Práctica 2 - Sistemas Legados

Pedro Allué Tamargo (758267) Juan José Tambo Tambo (755742)
Jesús Villacampa Sagaste (755739)

20 de octubre de 2020

Índice

1. Esfuerzos invertidos

- Pedro Allué Tamargo:
- Juan José Tambo Tambo:
- Jesús Villacampa Sagaste:

2. Instalación del emulador

Para la realización de esta práctica se ha utilizado el Sistema Operativo *Ubuntu*. Para instalar el emulador *x3270*[1] se ejecutarán las órdenes:

```
sudo apt update
sudo apt -y install x3270
```

Estas instrucciones instalarán el emulador y las herramientas de *scrapping* (*s3270*).

Para conectar el *scrapper* con el *mainframe* se ejecutará la orden:

```
s3270 155.210.152.51:101
```

3. Descripción de la aplicación legada

La aplicación legada se corresponde con una lista de tareas. El usuario podrá añadir dos tipos distintos de tareas: tareas generales y tareas específicas.
Esta distinción implica que las tareas específicas disponen de un campo "nombre" del que no disponen las generales.
Otro punto a tener en cuenta de la aplicación legada es que las tareas guardadas durante una ejecución no son persistentes. Es decir, las tareas no se conservan de una ejecución a otra.

Poner los problemas de longitud y los espacios en las tareas.

[1] `http://x3270.bgp.nu/`

4. Implementación del Wrapper

Se pide realizar una aplicación con interfaz gráfica que encapsule el acceso a la aplicación legada. Se ha elegido *Java* como lenguaje para implementar este *wrapper*.

4.1. Modelo de datos

El modelo de datos se compone de 2 clases (Figura 1) que representan las distintas estructuras de datos que son las tareas de la aplicación legada.

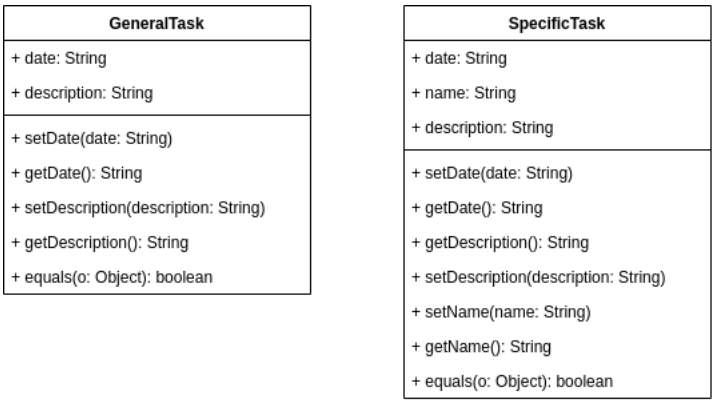

Figura 1: Diagrama de clases UML de Tareas

Las clases se componen de los mismos elementos que la aplicación legada. En el caso de *GeneralTask* se compone de 2 campos: fecha y descripción. En el caso de *SpecificTask* se compone de 3 campos: fecha, nombre y descripción.

4.2. Interfaz gráfica de usuario

La interfaz gráfica de usuario (GUI) se basa en las librerías gráficas *javax.swing* y *java.awt*.
Se ha estructurado la aplicación de tal manera que el usuario al iniciarla puede elegir entre acceder a las tareas generales o a las específicas (Figura 2).

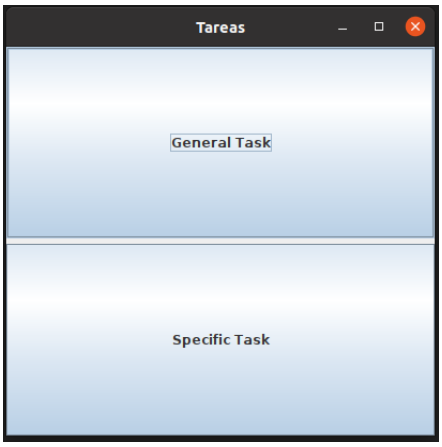

Figura 2: Pantalla principal de la aplicación

Tras acceder a la opción de *General Task* se puede observar una tabla que muestra las distintas tareas del tipo “General” que se han introducido en la aplicación (Figura 3).

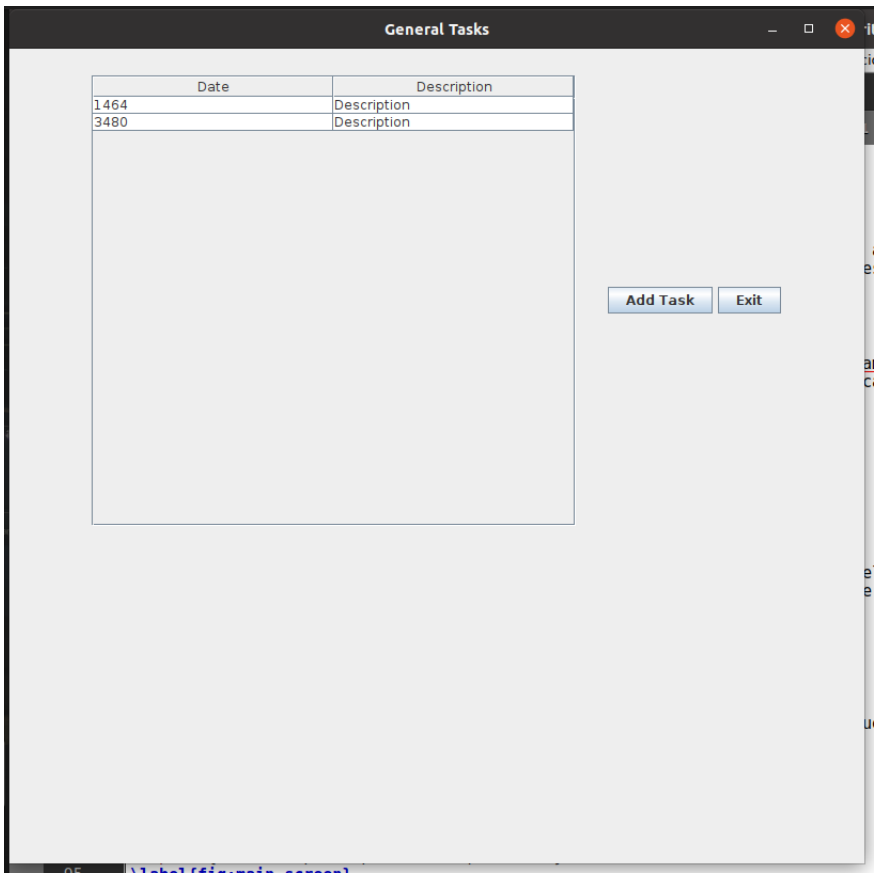

Figura 3: Pantalla que muestra las tareas generales de la aplicación

Si en esta pantalla se pulsa sobre el botón *"Add Task"* se abrirá una nueva ventana donde podremos introducir una nueva tarea a la aplicación (Figura 4).

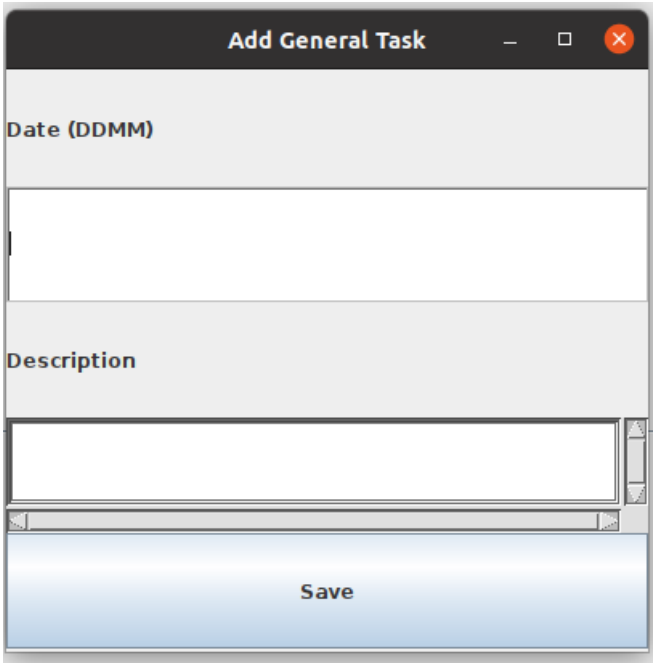

Figura 4: Pantalla que para añadir una tarea general a la aplicación

En el caso de las tareas "específicas" se mostrarían de manera análoga con la diferencia de que la tabla que las muestra es de 3 columnas (fecha, nombre y descripción).

4.3. Screen scrapping

Para poder realizar la comunicación con 3270, inicialmente se había considerado utilizar *WS3270* para realizar la comunicación, pero resultaba demasiado tedioso y generaba varios problemas. Es por ello que finalmente se ha utilizado *s3270*[2], una herramienta de *screen-scrapping*[3] para Linux.
Las funciones utilizadas para trabajar con el *scrapper* se han obtenido de la siguiente página.
Para el uso de esta herramienta en Java se han seguido las indicaciones de la memoria, ejecutando el comando de la siguiente manera:

```
final String commandLine = String.format("%s -model %s-%d %s:%d", s3270Path
        , type.getType(), mode.getMode(), hostname, port);

s3270 = Runtime.getRuntime().exec(commandLine);
out = new PrintWriter(new OutputStreamWriter(s3270.getOutputStream()
        , "ISO-8859-1"));
in = new BufferedReader(new InputStreamReader(s3270.getInputStream()
        , "ISO-8859-1"));
```

Se ha seguido un patrón de diseño *Singleton* para garantizar que solo se cree una única conexión con el *Mainframe*, junto a métodos encargados de añadir tareas (tanto específicas como generales) y mostrarlas. Para ello, se escribe en *scrapper* cada uno de los comandos necesarios para realizar las operaciones. Han aparecido una serie de problemas relacionadas con la inserción y lectura de texto, ya que, en ocasiones, el programa enviaba información al *scrapper* antes que este procesara la operación anterior, lo que provocaba sobreescrituras en el buffer. Para solventar este problema, se han añadido retrasos utilizando el comando "Thread.sleep(X)".

Hola Tambo (y Chusé) esta parte ya es tuya bro. Lo siento.

[2] http://x3270.bgp.nu/s3270-man.html
[3] https://x3270.miraheze.org/wiki/Screen_Scraping