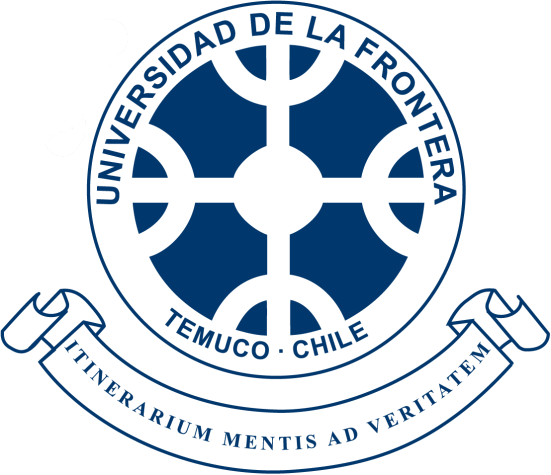
Informe Proyecto 1

Programación



Nombre: Diego Garrido

Profesor: Victor Aravena Diaz

# Índice

Introducción 3

Planteamiento del problema 4

Análisis del problema 5

Diseño del Software 6

Uml

Datos

Ventanas

Conclusión / Comentarios finales 11

# Introducción

Para contextualizar el desarrollo de esta actividad, se debe tener en mente que se está trabajando y poniendo a prueba el Modelado de Sistemas Informáticos, en este caso, utilizando el enfoque Orientado a Objetos ya que es el más útil para desarrollar el problema planteado más adelante.

Además se busca probar el UML y su eficacia a la hora de desarrollar un software, junto con comprobar sus ventajas, como por ejemplo cuando se debe hacer una reestructuración o algún cambio en el proyecto, teniendo un Diagrama de Clases, se realiza la modificación ahí y se debería obtener todo el nuevo código sin mayor esfuerzo.

# Planteamiento del problema

Para el caso de este proyecto se planteó el diseño de un sistema para gestionar datos de un colegio de enseñanza básica de chile, lo que incluye a: los alumnos; sus notas y anotaciones en el libro de clases; la separación por cursos de éstos; apoderados de los alumnos; profesores; asignaturas y asistencia de los alumnos, haciendo hincapié en las siguientes características que debe realizar el programa:

Ingresar la planificación de una asignatura, es decir, las actividades a realizar durante el semestre o año que se cursará esa asignatura.

Crear un vínculo entre los cursos, sus alumnos y sus profesores.

Vincular los apoderados con sus respectivos alumnos.

Permitir a los profesores ingresar las notas de una evaluación.

Permitir ingresar las asistencias a clase.

Ingresar las anotaciones de cada alumno.

# Análisis del problema

Para comenzar, se puede extraer del planteamiento del problema las Clases principales que harán el trabajo de simular los “objetos” de la vida real que utilizaremos en el diseño, de acuerdo a la programación Orientada a Objetos.

Estas clases son: Alumno, Apoderado, Profesor, Curso, Asignatura.

Para el desarrollo del proyecto se pide tener un poblamiento de datos, las características de estos datos también nos pueden ayudar al análisis de los datos del problema.

Debemos añadir “características” y “relaciones” de la vida real entre estos objetos, que nos pueden ser útiles para desarrollar y solucionar el problema con el software, a las Clases que establecimos previamente.

Algunas podrían ser los nombres, para el caso de Apoderado, Alumno, Profesor y asignatura.

Para el caso de la clase Curso vendría de ayuda el identificador, es decir el nivel y la letra, si es que hay más cursos del mismo nivel.

La clase alumno debe tener un identificador para el Curso al que pertenece.

Una asignatura debe tener su planificación, por lo que necesita un atributo que le permita almacenar este dato, también debe tener un profesor que la enseñe, estableciendo así la relación Asignatura-Profesor.

Un profesor debe hacer clases por lo menos a un curso, por lo que debe tener una relación con Curso.

Los alumnos también deben tener cada uno sus Notas y registros de Asistencia, además de estar relacionado con un Apoderado, el cual puede tener más de un hijo en el colegio.

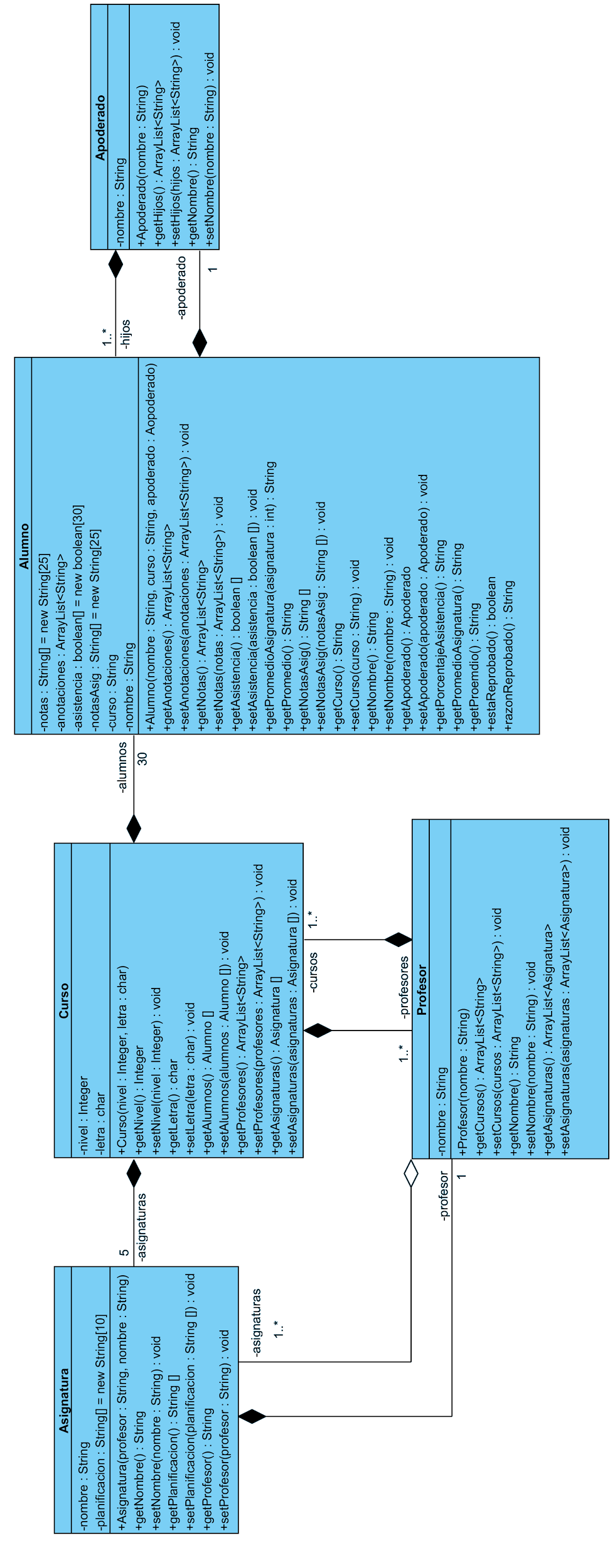
Y por último un Curso necesita tener asignaturas que sean impartidas a sus alumnos.

Se menciona establecer un estado de reprobación de los alumnos, es decir si pasa o no al siguiente curso. Para definir este estado se tomará como requisito para reprobar los mismos utilizados en los colegios básicos de Chile: tener un porcentaje de asistencia menor al 85%. Tener dos asignaturas con promedio rojo y un promedio general menor a 4.5. Tener tres asignaturas con promedio rojo.

También, en las especificaciones del poblamiento de datos se solicita que cada asignatura tenga diez actividades en su planificación, pero no especificaba el detalle de éstas, por lo que se tomará de la siguiente forma: cinco de esas actividades serán las notas que se toman en cuenta para el promedio de la asignatura, estas tienen cada una su ponderación a elección del usuario, de las cinco notas, cuatro son pruebas de la asignatura y la quinta es un promedio de las últimas cinco notas que vendrían a ser talleres o pequeños controles, todos con la misma ponderación.

El promedio general del alumno se toma con la misma ponderación para todos los promedios de cada asignatura.

# Diseño del Software

Luego de haber analizado el problema y los requisitos del poblamiento de datos podemos distribuir las clases y los atributos identificados en un Diagrama de Clases en UML, Poniendo especial énfasis en las relaciones y sus multiplicidades.

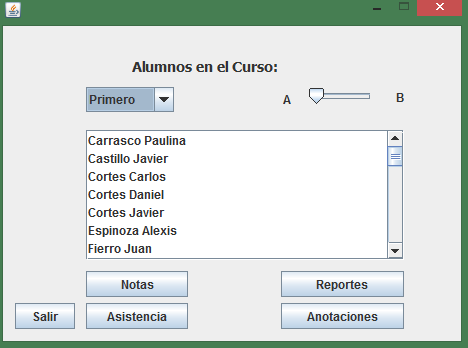
Para no perder los datos ingresados debemos realizar un grabado en el disco duro del computador con los atributos de los objetos, para esto crearemos un Package nuevo con clases que se encargarán de la administración de datos y ficheros.

Para la administración de datos se utilizarán archivos en formato Json y Xml, ya que son muy útiles a la hora de administrar y ordenar datos de los atributos de objetos en Programación orientada a Objetos.

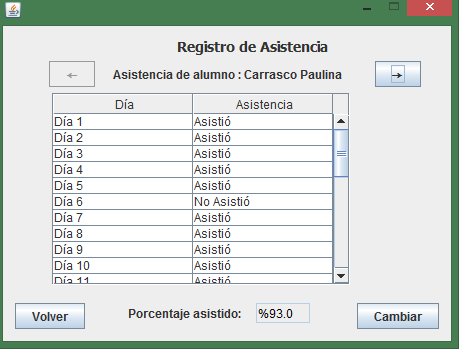
Además debemos realizar un poblamiento de datos, para lo que se creará una Clase externa al programa, la cual generará objetos de tipo Curso y los rellenará con Alumnos, Apoderados y Profesores de Nombres aleatorios y a los Alumnos les proveerá de notas y Registros de asistencia, también aleatorios. Los alumnos tienen su apoderado Respectivo, con el mismo apellido, si hay dos alumnos con un mismo apellido son asignados al mismo apoderado.

Para finalizar debemos crear un Front End para que el usuario interactúe con los datos y los modifique a su gusto, las Ventanas serán almacenadas en otro Package para mantener un orden entre Front End y Back End.

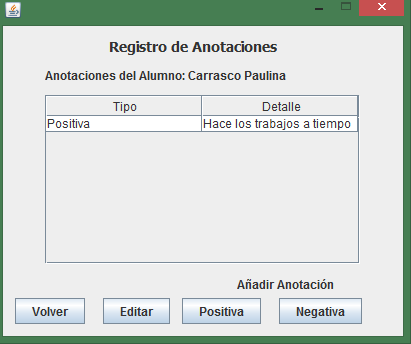
La ventana Curso es la ventana Principal y es la que mostrará a los alumnos en el curso seleccionado por el usuario y dará el paso a las ventanas que modificarán los datos de éstos.



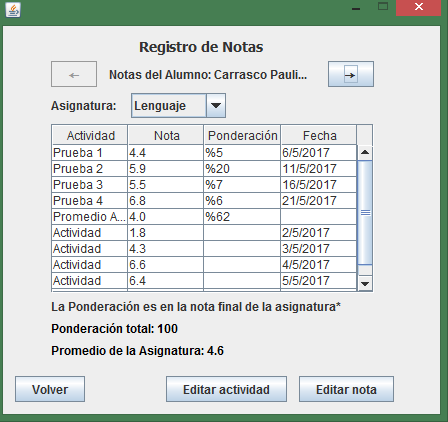
Si se selecciona un alumno de la lista y se presiona en el Botón Asistencia se abrirá la ventana Asistencia, que permitirá ingresar los días que el alumno asistió a clases y muestra el porcentaje de asistencia.



Si en la ventana Curso se selecciona un alumno y se presiona en Anotaciones, se abrirá una ventana que permite añadir una anotación, Positiva o Negativa, muestra las anotaciones actuales y permite Editarlas en caso de que haya habido un error al ingresarla.

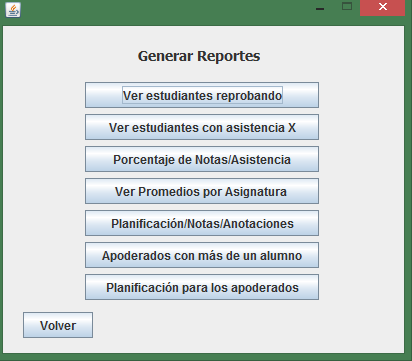


Si en la ventana Curso se Presiona el botón Notas, se abrirá una ventana mostrando las notas del alumno seleccionado. Permitiendo navegar entre las distintas asignaturas que está cursando el alumno, editar las notas y editar las actividades.

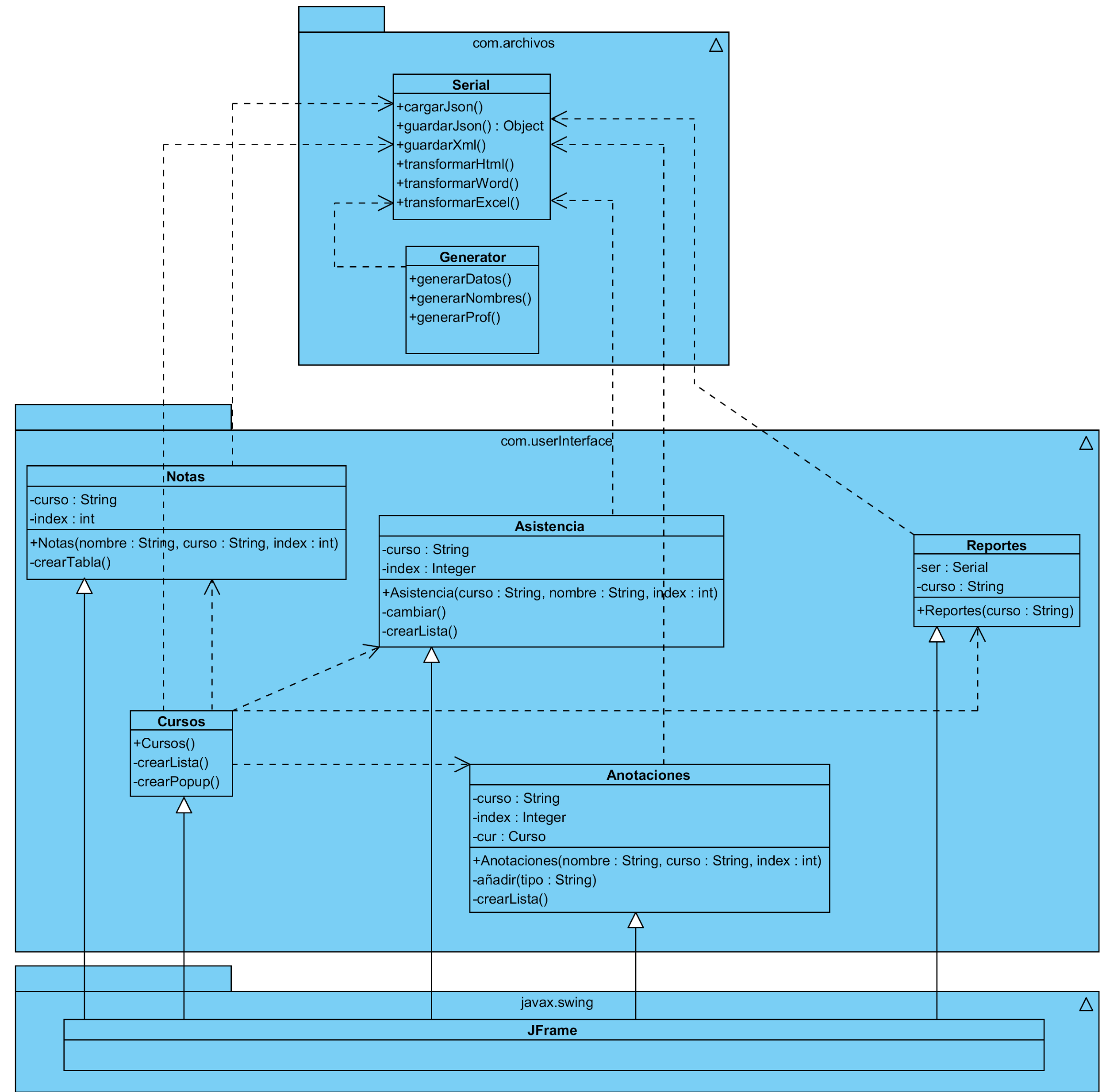


Por último al presionar el botón Reportes se abrirá una ventana con las opciones de guardar los datos solicitados en el planteamiento del problema, en formatos Json, Xml, Word, Excel y Html.

Al seleccionar una opción se guardan automáticamente los cuatro documentos en la Carpeta “reportes” correspondiente al curso anteriormente seleccionado.



Todas estas ventanas interactúan con la Clase administradora de archivos, es utilizada para guardar automáticamente cualquier cambio en los datos que el usuario realice.



# Conclusión / Comentarios finales

El objetivo de este proyecto era poner a prueba la materia pasada en la clase de Programación tal como el uso de Xml y Json para guardar objetos instanciados, el uso de Xsl para la transformación de Xml y comprobar la eficacia del desarrollo de Sistemas de Información con enfoque Orientado a Objetos junto y el uso de UML.

Si bien el uso de Xsl es bastante útil a la hora de leer y convertir Xml, puede ser difícil al principio ya que es una especie de “Lenguaje” para programar formatos de texto con enfoque y sintaxis diferentes a los de Java, pero al comprender la estructura de éste se pueden lograr muchos formatos diferentes a gusto.

Se puso a prueba la ventaja del Diagrama de Clases, cuando se necesitó modificar alguna relación de las clases principales y se comprobó que resulta realmente útil utilizar este diagrama para obtener un código, no fue difícil aplicar el cambio en éstas clases. Sin embargo para el caso de las ventanas no se hizo un diseño previo y fue engorroso realizar un cambio estructural en ellas ya que debía analizar todas las ventanas y si tenían relación con la ventana modificada.