2024

MEMORIA PED POO

TOMÁS SOLANO CAMPOS

UNED | tsolano8@alumno.uned.es

Contenido

1.Objetivo 3

2.Especificaciones 3

2.1.Requerimientos de información de los modelos de clases 3

2.2.Requerimientos funcionales 2

2.3.Requerimientos técnicos 4

3.Diseño 4

4.Conclusiones 6

1.Objetivo

El objetivo de está practica es crear un sistema de administración eficiente sobre el hospital, el hospital tiene muchos datos que en papel serían imposibles de manejar es por eso que esta aplicación resultará tan eficiente en comparación permitiéndonos ahorrar dinero y tiempo en administrar todo lo que engloba el hospital.

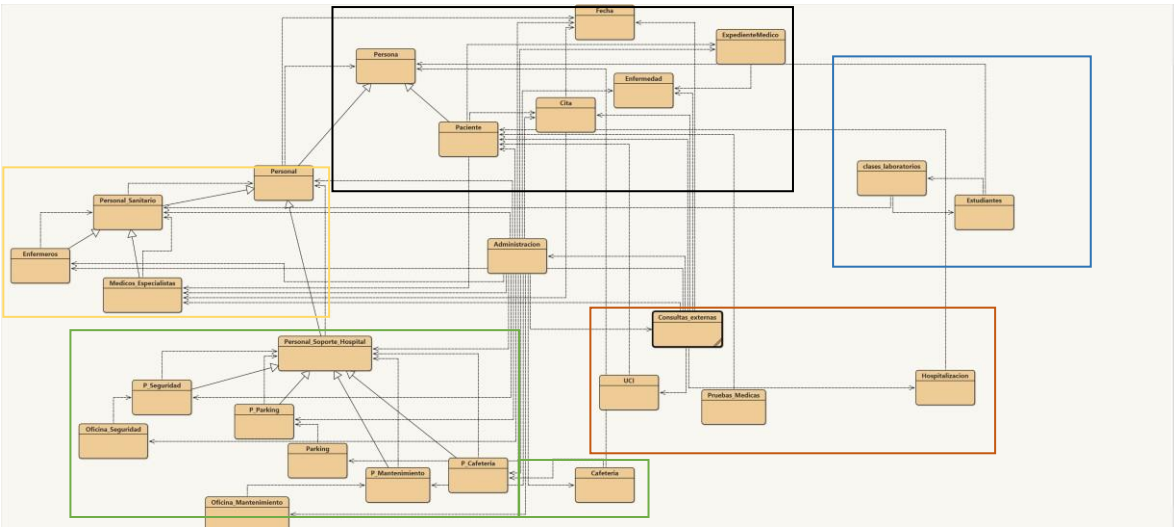
El sistema está pensado para que un administrativo registre en él todos los datos necesarios y tiene todo lo necesario para dar una idea general del estado del hospital y ser un gran apoyo a la hora de administrar fallos de mantenimiento ,citas ,pacientes ,médicos, etc .Sin necesidad de que vaya alguien personalmente a notificar todas estás cosas . A su vez sirve como una base de datos provisional hasta que se puedan conectar con bases de datos reales como SQL.

Por ultimo ,este programa servirá para que los técnicos que lo hayan programado hayan aprendido valiosas lecciones de Java.

2.Especificaciones

2.1.Requerimientos de información de los modelos de clases

El diagrama de relaciones queda así:



Como se puede ver lo he dividido en 5 partes.

- ➢ Negra: Lógica de cada paciente
- ➢ Azul: Lógica universitaria
- ➢ Rojo: Lógica de tratamiento del hospital
- ➢ Verde: Lógica del personal de soporte del hospital
- ➢ Amarillo: Lógica del personal Sanitario

Todas estas partes controladas ,por administración que almacena y modifica todos los datos disponibles de las partes y hace de comunicador entre los datos y el controlador enviándole datos o recibiéndolos. A su vez, tenemos una gran relación de herencia de persona ,la cual puede ser paciente o personal del hospital y a su vez estos tienen sus propias ramas y especialidades. En esta

herencia también existen fuertes dependencias entre cada especialidad y su lugar de trabajo ,enfermeros y médicos especialistas con consultas externas por ejemplo. Fuera de esta tenemos fuertes dependencias entre paciente,cita,enfermedades,expediente médico y fecha. Esto debido a que cada paciente goza de un expediente medico con las enfermedades que tiene y fechas de concepción de cada enfermedad, a su vez tiene citas con el medico es por ello que se ven dependencias a consultas externas y al médico en cuestión. También mencionar que los lugares de trabajo ,al menos los relevantes, tienen cada uno un arrayLists de sus empleados. Esto pese a no ser necesario se hace por si llega a existir varias instancias del mismo lugar de trabajo (dos oficinas de seguridad ,dos oficinas de mantenimiento ,etc).

A su vez ,decidí que controlador vería algunas clases importantes aparte de administración, y que la interfaz pudiera identificar ciertas clases para poder seleccionar una instancia en concreto de esas clases.

Por último ,los estudiantes ,que aunque en esta práctica no hagan nada tienen dependencias con su clase y con su tutor ,que es un miembro del personal sanitario ,ya sea un médico/a o un enfermero/a tendrán que seguirle a donde sea para aprender de él.

Nivel 1 - Puntuación total máxima a obtener: 3 puntos.

Lo que se pretende que el alumno desarrolle en este nivel son las relaciones de clase, herencia y demás que van asociadas al desarrollo de la práctica. Así, se pide realizar las siguientes tareas: ● Planteamiento del problema: actores participantes, relaciones entre actores, funcionalidad a cumplir por la práctica a desarrollar. ✓

- Establecimiento de diferentes clases a intervenir en la práctica, relaciones de dependencia entre clases, identificar diferentes jerarquías de clases, etc. ✓
- Elaboración de un documento escrito (memoria de la práctica) que contenga el primer punto y los correspondientes ficheros para BlueJ que implementen el segundo. ✓

2.2.Requerimientos funcionales

Nivel 2 - Puntuación total máxima a obtener: 7 puntos.

De este modo, el sistema deberá permitir lo siguiente:

- Dar de alta a los empleados del hospital, asignando cada uno a una unidad. ✓
- Dar de alta a los estudiantes, asignándoles clases en la unidad de formación con un miembro del personal médico/un miembro del personal de enfermería, o una cita (de mañana o tarde) con un miembro del personal médico para estar presentes durante las citas que tiene el especialista.(podría implementar la lógica pero sería exactamente lo mismo que he implementado con el personal del hospital y pacientes, así que siguiendo los consejos del tutor prioricé partes más importantes de la práctica las cuales no había implementado ,de todas formas la clase está creada con su tutor y un método vacío que dice que la clase está siguiendo a tutor.) ✓
- Dar de alta a los pacientes. ✓
- Dar una cita a un paciente para ver a un miembro del personal médico o un miembro del personal de enfermería (o para realizar una prueba médica) en una unidad en una fecha concreta, según la disponibilidad de cada especialista. La cita deberá quedar registrada en la agenda del miembro del personal médico/miembro del personal de enfermería correspondiente.(me falta darle cita para realizar alguna prueba médica, pero no lo consideré necesario debido a que ya que es agregar un nuevo método que hace lo mismo que hace lo mismo que revisionPaciente.). ✓
- Realizar el alta y la baja de pacientes ingresados en el hospital, asignándoles una habitación e indicando el procedimiento médico que tienen que recibir según su expediente médico. ✓
- Actualizar el expediente médico de una paciente una vez terminada una consulta, o tratamiento, por parte de un miembro del personal médico o un miembro del personal de enfermería. ✓
- Realizar búsquedas sencillas sobre los empleados y pacientes del sistema. ✓
- Realizar consultas y actualizaciones del calendario de cada miembro del personal médico/ miembro del personal de enfermería.(supongo que se refiere al turno de cada personal medico) ✓

Nivel 3 - Puntuación total máxima a obtener: 10 puntos.

Los alumnos que implementen este nivel de finalización de la práctica recibirán una puntuación máxima de 10 puntos. Sólo se podrá optar a este Nivel si se han implementado satisfactoriamente y en su totalidad los requerimientos especificados en el Nivel 2. Lo que se pretende es que el alumno desarrolle en este nivel la interfaz textual del sistema para las funciones identificadas en el nivel 2 y la generación de listados. De este modo, el sistema deberá permitir lo siguiente: ● Preparación y emisión de facturas para pacientes sin seguridad social o seguro médico privado. ✓

● Gestión de servicios de mantenimiento (alta y baja de recursos dañados que requieren mantenimiento o reparación y asignación de cada caso a un técnico). ✓

● Visualización de diferentes tipos de datos del sistema:

○ Los empleados que hay en el sistema, clasificados por su unidad de trabajo. ✓○ Listado de las agendas de cada especialista (los miembros del personal médico o los miembros del personal de enfermería).

○ Listado de pacientes que tiene cada miembro del personal médico en una fecha concreta. ✓(Tengo el metodo sacar lista de sanitarios no ocupados ,para que cada vez que alguien venga sin cita a la consulta o llegue tarde compruebe si en esa hora hay algún sanitario disponible para atenderlo ,decidí no implementar la lista ,puesto que no quería entrar en repeticiones de algo que ya había hecho varias veces como es mostrar empleados) ○ Listados de pacientes y sus expedientes médicos con diferentes criterios de búsqueda:

• Pacientes ingresados.

• Pacientes con citas en la agenda de consultas externas durante un periodo en concreto.

• Pacientes que tienen que ver un especialista (miembro del personal médico o miembro del personal de enfermería) en un periodo en concreto (día o semana).

○ Listado de ocupación de las habitaciones donde están ingresados los pacientes.

Esto último no me dio tiempo a hacerlo ,sin embargo hice un método general que da si un paciente está ingresado ,sus enfermedades ,gravedad de estas, fecha de tratamiento y concepción entre otras. Hacer un switch para decidir qué criterio de búsqueda quiero no sería difícil ,incluso tampoco las citas en periodos concretos ya que ya he trabajado con fechas y solucionado estos problemas antes ,una vez sacada la fecha podría ver si tiene que ver a un especialista en un periodo futuro o por el contrario la cita había expirado .Pero sentía que todo esto eran cosas con las que ya he trabajado en el personal del hospital y decidí dejarlo así por falta de tiempo.

2.3.Requerimientos técnicos

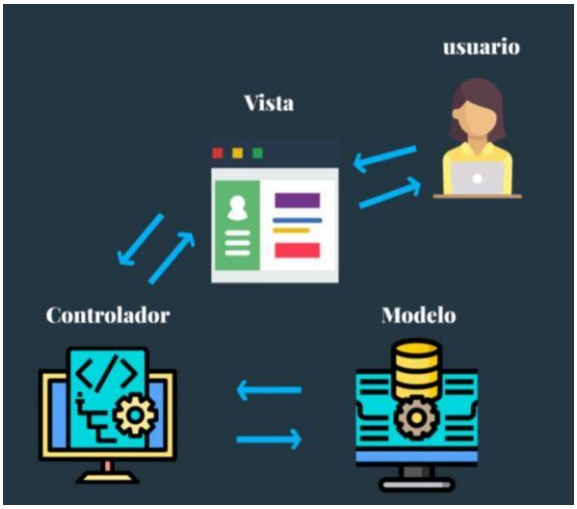
Aquí se nombrarán todas las limitaciones que tuve para realizar este trabajo. La primera y más obvia de todas es BlueJ, el cual parece tener problemas graves al manejar paquetes. Hasta el punto de no reconocerme paquetes o clases de otros paquetes ,a crashear cuando hay un error en código del que dependen varios paquetes ,etc .

El problema fue tal al introducir el código que ya tenía al sistema MVC que cada vez que habría el proyecto tenía 2 o 3 copias de clases sobrantes en sitios aparentemente aleatorios. A su vez otra limitación fue usar el propio MVC en sí y pasar bastante tiempo practicando en proyectos externos de la práctica para lograr implementar correctamente este modelo.

Gracias a esta última limitación , no me dio tiempo a implementar todo al pie de la letra de la practica y me tome libertad creativa para priorizar que era lo que debía enseñar que sabía hacer al tutor.

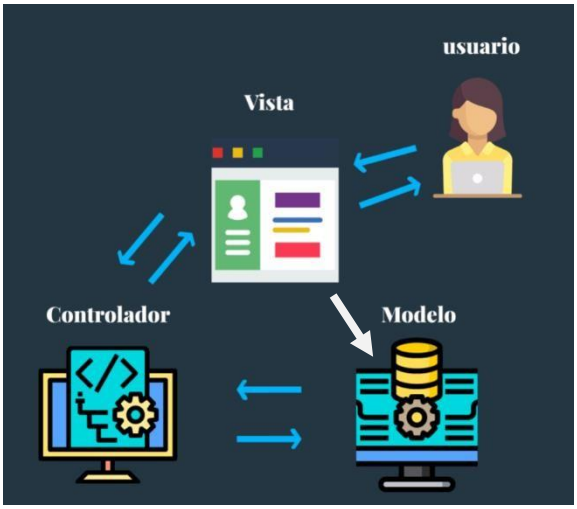
3.Diseño

Decidí seguir una arquitectura MVC clásica en un primero. Algo así:



Pero al complicárseme este, decidí implementar una variante de este hablada en clase ,en el cual la vista puede acceder al modelo en ciertas excepciones.

Estás excepciones únicamente se dan para manejar de forma más simple los datos que da el usuario, permitiendo que la vista sea capaz de reconocer al personal y a los pacientes para ser capaz de elegirlos de una lista y pasarlos correctamente al controlador para que esté llame a los métodos correspondientes dentro de administración. Esa foto ,con el agregado de mi flecha blanca, sintetiza el funcionamiento de mi MVC.



Hay un caso de uso en las que estás excepciones ocurren:

En este ejemplo necesito seleccionar a un paciente para mostrar su expediente .Es decir necesito seleccionar un dato concreto para modificar ciertos datos de este en el controlador.

```
Paciente pa=(Paciente) Controlador.getAdministracion().SacarListaEspecificaSanitarios("pacientes").get(index -1);//Cogemos el paciente
Controlador.getAdministracion().mostrarExpedienteMedico(pa);//Enseñamos el expediente medico del paciente
```

Este es la única excepción que se hace y creo que es correcto. Si se da el caso contrario ,dígame cual es el problema y lo solucionaré en cuanto pueda.

Persistencia de los datos:

Como se ha mencionado antes los datos en su mayoría se almacenan en administración ,la cual quizá debería de haber sido estática ,de esa forma hacer un almacén único de datos .Pero decidí no hacerla static ya que se puede dar el caso de que el programa pueda administrar varios hospitales con varias administraciones ,que se podría hacer con unas cuantas líneas de código. Por ende los datos quedaron guardados en listas .

En el caso de controlador sí que hice como variable static de administración ,que es la que se usa para sacar todos los datos y llevarlos a la interfaz ,decidí hacerla static para que cada controlador solo tuviera una media y una interfaz ,de forma que si quisieras escalar la aplicación ,necesitarías un controlador y una interfaz nueva por cada nueva instancia. Mientras que solo una instancia nueva de admin para cada controlador.

4.Conclusiones

Para mí es una locura el nivel que piden en está práctico, más con el MVC ,por eso intenté priorizar ciertas cosas antes que otras ya que no merece la pena dedicar tantísimo tiempo a algo que vale 2 puntos. También intenté hacer algunas cosas de distintas formas para que se vea que sé hacerlo de esa manera también.

En general no tiene sentido que está practica sea obligatoria si es consume tanto tiempo para hacerla ,y para un principiante en Java no tiene sentido que además de estudiarse bien toda la teoría tenga que hacer tanto en una práctica. Por otro lado el tema MVC se me complicó más de lo que esperaba e incluso ahora tengo dudas si está bien implementado del todo o no.

Por último, estaría bien que en la sesión obligatoria el profesor trajera un PowerPoint o algo del estilo para explicar mejor el modelo MVC ,quizá dando ejemplos de código y esas cosas.

Un saludo.