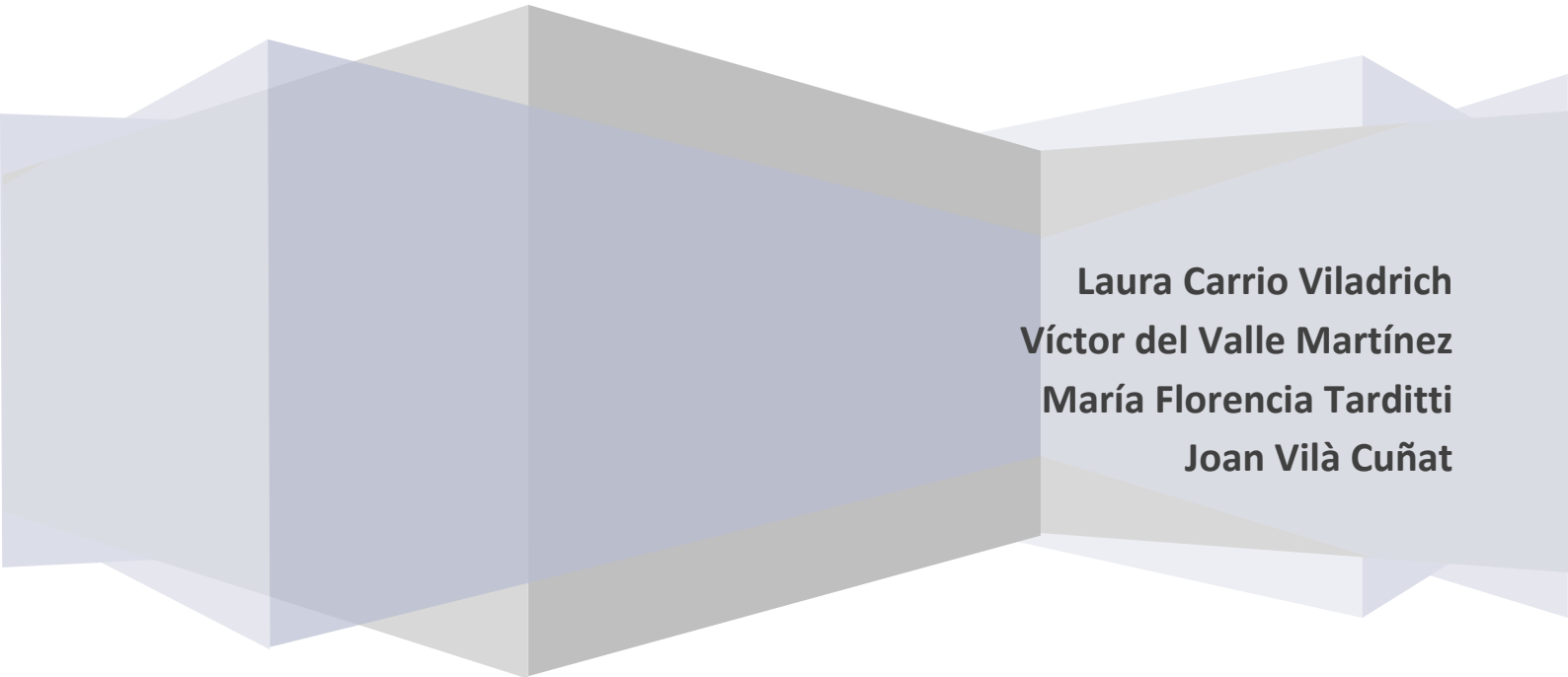


Proyectos de Programación

# Smiley: Documentación

## Cluster 45

PROFESSOR HORACIO RODRÍGUEZ HONTORIA



**Laura Carrio Viladrich  
Víctor del Valle Martínez  
María Florencia Tarditti  
Joan Vilà Cuñat**

## ÍNDIX

1. Manual de usuario .....	4
Introducción .....	4
1.1 Gestionar Mapas .....	6
1.1.1 Mapa Nuevo .....	7
1.1.2 Modificar Mapa .....	12
1.1.3 Guardar Mapa .....	13
1.1.4 Ver Mapa .....	14
1.2 Gestionar Reunión.....	15
1.3 Simular Reunión .....	17
1.4 Ver Resultados .....	20
1.4.1 Guardar Resultados.....	20
1.4.2 Resultados Guardados.....	21
1.5 Salir .....	21
Modificación de los juegos de prueba.....	21
2. Diagramas de clases.....	22
2.1 Dominio y controladores del dominio .....	22
2.2 Vistas y controladores de presentación .....	22
2.3 Gestores de disco .....	22
3. Descripción de las estructuras y algoritmos utilizados para implementar las funcionalidades principales.....	23
3.1 Funcionalidades principales .....	23
3.2 Gestión de mapas.....	23
3.2.1 Tramos.....	23
3.3 Definir la reunión .....	24
3.4 Hacer reunión (Algoritmo de Maximum Flow) .....	24
3.4.1 Obtener las rutas por las que pasan los agentes .....	25
3.5 Resultados .....	25

4.	Relación de las clases implementadas por cada grupo del cluster.....	26
5.	Relación de las clases implementadas por cada miembro del grupo.....	27
5.1	Presentación.....	27
5.2	Dominio .....	28
5.3	Persistencia .....	28
6.	Listado de funcionalidades implementadas y relación de diferencias entre entregas .....	29
6.1	Diagrama de clases.....	29
6.2	Casos de uso .....	29

## 1. MANUAL DE USUARIO

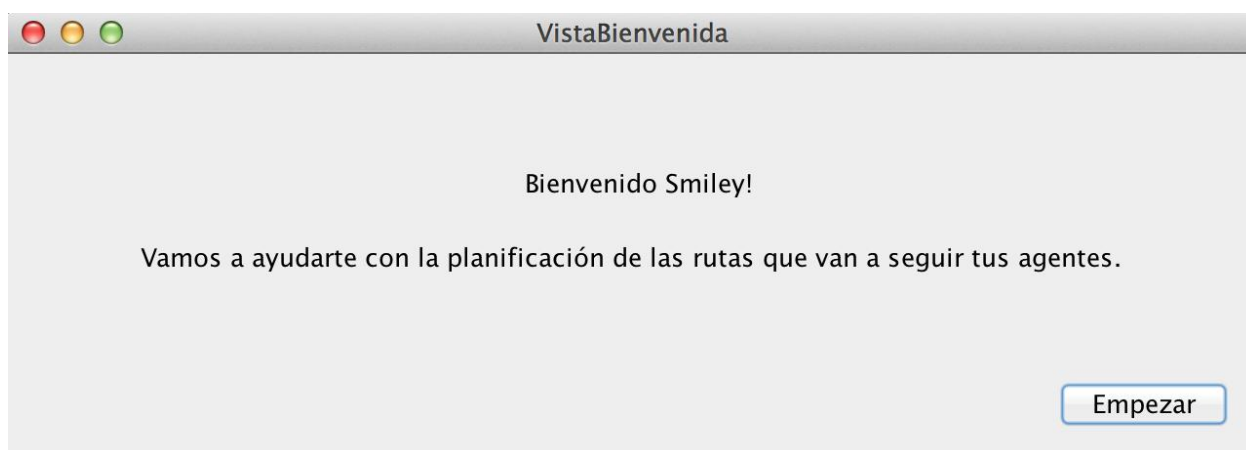
### INTRODUCCIÓN

Este manual es un documento de comunicación técnica que busca brindar asistencia a los sujetos que usan el sistema que da soporte a la gestión de reuniones organizadas por el “Circus”, el servicio secreto británico.

Proporciona las instrucciones necesarias para que el jefe de del MI6 pueda organizar reuniones dado un número de agentes y un mapa concreto.

Para ejecutar el programa nos situamos encima del fichero ejecutable Smiley.jar y hacemos doble clic para empezar la ejecución.

Una vez iniciada la ejecución del programa, la primera ventana que se muestra nos da la bienvenida al sistema.

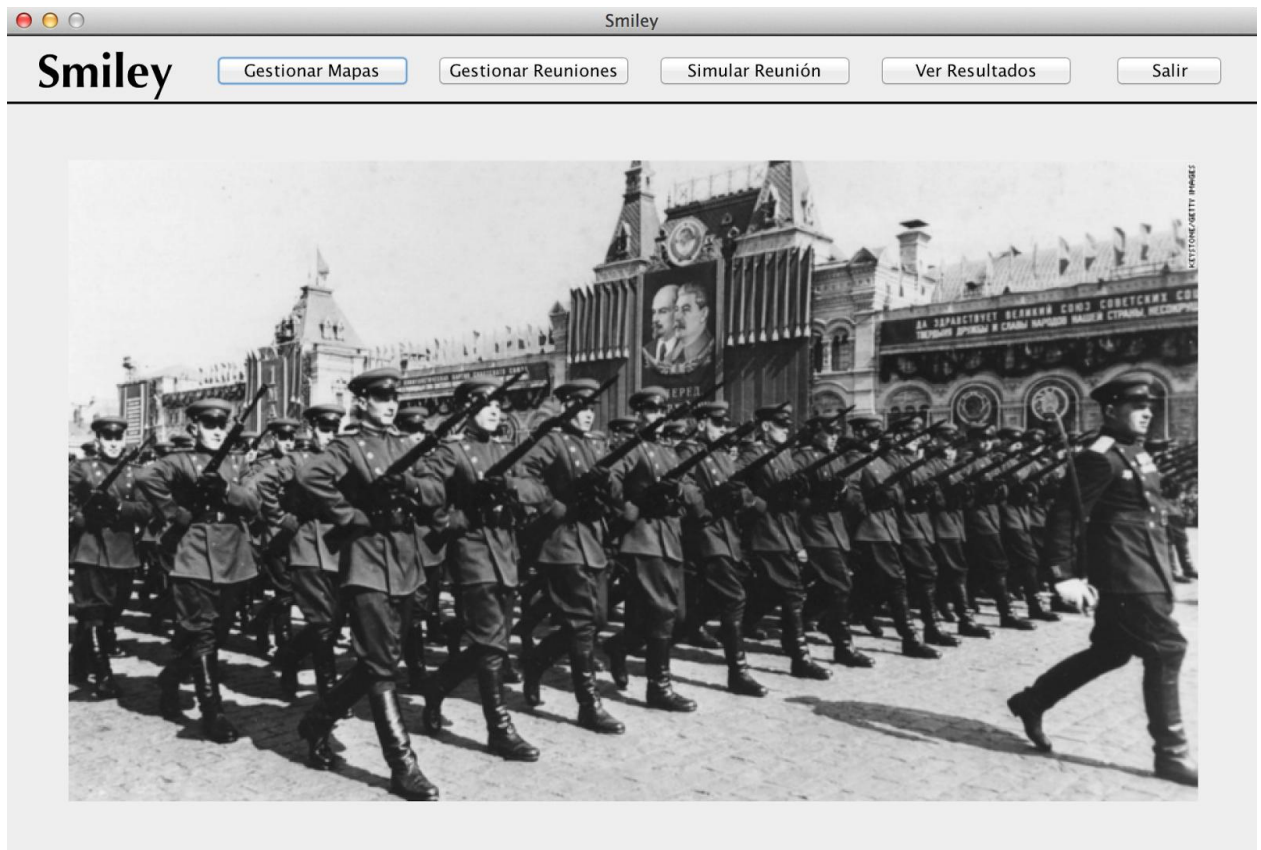


Para poder continuar debemos de hacer clic en el botón “Empezar”.

Una vez dentro del gestor de reuniones, veremos la ventana inicial del sistema.

En la parte superior de la ventana aparece el menú principal del sistema. Este menú horizontal está formado por cinco botones que representan las diferentes opciones que el sistema puede desarrollar. Estas opciones son:

- Gestionar Mapas
- Gestionar Reuniones
- Simular Reunión
- Ver Resultados
- Salir



La barra superior que contiene el menú principal siempre se mostrará en la aplicación independientemente de la ventana en que nos encontremos a lo largo de la ejecución. Es un menú fijo.

Poco a poco iremos explicando más detalladamente cada una de las opciones que forman este menú.

Hace falta decir que se ha intentado minimizar al máximo o completamente los posibles errores del usuario mediante gestión de errores como se verá a lo largo del manual.

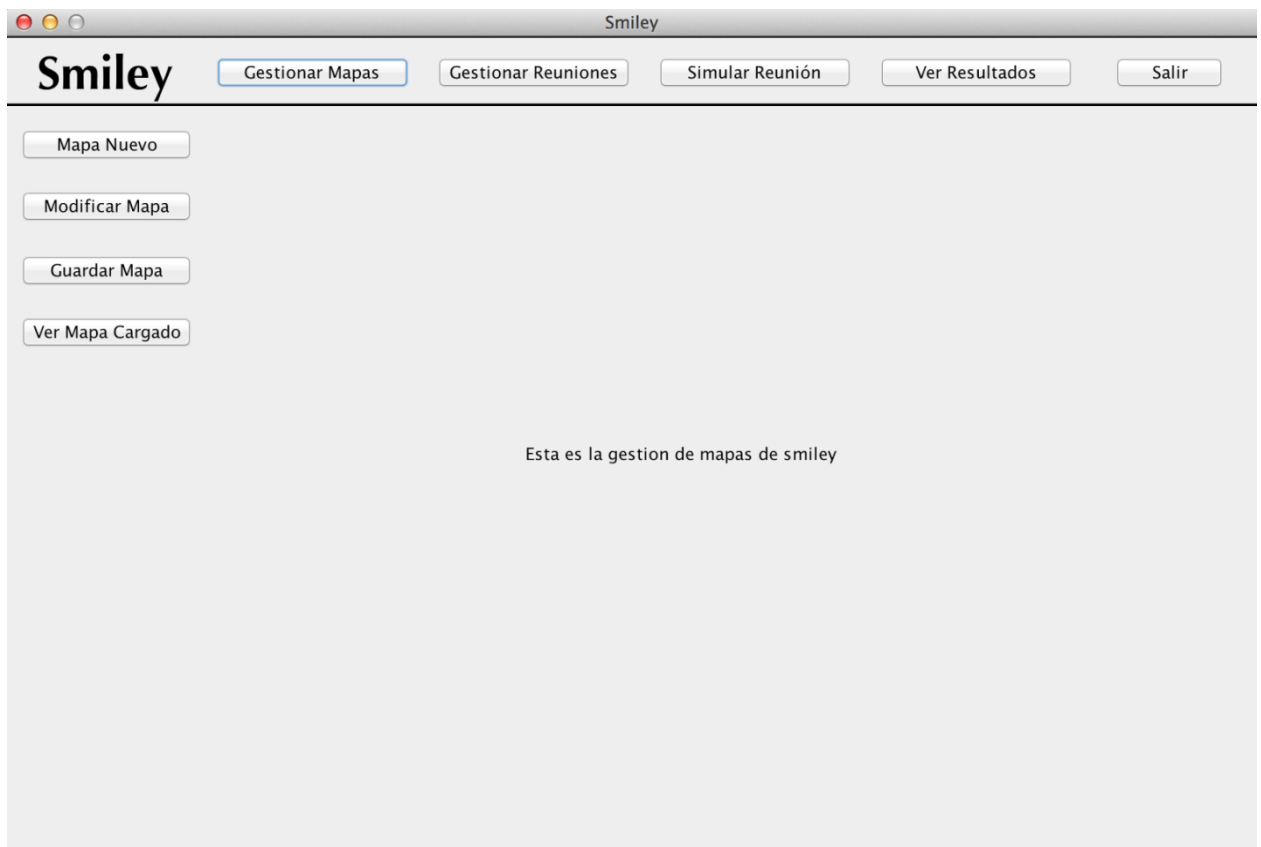
## 1.1 GESTIONAR MAPAS

La primera opción de nuestro menú es “Gestionar Mapas”. En esta opción se tratan los mapas sobre los que se van a gestionar y simular las reuniones. En definitiva, por dónde se van a desplazar nuestros espías.

Para poder acceder a la gestión de mapas, debemos de hacer clic en el primer botón del menú “Gestionar Mapas”.

Una vez seleccionada la opción, aparece un submenú vertical formado por todas las funcionalidades de la opción “Gestionar Mapas”. Las funcionalidades son:

- Mapa Nuevo
- Modificar Mapa
- Guardar Mapa
- Ver Mapa

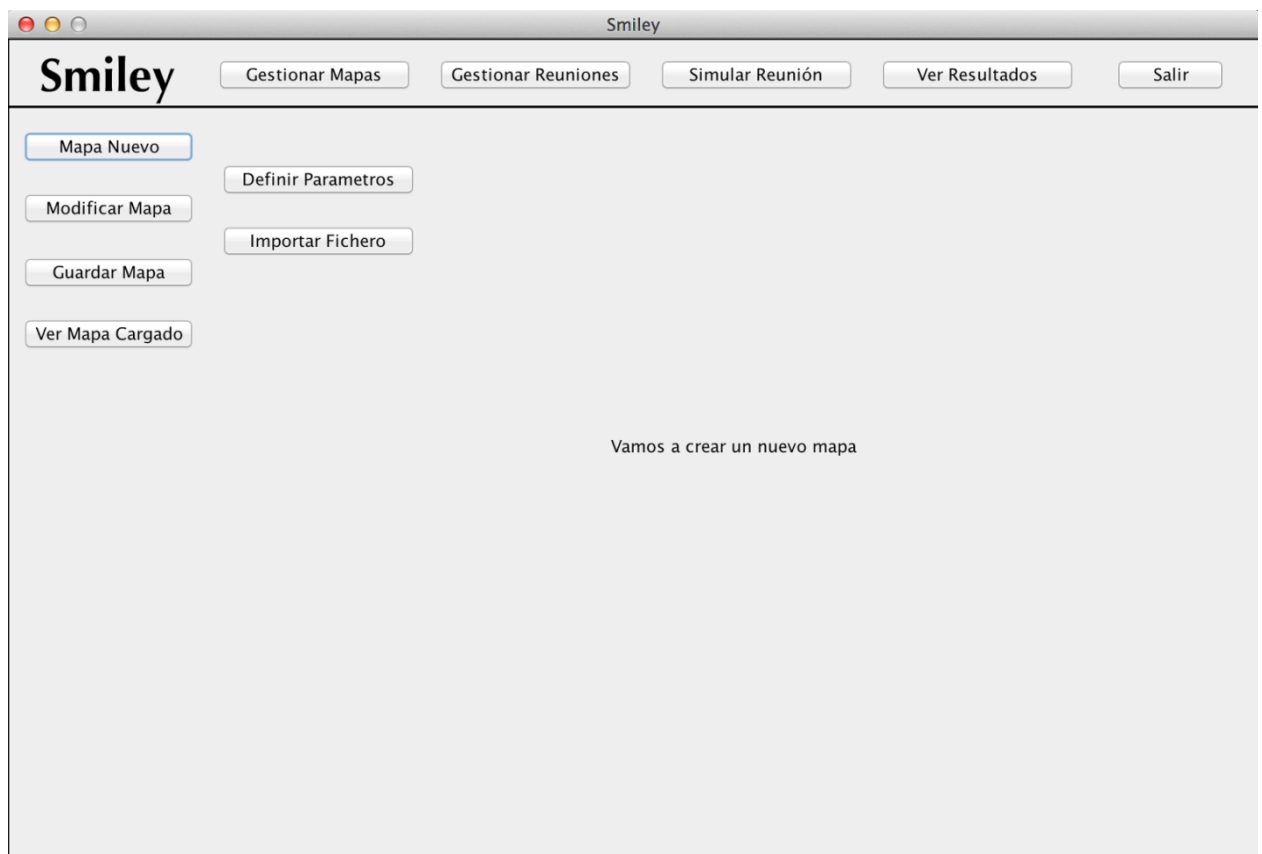


A continuación vamos a explicar cada una de las funcionalidades de “Gestionar Mapas”.

### 1.1.1 MAPA NUEVO

La primera funcionalidad es “Mapa Nuevo”. Su principal finalidad es cargar o crear el mapa por el que se van desplazar nuestros agentes en el sistema. Esta funcionalidad nos muestra a su vez dos opciones más:

- Definir Parámetros
- Importar Fichero



Estas dos opciones nos permiten introducir el mapa de dos formas

La primera opción, “Definir Parámetros”, nos permite crear un mapa nuevo desde la interface del programa. Y la segunda, “Importar fichero”, nos da la posibilidad de cargar un mapa ya creado que se encuentra en nuestros directorios. Si ya teníamos un mapa cargado en el sistema, cuando intentamos cargar otro, automáticamente se sobrescribe el viejo por el nuevo.

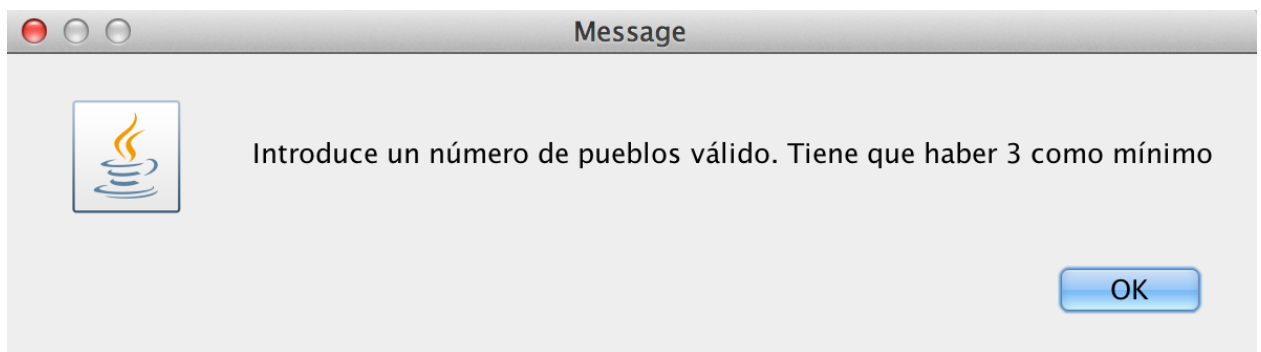
### 1.1.1.1 DEFINIR PARÁMETROS

Si lo que deseamos es crear un mapa nuevo introduciendo uno a uno cada una de los pueblos y cada uno de los tramos, deberemos hacer click sobre la opción “Definir Parámetros”. El sistema nos pedirá introducir diferentes datos necesarios para crear el mapa. Estos datos son:

- Nombre el Mapa
- Número de pueblos del Mapa
- Nombre de cada una de las pueblos
- Número de tramos
- Pueblo origen y destino para cada uno de los tramos

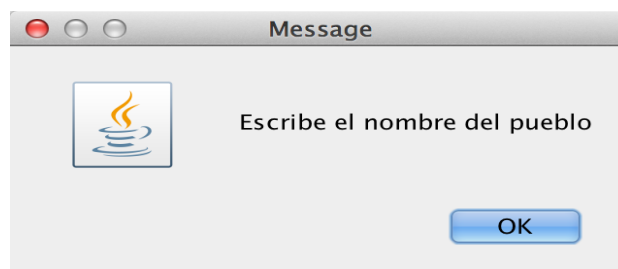
Inicialmente definiremos el nombre del mapa y el número de pueblos que lo componen.

**Atención:** El número mínimo de pueblos que pueden formar un mapa son tres pueblos. En el caso de que el número introducido por el usuario sea menor que 3, el sistema le mostrará un mensaje advirtiéndole de su error.



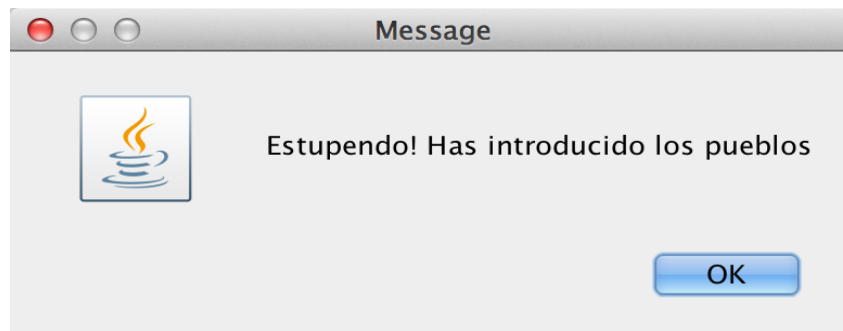
Seguidamente, el sistema asigna un identificador para cada pueblo y además, le pide al usuario que le asigne un nombre.

**Atención:** Si el usuario no le asigna ningún nombre al pueblo, el sistema avisa del error.



Una vez introducida toda la información referente a los pueblos del mapa nos aparece un mensaje que certifica que la creación de pueblos en el sistema ha sido correcta.

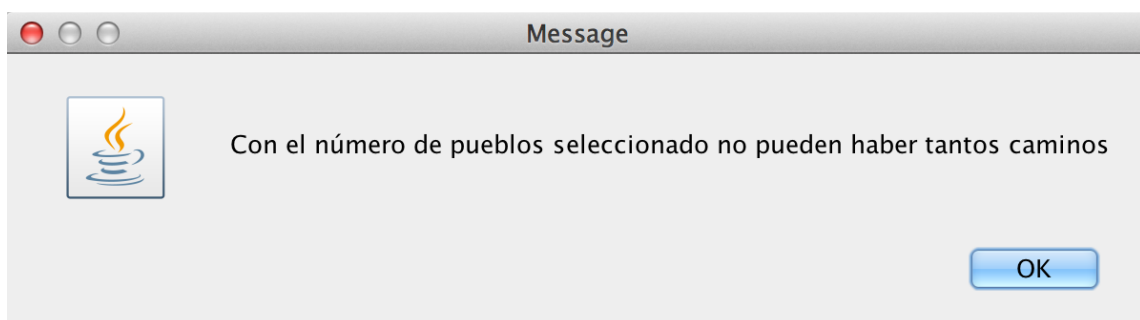




Después de crear los pueblos, llega el momento de definir los tramos que contiene el mapa.

En un mapa, cómo máximo cada pueblo puede tener tramo a todos los demás y obteniendo así un mapa completo. Además, un mapa cómo mínimo puede tener un tramo.

**Atención:** Si el número de tramos es mayor al límite superior o menor al límite inferior de tramos, nos advierte del error.



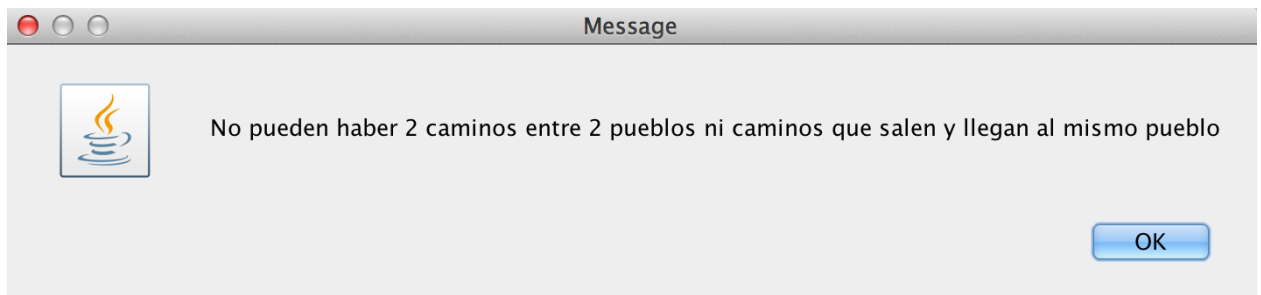
Una vez introducido un número correcto de tramos que introduciremos, el sistema asigna un identificador numérico para cada tramo. Además, el usuario debe de indicar para cada tramo su origen y final. Para poder definirlos, para cada tramo, nos aparecen dos listas con los nombres de todos los pueblos en cada tabla respectivamente.

En la primera tabla se define el pueblo origen y en la segunda el pueblo destino. Además, para cada tramo también hay que definir el tiempo (en minutos), la distancia (en km) y el coste (en €).

Si el tramo representa una carretera lo representaremos poniendo el coste del billete a 0.

**Atención:** Si el usuario asigna a un tramo el mismo pueblo origen que destino, el sistema informa de este hecho con un mensaje.

También lanza un mensaje de advertencia si el usuario intenta crear más de un tramo con el mismo origen y destino.



Una vez definidos el origen y destino de todos los tramos ya habremos terminado la creación del mapa y nos aparecerá un mensaje para confirmarlo.



Si clicamos en el botón “Aceptar” se muestra el mapa que hemos creado cargado en el sistema.

---

#### 1.1.1.2 IMPORTAR MAPA

Si lo que deseamos es importar al sistema un mapa que ya tenemos creado y guardado en forma de ficheros debemos de hacer clic en la opción “Importar Mapa”.

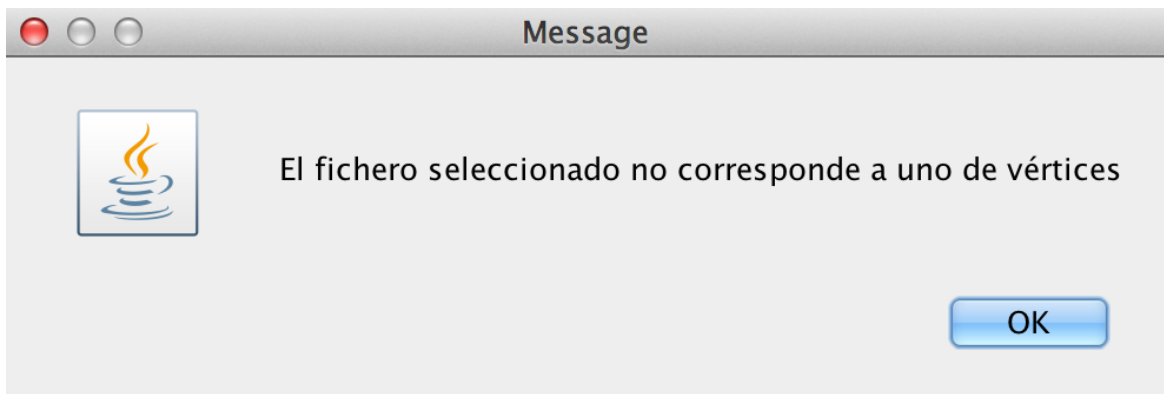
Para poder importar un mapa debemos de tener dos archivos. El primero de estos archivos debe de contener los pueblos y el segundo los tramos del mapa.

Además, para lograr una importación correcta los dos archivos estos deben de seguir la siguiente nomenclatura:

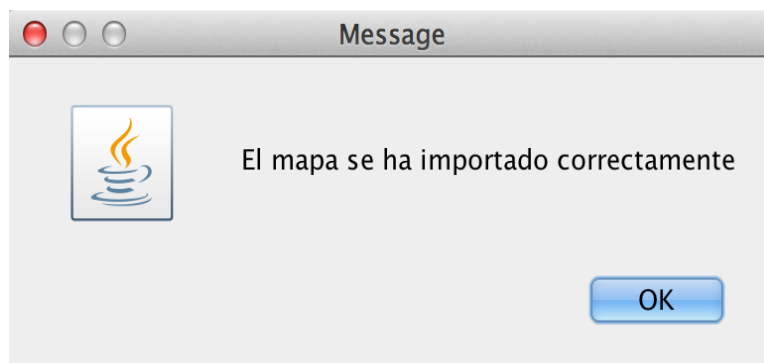
- Archivo de pueblos: *nombre\_del\_mapaV*
- Archivo de tramos: *nombre\_del\_mapaA*

El nombre de los archivos debe de ser el mismo que el nombre del Mapa y, para terminar, el nombre del archivo que contiene los pueblos debe de terminar en V y el archivo que contiene los tramos debe de terminar con A.

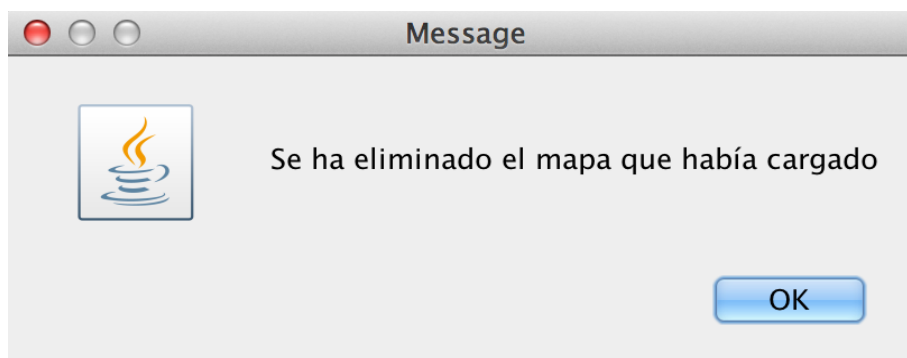
**Atención:** Si a la hora de importar los archivos no se respeta esta nomenclatura aparecerán mensajes comunicando que el archivo seleccionado no es el correcto.



Una vez hemos importado los dos archivos hacemos clic en la opción “! Importar Mapa!”. Si el mapa se ha importado correctamente aparecerá un mensaje informativo que lo confirmará y seguidamente podremos ver el mapa importado en el sistema.



**Atención:** Si en el sistema ya había un mapa cargado anteriormente, aparece un mensaje que informa al usuario de que el mapa cargado anteriormente ha sido eliminado.



### 1.1.2 MODIFICAR MAPA

La opción “Modificar Mapa” permite modificar los elementos del mapa que hay cargado en el sistema. Esta funcionalidad tiene como fin modificar los atributos de pueblos y tramos.

Para modificarlos debemos de identificar el elemento con deseo de modificar con el atributo id.

En el caso de los pueblos podemos cambiar su nombre y si se trata de un tramo, podemos cambiar su nombre, el tiempo, la distancia y su coste.

Una vez realizada la modificación correctamente, se informa al usuario con un mensaje emergente.

The screenshot shows a web application window titled 'Smiley'. At the top, there is a navigation bar with the application name 'Smiley' and five buttons: 'Gestionar Mapas', 'Gestionar Reuniones', 'Simular Reunión', 'Ver Resultados', and 'Salir'. Below this, the main content area is divided into two sections. The first section, 'Modificar el mapa del sistema', contains a sidebar with buttons 'Mapa Nuevo', 'Modificar Mapa' (highlighted), 'Guardar Mapa', and 'Ver Mapa Cargado'. The main part of this section is titled 'Modificar Pueblo' and contains two input fields for 'ID:' and 'Nombre:', followed by an 'Actualizar Pueblo' button. The second section, 'Modificar Tramo', contains five input fields for 'ID:', 'Nombre:', 'Tiempo:', 'Kilometros:', and 'Coste del billete:', followed by an 'Actualizar Tramo' button.

**Atención:** En el caso que se intente hacer una modificación y no haya ningún mapa cargado en el sistema, se comunicará al usuario que no hay ningún mapa cargado actualmente en el sistema.

Además, si se introduce un identificador de elemento erróneo también se notifica al usuario.

### 1.1.3 GUARDAR MAPA

“Guardar Mapa” es una opción que tiene como finalidad permitir al usuario guardar el mapa con el que opera el sistema a un directorio, es decir, exportar el mapa del sistema.

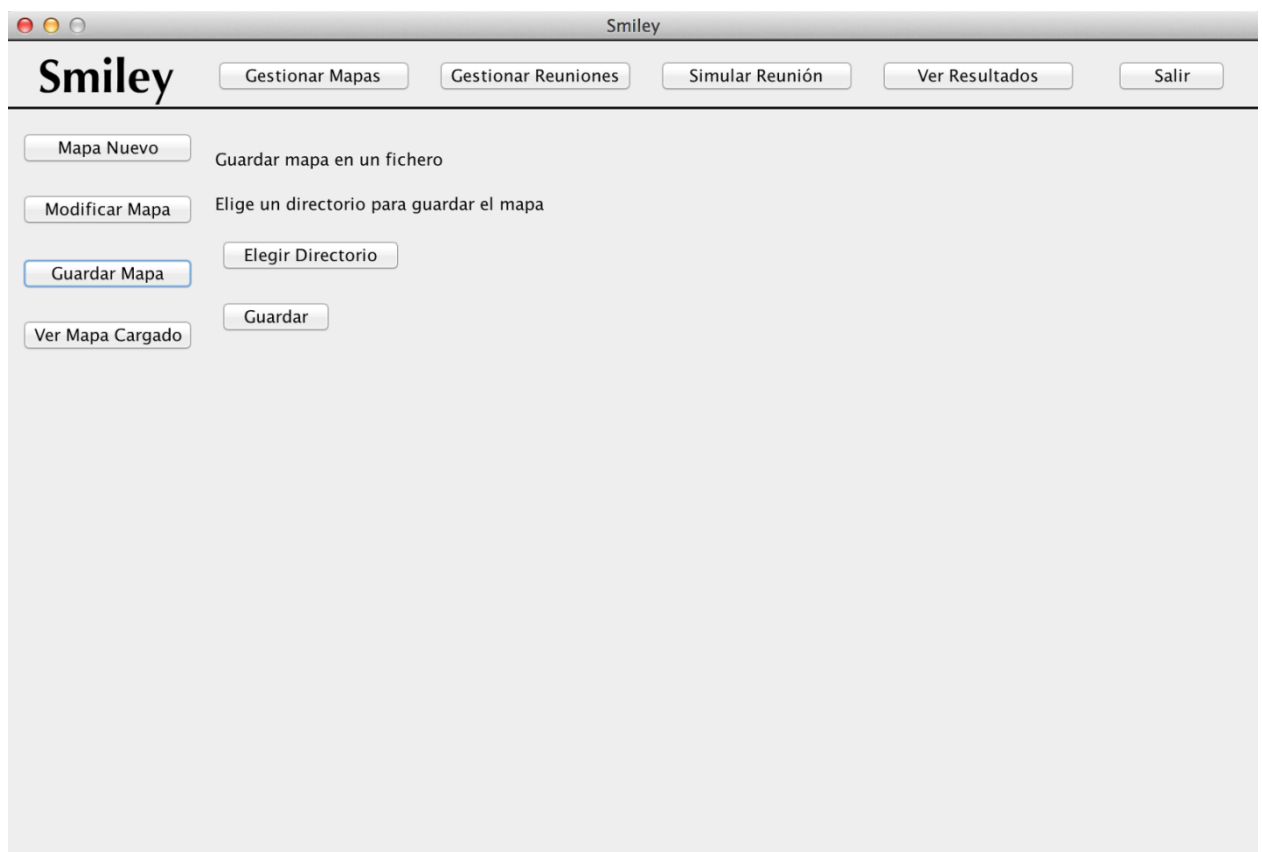
Primero el usuario debe de seleccionar el directorio a donde lo quiere exportar mediante el botón “Elegir Directorio” y para finalizar la exportación, el usuario debe de hacer clic en el botón “Guardar”.

En el directorio deseado se habrán guardado dos archivos. El primero representa los pueblos y el segundo representan los tramos.

Además, su nombre sigue la nomenclatura explicada anteriormente y que repetimos:

- Archivo de pueblos: *nombre\_del\_mapaV*
- Archivo de tramos: *nombre\_del\_mapaA*

El nombre de los archivos se denomina igual que el nombre del Mapa y, para terminar, el nombre del archivo que contiene los pueblos termina en V y el archivo que contiene los tramos termina con A.



#### 1.1.4 VER MAPA

La opción “Ver Mapa” nos muestra el mapa que el sistema contiene cargado en ese momento concreto. El usuario debe de tener en cuenta que el gestor sólo permite tener un mapa cargado a la vez.

Si en el momento que se hace la consulta “Ver Mapa” no hay ningún mapa, la aplicación no muestra nada.

Smiley

Gestionar Mapas Gestionar Reuniones Simular Reunión Ver Resultados Salir

Mapa Nuevo Mapa cargado en el sistema actualmente

Modificar Mapa Nombre del mapa: graph2

Guardar Mapa

Ver Mapa Cargado

Vértices (id:nombre):

- 2:d2
- 3:d3
- 4:t1
- 5:t2
- 6:t3
- 7:t4
- 8:t5
- 9:t6
- 10:t

Aristas (id:nombre:id pueblo origen:id pueblo destino:capacidad:flow:peso):

- 17:e17:2:6:1:0:0
- 18:e18:2:7:1:0:0
- 19:e19:2:8:1:0:0
- 20:e20:2:9:1:0:0
- 21:e21:3:4:1:0:0
- 22:e22:3:5:1:0:0
- 23:e23:3:6:1:0:0
- 24:e24:3:7:1:0:0
- 25:e25:3:8:1:0:0
- 26:e26:3:9:1:0:0

## 1.2 GESTIONAR REUNIÓN

La opción “Gestionar Reunión” es la que permite definir todos los parámetros de una reunión en el sistema. El usuario debe de tener en cuenta que para poder crear una reunión, debe de haber un mapa existente en la aplicación.

**Atención:** En el caso de querer crear una reunión sin la existencia de un mapa, se mostrará un mensaje por pantalla con la prohibición.

Para crear una reunión debemos de hacer clic en el botón “Crear una nueva reunión”. A continuación el sistema requiere que el usuario complete todos los parámetros de los que consta una reunión:

- Nombre de la reunión
- Número de agentes
- ID pueblo origen
- ID pueblo destino
- Criterio de obtención de rutas
- Algoritmo

The screenshot shows a web application window titled "Smiley". At the top, there is a navigation bar with five buttons: "Gestionar Mapas", "Gestionar Reuniones", "Simular Reunión", "Ver Resultados", and "Salir". The main content area is titled "Gestionar Reuniones" and contains the following fields and controls:


- Nombre de la reunión:** A text input field.
- Número de agentes:** A text input field.
- Pueblo origen (ID, nombre):** A text input field.
- Pueblo destino (ID, nombre):** A text input field.
- Criterio de obtención de rutas:** A section with three radio button options: "Tiempo", "Kilometros", and "Coste del Billete".
- Algoritmo:** A section with three radio button options: "Ford Fulkerson", "Edmonds Karp", and "Ford Fulkerson + Dijkstra".
- Crear una nueva reunión:** A blue button centered below the input fields.
- Hecho!:** A button located at the bottom right of the form.

Una vez definidos todos los parámetros, para finalizar con la gestión de la reunión, el usuario debe de hacer clic en el botón “¡Hecho!”. La opción de usar el algoritmo de Ford Fulkerson no

está disponible a causa de un problema con las clases compartidas que no nos ha permitido obtener una versión que nos funcione.

Si se quiere modificar cualquier parámetro de la reunión haciendo clic en crear una nueva reunión podemos cambiar todos los parámetros actuales o sólo los que nos interesan ya que se nos quedan guardados y no tenemos que volver a escribirlos todos.

Siempre se podrán consultar los parámetros de la reunión actual del sistema en la opción “Gestionar Reunión” del menú principal.



The screenshot shows a web application window titled "Smiley". The interface includes a header with the application name and five navigation buttons: "Gestionar Mapas", "Gestionar Reuniones", "Simular Reunión", "Ver Resultados", and "Salir". The main content area displays the current meeting parameters in a two-column layout:

Nombre de la reunión:	Reunión	Número de agentes:	40
Pueblo origen (ID, nombre):	0	Pueblo destino (ID, nombre):	2
Criterio de obtención de rutas:	tiempo	Algoritmo:	Ford Fulkerson + Dijkstra

At the bottom center of the main area is a button labeled "Crear una nueva reunión".



### 1.3 SIMULAR REUNIÓN

Una vez tenemos un mapa y una reunión definidos en el sistema mediante las funcionalidades “Gestionar Mapas” y “Gestionar Reuniones”, se puede realizar una simulación de la reunión. Para simular la reunión se usa la opción del menú principal “Simular Reunión”.

Una vez escogida la opción, se debe de pulsar el botón “¡Hacer la simulación!”. Esta siempre se hace sobre la última reunión definida.

Una vez lanzada la simulación hay dos posibles respuestas:

- Una reunión con solución
- Un reunión sin solución

Si la reunión se ha podido realizar con éxito, podremos consultar sus resultados en la opción del menú principal “Ver Resultados”

En cambio, si la reunión no se ha realizado con éxito, la aplicación proporciona tres soluciones:

- Cambiar el pueblo destino
- Hacer una subreunión
- Nada



Si el usuario escoge la opción “Cambiar la pueblo destino”, podrá volver a definir qué pueblo quiere como destino final de la reunión respetando los demás parámetros definidos anteriormente en la reunión. Una vez actualizado el destino se puede volver a hacer la simulación directamente.

The screenshot shows a web application window titled "Smiley". The header contains a navigation bar with the "Smiley" logo and five buttons: "Gestionar Mapas", "Gestionar Reuniones", "Simular Reunión", "Ver Resultados", and "Salir". Below the header, there is a status bar with two buttons: "Hacer la simulación!" and "Simulación hecha. Sin solución". The main content area displays a message: "La simulación no ha obtenido una ruta para cada agente. Hay algunos que no pueden llegar al pueblo de destino. ¿Qué desea hacer?". Below this message are three buttons: "Cambiar la ciudad de destino" (highlighted with a blue border), "Hacer una subreunión", and "Nada". Further down, there is a label "Nuevo destino:" followed by a text input field containing the number "5" and an "Actualizar destino" button.

Si el usuario escoge la opción “Hacer una subreunión”, podrá definir una nueva pueblo destino además de la que ya estaba definida, de esta manera, los agentes que no han podido llegar al destino inicial, quizás puedan llegar a la nueva pueblo dónde se realizará la subreunión. Cuando hemos actualizado el destino, si cuando volvemos a hacer la simulación sigue sin haber solución, saldrá el mensaje de que no hay solución y ya no se podrán escoger más alternativas.

The screenshot shows a window titled "Smiley" with a menu bar containing five buttons: "Gestionar Mapas", "Gestionar Reuniones", "Simular Reunión", "Ver Resultados", and "Salir". The main content area has a header with "Hacer la simulación!" and "Simulación hecha. Sin solución". Below this, a message states: "La simulación no ha obtenido una ruta para cada agente. Hay algunos que no pueden llegar al pueblo de destino. ¿Qué desea hacer?". There are three buttons: "Cambiar la ciudad de destino", "Hacer una subreunión", and "Nada". Further down, it says "Destino alternativo para la subreunión:" followed by a text input field containing the number "8" and an "Actualizar destino" button. At the bottom, a note reads: "Los agentes que no hayan podido llegar al destino principal intentaran llegar a este."

En el caso de que el usuario escoja la opción “Nada”, la reunión simplemente no se podrá realizar.

**Atención:** En algunas circunstancias el programa no simula la reunión. Concretamente en dos contextos. El primero de ellos podría ser porque el usuario ya ha simulado la reunión y, por lo tanto, ya dispone de los resultados. El segundo porque no existe ninguna reunión definida anteriormente. Es necesario que exista la reunión previamente y a su vez, el mapa.

## 1.4 VER RESULTADOS

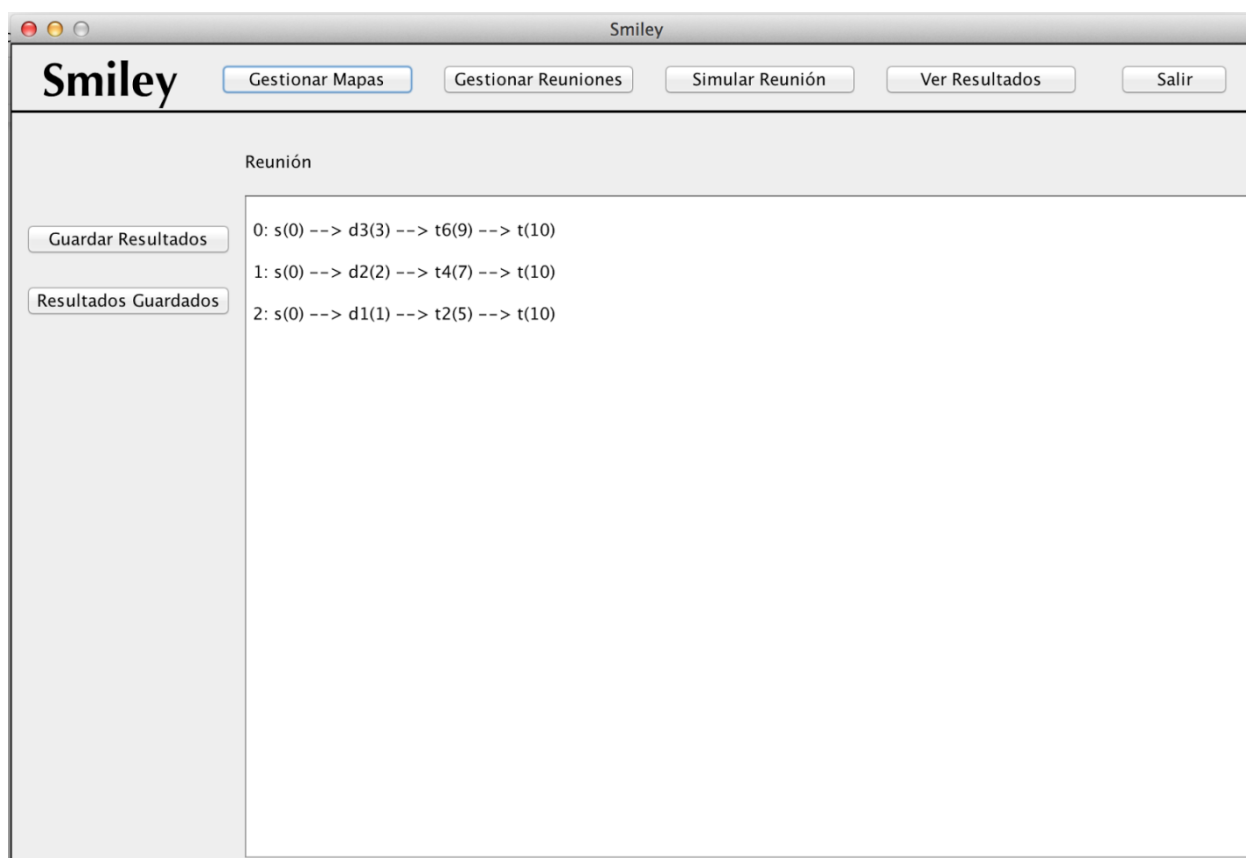
La opción “Ver Resultados” como bien dice su nombre, permite al usuario consultar los resultados de la última reunión realizada.

Además, en su interior contiene un submenú con las siguientes opciones:

- Guardar Resultados
- Resultados Guardados

Si la simulación no ha obtenido resultados, no saldrán resultados y la opción de “Guardar Resultados” no estará disponible.

El formato de los resultados es el siguiente. Por cada línea se muestra en primer lugar el identificador del agente seguido de los nombres de los pueblos y su id entre paréntesis por los que va pasando hasta llegar al final.



### 1.4.1 GUARDAR RESULTADOS

La opción “Guardar resultados” no permite guardar en los directorios del usuario los resultados de la última reunión simulada.

Estos resultados se guardan en un archivo, el cual su nombre sigue la siguiente nomenclatura: *Nombre de la Reunión*R.

---

#### 1.4.2 RESULTADOS GUARDADOS

La opción “Resultados Guardados” permite mostrar mediante la interfaz de la aplicación los resultados de reuniones que el usuario previamente tiene guardados en sus directorios.

#### 1.5 SALIR

Por último, para poder abandonar la aplicación, el usuario debe de hacer clic en la opción del menú principal “Salir”. Mediante este botón, la ejecución del sistema de gestión de reuniones Smiley finaliza.

#### MODIFICACIÓN DE LOS JUEGOS DE PRUEBA

Ha sido necesario modificar los juegos de prueba cambiando las capacidades a 1. Debido a que, en nuestro problema, solo puede pasar un soldado por cada tramo que hay en el mapa. Por tanto, el Max flow resultante es diferente al propuesto en la solución inicial.

- GRAFO 1  
Max flow = 2
- GRAFO 2  
Max flow = 3
- GRAFO 3  
Max flow = 5
- GRAFO 4  
Max flow = 6

Para hacer las pruebas que se crean pertinentes hemos proporcionado una carpeta llamada JP con 6 juegos de pruebas. Los 4 proporcionados por Horacio (graph1, graph2, graph3, graph4) y 2 más que son los mismos que proporcionamos en la segunda entrega (mapa1 y mapa2). Recordamos que cada juego de pruebas se compone de 2 archivos. Uno de pueblos (terminado en V) y otro de tramos (terminado en A). Y los dos son necesarios a la hora de importar los mapas.

## 2. DIAGRAMAS DE CLASES

### 2.1 DOMINIO Y CONTROLADORES DEL DOMINIO

El diagrama de clases se encuentra en la carpeta DiagramasUML de la entrega del proyecto por las dimensiones.

### 2.2 VISTAS Y CONTROLADORES DE PRESENTACIÓN

El diagrama de clases se encuentra en la carpeta DiagramasUML de la entrega del proyecto por las dimensiones.

### 2.3 GESTORES DE DISCO

El diagrama de clases se encuentra en la carpeta DiagramasUML de la entrega del proyecto por las dimensiones.

### 3. DESCRIPCIÓN DE LAS ESTRUCTURAS Y ALGORITMOS UTILIZADOS PARA IMPLEMENTAR LAS FUNCIONALIDADES PRINCIPALES

#### 3.1 FUNCIONALIDADES PRINCIPALES

##### Gestión de mapas

##### Definir reunión

##### Hacer reunión

##### Resultados

#### 3.2 GESTIÓN DE MAPAS

Un mapa puede tener 2 estados:

- **Cuando el mapa está cargado al sistema:**

En este estado el mapa queda guardado en 2 estructuras de datos. Un grafo y un conjunto de tramos. Tanto tramos como aristas tenga el grafo. El conjunto de tramos se representa con un `HashMap<String,Tramo>` donde el string es el id de la arista y del tramo para hacer el matching de arista-tramo más rápido.

Un tramo guarda el id de la carretera/tren, su nombre y sus tres pesos. Esto es así porque el grafo solo acepta un peso y nosotros queremos trabajar con 3 (tiempo, kilómetros y coste del billete). Y tenemos una función para cambiar el peso de todas las aristas del grafo por otro peso de los tramos.

- **Cuando el mapa está guardado en un fichero:**

En este estado el mapa queda guardado en 2 ficheros. Un fichero para almacenar los vértices. Este fichero empieza con el número de vértices en la primera línea. Y cada línea del fichero es un vértice, un id y un nombre. El fichero de las aristas es similar. Empieza con el número de aristas en la primera línea y luego cada línea tiene una arista con toda su información: el id de la arista, el nombre, el id del vértice origen, el id del vértice destino, la capacidad, el flow y los 3 pesos.

##### 3.2.1 TRAMOS

Un tramo se compone de los siguientes atributos: el id de la carretera o tren, el nombre, el tiempo que se tarda en recorrerla, los kilómetros o que tiene y el coste del billete. Si es una carretera, el coste de billete será 0 y si es un tren, será un valor diferente de 0.

### 3.3 DEFINIR LA REUNIÓN

Una reunión es una clase con los atributos necesarios para mantener la información de esta: un código (nombre de la reunión), el número de agentes que participan en ella, el id del pueblo origen, el id del pueblo destino y el algoritmo con el que se calcularan las rutas (Ford Fulkerson, Edmonds Karp o Ford Fulkerson + Dijkstra).

### 3.4 HACER REUNIÓN (ALGORITMO DE MAXIMUM FLOW)

Para llevar a cabo la reunión desde la clase Mapa llamamos al algoritmo de Maximum Flow pasándole el grafo, y los vértices origen y destino.

El algoritmo que calcula las rutas y las guarda en la clase Rutas donde se guardan los caminos de los diferentes agentes, usa las siguientes clases:

- **MaximumFlow:** Se encarga de ir obteniendo los caminos de menor peso del grafo con dijkstra o con el algoritmo que se seleccione y pasárselos a la GestioGraphResidual para que actualice el grafo con ese camino.
- **Dijkstra:** Dado un grafo obtiene el camino de menor peso.
- **ArcP:** Es una clase con dos atributos que identifican a una arista. El primer atributo es el peso y el segundo, el id de arista. Esta clase se utiliza en la priority queue del dijkstra para ordenar las aristas de menor a mayor peso. Creamos una clase aparte en vez de utilizar la clase arista porque es más manejable y nos evita tener que hacer cálculos innecesarios para obtener la información de la clase arista.
- **AristaVDestPeso:** Implementa una estructura de datos para guardar tres datos: el id de una arista, el vértice destino de esta arista y el peso que tiene. Se utiliza en el dijkstra para guardar la información de las aristas que salen del nodo actual (que se está analizando).
- **GestioGraphResidual:** Hace las modificaciones pertinentes al grafo para tapar el camino que ya se ha usado actualizando los flows.



---

#### 3.4.1 OBTENER LAS RUTAS POR LAS QUE PASAN LOS AGENTES

El algoritmo de Maximum Flow consta de 2 partes, la obtención de caminos del grafo mediante dijkstra o mediante el algoritmo que se seleccione en el momento de definir los parámetros de la reunión, y la gestión del grafo residual. Este proceso se repite o bien hasta que encontremos la solución o hasta que no haya más caminos.

Para calcular las rutas de los agentes, cada vez que obtenemos un camino lo guardamos como una ruta válida para un agente. Las rutas válidas las ponemos en un `HashMap<Integer, ArrayList<Integer>>` de la clase Rutas. La clave es el id del agente y el `ArrayList<Integer>` es la lista de los id's de los pueblos por los que pasa el agente (incluyendo el pueblo de salida y el de llegada). A partir del cálculo de cada ruta actúa la gestión del grafo residual incrementando el flow de las aristas de la ruta encontrada. El flow queda igualado a la capacidad y por lo tanto esta ruta ya no se tendrá en cuenta al volver a buscar otra ruta con el dijkstra. Este proceso se repite hasta que hayamos encontrado un camino para cada agente o hasta que no queden más caminos lo cual querría decir que no hay solución. En el caso que no haya solución, se informa al usuario para que decida que quiere hacer, dándole la opción de escoger otra ciudad de destino o de dividir la reunión en dos reuniones.

Si se escoge cambiar la ciudad de reunión se vuelve a hacer el mismo proceso pero con un destino diferente. En cambio, si se escoge hacer una subreunión, se conservan las rutas que se han obtenido y se hace otra reunión en otro destino pero los agentes que vayan hacia allí ya no podrán pasar por los tramos que ya hayan sido ocupados por los agentes que van a la ciudad original.

### 3.5 RESULTADOS

Para mostrar los resultados se recorren las Rutas y se va imprimiendo el id y la ruta de cada agente. Si se desea guardar el resultado en un archivo se guarda en forma de matriz donde cada fila representa un agente y cada columna los pueblos por los que pasa. Los resultados guardados siempre se pueden volver a consultar desde la vista de ver resultados.

#### 4. RELACIÓN DE LAS CLASES IMPLEMENTADAS POR CADA GRUPO DEL CLUSTER

CLASES	GRUP AUTOR	GRUPS CLIENTS
<b>Grafo</b>	Star Wars	Star Wars, Hospital i Smiley
<b>Vertice</b>	Star Wars	Star Wars, Hospital i Smiley
<b>Arista</b>	Star Wars	Star Wars, Hospital i Smiley
<b>Ford-Fulkerson</b>	Star Wars	Star Wars, Hospital i Smiley
<b>CtrlPersistència</b>	Smiley	Star Wars, Hospital i Smiley
<b>ImportarMatriz</b>	Smiley	Star Wars, Hospital i Smiley
<b>ExportarMatriz</b>	Smiley	Star Wars, Hospital i Smiley
<b>Dijkstra</b>	Smiley	Star Wars, Hospital i Smiley
<b>Edmonds Karp</b>	Hospital	Star Wars, Hospital i Smiley
<b>GestioGraphResidual</b>	Hospital	Star Wars, Hospital i Smiley

## 5. RELACIÓN DE LAS CLASES IMPLEMENTADAS POR CADA MIEMBRO DEL GRUPO

Clase implementada	Autor de la clase
Main.java	Joan

### 5.1 PRESENTACIÓN

Clase implementada	Autor de la clase
CtrlPresentacion.java	Flor, Joan, Laura y Víctor
VistaDrivers.java	Flor, Joan, Laura y Víctor
VistaBienvenida.java	Laura
VistaGestionMapas.java	Flor
VistaGestionReuniones.java	Joan
VistaGuardarMapa.java	Víctor
VistaImportarMapaFichero.java	Joan
VistaIntroducirParams.java	Víctor
VistaMapaNuevo.java	Joan
VistaModificarMapa.java	Laura
VistaPrincipal.java	Víctor
VistaResultados.java	Laura
VistaSimulacion.java	Flor
VistaVerMapa.java	Flor

## 5.2 DOMINIO

Clase implementada	Autor de la clase
ArcP.java	Laura
AristaVDestPeso.java	Joan
CtrlDomini.java	Flor y Joan
CtrlPersistencia.java	Joan y Víctor
Dijkstra.java	Joan y Laura
Mapa.java	Flor y Joan
Reunion.java	Laura
Rutas.java	Flor y Joan
Tramo.java	Flor

## 5.3 PERSISTENCIA

Clase implementada	Autor de la clase
ExportarMatriz.java	Víctor
ImportarMatriz.java	Víctor
CsvReader.java	Librería externa
CsvWriter.java	Librería externa

## 6. LISTADO DE FUNCIONALIDADES IMPLEMENTADAS Y RELACIÓN DE DIFERENCIAS ENTRE ENTREGAS

### 6.1 DIAGRAMA DE CLASES

A la hora de implementar el código, nos dimos cuenta que habían clases que se podían simplificar en una y evitarnos cálculos. Una de estas clases era la clase tramo, que aunque en el diseño la dividimos en las subclases Carretera y Tren, cuando lo implementamos vimos que salía más a cuenta eliminar las subclases. Si lo hubiésemos dividido en dos subclases, hubiésemos tenido que hacer funciones extra, por ejemplo, a la hora de calcular el algoritmo por el precio porque las Carreteras no hubiesen tenido ningún atributo de coste.

La clase Agente tampoco está presente en la versión final ya que nos dimos cuenta que podíamos cubrir su funcionalidad utilizando una única estructura de datos con la información de todos los agentes. Además, usar una estructura de datos permitía manejar los datos de manera más simple y reducida.

Por último, en la versión final tampoco existe la clase Pueblo ya que las funcionalidades que tenía que cubrir esta clase ya están cubiertas por la clase Nodo. Hicimos este cambio ya que decidimos compartir código para facilitar el trabajo del clúster.

### 6.2 CASOS DE USO

A la hora de implementar los casos de uso, nos encontramos con que era mejor hacer pequeñas variaciones.

Para empezar, en cuanto a la gestión de mapas, no hemos implementado un caso de uso específico para el borrado de un mapa sino que ofrecemos esta funcionalidad a través de la opción de mapa nuevo. Si el usuario decide crear un mapa nuevo y ya hay uno cargado, el actual se elimina para dar lugar al nuevo. Hemos decidido implementarlo de esta manera porque simplifica los pasos y permite un uso más directo por parte del usuario. En caso de que este tuviese dudas de la información cargada, hemos añadido un caso de uso para ver el mapa cargado. Debido a que la idea detrás de modificar mapa existente y crear un mapa nuevo es muy similar, decidimos eliminar la opción de crear un mapa existente basado en otro porque es una opción que ya se posibilita a través de la carga de un fichero existente y la modificación del mapa.

En cuanto a la gestión de reuniones, decidimos tomar un enfoque similar a la gestión de mapas y no ofrecer el borrado de una reunión sino que directamente dejamos que se modifique la que ya está cargada.

Por último, no ofrecemos la comparación de resultados sino que nos limitamos a ofrecer una consulta de los diferentes archivos guardados.