.getCmp('idlistaEstImp').getStore().getAt(seleccion[0]);

Ext.StoreMgr.get('storeImplicado').remove(seleccionado);
                        }
                        else{
                        Ext.Msg.show({
                                    msg: 'Informaci&oacute;n !!!. Debe
marcar el elemento a eliminar',
                                    title: 'Informaci&oacute;n!!!',
                                    icon: Ext.Msg.INFO,
                                    buttons: Ext.Msg.OK
                                });
                        }
                    }
                }, '-']
            }]
        }],
        buttons: [{
            text: 'Aceptar',
            icon : '../../images/ok.ico',
            cls:"x-btn-text-icon",
            handler: function(){
                var valido = false;
                var cont =
Ext.StoreMgr.get('storeImplicado').getCount();
                if (cont != 0) {
                    valido = true;

Ext.StoreMgr.get('storeImplicado').each(function(value){
                        if (value.data.caracter == '' ||
value.data.articuloInd == '' || value.data.incisoInd == '') {
                            valido = false;
                        }
                        if (value.data.caracter == "Leve" &&
(value.data.articuloMed == '' || value.data.incisoMed == '')) {
                            valido = false;
                        }
                    });
                }
                if (valido) {
                    var array = [];

Ext.StoreMgr.get('storeImplicado').each(function(){
                        array.push(this.data);
                    });
                    formAddIndisciplina.form.submit({
                        url: 'InsertarIndisciplina',
                        waitMsg: 'Guardando...',
                        waitTitle: 'Espere por favor',
                        params: {
                            implicados: Ext.encode(array)
                        },
                        success: function(param, valor){
                            Ext.Msg.show({
                                msg: 'Se ha registrado la Indisciplina
No : ' + valor.result.id,
                                title: 'Informaci&oacute;n',
                                icon: Ext.Msg.INFO,
                                buttons: Ext.Msg.OK
                            });
                            formAddIndisciplina.form.reset();
```

```
Ext.StoreMgr.get('storeImplicado').removeAll();
                        },
                        failure: function(){
                            if (Ext.getCmp('idFecha').getValue() == ''
|| Ext.getCmp('idDescripcion').getValue() == ''|| Ext.getCmp(
'idComboDirecResid').getValue()=='') {
                                Ext.Msg.show({
                                    msg: 'Error !!!. No pueden existir
campos vacios',
                                    title: 'ERROR!!!',
                                    icon: Ext.Msg.ERROR,
                                    buttons: Ext.Msg.OK
                                });
                            }
                            else {
                                Ext.Msg.show({
                                    msg: 'Error en la
conexi&oacute;n',
                                    title: 'ERROR!!!',
                                    icon: Ext.Msg.ERROR,
                                    buttons: Ext.Msg.OK
                                });
                            }
                        }
                    });
                }
                else {
                    Ext.Msg.show({
                        msg: 'Error !!!. No pueden existir campos
vacios',
                        title: 'ERROR!!!',
                        icon: Ext.Msg.ERROR,
                        buttons: Ext.Msg.OK
                    });
                }
            }
        }]
    });
});
```