



---

# PRÁCTICA 3

---

Sistemas legados



30 DE AGOSTO DE 2022

ÁLVARO FRAIDIAS Y RAFAEL RODRIGUEZ  
780336, 564786

## Contenido

Tabla de Ilustraciones.....	2
1. Objetivos del proyecto .....	3
2. Herramientas utilizadas .....	3
3. Reparto del trabajo y metodología .....	3
4. Problemas encontrados .....	4
5. Soluciones a los problemas planteados .....	4
6. Distribución del proyecto .....	5
Prototipo 0 .....	5
Prototipo 1 .....	6
Prototipo 2 .....	7
Prototipo 3 .....	8
Prototipo 4 .....	9
Prototipo 5 .....	9
Prototipo 6 .....	9
Prototipo 7 .....	10
Prototipo 8 .....	10
7. Conclusiones.....	11

## Tabla de Ilustraciones

Ilustración 1: Información sin parsear proveniente del mainframe Music.....	4
Ilustración 2: Método para eliminar la palabra "data:" del mainframe.....	5
Ilustración 3: Demo prototipo 0.....	5
Ilustración 4: Menú principal Music.....	6
Ilustración 5: Ventana para crear nueva agenda. ....	7
Ilustración 6: Ventana emergente para aceptar la creación de una nueva agenda. ....	7
Ilustración 7: Ventana para añadir una nueva tarea.....	8
Ilustración 8: Ventana para añadir una nueva tarea (Parte 1).....	8
Ilustración 9: Ventana para añadir una nueva tarea (Parte 2).....	8
Ilustración 10: Ventana para añadir una nueva tarea (Parte 3).....	8
Ilustración 11: Ventana para eliminar una tarea. ....	9
Ilustración 12: Ventana emergente para confirmar la eliminación de una tarea.....	9
Ilustración 13: Ventana que muestra la tarea que el usuario ha buscado mediante el panel superior. ....	9
Ilustración 14: Ventana que muestra todas las tareas de la agenda del usuario. ....	10
Ilustración 15: Ventana que muestra que se ha guardado correctamente la agenda.....	10
Ilustración 16: Ventana que muestra que se ha salido correctamente de la aplicación. ....	10



**Reconocimiento-No comercial-Sin  
derivados 4.0 Internacional  
(CC BY-NC-ND 4.0)**

## 1. Objetivos del proyecto

El objetivo de este proyecto es implementar un wrapper que permita el acceso a una aplicación legada ejecutada sobre un mainframe accedido desde un terminal TN3270.

Esta aplicación debe tener una interfaz gráfica para hacer más fácil la navegación para el usuario.

## 2. Herramientas utilizadas

Para implementar este proyecto se ha utilizado el lenguaje de programación Java, el entorno de programación seleccionado ha sido NetBeans y para la interfaz gráfica se ha elegido la conocida librería Java Swing.

## 3. Reparto del trabajo y metodología

Esta práctica ha sido realizada por Rafael Rodríguez (RR) y por Álvaro Fraidias (AF), dividiendo el proyecto en 8 prototipos. Los prototipos se desarrollan con mayor detenimiento más adelante (ver punto 6).

En un primer intento de desarrollar la práctica AF tendría como objetivo implementar la comunicación básica con el mainframe, es decir, el prototipo 0 mientras que RR iría desarrollando la interfaz gráfica. Tras la primera revisión se indicó un diseño erróneo por lo que hubo que reestructurar el proyecto.

Finalmente, AF modificó el diseño de tal forma que adaptara mejor a los principios de la programación orientada a objetos. A partir de ahí se han desarrollado el resto de prototipos encargándose AF de hacer los primeros 4 prototipos (0-3) bajo la supervisión de RR mientras que los últimos prototipos (4-8) los ha hecho Rafael Rodríguez bajo la supervisión de Álvaro Fraidias.

La metodología de programación utilizada ha sido la recomendada en otras ocasiones, por medio de videoconferencia, y, compartiendo pantalla, se ha programado de manera conjunta asumiendo uno el rol de programador y el otro el de observador, intercambiando los papeles posteriormente.

La elaboración del primer borrador del presente informe ha sido planteada por AF y, posteriormente, revisada por RR.

## 4. Problemas encontrados

- 1º Conexión al mainframe Music.
- 2º Implementación de la interfaz gráfica.
- 3º Parseo la información que nos llegaba del mainframe.
- 4º Bloqueo del mainframe en el uso de comandos
- 5º Implementar interfaces para implementar el modelo Music.
- 6º Eliminar una tarea.

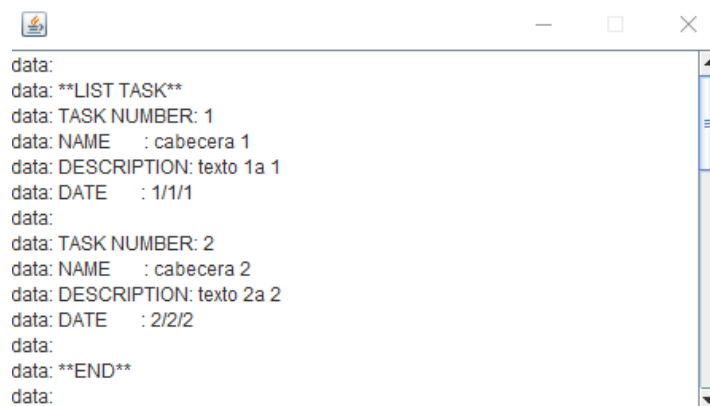
## 5. Soluciones a los problemas planteados

- 1º Conexión al mainframe Music.

**2º Implementación de la interfaz gráfica.** Al principio se invirtió mucho tiempo en intentar desarrollar una interfaz gráfica, que, aunque se pedía sencilla, se trató de que fuera funcional y amigable. No obstante, se recomendó que realizara con el asistente de NetBeans pero nos pareció más complejo y, por lo tanto, se tuvo que retomar la idea de plantearla con swing. En este segundo intento, se partió desde cero, con los conocimientos adquiridos en otras asignaturas en las que ya se ha programado previamente con swing. En este caso no se han priorizado los detalles de funcionamiento y estéticos como se podrá observar en la ejecución, pero ha permitido que su desarrollo sea algo más ágil.

Una vez ya desarrollada la interfaz gráfica, al unirla con el modelo aparecían bastantes problemas porque se quedaba la aplicación colgada por lo que se hizo uso del debugger para depurar el código y así lograr un funcionamiento correcto.

**3º Parsear la información.** La información recibida desde el mainframe llegaba de la siguiente forma:



```
data:
data: **LIST TASK**
data: TASK NUMBER: 1
data: NAME : cabecera 1
data: DESCRIPTION: texto 1a 1
data: DATE : 1/1/1
data:
data: TASK NUMBER: 2
data: NAME : cabecera 2
data: DESCRIPTION: texto 2a 2
data: DATE : 2/2/2
data:
data: **END**
data:
```

*Ilustración 1: Información sin parsear proveniente del mainframe Music.*

Para solucionar este problema se ha creado un método que elimina la palabra pasada como argumento:

```
public String parsearInformacionMainframe(String oracion,String palabra) {  
    if(oracion.contains(palabra))  
        return oracion.replaceAll(palabra, "");  
    return oracion;  
}
```

*Ilustración 2: Método para eliminar la palabra "data:" del mainframe.*

**4º Bloqueo del mainframe en el uso de comandos.** A la hora de navegar entre los distintos menús del mainframe ya que había que ser muy meticuloso con las instrucciones que mandábamos al mainframe. Al igual que en el punto anterior, suponía reiniciar el mainframe para poder continuar con la práctica. Se ha resuelto entendiendo el funcionamiento y logrando una mayor fluidez en cuanto al manejo.

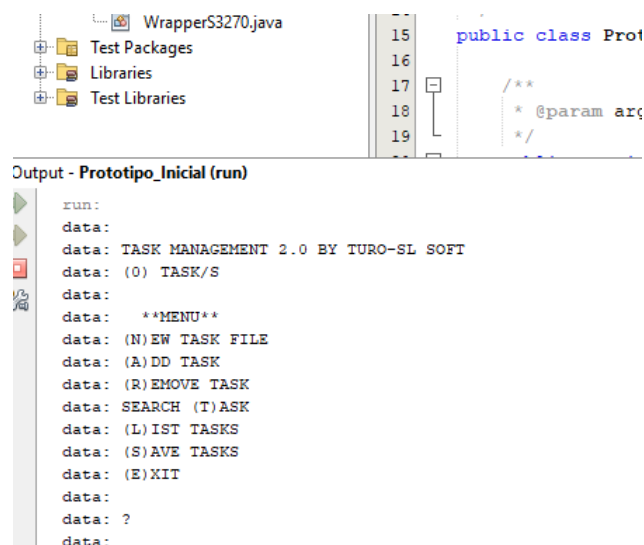
**5º Implementar interfaces para implementar el modelo Music.** Desde el principio se trabajó con un diseño erróneo entendiendo que no era el objetivo de la práctica, sino el desarrollo de la comunicación. Eso fue un error y se debía haber trabajado en un diseño más elaborado y correcto desde el punto de vista del paradigma de orientación a objetos. Una vez entendida la orientación que debía seguir el código, se han desarrollado una serie de interfaces y clases acordes a lo que requería.

**6º Eliminar una tarea.** A la hora de eliminar una tarea, el mainframe solicita confirmación de dicha eliminación por lo que hubo que crear una nueva ventana y volver a conectar con el mainframe para aceptar o rechazar la eliminación de dicha tarea.

## 6. Distribución del proyecto

### Prototipo 0 (23/6/22 – 30/6/22)

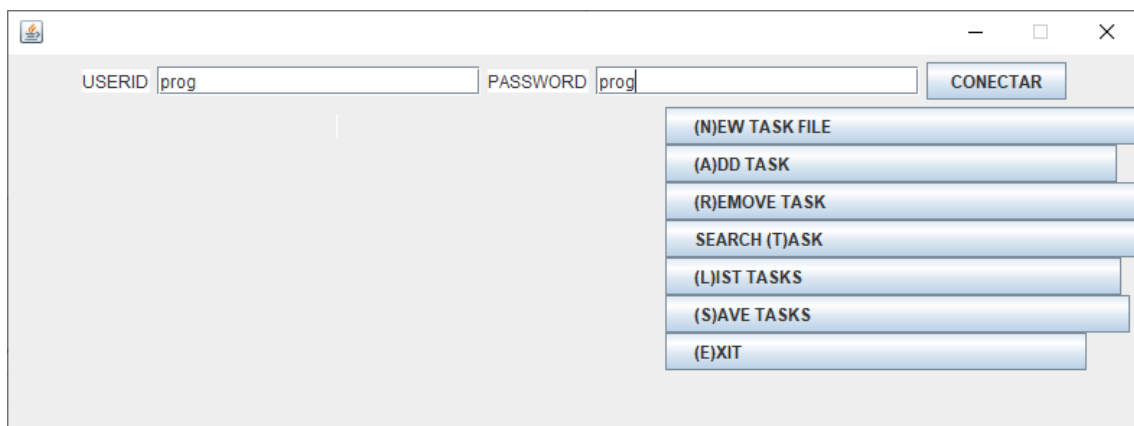
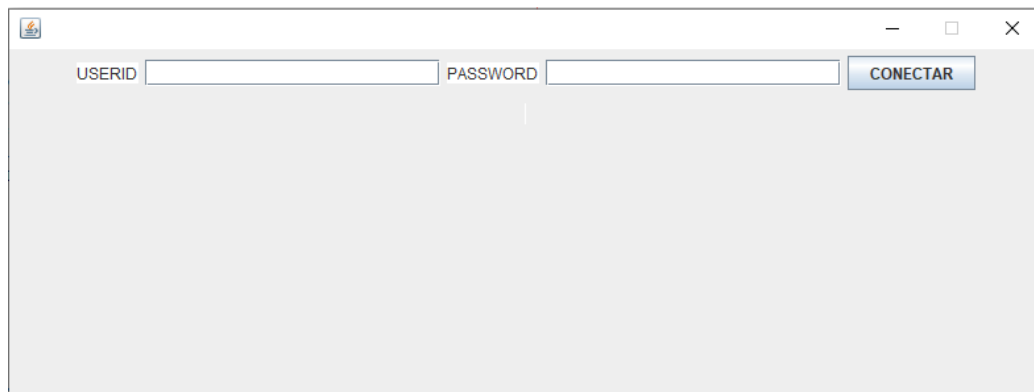
Se ha construido el modelo y una pequeña demo para comprobar que funciona correctamente.



*Ilustración 3: Demo prototipo 0.*

### Prototipo 1 (4/7/22 - 10/7/22)

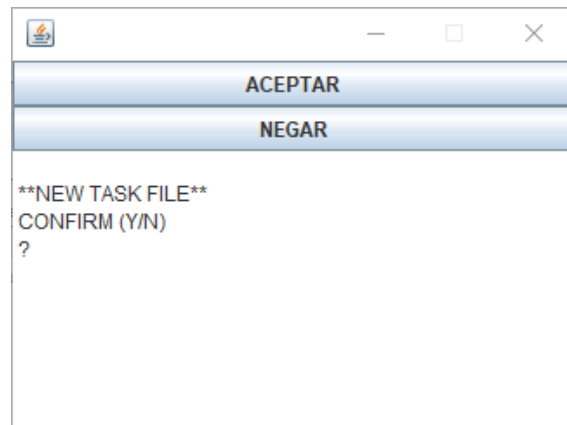
Desde este prototipo hasta el final se van a ir implementando gráficamente las diferentes funcionalidades de la aplicación. En este prototipo se crea la pantalla para iniciar sesión y la pantalla principal.



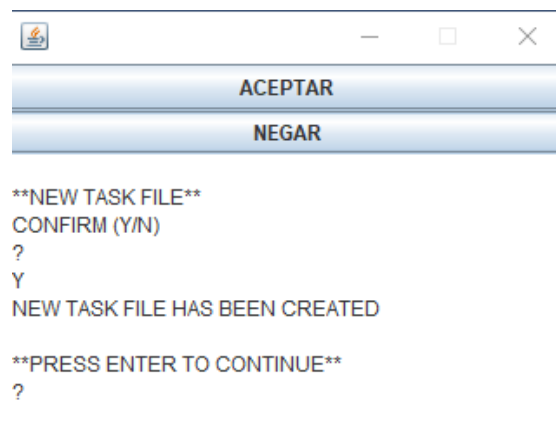
*Ilustración 4: Menú principal Music.*

## Prototipo 2 (12/7/22 - 17/7/22)

En este prototipo se ha añadido la funcionalidad de crear una nueva agenda.



*Ilustración 5: Ventana para crear nueva agenda.*



*Ilustración 6: Ventana emergente para aceptar la creación de una nueva agenda.*



### Prototipo 3 (19/7/22 - 24/7/22)

Se incluye la funcionalidad de añadir una nueva tarea. Antiguamente en el mainframe iba pidiendo los datos de uno en uno, pero la aplicación permite introducirlos todos a la vez para que la interfaz gráfica solo se conecte una vez al mainframe y añada la tarea directamente.



A screenshot of a web application window titled 'Añadir una nueva tarea'. The window contains a grid of input fields for adding a new task. The fields are arranged in two rows and four columns. The first row contains fields for 'Añadir un numero a la tarea', 'Añadir una cabecera a la tarea', 'Añadir un texto a la tarea', and an empty field. The second row contains fields for 'Añadir un día a la tarea', 'Añadir un mes a la tarea', 'Añadir un año a la tarea', and an empty field. At the bottom right of the grid is a button labeled 'ENVIAR'.

*Ilustración 7: Ventana para añadir una nueva tarea.*

Se divide la imagen a continuación para que se pueda ver con mayor detalle.



A close-up screenshot of the first part of the 'Añadir una nueva tarea' window. It shows a small icon in the top left corner, followed by the labels 'Añadir un numero a la tarea' and 'Añadir un día a la tarea'. To the right of these labels are two stacked input fields.

*Ilustración 8: Ventana para añadir una nueva tarea (Parte 1).*



A close-up screenshot of the second part of the 'Añadir una nueva tarea' window. It shows the labels 'Añadir una cabecera a la tarea' and 'Añadir un mes a la tarea'. To the right of these labels are two stacked input fields. At the bottom right of the section is a button labeled 'ENVIAR'.

*Ilustración 9: Ventana para añadir una nueva tarea (Parte 2).*



A close-up screenshot of the third part of the 'Añadir una nueva tarea' window. It shows the labels 'Añadir un texto a la tarea' and 'Añadir un año a la tarea'. To the right of these labels are two stacked input fields. The window has standard window controls (minimize, maximize, close) in the top right corner.

*Ilustración 10: Ventana para añadir una nueva tarea (Parte 3).*

#### Prototipo 4 (26/7/22 - 31/7/22)

Incorpora la opción de eliminar una tarea. El usuario puede escribir el número de la tarea que desea eliminar y el mainframe se ocupará del resto.



ESCRIBE EL NUMERO DE LA TAREA A ELIMINAR  **ENVIAR**

Ilustración 11: Ventana para eliminar una tarea.

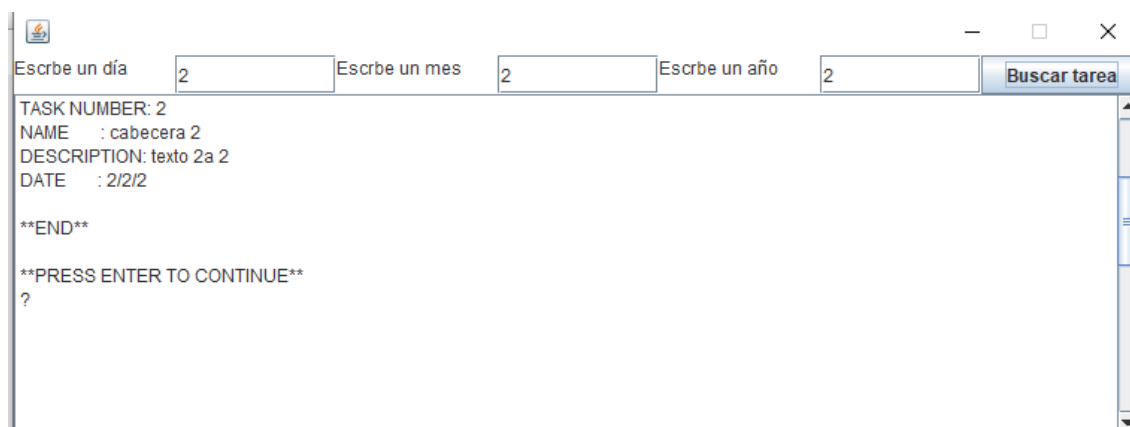


¿SEGURO QUE QUIERES ELIMINAR ESTA TAREA? **Eliminar**  
**Cancelar**

Ilustración 12: Ventana emergente para confirmar la eliminación de una tarea.

#### Prototipo 5 (2/8/22 - 7/8/22)

Se agrega la opción de buscar una tarea. Mostrará una nueva ventana parecida a la del prototipo anterior. En ella el usuario escribirá la tarea que desea buscar y el mainframe la buscará. Una vez encontrada se mostrará en la propia ventana mediante un Scroll.



Esrb e un día  Esrb e un mes  Esrb e un año  **Buscar tarea**

TASK NUMBER: 2  
NAME : cabecera 2  
DESCRIPTION: texto 2a 2  
DATE : 2/2/2  
  
\*\*END\*\*  
  
\*\*PRESS ENTER TO CONTINUE\*\*  
?

Ilustración 13: Ventana que muestra la tarea que el usuario ha buscado mediante el panel superior.

#### Prototipo 6 (9/8/22 - 14/8/22)

Permite listar todas las tareas que hay en una agenda. Se ha implementado un Scroll donde se mostrarán todas las tareas que haya en la agenda del mainframe como se muestra a continuación.

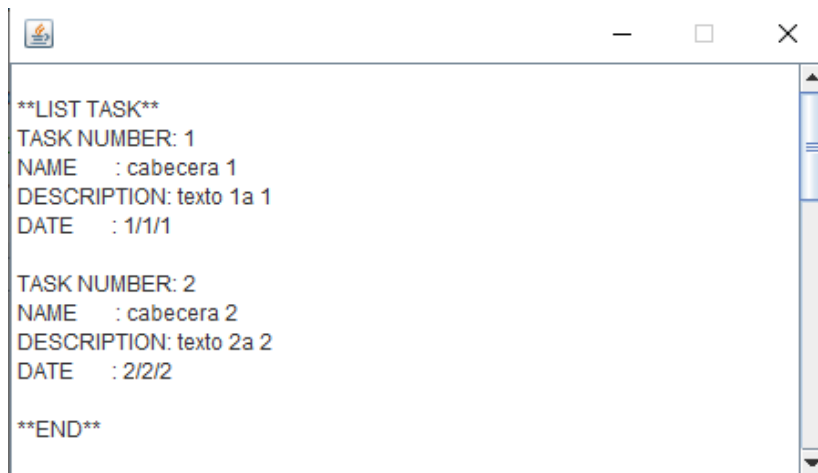


Ilustración 14: Ventana que muestra todas las tareas de la agenda del usuario.

### Prototipo 7 (16/8/22 - 19/8/22)

Permite el guardado de las tareas para poder ver las tareas al conectarnos por segunda vez.

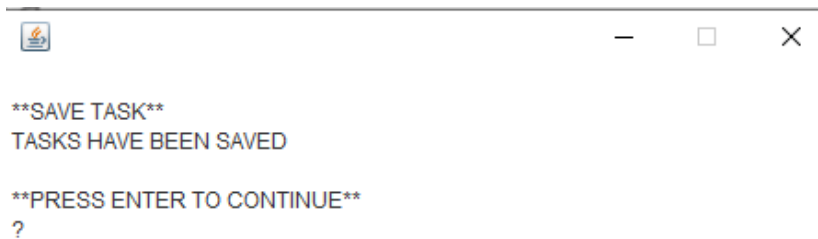


Ilustración 15: Ventana que muestra que se ha guardado correctamente la agenda.

### Prototipo 8 (20/8/22 - 22/8/22)

Desconexión del mainframe de forma segura.

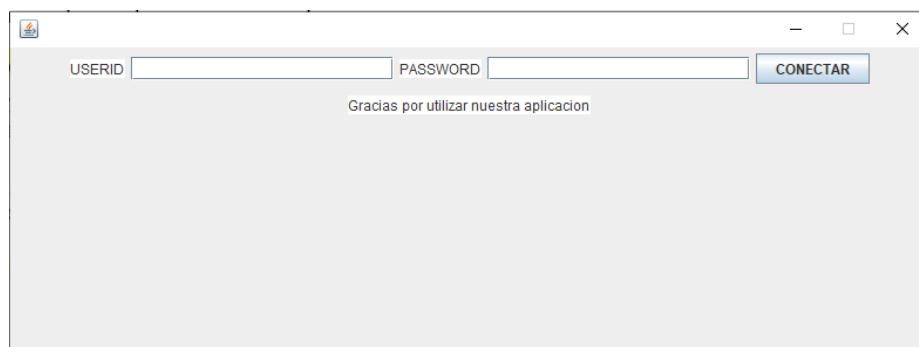


Ilustración 16: Ventana que muestra que se ha salido correctamente de la aplicación.

## 7. Conclusiones

Con el paso de los años, cada vez hay más sistemas legados. El coste de elaborar otros nuevos desde cero es muy alto. En muchas ocasiones su funcionalidad se encuentra todavía vigente, pero se requiere añadir algunas funciones adicionales o renovar su interfaz gráfica, por ejemplo. Es por ello que la implementación de los wrappers puede encontrarse como una solución satisfactoria para el ámbito económico y el tecnológico. De esta manera se garantiza que se pueda seguir trabajando con el sistema legado de una forma fiable y segura como se ha hecho hasta el momento, mientras que permite modificaciones funcionales adicionales si se desea.

Aunque pueda parecer una tarea sencilla, no lo es. Bien es cierto que la programación puede ser más o menos dificultosa, pero la clave, como se ha podido observar en el desarrollo de esta práctica, está en el diseño. Es importante tener muy bien definidas dos cosas. Por un lado, el funcionamiento del sistema legado. Es imprescindible conocer los pasos que sigue su funcionamiento, los tiempos en la comunicación y la respuesta que se obtiene en cada momento para poder gestionarla y tratarla desde la aplicación posteriormente. Teniendo ese punto claro, por el otro lado, es imprescindible plantear un diseño correcto, que, aunque se tenga que ver modificado en algún momento, sea lo más levemente posible.

Muchas veces estas tareas pasan desapercibidas y desde el principio no se ha tenido en cuenta este segundo punto, pensando que no sería tan importante de cara a la realización de la práctica, pero ha quedado claro que es imprescindible para que la aplicación, ya no solo funcione correctamente, sino que también pueda reutilizarse en un posible escenario real en el que se tuviera que añadir funciones adicionales que hubiera tener que implementar.