Proyecto 3

Luccas Rojas - 201923052

Juan Andrés Romero – 202013449

**Documento de diseño Proyecto 3**

Este documento se divide en varias partes, la primera, explicación sobre el formato y contenido de los archivos que utiliza la aplicación, y la segunda, explicación sobre detalles a tomar en cuenta dentro de la implementación de la aplicación, tercera justificaciones del diseño de la aplicación y cuarta un paso a paso cómo utilizar la aplicación.

**\* Parte 1**

El proyecto 2 se compone de lógica e interfaz para la aplicación de banner, cabe destacar distintos aspectos tenidos en cuenta para la carga del pensum y las materias de un estudiante, los cuales fueron:

1. La estructura del pensum es la siguiente:

Cada fila (materia) contiene:

-Nombre de la Materia

-Código

-Prerrequisitos

-Correquisitos

-Créditos

-Nivel

-Tipo Materia:

\*Tipos existentes: Curso Obligatorio, Req. Español, Req. Inglés, Electiva CBU CA, Electiva CBU PC, Electiva CBU CO, Tipo E, Tipo Épsilon, Curso de Libre Elección, Electiva Profesional, Electiva en Ciencias.

PD: Tipo Épsilon y Tipo E pueden estar contenidas dentro de cualquier tipo. Ej: Electiva CBU CA - Épsilon. Curso Obligatorio – Tipo E, Curso de Libre elección

-16 Semanas

-Semestre Sugerido para ver la materia

2. La estructura de los archivos de avance/registro de materias:

-Nombre del Estudiante

-Código

-Carrera

Materias vistas:

Código de la Materia | Nota | Créditos | Semestre en que la vio | Tipo de la Materia

3. La estructura de los archivos de planeación:

-Nombre del Estudiante

-Código

-Carrera

Materias Vistas:

Código de la Materia | Semestre en que se planeó ver

4. Estructura del archivo de reglas del pensum

Código Materia Vieja | Código de materia a homologar

Archivos incluidos en la carpeta data:

currículoISIS.csv - Currículo de la carrera en formato csv

currículoISISMod.csv – Currículo modificado de la carrera en formato csv

materias202013449.csv - Ejemplo de estudiante que ha cumplido con todos los requisitos para graduarse

prueba1.csv - Ejemplo 2 de estudiante con solo 2 materias

reglas.csv – Ejemplo de un archivo de reglas de transición entre pensums

cartelera.csv – Ejemplo de cartelera convertido a csv

Archivo en carpeta logs:

Logs.txt – Ejemplo del archivo de logs de errores

**\* Parte 2**

Las materias dentro de la aplicación se dividen en materias normales, las del pensum (Clase Materia) y las materias vistas por un estudiante (Clase MateriaEstudiante), su diferencia es que MateriaEstudiante es una subclase de materia y tiene atributos adicionales que le sirven al estudiante. Los atributos principales de Materia son:

-Nombre de la materia: String. Representa el nombre de la materia. Ej: Diseño y Programación Orientada a Objetos

-Código: String. Representa el código de la materia dentro del pensum de la carrera. Ej: ISIS-1226

-Prerrequisitos: String. Representa los códigos de los prerrequisitos de la materia, materias que tienen que haber sido vistas para poder inscribir la materia. Nota: un & representa dos prerrequisitos que tienen que haberse visto, un | representa la opción de tomar entre una u otra materia. Ej1: ISIS-1225, Ej2: (ISIS-1104|MATE-1102)&(ISIS-1206|ISIS-1226)&MATE-1105

-Correquisitos: String. Representa los códigos de las materias que son correquisitos de la materia, necesitan haberse inscrito antes de la materia en curso. Nota: un & representa dos requisitos que tienen que haberse visto, un | representa la opción de tomar entre una u otra materia. Ej: LENG-1512|LENG-1622

-Créditos: double. Representa la cantidad de créditos que tiene la materia. Ej: 3, 0.5,1,2, 0.5 etc.

-tipoMateria: String. Representa cualidades diferentes de la materia, los tipos existentes son los siguientes: Curso Obligatorio, Req. Español, Req. Inglés, Electiva CBU CA, Electiva CBU PC, Electiva CBU CO, Tipo E, Épsilon, Curso de Libre Elección, Electiva Profesional, Electiva en Ciencias. Nota: una materia puede tener varios tipos concatenados. Ej: Electiva CBU CA - Épsilon. Ej2: Curso Obligatorio – Tipo E, Curso de Libre elección.

-semanas16: boolean. Representa si el curso es de 16 semanas o de 8 semanas. True indica que es de 16, false indica que es de 8.

-nivel: int. Representa el nivel al cual pertenece la materia. Ej: 3.

-semestreSugerido: Double. Representa el semestre, en el cual el programa sugiere tomar la materia. Ej: 1.

La clase MateriaEstudiante tiene como atributos adicionales:

-Nota: String. Nota es un atributo de tipo String que puede guardar una nota numérica, de 1.5 a 5 o una nota cualitativa, de tipo A (Aprobado), R (Reprobado), PD (Pendiente Disciplinario), I (Incompleto), PE (Pendiente Académico)

- Semestre: double. Semestre es un double que representa el semestre en cual el estudiante vio o planea ver la materia. Los semestres pueden estar en rangos de 0.5, para así representar los intersemestrales o cursos de verano como el periodo entre 2 semestres. Ej, 1, 1.5, 4.5

-materiaRetirada: boolean. materiaRetirada es un atributo tipo booleano que indica si la materia vista sigue inscrita o ha sido retirada por el estudiante. True indica retiro, False indica que la materia sigue estando inscrita.

**Parte 3 – Justificaciones del diseño**

- Se decidió usar strings en la mayoría de los atributos porque esto nos permite guardar distintos tipos de datos como enteros, caracteres, doubles, etc. en una misma variable sin necesidad de tener más de estas. Además, en caso de que se necesite extraer un dato como un double o Int de la cadena, esto es posible a través de un try/catch. También, un String permite ser dividido en un array para la manipulación de elementos que están separados por un carácter especial, ej: un String con los prerrequisitos de una materia, se pueden dividir por el & y así saber qué necesita cumplir el estudiante.

- En la planeación y el reporte de notas para un semestre específico se decidió copiar (clonar) los datos y todo el objeto del estudiante a analizar para poder obtener resultados esperados sin la necesidad de alterar el estudiante inicial.

-Los datos son guardados y cargados mediante archivos csv porque éstos son fáciles de manipular en java y ocupan poco espacio en disco.

-Se decidió repintar paneles en la ventana principal por comodidad del usuario de que todo se realice en la misma ventana (a excepción de popups eventuales).

-Se decidió usar JOptionPanes en vez de JDialogs porque estos tienen una forma mucho más directa de implementación y son más sencillos de utilizar a la hora de obtener los datos.

- Se decidió usar múltiples tipos de materia en un solo atributo para simplicidad y facilidad de acceso a los datos.

-Se decidió crear una excepción propia para poder diferenciar excepciones de Java con las excepciones propias de la aplicación.

-Se decidió usar double para el tipo de dato de los semestres, debido a que fue la manera más sencilla de realizar el cambio de aceptar intersemestrales/curso de verano, etc. del proyecto 2 a este.

-Se decidió guardar un mapa de carteleras en pensum por la facilidad de acceder a él, además no vimos necesaria la creación de una nueva clase para esta, esto se debe a que consideramos que una cartelera/grupo de carteleras no tiene suficiente información propia como para necesitar de una clase propia.

Descripción Breve de las Clases del programa:

-SystemMain: Solía ser la clase principal del programa, pero pasó a ser un controlador mediador entre la lógica y la interfaz gráfica.

-AnalizadorArchivo: Es la clase que realiza todas las operaciones externas con archivos csv, desde carga de datos en el programa hasta guardado de reportes, materias etc.

-Materia: Es la clase que define el objeto de materia. Es usado para guardar las materias normales en el pensum del programa.

-MateriaEstudiante: Es una subclase de materia que le permite a las materias tener atributos que solo existen al haber sido vistas/inscritas por un estudiante.

-Pensum: Es la clase que define el objeto de pensum en el programa. Este guarda las materias de un programa académico y los niveles respectivos de cada materia.

-Estudiante: Es la clase que define el objeto de estudiante en el programa, guarda todos los datos de un estudiante en ella y realiza las operaciones de estos, entre las cuales se encuentra el registro de materias.

-CoordinadorAcademico: Es la clase que define el objeto de Coordinador Académico en el programa, guarda los estudiantes que está analizando y los edita.

-CandidaturaGrado: Es una clase que toma el estudiante, lo analiza y devuelve strings donde especifica si es posible que el estudiante en cuestión se pueda graduar o no.

-Planeador: Es una clase que toma la copia de un estudiante y opera sobre ella lo que el usuario quiera planear para el futuro. Además, permite guardar esta planeación en un csv (utiliza SystemMain).

-ReporteNotas: Es una clase que recibe un estudiante y devuelve un reporte con todos los datos relevantes e historial de las materias del estudiante recibido. De igual forma, puede recibir una copia modificada de un estudiante para devolver el reporte de un solo semestre.

-FiltroCSV: Es una clase utilizada para el guardado de archivos csv.

-VentanaCandidaturaGrado: Es una clase JPanel que dibuja la interfaz gráfica de la candidatura a grado.

-VentanaPlaneador: Es una clase JPanel que dibuja la interfaz gráfica del planeador.

-VentanaReporteNotas: Es una clase JPanel que dibuja la interfaz gráfica del reporte de notas.

-VentanaEstudiante: Es una clase JPanel que dibuja la interfaz gráfica / menú del estudiante, realiza las operaciones necesarias con Estudiante y SystemMain para su operación.

-VentanaCoordinador: Es una clase JPanel que dibuja la interfaz gráfica / menú del coordinador, realiza las operaciones necesarias con estudiantes y SystemMain para su operación.

-VentanaPrincipal: Clase principal de la aplicación con el método main, inicializa el programa y dibuja un simple menú de selección de usuario, esta ventana es repintada por el resto de las clases.

-VentanaReformaPensum: Es una clase JPanel que dibuja una comparativa del reporte de notas

-BannerException: Esta es una clase que extiende Exception, la cual es personalizada y nos permite devolver errores de acuerdo a lo que suceda en la aplicación.

**Parte 4 – Instrucciones de uso**

Para poder utilizar la aplicación como usuario final, es de vital importancia conocer el orden a seguir para poder acceder a todas las funcionalidades de Banner-

1. Primero se debe elegir el tipo de usuario y se debe llenar la información requerida.
2. Luego se debe realizar la carga del pensum desde sus archivos locales.
3. Teniendo el pensum cargado se debe hacer la carga manual de la cartelera para cada uno de los semestres a planear o cursar.
4. Después de tener todas las carteleras y el pensum cargado se deben inscribir las materias, sea manualmente o desde un archivo guardado localmente.
5. En este punto ya tendrá acceso a todas las funcionalidades de banner.

Anexo:

Proceso para cambio de pensum (programas reformados)

Para poder evaluar el cambio de pensum a un programa reformado es necesario.

1. Primero se debe elegir el tipo de usuario y se debe llenar la información requerida.
2. Luego se debe realizar la carga del pensum antiguo desde sus archivos locales.
3. Teniendo el pensum viejo cargado se debe hacer la carga manual de la cartelera para cada uno de los semestres a planear o cursar.
4. Después de tener todas las carteleras y el pensum cargado se deben inscribir las materias, sea manualmente o desde un archivo guardado localmente.
5. Posteriormente se debe intentar cargar un nuevo pensum, en este punto debe cargar desde sus archivos locales tanto el pensum nuevo como las reglas de homologación entre ambos pensums de manera secuencial.
6. Al terminar el paso anterior aparecerá una pantalla donde le permitirá elegir a que pensum acogerse, dependiendo de su elección se abrirá la pantalla principal donde podrá acceder a todas las funcionalidades de banner teniendo en cuenta el pensum elegido.

Todos los errores de los requerimientos funcionales de la aplicación fueron manejados tirando la excepción personalizada “BannerException”, esto ocurre cuando no se puede registrar una materia, hay un error de carga, error al meter los datos de una materia (ej: semestre que no es número, etc.)

**Parte 5 – Cambios con respecto al proyecto 2**

Respecto al proyecto 2 hicimos algunos cambios destacables a la aplicación para que se pudiera adaptar bien a los nuevos requerimientos. Los cambios realizados fueron los siguientes:

1. Cambio de int a double en el atributo de créditos (Clase Materia): Este cambio fue hecho para soportar cursos que tuvieran medio crédito, crédito y medio y así.

2. Cambio de int a double en el atributo de semestre (Clase MateriaEstudiante): Este cambio fue hecho para poder cumplir con el requerimiento de poder inscribir cursos intersemestrales/de verano de manera sencilla y sin tener que cambiar mucho en el programa. En este caso un intersemestral se puede ver como un semestre y medio ej: Visto en semestre 1.5, es decir la materia fue vista en el intersemestral después del semestre 1.

3. El método de inscribir materias cambió la mayoría de sus returns de ints a tirar BannerExceptions, esto pues ahora los errores usan excepciones y el estado correcto solo devuelve un 0.

4. Se cambió el hecho de mirar si la materia a inscribir estaba en el pensum a mirar si esa materia estaba en la cartelera del semestre correspondiente.

5. Se le agregó a estudiante los atributos de cantidad de créditos actuales y último semestre que inscribió, esto para poder calcular la cantidad de créditos por semestre que va teniendo el estudiante a medida que inscribe materias, todo para cumplir con el requerimiento de revisar si el estudiante puede extracreditarse o no.

6. Se quitaron algunos atributos de error de algunas clases puesto que fueron reemplazados por excepciones.

7. Se añadió la funcionalidad de cargar carteleras por semestre a la aplicación

8. Se añadió la posibilidad de realizar un cambio curricular, al realizar esto, el usuario debe cargar un archivo con las reglas de transición entre pensums. Para esto se creó una nueva clase de interfaz gráfica, la cual le permita al estudiante decidir con cuál avance quiere quedarse.

**Parte 6 – Explicación Diagrama**

El programa funciona de tal manera que está dividido en interfaz y sistema, de este modo, la ventana principal es la que contiene el método main que permite ejecutar el programa. Por un lado, se le pregunta al usuario si es estudiante o coordinador, con lo que la VentanaCoordinador y VentanaEstudiante repintan la ventana principal a partir de su decisión. Después, dependiendo de la funcionalidad elegida, la VentanaCandidaturaGrado, VentanaPlaneador, VentanaReformaPensum y VentanaReporteNotas repintan la pantalla principal para poder mostrar la información escogida. Por otro lado, el resto de las funcionalidades se ven evidenciadas en pop-ups que no modifican la ventana principal.

Adicional a la interfaz se encuentra el sistema o la lógica, el systemMain permite tener todas las funcionalidades del programa, conectando el modelo y la vista del programa como si fuera un controlador. En términos de lógica, para que un coordinador o un estudiante pueda acceder a las diferentes funcionalidades, es necesario que sean utilizadas las clases del mundo que tienen el mismo nombre de las interfaces, de este modo, se pueden llamar a las clases de Planeador, ReporteNotas o CandidaturaGrado.

Por otro lado, un estudiante posee una serie de materias, las cuales a su vez pueden ser de tipo MateriaEstudiante, clase que extiende de Materia pero añade la posibilidad de insertar un semestre, una nota y la posibilidad de que la materia esté retirada.

La clase de analizador de archivo le permite a Materia, systemMain, candidaturaGrado, Planeador, ReporteNotas y Usuario leer, crear y modificar información de archivos csv y txt en el computador del usuario. En su contraparte de interfaz, VentanaEstudiante y VentanaCoordinador, utilizan filtro csv, que extiende de FileFilter para así filtrar el tipo de archivos que pueden ser cargados.

Finalmente, la clase Banner Exception es una clase adicional que extiende Exception y permite devolver errores personalizados y adicionales que surjan a la hora de ejecutar una funcionalidad específica del programa.

**Documento de reflexión proyectos**

**¿Qué cosas salieron bien y qué cosas salieron mal?**

En general, el proyecto nos salió bien y fuimos resolviendo los errores que ocurrían a medida que probábamos la aplicación. Pudimos establecer un diseño de la interfaz gráfica relativamente rápida y concisa, lo cual nos permitió plasmarla mejor y más fácilmente en el código. Al principio tuvimos problemas creando la aplicación, debido a que no teníamos muy claro cómo hacerla desde cero, pero pudimos preguntar, resolver nuestras dudas a tiempo y seguimos desarrollando a buen ritmo. Podemos destacar que el diseño de la estructura del pensum salió muy bien debido a que esta pudo admitir todos los cambios que nos pusieron en el proyecto 3 sin necesidad de realizar un cambio drástico. De hecho, esta estructura soportó desde el inicio cambios a requerimientos pedidos en este proyecto.

**¿Qué decisiones resultaron acertadas y qué decisiones fueron problemáticas?**

Empezamos diseñando una estructura del proyecto 1 en un diagrama UML y pensamos cómo lo íbamos a hacer. Posteriormente comenzamos con la implementación y nos dimos cuenta de que muchas decisiones del diseño nos tocaba cambiarlas para la implementación. Sin embargo, nos dimos cuenta de que podíamos utilizar esto a nuestro favor y mantener una estructura similar a la pensada en el diseño. Creemos que fue acertado realizar la estructura del pensum y definir el archivo de este antes de pasar a escribir la aplicación en código, pues esto permitió adaptar nuestro código a como teníamos el pensum. Creemos que fue acertado hacer copias de estudiantes en algunos métodos, ya que esto nos permitía modificar datos mutables sin afectar los datos originales.

**¿Qué tipo de problemas tuvieron durante el desarrollo de los proyectos y a qué se debieron?**

Al principio tuvimos bastantes problemas debido a que no sabíamos muy bien programar en Java y no sabíamos cómo hacer lo que normalmente hacíamos en Python en java. Tuvimos que buscar bastante en internet como hacer todo lo que teníamos planeado de la mejor manera, especialmente el diseño de la interfaz gráfica porque nunca pudimos a la primera vez cuadrar los elementos en la parte de la pantalla que queríamos. Adicionalmente, para el primer proyecto nos dimos cuenta de que había mucho desconocimiento sobre el funcionamiento de banner, lo cual no está explicado en el enunciado del proyecto 1 y debido a esto para el primero proyecto hicieron falta bastantes funcionalidades.