Manual de Instalación

Pasos a seguir para instalar el Simulador de evacuaciones masivas DEMPS.

Proyecto Fondef ID15I10560 y ID15I20560
Plataforma de Apoyo a la Gestión de Emergencia y Aplicaciones

CITIAPS
Innovación en TI para Aplicaciones Sociales

UNIVERSIDAD DE SANTIAGO DE CHILE

Manual de instalación
Pasos a seguir para instalar el Simulador de evacuaciones masivas DEMPS.

Proyecto Fondef ID15I10560 y ID15I20560
Plataforma de Apoyo a la Gestión de Emergencia y Aplicaciones

Primera edición: marzo 2021.

Contacto
https://citiaps.usach.cl
citiaps@usach.cl

Tabla de contenidos

Tabla de contenidos **2**

Presentación | Resumen ejecutivo **3**

Estructura de directorios y archivos **3**

Requisitos mínimos **3**

Plataforma 3

Software base 3

Clonar el repositorio **4**

Instalación de dependencias **5**

JQ 5

OSMCTOOLS 5

Geographic lib 5

CGAL 5

OSRM backend 6

Compilación del simulador **7**

Comprobación del funcionamiento **7**

Ejecutar simulación de prueba 7

Comprobar la simulación de prueba 7

Visualización de la simulación de prueba 8

Presentación | Resumen ejecutivo

Este manual resume los pasos necesarios para la instalación del Simulador de evacuaciones masivas DEMPS, sus dependencias y comprobación de su correcto funcionamiento.

1. Estructura de directorios y archivos

A nivel global, el simulador se compone de dos secciones (Figura 1). La primera se relaciona con el código fuente y generación del ejecutable del simulador. Los detalles de este proceso se detallan en este documento. La segunda sección se relaciona con el uso del simulador y es el foco del documento "*Manual de Usuario*".

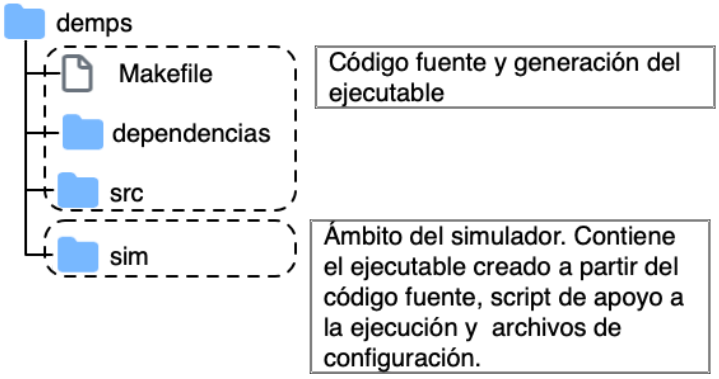

Figura 1

2. Requisitos mínimos

- ### Plataforma

SO - ubuntu 20.04 o mayor
1GB en memoria principal
100GB en almacenamiento secundario

- ### Software base

GIT versión 2.25 o superior
g++ 9.3.00 o superior
Make 4.2.1 o superior
Cmake 3.16.3 o superior

3. Clonar el repositorio

```
$ git clone https://github.com/citiaps/demps.git
```

El proyecto debe tener la siguiente estructura de directorios y archivos:

```
$ ls -l
total 24
-rw-r--r-- 1 user group 232  Oct  4 15:31 Makefile
drwxr-xr-x 3 user group 4096 Oct  4 15:31 dependencias
-rw-r--r-- 1 user group 2082 Oct  4 15:31 dependiencias.md
drwxr-xr-x 4 user group 4096 Oct  4 15:31 sim
drwxr-xr-x 5 user group 4096 Oct  4 15:31 src
```

La Figura 2 muestra la organización del repositorio que es de interés para la compilación del proyecto.

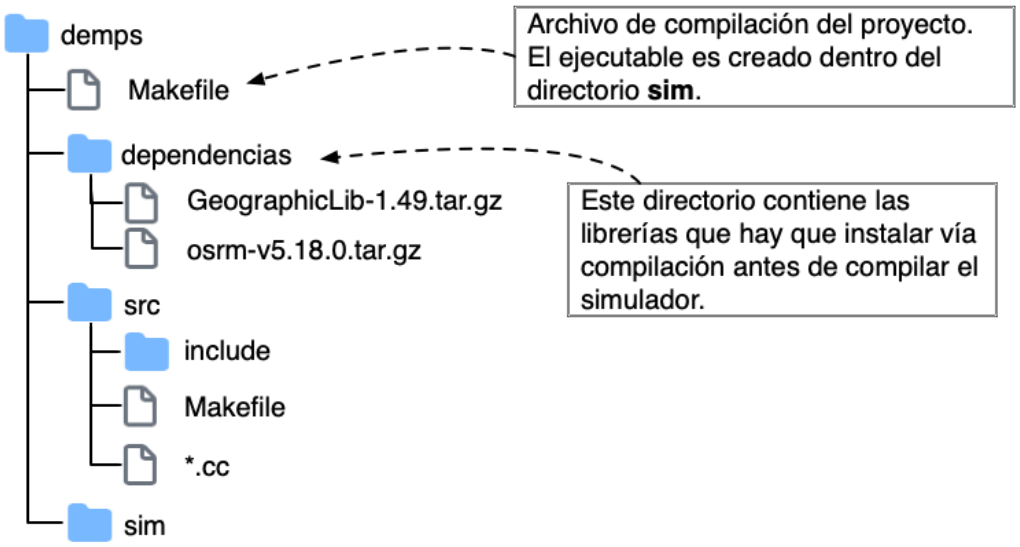

Figura 2

4. Instalación de dependencias

Antes de realizar la instalación se debe actualizar la lista de paquetes existentes y actualizar si corresponde.

```
$ sudo apt update && sudo apt upgrade
```

- ## JQ

Utilitario de línea de comando para procesar archivos JSON. Ver **https://stedolan.github.io/jq**.

```
$ sudo apt-get install jq
```

- ## OSMCTOOLS

Utilidades de línea de comandos para manipular archivos de OpenStreetMaps.

```
$ sudo apt-get install osmctools
```

- ## Geographic lib

Clases de C ++ para realizar conversiones entre coordenadas geográficas, UTM, UPS, MGRS, geocéntricas y cartesianas locales. Utilizar una versión igual o superior a la 1.49.

i. Bajar de **https://geographiclib.sourceforge.io**, o utilizar la versión disponible en el directorio **dependencias**, descomprimir y entrar al directorio respectivo.

```
$ tar xzf GeographicLib-X.XX.tar.gz
$ cd GeographicLib-X.XX
```

ii. Compilar e instalar la librería en el directorio por omisión /usr/local/:

```
$ mkdir build
$ cd build
$ ../configure
$ make
$ sudo make install
```

- ## CGAL

The Computational Geometry Algorithms Library, **https://www.cgal.org**. Utilizar una versión igual o superior a la 5.0.

```
$ sudo apt-get install libcgal-dev
```

- ## OSRM backend

Open Source Routing Machine: The OpenStreetMap Data Routing Engine, **http://project-osrm.org**.
Sitio original **https://github.com/Project-OSRM/osrm-backend/wiki/Building-OSRM**.

i. Dependencias: libtbb-dev, libbz2-dev, liblua5.3-dev, libexpat1-dev.

```
$ sudo apt-get install libtbb-dev, libbz2-dev, liblua5.3-dev,
libexpat1-dev
```

ii. Bajar de **https://github.com/Project-OSRM/osrm-backend/releases**, o utilizar la versión disponible en el directorio **dependencias**, descomprimir y entrar al directorio respectivo. Los mapas están procesados en base a la versión 5.18.0. Si instala una más reciente, debe procesar los mapas nuevamente a través del script **download.py** disponible en la carpeta sim/input/CiudadX.

```
$ wget https://github.com/Project-OSRM/osrm-backend/archive/v5.18.0.tar.gz
$ tar xzf v5.18.0.tar.gz
$ cd osrm-backend-5.18.0
```

iii. Compilar e instalar. Por omisión, --prefix=/usr/local

```
$ mkdir build
$ cd build
$ cmake .. -DCMAKE_BUILD_TYPE=Release  (!)
$ cmake --build .                      (!!)
$ sudo cmake --build . --target install
```

Observaciones:
!) En la versión 5.18.0 y superiores, la compilación en Ubuntu 20.04, arroja algunos errores que se solucionan con estos pasos:

- src/server/api/parameters_parser.cpp: línea 50, eliminar std::move().
- src/storage/io_config.cpp: línea 18, eliminar { y }.
- include/updates/csv_file_parser.hpp: Línea 150, eliminar std::move().

!!) Este proceso necesita al menos 3GB en RAM.

5. Compilación del simulador

El código fuente se encuentra en el directorio **src**. Para compilarlo y crear el ejecutable en el directorio **sim**, hay que ejecutar make en el directorio raíz del proyecto.

```
$ cd demps
$ make (!)
```

(!) Observación: si dispone de un sistema multicore, por ejemplo, con 4 cores, puede ejecutar **make -j4** para paralelizar el proceso de compilación.

6. Comprobación del funcionamiento

● Ejecutar simulación de prueba

El proceso de compilación, crea un el directorio **sim**. Para realizar una simulación de prueba en la ciudad de Iquique, utilizar el script **run.sh**.

```
$ cd sim
$ chmod +x run.sh (!)
$ ./run.sh -c iquique.config
```

(!) Observación: este comando se ejecuta por una única vez, para establecer el permiso de ejecución al script.

● Comprobar la simulación de prueba

El archivo de configuración **iquique.config** establece que el directorio donde se almacenan los archivos de salida de la simulación es "**output/iquique/**". Debe revisar si ese directorio existe y contiene los siguientes directorios y archivos.

```
$ ls -l output
drwxr-xr-x 5 user wheel  4096 Oct 28 11:13  iquique
$ ls -l output/iquique
drwxr-xr-x 2 user wheel  20480 Oct 28 11:13 agents
-rw-r--r-- 1 user wheel    289 Oct 28 11:13 animacion.config.json
-rw-r--r-- 1 user wheel   9538 Oct 28 11:13 animacion.html
drwxr-xr-x 2 user wheel   4096 Oct 28 11:13 input
drwxr-xr-x 2 user wheel   4096 Oct 28 11:13 stats
```

El archivo de configuración **iquique.config** también establece que el tiempo a simular son 3600 segundos y los reportes de posición de los agentes son cada 10 segundos. Estos se almacenan en el directorio agents. Se debe comprobar que existen esos archivos.

```
$ ls output/iquique/agents | sort
0000000000.txt
0000000010.txt
0000000020.txt
0000000030.txt
...
0000003570.txt
0000003580.txt
0000003590.txt
0000003600.txt
```

Adicionalmente, puede comprobar que el contenido de los archivos contiene datos geográficos.

```
$ head -3 output/iquique/agents/0000000000.txt
0 -20.20975336 -70.15134967 0 0
1 -20.22647930 -70.14592679 0 0
2 -20.27461671 -70.13011854 0 0
```

- ## Visualización de la simulación de prueba

Para visualizar la simulación realizada, basta con agregar la ruta completa del directorio de salida a un servidor web. Por ejemplo, para un servidor web Apache:

```
Alias /sim-test /ruta/al/directorio/
<Directory  /ruta/al/directorio>
        Order allow,deny
        Allow from all
        Require all granted
        AllowOverride all
        Options Indexes FollowSymLinks MultiViews
        IndexOptions FancyIndexing FoldersFirst NameWidth=*
</Directory>
```

Esto permitirá acceder a la animación de la simulación a través de:
http://dominio.tld/sim-test/animacion.html