UNet

Juan Andrés Bueno Hortua, Juan David Díaz García y Juan Diego Tovar Cárdenas
No. de Equipo Trabajo: 7

I. INTRODUCCIÓN

A continuación se realizarán las especificaciones del problema de aplicación a resolver como proyecto de clase. Para esto se realizará una descripción funcional del proyecto, se presentará el segundo avance realizado, se hará un breve análisis de los resultados obtenidos y una comparación con los resultados de la primera entrega.

II. DESCRIPCIÓN DEL PROBLEMA A RESOLVER

Actualmente el acceso a cierta información académica no es óptima y puede mejorar. El SIA no siempre permite ver el contenido de la asignatura, los conocimientos necesarios para verla o sus prerrequisitos. Esto lleva a que el estudiante tenga dificultades en su inscripción de materias ya que no tiene la información necesaria para tener seguridad sobre las materias que va a ver o desea ver debido a que el sistema no se la proporciona de manera adecuada. De manera análoga, en muchos casos, dada la falta de información de prerrequisitos, el estudiante no sabe qué materias puede ver y pierde la oportunidad de inscribir asignaturas interesantes porque desconocía la posibilidad de inscribirla.

Nuestra propuesta tiene como objetivo dar al estudiante un entorno amigable donde pueda reportar y visualizar su avance en su plan de estudios, así mismo como las materias que puede o no inscribir, además de sus respectivos planes, prerrequisitos y desbloqueos.

III. USUARIOS DEL PRODUCTO DE SOFTWARE

Los usuarios de la aplicación serán estudiantes activos del Departamento Ingeniería Mecánica y Mecatrónica de la Facultad de Ingeniería de la Universidad Nacional de Colombia que deseen organizar y visualizar sus avances de carrera. Podrán hacer uso diario de la aplicación y acceder tanto a sus datos personales como a las de las diferentes asignaturas.

IV. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

A. *Mostrar la malla curricular de la carrera de manera gráfica*
   1) *Descripción:* se muestra en pantalla la malla curricular y permite navegar por la misma.
   2) *Acciones iniciadoras y comportamiento esperado:* seleccionar el plan de estudio que desea consultar. El programa mostrará las materias correspondientes de dicho plan de forma esquemática por semestres.
   3) *Requerimientos:* acceder a las materias almacenadas de ese plan y su información básica como el nombre y mostrarlas en pantalla.

B. *Mostrar información general de la malla*
   1) *Descripción:* se muestran en colores las materias aprobadas, la nota de las mismas, también o materias disponibles para inscribir.
   2) *Acciones iniciadoras y comportamiento esperado:* el usuario puede elegir entre dos botones dependiendo si quiere observar las materias aprobadas o disponibles para inscribir. Una vez oprimido, la malla colorea las materias que cumplan estas condiciones.
   3) *Requerimientos:* consultar las estructuras de datos que guardan las materias que cumplan dicha condición.

C. *Mostrar y editar información particular de una materia del plan*
   1) *Descripción:* se muestra la tipología de la asignatura, nota de la misma, prerrequisitos, y que asignaturas desbloquea después de ser aprobada. Se considera también incluir una breve descripción de la materia
   2) *Acciones iniciadoras y comportamiento esperado:* una vez en el plan deseado, se selecciona la materia que se desea consultar y esta mostrará toda su información asociada.
   3) *Requerimientos:*
      i. Busqueda parcial en el conjunto de todas las materias y acceso a su información.
      ii. Edición y actualización de las notas.
      iii. Solo admite valores entre 0.0 y 5.0, de lo contrario, impide el registro de la información y avisa al usuario.

D. *Inserción, edición o eliminación de asignatura de libre elección u optativa*
   1) *Descripción:* dado que hay un gran catálogo elegibles en el componente de libre elección, se propone que el estudiante inserte manualmente las materias de libre elección o seleccione las materias optativas cursadas especificando mínimamente el nombre y el numero de creditos.
   2) *Acciones iniciadoras y comportamiento esperado:* se selecciona los créditos de libre elección y se insertan una o varias asignaturas. Se solicitará el nombre y el número de créditos. En caso de superar los 8 se mostrará un cuadro de texto preguntando si se tiene seguridad del valor (sólo serán entradas válidas los números naturales). En el caso de las optativas se puede seleccionar de una lista predefinida. Así mismo se permite la eliminación de las mismas. Se irá añadiendo gráficamente a la malla.
   3) *Requerimientos:*
      i. Búsqueda parcial de las materias cursadas, obtención de la nota de las mismas.
      ii. En caso de ingresar datos no admitidos, se detendrá el registro y se notificará al usuario del error.
      iii. Modificación de la interfaz

E. Calcular P.A.P.A.

1) *Descripción:* se calcula y muestra el P.A.P.A. del estudiante.
2) *Acciones iniciadoras y comportamiento esperado:* al editar alguna nota se recalcula el PAPA. Para consultarlo, se accede mediante un botón.
3) *Requerimientos:* busqueda parcial de las materias cursadas, obtención de la nota de las mismas y cálculo matemático.

F. Mostrar avance de carrera:

1) *Descripción:* dependiendo del número de créditos aprobados respecto a los créditos obligatorios, una ilustración avanzara por un camino que llega a un cartón de grado, representando cuanto falta para su graduación. A su vez, mostrará el porcentaje de avance general y por componente.
2) *Acciones iniciadoras y comportamiento esperado:* una vez se seleccione el plan de estudio deseada, esta opción estará será desplegada gráficamente para su consulta debajo de la malla.
3) *Requerimientos:* consultar listas que guardan las materias aprobadas y hacer los cálculos requeridos para mostrarlos en la interfaz.

V. AVANCE EN LA IMPLEMENTACIÓN DE LA INTERFAZ DE USUARIO

Mockups primera entrega:

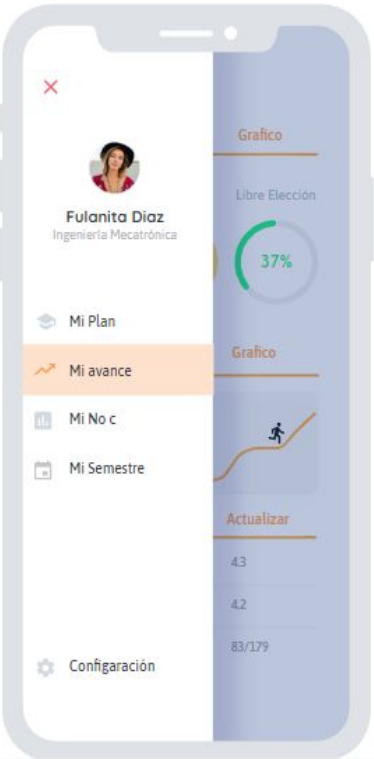

Fig. 1 Mockup opciones de la aplicación

Figura 2: Mockup información disponible en la sección mi plan

Fig. 3 Mockup información disponible en la sección mi avance

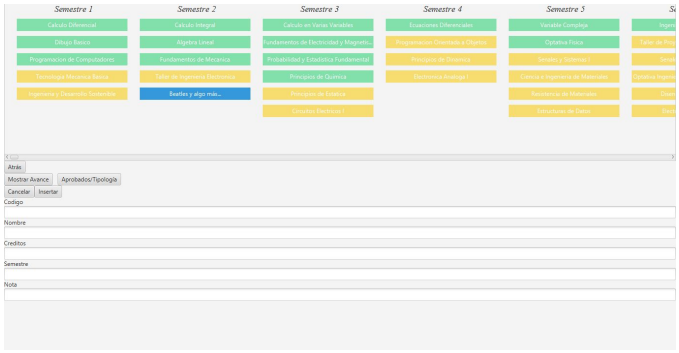

Fig. 4 Interfaz gráfica mostrando las materias según tipología

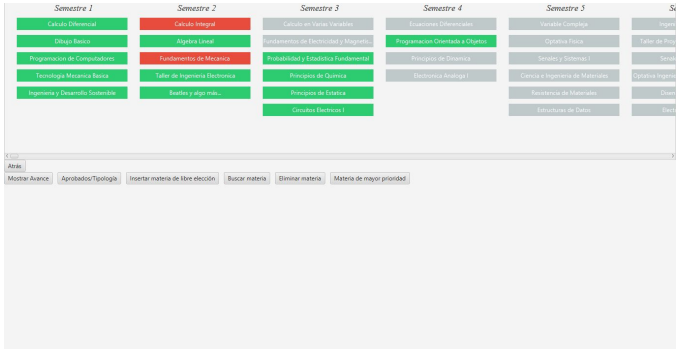

Fig. 5 Interfaz gráfica mostrando las materias según el estado de aprobado, reprobado o no cursado

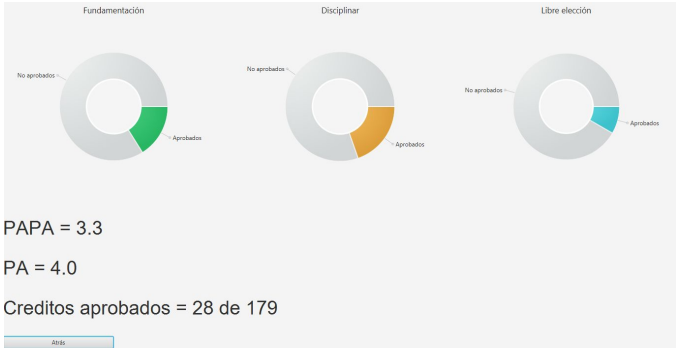

Fig. 6 Pantalla que muestra el avance en cada componente del plan de manera porcentual

VI. ENTORNOS DE DESARROLLO Y DE OPERACIÓN

Dada la portabilidad deseada para el producto, se considera que la mejor opción para el desarrollo de la aplicación será Android Studio, con el fin de lograr operar en dispositivos de sistema operativo Android versión 5.0 o superior. Sin embargo, se diseñarán ciertas partes del código por separado en Netbeans (JavaFx para la parte gráfica) para después ser implementarlas en Android Studio.

VII. DESCRIPCIÓN GENERAL DEL SEGUNDO PROTOTIPO IMPLEMENTADO

En la primera entrega, se hizo entrega de un primer prototipo que cumplia los requerimentos A y D de forma muy rudimentaria a nivel gráfico como de las estructuras utilizadas. Para esta segundo avance, se hizo el cambio de dichas estructuras por unas más eficientes como árboles y pilas sin afectar la funcionalidad. No obstante, algunas implementaciones, a consideración del equipo, no fueron alteradas ya que coordinaban adecuadamente con las nuevas estructuras en términos de eficiencia.

Lo anterior nos permitió enfocarnos en desarrollar el resto de requerimientos inicialmente planteados. En este avance, logramos incorporar las funcionalidades B, C, E y F por el lado lógico, además de mejorar las anteriormente incorporadas.

A nivel gráfico, nos encargamos de diseñar las vistas necesarias y que no habían sido creadas previamente y de mejorar la estética general de la interfaz. Con ello, logramos satisfacer los requerimientos C y F por el lado gráfico.

VIII. NUEVAS ESTRUCTURAS DE DATOS IMPLEMENTADAS.

A. *Árboles*

Implementamos de un árbol binario AVL que contuviera la información de la ubicación de una materia en la malla. Se guardó los datos del semestre, la posición en el semestre y el código. El último se utilizó como dato de organización del arbol. Con esta información, podemos a acceder a la matriz de materias por semestre de forma directa y realizar inserciones, consultas y eliminaciones.

B. *Pilas*

Las pilas nos permiten acceder al último elemento añadido a ella. Dicha abstracción fue aplicada a dos funciones: navegación por vistas. En esta, el último elemento es la última vista accedida lo que permite una navegación lineal. La implementación se hizo con una lista simplemente encadenada en la que se inserta y elimina por la cabeza.

C. *Colas de prioridad*

Uno de los objetivos del proyecto es poder informar al usuario acerca de las materias que debía inscribir con prioridad, para ellos se utilizó un Max Heap que organizaba las materias según el semestre, el número de materias que la tenían como prerrequisito y finalmente el código. Con esto se logra que el usuario no se atrase con materias de las diferentes líneas de su carrera.

IX. PRUEBAS DEL PROTOTIPO

En la siguiente tabla se presentan los tiempos de ejecución de las tres principales funcionalidades del prototipo usando listas y árboles que corresponden con la estructura principal implementada en cada entrega:

TABLA I
Tiempos de ejecución en milisegundos de las principales funcionalidades para distintos volúmenes de datos

N datos	Tiempos de ejecución por operaciones [ms]					
	Cargar materias		Consultar materias		Ingresar electiva	
	Listas	Árboles	Listas	Árboles	Listas	Árboles
500	287	52	3	0	0	0
1k	160	58	4	0	1	1
5k	407	75	5	0	1	1
10k	469	80	5	0	3	1
50k	11725	98	14	0	13	1
100k	78457	104	24	1	17	8
500k	1232902	126	OME	2	OME	9
1M	3400255	138	OME	2	OME	9
Complejidad	O(n)	*O(log(n))	O(n)	O(log(n))	O(n)	O(log(n))

OME = OutOfMemoryError *Amortizado

A continuación se presentan las pruebas realizadas sobre las nuevas estructuras de datos que no tienen contraparte en la primera entrega:

TABLA I
Tiempos de ejecución en milisegundos de las nuevas funcionalidades para distintos volúmenes de datos

N datos	Tiempos de ejecución por operaciones [ms]		
	Cargar materias	Eliminar materia raíz	Ingresar electiva
	Max Heap	Max Heap	Max Heap
500	8	0	0
1k	19	0	0
5k	28	0	0
10k	715	0	0
50k	10426	0	1
100k	21126	0	1
500k	360299	0	2
1M	1130053	0	2
Complejidad	O(nlog(n))	O(log(n))	O(log(n))

X. ANÁLISIS COMPARATIVO Y RESULTADOS DE PRUEBAS

Es evidente que la aplicación de las nuevas estructuras de datos disminuye de manera notable los tiempos necesarios para diferentes operaciones , sin embargo el uso simultáneo de diferentes estructuras de datos implica un gasto de memoria mayor, así si bien se logra una reducción en las operaciones de búsqueda se genera una mayor carga en la memoria del programa, este factor cobra peso a partir de las pruebas de 500 mil datos.

Se atribuye la baja capacidad de memoria a la gran cantidad de datos o información almacenada por cada objeto de interés, en este caso cada objeto materia tiene un total de 6 datos (nombre, código, semestre, tipología, prerrequisitos, nota, posición en el semestre). Además dado el uso de estructuras de datos de manera simultánea se aumenta el gasto de memoria.

Cabe explicar que se conservaron ciertas estructuras lineales dado que ciertas funcionalidades como la de dibujo, siempre tendría un costo lineal (dado que hay que dibujar cada elemento) así una estructura lineal se adapta mejor a esta tarea.

Así en conclusión la implementación de las nuevas estructuras, en específico la de los árboles de búsqueda AVL y la de los Heaps aumenta en gran medida la eficiencia y disminuyen el costo de las operaciones, pero así mismo implican un gasto de memoria mayor

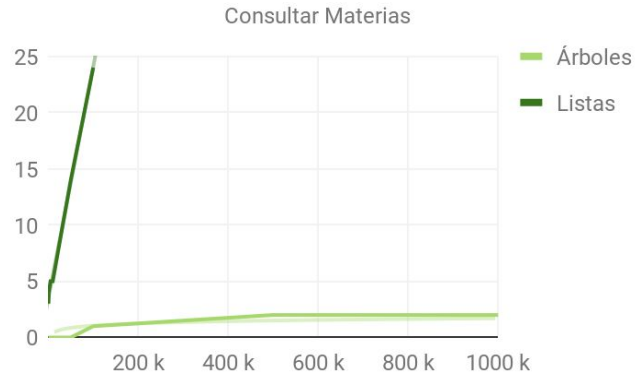

Fig. 7. Comparación de tiempo de ejecución (en ms) entre árboles y listas para la funcionalidad consultar materias

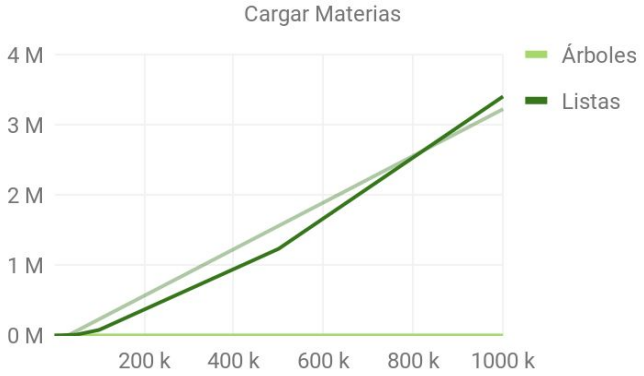

Fig. 8. Comparación de tiempo de ejecución (en ms) entre árboles y listas para la funcionalidad cargar materias

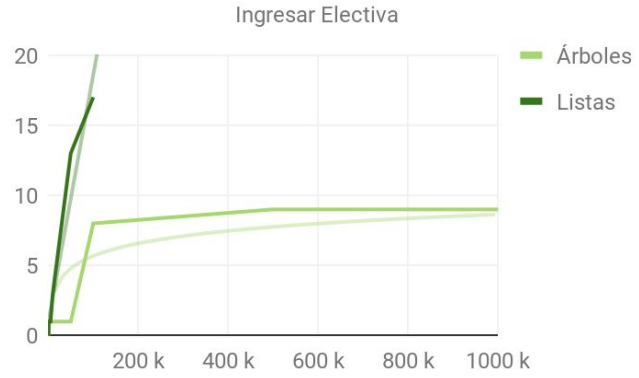

Fig. 9. Comparación de tiempo de ejecución (en ms) entre árboles y listas para la funcionalidad ingresar electiva

Estos fueron los resultados de las funcionalidades probadas en los Max Heaps para diferentes volúmenes de datos:

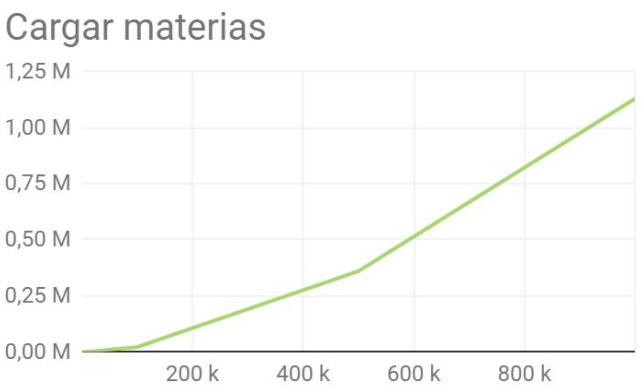

Fig. 10 Tiempo de ejecución (en ms) según la cantidad de datos para la funcionalidad cargar materias en el Heap

Fig. 11. Tiempo de ejecución (en ms) según la cantidad de datos para la funcionalidad eliminar la materia de mayor prioridad en el Heap

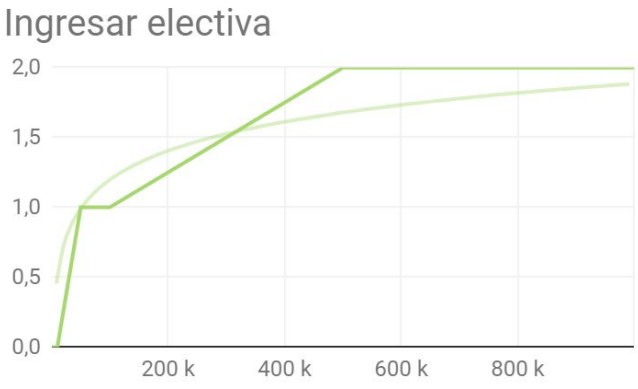

Fig. 12 Tiempo de ejecución (en ms) según la cantidad de datos para la funcionalidad ingresar una electiva en el Heap

Estos fueron los resultados de las funcionalidades probadas en los Árboles AVL para diferentes volúmenes de datos:

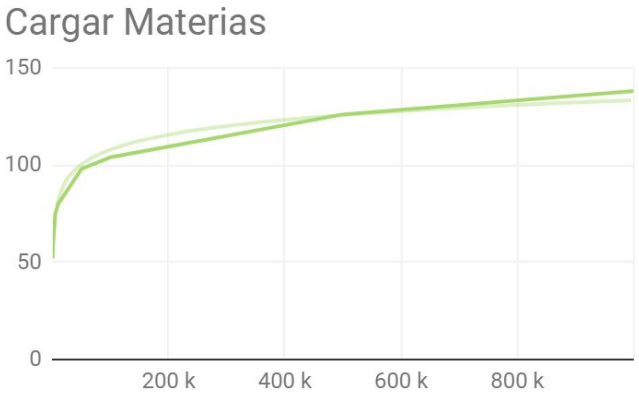

Fig. 13 Tiempo de ejecución (en ms) según la cantidad de datos para la funcionalidad cargar materias usando Árboles AVL

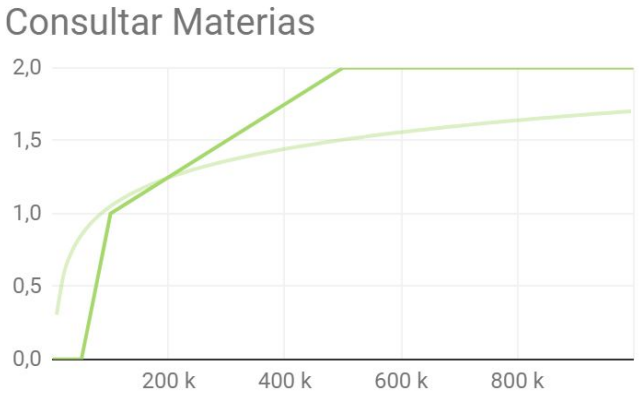

Fig. 14.Tiempo de ejecución (en ms) según la cantidad de datos para la funcionalidad consultar materias usando arboles AVL

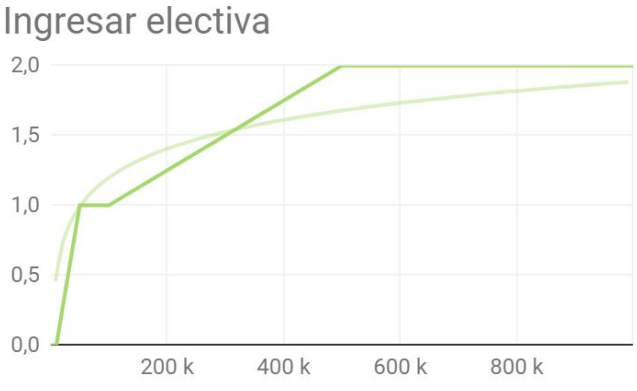

Fig. 15.Tiempo de ejecución (en ms) según la cantidad de datos para la funcionalidad ingresar una electiva usando arboles AVL

XI. DIFICULTADES Y LECCIONES APRENDIDAS

A continuación se resumen las dificultades de la segunda entrega del proyecto de clase y qué lección se aprendió por cada una de ellas:

TABLA III
Dificultades y lecciones aprendidas durante la primera entrega del proyecto de clase

Dificultad	**Lección aprendida**
Adaptación de las nuevas estructuras en el proyecto	Se aprehendió la implementación y el uso de las estructuras nuevas aprendidas en clases
Horarios de trabajo	Organizar con antelación un horario en el que todos puedan trabajar y evitar el trabajo sin la presencia de todo el grupo
Manejo de los estilos de la interfaz gráfica de JavaFX	Se adquirió experiencia con las opciones de las interfaces y fueron aplicadas al prototipo