# Anexo B

# Manuales

## SimulatorFN

Para la ejecución del simulador y el resto de las herramientas en Java, es necesario disponer de esta plataforma instalada. Se recomienda utilizar la versión 11 para todas estas aplicaciones.

El simulador se distribuye comprimido en el archivo *simulator.zip*. Tras descomprimir el modelo, en la carpeta generada hay dos archivos principales: *start_model.bat* y *start_model.command*. En función del sistema operativo, Windows o Linux respectivamente, se puede ejecutar haciendo doble clic en el que corresponda. En caso de que no se ejecutase al hacer doble clic, se deberá configurar la ejecución de archivos de texto si no está habilitada o ejecutar desde la terminal.

Una vez ejecutado, se muestra el simulador en una ventana como en la figura B.1, donde se han marcado las opciones relevantes para la ejecución, que se explican a continuación.

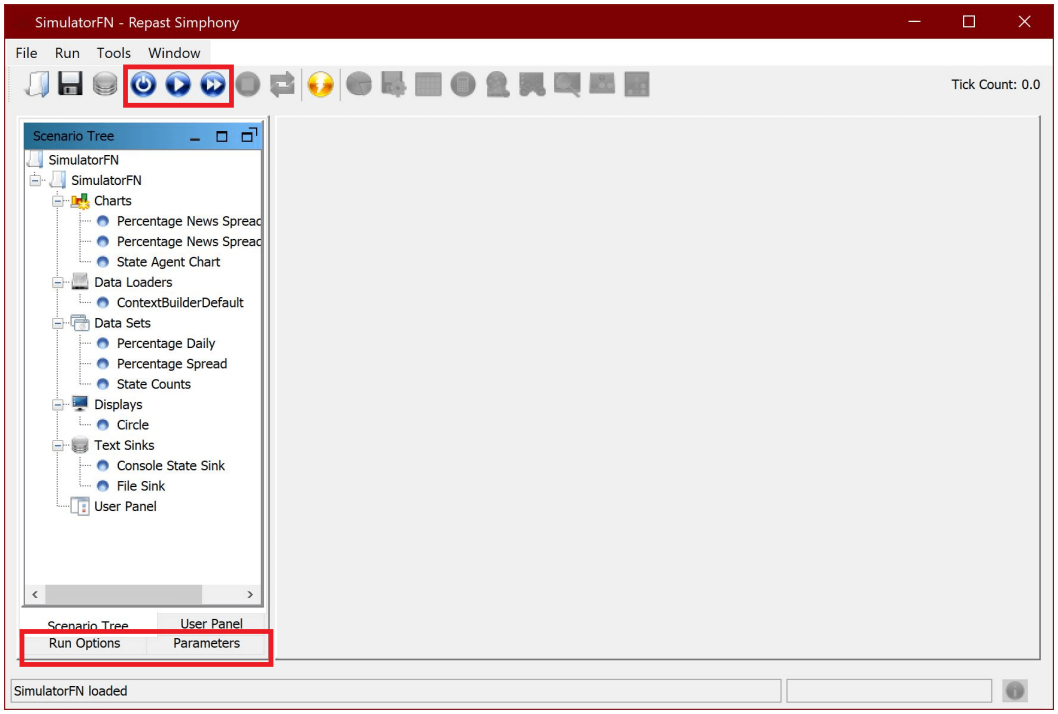


Figura B.1: Menú principal del simulador

La barra de herramientas principal, con los desplegables de *File*, *Run*, *Tools* y *Window*, ofrece la misma funcionalidad que los botones de la segunda barra que muestra el simulador. Para iniciar el modelo se van a utilizar los tres botones superiores marcados en rojo en la imagen B.1. El primer botón inicia la simulación, y se detiene en el instante inicial. El segundo botón inicia o continúa una ejecución pausada sin detenerse. El tercer botón permite avanzar cada instante temporal de forma manual.

En la parte inferior se han marcado las dos ventanas que permiten modificar la ejecución del modelo: *Run Options* y *Parameters*. Se muestran en la imagen B.2.

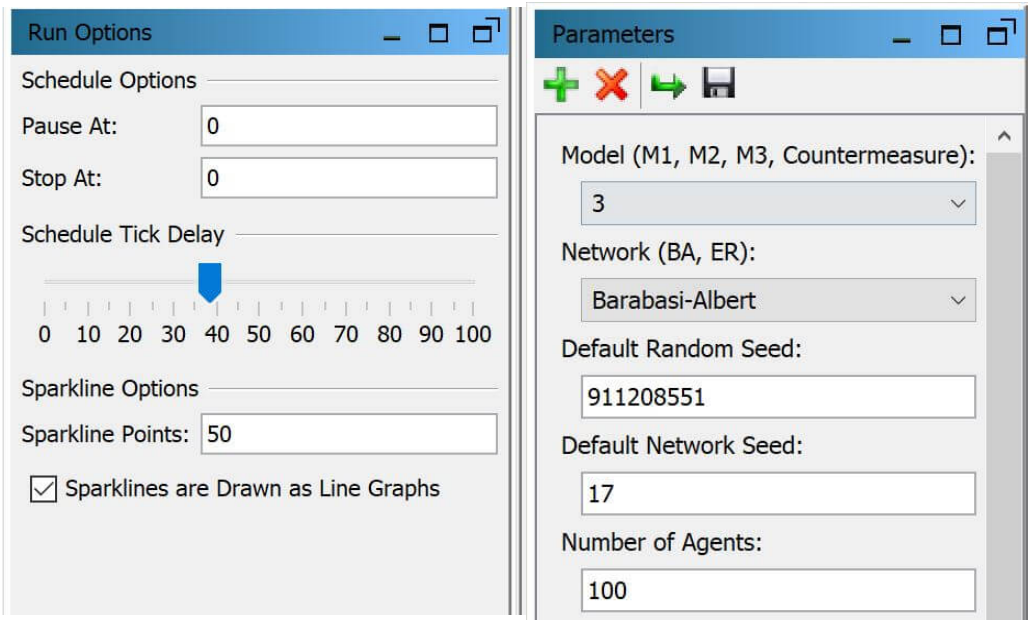


Figura B.2: Ventanas Run Options y Parameters

*Run Options* permite establecer valores para pausar o detener la simulación de forma definitiva. La opción *Schedule Tick Delay* permite modificar la velocidad en que avanza la simulación, si no se realiza de forma manual. Se recomienda establecer un valor entre 30 y 50 si se va a ejecutar de forma automática, para ver cómo evoluciona la simulación con más precisión.

En la ventana *Parameters* se muestran todos los parámetros que se pueden modificar en el modelo. Estos se listan a continuación:

1. *Model*. Se puede seleccionar entre los modelos 1-3 y el modelo 4, que se corresponde con la implementación de las contramedidas.
2. *Network*. Se selecciona el algoritmo de generación de la red, Barabási-Albert o Erdös-Rényi.
3. *Default Random Seed*. Semilla de generación de aleatorios de la simulación.
4. *Default Network Seed*. Semilla de generación de aleatorios para la construcción de la red.
5. *Number of Agents*. Número total de agentes.
6. *Number of Infected*. Número de agentes infectados en el instante inicial.
7. *Probability to Infect*. Probabilidad $P_{INF}$, en tanto por uno.

8. *Probability to Make Denier*. Probabilidad $P_{MD}$, en tanto por uno.
9. *Probability to Accept the Deny*. Probabilidad $P_{AD}$, en tanto por uno.
10. *Probability to Influence*. Probabilidad $P_{INFL}$ en tanto por uno, no aplica al primer modelo.
11. *Number of Bots*. Número de *bots*, sobre el total de agentes que se van a crear. No aplica al primer modelo.
12. *Number of Influencers*. Número de *influencers*, sobre el total de agentes que se van a crear. No aplica al primer modelo.
13. *Engagement of News*. Probabilidad *engagement* en tanto por uno. No aplica a los dos primeros modelos.
14. *Probability of Sharing*. Probabilidad de compartir una noticia que se ha marcado como falsa con una etiqueta de advertencia. Este parámetro no se puede modificar.
15. *Links per Node*. Número máximo de enlaces por cada nodo, para el algoritmo Barabási-Albert.
16. *Initial Nodes*. Número de nodos iniciales completamente conectados, para el algoritmo Barabási-Albert.
17. *Probability to Connect*. Probabilidad de conexión entre dos nodos cualesquiera, para el algoritmo de Erdös-Rényi.

Una vez se establecen los parámetros y comienza la ejecución, en el simulador aparecen la visualización de la red y los histogramas, como en la imagen B.3.

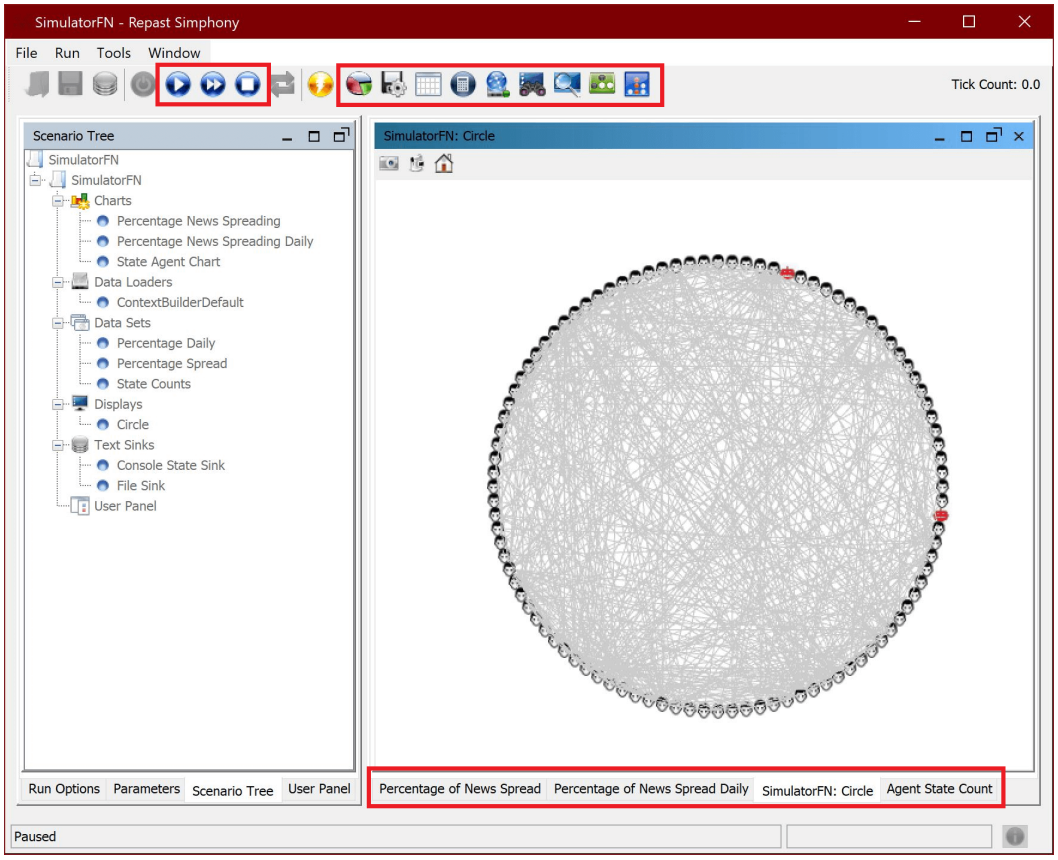


Figura B.3: Menú principal del simulador, tras comenzar la ejecución

Se puede observar que hay nuevos botones disponibles. En la parte superior izquierda marcada en rojo, se dispone de un nuevo botón que podrá detener la simulación. Una vez detenida, también se podrá reiniciar para cambiar los parámetros y realizar otra ejecución.

Los botones de la barra de herramientas superior, en la parte derecha marcada en la imagen, permiten exportar los datos de salida de los agentes a otras herramientas como una hoja de cálculo o WEKA para su análisis.

En la parte inferior aparecen cuatro vistas posibles:

1. *Percentage of News Spread*. Se puede visualizar un histograma con el porcentaje de usuarios que difunden la noticia en cada hora.
2. *Percentage of News Spread Daily*. Se presenta el histograma con el porcentaje de *endorsers* y *deniers*, en intervalos de 24 horas.
3. *SimulatorFN: Circle*. Es la imagen principal, que permite visualizar la red y los

usuarios.

4. *Agent State Count*. Se muestra un histograma con cada uno de los posibles estados de los agentes, por cada hora.

Una vez concluida la ejecución se puede ver la salida generada. Esta se ha almacenado en la carpeta *SimulatorFN*, en formato de texto y con el prefijo *ModelOutput*, seguido de la fecha de creación de cada una de ellas.

# Base de datos

Antes de poder ejecutar *ManagerSim* se debe configurar la base de datos.

Para la instalación será posible utilizar un contenedor Docker[1], para lo que se deberá disponer de la plataforma correspondiente según el sistema operativo. Se proporciona el fichero *docker-compose.yml* ya configurado. Si se altera alguno de los parámetros, también será necesario modificarlos en *ManagerSim*.

Para la carga de la base de datos se dispone de varias opciones:

1. Únicamente definir la base de datos con las tablas vacías. En este caso, en la carpeta *mysql/init/* se deberá mantener solo el archivo *init.sql*, que contiene la definición de las tablas.
2. Cargar la base de datos con los mejores resultados obtenidos para cada *dataset* estudiado. En este caso, no será necesario modificar nada.
3. Cargar la base de datos con todos los resultados obtenidos. En la carpeta *mysql/db_full/* se encuentran todos los resultados obtenidos. Se deberán sustituir los archivos en la carpeta *mysql/init/* por estos.

Es recomendable cargar la base de datos solo con los mejores resultados, ya que la carga completa puede suponer un tiempo de espera largo debido al alto número de ejecuciones que se han realizado durante la validación.

Una vez decididos los datos a cargar, se puede iniciar el contenedor. Para esto se ejecuta en la terminal desde la ruta donde se encuentra el fichero yml el comando: `docker compose up -d`. Es recomendable esperar uno o dos minutos hasta que se carguen las tablas, en el caso de utilizar la configuración recomendada, o comprobar el progreso de carga desde los logs del contenedor. Tras esto, la base de datos ya está

[1] `https://docs.docker.com/get-docker/`

lista para usar.

Una vez se haya terminado de usar, se puede detener con el comando: `docker stop mysqldb`, siendo `mysqldb` el nombre del contenedor por defecto. Se podrá volver a iniciar con: `docker start mysqldb`.

# ManagerSim

Esta herramienta se puede ejecutar directamente en línea de comandos mediante el archivo *jar* generado, o utilizar el archivo *bat* o *command* a tal efecto. Se disponen de dos opciones: la interfaz gráfica o consola de comandos. Para la interfaz gráfica, se puede hacer doble clic en el archivo anterior en función del sistema operativo. Si se decide emplear la consola de comandos, será necesario establecer las opciones a ejecutar de forma manual. Se dispone de las siguientes posibilidades al ejecutar:

–**h,** – –**help**. Muestra todas las opciones posibles.

–**c,** – –**config**. Selecciona un fichero de configuración distinto al establecido por defecto.

–**o,** – –**output**. Carga los ficheros de salida de una localización en la base de datos. Si no se añaden argumentos, únicamente carga un fichero. Si se añade el argumento *all*, incluye todos los ficheros.

–**d,** – –**dataset**. Carga un *dataset* en la base de datos. Necesita cuatro parámetros que se deben proporcionar en orden y separados por comas: el nombre del *dataset* que se almacena en la base de datos, la ruta del fichero del *dataset*, si el *dataset* se analiza de forma diaria u horaria (Y/N), o si el *dataset* está suavizado (Y/N).

–**m,** – –**metrics**. Calcula el RMSE de cada configuración y lo carga en la base de datos. Necesita tres argumentos separados por comas: el identificador del *dataset* a utilizar, la configuración inicial y la configuración final.

–**r,** – –**run_Batch**. Realiza las ejecuciones por lotes del simulador. Necesita un parámetro, para definir si se comienzan las ejecuciones desde el principio (Y) o se continúa donde está. Para poder realizar esta ejecución por lotes desde esta herramienta, es necesario instalar *Repast Simphony*.

–**g,** – –**graph**. Muestra el grafo para el mejor modelo seleccionado y *dataset*, que se deben pasar como argumentos separados por comas.

Antes de ejecutar este *jar*, se deben establecer ciertos parámetros en el fichero *db.properties*. Este archivo tiene los siguientes campos:

1. *URL_conn*, *user* y *pass*. Se debe establecer la dirección de la conexión de la base de datos, el usuario con el que conectarse y su contraseña, respectivamente. Si no se modifican los parámetros de la base de datos, estos valores se pueden mantener por defecto.
2. *prefix* y *suffix*. Estos valores son los que utiliza *Repast Simphony* por defecto, para la salida de las ejecuciones por lotes. Si se modifican en los parámetros del simulador, se deben modificar en este archivo para poder realizar la carga a la base de datos de forma correcta.
3. *output_folder* y *directory*. El primer valor, que solo aplica al ejecutar en línea de comandos, determina la localización de los archivos de salida del simulador, y el segundo especifica el directorio donde se almacenan estos ficheros una vez han sido cargados, que por defecto se corresponde con el directorio donde se encuentre el ejecutable.
4. *batch_params*. Este valor corresponde con un fichero XML que indica los parámetros que se modifican durante la ejecución por lotes. Se establecen el modelo y los valores para cada uno de los parámetros. Por defecto, se corresponde con el fichero *batchParams.xml* en la ruta de la herramienta.
5. *batch_file*, *batch_config*. Estos ficheros son necesarios para establecer los parámetros y la configuración de la ejecución por lotes del simulador desde la herramienta. El primero se corresponde con el fichero *batch_params.xml* en la carpeta *SimulatorFN/batch* dentro de *simulator*. El fichero de configuración se encuentra en la misma carpeta, bajo el nombre *batch_configuration.properties*. Este fichero es necesario editarlo manualmente para establecer las rutas de ejecución correctas, o generarlo de nuevo de forma automática. Para la segunda opción se debe ejecutar el simulador y acceder a *Configure Launch Batch Runs* dentro de *Tools* como en la figura B.4. Si se está ejecutando en Windows, será necesario definir las rutas con los caracteres de escape apropiados como los que figuran por defecto en este archivo.

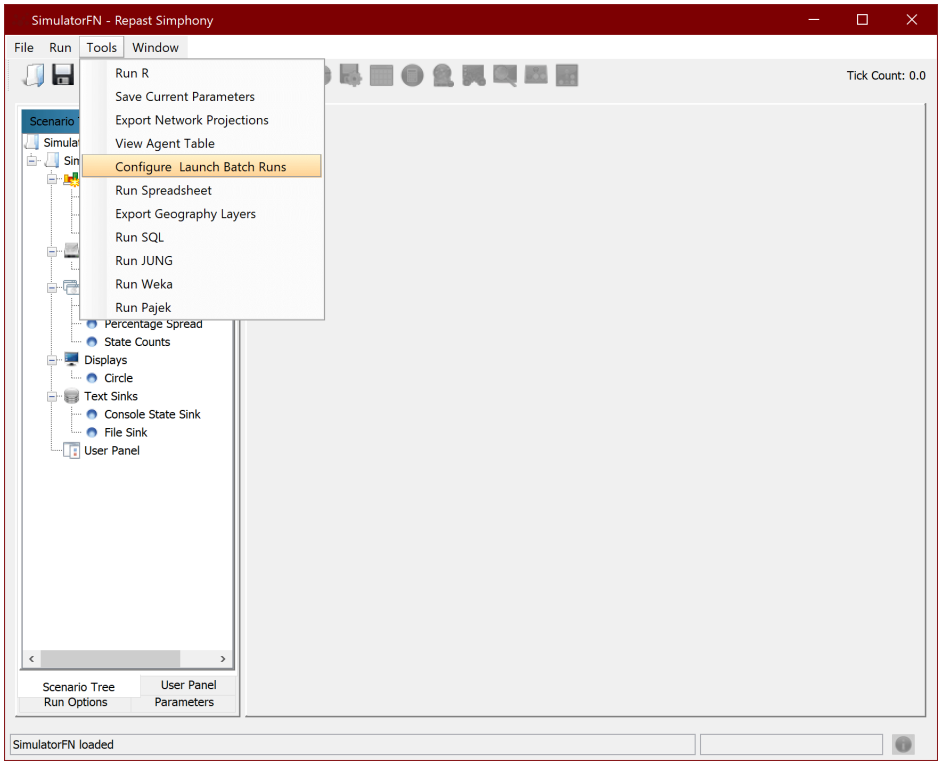


Figura B.4: Configuración del fichero *batch_configuration.properties* desde el simulador

Se muestra una ventana como B.5. Se selecciona la opción marcada en la imagen para sobrescribir el fichero de configuración con los parámetros correctos.

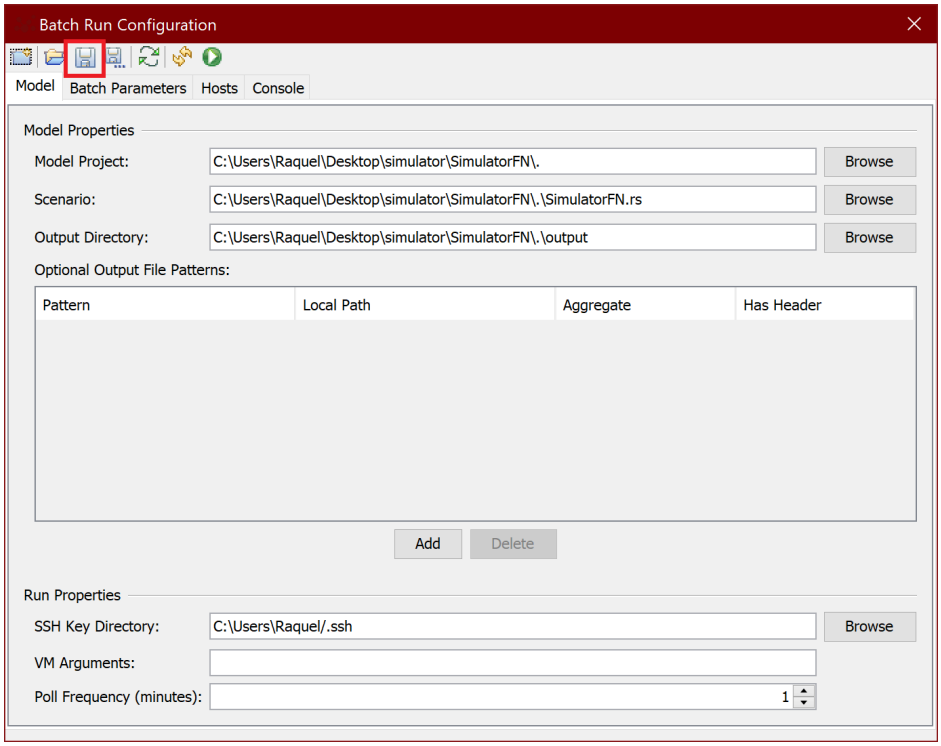


Figura B.5: Generación del fichero *batch_config*

6. *batch_runner*. Este valor se corresponde con el lanzador por lotes *batch_runner.jar* ubicado dentro de la carpeta de instalación de *Repast Simphony*. Para esto se debe instalar la versión 2.9.1 de *Repast Simphony*[2]. Únicamente será necesario instalar los componentes mínimos necesarios, como se muestra en B.6. Al seleccionar la carpeta de destino de la instalación, se debe tener en cuenta que se tiene que establecer en el fichero *db.properties*, para el parámetro *batch_runner*, la ubicación del lanzador *batch_runner.jar* dentro de esta carpeta.

[2] `https://repast.github.io/download.html`

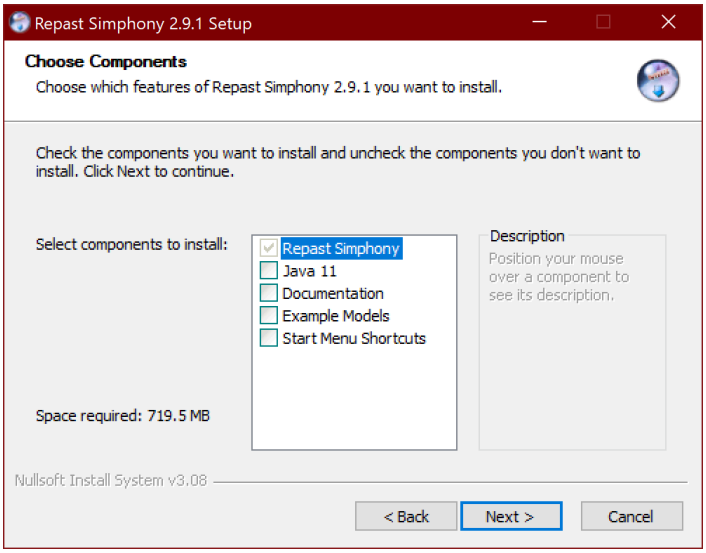


Figura B.6: Configuración de componentes de *Repast Simphony* en el instalador

El archivo *batch_runner* se distribuye a través del instalador de *Repast Simphony*. En el caso de que no se pueda utilizar el instalador y sea necesario instalar *Repast* en Eclipse directamente como en Linux, es necesario modificar ligeramente el tercer paso de la instalación para el correcto funcionamiento. Se deberá obtener Groovy Eclipse e4.20 desde el sitio: `https://dist.springsource.org/release/GRECLIPSE/4.4.0/e4.20`, a diferencia de la versión que figura en la página de *Repast Simphony*. Tras finalizar la instalación completa se tiene que ubicar el archivo *batch_runner.jar* para la versión 2.9.1, ubicado en la carpeta *SimulatorFN*, al mismo nivel que la carpeta de *eclipse*, que contiene el ejecutable de Eclipse.

El archivo XML de *batch_params* debe incluir todos los parámetros de los modelos, con el formato de B.7.

```
<parameters>
<parameter name="model">
    <value>2</value>
</parameter>
<parameter name="probInfect">
    <start>0.005</start>
    <end>0.005</end>
    <step>0.005</step>
</parameter>
<parameter name="probMakeDenier">
    <start>0.005</start>
    <end>0.005</end>
    <step>0.005</step>
</parameter>
<parameter name="probAcceptDeny">
    <start>0.005</start>
    <end>0.005</end>
    <step>0.005</step>
</parameter>
```

Figura B.7: Archivo XML para la ejecución por lotes

Con esta preconfiguración realizada, se ejecuta la herramienta con su interfaz gráfica. Se muestra el menú principal en B.8. Se dispone de 5 submenús que se detallan a continuación.

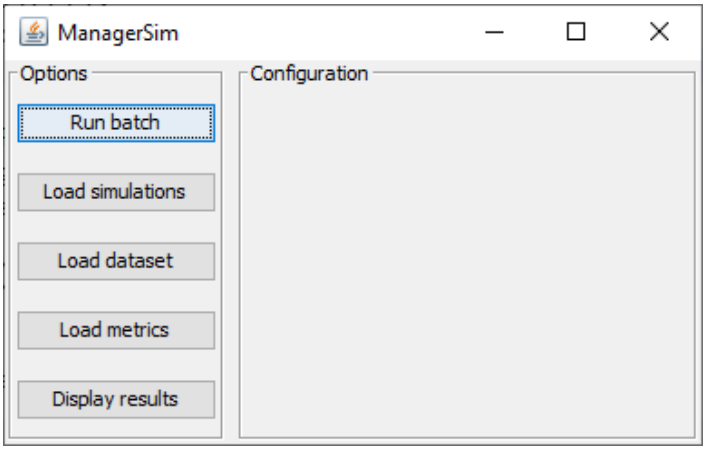


Figura B.8: Menú principal del simulador, tras comenzar la ejecución

1. *Run batch*. Se muestra en B.9. Se puede marcar la opción *Start over* para reiniciar las ejecuciones por lotes, según el XML ya establecido en el archivo de configuración. Tras esto, se inicia la ejecución por lotes con *Execute*. El tiempo de finalización dependerá del número de ejecuciones por cada configuración.

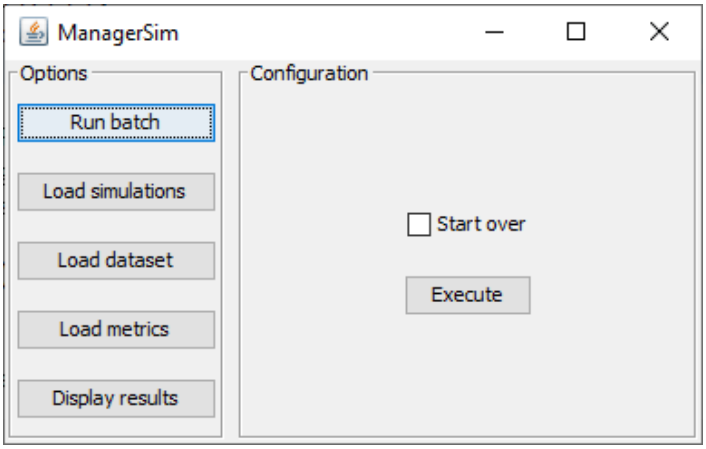


Figura B.9: Submenú *Run batch*

2. *Load simulations*. Se muestra en B.10. Se puede seleccionar la carpeta contenedora de las simulaciones en *Browse Input*, que se mostrará en *Browsed directory*. También se puede seleccionar si se incluyen todos los archivos marcando *Add all output* o sólo el más antiguo. Es necesario haber establecido la carpeta donde almacenar estos archivos previamente.

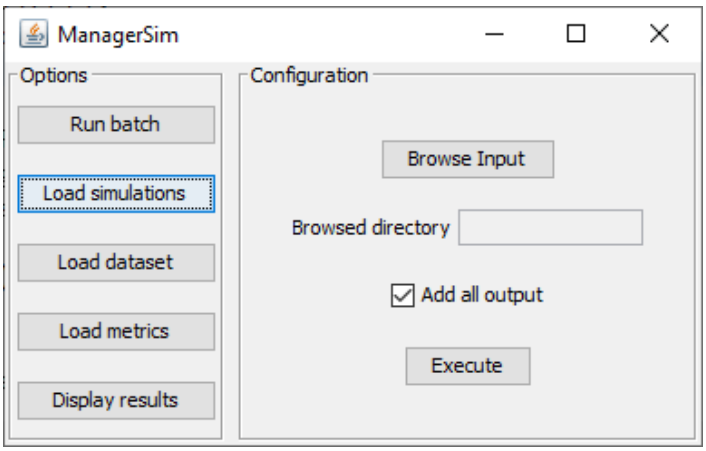


Figura B.10: Submenú *Load simulations*

3. *Load dataset*. Se muestra en B.11. Se selecciona el fichero a añadir a la base de datos con *Browse Dataset*, mostrándose en *Browsed file* una vez seleccionado. En *Name dataset* se escribe el nombre que usar a efectos de identificación en la base de datos, y dos características: si el *dataset* es diario u horario con *Is dataset daily* o si está suavizado con *Is dataset smooth*.

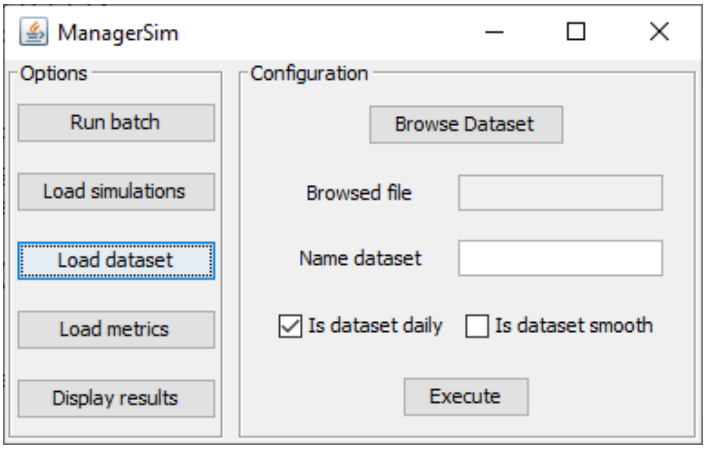


Figura B.11: Submenú *Load dataset*

4. *Load metrics*. Se muestra en B.12. Se selecciona del desplegable *Dataset* el *dataset* a utilizar para los cálculos, así como el rango de configuraciones para las que se calcula desde *Configuration from* hasta *Configuration to*.

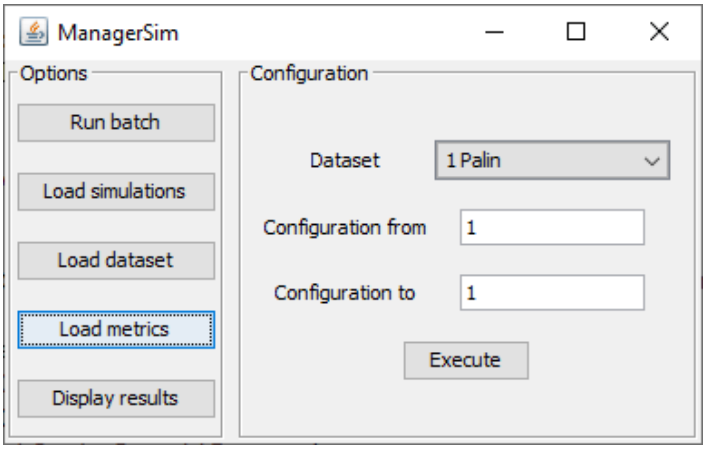


Figura B.12: Submenú *Load metrics*

5. *Display results*. Se muestra en B.13. Se selecciona el modelo a visualizar en el desplegable *Model* y el *dataset* en *Dataset*.

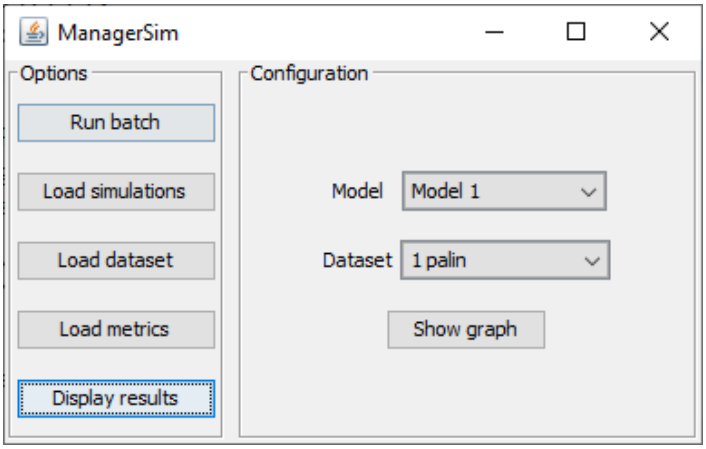


Figura B.13: Submenú *Display results*

Tras la ejecución de cualquiera de estas funciones, se bloquean los botones hasta que la operación haya terminado. En ese momento, se mostrará un mensaje en el log de la consola como en la imagen B.14. En el caso de operaciones más largas, también se imprime información de estado de forma periódica como se puede apreciar en la misma imagen, tras la carga de dos ficheros de simulaciones y la obtención de sus métricas.

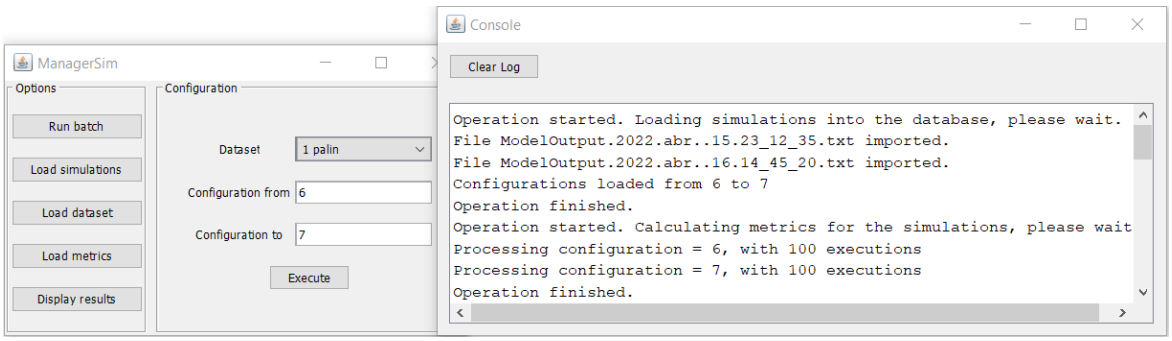


Figura B.14: Mensajes de estado de la consola de la herramienta

# Parser datasets

Para extraer los porcentajes de difusión de los *datasets* estudiados se ha utilizado una herramienta adicional, *parser.jar*. La ejecución se realiza con alguna de las siguientes combinaciones:

1. Para el *dataset* anonimizado y etiquetado de "Palin", se debe ejecutar con el siguiente comando: `java -jar -f ruta_dataset -l`, donde `ruta_dataset` es

la ubicación del dataset a parsear.

2. Para los *datasets* no etiquetados extraídos de Twitter, se debe ejecutar el comando: `java -jar -f ruta_dataset -t tiempo_parseo`. En este caso, como hay más de un *dataset* de Twitter, se proporciona el tiempo en minutos en que se analiza cada uno. Este fichero proporcionado debe ser el obtenido tras hidratar[3] los tweets, en formato JSON.

No se pueden distribuir los *datasets* extraídos de Twitter utilizados para parsear debido a las condiciones de uso de esta información, pero se proporcionan todos los *datasets* anonimizados y parseados para su consulta.

## Script de suavizado

Además del proceso inicial de parseado, para los *datasets* "Pope Coronavirus" y "Voter Fraud" se realiza un suavizado con un script en R[4]. Para la ejecución de este script se encuentra disponible el proyecto para importar y ejecutar directamente o, si no se dispone de la plataforma instalada, también se puede utilizar un contenedor Docker.

Para ejecutar el contenedor se proporciona el archivo *Dockerfile*. Primero será necesario construir la imagen con el comando en una terminal: `docker build -t script ruta_dockerfile` siendo `ruta_dockerfile` la ruta del fichero del mismo nombre.

Una vez termine el proceso de instalación de los paquetes necesarios, podemos iniciar el contenedor. Se ejecuta el comando:

```
docker run --rm -v ruta_output:/usr/local/src/script/output script
```

Se debe especificar la ruta completa en `ruta_output`, creando un volumen para poder consultar los archivos generados: los *datasets* suavizados y un archivo pdf con las gráficas obtenidas.

[3] `https://github.com/DocNow/hydrator`
[4] `https://www.r-project.org/`