Universidad Autónoma Metropolitana
Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Ingeniería en Computación

Reporte del Proyecto Terminal

"Aplicación de recomendación de inscripción para Ingeniería Mecánica"

Alumno:

Morales Miranda Julio César

Matrícula:

208300498

Trimestre 13-O

Asesores del proyecto:

Hilario Terres Peña
Profesor Titular
Departamento de Energía

Sergio Luis Pérez Pérez
Profesor Titular
UAM Cuajimalpa

Tabla de contenido

Índice de figuras

Índice de cuadros

Universidad Autónoma Metropolitana
Azcapotzalco

AUTORIZACIÓN DE INSCRIPCIÓN A PROYECTO DE INTEGRACIÓN EN INGENIERÍA COMPUTACIÓN

CBI DIVISIÓN CIENCIAS BÁSICAS E INGENIERÍA
UAM - Azcapotzalco

Trimestre en que se autoriza la propuesta: 2013P fecha: 10/07/13 PI-A-COM 2013P-40

DATOS DEL ALUMNO

Firma:

Nombre: Julio César Morales Miranda Matrícula: 208300496

Correo alterno: jcmm_uam-a@hotmail.com

ASESOR RESPONSABLE

Firma:

Nombre del asesor: Hilario Terres Peña

No. económico: 24990 Adscripción: Departamento de Energía

Área de investigación: Termofluidos

Correo institucional: tph@correo.azc.uam.mx

COASESOR O ASESOR EXTERNO

Firma:

Nombre del asesor: Sergio Luis Pérez Pérez

No. económico: 31881 Adscripción: UAM Cuajimalpa

Área de investigación:

Correo electrónico: sergio10barca@yahoo.com.mx

MODALIDAD DEL PROYECTO

(•) Proyecto tecnológico ( ) Estancia industrial ( ) Estancia de investigación ( ) Experiencia profesional

TÍTULO DEL PROYECTO

Aplicación de recomendación de inscripción para Ingeniería Mecánica

OBJETIVO GENERAL

Diseñar e implementar una aplicación para dispositivos móviles que le pueda recomendar a un alumno de Ingeniería Mecánica las UEA que debería inscribir en su próximo trimestre

UNIDADES DE ENSEÑANZA APRENDIZAJE QUE SE AUTORIZAN

Clave	UEA	Trimestre de inicio de vigencia	Trimestre de fin de vigencia
1100113	Proyecto de Integración en Ingeniería en Computación I	2013O	2014P
1100123	Proyecto de Integración en Ingeniería en Computación II	2013O	2014P
1100133	Introducción al Trabajo de Investigación en Ingeniería en Computación		
1151028	Trabajo de Investigación en Ingeniería en Computación		

Nombre y firma del Coordinador de Estudios

Dr. Francisco Javier Zaragoza Martínez

UNIVERSIDAD AUTÓNOMA METROPOLITANA Casa Abierta al Tiempo Azcapotzalco
CIENCIAS BÁSICAS E INGENIERÍA
10 JUL 2013
COORDINACIÓN DE ING. EN COMPUTACIÓN
UNIDAD AZCAPOTZALCO

Revisión 01 Julio 2013

Resumen

A lo largo de este reporte se presentan los detalles de elaboración de la "Aplicación de recomendación de inscripción para Ingeniería Mecánica" de la Universidad Autónoma Metropolitana Unidad Azcapotzalco. Dicha aplicación fue creada para orientar a los alumnos de Ingeniería Mecánica en la elección de las UEA que trimestre a trimestre inscriben, puesto que al llevar a cabo esta elección no consideran la carga de trabajo acumulada con cada UEA [1]. La aplicación permite generar una recomendación, con base en el kardex, de las UEA que se deben cursar el siguiente trimestre. El kardex es obtenido del Subsistema de Administración Escolar (SAE)[1] y del plan de estudios correspondiente. Cabe resaltar que dicho plan corresponde al plan que entró en vigor el trimestre 2013 Otoño, también se hace la recomendación siguiendo el orden de importancia de cada UEA dado según el Coordinador de la carrera.

Además se detalla el diseño y la implementación de cada uno de los módulos que conforman la aplicación así como el detalle del icono representativo de ésta. Finalmente se anexa el diagrama de seriación correspondiente a la licenciatura, el cual fue la base para el modelado del plan de estudios además de los diagramas de seriación correspondientes a cada área de concentración de la carrera.

[1] Unidad de Enseñanza Aprendizaje

1. Introducción

Actualmente los alumnos de la UAM Azcapotzalco eligen de forma libre las UEA que desean cursar en cada trimestre dentro de su estadía en la universidad, de tal forma que al tomar dicha decisión éstos no consideran la carga de trabajo que generan con cada una de las UEA que inscriben, tomando únicamente a consideración los créditos máximos permitidos para su inscripción, créditos que dependen de distintas condiciones tales como: sí el alumno renuncio durante el trimestre anterior a alguna UEA, en cuyo caso es probable que el alumno pudiera haber ocupado el lugar que otro alumno habría aprovechado de mejor forma. Además de lo anterior los alumnos también inscriben las UEA contemplando el profesorado que impartirá ésta en dicho trimestre.

Con la situación anteriormente descrita, lo único que provoca el alumno es que su avance no sea tan rápido o dinámico es decir, concluir la carrera en el número de trimestres considerado como normal por la universidad (12 trimestres) o por el contrario su estancia en la universidad sea lo más corta posible, puesto que no cuentan con una herramienta que los oriente en la elección de cada UEA, ya que estos únicamente consultan la programación de horarios del siguiente trimestre en el SAE sin obedecer criterios tales como: sí las UEA pertenecen a una cadena larga dentro del plan de estudios, entre otros.

Sin embargo, aunque se tiene registro de proyectos similares al descrito en este documento, debido a diferentes circunstancias no tuvieron la aceptación que se hubiera esperado. Estos proyectos son: "Tutor Virtual para alumnos de la carrera de Ingeniería en Computación" [3] y "Sistema Web de seriación e información para alumnos de ingeniería" [4], ambos trabajos son Proyectos Terminales desarrollados para la Universidad Autónoma Metropolitana Unidad Azcapotzalco. El primer proyecto tiene por objetivo generar una recomendación de las UEA que un alumno de Ingeniería en Computación debe inscribir en su próximo trimestre, a través de su historia académica y las horas disponibles para su estudio. Sin embargo, para poder generar esta recomendación es necesario recibir varias entradas, además de un registro previo por parte del alumno.

Por otro lado el segundo proyecto permite visualizar el diagrama de seriación del alumno, de tal manera que éste observa las UEA que puede inscribir y además de las UEA que son de gran importancia para evitar un retraso en su estancia dentro de la UAM Azcapotzalco sin embargo, aunque nuevamente se pide un registro previo por parte del alumno pero no obligatorio, de forma implícita sugiere las UEA que deben tener una prioridad mayor de inscripción (visualización de las UEA que son de gran importancia).

Con el objetivo de que la aplicación, la cual tiene por nombre **Mapp (MecánicaApp)**, se encuentre disponible para cualquier alumno, ésta fue diseñada y desarrollada para tecnología de dispositivos móviles, asumiendo que en la actualidad el crecimiento de esta tecnología

permite que cualquier persona cuente con un dispositivo móvil pasando un tiempo considerable en ellos. De forma similar al Tutor Virtual, la generación de la recomendación se realiza utilizando el plan de estudios de la licenciatura y el kardex del alumno, sin embargo la aplicación también utiliza la sección de "Información Académica", la cual se encuentra disponible al igual que la sección de "Kardex" en el SAE.

Con la finalidad de evitar que el alumno tenga que indicar cada una de las UEA aprobadas o cargar su kardex a la aplicación, se implementó un algoritmo de tal forma que con éste se estableciera una conexión con el SAE y posteriormente llevar a cabo la recuperación de las secciones de "Kardex" e "Información Académica". Por otro lado la aplicación como ya fue mencionado estará disponible para dispositivos móviles, sin embargo estará disponible únicamente para dispositivos que cuenten con sistema operativo Android.

La decisión de emplear Android para su desarrollo fue que actualmente este sistema domina más del 70 % del mercado de dispositivos moviles, dispositivos que van desde teléfonos celulares hasta tabletas electrónicas. Así también Mapp puede ser instalada en dispositivos que cuenten con versiones Android 2.3 hasta Android 4.2. La razón por la cual Mapp solo puede ser utilizada en este intervalo de versiones de Android, se debe que para versiones inferiores existen inconvenientes para el establecimiento de la conexión.

Los objetivos cumplidos en este proyecto son:

1.1. Objetivo general

Diseñar e implementar una aplicación para dispositivos móviles que le pueda recomendar a un alumno de Ingeniería Mecánica las UEA que debería inscribir en su próximo trimestre.

1.2. Objetivos particulares

- Implementar un algoritmo que establezca una conexión con el módulo de información escolar a partir de los datos de acceso de un alumno.
- Extraer la información necesaria del kardex de un alumno mediante un análisis sintáctico de la información devuelta por el módulo de información escolar.
- Diseñar e implementar un algoritmo que le sugiera al alumno que UEA inscribir el próximo trimestre en base a la información obtenida de su kardex y al plan de estudios correspondiente.

Estructura del documento

El resto del documento se encuentra estructurado de la siguiente manera: En la segunda sección se explicará el diseño de la interfaz de Mapp, así como los detalles del diseño del icono que la representa. En la tercera sección se explicará el diseño lógico de Mapp es decir, el mdelado del plan de estudios y el kardex, los detalles del algoritmo del establecimiento de la conexión con el SAE así como los algoritmos necesarios para el análisis de la información del alumno. En la cuarta sección se explicarán los criterios para la determinación de la recomendación de las UEA. Finalmente en la quinta sección se dará una conclusión sobre el proyecto realizado.

2. Diseño de la interfaz de Mapp

Con la finalidad de que Mapp contara con un diseño original, todo el diseño gráfico de su interfaz fue diseñado de tal forma que ésta representará de la forma más adecuada tanto a la Ingeniería Mecánica como a la unidad de la universidad. Tanto el icono así como cada una de las interfaces desplegadas se diseñaron bajo el hecho anterior y se mantuvo una línea constante en la combinación de colores en cada una de éstas. A continuación se presentan los detalles de cada uno de los elementos que conforman gráficamente la aplicación.

2.1. El icono de Mapp

El diseño del icono de la aplicación se elaboro desde un inicio siguiendo lo anteriormente mencionado, ya que este debería de representar de la mejor manera a la carrera es decir, la Ingeniería Mecánica. Tras una investigación y la realización de una encuesta a alumnos de la licenciatura en cuestión, se llegó a la conclusión que un "engrane" sería el elemento que mejor distingue a la Ingeniería Mecánica.

Sin embrago aún con esto se contaba con un inconveniente más, ya que la mayoría de los dispositivos móviles acostumbran representar algunas de sus aplicaciones con un engrane, como la opción de **"Herramientas"** o la opción de ***"Configuración"***. Tras este hecho el icono de Mapp se diseñó de tal forma que no se incurriera en un hecho de plagio y la aplicación se logrará identificar fácilmente. El icono de Mapp contiene básicamente tres aspectos principales dentro de su diseño, los cuales se describen a continuación.

1. El **color**, este icono presenta solo tres colores y cuyo color principal es el rojo, representando la unidad de la universidad para la cual fue desarrollada y en consecuencia solo alumnos pertenecientes a la carrera y unidad que podrán hacer uso de ésta.

2. El **engrane**, elemento que mejor representa la Ingeniería Mecánica.

3. La **tipografía**, en este caso además de incluir la leyenda de INGENIERÍA MECÁNICA, también se incluye el logotipo de la universidad identificando así a la institución para la cual fue desarrollada la aplicación.

Figura 1: Icono de Mapp

Cabe señalar que el icono fue desarrollado bajo las normas que Android tiene a su disposición para los desarrolladores de aplicaciones [5], así como de la pertinente herramienta para su creación [6].

2.2. Interfaz gráfica de Mapp

Esencialmente la ubicación donde Mapp será colocada tras su instalación, es básicamente la ruta en que buscamos las demás aplicaciones que nuestro dispositivo trae por defecto. En esta ruta encontraremos el icono antes mencionado y tras pulsar este se desplegará la interfaz de inicio *ver figura 3*. Cabe mencionar que todas las imágenes que muestran el diseño, pertenecen a imágenes de una tableta de 7 pulgadas donde fue instalada y probada Mapp.

Figura 2: Ubicación de Mapp

La pantalla de inicio al igual que el icono de la aplicación consta de elementos que hacen que pueda ser identificada fácilmente. En esta pantalla inicial se observa en la parte superior derecha el nombre desglosado de Mapp (MECÁNICAapp), en la parte inferior derecha el

elemento que mejor representa a la Ingeniería en cuestión, el engrane. También se encuentran tanto el nombre de la universidad y el logotipo de la misma, finalmente la combinación de colores que representan a la unidad de ésta misma.

Figura 3: Pantalla de inicio de Mapp

En esta pantalla inicial se encuentran los campos que permiten introducir los únicos datos que el alumno requiere para generar la recomendación, campos que corresponden a la Matrícula y Contraseña, información que corresponde a los datos con que ingresa el alumno al SAE *ver figura 3*. Debido a que Mapp establece una conexión con el SAE para generar la recomendación se debe contar con acceso a Internet, ya que si no es así Mapp desplegará un mensaje de error notificando dicho problema *ver figura 4*.

Figura 4: Notificación sobre conexión a Internet

Dentro de la aplicación son mostrados diferentes avisos como en el caso anterior. Si la matrícula o la contraseña introducidas no cumplen las especificaciones establecidas por el SAE, también se desplegarán las alertas o avisos correspondientes *ver figuras 5 y 6*.

Figura 5: Notificación del formato de la matrícula

Figura 6: Notificación del formato de la contraseña

Las especificaciones básicamente son las siguientes:

a. La matrícula debe ser una serie de dígitos entre 0-9 con una longitud de 9 o 10 dígitos respectivamente.

b. La contraseña debe ser una palabra de 10 a 15 caracteres que deberá contener al menos una letra mayúscula, una letra minúscula, un dígito y no debe contener tildes.

En caso de introducir los datos correctamente, la aplicación realiza una verificación más, ya es probable que los datos de ingreso sean incorrectos a pesar de tener el formato adecuado *ver figura 7.*

Figura 7: Notificación de datos de ingreso incorrectos

La anterior notificación se presenta debido a que el SAE devuelve un error tras un intento fallido del establecimiento de la conexión. Como ya fue mencionado en una sección anterior, Mapp sólo podrá utilizarse por alumnos que pertenezcan a Ingeniería Mecánica, por lo que si un alumno no perteneciente a ésta trata de ingresar, Mapp le notificará que no es alumno de dicha carrera *ver figura 8*. Para validar este hecho es necesario que se establezca la conexión y se recupere la sección de "Información Académica", la cual contiene la carrera a la cual pertenece.

Si el alumno que ingresa pertenece a la carrera y además ingresa sus datos de forma correcta, podrá generar la recomendación *ver figuras 9 y 10*.

Figura 8: Notificación de pertenencia a la carrera

Figura 9: Generación de recomendación

Figura 10: Despliegue de recomendación

Además también Mapp contiene una sección donde el alumno puede consultar las UEA optativas tanto del tronco Inter - Multidisciplinar, así como las de Integración *ver figuras 11 y 12*

Figura 11: UEA optativas Inter - Multidisciplinares

Figura 12: UEA optativas de Integración

Con el objetivo de enriquecer el diseño y evolución de Mapp, se implementó una opción que permite al usuario enviar comentarios acerca de lo que le parece ésta. Comentarios que se hacen llegar al Coordinador de Estudios y que se notificarán a su vez al desarrollador de Mapp para una posible evolución de la misma. Dichos comentarios se envían en un mensaje de correo electrónico, por lo que es necesario seleccionar la aplicación de correo electrónico de su preferencia *ver figura 13*.

La primer figura contiene básicamente un área de texto, en la cual el usuario introduce el mensaje o comentario hacia el coordinador. El correo del coordinador se encuentra incluido dentro del código de la lógica de esta interfaz, por lo que el usuario únicamente se preocupa por introducir el comentario.

Una vez que el usuario introduce su comentario, Mapp solicita seleccionar la aplicación de correo electrónico como hotmail o gmail *ver figura 14*, tras dicha elección aparece la interfaz de la aplicación de correo electrónico elegido *ver figura 15*, donde el alumno introducirá su cuenta de correo electrónico o esta aparecerá por defecto si dentro del dispositivo se encuentra agregada o almacenada una cuenta de correo electrónico de forma predeterminada, el alumno también puede modificar su comentario si así lo desea y posteriormente enviarlo.

Figura 13: Interfaz de envió de comentarios

Figura 14: Seleción del cliente de correo electrónico

Figura 15: Envio de e-mail

3. Diseño lógico de Mapp

3.1. Modelado del plan de estudios y del kardex

Para generar la recomendación de inscripción, Mapp hace uso tanto de la información almacenada en el kardex así como del plan de estudios, el cual refiere a la versión que entró en vigor el trimestre de 2013 Otoño. Para poder hacer uso de éste se tuvo que modelar de forma tal, que éste contuviera toda la información necesaria. La información que contiene es la siguiente: clave UEA, nombre UEA, créditos, tronco disciplinar, modalidad de UEA (Optativa u Obligatoria), oportunidades en evaluación global, oportunidades en evaluación de recuperación, seriación y prioridad dentro del plan de estudios. La prioridad fue definida bajo el criterio del coordinador de la carrera.

Clave	UEA	Cred.	Tronco	Mod.	Ev. Gl.	Ev. Rec.	Seriación	Prioridad
1100021	Optativa Técnica de Movilidad I	3	TI	OPT	0	0	240 CRÉDITOS,AUTORIZACIÓN	163
1100022	Optativa Técnica de Movilidad II	3	TI	OPT	0	0	240 CRÉDITOS,AUTORIZACIÓN	164
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
1201008	Comprensión de Textos	4	TNA	OBL	2	3	0	3

Cuadro 1: Modelado del Plan de Estudios de Ingeniería Mecánica

Además el plan fue ordenado por clave de UEA ascendentemente como se indica en el Cuadro 1. Bajo este ordenamiento se logró que la búsqueda de cada dato necesario para generar la recomendación se llevará de manera eficiente, pues la búsqueda de cada dato se realiza de forma binaria a través de la clave. El modelado completo del plan de estudios se puede ser consultado en: http://www.ingenieriamecanica/plan_estudios.com

El kardex al igual que el plan de estudios fue almacenado en una estructura de datos y cuya información quedo organizada de la siguiente forma:

Clave	UEA	Trimestre	Tipo Eval.	Calificación
1113084	ESTRUCTURA ATOMICA Y ENLACE QUIMICO	07O	GLO	EQ
1113085	LABORATORIO DE REACCIONES QUIMICAS	07O	GLO	EQ
...	...	...	...	...
...	...	...	...	...
1154045	SEGURIDAD E HIGIENE INDUSTRIAL	13P	GLO	EQ

Cuadro 2: Modelado del Kardex del alumno

3.2. Conexión con el SAE

Para realizar la conexión con el SAE y poder recuperar las secciones necesarias para generar la recomendación, Android proporciona dos API's por un lado nos ofrece la API ***HttpURLConnection***, basada en la API estándar de Java y la cual se recomienda para proyectos donde se desea obtener recursos web como en este caso. Pero por otro lado también existe la API ***ApacheHttpClient*** la cual es propia de Android. Aunque la más recomendable es *HttpURLConnection*, se optó en este caso en hacer uso de *ApacheHttpClient* debido a la eficiencia que se tendría una vez que se estableció dicha conexión, eficiencia que principalmente refiere al tiempo para el establecimiento de la conexión y la recuperación de la secciones antes mencionadas.

La secuencia de código para esta tarea se muestra a continuación.

```
1 HttpClient httpclient = new DefaultHttpClient();
2 HttpPost httpPost = new HttpPost("https://ayamictlan.uam.mx:8443/sae/azc/AEWBU004.
      oInicioSesion?mod=1");
```

En estas dos líneas se instancia el objeto HttpClient y posteriormente es indicada la dirección del sitio con el cual se pretende establecer conexión. Sin embargo, antes de ser establecida esta conexión se deben enviar ciertos parámetros adicionales a la matrícula y contraseña del usuario, parámetros que se envían a través de una lista en donde cada parámetro está compuesto de la variable y el valor de la misma.

```
1  List<NameValuePair> params1=new ArrayList<NameValuePair>();
2  params1.add(new BasicNameValuePair("SIGLAS_UNI_XX.E_UNIDAD.AE02.1","AZC"));
3  params1.add(new BasicNameValuePair("%23.E_UNIDAD.AE02.1","AxJDMQ%3D%3D"));
4  params1.add(new BasicNameValuePair("%23CRC.E_UNIDAD.AE02.1","00000024"));
5  params1.add(new BasicNameValuePair("NOMBRE.IDENTIFICACION.NONMODELED",matricula));
6  params1.add(new BasicNameValuePair("COMPLEMENTO.IDENTIFICACION.NONMODELED",password));
7  params1.add(new BasicNameValuePair("GO.IDENTIFICACION.NONMODELED","Entrar"));
8  params1.add(new BasicNameValuePair("%25.IDENTIFICACION.NONMODELED",""));
9  params1.add(new BasicNameValuePair("%23.WEB_INFO.SW01",""));
10 params1.add(new BasicNameValuePair("%23.WEB_MOD_ASO.SW01",""));
11 params1.add(new BasicNameValuePair("%23.USUARIO_ANEXO.SG02",""));
12 params1.add(new BasicNameValuePair("%23.MODULO_UWAS.SAE01",""));
```

Para llevar a cabo la ejecución de la conexión, se hizo uso del método *execute* proporcionado por la clase *HttpResponse* como se indica a continuación.

```
1     HttpResponse response = httpclient.execute(httpPost);
```

3.3. Recuperación de "Kardex" e "Información Académica"

Una vez establecida la conexión fue necesario mantener activa la sesión mientras se recuperaban ambas secciones, mediante dos peticiones al SAE. Para realizar este proceso fue necesario recuperar la información de respuesta que el SAE regresa una vez establecida la conexión y así mantener activa la sesión.

Una vez recuperada la respuesta fue necesario añadir dicha información de respuesta a cada una de las direcciones en las que se encuentra almacenado el contenido de cada una de las secciones requeridas. Para lograr esto se tuvieron que construir cada una de las peticiones, una vez realizado esto se enviarian al SAE de forma similar a como se estableció la conexión. Posteriormente se tuvo que recibir la respuesta del servidor, quedando recuperadas cada una de las secciones. Esta serie de pasos se muestra a continuación:

```
1 HttpGet httpget = new HttpGet("https://ayamictlan.uam.mx:8443/sae/azc/IEWBC020.oConsulta");
2
3   for (Header h : cookies)
4   httpget.addHeader("Cookie", h.getValue());
5
6   HttpResponse responseGet = null;
7   responseGet = httpclient.execute(httpget);
8   HttpEntity entrada=responseGet.getEntity();
9   kardex=EntityUtils.toString(entrada);
```

En la primer línea se indica la dirección donde se localiza la sección de "Kardex", dirección a la cual es indispensable añadir la respuesta del servidor una vez que se establece la conexión, la cual permite mantener la conexión activa. realizandose este proceso en las líneas 3 y 4. Posteriormente en las líneas 7 y 8 se lleva a cabo petición para recuperar el kardex. En la línea 9 es preparado el proceso para recibir la respuesta del servidor y posteriormente almacenar la sección correspondiente, proceso que se observa en la línea 10. Este conjunto de pasos fueron replicados de igual forma para recuperar la sección de "Información Académica", indicando la correspondiente dirección donde se encuentra ésta.

Finalmente una vez recuperadas ambas secciones, se procedio a cerrar la conexión.

3.4. Implementación del análisis sintáctico

En la sección anterior se describió de forma breve el proceso para el establecimiento de la conexión y con la cual fueron recuperados los archivos necesarios para generar la recomendación, sin embargo, al recuperarlos éstos incluyen código e información innecesaria, por lo que

en un segundo proceso se tuvo que implementar un análisis sintáctico (*parser*) para extraer únicamente la información útil, información que corresponde básicamente a la estructura del kardex y la debida información para realizar el análisis completo de la situación del alumno y así generar la recomendación, infromación como: ¿renuncio el alumno a UEA?, ¿inscribió créditos en evaluación de recuperación?, etc. Lo anterior sería recuperado de la sección de "Información Académica".

Para la implementación de dicho análisis, se hizo uso de una librería que permite la implementación del análisis sintáctico en pocos pasos. Aunque existen diferentes librerías que permiten realizar este proceso, por la eficiencia y el buen desempeño con Android se utilizó la librería ***Jsoup*** (para más información se puede visitar el sitio oficial: `http://jsoup.org/`)

A continuación se muestran los principales pasos de la implementación del análisis sintáctico.

```
1   Document doc = Jsoup.parse(kardex);
2   Elements ElementoHtml = doc.select("div#div_barra4");
3   Elements EtiquetaHtml = ElementoHtml.select("td");
4
5     for(int i = 0; i < EtiquetaHtml.size();++i)
6     {
7       Element row = EtiquetaHtml.get(i);
8       kardex[mi_indice][0] = row.text();
9       mi_indice++;
10    }
11    .
12    .
13    .
```

Como vemos en el código anterior, la línea 1 instancia el objeto al cual se le proporciona la sección correspondiente, en este caso el kardex. En la línea 2 es seleccionada la etiqueta <**div**> la cual contiene la tabla del kardex, posteriormente en la línea 3 seleccionamos las etiquetas <**td**> que contienen cada uno de los registros del kardex. A través de iteraciones, los datos son seleccionados y almacenados en un arreglo bidimensional (líneas 5-10), a partir de la línea 11 en adelante se implementan las demás iteraciones necesarias para extraer los datos restantes del kardex.

Ahora bien, para un mejor manejo de la información durante el proceso de la recomendación el arreglo anterior fue transformado en una estructura de mejor desempeño (matriz de listas), como se muestra a continuación:

```
1     for(int j = 0; j < mi_indice; j++)
2     {
3       CopiaKardex.add(new ArrayList());
4       for(int t = 0; t < 5; t++)
5       {
6         ((ArrayList)CopiaKardex.get(j)).add(kardex[j][t]);
7       }
8
9     }
```

Con el procedimiento anterior se implementó y modelo en una estructura el kardex del alumno. De igual forma fue implementado el análisis de la información académica (Renuncia

a UEA, Créditos en Evaluación de Recuperación y UEA Autorizadas), dichos datos fueron útiles para determinar el límite de créditos permitidos al alumno para su inscripción, límite impuesto por la universidad. Dicho límite puede ser de 40 créditos (normal) si el alumno renunció a una o más UEA en el trimestre anterior y 63 créditos (máximo) si no ocurrió la anterior situación.

Sin embargo, los anteriores datos no determinan en su totalidad el límite ya que existen más reglas que impone la universidad, ofreciendo al alumno poder solicitar un exceso de créditos. Dichas reglas están establecidas en el **Artículo 22 fracción V** del Reglamento de Estudios Superiores de la universidad [7] .

Como fue mencionado y mostrado en la sección 2 del documento, únicamente los alumnos de Ingeniería tienen acceso a generar la recomendación, lo cual antes de realizar el análisis anterior y construir las estructuras correspondientes, la sección de la Información Académica es enviada a un proceso que se encarga de realizar dicha validación, validación que se hace únicamente corroborando que dentro de esta se encuentre la subcadena "LICENCIATURA EN INGENIERIA MECANICA" o "Licenciatura en Ingeniería en Mecánica".

3.5. Determinación de las UEA viables a inscribir

Para determinar el conjunto de UEA que el alumno tiene habilitadas, son eliminadas del plan de estudios aquéllas UEA que ya fueron aprobadas así como las que ya no es posible inscribir es decir, las UEA que ya fueron reprobadas dos veces en evaluación global (clase frente a un profesor) y el plan de estudios especifica un máximo de dos oportunidades en esta modalidad. Una conformada la anterior estructura, las UEA posibles para poder ser inscritas por el alumno se determinaron siguiendo una serie de requisitos impuestos por el coordinador de la carrera, requisitos que se clasifican en cuatro tipos:

- Seriación de UEA (1112030)
- Correlación con UEA (C1124001)
- Autorización (1100116 este tipo de UEA necesitan ser autorizadas por el coordinador para ser cursadas).
- Cantidad de créditos (Este requisito varia de acuerdo a cada UEA según sea el caso)

Dentro del proceso de la determinación de las UEA viables que el alumno tiene para inscribir, son revisados cada uno de los requisitos anteriores para conocer si se han cumplido o no, para cada uno de ellos se realiza una búsqueda diferente. Para el caso de que la UEA necesite Autorización dicha información fue extraída de la sección de Información Académica obtenida del SAE, donde se encuentra una sección destinada a estas Autorizaciones y de la cual es obtenida únicamente las claves de las UEA que han sido autorizadas para posteriormente ser corroboradas cada que se encuentre este tipo de requisito.

Si el requisito de la UEA es una seriación, simplemente se busca esa clave en el kardex, si se encuentra la UEA, entonces es viable para su inscripción y su clave es agregada en una lista, en caso de que el requisito se trate de una correlación entonces la clave es buscada en el kardex y en caso de ser encontrada se tiene que la UEA es viable para su inscripción. Es posible que ésta no se encuentre ahí dándose el caso de que se encuentre en la lista de las UEA que ya han sido determinadas como viables, por lo que se lanza una nueva búsqueda en dicha lista, si se encuentra ahí la UEA, entonces está disponible y la clave de esta es agregada en la lista si no se encuentra en ninguna de las dos búsquedas, entonces la UEA no está disponible y no es agregada a la lista.

Si el requisito de la UEA se trata de una cantidad de créditos acumulados para poder cursarla, simplemente se hace una comparación de los créditos que requiere la UEA y los créditos acumulados del alumno, los cuales son obtenidos al verificar cada una de las UEA aprobadas en su kardex y buscando los créditos de estas en el plan de estudios. Finalmente hay UEA que no requieren requisito alguno para poder ser cursadas, tal es el caso de las UEA pertenecientes al Tronco de Nivelación Académica dando por habilitadas dichas UEA.

A continuación se muestran los algoritmos para la realización de las anteriores tareas:

Algoritmo 1 UEA posibles a inscribir

Entrada: plan_estudios, claves_uea_aprobadas, creditos_acumulados, autorizaciones.
Salida: Una lista 'posibles'con las claves de las UEA hábiles.

1: $i \leftarrow 0, tokens\_maximos \leftarrow 0, claveaut \leftarrow 0$
2: $seriacion \leftarrow$ "", $requisito \leftarrow$ "", $posible \leftarrow$ ""
3: *posibles* $\leftarrow$ lista con las claves de las UEA que el alumno tiene disponibles para su inscripción
4: *posible* $\leftarrow false$, variable que indica si la UEA se puede cursar o no
5: **mientras** i < longitud_del_plan_estudios **hacer**
6: *seriacion* $\leftarrow$ cadena de requisitos para cursar la UEA
7: Obtener requisito por requisito
8: *claveaut* $\leftarrow$ clave de la UEA de la cual se verifican sus requisitos para ser cursada
9: **mientras** tokens_maximos < 4 **hacer**
10: *requisito* $\leftarrow$ primer requisito solicitado por la UEA en cuestión
11: *posible* $\leftarrow$ puede_cursarla(requisito,claves_uea_aprobadas, creditos_acumulados, posibles, autorizaciones, claveaut)
12: **si** posible es **falso entonces**
13: $tokens\_maximos \leftarrow 4$
14: **si no**
15: $tokens\_maximos \leftarrow tokens\_maximos + 1$
16: **fin si**
17: **fin mientras**
18: **si** posible es **cierto entonces**
19: $tokens\_maximos \leftarrow 0$
20: *posibles* $\leftarrow$ clave de la UEA viable a inscripción
21: **si no**
22: $tokens\_maximos \leftarrow tokens\_maximos\ 0$
23: **fin si**
24: $i \leftarrow i + 1$
25: **fin mientras**
26: **devolver** posibles

El algoritmo 1 separa cada requisito y lo clasifica en cada tipo y posteriormente lo envía al algoritmo 2 para su verificación.

Algoritmo 2 Verificar requisito

Entrada: requisito,uea_aprobadas, creditos_acumulados, posibles, autorizaciones, clave_autorizada.

Salida: **cierto** si el requisito se cumple o **falso** en caso contrario.

```
1: cumple ← falso
2: clave ← 0
3: creditos_necesarios ← 0
4: si el requisito es un numero de creditos_necesarios entonces
5:    si creditos_acumulados >= creditos_necesarios entonces
6:       cumple es cierto
7:    si no
8:       cumple es falso
9:    fin si
10: si no, si requisto es un corregistro (comienza con 'C') entonces
11:    clave ← clave del corregistro
12:    cumple = isposible(clave,uea_aprobadas)
13:    si cumple es falso  entonces
14:       cumple = isposible(clave,posibles)
15:    fin si
16: si no, si requisito es AUTORIZACION entonces
17:    clave_autorizada ← clave de la UEA autorizada
18:    cumple = verifica_autorizacion(clave_autorizada,autoizaciones)
19: si no, si requisito es 0 entonces
20:    cumple es cierto
21: si no
22:    cumple = isposible(clave,uea_aprobadas)
23: fin si
24: devolver  cumple
```

El algoritmo 2 verifica cada uno de los tipos de requisitos mencionados anteriormente.

Una vez que se determinan las UEA hábiles, se procede a una segunda eliminación de UEA del plan de estudios, eliminando aquéllas UEA que no se encuentran habilitadas, quedando únicamente así en éste las UEA viables y posibles para su inscripción.

3.6. Mínimo y máximo número de créditos recomendados

Para la determinación del número mínimo y máximo de los créditos a recomendar existen dos tipos de alumnos, por un lado estan los alumnos que se encuentran en el primer y segundo trimestre respectivamente, y por otro lado están los alumnos que se encuentran en el tercer trimestre o superior. Para el primer grupo de alumnos, el mínimo y máximo se calculó únicamente realizando un simple promedio entre el número de créditos acumulados y los trimestres cursados.

Sin embargo para el segundo grupo, el mínimo número de créditos a recomendar se calcula utilizando la siguiente fórmula:

$$creditosminimos = \frac{creditostotalesplan - creditosrealesaprobados}{trimestresenconcluir - trimestrescursados} \quad (1)$$

donde:

créditostotalesplan: créditos mínimos que se deben cubrir para concluir la carrera, en este caso 491 créditos.

créditosrealesaprobados: créditos realmente acumulados es decir, la suma real de créditos según cada bloque en que se ubican cada UEA.

trimestresenconcluir: número de trimestres recomendados parta concluir la carrera, el cual puede ser 12, 18, 24 ó 30 según sea el caso del alumno.

trimestrescursados: número de trimestres cursados por el alumno.

Es necesario mencionar que el SAE no contempla el registro de aquéllos trimestres en los que al alumno no se ha inscrito o en su defecto ha solicitado una pausa voluntaria en sus estudios, por lo que para conocer el número de trimestres cursados fue necesario implementar un proceso, en el cual mediante la obtención del trimestre de ingreso de éste (el cual se encuentra dentro de la sección de Información Académica) y la obtención de la fecha en el momento que el alumno hace usó de Mapp fue posible conocer dicho parámetro mediante una simple resta entre estas fechas.

El máximo número de créditos se calculó con un promedio móvil con los créditos acumulados y aprobados durante los últimos tres trimestres, sin embargo, tanto para el cálculo del número mínimo y máximo de créditos se presentaron cinco casos posibles que a continuación se exponen:

- El mínimo es mayor que el máximo y además es mayor al límite de créditos otorgados por la universidad. En este caso el mínimo recomendado toma el valor que contiene el máximo siempre y cuando este no exceda el límite anteriormente mencionado. Y por lo tanto el máximo es el límite, en caso de que ambos sobrepasen dicho límite, tanto el máximo como el mínimo de créditos a recomendar se establecen como ese límite de créditos otorgado por la universidad.

- El máximo es menor al mínimo, para este caso el máximo toma inmediatamente el valor del mínimo y se le agrega el número de créditos de una UEA más, siempre y cuando el máximo no sea cero. Si el máximo es cero inmediatamente el máximo y el mínimo son iguales a el límite de créditos otorgados por la universidad.

- El máximo es igual al mínimo, para esta situación si ambos son distintos del límite otorgado, al máximo se le agrega el número de créditos de una UEA evitando que este tome un valor por encima del límite otorgado. En caso de que ambos sean iguales a el límite dichos valores se quedan de dicha forma, en caso de estar por encima del límite, inmediatamente toman el valor de dicho límite.

- El mínimo es menor al máximo, para este caso solo se suman los créditos de una UEA al máximo cuidando que no sobrepase el límite de créditos y en caso de ser así, establecer el máximo como el límite.

- El mínimo es menor que el máximo y además es cero, este caso se presenta básicamente cuando el alumno ha concluido la licenciatura con el mínimo de créditos necesarios, por lo que de forma inmediata el máximo toma el valor de cero.

4. Recomendación de UEA

El proceso de construir la recomendación de UEA que el alumno debería inscribir en su próximo trimestre se realizó tomando en cuenta distintos aspectos como: la prioridad que cada UEA mantiene dentro del plan de estudios, los créditos recomendados y demás parámetros que se explican a continuación.

4.1. Prioridad de las UEA

Como ya se mencionó la prioridad que cada UEA mantiene dentro del plan de estudios, es el factor más importante para construir de forma adecuada la recomendación. Dicha prioridad fue establecida según el criterio del coordinador de la carrera, el criterio de decisión de cada orden de importancia que la UEA mantiene en el plan fue establecido de acuerdo a las cadena de seriación más larga es decir, sí una UEA al ser aprobada habilita un número importante de UEA a comparación de otra, esta primer UEA mantiene una prioridad más alta respecto de la otra.

Otro aspecto considerado para establecer la prioridad fue si la UEA era un corregistro es decir, si ésta podía cursarse simultáneamente con otra o en su defecto haberse aprobado una UEA antes de cursarse ésta. Cuando ocurría ésta situación la prioridad más alta se colocó a la UEA que debería haberse acreditado antes del corregistro y la siguiente prioridad se destinaría a ésta. Este caso ocurrió en la mayoría de las ocasiones en UEA que correspondían a una modalidad práctica como los laboratorios o talleres.

Aunque el proceso anterior aparenta ser trivial no lo es, ya que para ordenar cada UEA primero se tuvieron que agrupar cada una de éstas en bloques que representaban cada uno de los trimestres de duración de la carrera y posteriormente asignar una prioridad a cada UEA por bloque.

4.2. UEA optativas

Para recomendar UEA optativas se considerarían aspectos importantes, en primer lugar los dos grandes conjuntos en que estas se dividen, ya que por un lado se encuentra el conjunto de UEA que pertenecen a las optativas de Integración y por otro las UEA pertenecientes al tronco Inter - Multidisciplinar. A continuación se explican de forma individual los criterios que se toman en cuenta para recomendar UEA de cada uno de estos conjuntos.

4.2.1. Optativas Inter - Multidisciplinar

Para recomendar UEA de este conjunto únicamente se toma en cuenta que se cumpla la seriación, puesto que las UEA de este conjunto solo solicitan como requisito cumplir con una cierta cantidad de créditos, siendo la misma para cada una de éstas. Al momento de recomendar una UEA de este tipo se verifica que los créditos mínimos a cubrir para este tipo de UEA no hayan sido cubiertos aún, ya que el alumno debe de acumular un mínimo de 24 créditos en este tipo de UEA. Una de las ventajas en recomendar este tipo de UEA, es que cada una de éstas tiene asignado un número de créditos que sigue el mismo patrón es decir, múltiplos de tres y siendo más precisos cada una de estas UEA tiene un valor de seis créditos, por lo que el alumno debe de cubrir al menos cuatro UEA.

Sin embargo dentro de la recomendación de estas UEA no se recomienda una UEA en concreto, ya que no se sabe que UEA desea inscribir el alumno por lo que al contemplar una UEA de este tipo únicamente es considerado el grupo al que pertenece "Inter - Multidisciplinar".

4.2.2. Optativas de Integración

Otro grupo contemplado para la recomendación es el grupo de las UEA optativas de Integración, al igual que el anterior grupo se necesita cubrir un mínimo de 69 créditos. Con esto el coordinador de la carrera propone una recomendación ideal, la cual consiste en acreditar siete UEA de nueve créditos y una de seis sumando así los créditos mínimos necesarios para cubrir las UEA de este grupo. Sin embargo, en este grupo a diferencia del anterior las UEA no siguen el mismo patrón, pues aunque en su mayoría son UEA donde los créditos de las mismas son múltiplos de tres (3,6,9,12 y 18), existen UEA que no siguen este patrón, ya que tienen asignado un valor de 8 créditos provocando que al acreditar una UEA de éstas, la recomendación ideal se modifique.

Para resolver este obstáculo fue necesario realizar la construcción de varios casos dependiendo si el alumno ha acreditado una o más de estas UEA, tratando que la suma de créditos en cada uno de ellos no excediera por más de dos créditos la recomendación ideal. El plan de estudios en este grupo contempla cinco UEA que tiene asignado un valor de 8 créditos. Los casos construidos así como el criterio de recomendación que se sigue en cada uno de éstos se describe a continuación.

- **CASO I.** *El alumno no ha acreditado ninguna UEA de ocho créditos.* Este es el caso ideal sugerido por el coordinador, por lo que la recomendación queda de la siguiente forma:

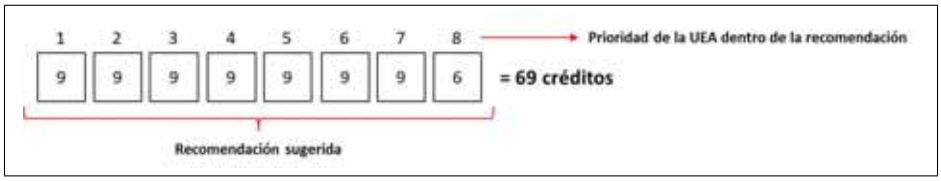


Figura 16: Optativas de Integración: Caso 1

- **CASO II.** *El alumno ha acreditado una UEA de ocho créditos.* En el caso de que el alumno ya haya acreditado al menos una UEA de 8 créditos, la recomendación sugiere acreditar un total de nueve UEA dando un total de 71 créditos.

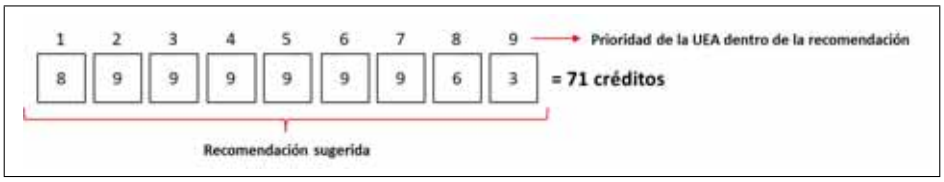


Figura 17: Optativas de Integración: Caso 2

- **CASO III.** *El alumno ha acreditado dos UEA de ocho créditos.* En caso de ya haber acreditado dos UEA de 8 créditos, la recomendación quedó de la siguiente forma con un total de 70 créditos.

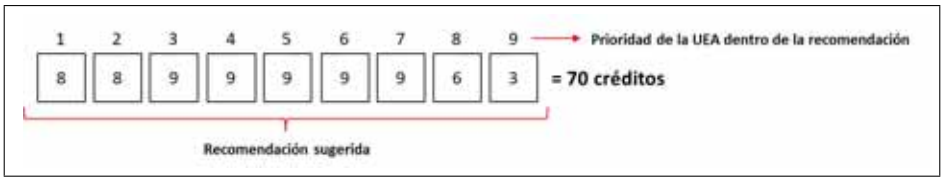


Figura 18: Optativas de Integración: Caso 3

- **CASO IV.** *El alumno ha acreditado tres UEA de ocho créditos.* En caso de ya haber acreditado tres UEA de 8 créditos, la recomendación sugerida suma un total de 69 créditos dando así la misma cantidad de créditos que la recomendación ideal.

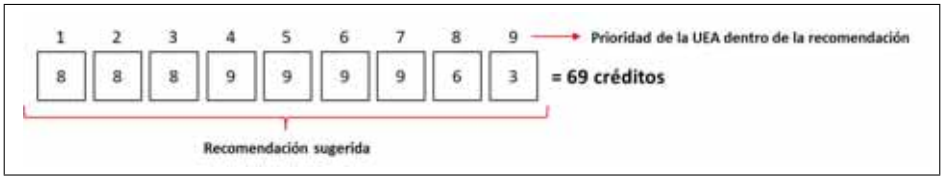


Figura 19: Optativas de Integración: Caso 4

- **CASO V.** *El alumno ha acreditado cuatro UEA de ocho créditos.* En caso de ya haber acreditado cuatro UEA de 8 créditos, la recomendación sugerida da un total de 71 créditos.

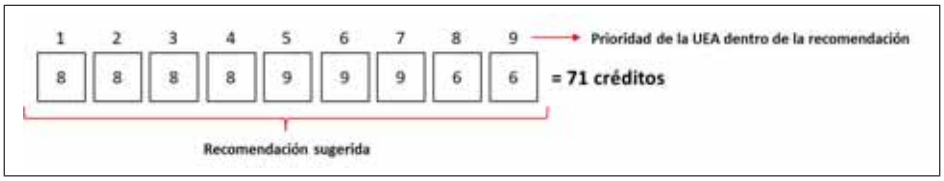


Figura 20: Optativas de Integración: Caso 5

- **CASO VI.** *El alumno ha acreditado cinco UEA de ocho créditos.* En caso de que el alumno ya haya acreditado las cinco UEA de 8 créditos existentes, la recomendación sugerida queda de la siguiente forma.

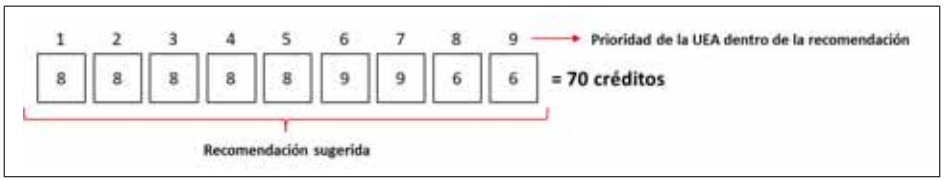


Figura 21: Optativas de Integración: Caso 6

Al construir cada una de éstas recomendaciones, se permite que el alumno siga un patrón lo más cercano al ideal tanto en número de créditos como en el número de UEA que debe acreditar, cabe mencionar que cada UEA mantiene una prioridad interna dentro de la recomendación sugerida. . La recomendación de este tipo de UEA básicamente se realiza verificando los créditos de cada una de las UEA ya acreditadas según sea el caso del alumno y se van otorgando las UEA de la recomendación, sin embargo, es probable que al final de este proceso quede un sobrante de créditos por lo que dicho sobrante es guardado y posteriormente dependiendo del valor de éste se recomiendan las UEA, de tal forma que el alumno complete una de las UEA sobrantes en la recomendación. Éste proceso es válido sólo si el alumno no ha cubierto el mínimo de créditos a aprobar de optativas de Integración.

Por ejemplo, si un alumno ha acreditado un total de 47 créditos distribuidos de la siguiente forma *ver figura 22*, vemos claramente que se otorgan seis UEA de las 9 que la recomendación contempla. Sin embargo, aún quedan libres tres UEA de 9 créditos y además el alumno tiene un sobrante de 3 créditos. Ahora bien con estos datos se recomienda al alumno que inscriba una UEA de 6 créditos permitiendo completar una UEA de 9 créditos y así poder cubrir una de éstas que aún le sobran, permitiendo que en futuras recomendaciones sólo se recomienden dos UEA de 9 créditos logrando cubrir así los créditos necesarios de optativas de Integración.

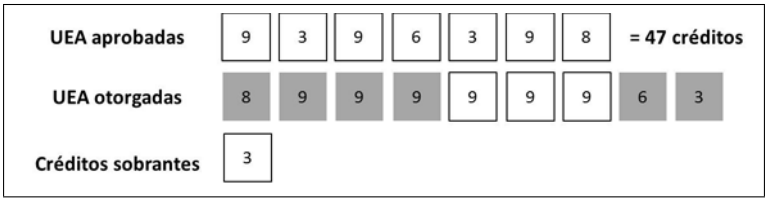


Figura 22: Ejemplo de recomendación de optativas de Integración

4.3. Créditos recomendados

Otro aspecto de gran importancia que se tomó para generar la recomendación fueron los créditos recomendados, los cuales se determinaron de la manera que se mencionó en la sección 3.5, teniendo así que el número de créditos recomendados con los cuales es generada la recomendación es siempre el máximo, ya que lo que se busca es que la estadía del alumno en la UAM sea lo más corta posible.

4.4. Algoritmo de recomendación

A continuación se describe el algoritmo que genera la recomendación considerando de forma conjunta todos los aspectos anteriormente descritos.[2]

El algoritmo implementado debe considerar además de la prioridad de cada UEA, el tipo de UEA del cual se trata puesto que existen dos tipos de UEA. Por un lado están las UEA de carácter obligatorio, de las cuales lo único a cuestionar es si al sumar los créditos de ésta a los acumulados tras ir recorriendo el plan de estudios (el cual contiene únicamente las UEA hábiles al final del análisis de la información del alumno) e ir agregando las UEA recomendadas no excede los créditos recomendados así como el valor del límite otorgado por la UAM, ya que al menos deben de ser iguales a éstos. La condición de paro de este recorrido es básicamente cuando los créditos acumulados de las UEA que se van recomendado es mayor a los recomendados o que el plan de estudios ha sido recorrido en su totalidad.

Otro aspecto que se tomó en cuenta, fue si la UEA evaluada necesitaba corregistro con otra UEA, en caso de que fuera verdad este hecho, fue necesario buscar dicho corregistro tanto en el kardex del alumno como en las UEA que se iban recomendado y agregando a la recomendación, ya que es probable que dicha UEA corregistro no se haya aprobado y se encuentre dentro de las UEA ya recomendadas. Éste caso se presenta cuando se tienen UEA, donde una es la parte teórica y la otra la parte práctica y además ambas son viables para su inscripción.

Sin embargo, tenemos también las UEA de carácter optativo y además agrupadas en dos grandes grupos: Integración y Multidisciplinar. Para recomendar UEA del segundo tipo, únicamente se necesita evaluar la condición de que al sumar los créditos de éstas a los acumulados sean al menos igual a los créditos que se recomiendan inscribir o el límite otorgado por la

[2]El algoritmo necesita que las UEA hábiles se encuentren ordenadas por prioridad

UAM. Dando como resultado que si se cumple el hecho anterior se agrega a la recomendación el identificador "TIM", el cual en un proceso posterior es reemplazado para mostrar como clave de UEA (11XXXXX), permitiendo con esto que el alumno elija la UEA que desee inscribir de manera libre es decir, no forzarlo a inscribir la UEA según la prioridad de ésta. Cabe mencionar que para este tipo de UEA los créditos de éstas son siempre igual y tienen un valor de 6 créditos y la recomendación de las mismas será válida siempre y cuando no se hayan cubierto los créditos mínimos necesarios para éstas es decir, 24 créditos.

Para el caso de las UEA de optativas Integración son tomados más aspectos, por un lado la recomendación siempre será válida si no se han cubierto los créditos mínimos de estas (69 créditos) si este hecho es verdadero, es necesario conocer las UEA que se tienen hábiles según lo explicado en la sección 4.2.2, ya que si existen sobrantes de créditos es necesario verificar el tipo de UEA recomendada que se puede otorgar, completando los créditos sobrantes como sigue:

- El sobrante de créditos es 0. En este caso la UEA que se pueden recomendar pueden se UEA de 3, 6 y 9 créditos, otorgando alguna de éstas siempre y cuando exista disponibilidad de alguna de ellas.
- El sobrante de créditos es 3. Para este caso existen dos posibilidades de recomendación, por un lado recomendar una UEA de 3 créditos y así poder completar una UEA de 6 créditos siempre y cuando exista alguna de éstas disponible. O por el contrario recomendar una UEA de 6 créditos que permita completar y otorgar una UEA de 9 créditos.
- El sobrante de créditos es 6. En este caso final la única posibilidad es recomendar una UEA de 3 créditos que permite completar y otorgar una UEA de 9 créditos.

Para cualquiera de los tres casos se toma en cuenta que al sumar sus créditos correspondientes sean menores o iguales a los créditos recomendados y al límite otorgado por la UAM, por lo que si este hecho se cumple se agrega a la recomendación el siguiente identificador "TI:créditos", con el cual es posible recuperar el tipo de UEA así como los créditos que se deben contemplar de la misma. De forma similar que las UEA Inter - Multidisciplinar en un proceso posterior se muestra únicamente la clave de la misma forma, permitiendo de igual forma que el alumno decida la UEA que desea inscribir de forma libre pero respetando los créditos de ésta.

Las cuatro siguientes figuras muestran el algoritmo implementado de forma general, los subprocesos solicitados son básicamente búsquedas en los cuales se devuelve falso o verdadero si se cumple la condición a validar.

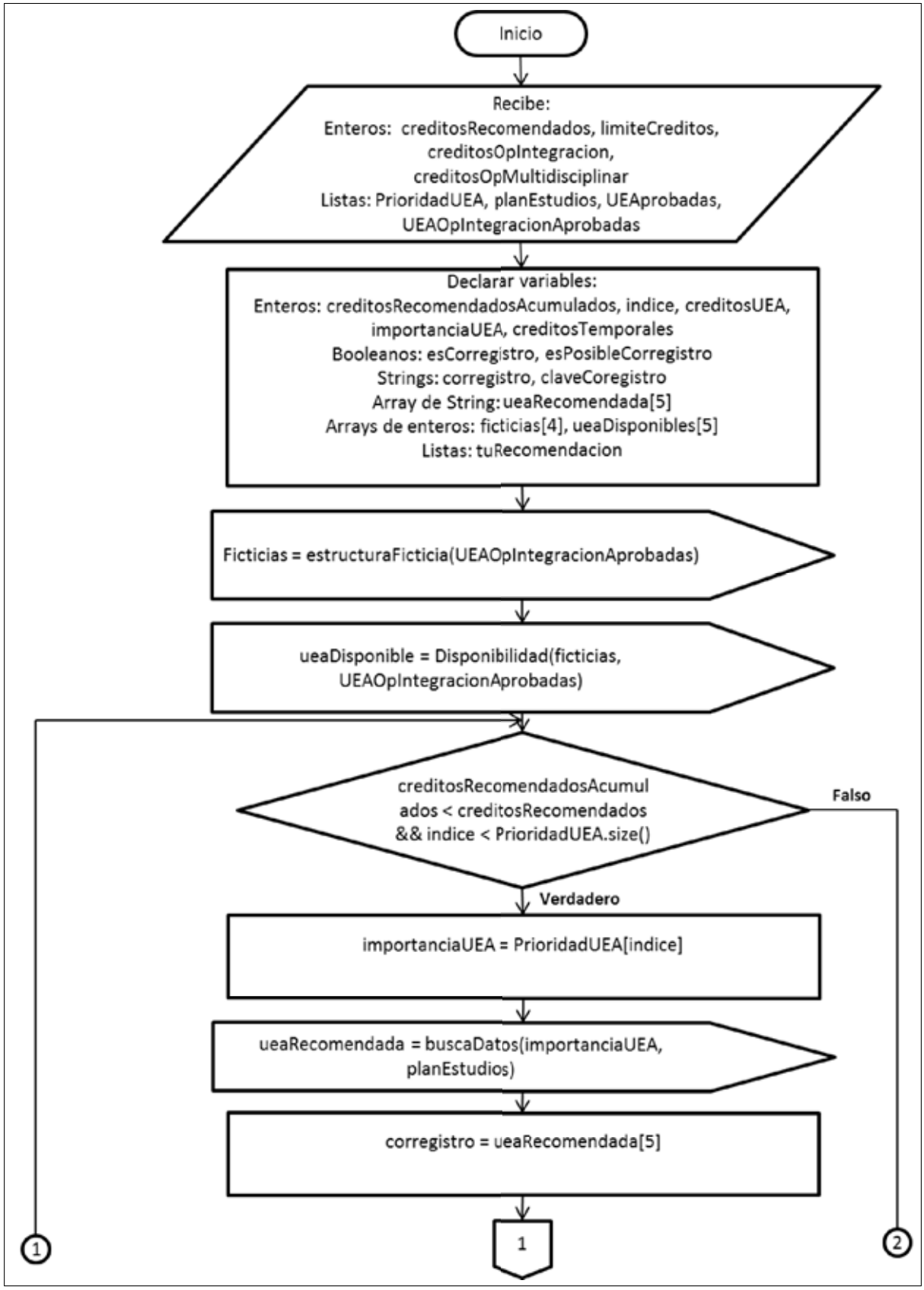


Figura 23: Algoritmo de Recomendación 4-1

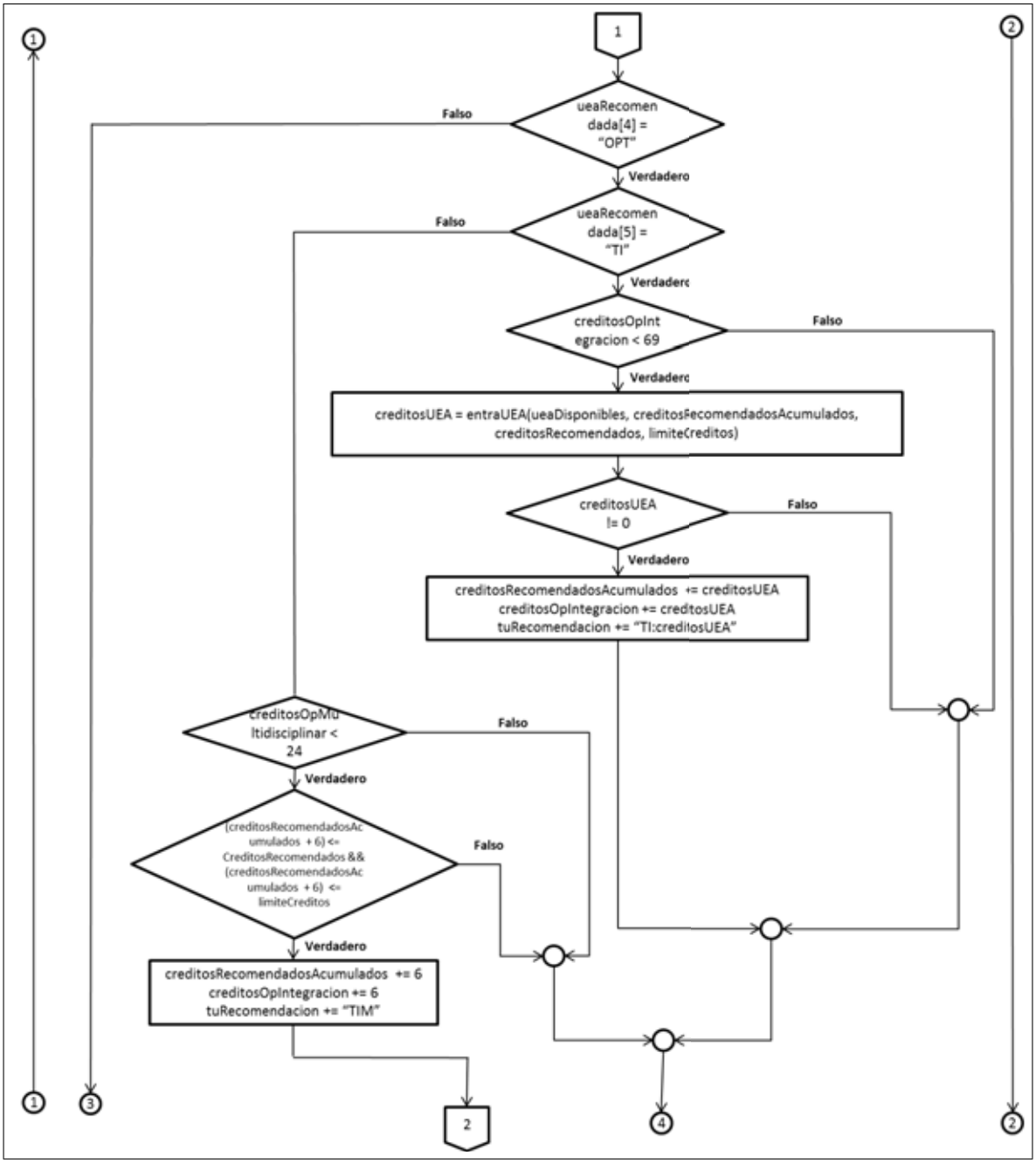


Figura 24: Algoritmo de Recomendación 4-2

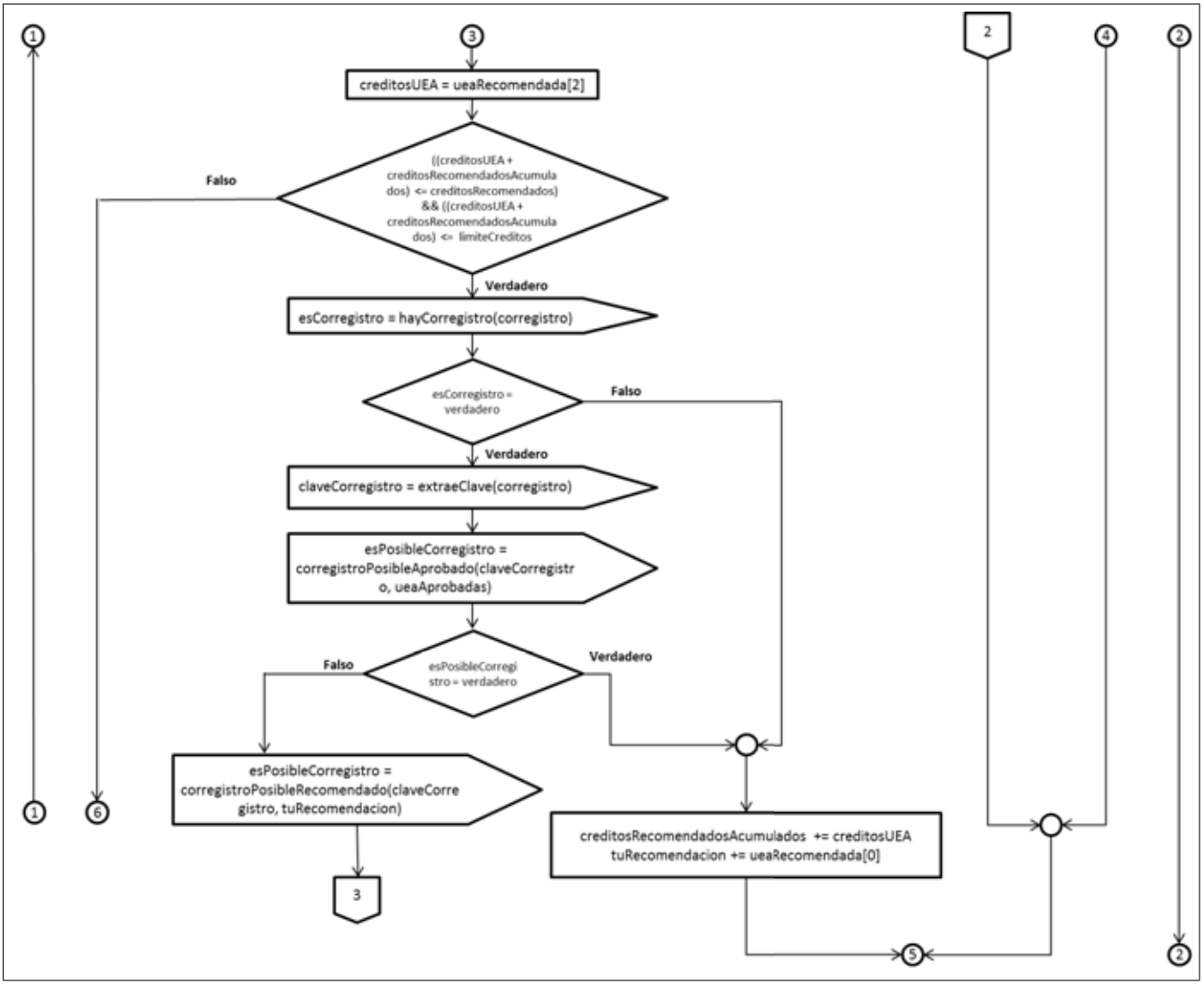


Figura 25: Algoritmo de Recomendación 4-3

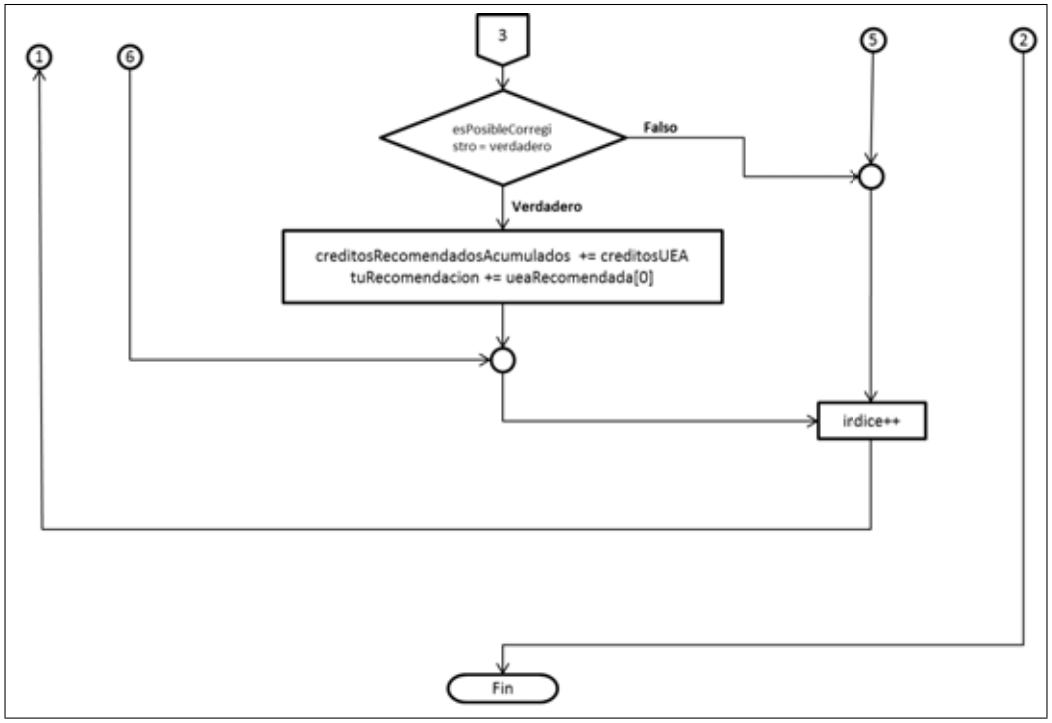


Figura 26: Algoritmo de Recomendación 4-4

5. Conclusiones

Durante el diseño y desarrollo del proyecto se presentaron diferentes obstáculos algunos ajenos a éste y otros no, que si bien no fueron de gran facilidad resolverlos tampoco serían un caso imposible. En primer lugar se contó con el detalle del establecimiento de la conexión con el SAE, ya que la página donde se encuentra alojado éste implementa tecnología de cifrado SSL, por lo que fue necesario diseñar un algoritmo que fuera capaz de realizar esta a pesar de la utilización de dicha tecnología. Dicho algoritmo se logró diseñar a través del conjunto de clases y métodos que Android nos provee.

Sin embargo, otro de los obstáculos se presentaría al momento de aplicar la fórmula del cálculo de los créditos mínimos a inscribir, puesto que el SAE no contempla el registro del trimestre o trimestres en el que el alumno no se inscribe al trimestre o en su defecto, éste solicita una pausa temporal en sus estudios. Para solucionar este inconveniente fue necesario extraer el trimestre de ingreso del alumno y posteriormente a través de la obtención de la fecha actual en que se está generando la recomendación, realizar el cálculo de los trimestres que ha cursado y así poder aplicar dicha fórmula. Cabe resaltar que al calcular el número de trimestres cursados por el alumno, se realizó según las fechas de inicio y cierre del trimestre impuestas en el calendario escolar, por lo que con este hecho deberá ser necesario actualizar la aplicación cada año. Este proceso fue implementado bajo el hecho de que el tiempo de estadía del alumno dentro de la universidad sigue corriendo de manera normal aún y cuando cualquiera de las dos situaciones anteriores se presenta.

Otro inconveniente se presentó al momento de realizar la recomendación, ya que para sugerir las UEA optativas de Integración fue necesario construir recomendaciones ficticias, puesto que al no tratar con las áreas de concentración existe la posibilidad que el alumno haya acreditado una o más UEA con un valor de 8 créditos, lo que obliga a que la recomendación sugerida por el coordinador tenga que modificarse. Al construir dichas recomendaciones se buscó que éstas fueran lo más cercanas a la recomendación propuesta por el coordinador tanto en número de créditos como de UEA a cursar.

Sin embargo a pesar de estos obstáculos, el objetivo general planteado desde un inicio para la realización del presente proyecto, el cual fue " Diseñar e implementar una aplicación para dispositivos móviles que le pueda recomendar a un alumno de Ingeniería Mecánica las UEA que debería inscribir en su próximo trimestre" se cumplió, puesto que se logró el diseño e implementación de una aplicación móvil que es capaz de generar una recomendación de inscripción para el próximo trimestre del alumno. De igual forma se cumplieron con éxito los objetivos particulares, ya que se logró diseñar un algoritmo que permitió establecer la conexión con el SAE y posteriormente extraer la información necesaria del alumno, con la cual se llevó a cabo el análisis de la situación de éste y posteriormente con el modelado del plan de estudios y el establecimiento de la prioridad de cada una de las UEA implementar el algoritmo que generaría la recomendación.

Con lo anterior podemos concluir que el cumplimiento del objetivo general así como los objetivos particulares fue satisfactorio y con lo cual se deja abierta la oportunidad que a futuro Mapp pueda evolucionar, de tal forma que Mapp pueda tratar con el área de concentración dependiendo de cada alumno es decir, que la recomendación de las UEA optativas de Integración se realice conforme al área de concentración a la cual pertenezca el alumno. Finalmente con la realización de este proyecto fueron reforzados los conocimientos adquiridos a lo largo de la carrera, poniendo en práctica el desarrollo de un proyecto de software y cada una de las tareas que se ven implicadas en éste.

Bibliografía

[1] Modulo de información escolar
https://ayamictlan.uam.mx:8443/sae/azc/aewbf001.omuestraframes?mod=1

[2] Planes y programas de estudio aprobados
http://cbi.azc.uam.mx/planes/ing_mecanica.html

[3] A. Cruz, Reyes M, *"Tutor Virtual para alumnos de la carrera de Ingeniería en Computación"*, proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2008.

[4] J. César Zamora, *"Sistema Web de Seriación e Información para Alumnos de Ingeniería"*, proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2009.

[5] Guia de estilo para iconos Android
http://developer.android.com/guide/practices/ui_guidelines/icon_design.html

[6] Android Asset Studio
http://android-ui-utils.googlecode.com/hg/asset-studio/dist/index.html

[7] Reglamento de Estudios Superiores de la Universidad Autónoma Metropolitana
http://www.uam.mx/legislacion/2013_abril/RES_legislacion_abril_2013/index.html#/8/zoomed

[8] Google Play
https://play.google.com/store

Apéndices

A. Manual de Uasuario

El presente manual de usuario presenta la forma de utilizar Mapp (Mecánica App), así como los posibles errores que pueden ocurrir durante su utilización.

A.1. Obtención e instalación de Mapp

Antes de poder generar la recomendación deberemos haber descargado Mapp, la cual estará disponible en la tienda oficial de aplicaciones de Android [8] e instalarla. De forma opcional Mapp también estará disponible temporalmente en: http://www.ingenieriamecanica/mapp

Donde únicamente se deberá dar cilc en *descargar*. Una vez descargado el archivo *Napp.apk*, una vez realizado ésto deberemos copiar el archivo mediante el USB al dispositivo, donde quedará almacenado en la memoria de este.

Posterioemente dentro del dispositivo deberemos indicar a éste que permita la instalación de aplicaciones externas a la tienda (Google Play). Para ello debemos ir a "Configuración >Seguridad" y activar la opción de "Fuentes desconocidas u Origenes desconocidos".

Finalmente deberemos acceder a la ubicación en que quedo alamcenado el archivo previamente descargado, la cual deberemos seleccionar desplegandose un mensaje notificandonos que la aplicación requiere de acceso a Internet, a continuación deberemos pulsar *Instalar* tras unos segundos de espera Mapp quedará instalada y lista para utilizarse.

A.2. Generar recomendación

Una vez que hemos instalado Mapp, deberemos ir a la ubicación de ésta y ejecutarla. Una vez abierta Mapp, para generar la recomendación se debe introducir la Matrícula y la Contraseña *ver figura 27* y posteriormente dar clic en el botón *Ingresar*, desplegándose así un cuadro de diálogo que indica que la solicitud se está procesando *ver figura 28*.

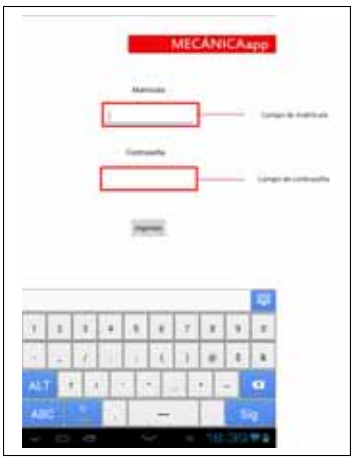


Figura 27: Pantalla de inicio de Mapp

Figura 28: Análisis de la información del alumno

Tras unos segundos de espera, se despliega la recomendación indicando al alumno a través de un mensaje lo siguiente *"Es probable que no siempre los créditos se ajusten exactamente a los créditos recomendados o ajustados."* es decir, no ha sido posible encontrar una combinación de UEA que sumen de manera exacta los créditos que se recomiendan. Al dar clic sobre este mensaje podrá observar la lista de UEA que se recomiendan cursar, así como opciones y parámetros adicionales y relevantes *ver figuras 29 y 30.*

Figura 29: Genreación de la recomendación

Figura 30: Despliegue de listado de UEA a cursar

A.3. Modificar recomendación

Para modificar la recomendación generada es necesario ajustar el número de créditos que se desean cursar, para llevar a cabo ésta acción basta con deslizar la barra de ajuste de créditos y ajustar los créditos deseados a inscribir. Los créditos ajustados se pueden observar encima de la barra de ajuste en la etiqueta *"Ajustar créditos" ver figura 31.*

Figura 31: Ajuste de créditos

Cabe resaltar que no siempre se encuentra una combinación de UEA que se ajuste a dicho número de créditos solicitado, por lo que el número de créditos realmente ajustados se muestran en la etiqueta de *"Ajuste posible"*, así también el límite de ajuste posible es el límite de créditos otorgados por la universidad *ver figura 32*.

Figura 32: Ajustar créditos al máximo permitido

A.4. Opciones adicionales

Dentro de la pantalla donde se despliega la recomendación, además de mostrar parámetros importantes para el alumno, también se le ofrecen a éste opciones adicionales siéndole de gran ayuda y que enriquecen a Mapp. Algunas de estas funciones adicionales son: el poder ajustar los créditos recomendados (tarea que fue explicada en un punto anterior), también se ofrece la opción de poder consultar las UEA optativas Inter - Multidisciplinar así como las de Integración habilitadas y una opción de poder enviar comentarios acerca del diseño y desempeño de la misma, comentarios que como se mencionó en la segunda sección del presente documento se hacen llegar al coordinador de la carrera así como al desarrollador de Mapp. A continuación se explica el procedimiento para el uso de cada una de éstas opciones.

A.4.1. Consultar UEA Optativas Inter - Multidisciplinares

Para poder consultar las UEA Optativas Inter - Multidisciplinar habilitadas y con opción a inscribir, basta con seleccionar la primer opción del grupo de tres que se ubica en la parte inferior de la pantalla de recomendación *ver figura 30*. Tras seleccionar dicha opción se abrirá una nueva pantalla que contendrá una lista con aquéllas UEA que el alumno tiene habilitadas *ver figura 33*.

Figura 33: Lista de UEA Optativas Inter - Multidisciplinares hábiles

Para volver a la pantalla de recomendación basta con presionar el botón de *retorno* de Android ubicado en la parte inferior del dispositivo o en la barra de notificaciones, el cual es un botón estandarizado y se encuentra presente en cualquier dispositivo Android.

A.4.2. Consultar UEA Optativas de Integración

De forma similar a la consulta de las UEA Inter - Multidisciplinar, también es posible consultar las UEA optativas de Integración que se tienen hábiles para su inscripción, basta nuevamente con seleccionar la opción correspondiente del grupo de opciones ya mencionado. Al seleccionar ésta opción, se despliega en una nueva pantalla el listado de todas aquéllas UEA de Integración que se tienen hábiles *ver figura 34*.

Figura 34: Lista de UEA Optativas de Integración

A.4.3. Envió de comentarios

Finalmente para hacer uso de la opción que permite enviar un comentario, opinión o sugerencia acerca de Mapp, basta con seleccionar la última opción del grupo ya mencionado, desplegándose la pantalla correspondiente *ver figura 35*. Ésta pantalla contiene un espacio para escribir el comentario y un botón que lo envía, una vez que se ha escrito el comentario y presionado el botón de *Enviar*, se despliega una alerta avisando que es necesario seleccionar una de las aplicaciones para el envió de correo electrónico que el dispositivo tiene instaladas *ver figura 36*, una vez seleccionada se nos abrirá la interfaz de dicha aplicación *ver figura 37* en la cual podemos modificar el comentario y posteriormente enviar. Tras el envió de dicho comentario Mapp regresará a la pantalla de la opinión.

Figura 35: Redactar comentario a enviar

Figura 36: Elección de la aplicación de correo electrónico

Figura 37: Envió de comentarios

Cabe mencionar que para generar una nueva recomendación con datos de ingreso distintos, es necesario reiniciar Mapp y repetir cada uno de los pasos anteriormente citados.

A.5. Posibles errores

Existen distintos errores que pueden presentarse durante el uso de Mapp, algunos de estos errores son errores del usuario y otros ajenos a este. Los errores del usuario que se pueden presentar son:

- El alumno introdujo mal sus datos de ingreso, este error puede deberse a que los datos de ingreso del alumno no contemplan las especificaciones antes vistas o los datos del alumno son incorrectos.
- Problema de conexión, este problema se presenta básicamente cuando el dispositivo en donde se está utilizando Mapp, no tiene asociada una red. Para solucionar éste problema se debe asociar el dispositivo con una red, ya que de lo contrario no se prodrá

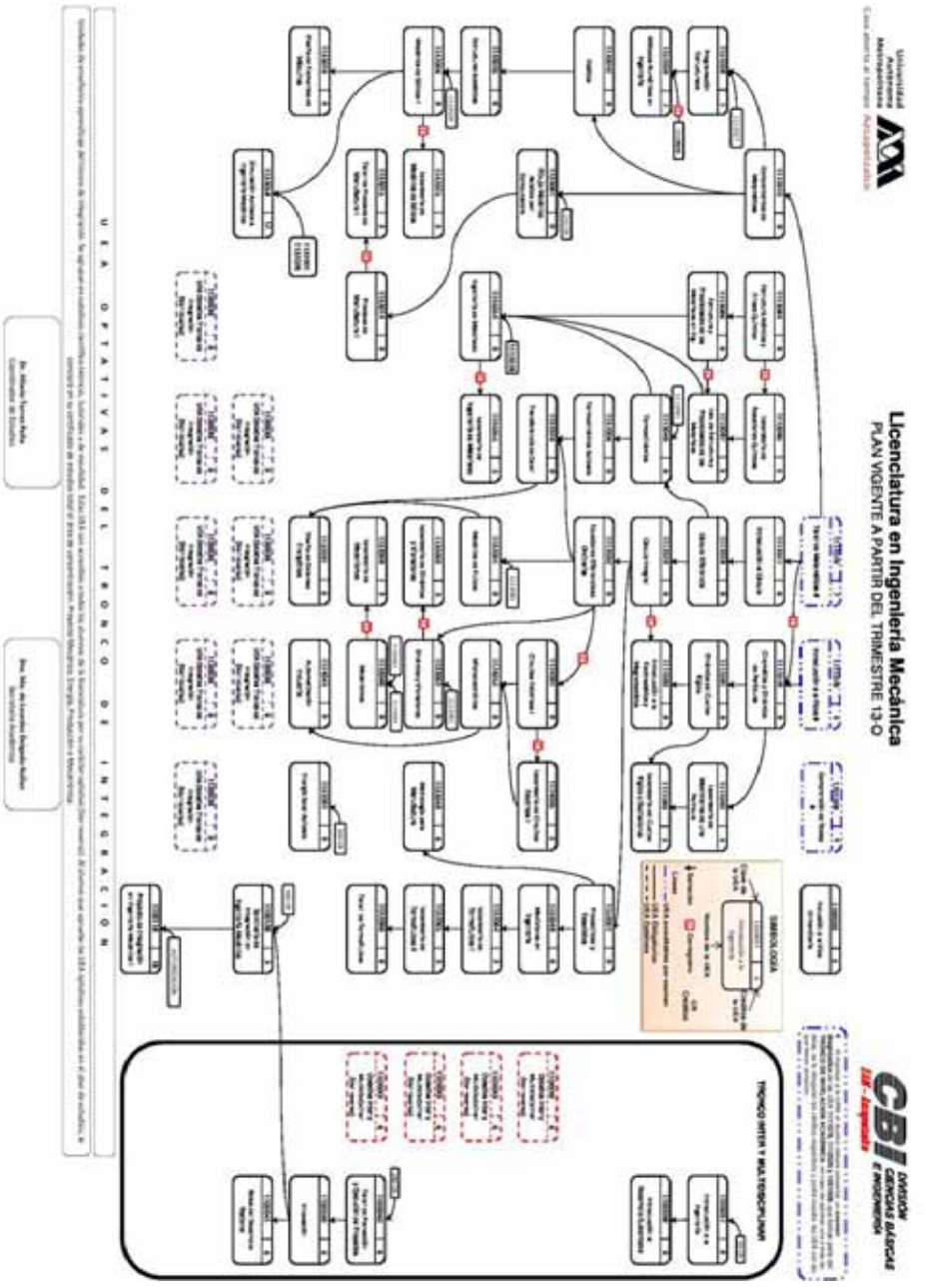
generar la recomendación. Para asociar el dispositivo con una red, se deberán ejecutar los siguientes pasos:

1. Ingresar a la ubicación de las aplicaciones instaladas en el dispositivo.
2. Buscar e ingresar en "Configuración".
3. Entrar a la opción "Wi-Fi" de la opción de "Conexiones Inalámbricas y Redes", donde se desplegarán las redes disponibles.
4. Se deberá seleccionar una de las redes disponibles y pulsar "Conectar". En caso de que la red nos solicite contraseña deberemos proporcionarla para poder hacer uso de ella, con esto el dispositivo quedará conectado y asociado a una red y podremos hacer uso de Mapp.

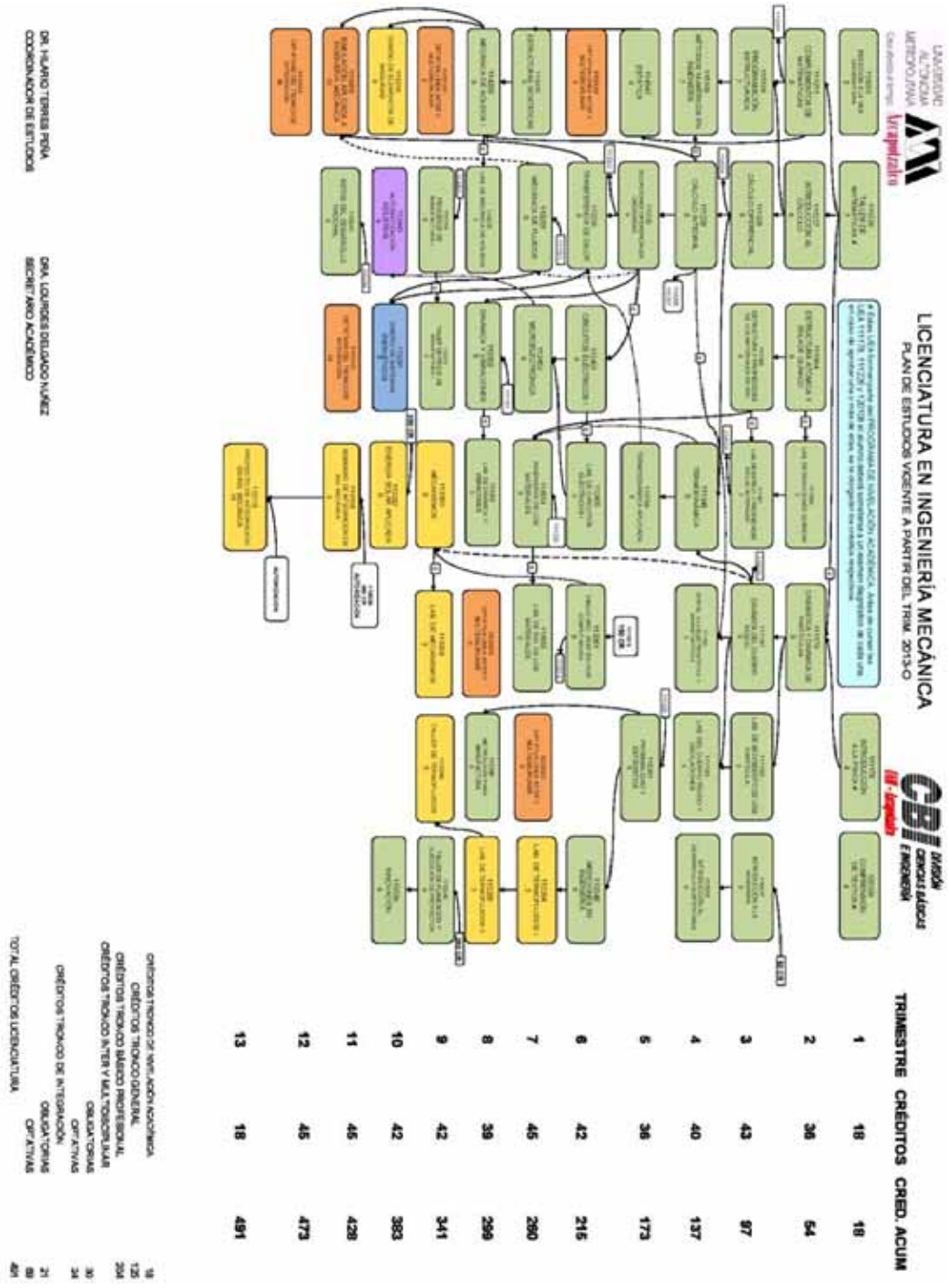
Mientras que los problemas ajenos al usuario, se presentan cuando el servidor del SAE no se encuentra disponible. Estos problemas se notifican a través de mensajes por parte de Mapp indicando la solución inmediata, la cual puede ser intentar más tarde el uso de Mapp o en otro caso si el problema persiste, enviar un correo electrónico al desarrollador (soportemapp2013@gmail.com) notificando el problema y el cual deberá dar solución a éste.

B. Diagramas de seriación

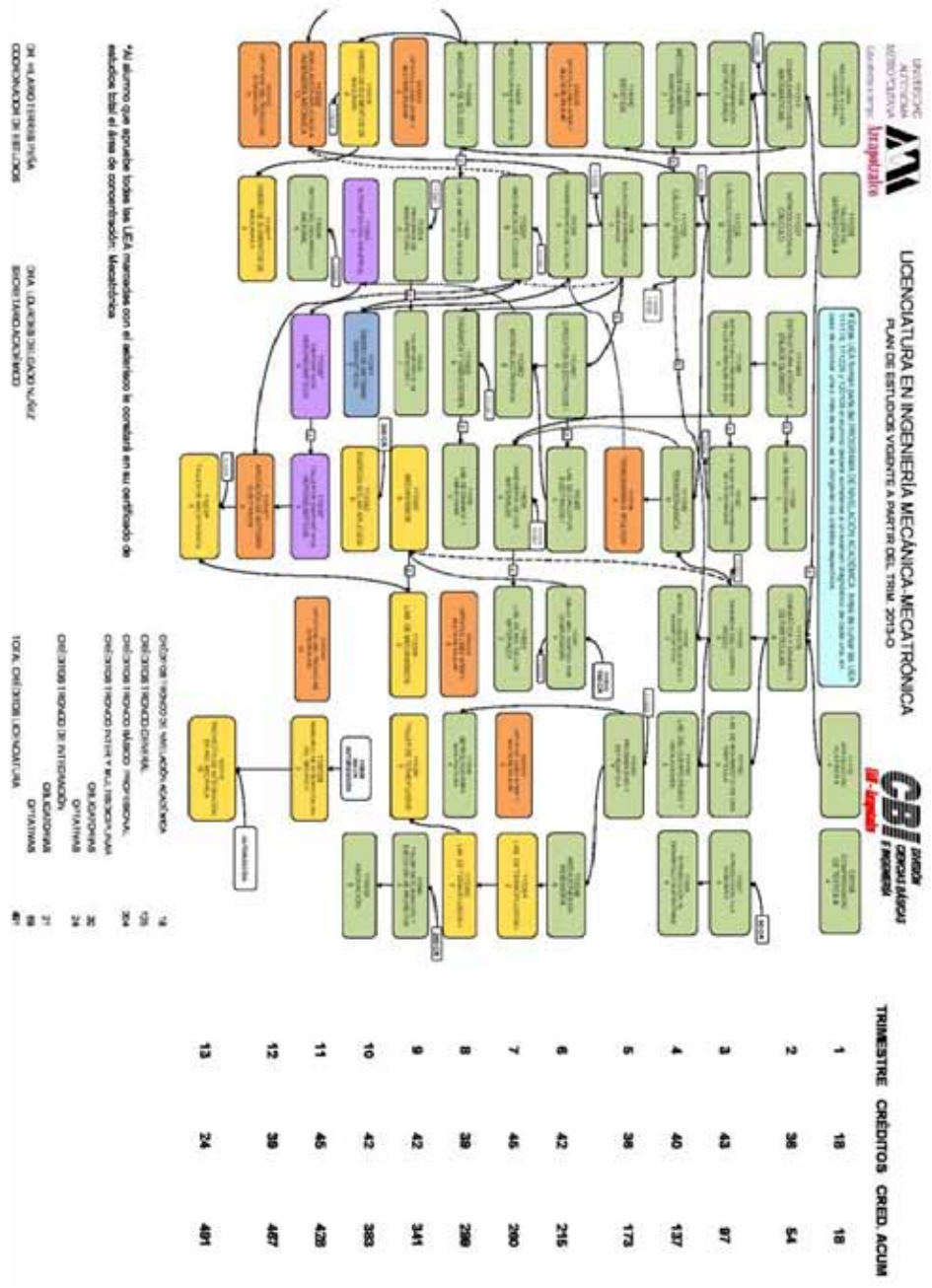
B.1. Diagrama de seriación genérico de Ingeniería Mecánica



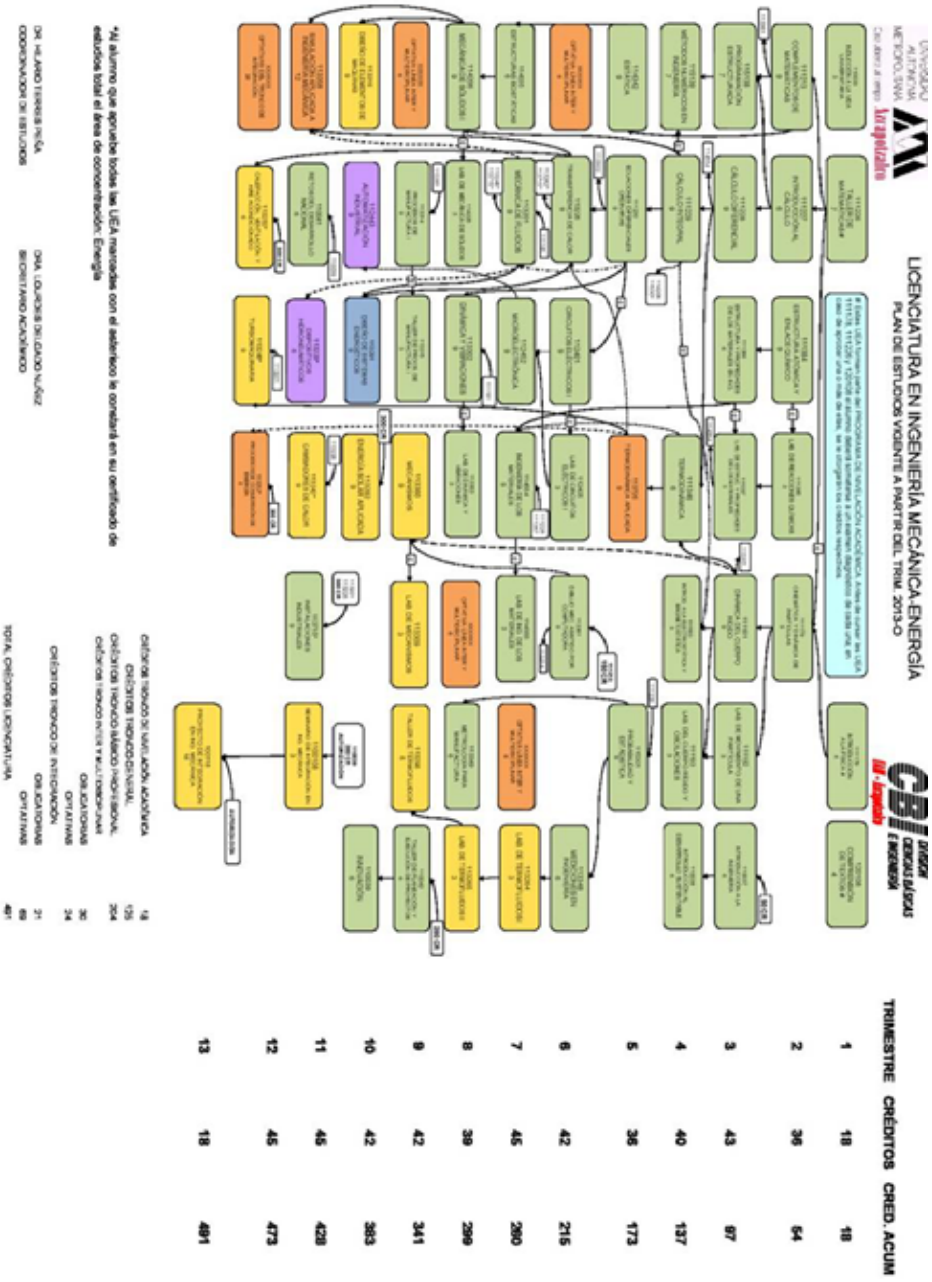
B.2. Diagrama de seriación de Ingeniería Mecánica Común



B.3. Diagrama de seriación de Ingeniería Mecánica - Mecatrónica

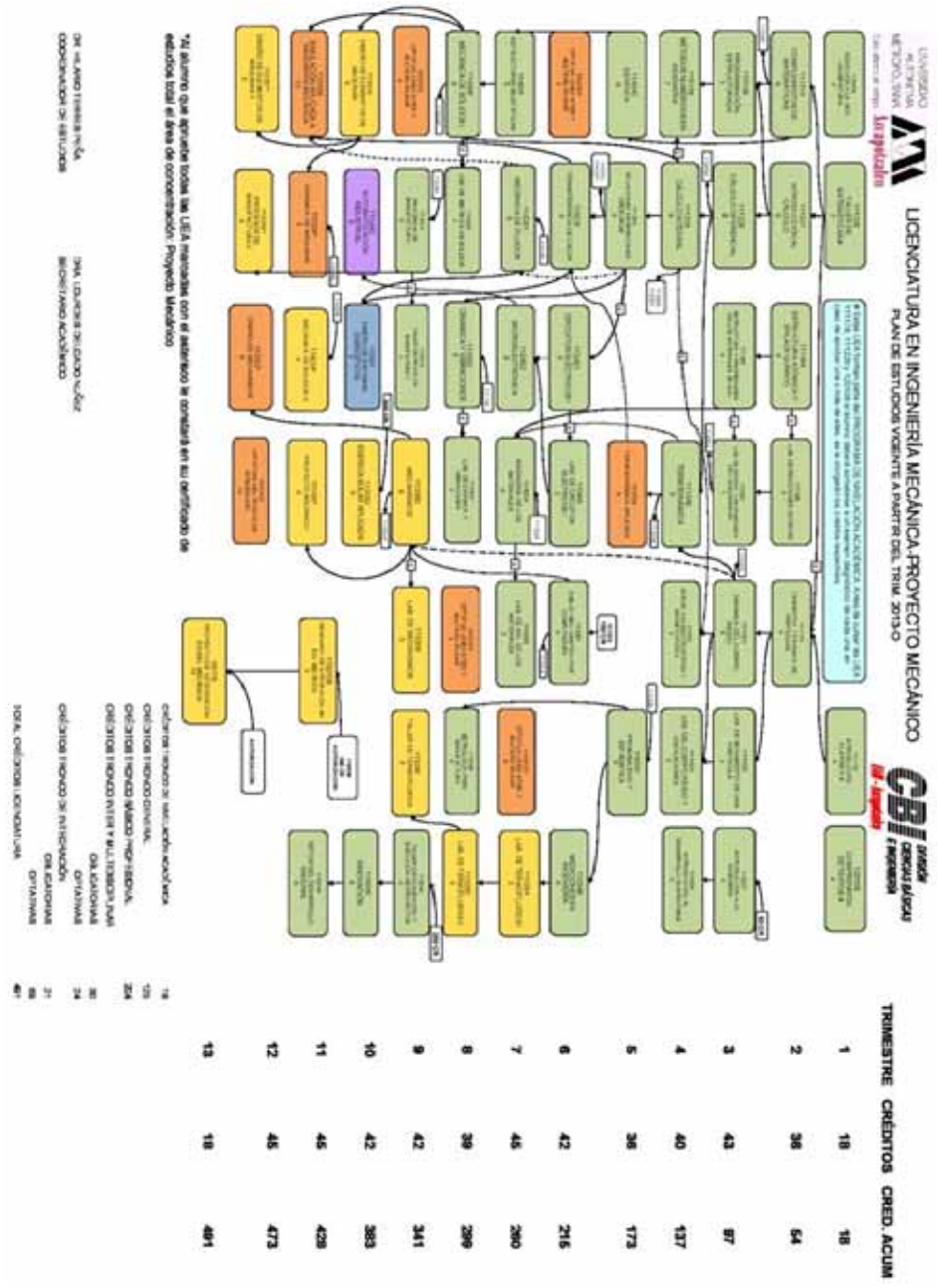


B.4. Diagrama de seriación de Ingeniería Mecánica - Energía



B.5. Diagrama de seriación de Ingeniería Mecánica - Producción

B.6. Diagrama de seriación de Ingeniería Mecánica - Proyecto Mecánico



C. Código fuente de Mapp

C.1. Código Java

C.1.1. Clase principal

```
1  package com.example.napp;
2
3  import java.io.BufferedReader;
4  import java.io.IOException;
5  import java.io.InputStream;
6  import java.io.InputStreamReader;
7  import java.io.UnsupportedEncodingException;
8  import java.util.ArrayList;
9  import java.util.Collections;
10 import java.util.Iterator;
11 import java.util.List;
12 import java.util.StringTokenizer;
13 import java.util.concurrent.ExecutionException;
14 import java.util.regex.Matcher;
15 import java.util.regex.Pattern;
16
17 import org.apache.http.Header;
18 import org.apache.http.HttpEntity;
19 import org.apache.http.HttpResponse;
20 import org.apache.http.NameValuePair;
21 import org.apache.http.client.ClientProtocolException;
22 import org.apache.http.client.entity.UrlEncodedFormEntity;
23 import org.apache.http.client.methods.HttpGet;
24 import org.apache.http.client.methods.HttpPost;
25 import org.apache.http.impl.client.DefaultHttpClient;
26 import org.apache.http.message.BasicNameValuePair;
27 import org.apache.http.params.HttpConnectionParams;
28 import org.apache.http.params.HttpParams;
29 import org.apache.http.util.EntityUtils;
30
31 import com.example.logica.Busqueda_binaria;
32 import com.example.logica.Cargar_plan;
33 import com.example.logica.Parametros;
34 import com.example.logica.Parsers;
35 import com.example.logica.Recorta_plan;
36 import com.example.logica.Uea_habilitadas;
37 import com.example.recomendacion.Prioridad;
38 import com.example.ueareprobadas.Uea_reprob;
39
40 import android.net.ParseException;
41 import android.os.AsyncTask;
42 import android.os.Bundle;
43 import android.app.Activity;
44 import android.app.ProgressDialog;
45 import android.content.Context;
46 import android.content.Intent;
47 import android.content.res.AssetManager;
48 import android.view.Menu;
49 import android.view.View;
50 import android.view.Window;
51 import android.widget.Button;
52 import android.widget.EditText;
53 import android.widget.Toast;
54
55 public class MainActivity extends Activity {
56
57   private EditText mat;
58   private EditText contrasenia;
59   private ArrayList plan1 = new ArrayList();
```

```
60  private Context context;
61  Boolean isInternetPresent = false;
62  ConnectionDetector cd;
63  ProgressDialog pg = null;
64  static ProgressDialog dialogo;
65  private ArrayList pln = new ArrayList();
66
67    @Override
68    protected void onCreate(Bundle savedInstanceState) {
69    super.onCreate(savedInstanceState);
70    this.requestWindowFeature(Window.FEATURE_NO_TITLE);
71    setContentView(R.layout.activity_main);
72
73    Cargar_plan carga = new Cargar_plan();
74    AssetManager am = getAssets();
75
76    try {
77      plan1 = carga.cargar_plan(am);
78      pln = (ArrayList) plan1.clone();
79
80    } catch (IOException e1) {
81      // TODO Auto-generated catch block
82      e1.printStackTrace();
83    }
84
85    Button aceptar;
86
87    mat = (EditText)findViewById(R.id.editText1);
88    contrasenia = (EditText)findViewById(R.id.editText2);
89    aceptar = (Button)findViewById(R.id.button1);
90
91    cd = new ConnectionDetector(getApplicationContext());
92    aceptar.setOnClickListener(new View.OnClickListener() {
93
94      @Override
95      public void onClick(View arg0) {
96        // TODO Auto-generated method stub
97
98        isInternetPresent = cd.isConnectingToInternet();
99
100       Alerta as = new Alerta();
101       if (!isInternetPresent) {
102         as.error(MainActivity.this, "Error", "Error de conexion a Internet, por favor
              verifica tu conexion.");
103       }
104       else
105       {
106
107         boolean correct;
108         String matricula="";
109         String password="";
110         matricula = mat.getText().toString();
111         password = contrasenia.getText().toString();
112         String bmat = "La matricula debe tener una longitud de 9 o 10 numeros";
113         String bpas = "La contrasenia debe estar compuesta por: \n" +
114             "1. Una palabra de 10 a 15 caracteres \n"+
115             "2. Debe contener al menos una letra mayuscula [A-Z]\n"+
116             "3. Debe contener al menos una letra minuscula [a-z]\n"+
117             "4. Debe contener al menos un numero de 0 a 9\n"+
118             "5. No debe contener caracteres especiales\n";
119
120         correct = as.valida_contrasenia(password);
121
122         if(matricula.length() < 9 || matricula.length() > 10 || matricula.isEmpty())
123         {
124           as.error(MainActivity.this, "Error",bmat);
125         }
```

```
126             else if(password.isEmpty() || password.length() < 10 || password.length() > 15 ||
                    !correct)
127             {
128               as.error(MainActivity.this, "Error", bpas);
129             }
130             else
131             {
132                 new Conexion().execute(matricula,password);
133             }
134           }
135         }
136     });
137   }
138
139   private class Conexion extends AsyncTask<String, String, Void> {
140
141     String matricula, password;
142
143     Alerta a = new Alerta();
144
145     @Override
146     protected void onPreExecute()
147     {
148        dialogo = new ProgressDialog(MainActivity.this);
149        dialogo.setIndeterminate(true);
150        dialogo.setCancelable(false);
151        dialogo.setTitle("Espere...");
152        dialogo.setMessage("Procesando solicitud");
153        dialogo.show();
154     }
155
156     @Override
157     protected void onProgressUpdate(String... strings)
158     {
159        a.error(MainActivity.this, "Error", strings[0]);
160     }
161
162     @Override
163     protected Void doInBackground(String... arg0)
164     {
165
166       matricula = arg0[0];
167       password = arg0[1];
168       String infoac = "";
169       String kardex = "";
170       int servidorActivo = 0;
171
172     try
173     {
174
175       DefaultHttpClient httpclient = new DefaultHttpClient();
176
177
178       List<NameValuePair> params1=new ArrayList<NameValuePair>();
179       params1.add(new BasicNameValuePair("SIGLAS_UNI_XX.E_UNIDAD.AE02.1","AZC"));
180       params1.add(new BasicNameValuePair("%23.E_UNIDAD.AE02.1","AxJDMQ%3D%3D"));
181       params1.add(new BasicNameValuePair("%23CRC.E_UNIDAD.AE02.1","00000024"));
182       params1.add(new
183       BasicNameValuePair("NOMBRE.IDENTIFICACION.NONMODELED",matricula));
184       params1.add(new
185       BasicNameValuePair("COMPLEMENTO.IDENTIFICACION.NONMODELED",password));
186       params1.add(new
187       BasicNameValuePair("GO.IDENTIFICACION.NONMODELED","Entrar"));
188       params1.add(new BasicNameValuePair("%25.IDENTIFICACION.NONMODELED",""));
189       params1.add(new BasicNameValuePair("%23.WEB_INFO.SW01",""));
190       params1.add(new BasicNameValuePair("%23.WEB_MOD_ASO.SW01",""));
191       params1.add(new BasicNameValuePair("%23.USUARIO_ANEXO.SG02",""));
192       params1.add(new BasicNameValuePair("%23.MODULO_UWAS.SAE01",""));
```

```
193
194         HttpPost httpPost = new
195             HttpPost("https://ayamictlan.uam.mx:8443/sae/azc/AEWBU004.oInicioSesion?mod=1");
196             try {
197
198               httpPost.setEntity(new UrlEncodedFormEntity(params1));
199             }
200             catch (UnsupportedEncodingException e)
201             {
202               // TODO Auto-generated catch block
203               e.printStackTrace();
204             }
205               HttpResponse responsePost = null;
206             try {
207               responsePost = httpclient.execute(httpPost);
208             } catch (ClientProtocolException e) {
209               // TODO Auto-generated catch block
210               e.printStackTrace();
211             }
212             catch (IOException e)
213             {
214               // TODO Auto-generated catch block
215               e.printStackTrace();
216             }
217               servidorActivo = responsePost.getStatusLine().getStatusCode();
218
219              if(servidorActivo != 200)
220              {
221                 dialogo.dismiss();
222                 publishProgress("El servidor no se encuentra disponible, favor de intentarlo
                        mas tarde");
223              }
224              else
225              {
226               Header[] cookies = responsePost.getHeaders("Set-Cookie");
227
228               if(cookies.length == 2)
229               {
230                 dialogo.dismiss();
231                 publishProgress("Datos invalidos, favor de verificarlos");
232               }
233               else
234               {
235                 HttpGet httpget = new
236                     HttpGet("https://ayamictlan.uam.mx:8443/sae/azc/IEWBC020.oConsulta");
237
238                   HttpGet httpget2 = new
239                       HttpGet("https://ayamictlan.uam.mx:8443/sae/azc/IEWBC007.oConsulta");
240
241                   for (Header h : cookies) {
242                     httpget.addHeader("Cookie", h.getValue());
243                   }
244
245                   for (Header h : cookies) {
246                     httpget2.addHeader("Cookie", h.getValue());
247                   }
248
249                   HttpResponse responseGet = null;
250                   try {
251                     responseGet = httpclient.execute(httpget);
252                   }
253                   catch (ClientProtocolException e)
254                   {
255                     // TODO Auto-generated catch block
256                     e.printStackTrace();
257                   }
258                   catch (IOException e)
259                   {
```

```
260                 // TODO Auto-generated catch block
261                 e.printStackTrace();
262               }
263
264               HttpEntity ent=responseGet.getEntity();
265
266               try {
267
268                 kardex=EntityUtils.toString(ent);
269               }
270               catch (ParseException ex)
271               {
272                 // TODO Auto-generated catch block
273                 ex.printStackTrace();
274               }
275               catch (IOException e)
276               {
277                 // TODO Auto-generated catch block
278                 e.printStackTrace();
279               }
280
281               try {
282                 responseGet = httpclient.execute(httpget2);
283               }
284               catch (ClientProtocolException es)
285               {
286                 // TODO Auto-generated catch block
287                 es.printStackTrace();
288               }
289               catch (IOException e)
290               {
291                  //TODO Auto-generated catch block
292                 e.printStackTrace();
293               }
294
295               ent=responseGet.getEntity();
296
297               try {
298
299                 infoac=EntityUtils.toString(ent);
300               }
301               catch (ParseException et)
302               {
303                 // TODO Auto-generated catch block
304                 et.printStackTrace();
305               }
306               catch (IOException e)
307               {
308                 // TODO Auto-generated catch block
309                 e.printStackTrace();
310               }
311
312             httpclient.getConnectionManager().shutdown();
313
314       Parsers p = new Parsers();
315
316       //PARSEAMOS LA INFOAC PARA SABER SI RENUNCIO Y METIO RECUPERACION
317       int ren [] = new int[2];
318       boolean alumno;
319       //EL ALUMNO ES DE LA CARRERA
320       alumno = p.is_alumno(infoac);
321
322        if(!alumno)
323        {
324          dialogo.dismiss();
325          publishProgress("Lo siento pero no perteneces a la carrera");
326        }
327        else
```

```
328         {
329           //PARSEAMOS LA INFORMACION ACADEMICA PARA SABER SI RENUNCIO Y METIO RECUPERACION
330           ren = p.is_renuncia(infoac);
331
332           //PARSEAMOS PARA SABER LAS AUTORIZACIONES
333           ArrayList autorizaciones = new ArrayList();
334           autorizaciones = p.hay_autorizaciones(infoac);
335           //PARSEAMOS PARA OBTNER EL TRIMESTRE DE INGRESO
336           String trim_ingreso;
337           trim_ingreso = p.ingreso(infoac);
338           //PARSEAMOS EL KARDEX
339           ArrayList kardex2 = new ArrayList();
340           kardex2 = p.parser_kardex(kardex);
341
342
343           //DETERMINAMOS LOS PARAMETROS NECESARIOS PARA LA RECOMENDACION
344
345           int creditos = 0;
346           int dltrim [] = new int[3];
347           Parametros pr = new Parametros();
348           creditos = pr.creditos_aprobados(kardex2,pln); //CREDITOS ACUMULADOS
349           dltrim = pr.datos_last_trim(kardex2,pln); //PARAMETROS DEL ULTIMO Y TRES ULTIMOS
                  TRIMESTRES
350
351           //COLOCAMOS EN UNA LISTA LAS UEAS DEL KARDEX QUE HAN SIDO APROBADAS
352           //Y EN OTRA LISTA LAS UEAS REPROBADAS
353
354           ArrayList claveskardex = new ArrayList();
355           ArrayList uea_reprobadas = new ArrayList();
356           ArrayList uea_habiles = new ArrayList();
357
358           for(int mm = 0; mm < kardex2.size(); mm++)
359           {
360             if(!((ArrayList<String>) kardex2.get(mm)).get(4).equals("NA"))
361             {
362               claveskardex.add(((ArrayList<String>) kardex2.get(mm)).get(0));
363             }
364             else
365             {
366               uea_reprobadas.add(((ArrayList<String>) kardex2.get(mm)).get(0));
367             }
368           }
369
370           //ORDENAMOS AMBAS LISTAS
371           Collections.sort(claveskardex);
372           Collections.sort(uea_reprobadas);
373
374           //DETERMINAMOS CREDITOS TOTALES O REALES, ASI COMO LOS CREDITOS DE LAS UEA
                  OPTATIVAS
375           //APROBADAS TANTO DE INTEGRACION COMO MULTIDISCIPLINAR
376           int creditos_totales = 0, creditos_ti = 0, creditos_tim = 0;
377           int [] totales = new int[3];
378           totales = pr.cred_totales(claveskardex,pln,creditos);
379           creditos_totales = totales[0];
380           creditos_ti = totales[1];
381           creditos_tim = totales[2];
382
383           //ELIMINAMOS LAS UEA QUE
384           //SE HAN REPROBADO DOS VECES EN EVALUACION GLOBAL Y EL PLAN
385           //ESTABLECE DOS OPORTUNIDADES EN ESA MODALIDAD (SIEMPRE Y CUANDO EL ALUMNO TENGA
                  UEA
386           //REPROBADAS)
387
388           if(uea_reprobadas.size() != 0)
389           {
390             Uea_reprob uap = new Uea_reprob();
391             uap.elimina_reprobadas(uea_reprobadas, claveskardex, plan1, kardex2);
392           }
```

```
393
394             //DETERMINAMOS LAS UEA HABILES Y REMOVEMOS LAS UEA QUE HAN SIDO APROBADAS
395
396             Uea_habilitadas habil = new Uea_habilitadas();
397             uea_habiles = habil.habilitaciones(plan1, claveskardex, creditos, autorizaciones);
398
399             //ELIMINAMOS LAS UEAS QUE NO ESTAN HABILES EN EL PLAN
400
401             Recorta_plan rp = new Recorta_plan();
402             rp.elimina_ueas(plan1,uea_habiles);
403
404             //CALCULAMOS LOS CREDITOS MINIMOS Y MAXIMOS RECOMENDADOS
405             //ASI COMO EL TOPE DADO POR LA UAM Y LO REGRESAMOS EN UN ARREGLO
406
407             int credper [] = new int[3];
408
409             credper = pr.creditos_habilitados(ren,dltrim,creditos_totales,trim_ingreso,
                    uea_habiles,plan1);
410
411             dialogo.dismiss();
412             //LLAMAMOS LA ACTIVIDAD QUE MOSTRARA LA RECOMENDACION
413
414             Intent i = new Intent(MainActivity.this,Recomendacion.class);
415             i.putExtra("creditos", credper);
416             i.putExtra("plan", plan1);
417             i.putExtra("kardex", kardex2);
418             i.putExtra("craprobados", creditos);
419             i.putExtra("aprobadas", claveskardex);
420             i.putExtra("ti", creditos_ti);
421             i.putExtra("tim", creditos_tim);
422             i.putExtra("miplan", pln);
423             MainActivity.this.startActivity(i);
424             finish();
425           }
426          }
427        }
428      }
429      catch(Exception ex)
430      {
431           a.error(MainActivity.this, "Error", "El servidor no se ecnuentra disponible" +
432               " en este momento, intente mas tarde. Si el problema persiste" +
433               " favor de notificarlo a: soportemapp2013@gmail.com");
434      }
435       return null;
436    }
437   }
438 }
```

C.1.2. Validación de conexión a Internet

```
1  package com.example.napp;
2
3  import android.content.Context;
4  import android.net.ConnectivityManager;
5  import android.net.NetworkInfo;
6
7  public class ConnectionDetector {
8
9  private Context context;
10
11   public ConnectionDetector(Context context)
12   {
13     this.context = context;
14   }
15
```

```
16    public boolean isConnectingToInternet(){
17      ConnectivityManager cm = (ConnectivityManager) context.getSystemService(Context.
            CONNECTIVITY_SERVICE);
18      if(cm != null)
19      {
20        NetworkInfo[] info = cm.getAllNetworkInfo();
21
22        if(info != null)
23        {
24          for(int i = 0; i < info.length; i++)
25          {
26            if(info[i].getState() == NetworkInfo.State.CONNECTED)
27            {
28              return true;
29            }
30          }
31        }
32      }
33      return false;
34    }
35  }
```

C.1.3. Validación de formato de contraseña y despliegue de alertas

```
1  package com.example.napp;
2
3  import java.util.regex.Matcher;
4  import java.util.regex.Pattern;
5
6  import android.app.AlertDialog;
7  import android.content.Context;
8  import android.content.DialogInterface;
9
10 public class Alerta {
11
12   void Alerta()
13   {
14
15   }
16
17   public void error(Context context, String title, String message) {
18     // TODO Auto-generated method stub
19     AlertDialog alertDialog = new AlertDialog.Builder(context).create();
20
21         alertDialog.setTitle(title);
22         alertDialog.setMessage(message);
23         alertDialog.setButton("OK", new DialogInterface.OnClickListener() {
24             public void onClick(DialogInterface dialog, int which) {
25
26             }
27         });
28
29         alertDialog.show();
30   }
31
32   public boolean valida_contrasenia(String password) {
33     // TODO Auto-generated method stub
34     Pattern p = Pattern.compile("[A-Z]");
35     Pattern b = Pattern.compile("[a-z]");
36     Pattern s = Pattern.compile("[0-9]");
37     Matcher m = p.matcher(password);
38     Matcher n = b.matcher(password);
39     Matcher o = s.matcher(password);
40
41     if(m.find() && n.find() && o.find())
```

```
42      {
43        return true;
44      }
45
46        return false;
47      }
48  }
```

C.1.4. Despliegue de recomendación

```
1  package com.example.napp;
2
3  import java.util.ArrayList;
4  import java.util.Iterator;
5
6  import android.app.Activity;
7  import android.content.Intent;
8  import android.content.res.Resources;
9  import android.os.Bundle;
10 import android.view.Gravity;
11 import android.view.View;
12 import android.view.Window;
13 import android.view.WindowManager;
14 import android.widget.Button;
15 import android.widget.EditText;
16 import android.widget.ListView;
17 import android.widget.RadioGroup;
18 import android.widget.TableRow;
19 import android.widget.RadioGroup.OnCheckedChangeListener;
20 import android.widget.SeekBar;
21 import android.widget.SeekBar.OnSeekBarChangeListener;
22 import android.widget.TableLayout;
23 import android.widget.TextView;
24 import android.widget.Toast;
25
26 import com.example.despliegues.ItemTexto;
27 import com.example.despliegues.ItemTextoAdapter;
28 import com.example.logica.Busqueda_binaria;
29 import com.example.recomendacion.Prioridad;
30 import com.example.recomendacion.Recomienda;
31
32 public class Recomendacion extends Activity{
33
34   Recomienda rec = new Recomienda();
35   RadioGroup opcion;
36   TextView minimo, maximo, acumulados, ajustados,ajustados_reales;
37   SeekBar ajuste;
38   TextView texto;
39   ListView lista;
40   ArrayList<ItemTexto> ueas ;
41   ItemTextoAdapter adaptador;
42
43   static int tope, opti, optim;
44   static ArrayList recomendacion = new ArrayList();
45   static ArrayList plan = new ArrayList();
46   static ArrayList mplan = new ArrayList();
47   static ArrayList kardex = new ArrayList();
48   static ArrayList prioridad = new ArrayList();
49   static ArrayList prioridad2 = new ArrayList();
50   static ArrayList aprobadas = new ArrayList();
51   static ArrayList intaprob = new ArrayList();
52   int creditos_ajustados = 0;
53   String ajuste_vista;
54
55   @Override
```

```
56  protected void onCreate(Bundle savedInstanceState) {
57    super.onCreate(savedInstanceState);
58    this.requestWindowFeature(Window.FEATURE_NO_TITLE);
59    getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_HIDDEN);
60    setContentView(R.layout.recomendacion);
61
62
63    Bundle extras = getIntent().getExtras();
64    int crpermitidos[] = new int[3];
65    Alerta a = new Alerta();
66    int min, max, aprobados;
67
68    ajuste = (SeekBar)findViewById(R.id.seekBar1);
69    ajustados = (TextView)findViewById(R.id.textView9);
70    minimo = (TextView)findViewById(R.id.textView3);
71    maximo = (TextView)findViewById(R.id.textView5);
72    acumulados = (TextView)findViewById(R.id.textView7);
73    opcion = (RadioGroup)findViewById(R.id.radioGroup1);
74    lista = (ListView)findViewById(R.id.recomendacion);
75    ajustados_reales = (TextView)findViewById(R.id.rajuste);
76
77    crpermitidos = extras.getIntArray("creditos");
78    aprobados = extras.getInt("craprobados");
79    plan = extras.getStringArrayList("plan");
80    kardex = extras.getStringArrayList("kardex");
81    prioridad = extras.getIntegerArrayList("prioridades");
82    aprobadas = extras.getStringArrayList("aprobadas");
83    opti = extras.getInt("ti");
84    optim = extras.getInt("tim");
85    mplan = extras.getStringArrayList("miplan");
86
87    min = crpermitidos[0];
88    max = crpermitidos[1];
89    tope = crpermitidos[2];
90
91    maximo.setText(Integer.toString(max));
92    minimo.setText(Integer.toString(min));
93    acumulados.setText(Integer.toString(aprobados));
94
95    ajuste.setMax(tope);
96
97
98      //ORDENAMOS LAS PRIORIDADES DE LAS UEA HABILITADAS
99
100     Prioridad orden = new Prioridad();
101     prioridad2 = orden.ordena(plan);
102
103     //OBTENEMOS LOS CREDITOS DE CADA UEA OPTATIVA DE INTEGRACION DEL ALUMNO PARA CONSTRUIR
            LA RECOMENDACION
104
105     intaprob = orden.integracion(aprobadas,mplan);
106
107     //CONSTRUIMOS LA RECOMENDACION
108
109     int recomendados;
110     recomendados = max;
111
112     //MANDAMOS MOSTRAR LA RECOMENDACION
113     construir_recomendacion(recomendados);
114     a.error(Recomendacion.this, "Aviso", "Es probable que no siempre los creditos se
            ajusten" +
115         " exactamente a los creditos recomendados o ajustados");
116
117     /////MODIFICAR CREDITOS CON SEEKBAR
118
119     ajuste.setOnSeekBarChangeListener(new OnSeekBarChangeListener(){
120
121     @Override
```

```
122         public void onProgressChanged(SeekBar seekBar, int progress,
123             boolean fromUser) {
124           // TODO Auto-generated method stub
125
126           creditos_ajustados = progress;
127           ajuste_vista = String.valueOf(creditos_ajustados);
128           ajustados.setText(ajuste_vista);
129         }
130
131         @Override
132         public void onStartTrackingTouch(SeekBar seekBar) {
133           // TODO Auto-generated method stub
134
135         }
136
137         @Override
138         public void onStopTrackingTouch(SeekBar seekBar) {
139           // TODO Auto-generated method stub
140           //SE LLAMA LA FUNCION QUE CONSTRUYE LA RECOMENDACION
141           construir_recomendacion(creditos_ajustados);
142         }
143
144       });
145
146       ///OPCIONES
147
148       opcion.setOnCheckedChangeListener(new OnCheckedChangeListener(){
149
150         public void onCheckedChanged(RadioGroup arg0, int cid) {
151                   // TODO Auto-generated method stub
152
153           if(cid == R.id.radio2)
154           {
155             Intent i = new Intent(Recomendacion.this,Opina.class);
156             Recomendacion.this.startActivity(i);
157           }
158           else if(cid == R.id.radio1)
159           {
160             //MOSTRAMOS LAS UEA TI HABILES A INSCRIBIR
161             Intent i = new Intent(Recomendacion.this,Integracion.class);
162             i.putExtra("plan",plan);
163             Recomendacion.this.startActivity(i);
164           }
165           else
166           {
167             //MOSTRAMOS LAS UEA TIM HABILES A INSCRIBIR
168             Intent i = new Intent(Recomendacion.this,Multidisciplinares.class);
169             i.putExtra("plan",plan);
170             Recomendacion.this.startActivity(i);
171           }
172               }
173
174       });
175
176     }
177
178     private void construir_recomendacion(int recomendados) {
179       // TODO Auto-generated method stub
180
181
182       ArrayList recomendada = new ArrayList();
183       recomendada = rec.arma_recomendacion(recomendados, prioridad2,plan,tope,aprobadas,opti,
              optim,intaprob);
184
185       //SE MANDA A CONSTRUIR EL LISTVIEW CON LAS UEA RECOMENDADAS
186
187       ueas = muestra_recomendacion(recomendada);
188       adaptador = new ItemTextoAdapter(this, ueas);
```

```
189     lista.setAdapter(adaptador);
190     adaptador.notifyDataSetChanged();
191
192     return;
193   }
194
195   private ArrayList<ItemTexto> muestra_recomendacion(ArrayList recomendada) {
196
197     String datos_uea[] = new String[2];
198     String clave;
199     String tu_recomendacion[][] = new String[recomendada.size()][3];
200     int obligatoria = 0, creditosAjustados = 0;
201     Busqueda_binaria busqueda = new Busqueda_binaria();
202     ArrayList<ItemTexto> items = new ArrayList<ItemTexto>();
203     items.add(new ItemTexto("Clave","\t UEA","Creditos"));
204
205
206     for(int i = 0; i < recomendada.size(); i++)
207     {
208       clave = (String) recomendada.get(i);
209       if(clave.equals("TIM")){
210         tu_recomendacion[i][0] = "11XXXX ";
211         tu_recomendacion[i][1] = "Optativa Inter-Multidisciplinar";
212         tu_recomendacion[i][2] = "6";
213       }
214       else if(clave.contains(":"))
215       {
216           tu_recomendacion[i][0] = "11XXXX ";
217           tu_recomendacion[i][1] = "Optativa de Integracion";
218           tu_recomendacion[i][2] = clave.substring(3);
219       }
220       else
221       {
222         obligatoria = Integer.parseInt(clave);
223         datos_uea = busqueda.uea_recomendada(obligatoria, plan);
224         tu_recomendacion[i][0] = clave;
225         tu_recomendacion[i][1] = datos_uea[0];
226         tu_recomendacion[i][2] = datos_uea[1];
227       }
228
229       creditosAjustados += Integer.parseInt(tu_recomendacion[i][2]);
230       items.add(new ItemTexto(tu_recomendacion[i][0],tu_recomendacion[i][1],tu_recomendacion
              [i][2]));
231     }
232
233     ajustados_reales.setText(String.valueOf(creditosAjustados).toString());
234
235     return items;
236   }
237 }
```

C.1.5. Despliegue de Optativas Multidisciplinares

```
1  package com.example.napp;
2
3  import java.util.ArrayList;
4  import java.util.Iterator;
5
6  import com.example.despliegues.ItemTextoAdapter;
7  import com.example.despliegues.ItemTexto;
8
9  import android.app.Activity;
10 import android.app.AlertDialog;
11 import android.content.Context;
12 import android.content.DialogInterface;
```

```
13 import android.content.res.Resources;
14 import android.graphics.Color;
15 import android.os.Bundle;
16 import android.text.method.ScrollingMovementMethod;
17 import android.view.Gravity;
18 import android.view.Window;
19 import android.view.WindowManager;
20 import android.widget.ListView;
21
22 import android.widget.TextView;
23
24 public class Multidisciplinares extends Activity{
25
26 private ArrayList planes = new ArrayList();
27
28   ListView lista;
29   ArrayList<ItemTexto> ueas ;
30   ItemTextoAdapter adaptador;
31
32   @Override
33   protected void onCreate(Bundle savedInstanceState) {
34     super.onCreate(savedInstanceState);
35     this.requestWindowFeature(Window.FEATURE_NO_TITLE);
36     getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_HIDDEN);
37     setContentView(R.layout.multidisciplinares);
38
39     Bundle extras = getIntent().getExtras();
40     planes = extras.getStringArrayList("plan");
41
42     lista = (ListView)findViewById(R.id.multidisciplinares);
43
44     ueas = agregar_filas(planes);
45     adaptador = new ItemTextoAdapter(this, ueas);
46     lista.setAdapter(adaptador);
47     adaptador.notifyDataSetChanged();
48   }
49
50   private ArrayList<ItemTexto> agregar_filas(ArrayList planes) {
51
52     String ueas[] = new String[3];
53     ArrayList<ItemTexto> items = new ArrayList<ItemTexto>();
54     items.add(new ItemTexto("Clave","\t UEA","Creditos"));
55
56     for(int j = 0; j < planes.size(); j++)
57     {
58
59       if(((ArrayList<String>) planes.get(j)).get(3).equals("TIM") &&
60           (((ArrayList<String>) planes.get(j)).get(4).equals("OPT")))
61       {
62         ueas[0] = ((ArrayList<String>) planes.get(j)).get(0);
63         ueas[1] = ((ArrayList<String>) planes.get(j)).get(1);
64         ueas[2] = ((ArrayList<String>) planes.get(j)).get(2);
65
66         items.add(new ItemTexto(ueas[0],ueas[1],ueas[2]));
67       }
68     }
69
70     return items;
71   }
72 }
```

C.1.6. Despliegue de Optativas de Integración

```
1 package com.example.napp;
2
```

```
3  import java.util.ArrayList;
4  import java.util.Iterator;
5
6  import com.example.despliegues.ItemTextoAdapter;
7  import com.example.despliegues.ItemTexto;
8
9  import android.app.Activity;
10 import android.app.AlertDialog;
11 import android.content.Context;
12 import android.content.DialogInterface;
13 import android.content.res.Resources;
14 import android.graphics.Color;
15 import android.os.Bundle;
16 import android.text.method.ScrollingMovementMethod;
17 import android.view.Gravity;
18 import android.view.Window;
19 import android.view.WindowManager;
20 import android.widget.ListView;
21
22 import android.widget.TextView;
23
24 public class Integracion extends Activity{
25
26 private ArrayList planes = new ArrayList();
27
28   ListView lista;
29   ArrayList<ItemTexto> ueas ;
30   ItemTextoAdapter adaptador;
31
32   @Override
33   protected void onCreate(Bundle savedInstanceState) {
34     super.onCreate(savedInstanceState);
35     this.requestWindowFeature(Window.FEATURE_NO_TITLE);
36     getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_HIDDEN);
37     setContentView(R.layout.integracion);
38
39     Bundle extras = getIntent().getExtras();
40     planes = extras.getStringArrayList("plan");
41
42     lista = (ListView)findViewById(R.id.integracion);
43
44     ueas = agregar_filas(planes);
45     adaptador = new ItemTextoAdapter(this, ueas);
46     lista.setAdapter(adaptador);
47     adaptador.notifyDataSetChanged();
48   }
49
50   private ArrayList<ItemTexto> agregar_filas(ArrayList planes) {
51
52     String ueas[] = new String[3];
53     ArrayList<ItemTexto> items = new ArrayList<ItemTexto>();
54     items.add(new ItemTexto("Clave","\t UEA","Creditos"));
55
56     for(int j = 0; j < planes.size(); j++)
57     {
58
59       if(((ArrayList<String>) planes.get(j)).get(3).equals("TI") &&
60           (((ArrayList<String>) planes.get(j)).get(4).equals("OPT")))
61       {
62         ueas[0] = ((ArrayList<String>) planes.get(j)).get(0);
63         ueas[1] = ((ArrayList<String>) planes.get(j)).get(1);
64         ueas[2] = ((ArrayList<String>) planes.get(j)).get(2);
65
66         items.add(new ItemTexto(ueas[0],ueas[1],ueas[2]));
67       }
68     }
69
70     return items;
```

```
71      }
72  }
```

C.1.7. Envió de comentarios

```
1  package com.example.napp;
2
3  import android.app.Activity;
4  import android.content.Intent;
5  import android.net.Uri;
6  import android.os.Bundle;
7  import android.view.Gravity;
8  import android.view.View;
9  import android.view.Window;
10 import android.view.WindowManager;
11 import android.widget.Button;
12 import android.widget.EditText;
13 import android.widget.RadioGroup;
14 import android.widget.Toast;
15
16 public class Opina extends Activity{
17
18   Button enviar;
19   EditText mensaje;
20
21   @Override
22   protected void onCreate(Bundle savedInstanceState) {
23     super.onCreate(savedInstanceState);
24     this.requestWindowFeature(Window.FEATURE_NO_TITLE);
25     getWindow().setSoftInputMode(WindowManager.LayoutParams.SOFT_INPUT_STATE_ALWAYS_HIDDEN);
26     setContentView(R.layout.opina);
27
28     enviar = (Button)findViewById(R.id.button1);
29     mensaje = (EditText)findViewById(R.id.comentario);
30
31     enviar.setOnClickListener(new View.OnClickListener() {
32
33       @Override
34       public void onClick(View v) {
35         // TODO Auto-generated method stub
36
37         String mens;
38         Alerta a = new Alerta();
39         mens = mensaje.getText().toString();
40
41         if(mens.isEmpty())
42         {
43           a.error(Opina.this, "Error", "Debes escribir un mensaje");
44         }
45         else
46         {
47           String para [] = {"tph@correo.azc.uam.mx"};
48           String cc[] = {"soportemapp2013@gmail.com"};
49           Intent email = new Intent(Intent.ACTION_SEND);
50           email.setData(Uri.parse("mailto:"));
51           email.putExtra(Intent.EXTRA_EMAIL, para);
52           email.putExtra(Intent.EXTRA_CC, cc);
53           email.putExtra(Intent.EXTRA_SUBJECT, "Opinion acerca de Mapp");
54           email.putExtra(Intent.EXTRA_TEXT, mens);
55           email.setType("message/rfc822");
56           startActivity(Intent.createChooser(email, "Elige una aplicacion de correo
                 electronico"));
57         }
58
59         mensaje.setText("");
60       }
61     });
62   }
63 }
```

C.1.8. Parser

```
1  package com.example.logica;
2
3  import java.util.ArrayList;
4  import java.util.Stack;
5
6  import org.jsoup.Jsoup;
7  import org.jsoup.nodes.Document;
8  import org.jsoup.nodes.Element;
9  import org.jsoup.select.Elements;
10
11 public class Parsers {
12
13   public int[] is_renuncia(String iaca) {
14     String renuncia = "";
15     int para[] = new int[2];
16     Document doc = Jsoup.parse(iaca);
17     Stack st = new Stack();
18     Elements tabla1 = doc.select("div#tab5");
19     Elements t2 = tabla1.select("td");
20
21     for(int i = 1; i < t2.size(); i+=12)
22     {
23       Element row = t2.get(i);
24       st.push(row.text());
25     }
26
27         renuncia = st.pop().toString();
28         para[0] = Integer.parseInt(renuncia);
29         para[1] = 0;
30
31         return para;
32   }
33
34   public boolean is_alumno(String infoac) {
35
36     if((infoac.indexOf("LICENCIATURA EN INGENIERIA MECANICA") != -1) || (infoac.indexOf("
         Licenciatura en Ingenieria Mecanica") != -1))
37     {
38       return true;
39     }
40
41     return false;
42   }
43
44   public ArrayList hay_autorizaciones(String infoac) {
45     // TODO Auto-generated method stub
46     ArrayList autoriza = new ArrayList();
47
48     Document doc = Jsoup.parse(infoac);
49     Elements t1 = doc.select("div#tab6");
50     Elements r2 = t1.select("td");
51
52     for(int i = 0 ; i< r2.size(); i+=4)
53     {
54       Element r = r2.get(i);
55       autoriza.add(r.text());
56
57     }
58     return autoriza;
59   }
60
61   public String ingreso(String infoac) {
62     // TODO Auto-generated method stub
63     String ingreso;
64
65     Document doc = Jsoup.parse(infoac);
```

```
66      Elements t1 = doc.select("div#estructura_tab");
67      Elements t2 = t1.select("td").select(":contains(Trimestre de Ingreso:)");
68      Element tim = t2.first();
69
70      ingreso = tim.text().replace("Trimestre de Ingreso:", "").trim();
71
72      return ingreso;
73    }
74
75    public ArrayList parser_kardex(String mikardex) {
76      // TODO Auto-generated method stub
77
78
79      ArrayList kardex = new ArrayList();
80      String [][] kar = new String[100][5];
81
82      int h = 0, incremento = 8;
83      Document doc = Jsoup.parse(mikardex);
84      Elements k = doc.select("div#div_barra4");
85      Elements t1 = k.select("td");
86
87      for(int i = 1; i < t1.size(); i+=incremento)
88      {
89        Element row = t1.get(i);
90        kar[h][0] = row.text();
91        h++;
92      }
93
94      h=0;
95
96      for(int i = 2; i < t1.size(); i+=incremento)
97      {
98        Element row2 = t1.get(i);
99        kar[h][1] = row2.text();
100       h++;
101     }
102
103     h=0;
104
105     for(int j = 3; j < t1.size(); j+=incremento)
106     {
107       Element r1 = t1.get(j);
108       kar[h][2] = r1.text();
109       h++;
110     }
111
112     h=0;
113
114     for(int j = 4; j < t1.size(); j+=incremento)
115     {
116       Element r2 = t1.get(j);
117       kar[h][3] = r2.text();
118       h++;
119     }
120
121     h=0;
122
123     for(int j = 5; j < t1.size(); j+=incremento)
124     {
125       Element r3 = t1.get(j);
126       kar[h][4] = r3.text();
127       h++;
128     }
129
130
131     for(int s = 0; s < h; s++)
132     {
133       kardex.add(new ArrayList());
```

```
134         for(int t = 0; t < 5; t++)
135         {
136           ((ArrayList)kardex.get(s)).add(kar[s][t]);
137         }
138
139       }
140
141       return kardex;
142     }
143 }
```

C.1.9. Almacenamiento del Plan de Estudios

```
1  package com.example.logica;
2
3  import java.io.BufferedReader;
4  import java.io.IOException;
5  import java.io.InputStream;
6  import java.io.InputStreamReader;
7  import java.util.ArrayList;
8  import java.util.StringTokenizer;
9  import android.content.res.AssetManager;
10
11 public class Cargar_plan {
12
13   void Cargar_plan()
14   {
15
16   }
17
18   public ArrayList cargar_plan(AssetManager am)  throws IOException {
19     // TODO Auto-generated method stub
20
21      //ArrayList plan1 = new ArrayList();
22     String linea=null;;
23     int k = 0;
24     int i = 0;
25     ArrayList plan1 = new ArrayList();
26
27     InputStream is = null;
28
29     is = am.open("plan/plan_mecanica6.txt");
30     if(is!=null)
31     {
32       InputStreamReader isr = new InputStreamReader(is,"ISO-8859-15");
33       BufferedReader br = new BufferedReader(isr);
34
35         while((linea=br.readLine())!=null && k < 168)
36             {
37               plan1.add(new ArrayList());
38               StringTokenizer token = new StringTokenizer(linea,"\t");
39
40               while(token.hasMoreTokens() && i < 9)
41               {
42
43                 ((ArrayList<String>) plan1.get(k)).add(token.nextToken());
44                 i++;
45               }
46               i = 0;
47               k++;
48             }
49             br.close();
50     }
51     return plan1;
52   }
```

```
}
```

C.1.10. Determinación de los parámetros necesarios

```
package com.example.logica;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;

public class Parametros {

  public Parametros()
  {

  }

  public int creditos_aprobados(ArrayList kardex, ArrayList plan1) {
    // TODO Auto-generated method stub

    int i = 0, creditos = 0, ct = 0;
    int uk;
    Busqueda_binaria b = new Busqueda_binaria();
    while(i < kardex.size())
    {
      if(!((ArrayList<String>) kardex.get(i)).get(4).equals("NA"))
      {
        uk = Integer.parseInt(((ArrayList<String>) kardex.get(i)).get(0));
        creditos = b.b_uea(uk, plan1);
        ct += creditos;
        i++;
      }
      else
      {
        i++;
      }
    }

    return ct;
  }

  public int[] datos_last_trim(ArrayList kardex, ArrayList plan1) {
    // TODO Auto-generated method stub

    String [][] kr = new String[kardex.size()][5];
    Busqueda_binaria c = new Busqueda_binaria();

    //COPIAMOS EL KARDEX EN UNA MATRIZ
    for(int y = 0; y < kardex.size(); y++)
    {
      for(int r = 0; r < 5; r++)
      {
        kr[y][r] = ((ArrayList<String>) kardex.get(y)).get(r);
      }
    }

     ///RECORREMOS DE ATRAS HACIA ADELANTE EL KARDEX Y CONTAMOS LAS CALIFICACIONES Y
        BUSCAMOS LOS CREDITOS DE
     //ESAS UEA EN EL PLAN

     int plt[] = new int[3];
     int y = kr.length - 1;
     int mb = 0, b = 0, s = 0, na = 0, clt = 0, calt = 0, claveuea = 0, numerodeUEA=0;

```

```
61        //clt = creditos de UEA del ultim trimestre
62        //calt = creditos inscritos de los ultimos tres trimestres
63
64        while(y >= 0)
65        {
66
67            if((kr[y][2].equals(kr[y-1][2]) && (kr[y][2].equals(kr[kr.length-1][2])))
68              || (kr[y][2].equals(kr[kr.length-1][2])))
69            {
70                 claveuea = Integer.parseInt(kr[y][0]);
71
72                 if(kr[y][4].equals("MB"))
73                 {
74                         mb++;
75                         clt = c.b_uea(claveuea,plan1);
76                         calt += clt;
77                         y--;
78                 }
79                 else if(kr[y][4].equals("B"))
80                 {
81                         b++;
82                         clt = c.b_uea(claveuea,plan1);
83                         calt += clt;
84                         y--;
85                 }
86                 else if(kr[y][4].equals("S"))
87                 {
88                         s++;
89                         clt = c.b_uea(claveuea,plan1);
90                         calt += clt;
91                         y--;
92                 }
93                 else
94                 {
95                         na++;
96                         clt = c.b_uea(claveuea,plan1);
97                         calt += clt;
98                         y--;
99                 }
100            }
101            else
102            {
103                 y = -1;
104            }
105       }
106
107       numerodeUEA = (na+s+b+mb);
108
109
110       //EL NUMERO DE MB ES MAYOR A LA SUMA DE B's y S's o TODAS LAS UEAS INSCRITAS SE
              APRBARON CON B
111       if(mb > (s+b) && (calt >= 32) || ((calt >= 32) && b == numerodeUEA))
112       {
113            plt[0] = 1; //SI ESTO ES VERDAD ES POSIBLE QUE PUEDA METER EXCESO
114       }
115       else
116       {
117            plt[0] = 0;
118       }
119
120       //DETERMINAMOS LOS CREDITOS APROBADOS DE LOS ULTIMOS TRES TRIMESTRES
121       int w = kr.length-1;
122       int t = 1, cred = 0, cac = 0;
123       int clave = 0;
124
125       while((w>=0) && (t < 4))
126       {
127            if(!(kr[w][4].equals("NA")))
```

```
128             {
129                 clave = Integer.parseInt(kr[w][0]);
130 
131                 if(kr[w][2].equals(kr[w-1][2]))
132                 {
133                       cred = c.b_uea(clave, plan1);
134                       cac += cred;
135                       w--;
136                 }
137                 else
138                 {
139                       cred = c.b_uea(clave, plan1);
140                       cac += cred;
141                       t++;
142                       w--;
143                 }
144             }
145             else
146             {
147                 if(kr[w][2].equals(kr[w-1][2]) && t < 4)
148                 {
149                       w--;
150                 }
151                 else
152                 {
153                       w--;
154                       t++;
155                 }
156             }
157       }
158 
159     plt[2] = cac;
160 
161     return plt;
162 }
163 
164   public int[] creditos_habilitados(int[] ren, int[] dltrim, int creditos,
165       String trim_ingreso, ArrayList uea_habiles, ArrayList plan1) {
166     // TODO Auto-generated method stub
167 
168     //Si renuncio es = 1
169     //Si metio recuperacion = 1
170     //Si sao mas MB que B y S es = 1
171     //Si metio al menos la mitad de los creditos normales es = 1
172     //dltrim[2] son los creitos acumulados en los utlimos tres trimestres
173 
174     String  season,fec;
175     int ñaoingreso,mesActual,ñaoActual,ñaostrans,trimtrans,dia;
176     int cmaxt = 0;
177     int cmint2 = 0, tope, peso = 9, crmin, crmax;
178     double  cmint = 0;ñ
179 
180     aoingreso = Integer.parseInt(trim_ingreso.substring(0, 2));
181     season = trim_ingreso.substring(2);
182     int [] crpos = new int[3];
183 
184     Date fecha = new Date();
185     SimpleDateFormat formato = new SimpleDateFormat("dd/MM/yy");
186 
187     fec = formato.format(fecha);
188 
189     dia = Integer.parseInt(fec.substring(0, 2));
190     mesActual = Integer.parseInt(fec.substring(3, 5));ñ
191     aoActual = Integer.parseInt(fec.substring(6,8));
192 
193     trimtrans = trimestres_transcurridos(season,dia,mesActual,ñaoActual,ñaoingreso);
194     tope = calcula_tope(ren,dltrim);
195 
```

```
196     if(trimtrans <= 2)
197     {
198       crmin = creditos / trimtrans;
199       crmax = crmin;
200
201       if(crmin > tope)
202       {
203         crmin = tope;
204         crmax = tope;
205       }
206     }
207     else
208     {
209       ///VERIFICAMOS EL MAXIMO QUE ES EL PROMEDIO MOVIL
210       cmaxt = dltrim[2] / 3;
211       if(trimtrans < 12)
212       {
213         cmint = ((double)(491 - creditos) / (12 - trimtrans));
214       }
215       else if(trimtrans < 18)
216       {
217         cmint = ((double)(491 - creditos) / (18 - trimtrans));
218       }
219       else if(trimtrans < 24)
220       {
221         cmint = ((double)(491 - creditos) / (24 - trimtrans));
222       }
223       else
224       {
225         cmint = ((double)(491 - creditos) / (30 - trimtrans));
226       }
227
228       //DETERMINAMOS EL MAXIMO Y MINIMO RECOMENDADO
229
230       if(cmaxt < cmint2)
231       {
232         if(cmaxt == 0 && cmint2 < tope)
233         {
234           cmaxt = tope;
235         }
236         else if(cmaxt == 0 && cmint2 > tope)
237         {
238           cmaxt = tope;
239           cmint = tope;
240         }
241         else if(cmaxt < 18 && cmint2 < tope)
242         {
243           cmint2 = ((cmaxt + cmint2)/2);
244           if(cmint2 > tope)
245           {
246             cmint2 = tope;
247             cmaxt = cmint2;
248           }
249           cmaxt = cmint2;
250         }
251         else if(cmaxt < 18 && cmint > tope)
252         {
253           cmint2 = ((cmaxt + cmint2)/2);
254           if(cmint2 > tope)
255           {
256             cmint2 = tope;
257           }
258           cmaxt = tope;
259         }
260         else
261         {
262           if(cmint2 < tope)
263           {
```

```
264             cmaxt = cmint2 + peso;
265
266             if(cmaxt > tope)
267             {
268               cmaxt = tope;
269             }
270           }
271           else
272           {
273             cmaxt = tope;
274             cmint2 = tope;
275           }
276         }
277       }
278       else if(cmint2 == 0)
279       {
280         cmint2 = 0;
281         cmaxt = 0;
282       }
283       else if(cmint2 == cmaxt)
284       {
285         if(cmint2 > tope)
286         {
287           cmint2 = tope;
288           cmaxt = tope;
289         }
290         else
291         {
292           cmaxt = cmint2 + peso;
293
294           if(cmaxt > tope)
295           {
296             cmaxt = tope;
297           }
298         }
299       }
300       else
301       {
302         cmaxt = cmint2 + peso;
303
304         if(cmaxt > tope)
305         {
306           cmaxt = tope;
307         }
308       }
309     }
310
311     crmin = cmint2;
312     crmax = cmaxt;
313
314     crpos[0] = crmin;
315     crpos[1] = crmax;
316     crpos[2] = tope;
317
318     return crpos;
319   }
320
321   private int calcula_tope(int[] ren, int[] dltrim) {
322     // TODO Auto-generated method stub
323     //VERIFICAMOS EL TOPE DE CREDITOS PERMITIDOS POR LA UNIVERSIDAD PUEDE SER 40 O 60
324     int tope;
325     if(ren[0] == 0 && dltrim[0] == 1 && dltrim[1] == 1)
326     {
327       tope = 63;
328     }
329     else
330     {
331       tope = 40;
```

```
332     }
333
334     return tope;
335   }
336
337   private int trimestres_transcurridos(String season, int mes_anio,int anioActual, int
        anioingreso) {
338     // TODO Auto-generated method stub
339     int tim_cursado = 0;
340
341     if((mesActual >= 3 && mesActual <= 7)) //TRIM DE PRIMABVERA
342     {
343           if((mesActual == 7 && dia <= 11) || mesActual == 3 && dia > 21) //VERIFICAMOS LOS
                DIAS EN QUE EMPIEZA Y TERMINA EL TRIMESTRE
344           {
345                 if(season.equals("P"))
346                 {
347                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 2 + 2;
348                 }
349                 else
350                 {
351                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 2 + 1;
352                 }
353           }
354           else if(mesActual == 7 && dia > 11)//VERIFICAMOS LOS DIAS EN QUE EMPIEZA Y TERMINA
                EL TRIMESTRE
355           {
356                 if(season.equals("P"))
357                 {
358                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 1 + 2;
359                 }
360                 else
361                 {
362                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 1 + 1;
363                 }
364           }
365           else
366           {
367                 if(season.equals("P"))
368                 {
369                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 2 + 2;
370                 }
371                 else
372                 {
373                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 2 + 1;
374                 }
375           }
376     }
377
378     if((mesActual >= 8 && mesActual < 12)) //TRIM ÑOTOO
379     {
380           if((mesActual == 8 && dia > 26) || (mesActual == 11 && dia < 19))//VERIFICAMOS LOS
                DIAS EN QUE EMPIEZA Y TERMINA EL TRIMESTRE
381           {
382                 if(season.equals("P"))
383                 {
384                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 1 + 2;
385                 }
386                 else
387                 {
388                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 1 + 1;
389                 }
390           }
391           else
392           {
393                 if(season.equals("P"))
394                 {
395                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 1 + 2;
```

```
396                 }
397                 else
398                 {
399                       tim_cursado = ((ñaoActual-ñaoingreso)*3) - 1 + 1;
400                 }
401           }
402     }
403
404     if((mesActual >= 1 && mesActual <= 3) || (mesActual == 11 && dia > 19) || (mesActual ==
          12))//TRIM DE INVIERNO
405     {
406           if((mesActual == 3 && dia <= 28)) //VERIFICAMOS LOS DIAS EN QUE EMPIEZA Y TERMINA
                EL TRIMESTRE
407           {
408                if(season.equals("P"))
409                {
410                     tim_cursado = ((ñaoActual-ñaoingreso)*3) + 1;
411                }
412                else
413                {
414                     tim_cursado = ((ñaoActual-ñaoingreso)*3) - 2;
415                }
416           }
417           else if((mesActual == 11 && dia > 19) || (mesActual == 12)) //VERIFICAMOS LOS DIAS
                EN QUE EMPIEZA Y TERMINA EL TRIMESTRE
418           {
419                if(season.equals("P"))
420                {
421                     tim_cursado = ((ñaoActual-ñaoingreso)*3) + 2;
422                }
423                else
424                {
425                     tim_cursado = ((ñaoActual-ñaoingreso)*3) + 1;
426                }
427           }
428           else
429           {
430                if(season.equals("P"))
431                {
432                    tim_cursado = ((ñaoActual-ñaoingreso)*3) - 1;
433                }
434                else
435                {
436                    tim_cursado = ((ñaoActual-ñaoingreso)*3) - 2;
437                }
438           }
439      }
440
441      return tim_cursado;
442 }
443
444   public int[] cred_totales(ArrayList claveskardex, ArrayList pln, int creditos) {
445     // TODO Auto-generated method stub
446
447     int indice = 0, ckardex=0, cred_op_tim = 0, ctotales = 0, cred_op_ti = 0;
448     int credop [] = new int[2];
449     Busqueda_binaria bb = new Busqueda_binaria();
450     int finales[] = new int[3];
451
452     while(indice < claveskardex.size())
453     {
454       ckardex = Integer.parseInt((String) claveskardex.get(indice));
455       //FUNCION QUE BUSCA LA UEA Y VERIFICA SI ES OPTATIVA Y DE QUE TRONCO ES
456       //TI = INTEGRACION o TIM = INTERMULTIDISCIPLINAR
457       credop = bb.cred_optativas(ckardex,pln);
458
459       if(credop[0] == 1)
460       {
```

```
461           cred_op_ti += credop[1];
462         }
463
464         if(credop[0] == 2)
465         {
466           cred_op_tim += credop[1];
467         }
468         indice++;
469       }
470
471       //DETERMINAMOS LOS CREDITOS REALES DESPUES DE SABER LOS CREDITOS ACUMULADOS
472       //DE LAS UEA OPTATIVAS DE CADA TRONCO
473
474       if((cred_op_ti > 69) && (cred_op_tim > 24))
475       {
476         ctotales = ((creditos - (cred_op_ti - 69)) - (cred_op_tim - 24));
477       }
478       else if((cred_op_ti > 69) && (cred_op_tim <= 24))
479       {
480         ctotales = (creditos - (cred_op_ti - 69));
481       }
482       else if((cred_op_ti <= 69) && (cred_op_tim > 24))
483       {
484         ctotales = (creditos - (cred_op_tim - 24));;
485       }
486       else
487       {
488         ctotales = creditos;
489       }
490
491       finales[0] = ctotales;
492       finales[1] = cred_op_ti;
493       finales[2] = cred_op_tim;
494
495       return finales;
496     }
497 }
```

C.1.11. Búsqueda de Información de UEA

```
1  package com.example.logica;
2
3  import java.util.ArrayList;
4
5  public class Busqueda_binaria {
6
7    public Busqueda_binaria()
8    {
9
10   }
11
12   public int b_uea(int uk, ArrayList pln) {
13     // TODO Auto-generated method stub
14
15     int j = 0;
16     int s = pln.size()-1;
17     int mitad = 0, cred = 0;
18     int up;
19
20     while(j <= s)
21     {
22       mitad = (j + s) / 2;
23       up = Integer.parseInt(((ArrayList<String>) pln.get(mitad)).get(0));
24
25       if(uk == up)
```

```
26        {
27          cred = Integer.parseInt(((ArrayList<String>) pln.get(mitad)).get(2));
28          return cred;
29        }
30        else if(uk < up)
31        {
32          s = mitad - 1;
33        }
34        else
35        {
36          j = mitad+1;
37        }
38      }
39
40      return 0;
41    }
42
43    public int[] cred_optativas(int ckardex, ArrayList pln) {
44      // TODO Auto-generated method stub
45
46      int inf = 0, sup = pln.size() - 1;
47      int medio = 0, uea_plan = 0, creditos_uea = 0;
48      int creditos [] = new int[2];
49      String optativa, tronco;
50
51      creditos[0] = 0;
52      creditos[1] = 0;
53
54      while(inf <= sup)
55      {
56        medio = (inf + sup) / 2;
57        uea_plan = Integer.parseInt(((ArrayList<String>) pln.get(medio)).get(0));
58
59        if(ckardex == uea_plan)
60        {
61          optativa = ((ArrayList<String>) pln.get(medio)).get(4);
62          tronco = ((ArrayList<String>) pln.get(medio)).get(3);
63
64          if(optativa.equals("OPT"))
65          {
66            if(tronco.equals("TI"))
67            {
68              creditos_uea = Integer.parseInt(((ArrayList<String>) pln.get(medio)).get(2));
69              creditos[0] = 1;
70              creditos[1] = creditos_uea;
71            }
72
73            if(tronco.equals("TIM"))
74            {
75              creditos_uea = Integer.parseInt(((ArrayList<String>) pln.get(medio)).get(2));
76              creditos[0] = 2;
77              creditos[1] = creditos_uea;
78            }
79
80          }
81
82          return creditos;
83        }
84        else if(ckardex < uea_plan)
85        {
86          sup = medio - 1;
87        }
88        else
89        {
90          inf = medio + 1;
91        }
92      }
93
```

```
94      return creditos;
95    }
96
97    public int cred_opintegracion(int caprobada, ArrayList mplan) {
98      // TODO Auto-generated method stub
99
100
101     int inf = 0, sup = mplan.size() - 1;
102     int medio = 0, uea_plan = 0, creditos_uea = 0;
103     String tronco, optativa;
104
105     while(inf <= sup)
106     {
107       medio = (inf + sup) / 2;
108       uea_plan = Integer.parseInt(((ArrayList<String>) mplan.get(medio)).get(0));
109
110       if(caprobada == uea_plan)
111       {
112         optativa = ((ArrayList<String>) mplan.get(medio)).get(4);
113         tronco = ((ArrayList<String>) mplan.get(medio)).get(3);
114         if(optativa.equals("OPT") && tronco.equals("TI"))
115         {
116           creditos_uea = Integer.parseInt(((ArrayList<String>) mplan.get(medio)).get(2));
117           return creditos_uea;
118         }
119         else
120         {
121           return 0;
122         }
123       }
124       else if(caprobada < uea_plan)
125       {
126         sup = medio - 1;
127       }
128       else
129       {
130         inf = medio + 1;
131       }
132     }
133
134     return 0;
135   }
136
137   public String[] uea_recomendada(int obligatoria, ArrayList plan) {
138     // TODO Auto-generated method stub
139
140     String datos_uea[] = new String[2];
141     int j = 0;
142     int s = plan.size()-1;
143     int mitad = 0;
144     int up;
145
146     while(j <= s)
147     {
148         mitad = (j + s) / 2;
149         up = Integer.parseInt(((ArrayList<String>) plan.get(mitad)).get(0));
150
151         if(obligatoria == up)
152         {
153           datos_uea[0] = ((ArrayList<String>) plan.get(mitad)).get(1);
154           datos_uea[1] = ((ArrayList<String>) plan.get(mitad)).get(2);
155           return datos_uea;
156         }
157         else if(obligatoria < up)
158         {
159           s = mitad - 1;
160         }
161         else
```

```
162         {
163           j = mitad+1;
164         }
165
166     }
167
168       return datos_uea;
169   }
170 }
```

C.1.12. Eliminar UEA del plan

```
1  package com.example.logica;
2
3  import java.util.ArrayList;
4
5  public class Recorta_plan {
6
7    public Recorta_plan()
8    {
9
10   }
11
12   public void elimina_ueas(ArrayList plan1, ArrayList uea_habiles) {
13     // TODO Auto-generated method stub
14
15     Recorta_plan rec = new Recorta_plan();
16     int p = 0;
17     int cp;
18     String clave;
19     int bandera;
20     while(p < plan1.size())
21     {
22       cp = Integer.parseInt(((ArrayList<String>) plan1.get(p)).get(0));
23       bandera = rec.elimina(cp, uea_habiles);
24        if(bandera == 0)
25        {
26          plan1.remove(plan1.get(p));
27        }
28        else
29        {
30          p++;
31        }
32     }
33
34   }
35
36   public int elimina(int cp, ArrayList uea_habiles) {
37     // TODO Auto-generated method stub
38     int  inf = 0, sup = uea_habiles.size() - 1;
39     int medio = 0, uea_habil;
40
41     while(inf <= sup)
42     {
43       medio = (inf + sup )/2;
44       uea_habil = Integer.parseInt((String) uea_habiles.get(medio));
45
46       if(cp == uea_habil)
47       {
48         return 1;
49       }
50       else if(cp < uea_habil)
51       {
52         sup = medio - 1;
53       }
```

```
54        else
55        {
56          inf = medio + 1;
57        }
58      }
59
60      return 0;
61    }
62  }
```

C.1.13. Determinación de las UEA hábiles

```
1  package com.example.logica;
2
3  import java.util.ArrayList;
4  import java.util.Iterator;
5  import java.util.StringTokenizer;
6
7  public class Uea_habilitadas {
8
9    void Uea_habilitadas()
10   {
11
12   }
13
14   public ArrayList<String> habilitaciones(ArrayList plan1, ArrayList claveskardex,
15       int creditos, ArrayList autorizaciones) {
16     // TODO Auto-generated method stub
17
18     //REMOVEMOS LAS UEAS DEL PLAN QUE YA FUERON APROBADAS
19
20     int p = 0;
21     int cp;
22     String clave;
23     int bandera;
24
25     while(p < plan1.size())
26     {
27       cp = Integer.parseInt(((ArrayList<String>) plan1.get(p)).get(0));
28       bandera = elimina_uea(cp, claveskardex);
29        if(bandera == 1)
30        {
31          plan1.remove(plan1.get(p));
32        }
33        else
34        {
35          p++;
36        }
37     }
38
39
40     //DETERMINAMOS LAS UEAS HABILITADAS
41
42     int contador = 0;
43     String seriacion = "", requisito,posible="";
44     int tmax = 0,claveaut;
45     ArrayList posibles = new ArrayList();
46     boolean cumple = false;
47
48     while(contador < plan1.size())
49     {
50       seriacion = ((ArrayList<String>) plan1.get(contador)).get(7);
51       StringTokenizer tokens = new StringTokenizer(seriacion,",");
52       claveaut = Integer.parseInt(((ArrayList<String>) plan1.get(contador)).get(0));
53
```

```
54        while(tokens.hasMoreTokens() && tmax < 4)
55        {
56          requisito = tokens.nextToken();
57          cumple = puede_cursarla(requisito,claveskardex,creditos,posibles,autorizaciones,
                claveaut);
58
59          if(cumple == false)
60          {
61            tmax = 4;
62          }
63          else
64          {
65            tmax++;
66          }
67
68        }
69
70        if(cumple == true)
71        {
72          tmax = 0;
73          posibles.add(((ArrayList<String>) plan1.get(contador)).get(0));
74        }
75        else
76        {
77          tmax = 0;
78        }
79        contador++;
80      }
81
82      return posibles;
83    }
84
85    private int elimina_uea(int clave_plan, ArrayList claveskardex) {
86      // TODO Auto-generated method stub
87
88      int  inf = 0, sup = claveskardex.size() - 1;
89      int medio = 0, uea_kardex;
90
91      while(inf <= sup)
92      {
93        medio = (inf + sup )/2;
94        uea_kardex = Integer.parseInt((String) claveskardex.get(medio));
95
96        if(clave_plan == uea_kardex)
97        {
98          return 1;
99        }
100       else if(clave_plan < uea_kardex)
101       {
102         sup = medio - 1;
103       }
104       else
105       {
106         inf = medio + 1;
107       }
108     }
109
110     return 0;
111   }
112
113   private boolean puede_cursarla(String requisito, ArrayList claveskardex,
114       int creditos, ArrayList posibles, ArrayList autorizaciones,
115       int claveaut) {
116     // TODO Auto-generated method stub
117
118
119     boolean cumple = false;
120     int clave = 0, cred_nec=0;
```

```
121
122     if(requisito.contains(" CREDITOS"))
123     {
124       requisito = requisito.replace(" CREDITOS", "");
125
126       cred_nec = Integer.parseInt(requisito);
127
128       if(creditos >= cred_nec)
129       {
130         cumple = true;
131       }
132       else
133       {
134         cumple = false;
135       }
136
137     }
138     else if(requisito.startsWith("C"))
139     {
140       requisito = requisito.replace("C","");
141       clave = Integer.parseInt(requisito);
142       cumple = es_posible2(clave,claveskardex);
143       if(cumple == false)
144       {
145         cumple = es_posible2(clave,posibles);
146       }
147
148     }
149     else if(requisito.equals("AUTORIZACION"))
150     {
151       cumple = verifica_auto(claveaut,autorizaciones);
152     }
153     else if(requisito.equals("0"))
154     {
155       cumple = true;
156     }
157     else
158     {
159       clave = Integer.parseInt(requisito);
160       cumple = es_posible2(clave,claveskardex);
161     }
162
163     return cumple;
164   }
165
166   private boolean verifica_auto(int claveaut, ArrayList autorizaciones) {
167     // TODO Auto-generated method stub
168     boolean cumple = false;
169     if(autorizaciones.size() == 1)
170     {
171       cumple = false;
172     }
173     else
174     {
175       cumple = es_posible2(claveaut,autorizaciones);
176     }
177
178     return cumple;
179   }
180
181   private boolean es_posible2(int clave, ArrayList claveskardex) {
182     // TODO Auto-generated method stub
183     int inf = 0, sup = claveskardex.size() - 1;
184     int medio = 0, uea_kardex;
185
186
187     while(inf <= sup)
188     {
```

```
189         medio = (inf + sup) / 2;
190         uea_kardex = Integer.parseInt((String) claveskardex.get(medio));
191 
192         if(clave == uea_kardex)
193         {
194           return true;
195         }
196         else if(clave < uea_kardex)
197         {
198           sup = medio - 1;
199         }
200         else
201         {
202           inf = medio + 1;
203         }
204 
205       }
206 
207       return false;
208     }
209 }
```

C.1.14. Determinación de las UEA reprobadas dos o más veces en evaluación global

```
1  package com.example.ueareprobadas;
2  
3  import java.util.ArrayList;
4  import java.util.Iterator;
5  
6  import com.example.logica.Recorta_plan;
7  
8  
9  public class Uea_reprob {
10 
11   void Uea_reprob()
12   {
13 
14   }
15 
16   public void elimina_reprobadas(ArrayList uea_reprobadas,
17       ArrayList claveskardex, ArrayList plan1, ArrayList kardex2) {
18     // TODO Auto-generated method stub
19 
20     int i = 0, elimina, clave_rep=0;
21     int j = 0;
22     int clave_plan = 0,teval, flag;
23     Recorta_plan r = new Recorta_plan();
24     Uea_reprob ua = new Uea_reprob();
25 
26     while(i < uea_reprobadas.size())
27     {
28       clave_rep = Integer.parseInt((String) uea_reprobadas.get(i));
29       elimina = r.elimina(clave_rep, claveskardex);
30 
31       if(elimina == 1)
32       {
33         uea_reprobadas.remove(i);
34       }
35       else
36       {
37         i++;
38       }
39     }
40 
```

```
41      //ALMACENAMOS UN SOLO REGISTRO DE LAS ANTERIORES UEAS ES POSIBLE QUE ESTEN REPEDTIDAS
42
43      for(i = 0; i < uea_reprobadas.size(); i++)
44      {
45        for(j = i+1; j <uea_reprobadas.size(); j++)
46        {
47          if(uea_reprobadas.get(i).equals(uea_reprobadas.get(j)))
48          {
49            uea_reprobadas.remove(j);
50          }
51        }
52      }
53
54
55      //CONSTRUIMOS UNA ESTRUCTURA QUE CONTENDRA: CLAVE_UEA, OP. EVAl. GLOBAL
56
57      int reprobadas[][] = new int[uea_reprobadas.size()][2];
58      int m = 0;
59
60      while(m < uea_reprobadas.size())
61      {
62        reprobadas[m][0] = Integer.parseInt((String) uea_reprobadas.get(m));
63        m++;
64      }
65
66      //DETERMINAMOS LAS OPORTUNIDADES EN EV. GLOBAL DE CAD UEA DE LA ESTRUCTURA ANTERIOR
67
68      int t = 0, bandera;
69
70      while(t < reprobadas.length)
71      {
72        clave_rep = reprobadas[t][0];
73        bandera = ua.busca_evaluacion(clave_rep,kardex2);
74        reprobadas[t][1] = bandera;
75        t++;
76      }
77
78
79      //FINALMENTE ELIMINAMOS LAS UEA QUE SE HAYAN REPROBADO DOS VECES EN EV. GLOBAL DEL PLAN
80
81      while(i < plan1.size())
82      {
83        clave_plan = Integer.parseInt(((ArrayList<String>) plan1.get(i)).get(0));
84        teval = Integer.parseInt(((ArrayList<String>) plan1.get(i)).get(5));
85
86        flag = ua.reprobada(clave_plan, reprobadas);
87
88        if(flag == 1 && teval == 2)
89        {
90          plan1.remove(plan1.get(i));
91        }
92        else
93        {
94          i++;
95        }
96      }
97    }
98
99    public int reprobada(int clave_plan, int[][] reprobadas) {
100     // TODO Auto-generated method stub
101
102     int  inf = 0, sup = reprobadas.length - 1, oport;
103     int medio = 0, uea_rep;
104
105
106     while(inf <= sup)
107     {
108       medio = (inf + sup )/2;
```

```
109         uea_rep = reprobadas[medio][0];
110
111         if(clave_plan == uea_rep)
112         {
113           oport = reprobadas[medio][1];
114           if(oport >= 2)
115           {
116             return 1;
117           }
118           else
119           {
120             return 0;
121           }
122
123         }
124         else if(clave_plan < uea_rep)
125         {
126           sup = medio - 1;
127         }
128         else
129         {
130           inf = medio + 1;
131         }
132       }
133
134       return 0;
135     }
136
137     public int busca_evaluacion(int clave_rep, ArrayList kardex2) {
138       // TODO Auto-generated method stub
139       int j = 0, global = 0, ckar;
140       String eval;
141
142       while(j < kardex2.size())
143       {
144         ckar = Integer.parseInt(((ArrayList<String>) kardex2.get(j)).get(0));
145
146         if(ckar == clave_rep)
147         {
148           eval = ((ArrayList<String>) kardex2.get(j)).get(3);
149           eval = eval.replace(".", "");
150           if(eval.equals("GLO"))
151           {
152             global++;
153             j++;
154           }
155           else
156           {
157             j++;
158           }
159         }
160         else
161         {
162           j++;
163         }
164       }
165
166       return global;
167     }
168
169 }
```

C.1.15. Construcción de recomendaciones fictícias de optativas de Integración

```
1 package com.example.recomendacion;
```

```
2
3  import java.util.ArrayList;
4  import java.util.Iterator;
5
6  public class Ficticias {
7
8    void Ficticias()
9    {
10
11   }
12
13   public int[] aramado_ficticio(ArrayList intaprob) {
14     // TODO Auto-generated method stub
15     int hechizas[] = new int[4];
16     int nuea = 0,tres= 0, seis = 0, ocho = 0, nueve = 0;
17
18
19     for(int i = 0; i < intaprob.size(); i++)
20     {
21       if(intaprob.get(i).equals("8"))
22       {
23         nuea++;
24       }
25     }
26
27
28     if(nuea == 0)
29     {
30       tres = 0;
31       seis = 1;
32       ocho = nuea;
33       nueve = 7;
34     }
35
36     if(nuea == 1)
37     {
38       tres = 1;
39       seis = 1;
40       ocho = nuea;
41       nueve = 6;
42     }
43
44     if(nuea == 2)
45     {
46       tres = 1;
47       seis = 1;
48       ocho = nuea;
49       nueve = 5;
50     }
51
52     if(nuea == 3)
53     {
54       tres = 1;
55       seis = 1;
56       ocho = nuea;
57       nueve = 4;
58     }
59
60     if(nuea == 4)
61     {
62       tres = 0;
63       seis = 1;
64       ocho = nuea;
65       nueve = 7;
66     }
67
68     if(nuea == 5)
69     {
```

```
70        tres = 0;
71        seis = 2;
72        ocho = nuea;
73        nueve = 2;
74      }
75
76      hechizas[0] = tres;
77      hechizas[1] = seis;
78      hechizas[2] = ocho;
79      hechizas[3] = nueve;
80
81      return hechizas;
82    }
83
84    public int[] disponibilidad(int[] hechiza, ArrayList intaprob) {
85      // TODO Auto-generated method stub
86      int disp[] = new int[5];
87      int sobrantes = 0, valor = 0, modulo = 0;
88      int tres = 0, seis = 0, ocho = 0, nueve = 0;
89
90      tres = hechiza[0];
91      seis = hechiza[1];
92      ocho = hechiza[2];
93      nueve = hechiza[3];
94
95      //DECREMENTAMOS LAS UEA DISPONIBLES SEGUN LO QUE EL ALUMNO HA APROBADO
96      for(int i = 0; i < intaprob.size(); i++)
97      {
98        if(intaprob.get(i).equals("3"))
99        {
100         if(tres > 0)
101         {
102           tres--;
103         }
104         else
105         {
106           sobrantes += Integer.parseInt((String) intaprob.get(i));
107         }
108       }
109       else if(intaprob.get(i).equals("6"))
110       {
111         if(seis > 0)
112         {
113           seis--;
114         }
115         else
116         {
117           sobrantes += Integer.parseInt((String) intaprob.get(i));
118         }
119       }
120       else if(intaprob.get(i).equals("8"))
121       {
122         if(ocho > 0)
123         {
124           ocho--;
125         }
126         else
127         {
128           sobrantes += Integer.parseInt((String) intaprob.get(i));
129         }
130       }
131       else if(intaprob.get(i).equals("9"))
132       {
133         if(nueve > 0)
134         {
135           nueve--;
136         }
137         else
```

```
138         {
139           sobrantes += Integer.parseInt((String) intaprob.get(i));
140         }
141       }
142       else
143       {
144         sobrantes += Integer.parseInt((String) intaprob.get(i));
145       }
146     }
147
148
149     //COMPROBAMOS SI HAY CREDITOS SOBRANTES Y SE PUEDEN OTORGAR MAS UEA COMO APROBADAS
150
151     if(sobrantes > 3)
152     {
153       while( sobrantes > 3)
154       {
155         if(sobrantes >= 9)
156         {
157           if(nueve > 0)
158           {
159             valor = sobrantes / 9;
160             sobrantes = sobrantes %9;
161
162             if(valor > nueve)
163             {
164               sobrantes = (valor - nueve);
165               nueve = ((-sobrantes)+(valor));
166             }
167             else
168             {
169
170               nueve = nueve - valor;
171             }
172           }
173           else if(seis > 0)
174           {
175             valor = sobrantes / 6;
176             sobrantes = sobrantes %6;
177
178             if(valor > seis)
179             {
180               sobrantes = (valor - seis);
181               seis = ((-sobrantes)+(valor));
182             }
183             else
184             {
185               seis = seis - valor;
186             }
187           }
188           else
189           {
190             if(tres > 0)
191             {
192               valor = sobrantes / 3;
193               sobrantes = sobrantes %3;
194
195               if(valor > tres)
196               {
197                 sobrantes = (valor - tres);
198                 tres = ((-sobrantes)+(valor));
199               }
200               else
201               {
202
203                 tres = tres - valor;
204               }
205             }
```

```
206           }
207         }
208         else
209         {
210           if(seis > 0)
211           {
212             seis--;
213             sobrantes = sobrantes - 6;
214           }
215           else
216           {
217             if(tres > 0)
218             {
219               valor = sobrantes / 3;
220               sobrantes = sobrantes %3;
221
222               if(valor > tres)
223               {
224                 tres = ((-tres)+(valor));
225                 sobrantes = valor - tres;
226               }
227               else
228               {
229                 tres = tres - valor;
230               }
231             }
232           }
233         }
234       }
235     }
236
237
238     if(sobrantes == 3)
239     {
240
241       if(tres > 0)
242       {
243         valor = sobrantes / 3;
244         sobrantes = sobrantes %3;
245
246         if(valor > tres)
247         {
248           tres = ((-tres)+(valor));
249           sobrantes = valor - tres;
250         }
251         else
252         {
253           tres = tres - valor;
254           sobrantes = 0;
255         }
256       }
257     }
258
259     disp[0] = tres;
260     disp[1] = seis;
261     disp[2] = ocho;
262     disp[3] = nueve;
263     disp[4] = sobrantes;
264
265     return disp;
266   }
267 }
```

C.1.16. Ordenar UEA hábiles por prioridad

```
1  package com.example.recomendacion;
2
3  import java.util.ArrayList;
4  import java.util.Collections;
5  import java.util.Iterator;
6
7  import com.example.logica.Busqueda_binaria;
8
9  public class Prioridad {
10
11   void Prioridad()
12   {
13
14   }
15
16   public ArrayList ordena(ArrayList plan1) {
17     // TODO Auto-generated method stub
18     ArrayList prioridades = new ArrayList();
19     int i = 0;
20     String priori;
21
22     while(i < plan1.size())
23     {
24       priori = ((ArrayList<String>) plan1.get(i)).get(8);
25       prioridades.add(Integer.parseInt(priori));
26       i++;
27     }
28
29     Collections.sort(prioridades);
30     return prioridades;
31   }
32
33   public ArrayList integracion(ArrayList aprobadas, ArrayList mplan) {
34     // TODO Auto-generated method stub
35     ArrayList credintegra = new ArrayList();
36     Busqueda_binaria busqueda = new Busqueda_binaria();
37     int caprobada, credinte;
38
39     for(int i = 0; i < aprobadas.size(); i++)
40     {
41       caprobada = Integer.parseInt((String) aprobadas.get(i));
42       credinte = busqueda.cred_opintegracion(caprobada, mplan);
43
44       if(credinte != 0)
45       {
46         credintegra.add(String.valueOf(credinte));
47       }
48     }
49
50     return credintegra;
51   }
52
53 }
```

C.1.17. Construcción de la recomendación

```
1  package com.example.recomendacion;
2
3  import java.util.ArrayList;
4  import java.util.Iterator;
5
6
7  public class Recomienda {
8    static int disponible[] = new int[5];
9    void Recomienda()
10   {
11
12   }
13
14   public String[] busca_prioridad(int prioridad, ArrayList plan) {
15     // TODO Auto-generated method stub
16
17     String datos_uea[] = new String[6];
18     int indice = 0;
19     int importancia;
20
21     while(indice < plan.size())
22     {
23       importancia = Integer.parseInt(((ArrayList<String>) plan.get(indice)).get(8));
24
25       if(prioridad == importancia)
26       {
27         datos_uea[0] = ((ArrayList<String>) plan.get(indice)).get(0);
28         datos_uea[1] = ((ArrayList<String>) plan.get(indice)).get(1);
29         datos_uea[2] = ((ArrayList<String>) plan.get(indice)).get(2);
30         datos_uea[3] = ((ArrayList<String>) plan.get(indice)).get(3);
31         datos_uea[4] = ((ArrayList<String>) plan.get(indice)).get(4);
32         datos_uea[5] = ((ArrayList<String>) plan.get(indice)).get(7);
33         return datos_uea;
34       }
35       indice++;
36     }
37
38     return datos_uea;
39   }
40
41   public boolean verifica_corregistro(String corregistro) {
42     // TODO Auto-generated method stub
43
44     if(corregistro.indexOf("C") != -1)
45     {
46       return true;
47     }
48
49     return false;
50   }
51
52   public boolean corregistro_posible(String clave_corregistro,
53       ArrayList aprobadas) {
54     // TODO Auto-generated method stub
55
56     int clave_corr, claveaprob;
57     int inf = 0, sup = aprobadas.size() - 1;
58     int medio = 0;
59
60     clave_corr = Integer.parseInt(clave_corregistro);
61
62     while(inf <= sup)
63     {
64       medio = (inf + sup) / 2;
65       claveaprob = Integer.parseInt((String) aprobadas.get(medio));
66
```

```
67        if(clave_corr == claveaprob)
68        {
69          return true;
70        }
71        else if(clave_corr < claveaprob)
72        {
73          sup = medio - 1;
74        }
75        else
76        {
77          inf = medio + 1;
78        }
79
80      }
81      return false;
82    }
83
84    public boolean es_recomendada(String clave_corregistro,
85        ArrayList uea_recomendada) {
86      // TODO Auto-generated method stub
87
88      int indice = 0;
89      String clavereco;
90
91      while(indice < uea_recomendada.size())
92      {
93        clavereco = (String) uea_recomendada.get(indice);
94        if(clave_corregistro.equals(clavereco))
95        {
96          return true;
97        }
98        indice++;
99      }
100
101     return false;
102   }
103
104   public ArrayList arma_recomendacion(int recomendados, ArrayList prioridad2,
105       ArrayList plan, int tope, ArrayList aprobadas, int opti, int optim, ArrayList intaprob
              ) {
106     // TODO Auto-generated method stub
107     System.out.println(opti+" "+optim);
108     Recomienda rec = new Recomienda();
109     Ficticias fic = new Ficticias();
110     int rec_acumulados = 0, i = 0;
111     int creditos_uea = 0, crtemporal = 0, importancia;
112     String uea_rec[] = new String[6];
113     String corregistro, clave_corregistro="";
114     boolean escorregistro, esposible;
115     ArrayList uea_recomendada = new ArrayList();
116     int hechiza [] = new int[4];
117
118
119     //COMPROBAMOS LOS CREDITOS DE LAS UEAS DISPONIBLES A RECOMENDAR, METODOS QUE SE
            ENCUENTRAN EN LA CLASE Ficticias.java
120
121     hechiza = fic.aramado_ficticio(intaprob);
122     disponible = fic.disponibilidad(hechiza,intaprob);
123
124     while((rec_acumulados < recomendados) && (i < prioridad2.size()))
125     {
126
127       importancia = (Integer) prioridad2.get(i);
128
129       //BUSCAMOS LA UEA CON ESA PRIORIDAD Y REGRESAMOS UN VECTOR CON 5 DATOS
130       //CLAVE, UEA, CREDITOS, TRONCO, OBL/OPT
131
132       uea_rec = rec.busca_prioridad(importancia,plan);
```

```
133         corregistro = uea_rec[5];
134         //SI LA UEA CON PRIORIDAD BUSCADA ES OPTATIVA SE HACE UNA VERIFICACION
135         if(uea_rec[4].equals("OPT"))
136         {
137           if(uea_rec[3].equals("TI"))
138           {
139             if(opti < 69)
140             {
141               int crint = 0;
142               //UNA FUNCION QUE COMPUREBE LOS CREDItOS DE LA UEA A RECOMENDAR
143               crint = entra(disponible, rec_acumulados, recomendados, tope);
144               if(crint != 0)
145               {
146                 rec_acumulados += crint;
147                 opti += crint;
148                 uea_recomendada.add("TI:"+crint);
149               }
150             }
151           }
152           else
153           {
154             if(optim < 24)
155             {
156               if(((6 + rec_acumulados) <= recomendados) && (6 + rec_acumulados) <= tope)
157               {
158                 optim += 6;
159                 rec_acumulados += 6;
160                 uea_recomendada.add("TIM");
161               }
162             }
163
164           }
165         }
166         else
167         {
168           creditos_uea = Integer.parseInt(uea_rec[2]);
169           crtemporal = creditos_uea;
170
171           if((crtemporal + rec_acumulados) <= recomendados && (crtemporal + rec_acumulados) <=
                  tope)
172           {
173             //SE DEBE COMPROBAR EL CORREGISTRO EN CASO DE HABER
174             escorregistro = rec.verifica_corregistro(corregistro);
175
176             if(escorregistro)
177             {
178               //OBTENEMOS LA CLAVE DEL CORREGISTRO
179               clave_corregistro = rec.regresa_clave_corregistro(corregistro);
180
181               //LLAMAMOS A LA FUNCION QUE NOS DIRA SI EL CORREGISTRO ES POSIBLE
182               esposible = rec.corregistro_posible(clave_corregistro, aprobadas); //UEAS
                      APROBADAS
183               if(esposible) // SI SE ENCUENTRA APROBADO EL CORREGISTRO
184               {
185                 //SE AGREGA A LA LISTA A DESPLEGAR
186                 uea_recomendada.add(uea_rec[0]);
187                 rec_acumulados += creditos_uea;
188               }
189               else
190               {
191                 esposible = rec.es_recomendada(clave_corregistro, uea_recomendada); //UEA
                        RECOMENDADAS
192                 if(esposible)
193                 {
194                   //SE AGREGA A LA LISTA A DESPLEGAR
195                   uea_recomendada.add(uea_rec[0]);
196                   rec_acumulados += creditos_uea;
197                 }
```

```
198             }
199           }
200           else
201           {
202             //SE AGREGA A LA LISTA A DESPLEGAR
203             uea_recomendada.add(uea_rec[0]);
204             rec_acumulados += creditos_uea;
205           }
206         }
207       }
208       i++;
209
210     }
211
212     return uea_recomendada;
213   }
214
215   private String regresa_clave_corregistro(String corregistro) {
216     // TODO Auto-generated method stub
217     String clavec="";
218
219     for(int j = 0; j < corregistro.length(); j++)
220     {
221       if(corregistro.charAt(j) == 'C')
222       {
223         clavec = corregistro.substring(j+1, j+8);
224       }
225     }
226     return clavec;
227   }
228
229   private int entra(int[] disponible, int rec_acumulados, int recomendados, int tope) {
230     // TODO Auto-generated method stub
231     int cti = 0;
232
233     //disponible[4] = creditos sobrantes
234     //disponible[0],disponible[1],disponible[2],disponible[3] = UEA de 3, 6, 8 y 9 creditos
          respectivamente
235
236     if(disponible[4] == 0) //SI SOBRAN 0 CREDITOS PODEMOS COMPLETAR UNA UEA DE 3,6 o 9
          CREDITOS
237     {
238       if(disponible[0] > 0)
239       {
240         if((rec_acumulados + 3 <= recomendados) && (rec_acumulados + 3 <= tope))
241         {
242           disponible[0]--;
243           return 3;
244         }
245         else
246         {
247           return 0;
248         }
249       }
250
251       if(disponible[1] > 0)
252       {
253         if((rec_acumulados + 6 <= recomendados) && (rec_acumulados + 6 <= tope))
254         {
255           disponible[1]--;
256           return 6;
257         }
258         else
259         {
260           return 0;
261         }
262       }
263
```

```
264         if(disponible[3] > 0)
265         {
266           if((rec_acumulados + 9 <= recomendados) && (rec_acumulados + 9 <= tope))
267           {
268             disponible[3]--;
269             return 9;
270           }
271           else
272           {
273             return 0;
274           }
275         }
276       }
277       else if(disponible[4] == 3) //SI SOBRAN 3 CREDITOS PODEMOS COMPLETAR UNA UEA DE 6 o 9
              CREDITOS
278       {
279         if(disponible[1] > 0)
280         {
281           if((rec_acumulados + 3 <= recomendados) && (rec_acumulados + 3 <= tope))
282           {
283             disponible[1]--;
284             disponible[4] = 0;
285             return 3;
286           }
287           else
288           {
289             return 0;
290           }
291         }
292
293         if(disponible[3] > 0)
294         {
295           if((rec_acumulados + 6 <= recomendados) && (rec_acumulados + 6 <= tope))
296           {
297             disponible[3]--;
298             disponible[4] = 0;
299             return 6;
300           }
301           else
302           {
303             return 0;
304           }
305         }
306       }
307       else //SI SOBRAN 6 CREDITOS PODEMOS COMPLETAR UNA UEA DE 9 CREDITOS
308       {
309         if(disponible[3] > 0)
310         {
311           if((rec_acumulados + 3 <= recomendados) && (rec_acumulados + 3 <= tope))
312           {
313             disponible[3]--;
314             disponible[4] = 0;
315             return 3;
316           }
317           else
318           {
319             return 0;
320           }
321         }
322       }
323
324       return 0;
325     }
326 }
```

C.1.18. Modelado del despliegue de la recomendación 1

```
1  package com.example.despliegues;
2
3  public class ItemTexto {
4    protected String clave;
5    protected String uea;
6    protected String creditos;
7
8    public ItemTexto()
9    {
10     this.clave = "";
11     this.uea = "";
12     this.creditos = "";
13   }
14
15   public ItemTexto(String clave, String uea, String creditos)
16   {
17     this.clave = clave;
18     this.uea = uea;
19     this.creditos = creditos;
20   }
21
22   public String getClave()
23   {
24     return clave;
25   }
26
27   public String getUea()
28   {
29     return uea;
30   }
31
32   public String getCreditos()
33   {
34     return creditos;
35   }
36
37   public String setClave()
38   {
39     return clave;
40   }
41
42   public String setUea()
43   {
44     return uea;
45   }
46
47   public String setCreditos()
48   {
49     return creditos;
50   }
51 }
```

C.1.19. Modelado del despliegue de la recomendación 1-2

```
1  package com.example.despliegues;
2
3  import java.util.ArrayList;
4
5  import com.example.napp.R;
6
7  import android.app.Activity;
8  import android.content.Context;
9  import android.graphics.Color;
10 import android.view.LayoutInflater;
11 import android.view.View;
12 import android.view.ViewGroup;
13 import android.widget.BaseAdapter;
14 import android.widget.TextView;
15
16 public class ItemTextoAdapter extends BaseAdapter{
17
18   protected Activity activity;
19   protected ArrayList<ItemTexto> items;
20
21   public ItemTextoAdapter(Activity activity, ArrayList<ItemTexto> items){
22     this.activity = activity;
23     this.items = items;
24   }
25
26   @Override
27   public int getCount() {
28     // TODO Auto-generated method stub
29     return items.size();
30   }
31
32   @Override
33   public Object getItem(int arg0) {
34     // TODO Auto-generated method stub
35     return 0;
36   }
37
38   @Override
39   public long getItemId(int arg0) {
40     // TODO Auto-generated method stub
41     return 0;
42   }
43
44   @Override
45   public View getView(int position, View contentView, ViewGroup parent) {
46     // TODO Auto-generated method stub
47     View v = contentView;
48
49     if(contentView == null)
50     {
51       LayoutInflater inflater = (LayoutInflater) activity.getSystemService(Context.
           LAYOUT_INFLATER_SERVICE);
52       v = inflater.inflate(R.layout.mi_lista, null);
53     }
54
55     ItemTexto texto = items.get(position);
56
57     TextView clave = (TextView)v.findViewById(R.id.clave);
58     clave.setText(texto.getClave());
59
60     TextView uea = (TextView)v.findViewById(R.id.uea);
61     uea.setText(texto.getUea());
62
63     TextView creditos = (TextView)v.findViewById(R.id.creditos);
64     creditos.setText(texto.getCreditos());
65
```

```
66
67      if(position %2 == 0)
68      {
69        v.setBackgroundColor(Color.parseColor("#B0C4DE"));
70
71      }
72      else
73      {
74        v.setBackgroundColor(Color.parseColor("#FFFFFF"));
75
76      }
77      return v;
78    }
79
80  }
```

C.2. Código XML

C.2.1. Archivo principal AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3
4      package="com.example.napp"
5
6      android:versionCode="1"
7
8      android:versionName="1.0" >
9
10     <uses-sdk
11
12         android:minSdkVersion="9"
13
14         android:targetSdkVersion="17" />
15
16     <application
17
18         android:allowBackup="true"
19
20         android:icon="@drawable/ic_launcher"
21
22         android:label="@string/app_name"
23
24         android:theme="@style/AppTheme" android:debuggable="true">
25
26         <activity
27
28             android:name="com.example.napp.MainActivity"
29
30             android:label="@string/app_name"
31             android:screenOrientation="portrait" >
32
33             <intent-filter>
34
35                 <action android:name="android.intent.action.MAIN" />
36
37                 <category android:name="android.intent.category.LAUNCHER" />
38
39             </intent-filter>
40
41         </activity>
42         <activity android:label="Mapp"
43             android:name=".Recomendacion"
44             android:configChanges="keyboard|keyboardHidden"
```

```
            android:screenOrientation="portrait" />
        <activity android:label="Mapp"
            android:name=".Opina"
            android:configChanges="keyboard|keyboardHidden"
            android:screenOrientation="portrait" />
        <activity android:label="Mapp"
            android:name=".Multidisciplinares"
            android:configChanges="keyboard|keyboardHidden"
            android:screenOrientation="portrait" />
        <activity android:label="Mapp"
            android:name=".Integracion"
            android:configChanges="keyboard|keyboardHidden"
            android:screenOrientation="portrait" />

    </application>

<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<supports-screens
    android:smallScreens="true"
    android:normalScreens="true"
    android:largeScreens="true"
    android:xlargeScreens="true"
    android:anyDensity="true"/>
</manifest>
```

C.2.2. Interfaces de dispositivos de tamaño pequeño

C.2.3. Interfaz de inicio

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:paddingBottom="@dimen/activity_vertical_margin"

    android:paddingLeft="@dimen/activity_horizontal_margin"

    android:paddingRight="@dimen/activity_horizontal_margin"

    android:paddingTop="@dimen/activity_vertical_margin"

    android:background="@drawable/fondo"

    tools:context=".MainActivity" >



    <EditText

        android:id="@+id/editText1"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_below="@+id/textView1"

        android:layout_centerHorizontal="true"

        android:inputType="number"
```

```
36
37          android:maxEms="15" />
38
39
40      <TextView
41
42          android:id="@+id/textView1"
43
44          android:layout_width="wrap_content"
45
46          android:layout_height="wrap_content"
47
48          android:layout_alignParentTop="true"
49
50          android:layout_centerHorizontal="true"
51
52          android:layout_marginTop="60dp"
53
54          android:text="Matricula"
55
56          android:textAppearance="?android:attr/textAppearanceMedium"
57
58          android:textSize="10dp"
59
60          tools:ignore="HardcodedText,SpUsage" />
61
62
63      <TextView
64
65          android:id="@+id/textView2"
66
67          android:layout_width="wrap_content"
68
69          android:layout_height="wrap_content"
70
71          android:layout_below="@+id/editText1"
72
73          android:layout_centerHorizontal="true"
74
75          android:layout_marginTop="20dp"
76
77          android:text="Contrasenia"
78
79          android:textAppearance="?android:attr/textAppearanceMedium"
80
81          android:textSize="10dp"
82
83          tools:ignore="HardcodedText,SpUsage" />
84
85
86      <EditText
87
88          android:id="@+id/editText2"
89
90          android:layout_width="wrap_content"
91
92          android:layout_height="wrap_content"
93
94          android:layout_below="@+id/textView2"
95
96          android:layout_centerHorizontal="true"
97
98          android:inputType="textPassword" >
99
100
101
102
103
```

```
104
105         <requestFocus />
106
107     </EditText>
108
109
110
111     <Button
112
113         android:id="@+id/button1"
114
115         style="?android:attr/buttonStyleSmall"
116
117         android:layout_width="wrap_content"
118
119         android:layout_height="wrap_content"
120
121         android:layout_below="@+id/editText2"
122
123         android:layout_centerHorizontal="true"
124
125         android:layout_marginTop="28dp"
126
127         android:text="Ingresar"
128
129         tools:ignore="HardcodedText" />
130
131
132
133 </RelativeLayout>
```

C.2.4. Interfaz de despliegue de recomendación

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5     xmlns:tools="http://schemas.android.com/tools"
6
7     android:layout_width="match_parent"
8
9     android:layout_height="match_parent"
10
11    android:background="@drawable/rec" >
12
13
14
15    <ListView
16
17        android:id="@+id/recomendacion"
18
19        android:layout_width="match_parent"
20
21        android:layout_height="wrap_content"
22
23        android:layout_alignParentTop="true"
24
25        android:layout_centerHorizontal="true"
26
27        android:layout_marginBottom="350dp"
28
29        android:layout_marginTop="30dp" >
30
31    </ListView>
32
```

```
33
34
35      <TextView
36
37          android:id="@+id/textView1"
38
39          android:layout_width="wrap_content"
40
41          android:layout_height="wrap_content"
42
43          android:layout_alignParentLeft="true"
44
45          android:layout_alignTop="@+id/listView1"
46
47          android:layout_marginLeft="10dp"
48
49          android:layout_marginTop="92dp"
50
51          android:text="Creditos recomendados"
52
53          android:textAppearance="?android:attr/textAppearanceSmall"
54
55          android:textColor="#df0101"
56
57          android:textSize="12dp"
58
59          tools:ignore="HardcodedText,UnknownIdInLayout,SpUsage" />
60
61
62
63      <TextView
64
65          android:id="@+id/textView2"
66
67          android:layout_width="wrap_content"
68
69          android:layout_height="wrap_content"
70
71          android:layout_alignLeft="@+id/textView1"
72
73          android:layout_below="@+id/textView1"
74
75          android:layout_marginTop="10dp"
76
77          android:text="Minimo:"
78
79          android:textAppearance="?android:attr/textAppearanceSmall"
80
81          android:textSize="8dp"
82
83          tools:ignore="HardcodedText,SpUsage" />
84
85
86
87      <TextView
88
89          android:id="@+id/textView3"
90
91          android:layout_width="wrap_content"
92
93          android:layout_height="wrap_content"
94
95          android:layout_alignBottom="@+id/textView2"
96
97          android:layout_marginLeft="10dp"
98
99          android:layout_toRightOf="@+id/textView2"
100
```

```
        android:text="0"

        android:textAppearance="?android:attr/textAppearanceSmall"

        android:textSize="8dp"

        tools:ignore="HardcodedText,SpUsage" />


    <TextView

        android:id="@+id/textView4"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_alignBaseline="@+id/textView3"

        android:layout_alignBottom="@+id/textView3"

        android:layout_alignRight="@+id/textView1"

        android:text="Maximo:"

        android:textAppearance="?android:attr/textAppearanceSmall"

        android:textSize="8dp"

        tools:ignore="HardcodedText,SpUsage" />


    <TextView

        android:id="@+id/textView5"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_alignBottom="@+id/textView4"

        android:layout_marginLeft="10dp"

        android:layout_toRightOf="@+id/textView4"

        android:text="0"

        android:textAppearance="?android:attr/textAppearanceSmall"

        android:textSize="8dp"

        tools:ignore="HardcodedText,SpUsage" />


    <TextView

        android:id="@+id/textView6"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_alignLeft="@+id/textView2"

```

```
169             android:layout_below="@+id/textView2"
170
171             android:layout_marginTop="15dp"
172
173             android:text="Creditos acumulados:"
174
175             android:textAppearance="?android:attr/textAppearanceSmall"
176
177             android:textSize="8dp"
178
179             tools:ignore="HardcodedText,SpUsage" />
180
181
182
183         <TextView
184
185             android:id="@+id/textView7"
186
187             android:layout_width="wrap_content"
188
189             android:layout_height="wrap_content"
190
191             android:layout_alignBaseline="@+id/textView6"
192
193             android:layout_alignBottom="@+id/textView6"
194
195             android:layout_alignLeft="@+id/textView4"
196
197             android:layout_marginLeft="5dp"
198
199             android:text="0"
200
201             android:textAppearance="?android:attr/textAppearanceSmall"
202
203             android:textSize="8dp"
204
205             tools:ignore="HardcodedText,SpUsage" />
206
207
208
209         <TextView
210
211             android:id="@+id/textView8"
212
213             android:layout_width="wrap_content"
214
215             android:layout_height="wrap_content"
216
217             android:layout_alignLeft="@+id/textView6"
218
219             android:layout_below="@+id/textView6"
220
221             android:layout_marginTop="10dp"
222
223             android:text="Ajustar creditos:"
224
225             android:textAppearance="?android:attr/textAppearanceSmall"
226
227             android:textSize="8dp"
228
229             tools:ignore="HardcodedText,SpUsage" />
230
231
232
233         <TextView
234
235             android:id="@+id/textView9"
236
```

```
237         android:layout_width="wrap_content"
238
239         android:layout_height="wrap_content"
240
241         android:layout_alignBaseline="@+id/textView8"
242
243         android:layout_alignBottom="@+id/textView8"
244
245         android:layout_toRightOf="@+id/textView6"
246
247         android:text="0"
248
249         android:textAppearance="?android:attr/textAppearanceSmall"
250
251         android:textSize="8dp"
252
253         tools:ignore="HardcodedText,SpUsage" />
254
255
256
257     <TextView
258
259         android:id="@+id/textView10"
260
261         android:layout_width="wrap_content"
262
263         android:layout_height="wrap_content"
264
265         android:layout_alignBottom="@+id/textView9"
266
267         android:layout_toRightOf="@+id/textView4"
268
269         android:text="Ajuste posible:"
270
271         android:textAppearance="?android:attr/textAppearanceSmall"
272
273         android:textColor="#df0101"
274
275         android:textSize="8dp"
276
277         tools:ignore="HardcodedText,SpUsage" />
278
279
280
281     <TextView
282
283         android:id="@+id/rajuste"
284
285         android:layout_width="wrap_content"
286
287         android:layout_height="wrap_content"
288
289         android:layout_alignBaseline="@+id/textView10"
290
291         android:layout_alignBottom="@+id/textView10"
292
293         android:layout_marginLeft="10dp"
294
295         android:layout_toRightOf="@+id/textView10"
296
297         android:text="0"
298
299         android:textAppearance="?android:attr/textAppearanceSmall"
300
301         android:textColor="#df0101"
302
303         android:textSize="8dp"
304
```

```
        tools:ignore="HardcodedText,SpUsage" />


    <SeekBar

        android:id="@+id/seekBar1"

        android:layout_width="match_parent"

        android:layout_height="wrap_content"

        android:layout_alignParentLeft="true"

        android:layout_below="@+id/textView8"

        android:layout_marginLeft="10dp"

        android:layout_marginRight="10dp"

        android:layout_marginTop="5dp" />


    <RadioGroup

        android:id="@+id/radioGroup1"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_alignLeft="@+id/seekBar1"

        android:layout_alignParentBottom="true"

        android:layout_marginBottom="10dp" >


        <RadioButton

            android:id="@+id/radio0"

            android:layout_width="wrap_content"

            android:layout_height="wrap_content"

            android:checked="true"

            android:text="Optativas Inter - Multidisciplinares"

            android:textSize="8dp"

            tools:ignore="HardcodedText,SpUsage" />


        <RadioButton

            android:id="@+id/radio1"

            android:layout_width="wrap_content"

            android:layout_height="wrap_content"

            android:text="Optativas de Integracion"
```

```
            android:textSize="8dp"

            tools:ignore="HardcodedText,SpUsage" />


        <RadioButton

            android:id="@+id/radio2"

            android:layout_width="wrap_content"

            android:layout_height="wrap_content"

            android:text="Proporciona tu opinion"

            android:textSize="8dp"

            tools:ignore="HardcodedText,SpUsage" />


    </RadioGroup>


</RelativeLayout>
```

C.2.5. Interfaz de despliegue de optativas de integración hábiles

```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/RelativeLayout1"

    android:layout_width="fill_parent"

    android:layout_height="fill_parent"

    android:background="@drawable/ti"

    android:gravity="center_horizontal"

    android:orientation="vertical"

    android:padding="5dp" >


    <ListView

        android:id="@+id/integracion"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_alignParentLeft="true"

        android:layout_alignParentTop="true"

        android:layout_marginBottom="50dp"
```

```
36
37          android:layout_marginLeft="10dp"
38
39          android:layout_marginRight="10dp"
40
41          android:layout_marginTop="50dp"
42
43          android:divider="#B0C4DE"
44
45          android:dividerHeight="1dp"
46
47          android:headerDividersEnabled="false"
48
49          android:paddingBottom="10dp"
50
51          android:paddingTop="10dp"
52
53          tools:ignore="ObsoleteLayoutParam" >
54
55      </ListView>
56
57
58
59  </RelativeLayout>
```

C.2.6. Interfaz de despliegue de optativas multidisciplinares hábiles

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5       xmlns:tools="http://schemas.android.com/tools"
6
7       android:id="@+id/RelativeLayout1"
8
9       android:layout_width="fill_parent"
10
11      android:layout_height="fill_parent"
12
13      android:background="@drawable/inter"
14
15      android:gravity="center_horizontal"
16
17      android:orientation="vertical"
18
19      android:padding="5dp" >
20
21
22
23      <ListView
24
25          android:id="@+id/integracion"
26
27          android:layout_width="wrap_content"
28
29          android:layout_height="wrap_content"
30
31          android:layout_alignParentLeft="true"
32
33          android:layout_alignParentTop="true"
34
35          android:layout_marginBottom="50dp"
36
37          android:layout_marginLeft="10dp"
38
```

```
39          android:layout_marginRight="10dp"
40
41          android:layout_marginTop="50dp"
42
43          android:divider="#B0C4DE"
44
45          android:dividerHeight="1dp"
46
47          android:headerDividersEnabled="false"
48
49          android:paddingBottom="10dp"
50
51          android:paddingTop="10dp"
52
53          tools:ignore="ObsoleteLayoutParam" >
54
55      </ListView>
56
57
58
59  </RelativeLayout>
```

C.2.7. Interfaz de envió de comentarios

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5       xmlns:tools="http://schemas.android.com/tools"
6
7       android:id="@+id/LinearLayout1"
8
9       android:layout_width="match_parent"
10
11      android:layout_height="match_parent"
12
13      android:background="@drawable/opinion"
14
15      android:gravity="center_horizontal"
16
17      android:orientation="vertical" >
18
19
20
21      <EditText
22
23          android:id="@+id/comentario"
24
25          android:layout_width="match_parent"
26
27          android:layout_height="wrap_content"
28
29          android:layout_marginLeft="30dp"
30
31          android:layout_marginRight="30dp"
32
33          android:layout_marginTop="70dp"
34
35          android:ems="10"
36
37          android:inputType="textMultiLine" >
38
39
40
41          <requestFocus />
```

```
42
43      </EditText>
44
45
46
47      <TextView
48
49          android:id="@+id/textView1"
50
51          android:layout_width="wrap_content"
52
53          android:layout_height="wrap_content"
54
55          android:layout_marginLeft="30dp"
56
57          android:layout_marginRight="20dp"
58
59          android:gravity="left"
60
61          android:maxLines="3"
62
63          android:text="*El mensaje sera enviado al Coordinador de estudios: tph@correo.azc.
                uam.mx"
64
65          android:textSize="7dp"
66
67          android:textStyle="bold|italic"
68
69          tools:ignore="HardcodedText,SpUsage" />
70
71
72
73      <Button
74
75          android:id="@+id/button1"
76
77          android:layout_width="wrap_content"
78
79          android:layout_height="wrap_content"
80
81          android:layout_marginTop="50dp"
82
83          android:text="Enviar"
84
85          android:textSize="12dp"
86
87          tools:ignore="HardcodedText,SpUsage" />
88
89
90
91  </LinearLayout>
```

C.2.8. Diseño de la lista de despliegue de recomendación

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5      xmlns:tools="http://schemas.android.com/tools"
6
7      android:id="@+id/LinearLayout1"
8
9      android:layout_width="match_parent"
10
11     android:layout_height="match_parent"
```

```
12
13      android:orientation="horizontal" >
14
15
16
17      <TextView
18
19          android:id="@+id/clave"
20
21          android:layout_width="wrap_content"
22
23          android:layout_height="wrap_content"
24
25          android:layout_gravity="center_horizontal"
26
27          android:layout_marginRight="5dp"
28
29          android:text="clave"
30
31          android:textAppearance="?android:attr/textAppearanceLarge"
32
33          android:textSize="10dp"
34
35          tools:ignore="HardcodedText,SpUsage,InefficientWeight" />
36
37
38
39      <TextView
40
41          android:id="@+id/uea"
42
43          android:layout_width="match_parent"
44
45          android:layout_height="wrap_content"
46
47          android:layout_gravity="center_horizontal"
48
49          android:layout_marginLeft="10dp"
50
51          android:layout_marginRight="15dp"
52
53          android:maxLines="5"
54
55          android:minLines="1"
56
57          android:text="uea"
58
59          android:textAppearance="?android:attr/textAppearanceLarge"
60
61          android:textSize="10dp"
62
63          tools:ignore="HardcodedText,SpUsage,InefficientWeight" />
64
65
66
67      <TextView
68
69          android:id="@+id/creditos"
70
71          android:layout_width="wrap_content"
72
73          android:layout_height="wrap_content"
74
75          android:layout_marginLeft="15dp"
76
77          android:layout_marginRight="15dp"
78
79          android:text="creditos"
```

```
80
81          android:textAppearance="?android:attr/textAppearanceLarge"
82
83          android:textSize="10dp"
84
85          tools:ignore="HardcodedText,SpUsage" />
86
87
88
89  </LinearLayout>
```

C.2.9. Interfaces de dispositivos de tamaño normal

C.2.10. Interfaz de inicio

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5       xmlns:tools="http://schemas.android.com/tools"
6
7       android:layout_width="match_parent"
8
9       android:layout_height="match_parent"
10
11      android:background="@drawable/fondo" >
12
13
14
15      <Button
16
17          android:id="@+id/button1"
18
19          style="?android:attr/buttonStyleSmall"
20
21          android:layout_width="wrap_content"
22
23          android:layout_height="wrap_content"
24
25          android:layout_below="@+id/editText2"
26
27          android:layout_centerHorizontal="true"
28
29          android:layout_marginTop="51dp"
30
31          android:text="Ingresar"
32
33          tools:ignore="HardcodedText" />
34
35
36
37      <TextView
38
39          android:id="@+id/textView1"
40
41          android:layout_width="wrap_content"
42
43          android:layout_height="wrap_content"
44
45          android:layout_alignParentTop="true"
46
47          android:layout_centerHorizontal="true"
48
49          android:layout_marginTop="86dp"
50
```

```
51          android:text="Matricula"
52
53          android:textAppearance="?android:attr/textAppearanceMedium"
54
55          android:textSize="@dimen/normal"
56
57          tools:ignore="HardcodedText" />
58
59
60
61      <EditText
62
63          android:id="@+id/editText1"
64
65          android:layout_width="wrap_content"
66
67          android:layout_height="wrap_content"
68
69          android:layout_alignLeft="@+id/editText2"
70
71          android:layout_below="@+id/textView1"
72
73          android:layout_marginTop="12dp"
74
75          android:ems="7"
76
77          android:inputType="number" />
78
79
80
81      <TextView
82
83          android:id="@+id/textView2"
84
85          android:layout_width="wrap_content"
86
87          android:layout_height="wrap_content"
88
89          android:layout_below="@+id/editText1"
90
91          android:layout_centerHorizontal="true"
92
93          android:layout_marginTop="23dp"
94
95          android:text="Contrasenia"
96
97          android:textAppearance="?android:attr/textAppearanceMedium"
98
99          android:textSize="@dimen/normal"
100
101         tools:ignore="HardcodedText" />
102
103
104
105     <EditText
106
107         android:id="@+id/editText2"
108
109         android:layout_width="wrap_content"
110
111         android:layout_height="wrap_content"
112
113         android:layout_below="@+id/textView2"
114
115         android:layout_centerHorizontal="true"
116
117         android:layout_marginTop="21dp"
118
```

```
119         android:ems="7"
120
121         android:inputType="textPassword" >
122
123
124
125         <requestFocus />
126
127     </EditText>
128
129
130
131 </RelativeLayout>
```

C.2.11. Interfaz de despliegue de recomendación

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5     xmlns:tools="http://schemas.android.com/tools"
6
7     android:layout_width="match_parent"
8
9     android:layout_height="match_parent"
10
11    android:background="@drawable/rec"
12
13    android:paddingBottom="@dimen/activity_vertical_margin"
14
15    android:paddingLeft="@dimen/activity_horizontal_margin"
16
17    android:paddingRight="@dimen/activity_horizontal_margin"
18
19    android:paddingTop="@dimen/activity_vertical_margin" >
20
21
22
23    <ListView
24
25        android:id="@+id/recomendacion"
26
27        android:layout_width="wrap_content"
28
29        android:layout_height="wrap_content"
30
31        android:layout_alignParentLeft="true"
32
33        android:layout_alignParentTop="true"
34
35        android:layout_centerHorizontal="true"
36
37        android:layout_marginBottom="350dp"
38
39        android:layout_marginLeft="5dp"
40
41        android:layout_marginRight="5dp"
42
43        android:layout_marginTop="20dp"
44
45        android:divider="#B0C4DE"
46
47        android:dividerHeight="1dp"
48
49        android:headerDividersEnabled="false"
```

```
50
51          android:paddingBottom="10dp"
52
53          android:paddingTop="10dp" >
54
55      </ListView>
56
57
58
59      <TextView
60
61          android:id="@+id/textView5"
62
63          android:layout_width="wrap_content"
64
65          android:layout_height="wrap_content"
66
67          android:layout_alignBaseline="@+id/textView4"
68
69          android:layout_alignBottom="@+id/textView4"
70
71          android:layout_marginLeft="16dp"
72
73          android:layout_toRightOf="@+id/textView4"
74
75          android:text="0"
76
77          android:textAppearance="?android:attr/textAppearanceMedium"
78
79          android:textSize="12dp"
80
81          tools:ignore="HardcodedText,SpUsage" />
82
83
84
85      <TextView
86
87          android:id="@+id/textView7"
88
89          android:layout_width="wrap_content"
90
91          android:layout_height="wrap_content"
92
93          android:layout_alignBaseline="@+id/textView6"
94
95          android:layout_alignBottom="@+id/textView6"
96
97          android:layout_marginLeft="16dp"
98
99          android:layout_toRightOf="@+id/textView6"
100
101         android:text="0"
102
103         android:textAppearance="?android:attr/textAppearanceMedium"
104
105         android:textSize="12dp"
106
107         tools:ignore="HardcodedText,SpUsage" />
108
109
110
111     <TextView
112
113         android:id="@+id/textView2"
114
115         android:layout_width="wrap_content"
116
117         android:layout_height="wrap_content"
```

```
118
119         android:layout_above="@+id/textView1"
120
121         android:layout_alignLeft="@+id/recomendacion"
122
123         android:layout_marginBottom="23dp"
124
125         android:text="Creditos recomendados"
126
127         android:textAppearance="?android:attr/textAppearanceLarge"
128
129         android:textColor="#df0101"
130
131         android:textSize="14dp"
132
133         tools:ignore="HardcodedText,SpUsage" />
134
135
136
137     <TextView
138
139         android:id="@+id/textView4"
140
141         android:layout_width="wrap_content"
142
143         android:layout_height="wrap_content"
144
145         android:layout_alignBaseline="@+id/textView1"
146
147         android:layout_alignBottom="@+id/textView1"
148
149         android:layout_alignRight="@+id/textView2"
150
151         android:text="Maximo:"
152
153         android:textAppearance="?android:attr/textAppearanceMedium"
154
155         android:textSize="12dp"
156
157         tools:ignore="HardcodedText,SpUsage" />
158
159
160
161     <TextView
162
163         android:id="@+id/textView3"
164
165         android:layout_width="wrap_content"
166
167         android:layout_height="wrap_content"
168
169         android:layout_alignBottom="@+id/textView1"
170
171         android:layout_marginLeft="12dp"
172
173         android:layout_toRightOf="@+id/textView1"
174
175         android:text="0"
176
177         android:textAppearance="?android:attr/textAppearanceMedium"
178
179         android:textSize="12dp"
180
181         tools:ignore="HardcodedText,SpUsage" />
182
183
184
185     <TextView
```

```
186
187         android:id="@+id/textView8"
188
189         android:layout_width="wrap_content"
190
191         android:layout_height="wrap_content"
192
193         android:layout_alignLeft="@+id/textView6"
194
195         android:layout_below="@+id/textView6"
196
197         android:layout_marginTop="14dp"
198
199         android:text="Ajustar creditos:"
200
201         android:textAppearance="?android:attr/textAppearanceMedium"
202
203         android:textSize="12dp"
204
205         tools:ignore="SpUsage,HardcodedText" />
206
207
208
209     <TextView
210
211         android:id="@+id/textView9"
212
213         android:layout_width="wrap_content"
214
215         android:layout_height="wrap_content"
216
217         android:layout_alignBottom="@+id/textView8"
218
219         android:layout_alignLeft="@+id/textView4"
220
221         android:text="0"
222
223         android:textAppearance="?android:attr/textAppearanceMedium"
224
225         android:textSize="12dp"
226
227         tools:ignore="HardcodedText,SpUsage" />
228
229
230
231     <TextView
232
233         android:id="@+id/textView13"
234
235         android:layout_width="wrap_content"
236
237         android:layout_height="wrap_content"
238
239         android:layout_alignBaseline="@+id/textView9"
240
241         android:layout_alignBottom="@+id/textView9"
242
243         android:layout_toRightOf="@+id/textView5"
244
245         android:text="Ajuste posible:"
246
247         android:textAppearance="?android:attr/textAppearanceMedium"
248
249         android:textColor="#df0101"
250
251         android:textSize="12dp"
252
253         tools:ignore="HardcodedText,SpUsage" />
```

```
254
255
256
257     <TextView
258
259         android:id="@+id/rajuste"
260
261         android:layout_width="wrap_content"
262
263         android:layout_height="wrap_content"
264
265         android:layout_alignBottom="@+id/textView13"
266
267         android:layout_marginLeft="14dp"
268
269         android:layout_toRightOf="@+id/textView13"
270
271         android:text="0"
272
273         android:textAppearance="?android:attr/textAppearanceMedium"
274
275         android:textColor="#df0101"
276
277         android:textSize="12dp"
278
279         tools:ignore="HardcodedText,SpUsage" />
280
281
282
283     <SeekBar
284
285         android:id="@+id/seekBar1"
286
287         android:layout_width="match_parent"
288
289         android:layout_height="wrap_content"
290
291         android:layout_alignLeft="@+id/textView8"
292
293         android:layout_below="@+id/textView8"
294
295         android:layout_marginBottom="20dp"
296
297         android:layout_marginTop="20dp" />
298
299
300
301     <TextView
302
303         android:id="@+id/textView1"
304
305         android:layout_width="wrap_content"
306
307         android:layout_height="wrap_content"
308
309         android:layout_above="@+id/radioGroup1"
310
311         android:layout_alignLeft="@+id/textView2"
312
313         android:layout_marginBottom="157dp"
314
315         android:text="Minimo:"
316
317         android:textAppearance="?android:attr/textAppearanceMedium"
318
319         android:textSize="12dp"
320
321         tools:ignore="HardcodedText,SpUsage" />
```

```
322
323
324
325     <TextView
326
327         android:id="@+id/textView6"
328
329         android:layout_width="wrap_content"
330
331         android:layout_height="wrap_content"
332
333         android:layout_alignLeft="@+id/textView1"
334
335         android:layout_centerVertical="true"
336
337         android:text="Creditos acumulados:"
338
339         android:textAppearance="?android:attr/textAppearanceMedium"
340
341         android:textSize="12dp"
342
343         tools:ignore="HardcodedText,SpUsage" />
344
345
346
347     <RadioGroup
348
349         android:id="@+id/radioGroup1"
350
351         android:layout_width="wrap_content"
352
353         android:layout_height="wrap_content"
354
355         android:layout_alignLeft="@+id/seekBar1"
356
357         android:layout_below="@+id/seekBar1" >
358
359
360
361         <RadioButton
362
363             android:id="@+id/radio0"
364
365             android:layout_width="wrap_content"
366
367             android:layout_height="wrap_content"
368
369             android:checked="false"
370
371             android:text="Optativas Inter - Multidisciplinar"
372
373             android:textSize="12dp"
374
375             tools:ignore="HardcodedText,SpUsage" />
376
377
378
379         <RadioButton
380
381             android:id="@+id/radio1"
382
383             android:layout_width="wrap_content"
384
385             android:layout_height="wrap_content"
386
387             android:text="Optativas de Integracion"
388
389             android:textSize="12dp"
```

```
390
391             tools:ignore="HardcodedText,SpUsage" />
392
393
394
395         <RadioButton
396
397             android:id="@+id/radio2"
398
399             android:layout_width="wrap_content"
400
401             android:layout_height="wrap_content"
402
403             android:text="Proporciona tu opinion"
404
405             android:textSize="12dp"
406
407             tools:ignore="HardcodedText,SpUsage" />
408
409     </RadioGroup>
410
411
412
413 </RelativeLayout>
```

C.2.12. Interfaz de despliegue de optativas de integración hábiles

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5     android:layout_width="match_parent"
6
7     android:layout_height="match_parent"
8
9     android:orientation="vertical"
10
11    android:background="@drawable/ti" >
12
13
14
15     <ListView
16
17         android:id="@+id/multidisciplinares"
18
19         android:layout_width="wrap_content"
20
21         android:layout_height="wrap_content"
22
23         android:layout_marginBottom="60dp"
24
25         android:layout_marginLeft="5dp"
26
27         android:layout_marginRight="5dp"
28
29         android:layout_marginTop="60dp"
30
31         android:divider="#B0C4DE"
32
33         android:dividerHeight="1dp"
34
35         android:headerDividersEnabled="false"
36
37         android:paddingBottom="10dp" >
38
```

```
    </ListView>


</LinearLayout>
```

C.2.13. Interfaz de despliegue de optativas multidisciplinares hábiles

```
 <?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    android:id="@+id/LinearLayout1"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:orientation="vertical"

    android:background="@drawable/inter" >


    <ListView

        android:id="@+id/multidisciplinares"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_marginBottom="60dp"

        android:layout_marginLeft="5dp"

        android:layout_marginRight="5dp"

        android:layout_marginTop="60dp"

        android:divider="#B0C4DE"

        android:dividerHeight="1dp"

        android:headerDividersEnabled="false"

        android:paddingBottom="10dp" >

    </ListView>


</LinearLayout>
```

C.2.14. Interfaz de envió de comentarios

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:background="@drawable/opinion"

    android:orientation="vertical" >


    <EditText

        android:id="@+id/comentario"

        android:layout_width="match_parent"

        android:layout_height="wrap_content"

        android:layout_marginLeft="30dp"

        android:layout_marginRight="30dp"

        android:layout_marginTop="100dp"

        android:ems="10"

        android:inputType="textMultiLine" />


    <TextView

        android:id="@+id/textView1"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_marginLeft="30dp"

        android:text="*El mensaje sera enviado al Coordinador de estudios: tph@correo.azc.
            uam.mx"

        android:textSize="8dp"

        tools:ignore="HardcodedText,SpUsage" />


    <Button

        android:id="@+id/button1"

        style="?android:attr/buttonStyleSmall"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_marginLeft="120dp"
```

```
66
67          android:layout_marginTop="30dp"
68
69          android:text="Enviar"
70
71          tools:ignore="HardcodedText" />
72
73
74
75  </LinearLayout>
```

C.2.15. Diseño de la lista de despliegue de recomendación

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5       xmlns:tools="http://schemas.android.com/tools"
6
7       android:layout_width="match_parent"
8
9       android:layout_height="match_parent"
10
11      android:orientation="horizontal" >
12
13
14
15      <TextView
16
17          android:id="@+id/clave"
18
19          android:layout_width="wrap_content"
20
21          android:layout_height="wrap_content"
22
23          android:layout_gravity="center_horizontal"
24
25          android:layout_marginRight="5dp"
26
27          android:text="clave"
28
29          android:textAppearance="?android:attr/textAppearanceLarge"
30
31          android:textSize="12dp"
32
33          tools:ignore="HardcodedText,SpUsage" />
34
35
36
37      <TextView
38
39          android:id="@+id/uea"
40
41          android:layout_width="408dp"
42
43          android:layout_height="wrap_content"
44
45          android:layout_gravity="center_horizontal"
46
47          android:layout_marginLeft="10dp"
48
49          android:layout_marginRight="15dp"
50
51          android:maxLines="5"
52
```

```
        android:minLines="1"

        android:text="uea"

        android:textAppearance="?android:attr/textAppearanceLarge"

        android:textSize="12dp"

        tools:ignore="HardcodedText,SpUsage" />


    <TextView

        android:id="@+id/creditos"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_marginLeft="15dp"

        android:layout_marginRight="15dp"

        android:text="creditos"

        android:textAppearance="?android:attr/textAppearanceLarge"

        android:textSize="12dp"

        tools:ignore="HardcodedText,SpUsage" />


</LinearLayout>
```

C.2.16. Interfaces de dispositivos de tamaño grande

C.2.17. Interfaz de inicio

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:paddingBottom="@dimen/activity_vertical_margin"

    android:paddingLeft="@dimen/activity_horizontal_margin"

    android:paddingRight="@dimen/activity_horizontal_margin"

    android:paddingTop="@dimen/activity_vertical_margin"

    android:background="@drawable/fondo"

    tools:context=".MainActivity" >


    <TextView

        android:id="@+id/textView1"
```

```
26
27          android:layout_width="wrap_content"
28
29          android:layout_height="wrap_content"
30
31          android:layout_alignParentTop="true"
32
33          android:layout_centerHorizontal="true"
34
35          android:layout_marginTop="148dp"
36
37          android:text="Matricula"
38
39          android:textAppearance="?android:attr/textAppearanceMedium"
40
41          tools:ignore="HardcodedText" />
42
43
44
45      <EditText
46
47          android:id="@+id/editText1"
48
49          android:layout_width="wrap_content"
50
51          android:layout_height="wrap_content"
52
53          android:layout_below="@+id/textView1"
54
55          android:layout_centerHorizontal="true"
56
57          android:layout_marginTop="31dp"
58
59          android:ems="10"
60
61          android:inputType="number"
62
63          android:maxLength="10"
64
65          tools:ignore="TextFields" >
66
67
68
69          <requestFocus />
70
71      </EditText>
72
73
74
75      <TextView
76
77          android:id="@+id/textView2"
78
79          android:layout_width="wrap_content"
80
81          android:layout_height="wrap_content"
82
83          android:layout_alignRight="@+id/textView1"
84
85          android:layout_below="@+id/editText1"
86
87          android:layout_marginTop="52dp"
88
89          android:text="Contrasenia"
90
91          android:textAppearance="?android:attr/textAppearanceMedium"
92
93          tools:ignore="HardcodedText" />
```

```
94
95
96
97      <EditText
98
99          android:id="@+id/editText2"
100
101         android:layout_width="wrap_content"
102
103         android:layout_height="wrap_content"
104
105         android:layout_alignLeft="@+id/editText1"
106
107         android:layout_below="@+id/textView2"
108
109         android:layout_marginTop="36dp"
110
111         android:ems="10"
112
113         android:maxLength="15"
114
115         android:inputType="textPassword" />
116
117
118
119     <Button
120
121         android:id="@+id/button1"
122
123         android:layout_width="wrap_content"
124
125         android:layout_height="wrap_content"
126
127         android:layout_below="@+id/editText2"
128
129         android:layout_centerHorizontal="true"
130
131         android:layout_marginTop="67dp"
132
133         android:text="Ingresar"
134
135         tools:ignore="HardcodedText" />
136
137
138
139 </RelativeLayout>
```

C.2.18. Interfaz de despliegue de recomendación

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3 <RelativeLayout xmlns:tools="http://schemas.android.com/tools"
4
5     xmlns:android="http://schemas.android.com/apk/res/android"
6
7     android:id="@+id/RelativeLayout1"
8
9     android:layout_width="match_parent"
10
11    android:layout_height="match_parent"
12
13    android:background="@drawable/rec"
14
15    android:paddingBottom="@dimen/activity_vertical_margin"
16
```

```
17      android:paddingTop="@dimen/activity_vertical_margin"
18
19      android:gravity="center_horizontal"
20
21      tools:context=".Recomendacion"
22
23      tools:ignore="HardcodedText" >
24
25
26
27       <ListView
28
29            android:id="@+id/recomendacion"
30
31            android:layout_width="wrap_content"
32
33            android:layout_height="wrap_content"
34
35            android:layout_alignParentLeft="true"
36
37            android:layout_alignParentTop="true"
38
39            android:layout_centerHorizontal="true"
40
41            android:layout_marginBottom="500dp"
42
43            android:layout_marginLeft="15dp"
44
45            android:layout_marginRight="15dp"
46
47            android:layout_marginTop="80dp"
48
49            android:divider="#B0C4DE"
50
51            android:dividerHeight="1dp"
52
53            android:headerDividersEnabled="false"
54
55            android:paddingBottom="10dp"
56
57            android:paddingTop="10dp"
58
59            tools:ignore="ObsoleteLayoutParam" >
60
61
62
63      </ListView>
64
65
66
67      <TextView
68
69           android:id="@+id/textView1"
70
71           android:layout_width="wrap_content"
72
73           android:layout_height="wrap_content"
74
75           android:layout_alignLeft="@+id/textView2"
76
77           android:layout_alignTop="@+id/textView2"
78
79           android:layout_marginTop="59dp"
80
81           android:text="Minimo:"
82
83           android:textAppearance="?android:attr/textAppearanceLarge"
84
```

```
85          android:textSize="@dimen/lar"
86
87          tools:ignore="HardcodedText" />
88
89
90
91      <TextView
92
93          android:id="@+id/textView3"
94
95          android:layout_width="wrap_content"
96
97          android:layout_height="wrap_content"
98
99          android:layout_alignBottom="@+id/textView1"
100
101         android:layout_marginLeft="26dp"
102
103         android:layout_toRightOf="@+id/textView1"
104
105         android:text="0"
106
107         android:textAppearance="?android:attr/textAppearanceLarge"
108
109         android:textSize="@dimen/lar"
110
111         tools:ignore="HardcodedText" />
112
113
114
115     <TextView
116
117         android:id="@+id/textView4"
118
119         android:layout_width="wrap_content"
120
121         android:layout_height="wrap_content"
122
123         android:layout_alignBaseline="@+id/textView3"
124
125         android:layout_alignBottom="@+id/textView3"
126
127         android:layout_marginLeft="101dp"
128
129         android:layout_toRightOf="@+id/textView3"
130
131         android:text="Maximo:"
132
133         android:textAppearance="?android:attr/textAppearanceLarge"
134
135         android:textSize="@dimen/lar"
136
137         tools:ignore="HardcodedText" />
138
139
140
141     <TextView
142
143         android:id="@+id/textView5"
144
145         android:layout_width="wrap_content"
146
147         android:layout_height="wrap_content"
148
149         android:layout_alignBaseline="@+id/textView4"
150
151         android:layout_alignBottom="@+id/textView4"
152
```

```
153         android:layout_marginLeft="18dp"
154
155         android:layout_toRightOf="@+id/textView4"
156
157         android:text="0"
158
159         android:textAppearance="?android:attr/textAppearanceLarge"
160
161         android:textSize="@dimen/lar" />
162
163
164
165     <TextView
166
167         android:id="@+id/textView6"
168
169         android:layout_width="wrap_content"
170
171         android:layout_height="wrap_content"
172
173         android:layout_alignLeft="@+id/textView1"
174
175         android:layout_below="@+id/textView1"
176
177         android:layout_marginTop="30dp"
178
179         android:text="Creditos acumulados:"
180
181         android:textAppearance="?android:attr/textAppearanceLarge"
182
183         android:textSize="@dimen/lar" />
184
185
186
187     <TextView
188
189         android:id="@+id/textView7"
190
191         android:layout_width="wrap_content"
192
193         android:layout_height="wrap_content"
194
195         android:layout_alignBaseline="@+id/textView6"
196
197         android:layout_alignBottom="@+id/textView6"
198
199         android:layout_toRightOf="@+id/textView2"
200
201         android:text="0"
202
203         android:textAppearance="?android:attr/textAppearanceLarge"
204
205         android:textSize="@dimen/lar" />
206
207
208
209     <TextView
210
211         android:id="@+id/textView8"
212
213         android:layout_width="wrap_content"
214
215         android:layout_height="wrap_content"
216
217         android:layout_alignLeft="@+id/textView6"
218
219         android:layout_below="@+id/textView6"
220
```

```
221         android:layout_marginTop="42dp"
222
223         android:text="Ajustar creditos:"
224
225         android:textAppearance="?android:attr/textAppearanceLarge"
226
227         android:textSize="@dimen/lar" />
228
229
230
231     <TextView
232
233         android:id="@+id/textView9"
234
235         android:layout_width="wrap_content"
236
237         android:layout_height="wrap_content"
238
239         android:layout_alignBaseline="@+id/textView8"
240
241         android:layout_alignBottom="@+id/textView8"
242
243         android:layout_alignRight="@+id/textView6"
244
245         android:layout_marginRight="17dp"
246
247         android:text="0"
248
249         android:textAppearance="?android:attr/textAppearanceLarge"
250
251         android:textSize="@dimen/lar" />
252
253
254
255     <SeekBar
256
257         android:id="@+id/seekBar1"
258
259         android:layout_width="match_parent"
260
261         android:layout_height="wrap_content"
262
263         android:layout_alignLeft="@+id/textView8"
264
265         android:layout_below="@+id/textView8"
266
267         android:layout_marginRight="40dp"
268
269         android:layout_marginTop="24dp" />
270
271
272
273     <RadioGroup
274
275         android:id="@+id/radioGroup1"
276
277         android:layout_width="match_parent"
278
279         android:layout_height="wrap_content"
280
281         android:layout_alignParentBottom="true"
282
283         android:layout_marginBottom="96dp"
284
285         android:layout_marginLeft="20dp" >
286
287
288
```

```
289         <RadioButton
290
291             android:id="@+id/radio0"
292
293             android:layout_width="wrap_content"
294
295             android:layout_height="wrap_content"
296
297             android:checked="false"
298
299             android:text="Optativas Inter - Multidisciplinares" />
300
301
302
303         <RadioButton
304
305             android:id="@+id/radio1"
306
307             android:layout_width="wrap_content"
308
309             android:layout_height="wrap_content"
310
311             android:text="Optativas de Integracion" />
312
313
314
315         <RadioButton
316
317             android:id="@+id/radio2"
318
319             android:layout_width="wrap_content"
320
321             android:layout_height="wrap_content"
322
323             android:text="Proporciona tu opinion" />
324
325     </RadioGroup>
326
327
328
329     <TextView
330
331         android:id="@+id/textView2"
332
333         android:layout_width="wrap_content"
334
335         android:layout_height="wrap_content"
336
337         android:layout_above="@+id/radioGroup1"
338
339         android:layout_alignParentLeft="true"
340
341         android:layout_marginBottom="257dp"
342
343         android:layout_marginLeft="35dp"
344
345         android:text="Creditos recomendados"
346
347         android:textAppearance="?android:attr/textAppearanceLarge"
348
349         android:textColor="#df0101"
350
351         tools:ignore="HardcodedText" />
352
353
354
355     <TextView
356
```

```
357          android:id="@+id/textView13"
358
359          android:layout_width="wrap_content"
360
361          android:layout_height="wrap_content"
362
363          android:layout_alignBottom="@+id/textView9"
364
365          android:layout_toRightOf="@+id/textView2"
366
367          android:text="Ajuste posible"
368
369          android:textAppearance="?android:attr/textAppearanceLarge"
370
371          android:textColor="#df0101"
372
373          android:textSize="@dimen/lar" />
374
375
376
377      <TextView
378
379          android:id="@+id/rajuste"
380
381          android:layout_width="wrap_content"
382
383          android:layout_height="wrap_content"
384
385          android:layout_above="@+id/seekBar1"
386
387          android:layout_marginLeft="16dp"
388
389          android:layout_toRightOf="@+id/textView13"
390
391          android:text="0"
392
393          android:textColor="#df0101"
394
395          android:textAppearance="?android:attr/textAppearanceLarge"
396
397          android:textSize="@dimen/lar" />
398
399
400
401  </RelativeLayout>
```

C.2.19. Interfaz de despliegue de optativas de integración hábiles

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5      xmlns:tools="http://schemas.android.com/tools"
6
7      android:id="@+id/RelativeLayout1"
8
9      android:layout_width="fill_parent"
10
11     android:layout_height="fill_parent"
12
13     android:background="@drawable/ti"
14
15     android:gravity="center_horizontal"
16
17     android:orientation="vertical"
```

```
18
19      android:padding="5dp" >
20
21
22
23      <ListView
24
25          android:id="@+id/integracion"
26
27          android:layout_width="wrap_content"
28
29          android:layout_height="wrap_content"
30
31          android:layout_alignParentLeft="true"
32
33          android:layout_alignParentTop="true"
34
35          android:layout_marginBottom="100dp"
36
37          android:layout_marginLeft="10dp"
38
39          android:layout_marginRight="10dp"
40
41          android:layout_marginTop="100dp"
42
43          android:divider="#B0C4DE"
44
45          android:dividerHeight="1dp"
46
47          android:headerDividersEnabled="false"
48
49          android:paddingBottom="10dp"
50
51          android:paddingTop="10dp"
52
53          tools:ignore="ObsoleteLayoutParam" >
54
55      </ListView>
56
57
58
59  </RelativeLayout>
```

C.2.20. Interfaz de despliegue de optativas multidisciplinares hábiles

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5       xmlns:tools="http://schemas.android.com/tools"
6
7       android:id="@+id/RelativeLayout1"
8
9       android:layout_width="fill_parent"
10
11      android:layout_height="fill_parent"
12
13      android:background="@drawable/inter"
14
15      android:gravity="center_horizontal"
16
17      android:orientation="vertical"
18
19      android:padding="5dp" >
20
```

```
21
22
23      <ListView
24
25          android:id="@+id/multidisciplinares"
26
27          android:layout_width="wrap_content"
28
29          android:layout_height="wrap_content"
30
31          android:layout_alignParentLeft="true"
32
33          android:layout_alignParentTop="true"
34
35          android:layout_marginBottom="100dp"
36
37          android:layout_marginLeft="10dp"
38
39          android:layout_marginRight="10dp"
40
41          android:layout_marginTop="100dp"
42
43          android:divider="#B0C4DE"
44
45          android:dividerHeight="1dp"
46
47          android:headerDividersEnabled="false"
48
49          android:paddingBottom="10dp"
50
51          android:paddingTop="10dp"
52
53          tools:ignore="ObsoleteLayoutParam" >
54
55      </ListView>
56
57
58
59  </RelativeLayout>
```

C.2.21. Interfaz de envió de comentarios

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5      xmlns:tools="http://schemas.android.com/tools"
6
7      android:id="@+id/LinearLayout1"
8
9      android:layout_width="match_parent"
10
11     android:layout_height="match_parent"
12
13     android:background="@drawable/opinion"
14
15     android:gravity="center_horizontal"
16
17     android:orientation="vertical" >
18
19
20
21     <EditText
22
23          android:id="@+id/comentario"
```

```
24
25          android:layout_width="match_parent"
26
27          android:layout_height="wrap_content"
28
29          android:layout_marginLeft="30dp"
30
31          android:layout_marginRight="30dp"
32
33          android:layout_marginTop="200dp"
34
35          android:ems="10"
36
37          android:inputType="textMultiLine" >
38
39
40
41          <requestFocus />
42
43      </EditText>
44
45
46
47      <TextView
48
49          android:id="@+id/textView1"
50
51          android:layout_width="wrap_content"
52
53          android:layout_height="wrap_content"
54
55          android:layout_marginLeft="30dp"
56
57          android:layout_marginRight="20dp"
58
59          android:gravity="left"
60
61          android:maxLines="3"
62
63          android:text="*El mensaje sera enviado al Coordinador de estudios: tph@correo.azc.
                uam.mx"
64
65          android:textSize="13dp"
66
67          android:textStyle="bold|italic"
68
69          tools:ignore="HardcodedText,SpUsage" />
70
71
72
73      <Button
74
75          android:id="@+id/button1"
76
77          android:layout_width="wrap_content"
78
79          android:layout_height="wrap_content"
80
81          android:layout_marginTop="100dp"
82
83          android:text="Enviar"
84
85          tools:ignore="HardcodedText" />
86
87
88
89  </LinearLayout>
```

C.2.22. Diseño de la lista de despliegue de recomendación

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/LinearLayout1"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:orientation="horizontal" >


    <TextView

        android:id="@+id/clave"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_gravity="center_horizontal"

        android:layout_marginRight="5dp"

        android:text="clave"

        android:textAppearance="?android:attr/textAppearanceLarge"

        android:textSize="16dp"

        tools:ignore="HardcodedText,SpUsage,InefficientWeight" />


    <TextView

        android:id="@+id/uea"

        android:layout_width="408dp"

        android:layout_height="wrap_content"

        android:layout_gravity="center_horizontal"

        android:layout_marginLeft="10dp"

        android:layout_marginRight="15dp"

        android:maxLines="5"

        android:minLines="1"

        android:text="uea"

        android:textAppearance="?android:attr/textAppearanceLarge"

        android:textSize="16dp"

        tools:ignore="HardcodedText,SpUsage,InefficientWeight" />


```

```
67      <TextView
68
69          android:id="@+id/creditos"
70
71          android:layout_width="wrap_content"
72
73          android:layout_height="wrap_content"
74
75          android:layout_marginLeft="15dp"
76
77          android:layout_marginRight="15dp"
78
79          android:text="creditos"
80
81          android:textAppearance="?android:attr/textAppearanceLarge"
82
83          android:textSize="16dp"
84
85          tools:ignore="HardcodedText,SpUsage" />
86
87
88
89  </LinearLayout>
```

C.2.23. Interfaces de dispositivos de tamaño muy grande

C.2.24. Interfaz de inicio

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5       xmlns:tools="http://schemas.android.com/tools"
6
7       android:layout_width="match_parent"
8
9       android:layout_height="match_parent"
10
11      android:background="@drawable/fondo" >
12
13
14
15      <TextView
16
17          android:id="@+id/textView1"
18
19          android:layout_width="wrap_content"
20
21          android:layout_height="wrap_content"
22
23          android:layout_alignParentTop="true"
24
25          android:layout_centerHorizontal="true"
26
27          android:layout_marginTop="258dp"
28
29          android:text="Matricula"
30
31          android:textAppearance="?android:attr/textAppearanceLarge"
32
33          android:textSize="30dp"
34
35          tools:ignore="HardcodedText,SpUsage" />
36
37
```

```
38
39      <EditText
40
41          android:id="@+id/editText1"
42
43          android:layout_width="wrap_content"
44
45          android:layout_height="wrap_content"
46
47          android:layout_below="@+id/textView1"
48
49          android:layout_centerHorizontal="true"
50
51          android:layout_marginTop="53dp"
52
53          android:ems="10"
54
55          android:inputType="number" >
56
57
58
59          <requestFocus />
60
61      </EditText>
62
63
64
65      <TextView
66
67          android:id="@+id/textView2"
68
69          android:layout_width="wrap_content"
70
71          android:layout_height="wrap_content"
72
73          android:layout_below="@+id/editText1"
74
75          android:layout_centerHorizontal="true"
76
77          android:layout_marginTop="94dp"
78
79          android:text="Contrasenia"
80
81          android:textAppearance="?android:attr/textAppearanceLarge"
82
83          android:textSize="30dp"
84
85          tools:ignore="HardcodedText,SpUsage" />
86
87
88
89      <EditText
90
91          android:id="@+id/editText2"
92
93          android:layout_width="wrap_content"
94
95          android:layout_height="wrap_content"
96
97          android:layout_alignLeft="@+id/editText1"
98
99          android:layout_below="@+id/textView2"
100
101         android:layout_marginTop="66dp"
102
103         android:ems="10"
104
105         android:inputType="textPassword" />
```

```
106
107
108
109     <Button
110
111         android:id="@+id/button1"
112
113         android:layout_width="wrap_content"
114
115         android:layout_height="wrap_content"
116
117         android:layout_below="@+id/editText2"
118
119         android:layout_centerHorizontal="true"
120
121         android:layout_marginTop="117dp"
122
123         android:text="Ingresar"
124
125         android:textSize="24dp"
126
127         tools:ignore="HardcodedText,SpUsage" />
128
129
130
131 </RelativeLayout>
```

C.2.25. Interfaz de despliegue de recomendación

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3 <RelativeLayout xmlns:tools="http://schemas.android.com/tools"
4
5     xmlns:android="http://schemas.android.com/apk/res/android"
6
7     android:id="@+id/RelativeLayout1"
8
9     android:layout_width="match_parent"
10
11    android:layout_height="match_parent"
12
13    android:background="@drawable/rec"
14
15    android:paddingBottom="@dimen/activity_vertical_margin"
16
17    android:paddingTop="@dimen/activity_vertical_margin"
18
19    android:gravity="center_horizontal"
20
21    tools:context=".Recomendacion"
22
23    tools:ignore="HardcodedText" >
24
25
26
27     <ListView
28
29         android:id="@+id/recomendacion"
30
31         android:layout_width="wrap_content"
32
33         android:layout_height="wrap_content"
34
35         android:layout_alignParentLeft="true"
36
```

```
37            android:layout_alignParentTop="true"
38
39            android:layout_centerHorizontal="true"
40
41            android:layout_marginBottom="600dp"
42
43            android:layout_marginLeft="15dp"
44
45            android:layout_marginRight="15dp"
46
47            android:layout_marginTop="80dp"
48
49            android:divider="#B0C4DE"
50
51            android:dividerHeight="1dp"
52
53            android:headerDividersEnabled="false"
54
55            android:paddingBottom="10dp"
56
57            android:paddingTop="10dp"
58
59            tools:ignore="ObsoleteLayoutParam" >
60
61
62
63    </ListView>
64
65
66
67    <TextView
68
69        android:id="@+id/textView1"
70
71        android:layout_width="wrap_content"
72
73        android:layout_height="wrap_content"
74
75        android:layout_alignLeft="@+id/textView2"
76
77        android:layout_alignTop="@+id/textView2"
78
79        android:layout_marginTop="59dp"
80
81        android:text="Minimo:"
82
83        android:textAppearance="?android:attr/textAppearanceLarge"
84
85        android:textSize="20dp"
86
87        tools:ignore="HardcodedText,SpUsage" />
88
89
90
91    <TextView
92
93        android:id="@+id/textView3"
94
95        android:layout_width="wrap_content"
96
97        android:layout_height="wrap_content"
98
99        android:layout_alignBottom="@+id/textView1"
100
101       android:layout_marginLeft="26dp"
102
103       android:layout_toRightOf="@+id/textView1"
104
```

```
105             android:text="0"
106
107             android:textAppearance="?android:attr/textAppearanceLarge"
108
109             android:textSize="20dp"
110
111             tools:ignore="HardcodedText,SpUsage" />
112
113
114
115         <TextView
116
117             android:id="@+id/textView4"
118
119             android:layout_width="wrap_content"
120
121             android:layout_height="wrap_content"
122
123             android:layout_alignBaseline="@+id/textView3"
124
125             android:layout_alignBottom="@+id/textView3"
126
127             android:layout_marginLeft="101dp"
128
129             android:layout_toRightOf="@+id/textView3"
130
131             android:text="Maximo:"
132
133             android:textAppearance="?android:attr/textAppearanceLarge"
134
135             android:textSize="20dp"
136
137             tools:ignore="HardcodedText,SpUsage" />
138
139
140
141         <TextView
142
143             android:id="@+id/textView5"
144
145             android:layout_width="wrap_content"
146
147             android:layout_height="wrap_content"
148
149             android:layout_alignBaseline="@+id/textView4"
150
151             android:layout_alignBottom="@+id/textView4"
152
153             android:layout_marginLeft="18dp"
154
155             android:layout_toRightOf="@+id/textView4"
156
157             android:text="0"
158
159             android:textAppearance="?android:attr/textAppearanceLarge"
160
161             android:textSize="20dp"
162
163             tools:ignore="SpUsage" />
164
165
166
167         <TextView
168
169             android:id="@+id/textView6"
170
171             android:layout_width="wrap_content"
172
```

```
173             android:layout_height="wrap_content"
174
175             android:layout_alignLeft="@+id/textView1"
176
177             android:layout_below="@+id/textView1"
178
179             android:layout_marginTop="30dp"
180
181             android:text="Creditos acumulados:"
182
183             android:textAppearance="?android:attr/textAppearanceLarge"
184
185             android:textSize="20dp"
186
187             tools:ignore="SpUsage" />
188
189
190
191         <TextView
192
193             android:id="@+id/textView7"
194
195             android:layout_width="wrap_content"
196
197             android:layout_height="wrap_content"
198
199             android:layout_alignBaseline="@+id/textView6"
200
201             android:layout_alignBottom="@+id/textView6"
202
203             android:layout_toRightOf="@+id/textView2"
204
205             android:text="0"
206
207             android:textAppearance="?android:attr/textAppearanceLarge"
208
209             android:textSize="20dp"
210
211             tools:ignore="SpUsage" />
212
213
214
215         <TextView
216
217             android:id="@+id/textView8"
218
219             android:layout_width="wrap_content"
220
221             android:layout_height="wrap_content"
222
223             android:layout_alignLeft="@+id/textView6"
224
225             android:layout_below="@+id/textView6"
226
227             android:layout_marginTop="42dp"
228
229             android:text="Ajustar creditos:"
230
231             android:textAppearance="?android:attr/textAppearanceLarge"
232
233             android:textSize="20dp"
234
235             tools:ignore="SpUsage" />
236
237
238
239         <TextView
240
```

```
241         android:id="@+id/textView9"
242
243         android:layout_width="wrap_content"
244
245         android:layout_height="wrap_content"
246
247         android:layout_alignBaseline="@+id/textView8"
248
249         android:layout_alignBottom="@+id/textView8"
250
251         android:layout_alignRight="@+id/textView6"
252
253         android:layout_marginRight="17dp"
254
255         android:text="0"
256
257         android:textAppearance="?android:attr/textAppearanceLarge"
258
259         android:textSize="20dp"
260
261         tools:ignore="SpUsage" />
262
263
264     <SeekBar
265
266         android:id="@+id/seekBar1"
267
268         android:layout_width="match_parent"
269
270         android:layout_height="wrap_content"
271
272         android:layout_alignLeft="@+id/textView8"
273
274         android:layout_below="@+id/textView8"
275
276         android:layout_marginRight="40dp"
277
278         android:layout_marginTop="24dp" />
279
280
281     <RadioGroup
282
283         android:id="@+id/radioGroup1"
284
285         android:layout_width="match_parent"
286
287         android:layout_height="wrap_content"
288
289         android:layout_alignParentBottom="true"
290
291         android:layout_marginBottom="96dp"
292
293         android:layout_marginLeft="20dp" >
294
295
296         <RadioButton
297
298             android:id="@+id/radio0"
299
300             android:layout_width="wrap_content"
301
302             android:layout_height="wrap_content"
303
304             android:checked="false"
305
306
307
308
```

```
309             android:text="Optativas Inter - Multidisciplinares"
310
311             android:textSize="20dp"
312
313             tools:ignore="SpUsage" />
314
315
316
317         <RadioButton
318
319             android:id="@+id/radio1"
320
321             android:layout_width="wrap_content"
322
323             android:layout_height="wrap_content"
324
325             android:text="Optativas de Integracion"
326
327             android:textSize="20dp"
328
329             tools:ignore="SpUsage" />
330
331
332
333         <RadioButton
334
335             android:id="@+id/radio2"
336
337             android:layout_width="wrap_content"
338
339             android:layout_height="wrap_content"
340
341             android:text="Proporciona tu óopinin"
342
343             android:textSize="20dp"
344
345             tools:ignore="SpUsage" />
346
347
348
349     </RadioGroup>
350
351
352
353     <TextView
354
355         android:id="@+id/textView2"
356
357         android:layout_width="wrap_content"
358
359         android:layout_height="wrap_content"
360
361         android:layout_above="@+id/radioGroup1"
362
363         android:layout_alignParentLeft="true"
364
365         android:layout_marginBottom="257dp"
366
367         android:layout_marginLeft="35dp"
368
369         android:text="Creditos recomendados"
370
371         android:textAppearance="?android:attr/textAppearanceLarge"
372
373         android:textColor="#df0101"
374
375         tools:ignore="HardcodedText" />
376
```

```
377
378
379     <TextView
380
381         android:id="@+id/textView13"
382
383         android:layout_width="wrap_content"
384
385         android:layout_height="wrap_content"
386
387         android:layout_alignBottom="@+id/textView9"
388
389         android:layout_toRightOf="@+id/textView2"
390
391         android:text="Ajuste posible:"
392
393         android:textAppearance="?android:attr/textAppearanceLarge"
394
395         android:textColor="#df0101"
396
397         android:textSize="20dp"
398
399         tools:ignore="SpUsage" />
400
401
402
403     <TextView
404
405         android:id="@+id/rajuste"
406
407         android:layout_width="wrap_content"
408
409         android:layout_height="wrap_content"
410
411         android:layout_above="@+id/seekBar1"
412
413         android:layout_marginLeft="16dp"
414
415         android:layout_toRightOf="@+id/textView13"
416
417         android:text="0"
418
419         android:textAppearance="?android:attr/textAppearanceLarge"
420
421         android:textColor="#df0101"
422
423         android:textSize="20dp"
424
425         tools:ignore="SpUsage" />
426
427
428
429 </RelativeLayout>
```

C.2.26. Interfaz de despliegue de optativas de integración hábiles

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5     xmlns:tools="http://schemas.android.com/tools"
6
7     android:id="@+id/RelativeLayout1"
8
9     android:layout_width="fill_parent"
```

```
10
11      android:layout_height="fill_parent"
12
13      android:background="@drawable/ti"
14
15      android:gravity="center_horizontal"
16
17      android:orientation="vertical"
18
19      android:padding="5dp" >
20
21
22
23      <ListView
24
25          android:id="@+id/integracion"
26
27          android:layout_width="wrap_content"
28
29          android:layout_height="wrap_content"
30
31          android:layout_alignParentLeft="true"
32
33          android:layout_alignParentTop="true"
34
35          android:layout_marginBottom="100dp"
36
37          android:layout_marginLeft="10dp"
38
39          android:layout_marginRight="10dp"
40
41          android:layout_marginTop="100dp"
42
43          android:divider="#B0C4DE"
44
45          android:dividerHeight="1dp"
46
47          android:headerDividersEnabled="false"
48
49          android:paddingBottom="10dp"
50
51          android:paddingTop="10dp"
52
53          tools:ignore="ObsoleteLayoutParam" >
54
55      </ListView>
56
57
58
59  </RelativeLayout>
```

C.2.27. Interfaz de despliegue de optativas multidisciplinares hábiles

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5       xmlns:tools="http://schemas.android.com/tools"
6
7       android:id="@+id/RelativeLayout1"
8
9       android:layout_width="fill_parent"
10
11      android:layout_height="fill_parent"
12
```

```
13      android:background="@drawable/inter"
14
15      android:gravity="center_horizontal"
16
17      android:orientation="vertical"
18
19      android:padding="5dp" >
20
21
22
23      <ListView
24
25          android:id="@+id/multidisciplinares"
26
27          android:layout_width="wrap_content"
28
29          android:layout_height="wrap_content"
30
31          android:layout_alignParentLeft="true"
32
33          android:layout_alignParentTop="true"
34
35          android:layout_marginBottom="100dp"
36
37          android:layout_marginLeft="10dp"
38
39          android:layout_marginRight="10dp"
40
41          android:layout_marginTop="100dp"
42
43          android:divider="#B0C4DE"
44
45          android:dividerHeight="1dp"
46
47          android:headerDividersEnabled="false"
48
49          android:paddingBottom="10dp"
50
51          android:paddingTop="10dp"
52
53          tools:ignore="ObsoleteLayoutParam" >
54
55      </ListView>
56
57
58
59  </RelativeLayout>
```

C.2.28. Interfaz de envió de comentarios

```
1   <?xml version="1.0" encoding="utf-8"?>
2
3   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4
5       xmlns:tools="http://schemas.android.com/tools"
6
7       android:id="@+id/LinearLayout1"
8
9       android:layout_width="match_parent"
10
11      android:layout_height="match_parent"
12
13      android:background="@drawable/opinion"
14
15      android:gravity="center_horizontal"
```

```
16
17      android:orientation="vertical" >
18
19
20      <EditText
21
22          android:id="@+id/comentario"
23
24          android:layout_width="match_parent"
25
26          android:layout_height="wrap_content"
27
28          android:layout_marginLeft="30dp"
29
30          android:layout_marginRight="30dp"
31
32          android:layout_marginTop="300dp"
33
34          android:ems="10"
35
36          android:inputType="textMultiLine" >
37
38
39          <requestFocus />
40
41      </EditText>
42
43
44      <TextView
45
46          android:id="@+id/textView1"
47
48          android:layout_width="wrap_content"
49
50          android:layout_height="wrap_content"
51
52          android:layout_marginLeft="30dp"
53
54          android:layout_marginRight="20dp"
55
56          android:gravity="left"
57
58          android:maxLines="3"
59
60          android:text="*El mensaje sera enviado al Coordinador de estudios: tph@correo.azc.
                uam.mx"
61
62          android:textSize="17dp"
63
64          android:textStyle="bold|italic"
65
66          tools:ignore="HardcodedText,SpUsage" />
67
68
69      <Button
70
71          android:id="@+id/button1"
72
73          android:layout_width="wrap_content"
74
75          android:layout_height="wrap_content"
76
77          android:layout_marginTop="200dp"
78
79
80
81
82
```

```
        android:text="Enviar"

        android:textSize="24dp"

        tools:ignore="HardcodedText,SpUsage" />


</LinearLayout>
```

C.2.29. Diseño de la lista de despliegue de recomendación

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/LinearLayout1"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:orientation="horizontal" >


    <TextView

        android:id="@+id/clave"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_gravity="center_horizontal"

        android:layout_marginRight="5dp"

        android:text="clave"

        android:textAppearance="?android:attr/textAppearanceLarge"

        android:textSize="20dp"

        tools:ignore="HardcodedText,SpUsage,InefficientWeight" />


    <TextView

        android:id="@+id/uea"

        android:layout_width="408dp"

        android:layout_height="wrap_content"

        android:layout_gravity="center_horizontal"

        android:layout_marginLeft="10dp"

        android:layout_marginRight="15dp"

        android:maxLines="5"
```

```
54
55          android:minLines="1"
56
57          android:text="uea"
58
59          android:textAppearance="?android:attr/textAppearanceLarge"
60
61          android:textSize="20dp"
62
63          tools:ignore="HardcodedText,SpUsage,InefficientWeight" />
64
65
66
67      <TextView
68
69          android:id="@+id/creditos"
70
71          android:layout_width="wrap_content"
72
73          android:layout_height="wrap_content"
74
75          android:layout_marginLeft="15dp"
76
77          android:layout_marginRight="15dp"
78
79          android:text="creditos"
80
81          android:textAppearance="?android:attr/textAppearanceLarge"
82
83          android:textSize="20dp"
84
85          tools:ignore="HardcodedText,SpUsage" />
86
87
88
89  </LinearLayout>
```