**Método de la Ingeniería**

**1.** **Identificación del Problema**

· **Descripción del contexto problemático**

Con el objetivo de mejorar la gestión de la información de un grupo de estudiantes de Ingeniería de Sistemas de la universidad Icesi, se quiere desarrollar una aplicación que nos permite acceder fácilmente a la información.

Esto sucede ya que, al tener tanta información, y de tantos estudiantes, puede llegar a ser complicado buscar algo específico de alguno de los estudiantes dentro de toda esta información. Además, la edición de la información también puede ser necesaria y de alguna otra manera puede ser más complicado

Gracias a esto, vemos que una aplicación para tener una mejor gestión de toda la información requerida, facilita muchos procesos en los que se necesite cierta información de los estudiantes o de las materias que está cursando.

· **Identificación de Necesidades**

+ Se requiere crear los estudiantes con los que va a contar la agenda

+ Se requiere visualizar los estudiantes que anteriormente han sido agregados, junto con toda su información

+ Se requiere eliminar un estudiante, el cual no va a poder volver a ser visualizado

+ Se requiere editar la información que se desee de cualquiera de los estudiantes

+ Se requiere crear nuevas materias distintas a las ya disponibles

+ Se requiere editar la información de las materias que han sido guardadas en la agenda

+ Se requiere visualizar las materias que están disponibles en la agenda, junto con toda su información

+ Se requiere buscar a los estudiante mediante el criterio elegido, sea nombre, apellido, edad, teléfono, materias, etc.

+ Se requiere mostrar cual es la cantidad promedio de materias que ven los estudiantes

+ Se requiere mostrar cual es la cantidad promedio de créditos que matriculan los estudiantes

+ Se requiere mostrar cual es la materia que mes fue matriculada por los estudiantes, en este semestre

· **Definición del Problema**

En estos momentos, contamos con la información de un grupo de estudiantes, de ingeniería de sistemas, de la universidad Icesi, y de todas las materias que están cursando en este primer semestre del 2020, pero no contamos un sistema de gestión de información eficiente, que nos permita visualizar y analizar la información de una manera más sencilla.

**2.** **Recopilación de Información Necesaria**

· **Agenda:**

“Relación ordenada de asuntos, compromisos o quehaceres de una persona en un período.”[1]

· **Operaciones CRUD:**

“El concepto CRUD está estrechamente vinculado a la gestión de datos digitales. CRUD hace referencia a **un acrónimo** en el que se reúnen las primeras letras de las cuatro operaciones fundamentales de aplicaciones persistentes en sistemas de bases de datos:

* **C**reate (Crear registros)
* **R**ead. **R**etrieve (Leer registros)
* **U**pdate (Actualizar registros)
* **D**elete. **D**estroy (Borrar registros)” [2]

· **Operaciones de Búsqueda**

Existen diferentes tipos de operaciones de búsqueda, unas más eficientes que otras, pero unas más útiles que otras también, todo dependiendo de la situación que se presente. Estos son algunos de los tipos de busqueda que existen:

* **Búsqueda lineal: “**Consiste en recorrer y examinar cada uno de los elementos del array hasta encontrar el o los elementos buscados, o hasta que se han mirado todos los elementos del array.” [3]
* **Búsqueda binaria: “**El algoritmo consiste en reducir paulatinamente el ámbito de búsqueda a la mitad de los elementos, basándose en comparar el elemento a buscar con el elemento que se encuentra en la mitad del intervalo y en base a esta comparación” [3]
* **Búsqueda mediante transformación de claves (hash): “**El método de transformación de claves nos permite encontrar directamente el registro buscado en tablas o archivos que no se encuentran necesariamente ordenados, en un tiempo independiente de la cantidad de datos, todo a través de una asignación de claves, a cada referencia que se necesite” [3]

· **Persistencia**

“Se llama persistencia a la capacidad de guardar la información de un programa para poder volver a utilizarla en otro momento. Es lo que los usuarios conocen como *Guardar el archivo* y después *Abrir el archivo*. Pero para un programador puede significar más cosas y suele involucrar un proceso de *serialización* de los datos a un archivo o a una base de datos o a algún otro medio similar, y el proceso inverso de recuperar los datos a partir de la información *serializada*.” [4]

**3.** **Búsqueda de Soluciones Creativas**

· **Ideas para buscar un estudiante**

- Algoritmo de búsqueda lineal

- Algoritmo de búsqueda binaria

- Algoritmo de búsqueda mediante transformación de claves

· **Ideas para ordenar la lista de contactos**

- Insertion sort

- Selection sort

- Bubble sort

- Cocktail sort

- Quicksort

- Timsort

**4.** **Transición de la formulación de ideas a los diseños preliminares**

· **Ideas para ordenar la lista de contactos**

- **Insertion sort:** Consiste en ir recorriendo la estructura de manera lineal, e ir comparando parejas de elementos, y reemplazar una por otra en caso de que la condición de ordenamiento se cumpla O(*n*2) [5]

- **Selection sort: “**Consiste en encontrar el menor de todos los elementos del vector e intercambiarlo con el que está en la primera posición. Luego el segundo más pequeño, y así sucesivamente hasta ordenarlo todo” [5], por lo que su complejidad temporal es de O(*n*2) [5]

- **Bubble sort: “**La Ordenación de burbuja funciona revisando cada elemento de la lista que va a ser ordenada con el siguiente, intercambiandolos de posición si están en el orden equivocado. Es necesario revisar varias veces toda la lista hasta que no se necesiten más intercambios, lo cual significa que la lista está ordenada.” [5], por lo anterior sabemos que su complejidad temporal es de O(*n*2)

- **Cocktail sort:** Este es una mejora del algoritmo de burbuja ya que tiene la misma metodología, solo que en vez de siempre volver al inicio al final de cada iteración, cuando llega al final, se devuelve comparando desde atrás hacia adelante, por lo que cuenta con la misma complejidad que el burbuja O(*n*2) [5]

- **Quicksort:** Este algoritmo se basa en escoger un elemento de la estructura y compararlo con los elementos restantes, y así ubicarlo en la posición que realmente debe de ir, intercambiando de posición con el elemento que está ocupando su posición, y su complejidad es de 0(*n*2) [5]

- **Timsort:** El algoritmo lo que hace es buscar subsecuencias de elementos que ya estén ordenados, y aprovecha estas para así poder ordenar de una manera más rápida el resto de elementos. [6]

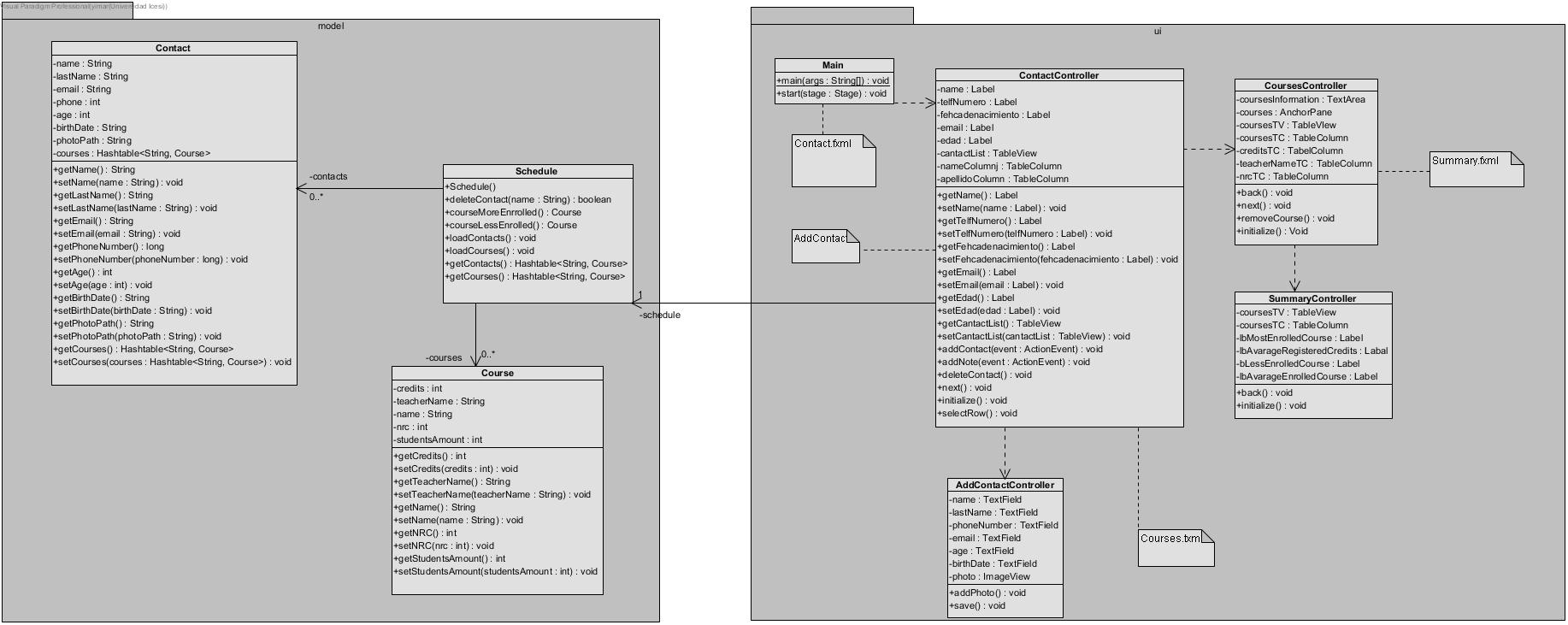
**5.** **Evaluación y selección de la mejor solución**

Las rúbricas van a ser evaluadas de 1 a 5, siendo 1 la nota más baja (para la opción difícil de ejecutar o que no satisfaga las necesidades del proyecto, todo según el campo que esté siendo evaluado en la rúbrica) y 5 la nota más alta (para la opción que sea fácil de ejecutar o que satisfaga las necesidades del proyecto, todo según el campo que esté siendo evaluado en la rúbrica)

- **Ideas para ordenar la lista de contactos**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Costo de implementación** | **Complejidad temporal** | **Complejidad espacial** | **Total** |
| **Insertion** | 1 | 3 | 4 | **8** |
| **Selection** | 1 | 3 | 4 | **8** |
| **Bubble** | 1 | 3 | 4 | **8** |
| **Cocktail** | 1 | 3 | 4 | **8** |
| **Quicksort** | 1 | 3 | 4 | **8** |
| **Timsort** | 5 | 4 | 4 | **13** |

**6.** **Diagramas**

****

# **Referencias**

|  |  |
| --- | --- |
| [1] | Real Academia Española, «RAE,» [En línea]. Available: <https://dle.rae.es/agenda> |
| [2] | Digital Guide IONOS by 1&1, «IONOS,» [En línea]. Available: <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/crud-las-principales-operaciones-de-bases-de-datos/> |
| [3] | Universidad del Cauca, «UNICAUCA,» [En línea]. Available: <http://artemisa.unicauca.edu.co/~nediaz/EDDI/cap02.htm> |
| [4] | Uniwebsidad, «UWS,» [En línea]. Available: <https://uniwebsidad.com/libros/algoritmos-python/capitulo-11/persistencia-de-datos> |
| [5] | Algorithmique, [En línea]. Available: <http://lwh.free.fr/pages/algo/tri/tri_es.htm> |

# 

|  |  |
| --- | --- |
| [6] | Wikipedia, [En línea]. Available: <https://en.wikipedia.org/wiki/Timsort> |